# Sufficient Conditions For An Optimal Solution To The Problem Of Scheduling A Bottleneck Of An N-Stage Process

Joseph S. Morrell, (E-mail: morrellj@mscd.edu), Metropolitan State College of Denver
Norman Pence, (E-mail: pencen@mscd.edu), Metropolitan State College of Denver

**ABSTRACT**

*This paper discusses the application of an algorithm for scheduling the bottleneck of an n-stage process involving multiple products to information systems development. The algorithm is presented along with Pence's Theorem, conditions sufficient for an optimal solution are presented, and the application of this algorithm and theorem to the development of information systems.*

## INTRODUCTION

Some of the available publications which address the problem of scheduling multiple products on a single machine are listed in the references. The paper by Pence, Megeath, and Morrell (1990) provides the basis for this paper in that it develops an algorithm for the scheduling of multiple products through a n-stage process where one of the machines necessarily dictates the schedule of the remaining n-1 machines and provides conditions for optimality while covering two general problems: One is the development of a generalized algorithm to schedule the bottleneck of an n-stage process and the other is the development of a generalized n-stage scheduling procedure to schedule an n-stage process.

Since the procedures are so important to the application of these results, the algorithm and theorem are presented herein. The generalized scheduling algorithm can be used for a variety of production problems. This algorithm was given the following name:

## BOTTLENECK SCHEDULING ALGORITHM

### Conditions

A collection of products, each requiring a certain amount of processing time, needs to be processed through a machine that is the bottleneck in an n-stage process. Preparation (set-up/tear-down) time is substantial and costly, and the total preparation time for a collection of products is a function of the sequence of the products. A single product quality variable determines the preparation (set-up/tear-down) time. The last product processed in the previous production schedule through this bottleneck is known.

### Objective

Schedule the bottleneck to minimize the total amount of preparation (set-up/tear-down) time for a collection of products.

1. Determine the products to be processed.
2. Identify the product quality variable influencing preparation (set-up/tear-down) time.
3. Select a unit of measure for this product quality variable and assign a numerical value to each of the products in step 1 according to this product quality variable.

4. Arrange the products from step 1 in either descending value order or ascending value order as determined in step 3 depending on whether a lower bound on product quality is desired or an upper bound on product quality is desired. Assign product numbers from 1 to n according to this numerical sequence.
5. Start the schedule with the same product that was last processed in the previous schedule if that product is demanded. If this product is not demanded, start the schedule with the next demanded product after the product last processed, in the numerical sequence from step 4. When the end of the numerical sequence from step 4 is reached, the next product processed will be no. 1 and continue the schedule following the numerical sequence until the demanded collection has been scheduled.
6. Determine the preparation (set-up/tear-down) time necessary for each pair of consecutive products in the step 5 sequence.
7. For the sequence of preparation (set-up/tear-down) times and products from steps 5 and 6, calculate the starting and finishing times for processing each of the products and the starting and finishing times for each of the preparations.
8. Schedule the bottleneck using the results of step 7.

**GENERALIZED PROCEDURE**

To schedule all n stages of the process, the following generalized n-stage scheduling procedure was developed. The $m^{th}$ stage of the procedure is the bottleneck.

**GENERALIZED N-STAGE SCHEDULING PROCEDURE**

1. Determine the demand for each of the products for time period 1 through time period n.
2. Reduce the demand in time period 1 through time period n for each of the products to reflect the finished inventory of each of the products and also to reflect the in-process inventory of each of the products.
1. Assign group numbers to each of the products:
4. Group 1 - products with demand in time period 1.
   Group 2 - products with demand in time period 2 and demand in time period 1 is 0.
   Group n - products with demand in time period n and demand in time periods 1 through n-1 is 0.
   Add demands for time period 1, time period 2,..., time period k for each of the group 1 products.
   Add demands for time period 2, time period 3,..., time period k for each of the group 2 products.

The demand for each of the group n products is simply the demand for time period n. NOTE: k represents number of time periods to be reflected in the production schedule.

5. Determine the schedule for the bottleneck, the $m^{th}$ stage of the process, by using the Bottleneck Scheduling Algorithm. Group 1 products are scheduled first, group 2 products are scheduled second, ... , group n products are scheduled last.
6. Schedule the m-1 stages before the bottleneck by using the schedule generated in step 5 for the bottleneck. Schedule in the order--stage 1 through stage m-1.
7. Schedule the n-m stages after the bottleneck using the schedule generated in step 5 for the bottleneck. Schedule in the order--stage m+1 through stage n.

**SUFFICIENT CONDITIONS**

Analysis of the scheduling of the bottleneck resulted in the discovery of three sufficient conditions for an optimal solution to the scheduling of the bottleneck.

Optimality is defined in the sense of minimizing the total amount of preparation (set-up/tear-down) time.

Suppose n products exist that can be processed using a given machine and they are designated as $P_i$ where i ranges from 1 to n. The n products are in descending or ascending order of some defined measure of product quality.

Let $(C_{i,j})_{nxn}$ be nxn matrix illustrated in Figure 1. below where $C_{i,j}$ represents the hours of preparation that are necessary between the end of the processing of product $P_i$ and the beginning of the processing of product $P_j$.

**Figure 1: The nxn matrix, $(C_{i,j})_{nxn}$, of cost of preparation between Product i and Product j**

| | $P_1$ | $P_2$ | $P_3$ | ... | $P_{n-1}$ | $P_n$ |
|---|---|---|---|---|---|---|
| $P_1$ | $C_{1,1}$ | $C_{1,2}$ | $C_{1,3}$ | ... | $C_{1,n-1}$ | $C_{1,n}$ |
| $P_2$ | $C_{2,1}$ | $C_{2,2}$ | $C_{2,3}$ | ... | $C_{2,n-1}$ | $C_{2,n}$ |
| $P_3$ | $C_{3,1}$ | $C_{3,2}$ | $C_{3,3}$ | ... | $C_{3,n-1}$ | $C_{3,n}$ |
| . . | . . | . | . | . | | |
| . . | . . | . | . | . | | |
| . . | . . | . | . | . | | |
| $P_{n-1}$ | $C_{n-1,1}$ | $C_{n-1,2}$ | $C_{n-1,3}$ | ... | $C_{n-1,n-1}$ | $C_{n-1,n}$ |
| $P_n$ | $C_{n,1}$ | $C_{n,2}$ | $C_{n,3}$ | ... | $C_{n,n-1}$ | $C_{n,n}$ |

Let $C[(P_1,P_2 \ldots P_n)] = \Sigma C_{i,i+1}$ for i = 1 to n-1 denote the Total Preparation Time for the vector of products $(P_1,P_2 \ldots P_n)$ where $C_{i,j}$ are as above. The vector $(P_1,P_2 \ldots P_n)$ is said to satisfy the **Pence Hypothesis (PH)** if

1. $C_{i,j} \geq C_{j,i}$ for $n \geq i > j \geq 1$.
2. For each row i = 1 to n, the sequence $C_{i,j}$ for $n \geq j \geq i$ is monotonically nondecreasing.
3. For each column j = 1 to n, the sequence $C_{i,j}$ for $j \geq i \geq 1$ are monotonically nonincreasing.

## THEOREM 1

### Pence's Theorem

Let $(P_1,P_2 \ldots P_n)$ be an n vector of products for which cleanup matrix $(C_{i,j})_{nxn}$ satisfies the Pence Hypothesis (PH). Then, $C[(P_1,P_2 \ldots P_n)]$ is optimal, i.e., if $\Pi((P_1,P_2 \ldots P_n))$ is any permutation of $(P_1,P_2 \ldots P_n)$. Then $C[\Pi((P_1,P_2 \ldots P_n))] \geq C[\Pi_o((P_1,P_2 \ldots P_n))]$ where $\Pi_o((P_1,P_2 \ldots P_n))$ is the trivially ordered set $(P_1,P_2 \ldots P_n)$.

The elegant proof of this theorem is omitted but is included in the reference Pence (1986) which also provides a counter-example for the necessity of these conditions.

## APPLYING THE ALGORITHM TO INFORMATION SYSTEMS DEVELOPMENT

The process of developing information systems includes requirement specification, systems and module design, coding or developing the programs, modules, or units, unit testing, system testing, implementation and installation. The system consists of a collection of programs, modules, or units which, for the sake of simplicity will be called a program. This paper concerned with the construction or development of programs which includes problems statement, process modeling, implementation in a selected language, and unit test for the program. The totality of these steps will be referred to as completion of the program. The time to complete a program depends on many factors, but one of these is the dependency on the development of another program. In some cases one program serves as a module of another program. In other cases, the similarities in the problem statements of two programs may allow for one program to be written as a revision of the other. In some cases, the modeling used in one program may be used as part of another with modifications. In other cases, the unit test of one program may depend on the existence of another to provide data for the test. In any event the completion time for one program may be dependent on the existence of another program.

In order to apply the Bottleneck Scheduling Algorithm and Pence's Theorem for optimization of development time for the system, we define application of this algorithm to information systems development, $P_1, P_2 \ldots P_n$ will represent programs required for the development of the system. Let $C_{i,j}$ represent completion time for $P_j$ assuming that program $P_i$ is completed prior to starting on the development of program $P_j$. Applying Pence's theorem for the optimization with $C[(P_1, P_2 \ldots P_n)]$ representing the summation of the completion times and referred to as the development time of the systems as described in the theorem we have the following. Suppose the sequence $P_1, P_2 \ldots P_n$ satisfies the following: The completion time for $P_j$ assuming that $P_i$ is completed must be greater or equal to the testing time for $P_j$ assuming it is completed before $P_i$. For any Program $P_i$ the completion time for $P_k$ assuming $P_i$ is completed must be greater than or equal to the completion time for $P_h$ assuming the completion of $P_i$ for all h,k such that $h > k > i$. For any program $P_j$ the completion time for $P_j$ assuming the completion of $P_h$ must be less than or equal to the completion time for $P_j$ assuming the completion of $P_k$ for all $h > k > j$. Then the programs should be completed chronologically in the order $P_1, P_2 \ldots P_n$ for the optimal (minimal) development time for the system.

Thus if the completion time dependencies for pairs of programs in a system being developed are known then there is an optimal scheduling of the programs which constitute the system provided some chronological ordering of the programs satisfy the Pence Hypothesis.

## CONCLUSIONS

The Bottleneck Scheduling Algorithm and Pence's Theorem provide for the optimal scheduling of writing modules and programs. The availability of the metrics for the completion dependencies may be difficult. Another obstacle to the use of the algorithm may occur in large systems where numerous modules or programs are required in the system completion process. Computer programs have been completed using the scheduling algorithm in manufacturing applications and the authors are pursuing the modification of these programs to the systems development setting. The observations in this paper provide a starting point of applying scheduling algorithms to information systems development. The successful application could be as much of an economic windfall to the IS industry as it was in its application to the manufacturing industry. The bottleneck scheduling algorithm was implemented at Coors Ceramics Company and generated a cost savings in excess of one million dollars.

## REFERENCES

1. Afentakis, P., 1985, "Simultaneous Lot Sizing and Sequencing for Multistage Production Systems," *IIE Transactions*, Vol. 17, No. 4, pp. 327-331.
2. Elmaghraby, S. E., 1968, "The Machine Scheduling Problem - Review and Extensions," *Naval Research Logistics Quarterly*, Vol. 15, No. 2, pp. 205-232.
3. Flood, M. M., 1955, "The Traveling-Salesman Problem," *Operations Research*, Vol. 4, No. 1, pp. 61-75.
4. Goyal, S. K., 1975, "Scheduling a Multi-Product Single-Machine System," *Operational Research Quarterly*, Vol. 24, No. 2, pp. 261-266.
5. Oliff, M. D. and Burch, E. E., 1985, "Multiproduct Production Scheduling at Owens-Corning Fiberglas," *Interfaces*, Vol. 15, No. 5, pp. 25-34.
6. Pence, N. E., 1986, "An Economic Model of the Multiproduct Lot Scheduling Problem Applied to Refined Ceramic Mineral Products", PhD dissertation, Colorado School of Mines.
7. Pence, N. E., Megeath, J. D., and Morrell, J. S., 1990, "Coping with Temporary Bottlenecks in a Several-Stage Process with Multiple Products", *Production and Inventory Management Journal,* Vol. 31, No. 3, pp. 5-6.
8. Woolsey, G., 1982, "An Essay on the Set-up, Tear-Down Problem or Being Clean, Being Profitable, or Both," *Interfaces*, Vol. 12, No. 4, pp. 11-13.