ABSTRACT

HANSON, BRIAN BENNETT. The Economic Lot Scheduling Problem: Exact Solutions and System Feasibility. (Under the direction of Dr. Thom J. Hodgson and Dr. Russell E. King).

The Economic Lot Scheduling Problem considers the single-machine, multi-product inventory system. The objective is to determine a production schedule which minimizes long-run inventory and setup costs while avoiding stock outs. The problem is NP-hard (Hsu, 1983). This has led to a variety of simplifying assumptions and scheduling heuristics. First, this thesis demonstrates that exact solutions can be obtained under the basic period assumption using the benchmark Bomberger problem. Second, a generalization of the "power-of-two" method, called the "power-of-primes," is introduced. An algorithm, motivated by the methodology of part I, is developed which obtains exact solutions under both the power-of-two and power-of-prime assumptions. Solutions are shown to be superior to existing methods for a variety of real-world parameters. Third, an analytical method is presented which determines the set of all inventory positions from which it is possible to avoid a stock out over the infinite time horizon (the "feasible region"). Similar methods are presented which determine the maximum time a stock out can be avoided and the minimum time required to recover sufficient inventory to apply a desired cyclic schedule.

© Copyright 2013 Brian B. Hanson

All Rights Reserved

The Economic Lot Scheduling Problem: Exact Solutions and System Feasibility

by
Brian B. Hanson

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Operations Research

Raleigh, North Carolina

2014

APPROVED BY:

Dr. Thom J. Hodgson Committee Co-Chair	Dr. Russell E. King Committee Co-Chair
Dr. Michael G. Kay	Dr. Kristen A. Thoney-Barletta

BIOGRAPHY

Brian Bennett Hanson is a native of Traverse City, Michigan. He received a B.S. in Mathematics from Grand Valley State University (2006) and a M.S. in Mathematics from Miami University (2008). Brian enrolled in the Operations Research Ph.D. program at North Carolina State University in 2009 and received a Masters in Operations Research in 2011.

ACKNOWLEDGMENTS

I would like to acknowledge the guidance of the mathematics faculty of Northwestern Michigan Community College, Grand Valley State University, and Miami University who's dedication to teaching is responsible for my continued studies. Specifically, Dr. Stephen Drake, NMC, who inspired me to pursue mathematics, Dr. Paul Fishback, GVSU, who encouraged me to continue my studies in graduate school, and my Master's advisor Dr. Zevi Miller, MU, who first guided me through the research process. I would also like to acknowledge Dr. Thom Hodgson for spending the last three years challenging my mathematical training until, I hope, I reached the point where I can "think like a mathematician" or "think like an engineer" with equal ability. Finally, I would like to acknowledge the rest of my thesis committee, Dr. Michael Kay, Dr. Russell King, and Dr. Kristen Thoney-Barletta, for their guidance and editing of the papers contained in this thesis.

TABLE OF CONTENTS

LIST OF TABLES vi
LIST OF FIGURES vii
Chapter 1 Introduction 1
Chapter 2 On the Lot Size Scheduling Problem, Part I: The Optimal Solution to the Bomberger data 3
1 Introduction 3
2 Literature Review 4
3 Exact Solution for Bomberger's Data 5
3.1 Bounds on Cost and Cycle Time 5
3.2 Bounds on the Basic Period 7
3.3 Bounds on the Multipliers 7
3.4 Partitioning the Basic Period into Intervals 7
3.5 A Search Method for Basic Period Intervals 8
3.5.1 Schedule Feasibility Conditions 9
3.5.2 Partial Example 11
3.6 Analysis of the Remaining Basic Period Intervals 11
4 Conclusion 13
CHAPTER 2 REFERENCES 14
CHAPTER 2 APPENDIX 16
Chapter 3 On the Lot Size Scheduling Problem, Part II: Determining Exact Solutions to the ELSP under the Power-of-Two and Power-of-Prime Assumptions 19
1 Introduction 19
2 Literature Review 21
3 Practical Motivation 22
4 Methodology 23
4.1 Upper and Lower Bounds on the Basic Period (T) 24
4.2 Determining all k with a (potentially) lower cost given $b \leq T \leq b + \delta$ 26
4.3 Determining the existence of a feasible production schedule given (k, T) 26
4.4 Mathematical Preliminaries 27
4.5 Determining τ_π (the maximum time required to setup and produce all products j where k_j is a power of π in any basic period 34
5 Experimentation 42
5.1 Bomberger Data 42
5.2 Experimental Design 43
5.3 Note on choice of the Experimental Design 44
5.4 Results 46
6 Conclusion 48
CHAPTER 3 REFERENCES 50
CHAPTER 3 APPENDIX 52

Chapter 4 On the Lot Size Scheduling Problem, Part III: Stock Out Prevention and System Feasibility 59
1 Introduction 59
2 Literature Review 60
3 Delaying a Stock Out 63
3.1 Preliminaries 64
3.2 Feasibility Condition on the Total Production Plus Setup Time 65
3.3 Procedure to Determine t Given T, f, and I 66
3.4 Analytical Alternative to Solve the SAP 69
3.5 Algorithm to Determine Maximum T given f and I 70
3.6 Search Procedure to Determine Maximum T given I 71
3.7 Heuristic to Determine Maximum T given I 71
4 Inventory Recovery 74
4.1 Method to Determine t given T, f, I, and U 74
4.2 Search to Determine f and t given T, I, and U 75
4.3 Determining the Minimum T Given I and U 75
5 The Feasible Region 76
6 Conclusion 82
CHAPTER 4 REFERENCES 83
CHAPTER 4 APPENDICES 86
Chapter 5 Conclusion 89
REFERENCES 90

LIST OF TABLES

Chapter 2

Table 1: Lower and Upper Bounds on Cycle Time....6
Table 2: Feasible Partial Schedules....12
Table 3: Bomberger's Data 17
Table 4: Re-Ordering of Products....17
Table 5: Possible values **k** for $23.6 \leq T \leq 23.7$ 17
Table 6: Interaction of "2 groups" and "3 groups" 18

Chapter 3

Table 1: Total Time by Period 29
Table 2: Partitions of 5 products into set of 3 and set of 2....35
Table 3: Partitions of 6 products into set of 4 and set of 2....36
Table 4: Partitions of 6 products into two sets of 3....37
Table 5: Partitions of 8 products into set of 3 and two sets of 2 38
Table 6: Comparison P2, PP, GA for Bomberger's Data....43
Table 7: Experimental Design....44
Table 8: Five Product Results 47
Table 9: Ten Product Results 47
Table 10: P2 and PP Optimal **k**'s for Bomberger's Problem....48
Table 11: Partitions of 7 products into set of 5 and set of 2....53
Table 12: Partitions of 7 products into set of 4 and set of 3....53
Table 13: Partitions of 8 products into set of 4 and two sets of 2 54
Table 14: Partitions of 8 products into two sets of 3 and set of 2 54
Table 15: Partitions of 8 products into set of 6 and set of 2....55
Table 16: Partitions of 8 products into set of 5 and set of 3....55
Table 17: Partition of 9 products into set of 5 and two sets of 2....56
Table 18: Partition of 9 products into set of 4, set of 3, and set of 2....56
Table 19: Partition of 8 products into three sets of 3 57
Table 20: $\eta_9 = 8$ using six basic periods 57

Chapter 4

Table 1: Data for Anderson's (1990) Example 69
Table 2: Computational Times for Heuristic (seconds) 74
Table 3: Percent of the Feasible Region....81
Table 4: Computational Times (Seconds)....87

LIST OF FIGURES

Chapter 2

Figure 1: Total Cost vs. Cycle Time of Product 1 6

Chapter 3

Figure 1: Total Cost vs. Cycle Time of Product 1 of the Bomberger Data..........................25
Figure 2: Illustration of z_1, z_2, z_3 for $[14|23|5]$..40
Figure 3: Adding products with $k_j = 9$ to an existing 3-period schedule............................40
Figure 4: Total Cost vs. Utilization..46

Chapter 4

Figure 1: Section 3 (Diagrammatically)...64
Figure 2: The Feasible Region 81

Chapter 1

Introduction

The Economic Lot Scheduling Problem (ELSP) considers a single-machine, multi-product inventory system. The machine is assumed to be capable of producing one product at a time, and requires a setup time before producing each product. A "production schedule" is required which indicates the order (production sequence) and quantity (lot size) in which to produce the products. The objective is to determine a cyclic (repeatable) production schedule which minimizes long-run inventory and setup costs.

In practice, variations in demand, machine disruptions, and other inconveniences of every day plant life prevent the cyclic production schedule from being implemented exactly as designed. As observed by Bomberger (1966), "repetitive schedules that call for the same quantity of an item to be made in each cycle cannot be strictly maintained in reality," but "it is often possible to adjust production to correct for such disturbances." Instead, the cyclic schedule is used as a "first attempt at a feasible schedule which can then be manipulated to meet the detailed requirements of the shop floor" (Anderson, 1990). Essentially, the production schedule acts as a "production plan" rather than a strict schedule.

Occasionally, a machine disruption or another event may be significant enough that the cyclic production schedule cannot easily be modified to meet demand. If inventory levels become critically low, the system is in danger of a stock out occurring. In practice, a stock out may prevent the rest of the production facility from operating (i.e. "put the line down"), and is considered an extremely adverse event (Hodgson, 1980). Therefore, if inventory levels are low, the principle concern of the manager is avoiding a stock out (not minimizing cost.

The ELSP was first formally posed by Rogers (1958) and has been the object of extensive study. The problem is NP-hard (Hsu, 1983), and various simplifying assumptions have been made to find economic cyclic schedules. The primary focus in the literature has been the development of various heuristics to find economic cyclic schedules. This thesis uses the classical Bomberger data to illustrate that it is possible to optimize over all possible production sequences and obtain the exact optimal solution under the basic period assumptions. Insights gained from this example lead to a generalization of the power-of-two method called the “power-of-primes.” An algorithm, motivated by the methodology of part I, is developed which obtains exact solutions to the general ELSP under the power-of-primes assumptions (and, consequently, the exact solutions under the power-of-two assumptions). Solutions are shown to be superior to existing methods for a variety of real-world parameters.

Finally, the situation where inventory levels are critically low is addressed using three analytical methods. One method determines the set of all inventory positions from which it is possible to avoid a stock out over the infinite time horizon. This set is called the “feasible region.” The remaining two methods address the cases where the initial inventory is in the feasible region, and the case where the initial inventory is not in the feasible region. For the first case, the minimum time required to obtain sufficient inventory to apply a desired cyclic schedule is found. In the second, the maximum time that a stock out can be avoided is determined.

The body of this paper consists of three papers. The first paper proves the solution to the Bomberger problem found by Doll and Whybark (1973) and Haessler and Hogue (1976) is optimal. The second paper develops the power-of-primes method. The third paper contains the three analytical methods to address the situation inventory levels are critically low.

Chapter 2

On the Lot Size Scheduling Problem, Part I: The Optimal Solution to the Bomberger data

Abstract: The Economic Lot Scheduling Problem (ELSP) is a classical scheduling problem with the objective of minimizing the long-run inventory and setup costs of a single machine, multi-product inventory system. A classical data set of ten "more or less typical metal stampings" was presented by Bomberger in 1966. The data set has been extensively studied by numerous authors including Doll and Whybark in 1973, and Haessler and Hogue in 1976. This note determines the exact solution to Bomberger's data under the basic period assumptions, proving that the solution found by Doll and Whybark, and Haessler and Hogue is optimal.

1 Introduction:

The Economic Lot Scheduling Problem (ELSP) consists of a single machine assigned to produce n products. The machine is only capable of producing a single product at a time and requires a "setup" before a product can be produced. The daily demand rate of product j is d_j, the daily production rate is p_j, and the setup time in days is s_j for $j = 1, \ldots n$. Each time the machine is setup for product j, a cost of a_j is incurred. The unit cost of product j, c_j, is the combined material and production cost. Additionally, there is a daily internal inventory cost rate of \$$i$ /\$ inventory/day that is proportional to the on-hand inventory. All parameters are assumed to be known constants and are independent of the order in which the products are produced. The task is to find a lot size-schedule which minimizes long run inventory and setup costs while satisfying demand (stock outs are not allowed).

The ELSP is NP-hard (Hsu, 1983) and no known method to obtain the exact solution exists. Bomberger (1966) introduced the "basic period" approach which considers a restricted version of the problem (also NP-hard, Hsu, (1983)). The restricted problem assumes that each production run of product j has the same lot size (equal lot sizes), and that each time product j begins production, the inventory of product j is zero (zero-switch rule). The combination of equal lot sizes and the zero-switch rule imply that the production runs of each product are evenly spaced (in the time domain). In order to produce n products on a single machine without a scheduling conflict, a production sequence is required. The "basic period” approach assumes that there is a fundamental cycle time referred to as the basic period (T). Each product is assumed to be produced every $k_j T$ time units, where k_j is a dimensionless integer. The basic period T and the k_j’s (sometimes referred to simply as “multipliers”) are variables and determine the lot size of each product.

A classical data set of ten “more or less typical metal stampings” was presented by Bomberger (1966). This data set has been extensively studied by numerous authors including Doll and Whybark (1973) and Haessler and Hogue (1976). The primary contribution of this note is to prove that the schedule provided by both Doll and Whybark (1973), and Haessler and Hogue (1976) is in fact the optimal solution for Bomberger’s data.

2 Literature Review:

A great deal of literature exists on the ELSP. Bomberger’s data has been used by a number of authors as a benchmark for the performance of their heuristics. This literature review is limited primarily to papers published on the ELSP prior to 1966 and those which use the Bomberger (1966) data.

The ELSP was formally posed by Rogers (1958). Rogers introduced the Independent Solution approach which determines optimal lot sizes for each product without regard to the fact that all the products must be produced on the same machine. Conflicts naturally arise when more than one product is scheduled to be produced at the same time. A heuristic is used to adjust lot sizes and startup times to resolve the conflicts.

The Common Cycle or Rotation Cycle approach was first introduced by Hanssmann (1962) and Elion (1962). Each product is assumed to be produced once during a fixed time period (T). The setup and inventory costs can then be minimized over T. Feasible solutions are easy to obtain for a rotation cycle, but in general are not optimal for the ELSP. Jones and Inman (1989) demonstrate that the rotation cycle is near optimal for a wide range of realistic problems and suggest the method be considered before attempting more complex procedures.

Bomberger (1966) introduced the Basic Period (BP) approach described in the Introduction. He makes the additional assumption that the BP is sufficient to setup and produce a lot of each product. This assumption has not typically been made by subsequent authors and is not made in this paper. His dynamic programming approach is shown to be effective for low to medium loads. Stankard and Gupta (1969) propose a "grouping" procedure. The products for which it would be economically advantageous to produce more frequently form group "A" and the other products are divided into groups "B", "C", and "D". The groups, rather than the individual products, are then scheduled. Hodgson (1970) uses a pseudo dynamic programming procedure to improve the optimization. Doll and Whybark (1973) developed an iterative procedure to determine the multipliers for the BP approach. Schweitzer and Silver (1983) demonstrate the need for a lower bound on the basic period to insure Doll and Whybark's solution is feasible. For additional references on the limitations and feasibility of BP solutions see Haessler and Hogue (1976) and Andres and Emmons (1976). Haessler and Hogue (1976) also introduce a "power-of-two" method which restricts the multipliers to a power of 2 (i.e. $k_j = 2^m$ for some m). A survey and the extended BP approach are provided by Elmaghraby (1978).

3 Exact Solution for Bomberger's Data:

The Bomberger (1966) data can be found in Table 3 in the Appendix. Without loss of generality, the products are re-numbered for the convenience of the reader in order to facilitate a depth first search (See Table 4 in the Appendix), which will become relevant later in the paper. In this section, bounds on cost, cycle time, the basic period, and on multipliers are presented. Next, the basic period is partitioned into intervals, and bounds on the multipliers are recomputed. For each interval, the remaining multipliers are searched. This eliminates multipliers whose lower bound on cost is greater than the cost of the best known solution, or do not satisfy the feasibility conditions (see Section 3.5.1). The remaining multipliers are then analyzed to find the optimal solution.

3.1 Bounds on Cost and Cycle Time:

The total cost rate of the best known solution is \$32.07/day. A well-known lower bound on the total cost is the sum of the costs assuming that each product is run using the Economic Lot Quantity (ELQ), Rogers (1958). The lower bound on the total cost for the Bomberger data is

$$\sum_{j=1}^{10} \sqrt{2 d_j a_j i c_j (1 - d_j/p_j)} = 31.62 \text{ (\$/day)}. \quad (1)$$

In order to determine the optimal solution, all schedules with a total cost less than \$32.07/day need to be considered. There is a total of 32.07-31.62 =\$0.45/day "slack" between the lower bound and the best known solution. In order for the total cost to be less than \$32.07, each product can cost no more than

$$0.45 + \sqrt{2d_j a_j i c_j (1 - d_j/p_j)}, \text{ for } j = 1, \ldots, 10. \tag{2}$$

Since the total cost of product j is a convex function of product j's cycle time, t_j, the range of cycle times in which the total cost is less than the amount computed in Equation 2 can be easily found. Thus, the upper bound on the cost of an individual product j can be used to obtain an upper bound, t_j^{max}, and lower bound, t_j^{min}, on the cycle time of each product j. For example, by applying Equation 2, product 1 can cost at most \$13.12/day. This implies that the cycle time of product 1 must be between 15 and 27 (see Figure 1 below). The upper and lower bounds on the cycle times computed in this manner can be found in Table 1. Note that t_j^{min} is rounded down and t_j^{max} is rounded up in the table to include the entire interval.

Table 1: Lower and Upper Bounds on Cycle Time

Product	1	2	3	4	5	6	7	8	9	10
t_j^{min}	15	42	7	31	18	15	7	41	119	24
t_j^{max}	27	89	49	78	84	93	210	275	351	1154

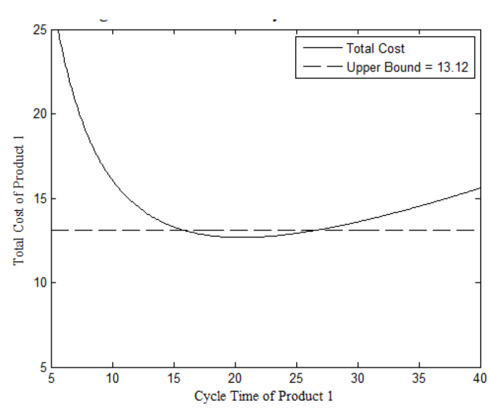

Figure 1: Total Cost vs. Cycle Time of Product 1

3.2 Bounds on the Basic Period:

In this section, an upper bound, T^{max}, and a lower bound, T^{min}, on the basic period are found. Because the cycle time of product j is $t_j = k_j T$, each product's cycle time is greater than or equal to the basic period ($t_j \geq T \; \forall j$). Therefore, the upper bounds on the t_j's are upper bounds on T ($t_j^{max} \geq t_j \geq T \; \forall j$). Thus an upper bound on T is

$$T^{max} = \min_j \{t_j^{max}\}. \tag{3}$$

For Bomberger's data, $T^{max} = 27$.

The lower bounds on the cycle times can be used to obtain a lower bound on the basic period since the basic period has to be long enough to setup and produce a product for its minimum cycle time. Thus, a lower bound on T is

$$T^{min} = \max_j \left\{ s_j + \frac{t_j^{min} \, d_j}{p_j} \right\} \tag{4}$$

For Bomberger's data, the largest minimum setup plus production time is 7.89 for product 2. Therefore, if the total cost rate is less than or equal to \$32.07/day, then $7.89 \leq T \leq 27$.

3.3 Bounds on the Multipliers:

Since the cycle time of product j is $t_j = k_j T$, bounds for t_j were found in Section 3.1, and bounds for T were found in Section 3.2, bounds for multiplier k_j can be easily computed. Let k_j^{min} and k_j^{max} denote lower and upper bounds on the multipliers of product j, respectively. Then

$$k_j^{min} = \left\lceil \frac{t_j^{min}}{T^{max}} \right\rceil \text{ and } k_j^{max} = \left\lfloor \frac{t_j^{max}}{T^{min}} \right\rfloor \tag{5a,5b}$$

However, this results in roughly 7.94×10^{11} possible combinations of multipliers, each of which may result in many more possible schedules.

3.4 Partitioning the Basic Period into Intervals:

In order to make the problem computationally feasible, the interval [7.89,27] is partitioned into subintervals of width 0.1 (with the exception of the first interval which is [7.89,8]). Given that the basic period is contained in the interval $[b, b + 0.1]$, it may be possible to improve the lower bound on the total cost rate. For example, if the basic period is contained in [8,8.1] then the only possible cycle times are $[8,8.1] \cup [16,16.2] \cup [24,24.3] \cup \cdots$. For example, the optimal cycle time of product 3 for the corresponding ELQ problem is 19.5283 (which is not possible if the basic period is between 8 and 8.1). Because the cost function is convex, the optimal cycle time of product 3 for the corresponding ELQ problem, given the basic period is between 8 and 8.1, is either 16.2 or 24 (whichever has the lower cost). In this case, the optimal cycle time is 16.2 and has a total cost rate of 1.0421. This increases the lower bound on the total cost rate for product 3 for the ELSP from 1.0244 to 1.0421. By applying this process to each product, a new lower bound for the total cost rate for the ELSP is found. For the interval [8,8.1], the new lower bound is 31.8027 compared to 31.6208. The improved lower bound on cost is used to update the bounds on the cycle time of each product as in Section 3.1. The updated bounds on the cycle times can be used to update the k_j^{min}'s and k_j^{max}'s as in Section 3.3, with $T^{min} = b$ and $T^{max} = b + 0.1$. This reduces the possible combinations of multipliers more significantly for some intervals than others. For example, the number of possible multipliers for the interval [8,8.1] is approximately 5×10^9 where as the number of possible multipliers for the interval [24,24.1] is 3,600.

3.5 A Search Method for Basic Period Intervals:

While the sub-problems are orders of magnitude smaller than the original problem, some subintervals are still too large to easily be searched to exhaustion. A search procedure is now developed to address these intervals. At this point, the order of the products becomes relevant to the efficiency of the procedure. The re-ordering described at the beginning of Section 3 results in the products being arranged in increasing order from the fewest possible multipliers to the most (with ties broken arbitrarily). A depth-first search is performed over the range of feasible multipliers for all intervals. To proceed, the multipliers for the products are selected one at a time (in order). When a multiplier $k_i = q$ is selected for product i, the lower bound on the total cost rate for that branch is updated by solving the corresponding ELQ problem for product i over the range $[qb, q(b + 0.1)]$ and adding it to the existing lower bounds on cost for all of the other products. In addition, schedule feasibility conditions(discussed in Section 3.5.1) are checked. If the total cost rate is greater than \$32.07/day or any of the feasibility conditions are not met, then the branch is fathomed. Otherwise, k_l^{min} and k_l^{max} are updated for the current branch for all products l where a multiplier has not yet been selected, and the branch continues to be searched. The procedure discussed in Sections 3.1 and 3.3 is used to update the bounds on the multipliers.

3.5.1 Schedule Feasibility Conditions:

In this section, the feasibility of a schedule over the interval $[b, b + 0.1]$ is first discussed. Then, expressions representing the actual and minimum amount of setup plus processing time incurred in every cycle of length $(b + 0.1)$ are defined individually for all products with multipliers 1 and 2, respectively. Next, expressions are presented representing the minimum amount of setup plus processing time incurred in at least one of the m groups of length $(b + 0.1)$ in which all products with multipliers of m are scheduled. Finally, these expressions are combined into necessary feasibility conditions for multipliers from 1 to 6.

In order for the vector of multipliers, **k**, to lead to a solution, there must exist a feasible schedule (i.e., a schedule where the time required to setup and produce all products scheduled during each basic period is at most T). If a schedule is infeasible with basic period V, then the schedule is infeasible for all V_0 where $V_0 < V$. Thus, if a schedule is infeasible for $(b + 0.1)$, it is infeasible for any $V \in [b, b + 0.1]$. Let

$$\tau_j = s_j + \frac{k_j(b + 0.1)d_j}{p_j} \tag{6}$$

be the sum of the setup time and the production time to satisfy demand during $k_j(b + 0.1)$ for product j.

Let f_m be a lower bound on the total time required to setup and produce all products j with multiplier $k_j = m$ in every cycle of length $(b + 0.1)$. Since all products with $m = 1$ are produced in every cycle, the amount of setup and processing time required for all products with $m = 1$ in every cycle of length $(b + 0.1)$ is

$$f_1 = \sum_{\text{all } j, k_j = 1} \tau_j. \tag{7}$$

Any schedule will divide the products with $m = 2$ into two groups. The combined setup and processing time in either group is at most $(b + 0.1)$. Therefore, a lower bound on the total time required to setup and produce all products with $k_j = 2$ in every cycle of length $(b + 0.1)$ is

$$f_2 = \max\left\{0, \sum_{\text{all } j, k_j=2} (\tau_j) - (b + 0.1)\right\} \quad (8)$$

Let g_m be a lower bound on the total time required to setup and produce all products j with multiplier $k_j = m \geq 2$ in at least one of the m groups in which they are scheduled. Clearly, the product with a multiplier of m with the largest total setup and production time will be in one of the groups. In addition, one of the groups must have at least $\frac{1}{m}$ of the total time. Therefore, let

$$g_m = \max\left\{\max_{\text{all } j, k_j=m} \tau_j, \frac{1}{m} \sum_{\text{all } j, k_j=m} \tau_j\right\}, \quad m \geq 2. \quad (9)$$

Combining the expressions defined in Equations 7 thru 9 yields the feasibility conditions in Equations 10a – d that are used in the search. Because 2 and 3 are relatively prime, each “2 group” will eventually be scheduled during the same cycle as each “3 group” (See “Note”, Appendix) (10a). If the multipliers are not relatively prime, then this may not be the case. For example, a “2 group” may never be scheduled in the same basic period as a “4 group”. However, 3 and 4 are relatively prime, so each “3 group” will eventually be scheduled during the same cycle as each “4 group” (10b). Each “2 group”, “3 group”, and “5 group” will eventually be scheduled during the same cycle (10c). In addition, each “3 group”, “5 group”, and “6 group” will eventually be scheduled during the same cycle, but not necessarily with a 2 “group” (10d).

Equations 10a thru d are conditions that are necessary but not sufficient for a schedule with multipliers **k** to be feasible with $V \in [b, b + 0.1]$.

$$f_1 + g_1 + g_3 \leq b + 0.1 \quad (10a)$$

$$f_1 + f_2 + g_3 + g_4 \leq b + 0.1 \quad (10b)$$

$$f_1 + g_2 + g_3 + g_5 \leq b + 0.1 \quad (10c)$$

$$f_1 + f_2 + g_3 + g_5 + g_6 \leq b + 0.1 \quad (10d)$$

Additional necessary conditions can be generated using larger multipliers. However, the computational cost of checking the condition has to be weighed against the computational benefit of potentially terminating a branch. The scarcity of multipliers greater than six in the Bomberger data made using multipliers less than or equal to six a logical choice.

***3.5.2 Partial Example*:**

The interval [23.6, 23.7] is used as an illustrative example. Initialize $f_m = g_m = 0 \; \forall \; m$. The only possible value for k_1 is 1. The minimum cost for product 1 given that $k_1 = 1$ and $23.6 \leq T \leq 23.7$ occurs at 23.6. The new lower bound on the cost under these assumptions is the sum of the total cost rate of product 1 with a cycle time of 23.6 and the existing lower bounds on the total cost rate of the other 9 products. In this case, this does not improve the lower bound on the total cost rate as $k_1 = 1$ obtains the previous lower bound for product 1. The value f_1 is updated to 6.698. Since the lower bound on the total cost, 31.89, is less than the value of the best known solution, 32.07, and none of the constraints (10a-d) are violated, product 2 is now considered. The possible values for k_2 are 2 and 3. Arbitrarily choose $k_2 = 2$ (The case where $k_2 = 3$ will be considered later in the search.). The updated lower bound on the total cost rate is 32.05. The value f_2 is updated to 0, and g_2 is updated to 8.808. Since the lower bound on the total cost is less than the value of the best known solution and none of the constraints (10a-d) are violated, the search continues. This process can be repeated until a multiplier has been selected for each product or the branch is fathomed. Using a depth-first search over all possible multipliers obtains all values of **k** with a cost less than \$32.07/day and $T \in [23.6, 23.7]$ (see Table 5 in the Appendix). This process reduces the 18,000 possible **k**'s for the interval [23.6, 23.7] to only 23 in less than 0.02 seconds on a 3 GHz machine.

The entire search overall intervals required 0.12 seconds on a 3 GHz machine coded in Matlab. Intervals containing at least one value of **k** range from $10.9 - 12.4$, $22.5 - 23.3$, $23.4 - 23.7$, and $23.9 - 24.0$.

3.6 Analysis of the Remaining Basic Period Intervals:

The intervals from $10.9 - 12.4$ can be eliminated using the following argument. In each case, $k_1 = 2, k_2 = 4$, and $k_3 = 2$. There is not sufficient time to setup and produce product 2 and either product 1 or product 3 in the same period. This forces products 1 and 3 to be produced in the same period. However, there is also not sufficient time to setup and produce products 1 and 3 in the same period. Therefore, none of the intervals between $10.9 - 12.4$ yield values for **k** which have a feasible schedule and a lower cost than \$32.07/day.

Next, the remaining intervals ($22.5 - 23.3$, $23.4 - 23.7$, and $23.9 - 24.0$) are considered. In each case, $k_1 = k_3 = 1$ and $k_2 = k_4 = k_5 = k_6 = k_7 = 2$. Therefore, products 1 and 3 will be setup and produced in every period, and products 2 and $4 - 7$ can either be setup and produced in periods 1,3,5,… or 2,4,6,…(i.e. the "odd" or "even" periods) . Without loss of generality, assume product 2 is setup and produced in the odd periods. There are 16 distinct ways to schedule products $4 - 7$. For illustrative purposes, the interval $[23.9, 24.0]$ is considered. Table 2 shows the 4 of the 16 ways that are feasible. Each of the three remaining products (8, 9, 10) need to be added to the schedule.

Table 2: Feasible Partial Schedules

	Odd Periods			Even Periods		
Grouping	Products Scheduled	Total Time	Remaining Time	Products Scheduled	Total Time	Remaining Time
I	1, 3, 2, 4	23.2624	0.7376	1, 3, 5, 6, 7	20.1644	3.8356
II	1, 3, 2, 6	23.3654	0.6346	1, 3, 4, 5, 7	20.0614	3.9386
III	1, 3, 2, 7	22.2501	1.7499	1, 3, 4, 5, 6	21.1767	2.8233
IV	1, 3, 2	20.8504	3.1496	1, 3-7	22.5764	1.4236

The smallest values for k_8, k_9, k_{10} in any **k** with a cost less than \$32.07/day are 4, 8, and 6, respectively (see Table 7 in the Appendix for the complete table). This implies that products 8,9, and 10 require at least $1.5247, 2.9120$, and 2.0370 days, respectively, to be setup and produced. Recall that the total time in any basic period is at most 24. This eliminates Grouping III as product 9 requires at least 2.9120 days to be setup and produced and the remaining time in the odd periods is at most 1.7499 and the remaining time in the even periods is at most 2.8233.

Groupings I, II, and IV have only one period which has sufficient remaining time to schedule any of the remaining products. If a multiplier is odd, then the product will be alternate between being produced in the odds periods and even periods. Therefore, $k_8, k_9,$ or k_{10} cannot be odd. This leaves $\mathbf{k} = [1\ 2\ 1\ 2\ 2\ 2\ 2\ 4\ 8\ 6]$ and $\mathbf{k} = [1\ 2\ 1\ 2\ 2\ 2\ 2\ 4\ 8\ 8]$. The first possibility for **k** requires that products 8 and 10 be produced in the same period. However, there is not sufficient time remaining in Groupings I, II, or IV to do this. The second possibility is the set of multipliers found by Doll and Whybark (1973) and Haessler and Hogue (1976).

The values vary slightly, but the logic is analogous for the remaining basic period intervals. In each case, all multipliers except $\mathbf{k}^* = [1\ 2\ 1\ 2\ 2\ 2\ 2\ 4\ 8\ 8]$ are eliminated. The value of T (23.42) found by Doll and Whybark, and Haessler and Hogue minimizes cost for $\mathbf{k}^*$. Therefore, the solution found by Doll and Whybark, and Haessler and Hogue is optimal.

4 Conclusion:

The solution to Bomberger's data found independently by Doll and Whybark (1973) and Haessler and Hogue (1976) has been shown to be optimal. The primary approach was to perform a depth-first search with bounds on the cost being updated and feasibility conditions checked after each step. Generalizing the procedure will be the subject of Part II.

CHAPTER 2 REFERENCES

Andres, F.M. and Emmons, H. (1976) On the optimal packaging frequency of products jointly replenished. *Management Science*, **22(**10), 1165-1166.

Bomberger, E. (1966) A dynamic programming approach to a lot size scheduling problem. *Management Science*, **12**(11), 778-784.

Doll, C. and Whybark, D. (1973) An iterative procedure for the single-machine multi-product lot scheduling problem. *Management Science*, **20**(1), 50-55.

Eilon, S. (1962) *Elements of Production Planning and Control*, Macmillan, New York, Chapter 10, 227-263.

Elmaghraby, S. (1978) The economic lot scheduling problem (ELSP): review and extensions. *Management Science*, **20**(1), 50-55.

Haessler, R. and Hogue, S. (1976) A note on the single-machine multi-product lot scheduling problem. *Management Science*, **22**(8), pp. 909-912.

Hanssmann, F. (1962) *Operations Research in Production and Inventory Control*, Wiley, New York.

Hodgson, T.J. (1970) Addendum to Stankard and Gupta's Note on Lot Size Scheduling. *Management Science*, **16**(7), 514-517.

Hsu, W.L. (1983) On the general feasibility test of scheduling lot sizes for several products on one machine. *Management Science*, **29**(1), pp. 93-105.

Jones, P. and Inman, R. (1989) When is the economic lot scheduling problem easy? *IEE Transactions*, **21**(1), 11-20.

Rogers, J. (1958) A computational approach to the economic lot scheduling problem. *Management Science*, 4(3), 264-291.

Schweitzer, P. and Silver, E. (1983) Mathematical pitfalls in the one machine multiproduct economic lot scheduling problem. *Operations Research*, **31(**2), 401-405

Stankard, M. and Gupta, S. (1969) A note on Bomberger's approach to lot size scheduling: heuristic proposed. *Management Science*, **15**(7), 449-552.

CHAPTER 2 APPENDIX

Table 3: Bomberger's Data*

Product	Setup Cost: *a* ($)	Prod. Cost: *c* ($/unit)	Prod. Rate: *p*(unit/day)	Demand: *d* (unit/day)	Setup Time: *s* (days)
1	130	5.9	1300	340	0.5
2	200	0.9	2000	340	0.75
3	10	0.1	7500	1600	0.125
4	110	2.785	2000	80	0.5
5	30	0.1275	9500	800	0.25
6	20	0.1775**	8000	400	0.125
7	5	0.04	15000	400	0.125
8	50	0.2675	6000	80	0.25
9	310	1.5	2400	24	1
10	15	0.0065	30000	400	0.125

*A single 8-hour shift is run 240 days per year and the inventory carrying cost is $i = \$\ 0.1/240$ /$ inventory/day.

**A data entry error in the Doll and Whybark (1973) paper altered the production cost rate of product 6 from 0.1175 ($/unit) to 0.1775 ($/unit). This does not affect the optimal schedule.

Table 4: Re-Ordering of Products

New Product Order	8	9	4	5	3	2	10	6	7	1

Table 5: Possible values **k** for $23.6 \leq T \leq 23.7$

$(k_1 = k_3 = 1, k_2 = k_4 = k_5 = k_6 = k_7 = 2)$

k_8	4	4	4	4	4	4	4	4	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5
k_9	8	8	8	8	9	9	9	9	9	9	8	8	8	8	8	8	9	9	9	9	9	9	9
k_{10}	6	7	8	9	5	6	7	8	9	10	5	6	7	8	9	10	5	6	7	8	9	10	11

NOTE: Two integers x and y are relatively prime if the greatest common divisor of x and y is 1. Let 2a, 2b be the two groups of products with $k_j = 2$ and 3a, 3b, 3c be the three groups of products with $k_j = 3$. Table 6 (below) illustrates how each “2 group” is eventually paired with each “3 group”.

Table 6. Interaction of “2 groups” and “3 groups”

Period	1	2	3	4	5	6
"2 Groups"	2a	2b	2a	2b	2a	2b
"3 Groups"	3a	3b	3c	3a	3b	3c

Chapter 3

On the Lot Size Scheduling Problem, Part II: Determining Exact Solutions to the ELSP under the Power-of-Two and Power-of-Primes Assumptions

Abstract: The Economic Lot Scheduling Problem (ELSP) is a classical scheduling problem with the objective of minimizing the long-run inventory and setup costs of a single machine, multi-product inventory system, subject to feasibility constraints. The problem is NP-hard (Hsu, 1983) and has been the subject of extensive study. The complexity of the problem has led to the use of various heuristics for restricted versions of the problem. Examples of classes of heuristics include basic period, power-of-two, and genetic algorithms. This paper introduces a new method, the "power-of-primes," and an algorithm which determines the optimal solution over all production schedules for both the power-of-two and power-of-prime methods. By optimizing over all production schedules, the results are guaranteed to be better than any heuristic under the same assumptions. The power-of-primes method is compared to Khouja et al.'s (1998) basic period genetic algorithm with favorable results. An experimental design is constructed which more accurately reflects problems commonly encountered in practice than Dobson's (1987) design.

1 Introduction:

The Economic Lot Scheduling problem (ELSP) consists of a single machine assigned to produce n products. The machine is only capable of producing a single product at a time and requires a "setup time" before a product can be produced. The demand rate for product j is d_j, the production rate p_j, and setup time s_j for $j = 1, \ldots n$. Each time the machine is setup for product j a cost of a_j is incurred. The unit cost of producing product j is c_j. Additionally, there is an internal inventory cost rate i proportional to the on-hand inventory. All parameters are assumed to be known constants and are independent of the order the products are produced. The task is to find a cyclic (repeatable) production schedule which minimizes long run inventory and setup costs while satisfying demand (stock outs are not allowed). In the first of this 3-part series, a procedure was developed which proves the optimality of the solution found by Haessler and Hogue (1976), and Doll and Whybark (1973) for the Bomberger (1966) data. The procedure is generalized in this paper.

The basic period approach assumes each production run of product j has the same lot size (equal lot sizes) and each time product j begins production the inventory of product j is zero (zero-switch rule). The combination of equal lot sizes and the zero-switch rule imply that the time between production runs of product j is a fixed constant. The basic period approach assumes product j is produced every k_jT units time where T is a fixed positive constant independent of j, and k_j is an integer. The variable T is referred to as the "basic period" and the k_j's are referred to as the multipliers. This implies the lot size of product j is d_jk_jT, the cycle time, t_j, is k_jT, and the total variable cost for all n products is

$$\sum_{j=1}^{n}\left(\frac{a_j}{k_jT}+ic_j\left(\frac{d_jk_jT}{2}\right)\left(1-\frac{d_j}{p_j}\right)\right). \tag{1}$$

For a given $\mathbf{k}=[k_1\cdots k_n]$ and T, there is no guarantee a production schedule exists where d_jk_jT product j is produced every k_jt time units for $j=1,\dots,n$. Determining if such a production schedule exists is NP-hard (Hsu, (1983)). If a production schedule does exist, then $(\mathbf{k},T)$ is said to be feasible.

The "power-of-two" method adds the constraint $k_j=2^{z_j}$ where z_j is a non-negative integer. The motivation for the restriction is to aid in finding a production schedule for a given $\mathbf{k}$ and T. The "power-of-primes" method assumes that k_j is a power of a prime number in a set R. More formally, $k_j=r_l^{m_j}$ where $r_l\in R$ and m_j is a non-negative integer. The power-of-primes method includes the powers of 2, and the powers of 3, 5, 7, etc. Thus, the values of $k_j<10$ available are 1, 2, 3, 4, 5, 7, 8, and 9. For most practical problems, this is sufficient for analysis. Although the power-of-primes is more general than the power-of-two, it has many of the same properties which aid in finding a feasible production schedule.

Section 2 provides a brief review of the relevant literature. Section 3 discusses the ELSP in practice. Section 4 develops the algorithm which finds optimal solutions for the ELSP under the power-of-two and power-of-prime assumptions. Section 5 includes experimental results and conclusions are presented in Section 6.

2 Literature Review:

The Economic Lot Scheduling Problem was first formally posed by Rogers (1958). Rogers determined the optimal cycle time for each product and developed a heuristic to adjust lot sizes and startup times to resolve any scheduling conflicts. Other early approaches include the rotation cycle approach (Hanssmann (1962), Elion (1962)) and various "grouping" procedures (Stankard and Gupta (1969), Hodgson (1970)).

The basic period approach discussed in the introduction originated with Bomberger (1966). Bomberger provided a dynamic program which is effective for low to medium loads and constructed a lower bound for the total cost of a cyclic schedule. Doll and Whybark (1973) used an iterative method to determine the multipliers. Haessler and Hogue (1976) developed a power-of-two version of Doll and Whybark's work. Elmaghraby (1978) provides a review of other earlier works.

Several later developments relevant to this paper include Hsu (1983), Dobson (1987), Khouja et al. (1998), and Hanson et al. (2013). Hsu showed determining if a feasible production schedule exists for a fixed $(\boldsymbol{k}, T)$ is NP-hard (and, consequently, the ELSP is NP-hard). Dobson showed that every production sequence is feasible for some $T > 0$ provided $\sum_{i=1}^{n} d_i/p_i < 1$ where d_i is the demand rate for product i, p_i is the production rate for product i, and n is the number of products being produced on the machine. Note that $\sum_{i=1}^{n} d_i/p_i$ is the proportion of time the machine must be in production (not including setup time). Therefore, if $\sum_{i=1}^{n} d_i/p_i \geq 1$ the machine does not have the capacity to meet demand. Additionally, Dobson introduced an unequal lot sizes scheduling heuristic.

Khouja et al. (1998) develop a genetic algorithm approach under the basic period assumptions. For other genetic algorithms used to solve the ELSP see Raza and Akgunduz (2008).

Hanson, et al (2013) considered the classical ten product ELSP data set introduced by Bomberger (1966) under the basic period assumptions. A scalable depth-first search method was used to determine all sets of multipliers **k** which satisfy necessary feasibility conditions and have a cost less than the current best known solution (found independently by Doll and Whybark, 1973 and Haessler and Hogue, 1976). Ad hoc reasoning was used to eliminate the remaining choices of **k** and conclude the solution found by Doll and Whybak, 1973 and Haessler and Hogue, 1976 is optimal under the basic period assumptions. To the best of the authors' knowledge, this is the first time a problem of any significant complexity has been optimized over all production schedules.

The primary contribution of this paper is an algorithm which obtains exact solutions under the power-of-two and power-of-prime assumptions. The algorithm is novel in that it does not rely on a heurisitic to select the production schedule. Obtaining exact solutions over all possible production sequences under the power-of-two and power-of-primes assumptions is shown to obtain competitive solutions even in comparison to more relaxed assumptions such as the basic period genetic algorithm of Khouja et al. (1998) and time-varying lot size approaches (Dobson (1987), Raza and Akgunduz (2008), etc.) for parameters with values reflective of those seen in practice.

3 Practical Motivation:

In practice, the production schedule is really a production "plan" which is modified to adjust for slight fluctuations in demand, machine disruptions, approximation of parameters, etc. According to Bomberger (1966), "Naturally, repetitive schedules that call for the same quantity of an item to be made in each cycle cannot be strictly maintained in reality," but "it is often possible to adjust production to correct for such disturbances." The cyclic schedule "serves to guide the manager" and is a "workable first approximation." This observation is also made by Anderson (1990), who states "many companies that operate with production systems of this type (cyclic schedules), employ some kind of decision support system. In such systems, it is usually important to supply the user with a first attempt at a feasible schedule which can then be manipulated to meet the detailed requirements of the shop floor."

One appeal of the power-of-two, power-of-prime, and basic period methods is that the basic period structure can naturally result in a "simple" schedule. If the optimal value of the basic period is near adjusted slightly to correspond with the end of a shift, day, or week, then the "first approximation" is simply the products to be produced during that time period. This makes it immediately apparent to management what needs to be produced during the time period. The more complex unequal lot size schedules obtained by Dobson (1987) and genetic algorithms (Raza and Akgunduz, 2008, etc.) may require more consideration when making adjustments.

4 Methodology:

The following algorithm determines the minimum cost for the ELSP over all possible values of T and a set of power-of-prime multipliers. Typically, optimal lot sizes for products produced on the same machine have not been observed to deviate by more than a factor of 10. Throughout the remainder of the paper, the possible power-of-two multipliers will be 1, 2, 4, and 8, and power-of-prime multipliers will be 1, 2, 3, 4, 5, 7, 8, 9. Note that the power-of-primes method allows for nearly all multipliers less than 10 (the only exception being 6). The development is analogous for any choice of primes and multipliers.

The procedure to obtain the optimal solution for the power-of-two algorithm consists of four main steps. First, the minimum cost of the rotation schedule is calculated and used to determine an upper bound, T^{max}, and a lower bound, T^{min}, for the basic period, T. The range for T is partitioned into intervals of width at most δ to condition on T in the next step. The value of δ effects computational time, but all values of δ obtain the optimal solution. For this paper, $\delta = 1$ is chosen.

Second, T is assumed to be in the rightmost interval (i.e. $[T^{max} - \delta, T^{max}]$). All values of **k** with a potential lower cost than the current best solution (given T is contained in the rightmost interval) are found using Hanson, et al (2013). Third, we determine if each **k** has a feasible production with a lower cost than the current best solution. If a lower cost is found, the solution is updated. Fourth, steps two and three are repeated for each interval, from right to left, until the entire range for T has been searched.

1. Use the rotation (power-of-two) schedule to determine upper and lower bounds for T. Partition the range into sub-intervals of width at most $\delta > 0$.
2. Determine all **k** which satisfy necessary feasibility conditions and have a potential lower total cost than the current best solution in the right-most sub-interval from (2) (Hanson, et al (2013).

3. For each **k** found in step 3, determine if a feasible production schedule exists with a lower cost.
4. Repeat steps 2-3 for each interval

The algorithm terminates with the optimal solution when the final sub-interval has been considered. Sections 4.1, 4.2 summarize the relevant work from Hanson, et al (2013) necessary to perform steps 1 and 2, respectively. Section 4.3 provides an efficient method to determine if a feasible production schedule for a fixed $(\mathbf{k}, \mathbf{t})$ using a necessary and sufficient condition (Theorem 1), optimization results (Theorems 2,3,4), and "feasibility" tables.

The methodology is identical for power-of-primes, except that the initial solution is the optimal power-of-two solution (instead of the rotation schedule).

4.1 Upper and Lower Bounds on the Basic Period (T):

The initial feasible schedule to obtain the power-of-two schedule is the rotation schedule. The optimal value of T for the rotation schedule is the maximum of the global minimizer and the minimum time required for feasibility:

$$T = \max\left\{\sqrt{\frac{2\sum_{j=1}^{n} a_j}{i\sum_{j=1}^{n} c_j\left(1-\frac{d_j}{p_j}\right)d_j}}, \frac{\sum_{j=1}^{n} s_j}{1-\sum_{j=1}^{n}(d_j/p_j)}\right\}. \quad (2)$$

For the power-of-primes, the optimal power-of-two solution found by applying the algorithm is used as an initial solution.

Hanson, et al (2013) determine an upper bound, T^{max}, and a lower bound, T^{min}, for the basic period. First, upper and lower bounds for the cycle times, t_j, are found based on cost. The variable cost of product j is a convex function of product j's lot size. The unconstrained minimum variable cost of product j (below) is obtained using the well-known Economic Lot Quantity (ELQ):

$$\sqrt{2d_j a_j i c_j(1-d_j/p_j)}. \quad (3)$$

The sum of the ELQ cost's over the n products forms a bound on the total variable cost. Let M denote the total variable cost of the production schedule from Section 3.1. If the total variable cost is at most M, then the individual product cost is at most

$$\sqrt{2d_j a_j i c_j (1 - d_j/p_j)} + \left(M - \sum_{j=1}^{n} \sqrt{2d_j a_j i c_j \left(1 - \frac{d_j}{p_j}\right)} \right) \quad (4)$$

(See Hanson, et al (2013)). Note that this is a necessary but not sufficient condition. The bound on variable cost can be used to find bounds on the cycle time using the quadratic formula.

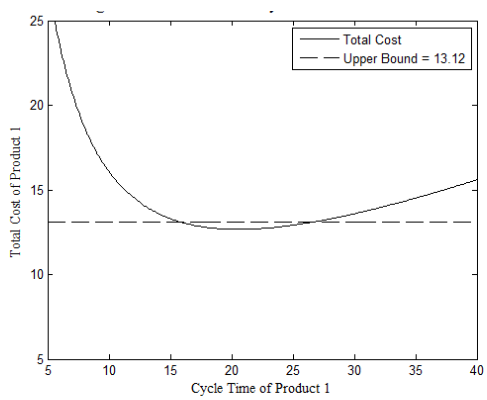

Figure 1: Total Cost vs. Cycle Time of Product 1 of the Bomberger Data

Let t_j^{max} and t_j^{min} denote the upper and lower bounds of t_j, respectively. The cycle time of each product is at least the basic period ($t_j = k_j T, k_j \geq 1$). Therefore, the basic period must be less than t_j^{max} for $j = 1, \ldots, n$. Additionally, the basic period must be long enough to setup and produce at least $d_j t_j^{min}$ product j for all j. Let T^{min} and T^{max} denote upper and lower bounds of T. Then

$$T^{min} = \max_j \left\{ s_j + \frac{t_j^{min} \, d_j}{p_j} \right\} \leq T \leq \min_j \{ t_j^{max} \} = T^{max}. \quad (5)$$

The interval $[T^{min}, T^{max}]$ is divided into intervals of width $\delta = 1$.

4.2 Determining all k with a (potentially) lower cost given $b \leq T \leq b + \delta$:

Consider the case where $T \in [b, b + \delta]$ for some fixed constant b. A method is required to determine all **k** which have a potential lower cost than the current best solution given $T \in [b, b + \delta]$. Recall the cycle time of product j, t_j, is $k_j T$. The bounds for t_j and $T$from the previous section can be combined to find upper and lower bounds for k_j. Let k_j^{min} and k_j^{max} denote upper and lower bounds for k_j.

$$k_j^{min} = \left\lceil \frac{t_j^{min}}{T^{max}} \right\rceil \text{ and } k_j^{max} = \left\lfloor \frac{t_j^{max}}{T^{min}} \right\rfloor \tag{6}$$

Hanson, et al (2013) re-index the products from the fewest possible multipliers to the most (the number of possible multipliers is $k_j^{max} - k_j^{min} + 1$). The depth-first search method conditions on k_1. The minimum variable cost of product 1 given $k_1 = l$ where $l \in k_1^{min}, \dots, k_1^{max}$ and $b \leq T \leq b + \delta$ is used to update the k_j^{min} and k_j^{max}. If necessary feasibility conditions are satisfied, the process conditions on k_2. This process is repeated until either a **k** is found which satisfies the necessary feasibility conditions and has a potential cost less than the current solution or it is determined that no such **k** exists on that branch.

4.3 Determining the existence of a feasible production schedule given $(\boldsymbol{k}, T)$:

After searching an interval, a set of **k**'s is obtained which may have a lower cost than the current best solution. The uncertainty is due to the fact that, when the costs of the products are minimized in Section 3, the basic period is allowed to be different values for different products. After the **k**'s are found, the interval constraint is dropped. The minimum cost is obtained for **k** is obtained by

$$T^* = \sqrt{\frac{2\sum_{j=1}^{n}\left(\frac{a_j}{k_j}\right)}{i\sum_{j=1}^{n} c_j \left(1 - \frac{d_j}{p_j}\right) d_j k_j}}. \tag{7}$$

The total cost is found for the maximum of T^* and Bomberger's lower bound for T to obtain the minimum cost of **k** which may have a feasible production schedule. Denote the maximum of T^* and Bomberger's lower bound by T_{lb}. If the cost is greater than the current best solution, then the value of **k** is disregarded and need not be reconsidered in future intervals.

If the cost is less, then a methodology is required to determine the minimum cost of a feasible production schedule for **k** over T. Subsections 4.3.1 and 4.3.2 develop a method to achieve this goal. Note that increasing the basic period of a feasible production schedule preserves feasibility (provided $\sum_{j=1}^{n} d_j/p_j < 1$). The total cost equation can be solved for the largest value of T which has a cost less than or equal to the current best solution. Denote this value of T by T_{ub}. Feasibility is tested for $T = T_{ub}$. If infeasible, then we conclude that no feasible production schedule with a cost less than the current solution exists for any value of T and **k** is disregarded. If a feasible production schedule is found, then a feasible production schedule exists with a lower cost. In this case, $T = T_{lb}$ does not have a feasible production schedule and $T = T_{ub}$ has a feasible production schedule. The smaller the value of T on $[T_{lb}, T_{ub}]$ the lower the total cost (as $T^* \leq T_{lb}$ and the total cost function is convex). Therefore, a one dimensional search over T on the interval $[T_{lb}, T_{ub}]$ to determine the minimum value of T which has a feasible production schedule will minimize total cost.

4.4 Mathematical Preliminaries:

If **k** and T are fixed, then the time required to setup and produce a lot of a product, j, is a fixed constant. Define $t_j = s_j + d_j k_j T/p_j$ to be the total setup and production time of product j given **k** and T. If the time required to setup and produce a product is greater than the basic period, then, clearly, there is no feasible production schedule. Without loss of generality, assume $t_j \leq T$ for all j for the remainder of the section.

Note that a cyclic production schedule has a fixed number of basic periods. For example, if $\mathbf{k} = [1\ 2\ 2]$ the production schedule $[1\ 2|1\ 3]$ has two basic periods ([1 2] and [1 3]). In Lemma 1, it is shown that every production schedule for a fixed **k** has the same number of basic periods.

Lemma 1: A cyclic production schedule with $\mathbf{k} = [k_1 \cdots k_n]$ has $b = lcm(k_1, \dots, k_n)$ basic periods, where $lcm(k_1, \dots, k_n)$ denotes the least common multiple of $k_1, \dots, k_n$.

Proof: Note the number of basic periods in a cyclic schedule is how often the schedule is repeated. Suppose that a cyclic schedule, θ, repeats every z basic periods. If the first production run of product j in θ occurs in period f_j, then product j is produced in periods $f_j + \lambda k_j, \lambda = 1,2,3$ .... The first production run of product j in the second iteration of the schedule occurs in period $z + f_j$. Therefore, $z + f_j = f_j + \lambda_j k_j$ for some integer λ_j for $j = 1, \dots, n$. Thus, $z = \lambda_1 k_1 = \lambda_2 k_2 = \cdots = \lambda_n k_n$ for some $\lambda_1, \dots, \lambda_n$ and z is a common multiple of $k_1, \dots, k_n$. Therefore, the first time the schedule is repeated is $lcm(k_1, \dots, k_n)$ and $\mathbf{k}$ has $lcm(k_1, \dots, k_n)$ basic periods. Q.E.D.

We now illustrate how the power-of-primes assumption aids in determining if a feasible schedule exists. Consider $\mathbf{k} = [2\ 2\ 3\ 3\ 4]$ as a motivating example. Let ρ be a production schedule satisfying $\mathbf{k}$. The least common multiple of {2,2,3,3,4} is 12; so Lemma 1 implies ρ has 12 basic periods.

Note that products 1, 2, and 5 have multipliers which are a power of 2 and products 3 and 4 have multipliers which are a power of 3. Let ρ_2 denote the sub-schedule of ρ consisting of products 1, 2, and 5, and ρ_3 denote the sub-schedule of ρ consisting of products 3 and 4. Lemma 1 implies ρ_2 has $lcm\{2,2,4\} = 4$ basic periods. Therefore, the total time required to setup and produce the products scheduled by ρ_2 in a basic period rotates between four values: $t_2^1, t_2^2, t_2^3, t_2^4$. Similarly, Lemma 1 implies ρ_3 has 3 basic periods and the total setup and production time of the products scheduled by ρ_3 rotates between three values: t_3^1, t_3^2, t_3^3. Table 1 illustrates the total time required for each period for all products. The schedule then repeats indefinitely.

Table 1: Total Time by Period

Period	1	2	3	4	5	6	7	8	9	10	11	12
Prod.1, 2,5	t_2^1	t_2^2	t_2^3	t_2^4	t_2^1	t_2^2	t_2^3	t_2^4	t_2^1	t_2^2	t_2^3	t_2^4
Prod. 3,4	t_3^1	t_3^2	t_3^3	t_3^1	t_3^2	t_3^3	t_3^1	t_3^2	t_3^3	t_3^1	t_3^2	t_3^3
Total	$t_2^1+t_3^1$	$t_2^2+t_3^2$	$t_2^3+t_3^3$	$t_2^4+t_3^1$	$t_2^1+t_3^2$	$t_2^2+t_3^3$	$t_2^3+t_3^1$	$t_2^4+t_3^2$	$t_2^1+t_3^3$	$t_2^2+t_3^1$	$t_2^3+t_3^2$	$t_2^4+t_3^3$

Note that each t_2^x is scheduled during the same period as each t_3^y for $x = 1,2,3,4$ and $y = 1,2,3$ exactly once. Hanson, et al (2013) observed this will always be the case provided that the number of basic periods of the two sub-schedules are "relatively prime (i.e. they do not share any common factors). Therefore, the production schedule is feasible if and only if

$$\max\{t_2^x | x = 1,2,3,4\} + \max\{t_3^y | y = 1,2,3\} \leq T \tag{7}$$

This result is generalized in Theorem 1.

Theorem 1: Let ρ be a production schedule with multipliers $\mathbf{k} = [k_1 \cdots k_n]$ and a fixed basic period T. Define $R = \{r_1, \dots, r_v\}$ to be a set of prime numbers and assume $k_j = \pi^{e_j}$ for some $\pi \in R$, and $e_j \in \mathbb{Z}^+$. Let ρ_π be the sub-schedule of ρ consisting of all products whose multiplier is a power of π for $\pi = r_1, \dots, r_v$. The number of basic periods in ρ_π is

$$b_\pi = \max\{k_j | k_j = \pi^{e_j},\ j = 1, \dots, n\}. \tag{8}$$

Therefore, the total time required to setup and produce the products scheduled by ρ_π rotates between b_π values: $t_\pi^1, \dots, t_\pi^{b_\pi}$. Finally, define $\tau_\pi = \max\{t_\pi^z | z \in 1, \dots, b_\pi\}$. The production schedule ρ is feasible if and only if $\tau_1 + \cdots + \tau_v \leq T$.

Proof: We will show that $\tau_1 + \cdots + \tau_v \leq T$ implies ρ is feasible. Note that each product is in exactly one sub-schedule. Therefore, the products scheduled for production in a basic period by ρ is the union of all products scheduled in the basic period by $\rho_{r_1}, \dots, \rho_{r_v}$. The time required to setup and produce all products scheduled by ρ_π is at most the maximum time τ_π, for $\pi = r_1, \dots, r_v$. Therefore, the time required to setup and produce every product scheduled by ρ is at most $\tau_1 + \cdots + \tau_v$ and, as we assumed $\tau_1 + \cdots + \tau_v \leq T$, the production schedule ρ is feasible.

It remains to be shown that ρ is feasible implies $\tau_1 + \cdots + \tau_v \leq T$. Recall each sub-schedule, ρ_π, has b_π basic periods. Define the b_π basic periods of ρ_π to be $\rho_\pi^1, \dots, \rho_\pi^{b_r}$ for $\pi = r_1, \dots, r_v$. Without loss of generality, suppose ρ_π^1 is scheduled for $\pi = r_1, \dots, r_v$ in the same basic period. By reasoning analogous to Lemma 1, the next time $\rho_{r_1}^1, \dots, \rho_{r_v}^1$ will be scheduled in the same basic period is $lcm(b_{r_1}, \dots, b_{r_v})$. As each b_π is the power of a distinct prime, $lcm(b_{r_1}, \dots, b_{r_v}) = b_{r_1} b_{r_2} \cdots b_{r_v}$. Furthermore, there are only $b_{r_1} b_{r_2} \cdots b_{r_v}$ distinct ways to select one ρ_π^y for each $\pi = r_1, \dots, r_v$, $y \in 1, \dots, b_\pi$. Therefore, every combination of ρ_π^y's must occur exactly once before $\rho_{r_1}^1, \dots, \rho_{r_v}^1$ is repeated: including $t_\pi^y = \tau_\pi$ for $\pi = 1, \dots, v$. Therefore, as ρ is feasible, $\tau_1 + \cdots + \tau_v \leq T$. QED.

Define N_π to be the set of all products whose multiplier is a power of a fixed prime π. Theorem 1 implies that feasibility can be determined by minimizing each τ_π, the maximum total time required to setup and produce all products in N_π scheduled in the same basic period, over all production schedules with $\mathbf{k} = [k_1 \cdots k_n]$ independently. Theorems 2, 3, and 4 are now presented to aid in the minimization of a given τ_π.

Let π be a prime number in R. If it is possible to schedule all products in N_π so that at most one product is scheduled per basic period, then τ_π is simply the maximum time to setup and produce an individual product. That is,

$$\tau_\pi = \max\{t_j \mid k_j = \pi^{e_j}, e_j \in \mathbb{Z}^+, j = 1, \dots, n\}. \tag{9}$$

Theorem 2 states that, if the total number of production runs for all products in N_π is less than the number of basic periods in the schedule ρ_π, then it is always possible to schedule one product per period.

Theorem 2: Let π be a prime number in R and assume there are η products in N_π. Without loss of generality, assume products $1, \dots, \eta$ are in N_π and $k_1 \leq k_2 \leq \cdots \leq k_\eta$. If $\sum_{j=1}^{\eta} 1/k_j \leq 1$, then $\tau_\pi = \max\{t_j | j \in 1, \dots, \eta\}$.

Proof: Note $k_1 \leq k_2 \leq \cdots \leq k_\eta$ implies k_η is the largest power of π used as a multiplier. Therefore, $lcm(k_1, \dots, k_\eta) = k_\eta$ and Lemma 1 implies any production schedule for products $1, \dots, \eta$ has exactly k_n basic periods. Recall product j is produced every k_j periods. Therefore, product j is produced k_n/k_j times in a production schedule and the total number of production runs over products $1, \dots, \eta$ is $\sum_{j=1}^{\eta} k_\eta/k_j$. Thus, the number of production runs is less than or equal to the number of basic periods if and only if

$$\sum_{j=1}^{\eta} \frac{k_\eta}{k_j} \leq k_\eta \text{ or } \sum_{j=1}^{\eta} \frac{1}{k_j} \leq 1. \tag{10}$$

The first production run of each product determines the production schedule. If the first production runs are assigned, in order from product 1 to product η, to the first available basic period, then each production run is in a distinct period. Q.E.D.

Consider the case where only one power, α, of a specific prime, π, is used as a multiplier. By Theorem 2, if the number of products in N_π is less than or equal to π^α, then

$$\tau_\pi = \max\{t_j | k_j = \pi^\alpha\}. \tag{11}$$

A closed form solution for τ_π is now determined for the case where the number of products in N_π is $\pi^\alpha + 1$.

Theorem 3: Let π be a prime number in R and assume there are $\eta = \pi^\alpha + 1$ products in N_π. Without loss of generality, assume products $1, \dots, \eta$ are in N_π and $t_1 \leq t_2 \leq \cdots \leq t_\eta$. Assume $k_j = \pi^\alpha$ for $j = 1, \dots, \eta$. Then $\tau_\pi = \max\{t_1 + t_2, t_\eta\}$.

Proof: A production schedule for products $1, \dots, \eta$ has π^α basic periods, and each product is produced in exactly one basic period. Therefore, as there are π^α basic periods and $\pi^\alpha + 1$ products, at least two products must be scheduled in the same basic period. The minimum time required to produce two products is $t_1 + t_2$. Furthermore, t_η appears in some basic period. Therefore, $\tau_\pi \geq \max\{t_1 + t_2, t_\eta\}$. Any production schedule where products 1 and 2 are scheduled in the same period and all other products are scheduled in a distinct periods will obtain $\tau_\pi = \max\{t_1 + t_2, t_\eta\}$. Q.E.D.

Without loss of generality, assume products $1, \dots, \eta$ are in N_π and define $\mathbf{k}_\pi = [k_1 \cdots k_\eta]$, $k_1 \leq k_2 \leq \cdots \leq k_\eta$ and $\mathbf{t}_\pi = [t_1 \cdots t_\eta]$. For some values of $(\mathbf{k}_\pi, \mathbf{t}_\pi)$ it is possible to define an equivalent problem, $(\mathbf{k}'_\pi, \mathbf{t}'_\pi)$, which has the same value of τ_π as $(\mathbf{k}_\pi, \mathbf{t}_\pi)$. If τ_π has already been found for $(\mathbf{k}'_\pi, \mathbf{t}'_\pi)$, then there is no need to solve τ_π for $(\mathbf{k}_\pi, \mathbf{t}_\pi)$. An illustrative example is provided below.

Let $\mathbf{k}_\pi = [2\ 2\ 4\ 8], \mathbf{t}_\pi = [t_1\ t_2\ t_3\ t_4]$, and θ be an arbitrary production schedule for products 1, 2, and 3. Note θ has four basic periods and product 4 is produced every eight periods. Therefore, product 4 is scheduled every other iteration of θ. The maximum time required to setup and produce each product scheduled during a basic period (τ_2) clearly occurs during an iteration of θ with product 4 (as it is the same production schedule, plus an additional product). Therefore, theoretically, if product four is produced in **every iteration of $\boldsymbol{\theta}$ (but with the same value for $\boldsymbol{t_4}$),** then the value of τ_2 will be the same. Thus, $\mathbf{k}_\pi = [2\ 2\ 4\ 8], \mathbf{t}_\pi = [t_1\ t_2\ t_3\ t_4]$ and $\mathbf{k}'_\pi = [2\ 2\ 4\ 4], \mathbf{t}'_\pi = [t_1\ t_2\ t_3\ t_4]$ have the same value of τ_2.

Alternatively, a fifth product with, $k_5 = 8$, $t_5 \leq t_4$, could be added without changing τ_2 by scheduling product 5 in product 4's place in the alternate iterations of θ. In general, if the number of products with $k_j = k_\eta$ (note: k_η is the largest power of π in $\mathbf{k}_\pi$) is less than or equal to k_η divided by the second largest power (ignoring multiplicity) in $\mathbf{k}_\pi$, then the set of products with $k_j = k_\eta$ can be regarded as a single product, j^*, with k_{j^*} equal to the second largest power of π in in $\mathbf{k}_\pi$ and t_{j^*} equal to the largest total setup and production among the products with $k_j = k_\eta$.

Theorem 4: Let π be a prime number in R. Without loss of generality, assume products $1, \dots, \eta$ are in N_π, and $k_1 \leq k_2 \leq \cdots \leq k_\eta$. Define $\mathbf{k}_\pi = [k_1 \cdots k_\eta]$, and $\mathbf{t}_\pi = [t_1 \cdots t_\eta]$. Let m_1 and m_2 be the largest and second largest (without counting multiplicity) powers of π in

$\mathbf{k}_\pi$, respectively. Let M be the number of products with $k_j = m_1$. If $M \leq m_1/m_2$, then τ_π is the same for $(\mathbf{k}_\pi, \mathbf{t}_\pi)$ and $\mathbf{k}'_\pi = [k_1, \ldots, k_{\eta-M}, m_2]$, $\mathbf{t}'_\pi = \left[t_1, \ldots, t_{\eta-M}, \max\left\{t_{\eta-M+1}, \ldots, t_\eta\right\}\right]$.

Proof: Observe the first $\eta - M$ products in $(\mathbf{k}_\pi, \mathbf{t}_\pi)$ and $(\mathbf{k}'_\pi, \mathbf{t}'_\pi)$ are identical. Let θ be an arbitrary production schedule for products $1, \ldots, \eta - M$. The final product in $\mathbf{k}'_\pi$, $\eta - M + 1$, has a multiplier of m_2 and its total setup and production time is $\max\left\{t_{\eta-M+1}, \ldots, t_\eta\right\}$. Let j^* be a product in $\mathbf{k}_\pi$ such that $t_{j^*} = \max\left\{t_{\eta-M+1}, \ldots, t_\eta\right\}$ and $k_{j^*} = m_1$. Note product $\eta - M + 1$ in $\mathbf{k}'_\pi$ and product j^* in $\mathbf{k}_\pi$ have the same total setup and production time. Therefore, scheduling $\eta - M + 1$ in $\mathbf{k}'_\pi$ or j^* in $\mathbf{k}_\pi$ in the same basic period of θ will result in the same τ_π. Note, product j^* in $\mathbf{k}_\pi$ will appear in fewer iterations of θ, but this does not alter τ_π. Furthermore, each of the remaining products in $\mathbf{k}_\pi$ with a multiplier of m_1 can be scheduled in j^*'s place in the remaining $m_1/m_2 - 1$ iterations of θ without altering τ_π. Q.E.D.

Theorems 1-4 provide methods to quickly determine the feasibility of $(\mathbf{k}, \mathbf{t})$ for many values of $\mathbf{k}$ and $\mathbf{t}$. To illustrate, consider the following example: $\mathbf{k} = [1\ 2\ 2\ 3\ 3\ 3\ 4]$, $\mathbf{t} = [5\ 4\ 2\ 2\ 3\ 3\ 3]$, and $T = 15$. Product 1 is produced in every period. Therefore, we subtract t_1 from T and obtain $\mathbf{k}' = [2\ 2\ 3\ 3\ 3\ 4]$, $\mathbf{t}' = [4\ 2\ 2\ 3\ 3\ 3]$, $T' = 10$.

By Theorem 1, $(\mathbf{k}', \mathbf{t}')$ is feasible if and only if $\tau_2 + \tau_3 \leq 10$. Note $N_2 = \{1', 5', 6'\}$ and $N_3 = \{3', 4', 5'\}$. Theorem 4 implies [2 2 4] has the same value of τ_2 as [2 2 2] with the same production times. Theorem 3 implies τ_2 for [2 2 2] is $\max\{t'_1 + t'_2, t'_3\}$ where $t'_1 \leq t'_2 \leq t'_3$ are the production times of the three products. Therefore, $\tau_2 = \max\{2 + 3, 4\} = 5$.

Consider N_3. As $1/k_{3'} + 1/k_{4'} + 1/k_{5'} = 1/3 + 1/3 + 1/3 \leq 1$, Theorem 2 implies $\tau_3 = \max\{2,2,3\} = 3$. Therefore, $\tau_2 + \tau_3 = 8 < 10$ and we conclude a feasible schedule exists for $\mathbf{k} = [1\ 2\ 2\ 3\ 3\ 3\ 4]$, $\mathbf{t} = [5\ 4\ 2\ 2\ 3\ 3\ 3]$, and $T = 15$. Note that there is no benefit to constructing the actual schedule during the implementation of the algorithm. Only when the optimal solution is found does an actual schedule need to be found.

4.5 Determining τ_π (the maximum time required to setup and produce all products j where k_j is a power of π in any given basic period):

Different production schedules may minimize τ_π for different values of $\mathbf{t}$ and the same $\mathbf{k}$. Our approach is to generate a set of production schedules for each $\mathbf{k}$ such that at least one production schedule will minimize τ_π. Towards this goal, it is unnecessary to consider permutations of the same schedule. Additionally, without loss we assume $k_1 \leq k_2 \leq \cdots \leq k_n$ and $t_j \leq t_{j_0}$ if $k_j = k_{j_0}$ and $j < j_0$ for the remainder of the section.

The case where $R = \{2,3,5,7\}$ and 1,2,3,4,5,7,8,9 are allowed as multipliers is now considered. First, the setup and production time of all products with $k_j = 1$ can be subtracted from the basic period, T, as in the previous example. Second, the products are divided into the sets N_2, N_3, N_5, and N_7 as defined in Section 3.4.1. Note that the powers of two are 2,4,8, the powers of three are 3,9, the only power of five is 5, and the only power of seven is 7.

Determining τ_7: The set of products with $k_j = 7$ (i.e. N_7) is considered first because it is the simplest case. Each product in N_7 has a multiplier of 7, so the only variable is the number of products in N_7. Theorem 2 can be applied if there are seven or fewer products in N_7 and Theorem 3 can be applied if there are eight products in N_7.

The only remaining possibility is nine products with a multiplier of seven. If there are nine products and seven basic periods, the number of products scheduled in each period (ignoring permutations) is either [3,1,1,1,1,1,1] or [2,2,1,1,1,1,1]. In the first case, the products with the three smallest production times, 1,2,3 should be scheduled together and the products with the six largest production times, products 4,5,6,7,8,9, should be scheduled in distinct basic periods (i.e. [123|4|5|6|7|8|9]).

In the second case, it is immediately clear that the five products with the largest production times should be scheduled by themselves. There are three ways to schedule products 1,2,3,4 in two periods with two products scheduled per period: [12|34], [13|24], and [14|23]. A simple “interchange” argument can be used to show [14|23] will always minimize the maximum total setup and production time for these two periods. This argument is used a number of times in the remainder of the section.

Determining τ_5: Again, the only power of five used is 5, and the only variable is the number of products in N_5. Theorems 2 and 3 address the cases where there are six or fewer products in N_5. The remaining possibilities are considered by cases:

Case 1: Suppose $|N_5| = 7$. The number of products produced per basic period, ignoring permutations, is either [3,1,1,1,1] or [2,2,1,1,1]. By an argument analogous to $|N_7| = 9$, $\tau_5 = \min\{max\{t_1 + t_2 + t_3, t_7\}, max\{t_1 + t_4, t_2 + t_3, t_7\}\}$.

Case 2: Suppose $|N_5| = 8$. The number of products per basic period is either [4,1,1,1,1], [3,2,1,1,1], or [2,2,2,1,1]. For [4,1,1,1,1], the products with the four smallest production times should be scheduled together (just as in [3,1,1,1,1,1,1]). For [3,2,1,1,1], products six, seven, and eight should be scheduled in a distinct period. The first five products need to be partitioned into a set of three and a set of two. However, not all of the ten possible partitions need to be considered. For example, $[145|23]$ will always have a higher value of τ_5 than $[123|45]$ because 145 has a greater total setup and production time than both 123 and 45 (a two-product interchange). In general, if the largest numbered product in the set of three is greater than the largest numbered product in the set of two and the second largest numbered product in the set of three is greater than the second largest numbered product in the set of two, then a two-product interchange exists which will result in a production schedule with a lower (or equal) value of τ_5. The relevant partitions are shown in Table 2.

Table 2: Partitions of 5 products into set of 3 and set of 2

Set of 3 Products	Set of 2 Products
123	45
124	35
125	34
134	25
234	15

Each partition corresponds to a production schedule whose value of τ_5 is the maximum of the total setup and production times of the set of three products, the set of two products, and the largest product scheduled by itself, t_8. Therefore, the minimum value of τ_5 for [3,2,1,1,1] is simply the minimum value of τ_5 of these 5 partitions.

For [2,2,2,1,1], products 5,6,7 are scheduled by themselves and products 1-4 need to be partitioned into three sets of two. We have already considered the case where four products were partitioned into two sets of two and observed the only production schedule which needs to be considered is $[14|23]$. Similarly, each pair of 2-product sets has an interchange which results in a lower value of τ_5 unless the schedule has the form $[ad|bc]$ where $a \le b \le c \le d$. The only production schedule with this property simultaneously for each pair of the three 2-product sets is $[16|25|34]$.

Case 3: Suppose $|N_5| = 9$. The number of products per basic period is either [5,1,1,1,1], [4,2,1,1,1], [3,3,1,1,1], or [3,2,2,1,1]. For [5,1,1,1,1], the five products with the smallest t_j's are scheduled together. For [4,2,1,1,1], products 7,8,9 should be scheduled by themselves and products 1-6 need to be partitioned into a set of four and a set of two so that no two-product interchange exists (similar to the [3 2 1 1 1] in Case 2). The possible partitions are given in Table 3.

Table 3: Partitions of 6 products into set of 4 and set of 2

Set of 4 Products	Set of 2 Products
1234	56
1235	46
1245	36
1345	26
2345	16
1236	45

For [3,3,1,1,1], products 7,8,9 are scheduled by themselves and products 1-6 need to be partitioned into two sets of three. An interchange exists if and only if the production schedule is of the form $[abc|def]$ where $a \leq d, b \leq e, c \leq f$. The remaining partitions are shown in Table 4.

Table 4: Partitions of 6 products into two sets of 3

First Set of 3	Second Set of 3
126	345
136	245
146	235
156	234
145	236

For [3,2,2,1,1], products 8 and 9 are scheduled by themselves and products 1-7 need to be partitioned into a set of three products and two sets of two products. Once the set of three products is chosen, the remaining four products form a [2|2] problem and must be scheduled in the form $[ad|bc]$ where $a \leq b \leq c \leq d$. Furthermore, the largest and second largest numbered products in the set of three cannot be larger the largest and second largest numbered products in either pair, respectively (just as in the [3|2|1|1|1] case). The possible partitions are displayed in Table 5.

Table 5: Partitions of 8 products into set of 3 and two sets of 2

Set of 3 Products	Set of 2 Products	Set of Two Products
123	47	56
124	37	56
125	37	46
126	37	45
127	36	45
134	27	56
135	27	46
136	27	45
145	27	36
234	17	56
235	17	46
236	17	45
245	17	36
345	17	26

Determining τ_3: The case for τ_3 is more complex as two powers of three, 3 and 9, are allowed as multipliers. The cases where only 9 or only 3 is used as a multiplier can be approached in similar fashion to τ_7 and τ_5 in the previous two sections.

Case A: Let $k_j = 9$ for all $j \in N_3$. Theorem 2 addresses all cases.

Case B: Let $k_j = 3$ for all $j \in N_3$. Theorems 2 and 3 address $|N_3| \leq 4$. As $k_j = 3$ for all $j \in N_j$, any production schedule will have three basic periods. For $|N_3| = 5$, the possible number of products produced per period is [3,1,1] and [2,2,1] which minimize τ_3 using $[123|4|5]$ and $[14|23|5]$. For $|N_3| = 6$, the possible number of products produced per period is [4,1,1], [3,2,1], and [2,2,2]: which are solved using $[1234|5|6]$, Table 2, and [16|45|23][1], respectively. For $|N_3| = 7$, the possible number of products produced per period are [5,1,1], [4,2,1], [3,3,1], and [3,2,2]: which are solved using $[12345|6|7]$, Table 3, Table 4, and Table 5, respectively.

The cases where $|N_3| = 8$ and $|N_3| = 9$ can be solved in an analogous manner, but additional tables need to be generated. For $|N_3| = 8$, the possible number of products produced per

---

[1] [2|2|2|1|1] was considered in Section 4.3.2.2.

period are [6,1,1], [5,2,1], [4,3,1], [4,2,2], and [3,3,2]. The production schedule $[123456|7|8|9]$ minimizes τ_3 for [6,1,1]. For [5,2,1] and [4,3,1], product 8 is scheduled by itself and tables containing all partitions without an interchange can be constructed for [5,2] and [4,3] (similar to [3,2], [4,2]). The [4,2,2] case can be addressed in analogous fashion to the [3,2,2] case. For [3,3,2], six products are chosen for the two 3-product periods. These six products can be partitioned using all five rows in table 4 (the [3,3] table) where the products are ordered by index (i.e. if 1,2,4,5,7, and 8 are selected then $1' = 1, 2' = 2, 3' = 4, 4' = 5, 5' = 7, 6' = 8$). However, only the partitions where a two-product interchange does not exist between one of the 3-product periods and the 2-product period need be considered. Conditions for a 2-product interchange between a 3-product set and 2-product set are discussed in the [3,2,1,1,1] case at the bottom of page 15 (See Appendix for Tables).

For $|N_3| = 9$, the possible number of products produced per period are [7,1,1], [6,2,1], [5,3,1], [5,2,2], [4,3,2], and [3,3,3]. For [7,1,1], the production schedule $[1234567|8|9]$ minimizes τ_3. Again, product 9 is scheduled by itself and tables can be generated for all partitions without an interchange for [6,2,1] and [5,3,1]. Cases [5,2,2], [4,3,2], and [3,3,3] can be addressed in similar fashion to [4,2,2] and [3,3,2] above.

Case C: Theorem 4 can be combined with the work in Case 2 to solve any set of multipliers with three or fewer nines and Theorem 2 address some additional cases where the number of production runs is less than or equal to nine. The remaining cases are 339999, 3339999, 3399999, 33339999, 33399999, 33999999, 39999999, 333339999, 333399999, 3333999999, 339999999, 399999999.

Because there are two different multipliers, it is no longer the case that $t_1 \le t_2 \le \cdots \le t_n$ and the number of cases is potentially much greater. Our approach is to consider all possible ways to schedule the products with multipliers of three. This results in a production schedule with three basic periods. Let the total setup and production time for the basic periods in the production schedule be z_1, z_2, and z_3 where, without loss, $z_1 \le z_2 \le z_3$. For example, if there are five products with $k_j = 3$ then one way to schedule these products is [14|23|5]. Then $z_1 = t_1 + t_4$, $z_2 = t_2 + t_3$, and $z_3 = t_5$ (illustrated in Figure 2).

Period 1			Period 2			Period 3	
1	4	Idle	2	3	Idle	5	Idle
z_1		Idle	z_2		Idle	z_3	Idle

Figure 2: Illustration of z_1, z_2, z_3 for $[14|23|5]$

Furthermore, the production schedule for the products with $k_j = 3$ is repeated three times so there are three basic periods with z_1 scheduled time units, three basic periods with z_2 scheduled time units, and three basic periods with z_3. The only remaining step is to schedule the products with $k_j = 9$. An example production schedule where there are four products, 6, 7, 8, 9, with $k_j = 9$ is illustrated in Figure 3.

Period 1			Period 2			Period 3		Period 4			Period 5		Period 6		Period 7			Period 8		Period 9	
z_1	9	I*	z_2	6	I	z_3	I	z_1	8	I	z_2	I	z_3	I	z_1	7	I	z_2	I	z_3	I

*I is idle time

Figure 3: Adding products with $k_j = 9$ to an existing 3-period schedule

Note that the number of products with $k_j = 3$ (assuming there is at least two) is independent of how to schedule the products with $k_j = 9$, and that the case where there are 1, 2 or 3 products with $k_j = 9$ has already been addressed by Theorem 4. Therefore, the remaining cases are when there are 4,5,6,7 or 8 products with $k_j = 9$.

Before we consider the cases explicitly, we address some arguments common to all cases. Without loss, re-index the products with $k_j = 9$ from $1, \dots, \eta_9$ where η_9 is the number of products with $k_j = 9$. All three of the "z_1" basic periods will always be used. The following observations apply.

(1) If only three basic periods are used, then the problem is analogous to the case where $k_j = 3$ instead of 9.

(2) If products $1, \ldots, \eta_9$ are scheduled in four basic periods, then the products are schedule in the three z_1 basic periods and one z_2 basic period. Furthermore, two or more products will not be scheduled in the z_2 period (as there are three z_2 periods available) and if the single product scheduled in z_2 is (strictly) greater than one, then any lower numbered product could be scheduled in a different z_2 period with a (potentially) lower value of τ_3. Therefore, product 1 is the only product scheduled in the z_2 period. Products $2, \ldots, \eta_9$ are analogous to $\eta_9 - 1$ products with $k_j = 3$.

(3) Similarly, if five periods are used to schedule products $1, \ldots, \eta_9$, then products 1 and 2 are scheduled in different z_2 periods and products $3, \ldots, \eta_9$ are analogous to $\eta_9 - 2$ products with $k_j = 3$.

(4) If each product is scheduled in a distinct basic period, then the products are scheduled in decreasing order of total setup and production time in the z_1 periods, the z_2 periods, and then the z_3 periods.

Observations (1) and (2) address the case $\eta_9 = 4$. Observations (1), (2), and (4) address $\eta_9 = 5$. Observations (1), (2), (3), and (4) address $\eta_9 = 6$. For $\eta_9 = 7$, either 3, 4, 5, 6, or 7 basic periods are used. The cases where 3, 4, 5, or 7 basic periods are used are addressed by observations (1), (2), (3), and (4), respectively.

The case where six of the nine periods are used is slightly more complex. All three z_1 and all three z_2 periods will be used. No interchange can exist between any of the three z_1 periods, and similarly, no interchange can exist between any of the three z_2 periods. Furthermore, a relationship exists between each z_2 schedule and each z_1 schedule. Because $z_2 \geq z_1$, the total setup and production time of the products scheduled in a z_2 period must be less than the total setup and production time of the products scheduled in all three of the z_1 periods. Therefore, either no interchange exists between a z_2 schedule and a z_1 schedule, or else the z_2 schedule requires less time than the z_1 schedule (See Table 20 in Appendix).

For $\eta_9 = 8$, the only possibility is one product with a multiplier of three. There are six basic periods which have no scheduled products, so at least 6 periods are used to schedule the eight products with $k_j = 9$. If exactly six basic periods are used, then no interchanges are allowed and the two relevant production schedules are $[123|4|5|6|7|8]$ and $[14|23|5|6|7|8]$. If seven basic periods are used, then the six of the basic periods used will have no scheduled products and one will have the product with $k_j = 3$ already scheduled. By the same reasoning as observation (2), product 1 must be the only product added to the basic period where the product with $k_j = 3$ is already scheduled. Products 2 and 3 are scheduled together, and products 4, 5, 6, 7, and 8 are scheduled by themselves. If eight basic periods are used, observation (4) applies.

The minimum value of τ_2 can be found in similar fashion. Recall the multipliers 2,4,8 are the powers of two that are allowed. The cases where only 2, only 4, and only 8 are used as multipliers can be addressed in similar fashion to the tables for 5, 7, only 3, and only 9. The cases for only 2 and 4, only 2 and 8, and only 4 and 8 as multipliers can be constructed in similar fashion to the tables for 3 and 9. Note the case 2, 4, and 8 are used is equivalent to only 4 and only 8.

The above work provides necessary and sufficient conditions to determine the feasibility of all **k** for a given **t**. The values from **t** can be substituted into the conditions to quickly determine the feasibility of **k**.

5 Experimentation:

5.1 Bomberger Data: The Bomberger data is considered at a variety of utilization levels by scaling the demand rates appropriately. The solution proven to be optimal in Hanson et al. (2013) for the 0.8824 utilization considered by many authors is obtained by the power-of-two and power-of-prime methods. The power-of-two solution is less than the GA solution for 14 of the 15 utilization levels considered and the power-of-prime solution is less than the GA solution for all 15 utilizations. The power-of-prime solution considers all power-of-two schedules, and, consequently, will always have a lower (or equal) cost than the power-of-two solution. The power-of-prime solution is (strictly) superior to the power-of-two solution in 8 of the 15 utilizations. Over all, the power-of-two method averaged 1.63% above the lower bound, the power-of-prime method 1.01% above the lower bound, and the GA 9.61% above the lower bound for the utilizations less than or equal to 0.92. The performance of all three algorithms for 0.95 and 0.97 utilizations is significantly worse for all three methods. This will be addressed in Section 5.3. The results are provided in the table below where P2 is exact solution for the power-of-two method, PP is the exact solution for the power-of-primes method, GA is the genetic algorithm of Khouja, et al. (1998), and BLB is Bomberger's (1966) lower bound.

Table 6: Comparison P2, PP, GA for Bomberger's Data

Util.	P2	PP	GA	BLB	P2/BLB	PP/BLB	GA/BLB	GA/P2	GA/PP
0.6618	28.51	28.17	29.29	28.08	1.0153	1.0032	1.0431	1.0274	1.0398
0.8824	32.07	32.07	36.59	31.62	1.0142	1.0142	1.1572	1.1409	1.1409
0.50	25.25	24.91	25.16	24.84	1.0165	1.0028	1.0129	0.9964	1.0100
0.55	26.33	25.99	26.37	25.91	1.0162	1.0031	1.0178	1.0015	1.0146
0.60	27.34	27	27.59	26.92	1.0156	1.0030	1.0249	1.0091	1.0219
0.65	28.3	27.95	28.81	27.86	1.0158	1.0032	1.0341	1.0180	1.0308
0.70	29.2	28.91	30.81	28.76	1.0153	1.0052	1.0713	1.0551	1.0657
0.75	30.04	29.88	32.46	29.6	1.0149	1.0095	1.0966	1.0806	1.0863
0.80	30.84	30.83	33.73	30.4	1.0145	1.0141	1.1095	1.0937	1.0941
0.83	31.3	31.3	34.38	30.85	1.0146	1.0146	1.1144	1.0984	1.0984
0.86	31.75	31.75	35.64	31.3	1.0144	1.0144	1.1387	1.1225	1.1225
0.89	32.18	32.18	36.98	31.73	1.0142	1.0142	1.1655	1.1492	1.1492
0.92	33.11	33.11	40.61	32.14	1.0302	1.0302	1.2635	1.2265	1.2265
0.95	49.79	49.79	50.08	35.08	1.4193	1.4193	1.4276	1.0058	1.0058
0.97	71.39	71.39	71.43	47.05	1.5173	1.5173	1.5182	1.0006	1.0006

5.2 Experimental Design: The primary focus of this paper is on obtaining production schedules which easily translate to the conditions encountered in practice. Towards this goal, the choice of experimental design should reflect values for parameters commonly found in practice. Mallya's (1992) case study is the most recent example of "real world" data found by the authors in the literature. Mallya considers a variation of the ELSP where products are not available until a lot is complete. Unfortunately, this prevents a direct comparison. However, this does not alter the demand and productions rates, setup times, costs, etc. To construct the design, each parameter is selected from intervals containing Mallya's data using a uniform distribution. For the ten product case, the demand rates are divided by two in order to obtain reasonable utilizations. Values of internal interest rates vary depending on author, company, interest rates, etc. For example, Bomberger used 0.1 and Mallya 0.35. A value of 0.2 is used for the internal interest rate as a moderate value.

Table 7: Experimental Design

	5 Products		10 Products	
Parameter	Min	Max	Min	Max
Production Rate	1500	3000	1500	3000
Demand Rate	100	1000	50	500
Setup Time	0.15	0.35	0.15	0.35
Procurement Cost	0.25	1.25	0.25	1.25
Setup Cost	50	150	50	150

Problems were generated until 25 problems were found for each utilization level in the tables below for five and 10 products.

5.3 Note on choice of the Experimental Design: High utilization problems are among the most difficult to schedule and have been a popular area of study. For example, consider the case where the utilization is 0.99. If the machine is producing products 99% of the time, then the setup times must require less than 1% of the total time. As a result, the cycle time of the products are forced to be significantly larger than is cost effective. To quantify the difference in cost, Bomberger's (1966) lower bound is used as a conservative estimate of total cost and compared to the cost of producing the products independently.

Dobson's (1987) experimental design has been a popular choice for subsequent unequal-lot size approaches such as Raza and Akgunduz (2008). The methods obtained by these authors provide excellent solutions for these parameters. However, the total cost for these problems is extremely high due to the combination of high utilization and setup times.

Ten problems were generated using experimental designs 1, 2, and 3 (Raza and Akgunduz, 2008). Bomberger's lower bound (1966) is used as a conservative estimate of the total cost of production on a single machine. For comparison, Bomberger's lower bound is divided by the cost of producing the products "independently" (on separate machines). The average ratio for Raza and Akgunduz is approximately 10 at a utilization of 0.7 and 100 at 0.975. The average ratio of each sample vs. utilization is shown in Figure 5. In comparison, for the "real world" data provided by Bomberger (1966) and Mallya (1992), the ratio is 1.0 for utilizations less than 0.925 and 0.959, respectively. The values of the parameters from Dobson's (1987) experimental design appear to be too cost prohibitive to be observed in practice.

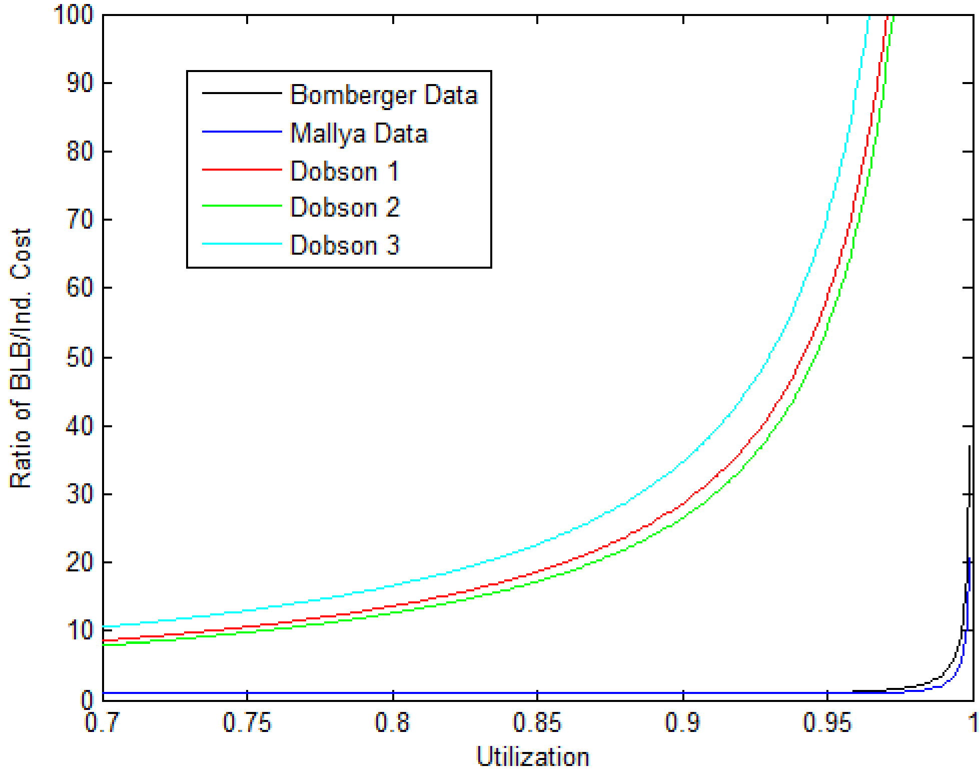

Figure 4: Total Cost vs. Utilization

5.4 Results: On average, the power-of-primes method for the 75, five-product problems with utilizations between 0.6 and 0.9 obtained solutions 0.81% above the lower bound. For the 75 ten product problems, the power-of-prime method was, on average, 0.66% above the lower bound. In comparison, the power-of-two method averaged 0.98% and 1.29% above the lower bound respectively.

Both the power-of-two and power-of-prime methods obtain results near the lower bound with the power-of-primes method performing better for problems with utilizations between 0.6 and 0.7. Specifically, for the five product case, the power-of-primes method (strictly) out-performed the power-of-two method in 29 of the 75 problems with utilization between 0.6 and 0.9. For ten products, the power-of-primes algorithm obtained strictly better solutions for 73/75 problems with utilizations between [0.6,0.9]. In general, it appears that the greater the number of products, the greater the potential for the power-of-primes method to outperform the power-of-two method. As a direct consequence of Theorem 1, having a single product, j, with a $k_j = 3$ essentially triples the utilization of that product. However, if there are multiple products with a multipliers which are a power of 3, this is not the case. It is easy to see that, for even moderately loaded machines, it may be difficult to find a feasible production schedule without several products having multipliers which are powers of the same prime. The more products there are the greater the number of possible combinations which may allow for use of multipliers which are powers of multiple primes.

Table 8: Five Product Results

	Mean			Stan. Dev.		
Utilization	P2/BLB	PP/BLB	P2/PP	P2/BLB	PP/BLB	P2/PP
[0.6,0.7]	1.010482	1.006267	1.004193	0.003752	0.003261	0.003668
[0.7,0.8]	1.008031	1.007055	1.00097	0.003878	0.003761	0.002081
[0.8,0.9]	1.010968	1.010968	1	0.006216	0.006216	0
[0.9,1]	1.044783	1.044599	1.000176	0.043214	0.043245	0.000871

Table 9: Ten Product Results

	Mean			Stan. Dev.		
Utilization	P2/BLB	PP/BLB	P2/PP	P2/BLB	PP/BLB	P2/PP
[0.6,0.7]	1.012631	1.004947	1.007648	0.003641	0.002543	0.003196
[0.7,0.8]	1.012968	1.005919	1.007012	0.002025	0.002301	0.002735
[0.8,0.9]	1.013047	1.008829	1.004186	0.003299	0.003738	0.003037
[0.9,1]	1.06662	1.06662	1	0.047232	0.047727	0

The algorithm was run on a 3 GHz machine using Matlab 2012. The average computation time for the 5-product problems was 0.0512 seconds for the power-of-two algorithm and

0.1694 seconds for the power-of-primes algorithm. The average computation time for the 10-product problems was 0.7492 seconds for power-of-two and 1.5428 seconds for power-of-prime. Some high utilization (greater than 0.9) ten product problems timed out (after 1 min.) after not finding any feasible production schedule with a cost less than the rotation schedule and after considering in excess of 100,000 production schedules with potential lower costs. These cases are similar to the problems generated using Dobson's (1987) design. If utilization is increased while holding other parameters constant, it is typical that the optimal production schedule "converges" to the rotation schedule. Table 10 gives an illustrative example using the Bomberger data.

Table 10: P2 and PP Optimal **k**'s for Bomberger's Problem

Utilization	Optimal P2 **k**	Optimal PP **k**
0.8	9331399133	8221248122
0.83	8221258122	8221248122
0.86	8221248122	8221248122
0.89	8221248122	8221248122
0.92	2121288124	2121288124
0.95	1111111111	1111111111

6 Conclusion:

An algorithm was developed which obtains optimal solutions over all production schedules under the power-of-two and power-of-prime assumptions. The power-of-primes method is compared to Khouja et al.'s (1998) basic period genetic algorithm using the well-known Bomberger (1966) data at various loads with favorable results. An experimental design is constructed which more accurately reflects problems commonly encountered in practice than Dobson's (1987) design. For a wide variety of problems, the power-of-primes method obtains solutions within 1% of the lower bound. Analysis was performed to evaluate the effect of high utilization and setup time on total cost for problems such as those generated by Dobson (1987) and Raza and Akgunduz (2008).

In part III, recovery from critically low inventory positions is addressed.

CHAPTER 3 REFERENCES

Andres, F.M. and Emmons, H. (1976) On the optimal packaging frequency of products jointly replenished. *Management Science*, **22(**10), 1165-1166.

Bomberger, E. (1966) A dynamic programming approach to a lot size scheduling problem. *Management Science*, **12**(11), 778-784.

Doll, C. and Whybark, D. (1973) An iterative procedure for the single-machine multi-product lot scheduling problem. *Management Science*, **20**(1), 50-55.

Eilon, S. (1962) *Elements of Production Planning and Control*, Macmillan, New York, Chapter 10, 227-263.

Elmaghraby, S. (1978) The economic lot scheduling problem (ELSP): review and extensions. *Management Science*, **20**(1), 50-55.

Haessler, R. and Hogue, S. (1976) A note on the single-machine multi-product lot scheduling problem. *Management Science*, **22**(8), 909-912.

Hanson, B., Hodgson, T., Kay, M., King, R., Thoney-Barletta, K., (2013) On the Lot Size Scheduling Problem, Part I: The Optimal Solution to the Bomberger data. Submitted to: *International Journal of Production Research*

Hanssmann, F. (1962) *Operations Research in Production and Inventory Control*, Wiley, New York.

Hodgson, T.J. (1970) Addendum to Stankard and Gupta's Note on Lot Size Scheduling. *Management Science*, **16**(7), 514-517.

Hsu, W.L. (1983) On the general feasibility test of scheduling lot sizes for several products on one machine. *Management Science,* **29**(1), pp. 93-105.

Jones, P. and Inman, R. (1989) When is the economic lot scheduling problem easy? *IEE Transactions*, **21**(1), 11-20.

Khouja, M., Michalewicz, Z., Wilmont, M. (1998) The use of genetic algorithms to solve the economic lot size scheduling problem. *European Journal of Operational Research*, **110**, 509-524.

Raza, A. and Akgunduz, A. (2007) A comparative study of heuristic algorithms on Economic Lot Scheduling Problem. *Computers & Industrial Engineering*, **55**, 94-109

Rogers, J. (1958) A computational approach to the economic lot scheduling problem. *Management Science*, 4(3), 264-291.

Schweitzer, P. and Silver, E. (1983) Mathematical pitfalls in the one machine multiproduct economic lot scheduling problem. *Operations Research*, **31(**2), 401-405

Stankard, M. and Gupta, S. (1969) A note on Bomberger's approach to lot size scheduling: heuristic proposed. *Management Science*, **15**(7), 449-552.

CHAPTER 3 APPENDIX

Table 11: Partitions of 7 products into set of 5 and set of 2

Set of 5 Products	Set of 2 Products
12345	67
12346	57
12356	47
12456	37
13456	27
23456	17
12347	56

Table 12: Partitions of 7 products into set of 4 and set of 3

Set of 4 Products	Set of 3 Products
1234	567
1235	467
1236	457
1237	456
1245	367
1246	357
1247	356
1256	347
1257	346
1267	345
1345	267
1346	257
1347	256
1356	247
1357	246
1456	237
2345	167
2346	157
2347	156
2356	147
2456	137
3456	127

Table 13: Partitions of 8 products into set of 4 and two sets of 2

Set of 4 Products	Set of 2 Products	Set of 2 Products
1234	58	67
1235	48	67
1236	48	57
1237	48	56
1238	47	56
1245	38	67
1246	38	57
1247	38	56
1256	38	47
1345	28	67
1346	28	57
1347	28	56
1356	28	47
1456	28	37

Table 14: Partitions of 8 products into two sets of 3 and set of 2

Set of 3 Products	Set of 3 Products	Set of 2 Products
All rows in Table 4 using 1,2,3,4,5,6		78
All rows in Table 4 using 1,2,3,4,5,7		68
All rows in Table 4 using 1,2,3,4,6,7		58
All rows in Table 4 using 1,2,3,5,6,7		48
All rows in Table 4 using 1,2,4,5,6,7		38
All rows in Table 4 using 1,3,4,5,6,7		28
All rows in Table 4 using 2,3,4,5,6,7		18
All rows in Table 4 using 1,2,3,4,5,8		67
128	346	57
138	246	57
148	236	57
146	238	57
128	356	47
138	256	47
156	238	47
128	456	37
128	347	56
138	247	56
148	237	56
147	238	56

Table15: Partitions of 8 products into set of 6 and set of 2

Set of 6 Products	Set of 2 Products
123456	78
123457	68
123467	58
123567	48
124567	38
134567	28
234567	18
123458	67

Table 16: Partitions of 8 products into set of 5 and set of 3

Set of 5 products	Set of 3 Products
12345	678
12346	578
12356	478
12456	378
13456	278
23456	178
12347	568
12357	468
12457	368
13457	268
23457	168
12367	458
12467	358
13467	258
23467	158
12567	348
13567	248
23567	148
14567	238
24567	138
34567	128
12348	567
12358	467
12458	367
13458	267
23458	167
12368	457
12378	456

Table 17: Partitions of 9 products into set of 5 and two sets of 2

Set of 5 Products	Set of 2 Products	Set of 2 Products
All 7 choose 5 ways to pick 5 products from 1,2,3,4,5,6,7	Schedule remaining four products using:	$[ad\|bc]$ where $a \leq b \leq c \leq d$
12348	59	67
12358	49	67
12458	39	67
13458	29	67
12349	58	67
12389	47	56

Table 18: Partitions of 9 products into set of 4, set of 3, and set of 2

Set of 4 Products	Set of 3 Products	Set of 2 Products
All Partitions for 7 products into a set of 4 and set of 3 using 1,2,3,4,5,6,7		89
All Partitions for 7 products into a set of 4 and set of 3 using 1,2,3,4,5,6,8		79
All Partitions for 7 products into a set of 4 and set of 3 using 1,2,3,4,5,7,8		69
All Partitions for 7 products into a set of 4 and set of 3 using 1,2,3,4,6,7,8		59
All Partitions for 7 products into a set of 4 and set of 3 using 1,2,3,5,6,7,8		49
All Partitions for 7 products into a set of 4 and set of 3 using 1,2,3,5,6,7,8		39
All Partitions for 7 products into a set of 4 and set of 3 using 1,3,4,5,6,7,8		29
All Partitions for 7 products into a set of 4 and set of 3 using 1,3,4,5,6,7,8		19
All Partitions for 7 products into a set of 4 and set of 3 using 1,2,3,4,5,6,9		78
All Partitions for 7 products into a set of 4 and set of 3 using 1,2,3,4,5,7,9 where 7 and 9 are not in the same period.		68
1239	467	58
1249	367	58
1267	349	58
1349	267	58
1367	249	58
1467	239	58
2349	167	58
2367	149	58
2467	139	58
3467	129	58
1239	567	48
1567	239	48
2567	139	48
3567	129	48
4567	129	38
All Possibilities in Table for [4:3]using 1,2,3,4,5,8,9 where 8 and 9 are not in the same period.		67

Table 19: Partitions of 8 products into three sets of 3

Set of 3 products	Set of 3 Products	Set of 3 Products
All Possibilities in Table for [3:3] for 2,3,4,5,6,7		189
All Possibilities in Table for [3:3] for 2,3,4,5,6,8		179
All Possibilities in Table for [3:3] for 2,3,4,5,7,8		169
All Possibilities in Table for [3:3] for 2,3,4,6,7,8		159
All Possibilities in Table for [3:3] for 2,3,5,6,7,8		149
All Possibilities in Table for [3:3] for 2,4,5,6,7,8		139
All Possibilities in Table for [3:3] for 3,4,5,6,7,8		129
All Possibilities in Table for [3:3] for 2,3,4,5,6,9		178
239	457	168
249	357	168
259	347	168
257	349	168
239	467	158
249	367	158
267	349	158
239	567	148
239	458	167
249	358	167
259	348	167
258	349	167

Table 20: $\eta_9 = 8$ using six basic periods

z_1	z_1	z_1	z_2	z_2	z_2
123	7	8	4	5	6
124	7	8	3	5	6
125	7	8	3	4	6
126	7	8	3	4	5
134	7	8	2	5	6
135	7	8	2	4	6
136	7	8	2	4	5
145	7	8	2	3	6
146	7	8	2	3	5
156	7	8	2	3	4
234	7	8	1	5	6
235	7	8	1	4	6
236	7	8	1	4	5
245	7	8	1	3	6
246	7	8	1	3	5
256	7	8	1	3	4
345	7	8	1	2	6
346	7	8	1	2	5
356	7	8	1	2	4

Table 20 Continued

456	7	8	1	2	3
6	7	8	123	4	5
16	23	8	4	5	7
17	23	8	4	5	6
15	24	8	3	6	7
16	24	8	3	5	7
16	25	8	3	4	6
15	34	8	2	6	7
16	34	8	2	5	7
16	35	8	2	4	7
17	35	8	2	4	6
17	36	8	2	4	5
16	45	8	2	3	7
17	45	8	2	3	6
17	46	8	2	3	5
17	56	8	2	3	4
25	34	8	1	6	7
26	34	8	1	5	7
27	34	8	1	5	6
26	35	8	1	4	7
27	35	8	1	4	6
27	36	8	1	4	5
26	45	8	1	3	7
27	45	8	1	3	6
27	46	8	1	3	5
27	56	8	1	3	4
36	45	8	1	2	7
37	45	8	1	2	6
37	46	8	1	2	5
37	56	8	1	2	4
47	56	8	1	2	3
6	7	8	14	23	5

Chapter 4

On the Lot Size Scheduling Problem, Part III: Stock Out Prevention and System Feasibility

Abstract: The Economic Lot Scheduling Problem (ELSP) is a classical scheduling problem with the objective of minimizing the long-run inventory and setup costs of a single machine, multi-product inventory system. Demand rates, production rates, and setup times are assumed to be deterministic. The problem has been extensively studied and methods exist to obtain cyclic schedules which minimize total cost. In order to apply a cyclic schedule without a stock out occurring, certain initial inventory levels are required. This paper considers the scenario where inventory levels are critically low and the cyclic schedule cannot be applied. Analytical methods are developed to determine: if the initial inventory is sufficient to prevent a stock out from occurring; the maximum time until a stock out will occur if a stock out is unavoidable; and the minimum time required to obtain the necessary inventory to resume the cyclic schedule if a stock out can be avoided.

1 Introduction:

A single machine, multi-product facility is assumed to be employing a cyclic schedule determined by an existing Economic Lot Scheduling Problem (ELSP) technique. In order to apply a cyclic schedule without a stock out occurring, it is necessary to have certain minimum initial inventory levels. The ELSP literature "universally" assumes that the inventory to apply the cyclic schedule is always available when, in practice, this is not necessarily the case (Gallego & Moon, 1996).

Inventory levels may become critically low due to variation in demand, a machine breakdown, or any other disruption which temporarily makes the machine unavailable. A stock out may prevent an entire production facility from operating. "Putting the line down" is viewed as an extremely adverse event and may well cost a manager his/her job (Hodgson, 1980). Therefore, if inventory levels are low, the principle concern of a manager changes from minimizing long run costs to stock out avoidance.

Regardless of the reason, management needs to determine if the current inventory is sufficient to avoid a stock out. If not, an interim schedule is required to delay a stock out until additional production capacity can be procured. It is assumed throughout the paper that the additional production capacity is sufficient to make avoiding a stock out and regaining the necessary inventory to apply the desired cyclic schedule trivial. Examples of additional production capacity include using another machine in the facility, overtime, and outsourcing. If the current inventory is sufficient to avoid a stock out, then an interim schedule is required to recover the necessary inventory to return to the cyclic schedule.

Section 2 of this paper provides an overview of the literature on the ELSP. In Section 3, the problem of delaying a stock out is addressed. Section 4 generalizes Section 3 to address the problem of recovering sufficient inventory to apply a cyclic schedule. In Section 5, a method is presented to determine the feasible region of all initial inventory levels for which a stock out can be avoided. Conclusions are discussed in Section 6.

2 Literature Review:

The Economic Lot Size Scheduling Problem was first formally posed by Rogers (1958). Rogers introduced the Independent Solution (IS) approach which determines optimal lot sizes for each product without regard to the fact that all the products must be produced on the same machine. Conflicts naturally arise when more than one product is scheduled to be produced at the same time. A heuristic is used to adjust lot sizes and startup times to resolve the conflicts. It is unclear how much this affects the total cost.

Bomberger (1966) introduced the Basic Period (BP) approach. The cycle length of each product is assumed be an integer multiple (the multiplier) of some fundamental cycle (the basic period). The lot sizes are assumed to be fixed and each product is assumed to have zero inventory when it begins production. A dynamic programming approach is shown to be effective for low to medium loads. Doll and Whybark (1973) developed an iterative procedure to determine the multipliers for the BP approach. Schweitzer and Silver (1983) demonstrated the need for a lower bound on the basic period to insure Doll and Whybark's solution is feasible. In general, determining if a feasible schedule exists for a given set of multipliers and BP is an NP-hard problem (Hsu, 1983). For additional references on the limitations and feasibility of BP solutions see Haessler and Hogue (1976) and Andres and

Emmons (1976). A survey and the extended BP approach are provided by Elmaghraby (1978).

The Common Cycle (CC) or Rotation Cycle (RC) approach was first introduced by Hanssmann (1962) and Elion (1962). The assumption is made that each product is produced once during a fixed time period, referred to as the rotation cycle or T^{RC}. This implies that the lot size for each product must be equal to the demand for the product over T^{RC}. The setup and inventory costs can then be minimized over T^{RC} (Jones and Inman, 1989). Feasible solutions are easy to obtain for a rotation cycle, but need not be optimal for the general problem. Jones and Inman (1989) demonstrated that the RC approach is near optimal for a wide range of realistic problems and suggest the approach be considered before attempting more complex methods.

Stankard and Gupta (1969) suggested "grouping" products into several categories. Group A consists of those products that would be economically advantageous to produce more frequently, and the rest are distributed to groups B, C, and D. The "groups" are scheduled rather than the individual products. The order in which each product is produced within a group is fixed. The schedule is optimized for group A, assuming equal lot sizes. In order for it to be feasible, the production times and setup times for groups B, C, and D must be less than the idle time between production runs of group A. Hodgson (1970) used a pesudo dynamic programing procedure which generalizes Stankard and Gupta's grouping procedure. The solution of the procedure may not be feasible, but a heuristic is developed to find a feasible solution. Boctor (1987) proposed a group-based partial enumeration procedure.

Maxwell (1964) introduced the Time Varying Lot Sizes (TVLS) approach which allows unequal lot sizes, but requires that the inventory of each product is zero when it begins production (zero-switch rule). Determining a solution requires minimizing a quadratic function subject to linear constraints over all possible schedules. The large number of possible schedules make the formulation impractical for large problems. Delporte and Thomas (1977) used a similar approach, but reduce the problem to a convex quadratic program by assuming the schedule and the cycle time are known. Both lot sizes and idle times are considered as variables. Delporte and Thomas then develop a heuristic to generate potential schedules and cycle times based off of an unequal lot sizes version of Doll and Whybark's (1973) and Haessler and Hogue's (1976) work.

Hodgson and Nuttle (1986) showed that the optimal lot sizes for a given schedule can be determined using linear programming provided that the lot sizes are assumed to be equal (the zero-switch rule is not required). Dobson (1987) showed that every production sequence can be made into a feasible schedule provided that $\sum_{j=1}^{n} d_j/p_j < 1$, where d_j is the demand rate for product j, p_j is the production rate for product j, and n is the number of products being produced on the machine. However, this may require arbitrarily large cycle times. Zipkin (1991) determined optimal idle and production times for a given schedule by formulating a parametric quadratic program and performing some "EOQ-like" calculations. When Dobson's heuristic (1987) is used to select a schedule and Zipkin's algorithm is used to determine optimal idle and production times, the result is a "plausible" schedule which yields a "near-optimal" solution.

Anderson (1990) showed that the problem of determining if the given initial inventory levels are sufficent to avoid a stock out is an NP-hard problem. He developed a heuristic that "works reasonably well on a large number of problems," but fails to find relatively simple solutions (i.e., production sequences with relatively few production runs) in some cases. Anderson stated that "the degree of effectivness of this type of procedure is hard to measure" and that "to fully test the effectivness of the procedure we would need to have a computational method to make exhaustive checks on the feasibility."

Gallego and Moon (1996) considered the Multiple Product Single Facility Stockout Avoidance Problem (SAP) and the related Weighted Stockout Problem (WSP). The objective of the SAP is to determine if a production schedule exists which will avoid a stock out until time T given the initial inventory level. The WSP assumes linear stock out penalties. The objective is to minimize the total cost (inventory, setup, and stock out costs) until time T. Gallego and Moon developed a mixed integer program which is applied to 3 and 4 product versions of both problems.

This paper addresses two critical issues associated with the ELSP. First, an analytic approach to the SAP is presented. A sequencing heuristic is combined with the SAP approach to maximize the time until a stock out can be avoided. The analytic (SAP) approach can be modified so that the inventory at time T is sufficient to apply a desired cyclic schedule (rather than zero for all products). The time required to recover a desired inventory is then minimized. Second, an analytical method is presented which determines the set of all initial inventory levels for a given set of parameters which are sufficient to avoid a stock out. This set is referred to as the feasible region. The method can make "exhaustive" checks on the feasibility of an initial inventory and will never fail to find a "simple" solution (Anderson, 1990).

3 Delaying a Stock Out:

Let n be the number of products scheduled on a single machine. Product j has demand rate d_j', production rate p_j', and setup time s_j'. The initial inventory vector is denoted by $\boldsymbol{I} = [i_1 \cdots i_n]$, where i_j denotes the initial inventory of product j. Without loss of generality, assume the machine is set up for product 1. Let $\boldsymbol{f} = [f_1 \cdots f_k]$ be the production run schedule, where f_a is the product produced in production run a, and let $\boldsymbol{t} = [t_1 \cdots t_n]$ be the corresponding production times. Furthermore, let T be the fixed time until additional production capacity becomes available. The SAP considered by Gallego and Moon (1996) is to determine if a production schedule, $\boldsymbol{f}$, exists which will avoid a stock out until time T given an initial inventory vector, $\boldsymbol{I}$. Section 3 provides an analytical approach to solve the SAP.

In Section 3.1, preliminary observations are presented. Section 3.2 develops a feasibility condition on the total production and setup time of a schedule. In Section 3.3, an algorithm is presented to determine (if one exists) a feasible vector of production times, $\boldsymbol{t}$, to avoid a stock out until time T, given a schedule, $\boldsymbol{f}$, and an initial inventory level, $\boldsymbol{I}$. Section 3.4 describes an analytical alternative to Gallego and Moon's (1996) MIP approach to solve the SAP. Section 3.5 shows how to determine the maximum time a schedule, $\boldsymbol{f}$, and an initial inventory level, $\boldsymbol{I}$, can avoid a stock out. In Section 3.6, a search procedure to find the maximum time a stock out can be avoided over any production schedule, $\boldsymbol{f}$, for a given $\boldsymbol{I}$ is discussed. A heuristic for large problems that is based on the procedure described in Section 3.6 is presented in Section 3.7.

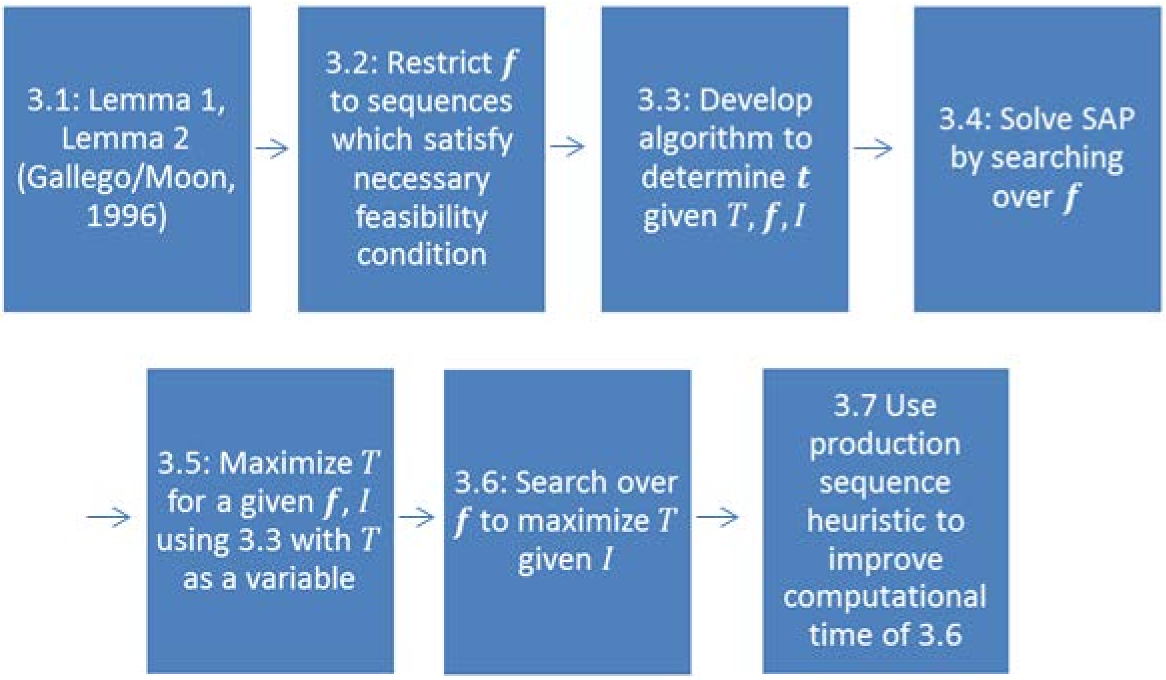

Figure 1: Section 3 (Diagrammatically)

3.1 Preliminaries:

A common transformation is to divide the demand rate, production rate, and initial inventory of each product j by the product's demand rate $d_j{}'$. Formally, the transformation is defined by $p_j = p_j'/d_j{}'$, $d_j = d_j'/d_j{}' = 1$, and $s_j = s_j'/d_j{}'$ for $j = 1, \dots, n$. This results in the inventory levels being expressed in time units. For example, if $i_1 = 10$ hours, then there is sufficient inventory for product 1 to satisfy demand for 10 hours. Note that this transformation results in a demand rate of 1 for each product. This reduces the dimensionality of the problem. Without loss of generality, assume that $d_j = 1$ for all j for the remainder of the paper. Additionally, if a product has sufficient inventory to satisfy demand until time T, it does not need to be produced. Therefore, we assume $i_j < T$ for every $j = 1, \dots, n$ throughout the paper.

Several observations of Gallego and Moon (1996) are critical for development of this paper and are summarized in this section.

Lemma 1. (Gallego and Moon, 1996) If it is possible to avoid a stock out until time T, then a production sequence $\boldsymbol{f}$ and production times $\boldsymbol{t}$ exist such that:

(i) The machine is not idle, except possibly after the last production run.

(ii) The inventory of each product at time T is 0.

(iii) At the beginning of each production run, the inventory of the product being produced is zero with the (possible) exception of the first production run of each product.

Lemma 2. (Gallego and Moon, 1996) Index the products so that the setup times are in non-decreasing order, $s_1 \leq s_2 \leq \cdots \leq s_n$. The number of "positions" (i.e., the number of production runs of individual products) of any production sequence used to solve the SAP is at most $N = \max(N_1, N_2)$, where

$$N_1 = 2\left\lfloor \frac{\left(1 - \sum_{j=1}^{n} \frac{1}{p_j}\right) T + \sum_{j=1}^{n} \frac{i_j}{p_j} - \sum_{j=3}^{n} s_j}{(s_1 + s_2)} \right\rfloor + n - 2 \tag{1}$$

$$N_2 = 2\left\lfloor \frac{\left(1 - \sum_{j=1}^{n} \frac{1}{p_j}\right) T + \sum_{j=1}^{n} \frac{i_j}{p_j} - \sum_{j=3}^{n} s_j + s_2}{(s_1 + s_2)} \right\rfloor + n - 3 \tag{2}$$

Note that the assumption that each product has 0 inventory at time T is useful to determine if a given production sequence can be used to avoid a stock out before time T.

3.2 Feasibility Condition on the Total Production Plus Setup Time:

The total production plus setup time can be used to eliminate some production sequences as candidates to avoid a stock out. Define the total production time of product j in a sequence $\boldsymbol{f}$ to be

$$x_j = \sum_{\forall a, f_a = j} t_a. \tag{3}$$

where a represents positions in the sequence.

Note that the demand for product j during $[0, T]$ is $d_j T = (1)T = T$, and, by Lemma 1(ii), the ending inventory of product j at time T is zero. The ending inventory is equal to the starting inventory plus production minus demand. Therefore, $i_j + p_j x_j - T = 0$. Consequently,

$$x_j = \frac{T - i_j}{p_j}. \tag{4}$$

In order for a production sequence with k positions to be used to avoid a stock out until time T, it is necessary for the sum of the total production times and the setup times to be less than or equal to T. Therefore, only production sequences where

$$\sum_{j=1}^{n} \frac{T - i_j}{p_j} + \sum_{a=2}^{k} s_{f_a} \leq T \tag{5}$$

need be considered. If the machine has sufficient capacity to produce all n products, then $\sum_{j=1}^{n}(d_j/p_j) = \sum_{j=1}^{n}(1/p_j) < 1$. If this is the case, then (5) is equivalent to

$$\frac{\left(\sum_{a=2}^{k} s_{f_a} - \sum_{j=1}^{n} \frac{i_j}{p_j}\right)}{\left(1 - \sum_{j=1}^{n} \frac{1}{p_j}\right)} \leq T. \tag{6}$$

3.3 Procedure to Determine t Given T, f, and I:

Once the total production times have been calculated using Equation 4 for a schedule $\boldsymbol{f}$ that satisfies Equation 6, the remaining task is to determine if a $\boldsymbol{t} = [t_1 \cdots t_k]$ exists which will avoid a stock out until time T. In this section, an algorithm is presented that determines a solution for $\boldsymbol{t}$ if one exists (summarized below). The solution may not be unique.

1. Check if

$$\frac{\left(\sum_{a=2}^{k} s_{f_a} - \sum_{j=1}^{n} \frac{i_j}{p_j}\right)}{\left(1 - \sum_{j=1}^{n} \frac{1}{p_j}\right)} \leq T.$$

If the inequality does not hold, STOP. Schedule $\boldsymbol{f}$ cannot avoid a stock out until time T starting with inventory level $\boldsymbol{I}$. Otherwise, continue to Step 2.

2. Initialize $v = k$ and $z_j^k = \frac{T-i_j}{p_j},\ j = 1,2,\dots,n$
3. If $v = m_{f_v}$, then $t_v = z_{f_v}^v$. Otherwise, $t_v = z_{f_v}^v - \left(\frac{\sum_{j=1,j\neq f_v}^{n} z_j^v + \sum_{h=2}^{v} s_{f_h} - i_{f_v}}{p_{f_v}-1}\right)$.
4. If $v - 1 = 0$, set $v = 0$ and go to Step 6. Otherwise, update z_j^{v-1} for $j = 1,2,\dots,n$ as follows:

$$z_j^{v-1} = \begin{cases} z_j^v - t_v, & f_v = j \\ z_j^v, & f_v \neq j \end{cases}$$

5. Set $v = v - 1$. Go to Step 3.
6. Calculate τ_{m_j} using Equation 13 for $j = 2,\dots,n$. If $i_j \geq \tau_{m_j}$ for all $j = 2,\dots,n$, then $\boldsymbol{t}$ is a feasible vector of production times that avoids a stock out until time T for schedule $\boldsymbol{f}$ and inventory level $\boldsymbol{I}$. Otherwise, schedule $\boldsymbol{f}$ cannot avoid a stock out until time T starting with inventory level $\boldsymbol{I}$.

Several variables are introduced to aid in the development of the algorithm. The variable y_j^v is the sum of the production times of product j (strictly) before position v, and z_j^v is the sum of the production times of product j up to and including position v. Formally, let

$$y_j^v = \sum_{\forall a, a<v, f_a=j} t_a\ ,\ j = 1,2,\dots,n \text{ and } v = 1,2,\dots,k \tag{7}$$

and

$$z_j^v = \sum_{\forall a, a\leq v, f_a=j} t_a\ ,\quad j = 1,2,\dots,n \text{ and } v = 1,2,\dots,k. \tag{8}$$

Then, each of the production times can be calculated by

$$t_v = z_{f_v}^v - y_{f_v}^v\ ,\quad v = 1,2,\dots,k. \tag{9}$$

Clearly,

$$z_j^k = x_j = \frac{T - i_j}{p_j},\qquad j = 1,2,\dots,n. \tag{10}$$

If product j is not produced in production run v, then the sum of the production times of product j up to and including run v and run $v-1$ are the same. If product j is produced in production run v, then the sum of production times of product j up to and including run v is t_v more than for run $v-1$. Therefore, z_j^v can be updated as follows:

$$z_j^{v-1} = \begin{cases} z_j^v - t_v, & f_v = j \\ z_j^v, & f_v \neq j \end{cases}, \qquad j = 1,2,\dots,n \text{ and } v = 2,3,\dots,k. \tag{11}$$

In order to calculate t_v for $v = 1,2,\dots,k$, the y_j^v's must also be determined. If run v is the first production run of product f_v, then $y_{f_v}^v = 0$. If v is not the first production run of product f_v, then its inventory at the start of production run v is 0 (by Lemma 1 (iii)). Let τ_v denote the start time of production run v for $v = 1,\dots,k$. In order to transition from i_{f_v} inventory at time 0 to 0 inventory at time τ_v, $\tau_v - i_{f_v}$ product f_v must be produced before time τ_v. Thus,

$$y_{f_v}^v = \left(\tau_v - i_{f_v}\right)/p_{f_v}. \tag{12}$$

Recall that there is no idle time before the last production run (Lemma 1 (i)). Therefore, the time production run v starts is the sum of the first $v-1$ production runs and the setup times. That is,

$$\tau_v = t_1 + s_{f_2} + t_2 + s_{f_3} + \cdots + t_{v-1} + s_{f_v} \tag{13a}$$

The production times can be grouped by product and the total production time of product j before production run v is simply y_j^v. Furthermore, z_j^v is equal to y_j^v except for product f_v. Therefore,

$$\tau_v = y_{f_v}^v + \sum_{j=1, j\neq f_v}^{n} y_j^v + \sum_{h=2}^{v} s_{f_h} \tag{13b}$$

$$= y_{f_v}^v + \sum_{j=1, j\neq f_v}^{n} z_j^v + \sum_{h=2}^{v} s_{f_h}. \tag{13c}$$

Solving Equation 12 for τ_v and setting the result equal to Equation 13c yields

$$y_{f_v}^v = \frac{\sum_{j=1, j\neq f_v}^{n} z_j^v + \sum_{h=2}^{v} s_{f_h} - i_{f_v}}{p_{f_v} - 1}. \tag{14}$$

The production times can now be calculated using equation (9). The only remaining step is to determine if $(\boldsymbol{f}, \boldsymbol{t})$ prevents a stock out until time T. To avoid a stock out, the first production run of each product must occur before the initial inventory is depleted. This is equivalent to $i_j \geq \tau_{m_j}$ for each $j = 2,\dots,n$, where m_j is the first production run of product j

in $\boldsymbol{f}$. The construction of $\boldsymbol{t}$ insures each product will have 0 inventory at the start of each subsequent production run and at time T. Therefore, if a product does not have a stock out before its first production run, it will not stock out before time T.

3.4 Analytical Alternative to Solve the SAP:

To solve the SAP, first Lemma 2 is used to calculate N, the maximum number of positions of any production sequence that solves the SAP. Then, a production sequence of maximum length N is investigated using the algorithm described in Section 3.3. If a feasible $\boldsymbol{t}$ is found for the production sequence $\boldsymbol{f}$ currently being investigated, then $\boldsymbol{f}$ is a solution to the SAP. Otherwise, investigate another production sequence of maximum length N. Continue searching until a solution is found or all production sequence of maximum length N have been investigated.

A problem presented by Anderson (1990) is used as an example for the analytical approach to the SAP and throughout the paper. The problem is shown in Table 1.

Table 1: Data for Anderson's (1990) Example

Product	Demand Rate	Production Rate	Setup Time	Initial Inventory
1	1	6	3	14
2	1	2	1	11
3	1	10	5	27

The search described in Section 3.4 is applied to the problem in Table 1 with $T = 70$ and the production sequences being searched in order of increasing number of positions. First $\boldsymbol{f} = [1\ 2\ 3]$ is investigated, and $\boldsymbol{f} = [1\ 2\ 3\ 1]$ is investigated second. Both of these sequences cannot avoid a stock out until time 70. The third sequence investigated is $\boldsymbol{f} = [1\ 2\ 3\ 2]$. The algorithm in Section 3.3 is applied for this sequence as follows.

Steps 1 and 2: $-15.14 \leq 70$ $v = 4, z_1^4 = 9.33, z_2^4 = 29.5, z_3^4 = 4.3$

Steps 3-5: $t_4 = 29.5 - 9.63 = 19.87, z_1^3 = 9.33, z_2^3 = 9.63, z_3^3 = 4.3, v = 3$

Steps 3-5: $t_3 = 4.3, z_1^2 = 9.33, z_2^2 = 9.63, z_3^2 = 0, v = 2$

Steps 3-5: $t_2 = 9.63, z_1^1 = 9.33, z_2^1 = 0, z_3^1 = 0, v = 1$

Steps 3-4: $t_1 = 9.33, v = 0$

Step 6: $\tau_{m_2} = 10.33 \leq 11, \tau_{m_3} = 24.96 \leq 27$

Consequently, $\boldsymbol{f} = [1\ 2\ 3\ 2]$ avoids a stock out until time 70 with $\boldsymbol{t} = [9.33\ 9.63\ 4.30\ 19.87]$.

3.5 Algorithm to Determine Maximum T given f and I:

The algorithm in Section 3.3 can be modified with T as a variable to determine the maximum time a schedule, $\boldsymbol{f}$, and an inventory level, $\boldsymbol{I}$, can avoid a stock out. The result is a linear expression for each τ_j in terms of T. The first production run of each product must occur before the inventory is depleted, which results in a system of inequalities, which can be used to obtain an upper bound for T. For example, consider Anderson's (1990) problem with and $\boldsymbol{f} = [1\ 2\ 1\ 3]$. The total production times are found as follows.

Step 2: $v = 4, z_1^4 = \frac{T-14}{6}, z_2^4 = \frac{T-11}{2},\ z_3^4 = \frac{T-27}{10}$

Steps 3-5: $t_4 = \frac{T-27}{10},\ z_1^3 = \frac{T-14}{6}, z_2^3 = \frac{T-11}{2},\ z_3^3 = 0, v = 3$

Steps 3-5: $t_3 = \frac{T-14}{6} - \left(\frac{T-31}{10}\right) = \frac{2T+23}{30},\ z_1^2 = \frac{T-31}{10}, z_2^2 = \frac{T-11}{2},\ z_3^2 = 0, v = 2$

Steps 3-5: $t_2 = \frac{T-11}{2},\ z_1^1 = \frac{T-31}{10}, z_2^1 = 0\ \ z_3^1 = 0, v = 1$

Steps 3-4: $t_1 = \frac{T-31}{10},\ v = 0$

Step 6: $\tau_{m_2} = t_1 + s_2 = \frac{T-31}{10} + 1 \leq 11 \Rightarrow T \leq 131$ (Product 2 stock out constraint)

$$\tau_{m_3} = t_1 + s_2 + t_2 + s_1 + t_3 + s_3 = \frac{T-31}{10} + 1 + \frac{T-11}{2} + 3 + \frac{2T+23}{30} + 5 \leq 27$$

$$\Rightarrow T \leq 38.75 \text{ (Product 3 stock out constraint)}$$

In order for product 2 not to stock out before its first production run, T can be at most 131. Similarly, in order for product 3 not to stock out before its first production run, T can be at most 38.75. Therefore, the maximum time until $\boldsymbol{f}$ can prevent a stock out from occurring is 38.75 and is obtained by $\boldsymbol{t} = [0.775\ 13.875\ 3.35\ 1.175]$.

3.6 Search Procedure to Determine Maximum T given I:

Recall that the upper bound on the number of positions, N, is dependent upon T. However, with some care it is possible to adapt the search described in Section 3.4 to incorporate the algorithm of Section 3.5. This procedure is as follows:

1. Set T equal to the maximum time the rotation production sequence $[1 \cdots n]$ can prevent a stock out using the algorithm in Section 3.5 and determine N from Lemma 2 and let $Q = 0$.If $v = m_{f_v}$, then $t_v = z_{f_v}^v$. Otherwise, $t_v = z_{f_v}^v - \left(\frac{\sum_{j=1, j\neq f_v}^{n} z_j^v + \sum_{h=2}^{v} s_{f_h} - i_{f_v}}{p_{f_v} - 1}\right)$.
2. Search all production sequences of length $Q + 1$ up to and including length N and use the algorithm described in Section 3.5 to compute the maximum time a stock out can be delayed. Call the largest of these maximum times T^{max}. If $T^{max} > T$, then go to Step 3. Otherwise, the maximum time a stock out can be avoided using any schedule is T.
3. Set T equal T^{max} and recalculate N. If the recalculated N is greater than the previously calculated N, then go to Step 2. Otherwise, the maximum time a stock out can be avoided using any schedule is T.

3.7 Heuristic to Determine Maximum T given I:

Searching over all possible production sequences with at most N positions can be computationally prohibitive for large problems. If a solution exists, it may be found quickly. However, the only known way to verify if a solution does not exist is to consider all possible production sequences of length at most N. See Table A1 in the Appendix A for a summary of computational times if all production sequences need to be searched. If $n > 4$ or for large values of N, a heuristic may be appropriate.

The heuristic developed in this section employs a partial enumeration strategy. The production sequences are searched by increasing number of positions. The partial enumeration starts with the single sequence with n positions, $[1 \cdots n]$. Suppose that production sequences of length k have been considered. The production sequences of length k+1 which prevent a stock out the longest are likely to either start with (i) one of the production sequences of length k which prevented a stock out the longest or (ii) a production sequence of length k for products $1, \ldots, n-1$ only (i.e. in which product n has not yet been considered). The partial enumeration restricts the first k positions of a schedule of length $k+1$ to the M production sequences of length k which can avoid a stock out the longest and all production sequences of length k for products $1, \ldots, n-1$. Increasing M improves the accuracy of the heuristic, but requires additional computational time.

The bounds in Equations (1) and (2) can be modified with $n = n - 1$ and $T = i_n - s_n$ to determine the maximum number of positions for a production sequence of products $1, \ldots, n-1$ which can avoid a stock out until product n must be setup. Therefore, it is only necessary to consider production sequences of this form if $k \leq N'$ where $N' = \max(N_1', N_2')$ and $N_1', N_2{}'$ are defined as:

$$N_1{}' = 2\left\lfloor \frac{\left(1 - \sum_{j=1}^{n-1} \frac{1}{p_j}\right)(i_n - s_n) + \sum_{j=1}^{n-1} \frac{i_j}{p_j} - \sum_{j=3}^{n-1} s_j}{(s_1 + s_2)} \right\rfloor + n - 3, \tag{15}$$

$$N_2{}' = 2\left\lfloor \frac{\left(1 - \sum_{j=1}^{n-1} \frac{1}{p_j}\right)(i_n - s_n) + \sum_{j=1}^{n-1} \frac{i_j}{p_j} - \sum_{j=3}^{n-1} s_j + s_2}{(s_1 + s_2)} \right\rfloor + n - 4. \tag{16}$$

To illustrate the process, a modified version of Anderson's (1990) problem is considered. The modification is necessary because Anderson (1990) showed his data can avoid a stock out over the infinite horizon. Let $P_j' = .9P_j$ where P_j' is the modified production rate and P_i is the production rate from Anderson's (1990) problem. Product 1 is assumed to be produced first (recall that the machine is setup for product 1). The remaining products are assumed to first appear in order of increasing inventory.

Before implementing the heuristic, a choice of M must be made. Let $M = 5$ for this example. Define T_b to be the set of candidate schedules with at most b positions. The heuristic is not relevant until there are (strictly) greater than M production sequences to consider. As there is only one schedule with three positions and three schedules with four positions, T_3 and T_4 are simply the set of all schedules with at most three and four positions, respectively.

$$T_3 = \{[1\ 2\ 3]\}, \tag{17a}$$

$$T_4 = \{[1\ 2\ 3\ 2], [1\ 2\ 3\ 1], [1\ 2\ 3], [1\ 2\ 1\ 3]\}, \tag{17b}$$

The production sequences of length 5 to be searched start with a 4-position sequence in T_4 or a 4-position sequence for products 1 and 2 (i.e., $[1\ 2\ 1\ 2]$). Therefore, the 5-position sequences which need to be evaluated are $[1\ 2\ 3\ 2\ 1], [1\ 2\ 3\ 2\ 3], [1\ 2\ 3\ 1\ 2\ ], [1\ 2\ 3\ 1\ 3], [1\ 2\ 1\ 3\ 1]$, $[1\ 2\ 1\ 3\ 2]$, and $[1\ 2\ 1\ 2\ 1\ ]$. The 5 production sequences which can avoid a stock out the longest among the 5-position sequences that need to be evaluated and those in T_4 constitute T_5 and are as follows:

$$T_5 = \{[1\ 2\ 3\ 2\ 1], [1\ 2\ 3\ 1\ 2], [1\ 2\ 3\ 2\ 3], [1\ 2\ 3\ 2], [1\ 2\ 1\ 3\ 2]\}. \tag{17c}$$

The heuristic implies that the production sequences of length 6 to be searched start with a 5-position sequence in T_5 or a 5-position sequence for products 1 and 2 (i.e. $[1\ 2\ 1\ 2\ 1]$). Therefore, the 6-position sequences which need to be evaluated are $[1\ 2\ 3\ 2\ 1\ 2], [1\ 2\ 3\ 2\ 1\ 3]$, $[1\ 2\ 3\ 1\ 2\ 1], [1\ 2\ 3\ 1\ 2\ 3], [1\ 2\ 3\ 2\ 3\ 1], [1\ 2\ 3\ 2\ 3\ 2], [1\ 2\ 1\ 3\ 2\ 1], [1\ 2\ 1\ 3\ 2\ 3]$, and $[1\ 2\ 1\ 2\ 1\ 3]$. The 5 production sequences from this list which can avoid a stock out the longest constitute T_6 and are as follows:

$$T_6 = \{[1\ 2\ 3\ 2\ 1\ 2], [1\ 2\ 3\ 2\ 1\ 3], [1\ 2\ 3\ 2\ 1], [1\ 2\ 3\ 1\ 2\ 1], [1\ 2\ 3\ 2\ 3\ 1]. \tag{18}$$

The method can be repeated as long as desired or until $T_m = T_{m+1}$.

When the method was applied to Anderson's (1990) modified problem, the optimal solution ([1 2 3 2 1 2 3 2 1 2 3 2 1 2 1 2 1 2]; 191.02 units time) was found using $M = 1$ in 0.09 seconds. If $M = 10$, the optimal solution was found in 0.39 seconds, and if $M = 100$, the optimal solution was found in 2.8 seconds. Additional times are provided in the Table 2. In comparison, searching all schedules requires 274 seconds.

Table 2: Computational Times for Heuristic (seconds)

	Max # of Positions				
M	**15**	**18**	**20**	**25**	**30**
10	0.30	0.39	0.41	0.59	0.70
100	1.80	2.80	3.09	4.42	5.75

4 Inventory Recovery:

If a stock out can be avoided, an interim schedule is required to recover sufficient inventory to apply the cyclic schedule. Management would like to be able to re-implement the desired cyclic schedule as soon as possible. In this section, we determine a production sequence, $\boldsymbol{f}$, which minimizes the time, T, at which an inventory vector, $\boldsymbol{U} = [u_1 \cdots u_n]$, sufficient to apply a desired cyclic schedule, can be obtained given initial inventory $\boldsymbol{I}$. First, a method is presented in Section 4.1 which determines if a feasible vector of production times, $\boldsymbol{t}$, exists for a fixed schedule, $\boldsymbol{f}$, which results in inventory $\boldsymbol{U}$ at time T given an initial inventory level, $\boldsymbol{I}$. A search procedure, based on Section 4.1, over all possible production sequences is presented in Section 4.2. Finally, the method from Section 4.1 and the search procedure from Section 4.2 are combined in Section 4.3 to obtain a procedure which determines the minimum time T that results in inventory $\boldsymbol{U}$ at T, over all $\boldsymbol{f}$ given the initial inventory level is $\boldsymbol{I}$.

4.1 Method to Determine $\boldsymbol{t}$ given $\boldsymbol{T}$, $\boldsymbol{f}$, $\boldsymbol{I}$, and $\boldsymbol{U}$:

The algorithm presented in Section 3.3 can be modified to determine if it is possible to attain inventory $\boldsymbol{U}$ at time T by applying a given schedule $\boldsymbol{f}$ given an initial inventory level of $\boldsymbol{I}$. In order to have u_j of product j at time T (instead of zero), the total production time of product j needs to be increased by u_j/p_j. Therefore, the total production time of product j is now $(T + u_j - i_j)/p_j$. Hence, the following 2 changes are necessary to the algorithm in Section

3.3 to determine if schedule $\boldsymbol{f}$ can yield an inventory level of $\boldsymbol{U}$ at time T when the initial inventory is $\boldsymbol{I}$:

(1) In Step 1, the feasibility check is now $\frac{\left(\sum_{a=2}^{k} s_{f_a} - \sum_{j=1}^{n} \frac{i_j - u_j}{p_j}\right)}{\left(1 - \sum_{j=1}^{n} \frac{1}{p_j}\right)} \leq T$.

(2) In Step 2, initialize $z_j^k = \frac{T + u_j - i_j}{p_j}$, $j = 1,2,\dots,n$ instead of $z_j^k = \frac{T - i_j}{p_j}$.

4.2 Search to Determine f and t given T, I, and U:

Here we wish to determine if a production sequence exists which results in inventory $\boldsymbol{U}$ being obtained at time T given an initial inventory level $\boldsymbol{I}$. The approach requires production sequences of at most $N'' = \max(N_1'', N_2'')$ positions to be searched where

$$N_1'' = 2\left\lfloor \frac{\left(1 - \sum_{j=1}^{n} \frac{1}{p_j}\right) T + \sum_{j=1}^{n} \frac{i_j - u_j}{p_j} - \sum_{j=3}^{n} s_j}{(s_1 + s_2)} \right\rfloor + n - 2, \tag{19}$$

$$N_2'' = 2\left\lfloor \frac{\left(1 - \sum_{j=1}^{n} \frac{1}{p_j}\right) T + \sum_{j=1}^{n} \frac{i_j - u_j}{p_j} - \sum_{j=3}^{n} s_j + s_2}{(s_1 + s_2)} \right\rfloor + n - 3. \tag{20}$$

The search is performed in the same manner as the search in Section 3.4, except that the modification of the algorithm in Section 3.3 that is presented in Section 4.1 is used instead of the algorithm in Section 3.3, and production sequences of maximum length N'' are searched instead of maximum length N.

4.3 Determining the Minimum T Given I and U:

The minimum time required to obtain inventory $\boldsymbol{U}$ without a stock out occurring with initial inventory $\boldsymbol{I}$ is now determined. Without loss of generality, assume that all products need to be produced at least once. Idle time represents time not spent producing or setting up the machine and does not aid in recovering inventory in any way. Therefore, assume the machine is never idle. The (unique) time required to obtain inventory $\boldsymbol{U}$ by applying production sequence $\boldsymbol{f}$ with initial inventory $\boldsymbol{I}$ under these assumptions is

$$T' = \sum_{j=1}^{n} x_j + \sum_{a=2}^{k} s_{f_a} = \sum_{j=1}^{n} \frac{T' + u_j - i_j}{p_j} + \sum_{a=2}^{k} s_{f_a}. \tag{21}$$

Therefore,

$$T' = \left(\frac{1}{1 - \sum_{j=1}^{n} \frac{1}{p_j}} \right) \left(\sum_{a=2}^{k} s_{f_a} + \sum_{j=1}^{n} \frac{u_j - i_j}{p_j} \right). \tag{22}$$

Note that the only term dependent upon the production sequence in Equation 22 is $\sum_{a=1}^{k} s_{f_a}$. Let S_f be the total setup time of sequence $\boldsymbol{f}$, i.e. $S_f = \sum_{a=1}^{k} s_{f_a}$. The right-hand side of (22) is a linear equation in terms of S_f with a positive slope (as $\sum_{j=1}^{n} 1/p_j < 1$). Therefore, the production sequence with the smallest value of S_f which can obtain $\boldsymbol{U}$ with initial inventory $\boldsymbol{I}$ minimizes the recovery time over all production sequences. To determine the minimum time required to obtain inventory $\boldsymbol{U}$, search the production sequences in order of increasing setup time until the first one is found which can obtain $\boldsymbol{U}$ with initial inventory $\boldsymbol{I}$. For each sequence $\boldsymbol{f}$, the algorithm developed in Section 4.1 can be applied with $T = T'$ to determine if it is possible to obtain $\boldsymbol{U}$ by applying $\boldsymbol{f}$ with initial inventory $\boldsymbol{I}$ without a stock out occurring. If it is possible, then T' is the minimum time required to obtain $\boldsymbol{U}$ using sequence $\boldsymbol{f}$.

5 The Feasible Region:

Recall an inventory vector $\boldsymbol{I}$ is feasible if it is possible to start with inventory vector $\boldsymbol{I}$ and prevent a stock out over the infinite time horizon, and the feasible region is the set of all feasible inventory vectors. Traditionally, to show an inventory level vector, $\boldsymbol{I}$, is feasible, a production schedule is found which, when applied to $\boldsymbol{I}$, has an ending inventory $\boldsymbol{I}' \geq \boldsymbol{I}$. An alternative method is used in this section. Suppose an inventory vector $\boldsymbol{J}$ is known to be feasible. If it is possible to start with inventory vector $\boldsymbol{I}$ and obtain inventory vector $\boldsymbol{J}$, then $\boldsymbol{I}$ is also feasible. Furthermore, if $\boldsymbol{I}$ is in the interior of the feasible region, then it is possible to obtain any desired inventory vector (given sufficient time); including $\boldsymbol{J}$. Therefore, the (interior) of the feasible region is equivalent to the set of all initial inventory vectors which can obtain a fixed feasible inventory vector $\boldsymbol{J}$.

A method is now presented to obtain the set of all initial inventory vectors which can obtain inventory vector $\boldsymbol{J}$ (without a stock out occurring) using a particular fixed production sequence $\boldsymbol{f}$. If it is possible to start with an initial inventory vector, $\boldsymbol{I}$, apply $\boldsymbol{f}$, and later obtain $\boldsymbol{J}$ with any idle time in the schedule, then it is possible to start with some inventory vector $\boldsymbol{I}'' \leq \boldsymbol{I}, \boldsymbol{I}'' \neq \boldsymbol{I}$. Therefore, if an initial inventory vector is on the boundary of the set of all initial inventory vectors which obtain inventory vector $\boldsymbol{J}$ using production sequence $\boldsymbol{f}$, then there is no idle time in the schedule. Therefore, the total time required to apply $\boldsymbol{f}$, i.e., T, is the sum of the setup times in $\boldsymbol{f}$ and the total production times of all products. Thus, T is not treated as a variable in Section 5.

The algorithm in Section 3.3 is now generalized where T is constant and the initial inventory levels, $i_j, j = 1, \ldots n$ are variables. The total time required to implement $\boldsymbol{f}$ is $x_1 + \cdots + x_n + S_f$. In order to end with inventory u_j, the initial inventory plus production needs to equal u_j plus demand:

$$i_j + p_j x_j = u_j + \left(x_1 + \cdots + x_n + S_f\right), \text{ for } j = 1, \ldots, n. \tag{23}$$

Note that each i_j can be represented as a function of the x_j's. This yields a system of n equations which can be solved for $x_1, \ldots, x_n$. The system is $Ax = b$ where

$$A = \begin{bmatrix} p_1 - 1 & -1 & \cdots & -1 \\ -1 & p_2 - 1 & & \vdots \\ \vdots & & \ddots & -1 \\ -1 & \cdots & -1 & p_n - 1 \end{bmatrix}, \quad x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \quad b = \left[u_j + S_f - i_j\right]_{n \times 1}.$$

To reduce this matrix subtract row 1 from all other rows.

$$[A|b] = \left(\begin{array}{cccc|c} p_1 - 1 & -1 & \cdots & -1 & u_1 + S_f - i_1 \\ -p_1 & p_2 & & \vdots & u_2 - i_2 - u_1 + i_1 \\ \vdots & & \ddots & 0 & \vdots \\ -p_1 & \cdots & 0 & p_n & u_n - i_n - u_1 + i_1 \end{array}\right) \tag{24a}$$

Let R_v denote row v. Now let $R_1 = R_1 + \frac{1}{p_2} R_2 + \cdots + \frac{1}{p_n} R_n$.

$$[A|b] = \left(\begin{array}{cccc|c} p_1(1-\sum_{\forall j} 1/p_j) & 0 & \cdots & 0 & S_f + (u_1 - i_1) + \sum_{j=2,..,n} \frac{u_j - i_j - (u_1 - i_1)}{p_j} \\ -p_1 & p_2 & & \vdots & u_2 - i_2 - u_1 + i_1 \\ \vdots & & \ddots & 0 & \vdots \\ -p_1 & \cdots & 0 & p_n & u_n - i_n - u_1 + i_1 \end{array}\right) \quad (24b)$$

Finally, let $R_i = R_i + \frac{1}{(1-\sum_{\forall j} 1/p_j)} R_1$ for $i = 2, \dots, n$ and

$$c = S_f + (u_1 - i_1)\left(1 - \sum_{j=2,..,n} \frac{1}{p_j}\right) + \sum_{j=2,..,n} \frac{u_j - s_j}{p_j}. \quad (24c)$$

$$[A|b] = \left(\begin{array}{cccc|c} p_1(1-\sum_{\forall j} 1/p_j) & 0 & \cdots & 0 & c \\ 0 & p_2 & & \vdots & (u_2 - i_2 - u_1 + i_1) + \frac{c}{(1-\sum_{\forall j} 1/p_j)} \\ \vdots & & \ddots & 0 & \vdots \\ 0 & \cdots & 0 & p_n & (u_n - i_n - u_1 + i_1) + \frac{c}{(1-\sum_{\forall j} 1/p_j)} \end{array}\right) \quad (24d)$$

Therefore,

$$x_1 = \frac{c}{p_1(1-\sum_{\forall j} 1/p_j)} = \frac{S_f + (u_1 - i_1) + \sum_{j=2,..,n} \frac{u_j - i_j - (u_1 - i_1)}{p_j}}{p_1(1-\sum_{\forall j} 1/p_j)} \quad (24e)$$

$$x_j = \frac{1}{p_j}\left(u_j - i_j - u_1 + i_1 + \frac{c}{(1-\sum_{\forall j} 1/p_j)}\right) = \frac{(u_j - i_j - u_1 + i_1)}{p_j} + \frac{p_1 x_1}{p_j} \text{ for } j = 2, \dots, n. \quad (24f)$$

The algorithm in Section 3.3 can now be applied to calculate $\boldsymbol{t}$ if we start with Step 2 and initialize z_j^k in Step 2 to the corresponding x_j values calculated in Equations 24e and 24f, for $j = 1,2, \dots, n$.

For example, consider the 3 product problem where $p_j = 30, d_j = 8, s_j = 1$ for $j = 1,2,3$. The problem can be transformed to $p_j = 3.75, d_j = 1, s_j = 1$ for $j = 1,2,3$. Consider the sequence $\boldsymbol{f} = [1\ 2\ 1\ 3]$ and $\boldsymbol{U} = [0\ 5\ 10]$. Any feasible inventory is valid choice for $\boldsymbol{U}$. In this case, $\boldsymbol{U}$ is sufficient inventory to return to the rotation schedule [1 2 3] with $\mathbf{t} = [\mathbf{4\ 4\ 4}]$. The total production times are:

$$x_1 = -.6222i_1 - .3556i_2 - .3556i_3 + 10.6667 \quad (25a)$$

$$x_2 = -.3556i_1 - .6222i_2 - .3556i_3 + 12.0000 \quad (25b)$$

$$x_3 = -.3556i_1 - .3556i_2 - .6222i_3 + 13.3333 \quad (25c)$$

The results of applying the algorithm in Section 3.3 with the changes noted in this section for $\boldsymbol{f} = [1\ 2\ 1\ 3]$ and $\boldsymbol{U} = [0\ 5\ 10]$ are as follows (see Appendix B for details):

$$t_1 = -.4929i_1 - .2263i_2 - .1293i_3 + 5.8182 \quad (26a)$$

$$t_2 = -.3556i_1 - .6222i_2 - .3556i_3 + 12.0000 \quad (26b)$$

$$t_3 = -.1293i_1 - .1293i_2 - .2263i_3 + 4.8485 \quad (26c)$$

$$t_4 = -.3556i_1 - .3556i_2 - .6222i_3 + 13.3333 \quad (26d)$$

$$\tau_{m_2} = t_1 + s_2 = -.4929i_1 - .2263i_2 - .1293i_3 + 6.8182 \leq i_2 \quad (27)$$

$$\tau_{m_3} = t_1 + s_2 + t_2 + s_1 + t_3 + s_3 = -.9778i_1 - .9778i_2 - .7111i_3 + 25.6667 \leq i_3. \quad (28)$$

The set of all initial inventories which can obtain $\mathbf{U} = [0\ 5\ 10]$ by implementing $\mathbf{f} = [1\ 2\ 1\ 3]$ without a stock out occurring is defined by the inequalities in Equations 27 and 28 and $t_j \geq 0$ for $j = 1,2,3,4$.

By applying the method to all production sequences, the feasible region can be found. Figure 1 shows the feasible region after searching all production sequences of length 7, where the products first appear in order of increasing inventory for $i_1 = 0$ and $i_2 \leq i_3$. In order to aid in the convergence of the feasible region, a value of $\boldsymbol{U}$ with low inventory levels is desired. The rotation schedule is efficient in regards to feasibility because it has the minimum amount of setup time for any cyclic schedule. If the rotation schedule [1 2 3] is in "steady state," the inventory levels at the beginning of the production runs will rotate between $[0\ k\ 2k]$, $[2k\ 0\ k]$, $[k\ 2k\ 0]$ where $k = \frac{d_1 p_1 s_1}{p_1 - 3d_1}$.

Therefore, $\boldsymbol{U} = [0\ 40\ 80]$, (the inventory vector) is prior to the transformation described in section 3.1. The area of the feasible region bounded by $i_2 \leq i_3 \leq 200$ is used as a metric for convergence. Note that in practice, there is no need to generate the feasible region beyond the largest planned inventory. Consider Figure 1. As the number of positions in the production sequences considered increases, the gain in the "area" of the feasible region becomes negligible very quickly. All sequences with 7 or fewer positions were considered. The region generated by the rotation schedule alone is 95.775% of the feasible region found by all sequences with 7 or fewer positions, and the area generated by all schedules up to length 6 represents 99.797% of the feasible region found by all sequences of length at most 7(see Table 3).

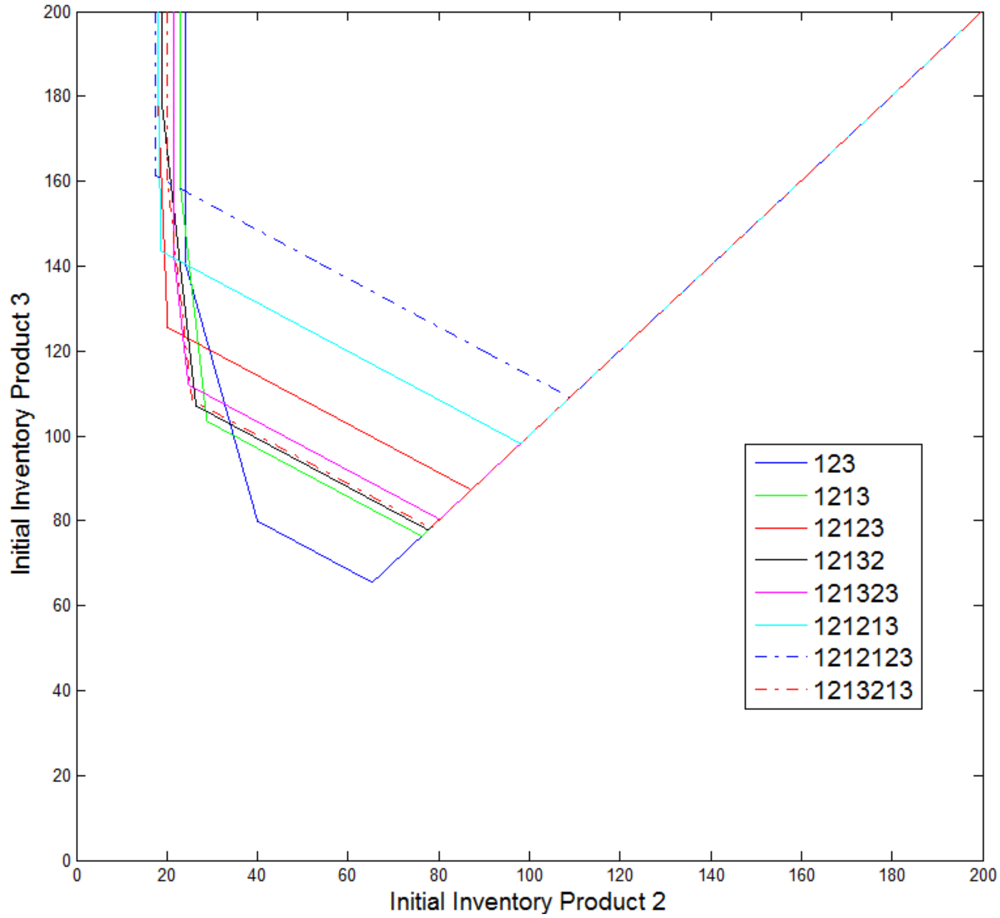

Figure 2: The Feasible Region

Table 3: Percent of the Feasible Region

Length	Area/Area Length 7
3	0.95775
4	0.96746
5	0.99616
6	0.99797

Furthermore, inventory vectors near the boundary of the feasible region are not desirable for two reasons: (1) if the parameters are not truly deterministic then a stock out might still occur, and (2) the time required to recover the necessary inventory to apply a cyclic schedule may be arbitrarily large. In practice, it is critical to maintain a distance from the boundary of the feasible region, as the manufacturing system is vulnerable for those inventory levels. Determining the optimal policy, while maintaining a given reliability (i.e. safety stock), is an area of future research.

6 Conclusion:

An analytical method was developed which determines the maximum time a stock out can be avoided given a production sequence $\boldsymbol{f}$ and initial inventory level, $\boldsymbol{I}$. A bound on the number of production runs in $\boldsymbol{f}$ was also found. A search up to the bounds determines the maximum time a stock out can be avoided (using any schedule). Further modification enabled us to determine the minimum time required to obtain sufficient inventory to apply a desired cyclic schedule. Finally, the method was generalized by treating the initial inventory levels as variables to determine the feasible region.

Collectively, the results in this paper provide managers with an early detection system for systems with low inventory levels (Section 5), and the necessary tools to, if necessary, prevent a stock out (if possible) until additional production capacity can be obtained (Section 3) and return inventory levels to the desired level in the minimum amount of time (Section 4). In addition, the ability to determine the feasible region provides a representation of all inventory positions from which it is possible to avoid a stock out. This may lead to additional insights into appropriate inventory control policies.

CHAPTER 4 REFERENCES

Anderson, E. (1990) Testing feasibility in the lot scheduling problem. *Operations Research*, **38**, 1079-1088.

Andres, F.M. and Emmons, H. (1976) On the optimal packaging frequency of products jointly replenished. *Management Science*, **22(**10), 1165-1166.

Arkin, E. M., Gallego, G., and Roundy, R.O. (1988) Complexity of the multiple product single facility stock out avoidance problem. Tech. Report No. 822, School of Operations Research. Res. And Ind. Eng., Cornell University.

Boctor, F. (1987) The G-group heuristic for single machine lot scheduling. *International Journal of Production Research*, **25(**3), 363-379.

Bomberger, E. (1966) A dynamic programming approach to a lot size scheduling problem. *Management Science*, **12**(11), 778-784.

Delporte, C. and Thomas, L (1977) Lot sizing and sequencing for *N* products on one facility. *Management Science*, **23**(10), 1070-1079.

Dobson, G. (1987) The economic lot-scheduling problem: achieving feasibility using time-varying lot sizes. *Operations Research*, **35**(5), 764-771.

Doll, C. and Whybark, D. (1973) An iterative procedure for the single-machine multi-product lot scheduling problem. *Management Science*, **20**(1), 50-55.

Eilon, S. (1962) *Elements of Production Planning and Control*, Macmillan, New York, Chapter 10, 227-263.

Elmaghraby, S. (1978) The economic lot scheduling problem (ELSP): review and extensions. *Management Science*, **20**(1), 50-55.

Gallego, G. (1990) Scheduling the production of several items with random demands in a single facility. *Management Science*, **36**(12), 1579-1592.

Gallego, G. and Moon, I. (1996) How to avoid stock outs when producing several items in a single facility? What to do if you can't. *Computers Operations Research*, **23,** 1-12

Haessler, R. and Hogue, S. (1976) A note on the single-machine multi-product lot scheduling problem. *Management Science*, **22**(8), pp. 909-912.

Hanssmann, F. (1962) *Operations Research in Production and Inventory Control*, Wiley, New York.

Hodgson, T.J. (1970) Addendum to Stankard and Gupta's note on lot size scheduling. *Management Science*, **16**(7), 514-517.

Hodgson, T. (1980) On-line scheduling in the production environment: an "unsuccessful" case history. *Interfaces*, **10(1)**, 77-80

Hodgson, T.J. and Nuttle, H. (1986) A note on linear programming and the single machine lot size scheduling problem. *International Journal of Production Research*, **24**(4) 939-943.

Hsu, W.L. (1983) On the general feasibility test of scheduling lot sizes for several products on one machine. *Management Science*, **29**(1), pp. 93-105.

Jones, P. and Inman, R. (1989) When is the economic lot scheduling problem easy? *IEE Transactions*, **21**(1), 11-20.

Maxwell, W.L. (1964) The scheduling of economic lot sizes. *Naval Research Logistics Quarterly*, **11**(213), 89-124.

Rogers, J. (1958) A computational approach to the economic lot scheduling problem. *Management Science*, 4(3), 264-291.

Schweitzer, P. and Silver, E. (1983) Mathematical pitfalls in the one machine multiproduct economic lot scheduling problem. *Operations Research*, **31(**2), 401-405.

Stankard, M. and Gupta, S. (1969) A note on Bomberger's approach to lot size scheduling: heuristic proposed. *Management Science*, **15**(7), 449-552.

Zipkin, Paul H. (1991) Computing optimal lot sizes in the economic lot scheduling problem. *Operations Research*, **39**(1), 56-63.

CHAPTER 4 APPENDICES

Appendix A

Table A1 contains the computational times for searching all production sequences for n products of length at most N using Matlab R2011a on a 64 bit, 3 GHz quad-core processor. The first production run is assumed to be for the product currently setup, and the first appearance of the remaining products are assumed to be in order of increasing initial inventory levels.

Table 4: Computational Times (Seconds)

N/n	3	4	5	6	7
3	0.045				
4	0.0354	0.031			
5	0.0362	0.052	0.058		
6	0.034	0.087	0.52	3.25	
7	0.068	0.42	7.55	82.42	578.126
8	0.078	3.35	117.31	**	**
9	0.1746	29.1	**	**	**
10	0.47	258.29	**	**	**
11	1.66	**	**	**	**
12	6.21	**	**	**	**
13	24.11	**	**	**	**
14	95.15	**	**	**	**

**Times are greater than last entry in column.

Appendix B

The algorithm in Section 3.3 is applied with the changes noted in Section 5 for $\boldsymbol{f} = [1\ 2\ 1\ 3]$ and $\boldsymbol{U} = [0\ 5\ 10]$ as follows:

Step 2: $v = 4;$ $z_1^4 = x_1 = -.6222i_1 - .3556i_2 - .3556i_3 + 10.6667;$

$z_2^4 = x_2 = -.3556i_1 - .6222i_2 - .3556i_3 + 12.0000;$

$z_3^4 = x_3 = -.3556i_1 - .3556i_2 - .6222i_3 + 13.3333.$

Steps 3-5: $t_4 = -.3556i_1 - .3556i_2 - .6222i_3 + 13.3333$;
$z_1^3 = -.6222i_1 - .3556i_2 - .3556i_3 + 10.6667$;
$z_2^3 = -.3556i_1 - .6222i_2 - .3556i_3 + 12.0000$;
$z_3^3 = 0$; $v = 3$.

Steps 3-5: $t_3 = z_1^3 - \frac{(z_2^3+0+s_f)-i_1}{p_1-1} = -.1293i_1 - .1293i_2 - .2263i_3 + 4.8485$;
$z_1^2 = -.4929i_1 - .2263i_2 - .1293i_3 + 5.8182$;
$z_2^2 = -.3556i_1 - .6222i_2 - .3556i_3 + 12.0000$;
$z_3^2 = 0$;$v = 2$.

Steps 3-5: $t_2 = -.3556i_1 - .6222i_2 - .3556i_3 + 12.0000$;
$z_1^1 = -.4929i_1 - .2263i_2 - .1293i_3 + 5.8182$;
$z_2^1 = 0$; $z_3^1 = 0$;$v = 1$.

Steps 3-4: $t_1 = -.4929i_1 - .2263i_2 - .1293i_3 + 5.8182$;$v = 0$.

Steps 6: $\tau_{m_2} = t_1 + s_2 = -.4929i_1 - .2263i_2 - .1293i_3 + 6.8182 \leq i_2$;
$\tau_{m_3} = t_1 + s_2 + t_2 + s_1 + t_3 + s_3 = -.978i_1 - .978i_2 - .711i_3 + 25.667 \leq i_3$.

Chapter 5

Conclusion

There are several areas of future research. First, the power-of-two and power-of-primes methods could be extended to n-products produced on m machines arranged in parallel. The machines are assumed to be capable of producing any of the n products. However, typically the assumption is made that each product is only produced on a single machine. This simplifies the problem and is a reasonable assumption for practice. Scheduling the machines consists of an assignment problem determining which products are produced on which machine, and m, ELSP problems. As previously noted, methods for the ELSP have been well studied. However, the current understanding of the relationship between the assignment problem and the minimization of total cost is limited. The power-of-two and power-of-prime methods provide a natural way of determining which products are more cost effective to be produced on the same machine.

Second, the situation where inventory levels become critically low is also relevant for the n-product, m-machine problem. If one machine is disrupted, then the products produced on that machine could be distributed to the other machines in an effort to delay a stock out. This adds a significant degree of complexity to an already NP-hard problem.

Third, the minimization of total cost is done assuming a completely reliable system, and, as such, does not consider the risk of a stock out. For the reasons discussed at length in the third paper, events may prevent the production schedule from being implemented exactly as planned. There is some risk that an event will cause the inventory position will become "infeasible" and additional production capacity must be obtained at additional cost. This motivates the minimization of total cost while maintaining a specified system reliability in the presence of disturbances (i.e. determining necessary safety stock). If the cost of recovery (overtime, outsourcing, etc.) is known, the total operating cost including inventory cost, setup cost, and recovery cost could be minimized.

REFERENCES

Anderson, E. (1990) Testing feasibility in the lot scheduling problem. *Operations Research*, **38**, 1079-1088.

Andres, F.M. and Emmons, H. (1976) On the optimal packaging frequency of products jointly replenished. *Management Science*, **22(**10), 1165-1166.

Arkin, E. M., Gallego, G., and Roundy, R.O. (1988) Complexity of the multiple product single facility stock out avoidance problem. Tech. Report No. 822, School of Operations Research. Res. And Ind. Eng., Cornell University.

Boctor, F. (1987) The G-group heuristic for single machine lot scheduling. *International Journal of Production Research*, **25**(3), 363-379.

Bomberger, E. (1966) A dynamic programming approach to a lot size scheduling problem. *Management Science*, **12**(11), 778-784.

Delporte, C. and Thomas, L (1977) Lot sizing and sequencing for *N* products on one facility. *Management Science*, **23**(10), 1070-1079.

Dobson, G. (1987) The economic lot-scheduling problem: achieving feasibility using time-varying lot sizes. *Operations Research*, **35**(5), 764-771.

Doll, C. and Whybark, D. (1973) An iterative procedure for the single-machine multi-product lot scheduling problem. *Management Science*, **20**(1), 50-55.

Eilon, S. (1962) *Elements of Production Planning and Control*, Macmillan, New York, Chapter 10, 227-263.

Elmaghraby, S. (1978) The economic lot scheduling problem (ELSP): review and extensions. *Management Science*, **20**(1), 50-55.

Gallego, G. (1990) Scheduling the production of several items with random demands in a single facility. *Management Science*, **36**(12), 1579-1592.

Gallego, G. and Moon, I. (1996) How to avoid stock outs when producing several items in a single facility? What to do if you can't. *Computers Operations Research*, **23,** 1-12

Haessler, R. and Hogue, S. (1976) A note on the single-machine multi-product lot scheduling problem. *Management Science*, **22**(8), pp. 909-912.

Hanssmann, F. (1962) *Operations Research in Production and Inventory Control*, Wiley, New York.

Hodgson, T.J. (1970) Addendum to Stankard and Gupta's note on lot size scheduling. *Management Science*, **16**(7), 514-517.

Hodgson, T. (1980) On-line scheduling in the production environment: an "unsuccessful" case history. *Interfaces*, **10(1)**, 77-80

Hodgson, T.J. and Nuttle, H. (1986) A note on linear programming and the single machine lot size scheduling problem. *International Journal of Production Research*, **24**(4) 939-943.

Hsu, W.L. (1983) On the general feasibility test of scheduling lot sizes for several products on one machine. *Management Science*, **29**(1), pp. 93-105.

Jones, P. and Inman, R. (1989) When is the economic lot scheduling problem easy? *IEE Transactions*, **21**(1), 11-20.

Khouja, M., Michalewicz, Z., Wilmont, M. (1998) The use of genetic algorithms to solve the economic lot size scheduling problem. *European Journal of Operational Research*, **110**, 509-524.

Maxwell, W.L. (1964) The scheduling of economic lot sizes. *Naval Research Logistics Quarterly*, **11**(213), 89-124.

Raza, A. and Akgunduz, A. (2007) A comparative study of heuristic algorithms on Economic Lot Scheduling Problem. *Computers & Industrial Engineering*, **55**, 94-109

Rogers, J. (1958) A computational approach to the economic lot scheduling problem. *Management Science*, 4(3), 264-291.

Schweitzer, P. and Silver, E. (1983) Mathematical pitfalls in the one machine multiproduct economic lot scheduling problem. *Operations Research*, **31(**2), 401-405.

Stankard, M. and Gupta, S. (1969) A note on Bomberger's approach to lot size scheduling: heuristic proposed. *Management Science*, **15**(7), 449-552.

Zipkin, Paul H. (1991) Computing optimal lot sizes in the economic lot scheduling problem. *Operations Research*, **39**(1), 56-63.