FACILITY CAPITAL EQUIPMENT AND LABOR DECISION SUPPORT SYSTEM USING A DISCRETE-EVENT SIMULATION AND BOTTLENECK DETECTION APPROACH

by

MOHAMMED ALMANSOURI

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON
May 2014

Acknowledgements

I would have never been able to complete my dissertation without the help and support from the committee members, family, and friends.

I would like to express my deepest appreciation to my committee chair Professor Brian Huff for his superior guidance, support, and patience.

I would like to thank my committee members, Professor Donald Liles and Professor John Priest for their thoughtful criticism and encouragement words.

Finally, I would like to thank my parents and family for their prayers, encouragement, support, and patience during this time. Thank you all for everything.

April 16, 2014

Abstract

FACILITY CAPITAL EQUIPMENT AND LABOR DECISION SUPPORT SYSTEM USING A DISCRETE-EVENT SIMULATION AND BOTTLENECK DETECTION APPROACH

Mohammed AlMansouri, PhD

The University of Texas at Arlington, YYYY

Supervising Professor: Brian Huff

Market demand is constantly changing. Therefore, it is critical for companies to be flexible and willing to adapt in order to remain competitive. This study will evaluate bottleneck detection techniques that have been identified in previous research by comparing the performance of each method on all case study models that can be replicated from the literature. The purpose of this analysis is to identify the most efficient and reliable bottleneck detection algorithm(s) which are capable of supporting constraint identification on a wide range of production system classes and configurations. The primary objective will be to identify a robust bottleneck detection algorithm, or a small set of algorithms, that can be broadly applied to various types of production or service provision environments. Discrete-event simulation will be used to support this algorithm evaluation task and will also play a key role in the application of bottleneck detection methods in real-world production scenarios.

This research will also integrate financial project justification methods to verify that each proposal to increase production can be financially justified. The net present value and internal rate of return performance measures in conjunction with the equity cash flow and minimum annual revenue requirements project justification methods will be used for this purpose. Both the bottleneck detection techniques and the project justification methods will be applied in order to solve the problem of both demand reduction as well as demand growth.

To demonstrate the general applicability of the proposed facilities analysis methods to real-world production or service provision system, a prototype decision support system was developed that can help decision makers determine what modifications are required in order to allow their production systems to adapt to shifts in market demand. The features of the prototype decision support tool will be demonstrated on models of production systems that are derived from actual production operations, with the goal of providing a general capacity analysis and financial justification tool that can be applied to a wide range of production or service provision system scenarios and designs.

Table of Contents

List of Illustrations

List of Tables

Chapter 1

Introduction

Market demand is constantly changing due to economic constraints. Successful companies adapt rapidly to higher demand by increasing their throughput (Rahman, 1998). If demand decreases, successful organizations can also move quickly to eliminate unnecessary costs. Companies obviously prefer the increase in demand since it is tied to higher profits, however increasing throughput is not an easy undertaking for many companies (Kasemset & Kachitvichyanukul, 2007) (Roser et al., 2001) and has therefore been the focus of much research. Studies have shown that augmenting productivity can potentially be cost-prohibitive and could prevent decision-makers from considering a particular enhancement in their manufacturing facility (Li, 2009). The objective of this dissertation is to develop a decision support system that will assist managers to adapt to changes in demand, and to identify the section of the factory that requires enhancement based on bottleneck analysis. It will also demonstrate how to justify the costs incurred by making such modifications by utilizing equity cash flow and minimum annual revenue requirement methods using “net present value” and “internal rate of return” performance measures.

1.1 Impact of Increasing or Decreasing Demand

When demand increases, company executives will start exploring the options of increasing raw materials input, installing additional equipment, hiring additional workers, or a combination thereof in order to increase production line throughput. Such a decision is critical when an organization is unable to meet the increase in demand, as the company will be unable to benefit from the opportunity to increase sales and profit, leading to a potential loss of market share. This could also result in loss of customer loyalty and trust as they seek out alternate suppliers who can meet their expectations.

Decision-makers are frequently challenged with the task of selecting the most appropriate solution when faced with a wide range of options. Studies have shown that a single resource, or a small set of co-constraints, most often limits the capacity of a production line (Goldratt, 1992) (Roser et al., 2001) (Roser et al., 2002) (Roser et al., 2003) (Li, 2009), and suggest that identifying constraints in the production process is the most efficient method of determining the best alternative to increase production. These constraints will be called “bottlenecks” throughout this study.

If demand decreases and the company is producing in excess of the current demand, the company will be less profitable due to a buildup of finished goods inventories and work in progress. Executives will once again be faced with a wide range of options to reduce expenses and curtail production in order to meet consumer demand. Constraint-based capacity analysis techniques can be implemented to identify the least critical resource, which can be eliminated to reduce costs and to achieve the lower production target.

1.2 Current Approaches

Measuring and evaluating the productivity of companies has been the focus of numerous researchers (Goldratt, 1992) (Roser et al., 2001) (Roser et al., 2002) (Roser et al., 2003) (Leporis & Králová, 2010) (Li, 2009) (Kasemset & Kachitvichyanukul, 2008) (Taha, 2008). Four general types of methods are used to measure the performance of a manufacturing facility or system: analytical method, simulation-based method, simulation and optimization, and bottleneck analysis.

The analytical method converts the problem into a mathematical model. Based on the application scope, the mathematical model must be limited to the assumptions required by the application (Taha, 2008). Although this method has been used extensively in copious studies, it might not be suitable as a standard method of measuring performance (Taha, 2008) (Kelton et al., 2010). Law & Kelton (1991) stated that "*Complex real-world systems with stochastic elements cannot be accurately described by mathematical models that can be evaluated analytically*." According to Li (2009), analytical methods are not practical when measuring the throughput of a serial production line consisting of more than two stations and a single buffer. A prominent example of an analytical method is the queuing theory. Kelton, Sadowski, and Swets (2010) summarize its limitation as follows: Converting a problem in queuing theory model requires the use of exponential distribution for inter-arrival time and service time which might not be appropriate for all the real-life systems. Moreover, in order to obtain an accurate result, the model must be in a steady state, which is achieved by running it for a sufficient length of time. This limits the analyst from testing the model for a pre-defined duration if it is a shorter time period than is required to bring the model to the steady state (Taha, 2008) (Kelton et al., 2010).

Similar to the analytical method, simulation can be used to analyze waiting lines, which is a way of replicating the real system or service in order to measure performance (Taha, 2008). Researchers have identified the advantages of using simulation. Kelton, Sadowski, and Swets (2010), Law and Kelton (1991), and Waller, Anthony (2006) pointed out that, unlike analytical methods, the simulation approach can provide useful findings for complex models. They also agreed that the simulation approach is more flexible because the researcher can apply various simulations of diverse operating conditions to the same model with lower costs (Svancara & Kralova, 2009). Taha (2008) also pointed out that simulation can be used to analyze the transient behavior of the system observed before stead state is achieved. Analytical methods, like queuing theory, only represent the steady state behavior of a system.

Although decision-makers tend to use the simulation-based approach to analyze the facilities or systems in order to achieve their goals and objectives (Li, 2009), the simulation-based method has disadvantages. Previous studies have also illustrated how a stochastic simulation might produce a different output for every run, and therefore the results might not be reliable (Law & Kelton, 1991) (Kelton et al., 2010). In addition, Li (2009) identified some of the following drawbacks: model development is complex, modifying the model is challenging, and result interpretation is ambiguous. Moreover, the cost of conducting simulation experiments could be substantial in terms of budget and time. For instance, suppose there is a need to evaluate k different solutions that require testing N times in a manufacturing facility. In that case, kN trials must be conducted to finalize a decision. In order to have accurate results, N must be high. Additionally, if there are a large number of solutions, the required number of trial runs (kN) would be considerably high (Chen et al., 1997).

Simulation optimization is a combination of analytical methods and simulation-based analysis that can be applied to find the best design with optimal throughput (Fu, 2002). It is important to note that the simulation model is independent from the simulation optimization process, as shown in Figure 1-1. This approach renders the process more flexible and common to various types of simulation models. Simulation optimization depends on defining the objective function, which is the performance measure of the model. With this method, the simulation model must run every time the objective function is evaluated (Better et al., 2008). Moreover, stochastic models produce stochastic noise every time the model runs, a variance that can be reduced by running multiple replications of the simulation scenario. This means that the objective function performance measures can be misleading if an insufficient number of replications are run (Fu, 2002). Despite this potential limitation, simulation optimization helps the analyst to experiment with the model elements such as raw materials, resources, etc. to determine the optimal design, along with the ideal flow of material, to increase productivity.

The simulation optimization process begins with the development of a base simulation model that defines the process flows, resource requirements, processing times, and operational rules that characterize the behavior of the production system. In order to optimize the performance of the simulation model, the user must first outline the objective function that requires maximizing or minimizing. Additionally, the user must also identify a set of constraints that simultaneously defines and limits the potential system configurations that can be evaluated for optimality. System configuration variables such as the quantity of each type of equipment, the number of workers per shift, the number of shifts per day, etc., combined with their range of allowable values, are defined as system constraints. If stochastic variables are used in the model, the simulation model output will produce a variance. In order to minimize the impact of this variance, the user must define the number of independent replications to be run for each potential simulated system configuration.

Afterwards, the simulation optimization process continues by changing the input parameters of the base model and providing a family of suggested solutions. The process carries on and runs the family of solutions through the simulation package until the suggested solution satisfies the previously-defined objective function. As the reader might wonder, this process might require a significant length of time to explore every possible solution recommended in the family of suggested solutions. For that reason, search algorithms, such as *Tabu* and *Scatter Search,* were developed, which take advantage of memory and population sampling techniques to reduce the number of experiments based on the results of previous experimental runs (Fu, 2002) (Better et al., 2008).

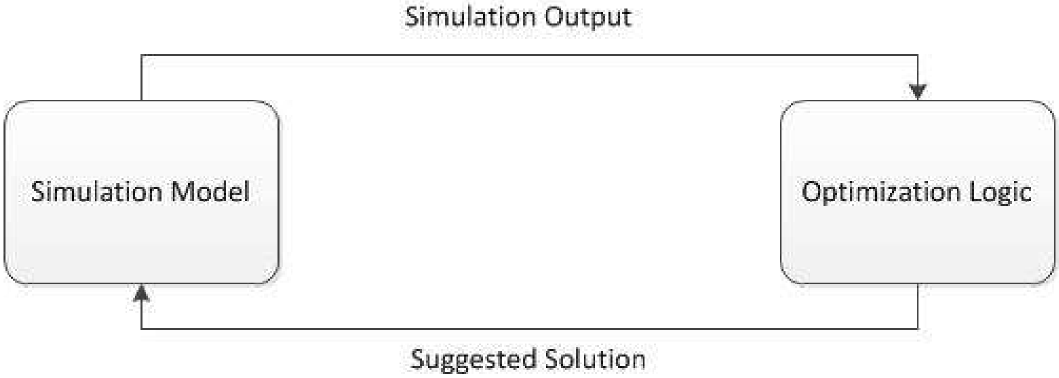


Figure 1-1 Simulation Optimization Process (Fu, 2002)

Bottleneck analysis was derived from the concept of the theory of constraints (TOC) introduced in the book “The Goal” (Goldratt, 1992). Bottleneck resource is the term that researchers have agreed on to define the resource that limits the throughput of a manufacturing facility or service (Goldratt, 1992) (Kasemset & Kachitvichyanukul, 2008) (Leporis & Králová, 2010) (Li, 2009) (Roser et al., 2002) (Roser et al., 2003). There have been an abundance of definitions of bottleneck resource that helped individuals understand the concept from different angles. For example, Leporis and Králová (2010) define bottleneck as follows: “*The bottleneck in production system occurs when workloads arrive at a given point more quickly than that point can handle them. The bottleneck situation causes unneeded inventory and prolongs manufacturing lead times. In a wider sense of the word, any element of a production system (machine, conveyor, AGV, buffer, labor etc.) can turn into a bottleneck*.” Li (2009) defines bottleneck as *“Bottleneck machine will often cause the upstream machines to become blocked and the downstream machines to become starved. The term ‘blocked’ means a machine has finished its operations on a part, but cannot deliver it to the downstream machine; the term ‘starved’ means the machine is idle and is awaiting parts from the upstream* *machine. A bottleneck machine will also have a shorter total blockage plus starvation time than that of adjacent machines."* It should be noted that the first definition focuses on the cycle time of the bottleneck machine, whereas, Li's quote focuses on the relationship between the bottleneck resource and the adjacent resources. Even though the two definitions are trying to solve the same problem of identifying the bottleneck, each definition approached the problem from a different perspective. According to Leporis and Králová (2010), a company's productivity is governed by analyzing the bottlenecks that exist on the production line or service. Leporis and Králová (2010) have argued: "*Both theory and practice of production management pay great attention to the bottleneck analysis in order to increase throughput of a production system, i.e. the rate at which the system generates money through sales of its products."* Currently, the simulation-based method is most commonly used by researchers to detect bottlenecks (Roser et al., 2003) (Chang et al., 2007).

Improving the throughput by solving the bottleneck dilemma requires a capital investment. Numerous evaluation methods have been developed to justify the economic benefits of such an investment for a single project. Examples include: internal rate of return, net present value, and payback. This study will focus on the net present value of the equity cash flow and minimum annual revenue requirement methods in order to determine the economic benefit of an improved system. Net present value and internal rate of return require the calculation of cash flow and time value of money and therefore provide a more realistic forecast of costs (Wood, 2010). The decision criteria would start by calculating the equity cash flow, followed by determining the minimum annual requirements. If the net present value is greater than or equal to zero for equity cash flow and minimum annual revenue requirements, then the improvement will have an economic benefit. Otherwise, the improvement would be less desirable for investors.

Previous research focused on automating the mechanism to identify bottlenecks that relied on only one method. Nevertheless, no research has been found that integrates the automation of finding bottlenecks, using multiple algorithms to locate bottlenecks, and incorporating the cost associated with the improvement. Moreover, none of the existing studies have encompassed the cost justification for the required modification on the system to meet the target demand.

This dissertation is structured into eight chapters. Chapter 2 presents the literature review of the previous researches conducted in the areas of analytical, simulation-based, optimization, and bottleneck detection approaches. In addition, it presents financial performance measurement tools that justifies the production line changes. Chapter 3 demonstrate the process of automating the bottleneck detection methods. The decision support system design is presented in Chapter 4. Chapter 5 evaluates the bottleneck detection methods based on the case study models extracted from previous literature using the decision support system developed in this dissertation. The financial justification process is illustrated in Chapter 6. Chapter 7 shows the capability of the decision support system to work with a real-world system. Finally, in Chapter 8, conclusion and future work is discussed.

Chapter 2

Literature Review

This chapter summarizes the previous works with regards to improving the productivity of a facility or service. The financial engineering aspect from previous researchers of justifying the cost for such an improvement will also be discussed. This examination of the existing research demonstrated the necessity for such study, and has informed the design process for the proposed decision support system.

2.1 Analytical Method

The analytical method is an approach whereby the problem is converted into a closed form of mathematical formulas used to analyze the problem with changing demand conditions (Kolker, 2010). However, those mathematical formulas can only be constructed for a small number of pre-defined and simplified models to produce reliable solutions (Kolker, 2010) (Law & Kelton, 1991) (Taha, 2008) (Kelton et al., 2010). According to Li (2009), the analytical method becomes very difficult when measuring the throughput of a serial production line consisting of more than two stations and a single buffer. Kelton, Sadowski, and Swets (2010) summarized the queuing theory limitation as follows: Converting the problem in queuing theory model requires the use of exponential distribution for inter-arrival time and service-time, which might not be the case for all the real-life systems. In healthcare systems, for example, the arrival rate and the number of patients to an emergency room vary from day to day, such as in the case of a car accident, where several patients might arrive in the ER at the same time (Kolker, 2010). Moreover, in order to obtain accurate results, the model must be in a steady state which is achieved by running the model for a sufficient period of time. This limits the analyst from testing the model for a pre-defined length of time if the pre-defined duration is shorter than the time required to bring the model to the steady state (Taha, 2008) (Kelton et al., 2010). For that reason, the model must warm up until it reaches the point where the service time is longer than the arrival rate in order to obtain useful results (Kelton et al., 2010). Let's talk about Kolker's (2010) example which outlines a problem wherein the average service time for two nurses (servers) and the arrival rate of patients are 0.0416 hour and 54 patients per hour respectively. The analytical method will fail since the steady state $\rho = (\lambda * \mu) / N = (54 * 0.0416) / 2 = 1.25$ is greater than 1 (Alexander Kolker 2010) (Gross & Thompson, 2008). Along with the steady state, the example assumes that the waiting space is unlimited, which is unrealistic. Adding such a variable into the equation would make the problem nearly impossible to solve (Kolker, 2010).

2.2 Simulation-based Method

The simulation-based method is another technique used to study models that measure and analyze performance. It imitates real system or service in order to provide results that help the analyst understand the environment (Taha, 2008). Readers must understand that simulation does not provide optimal solutions. An analyst must correctly interpret and translate that result into useful information (Law & McComas, 2002). Even though the steady state example provided in section 2.1 is not a complex scenario, it can be an excellent tool for comparing the differences between analytical method and simulation-based method. The example also demonstrates how the simulation-based method can overcome the analytical method's shortcomings. In fact, Bukchin (1998) declared that simulation is the only practical way of studying throughput. Because the simulation-based approach has the ability to mimic the behavior of the problem scenario, it is able to adjust the model to produce useful results. Hence, as the service time increased, so did the waiting time in the queue. This flexibility allows the simulation approach to overcome the problem of not reaching the steady state. Moreover, simulation also has the ability to produce useful results for the analyst if a limited buffer size is introduced to the model. This capability allows researchers to investigate and simulate new scenarios at low cost, and eliminates patient suffering due to having to wait in line, as was the case in the example (Kolker, 2010). Nevertheless, there are disadvantages of using the simulation-based method. Kelton, Sadowski, and Swets (2010), and Law and Kelton (1991) have agreed that the results gathered by the simulation may not be reliable as the stochastic input could produce a different output every time the model runs. In addition, the model designer must fully understand the process and the structure of the system in order to make the proper assumptions, and in turn produce a reliable model (Li, 2009). As was mentioned earlier, the analyst is responsible for determining and proposing the suitable adjustments to improve the performance. Research has shown that analysts encounter a great deal of difficulty in interpreting the simulation result (Roser et al., 2001). The model in the example presented in Roser et al., (2001) consisted of eight serial machines; each machine having a buffer of size 3. The simulation ran for 130 hours plus one hour as warm-up time. The result showed that machines #2 and #4 had workloads of 97% and 99% respectively. How can the analyst isolate the machine that is limiting the system's performance? This example proves that simulation-based methods will only mimic the system and produce results, but is not capable of providing solutions.

2.3 Simulation Optimization

Simulation optimization is a combination of analytical optimization methods' and simulation-based analysis that can be used to find the best design with the best throughput (Fu, 2002). In other words, the simulation optimization approach captures the output of the simulation model and processes it into an optimization engine to produce solution families. Finally, it feeds those solution families back to the simulation package and, based on the output, selects the optimal solution. In order to produce those solution families, the optimization engine depends on defining the objective function, which is a performance measure of the model. This model nevertheless, must run every time the objective function is needed (Better et al., 2008). In addition, noise level is a major concern associated with stochastic models. In order to reduce noise, the simulation optimization method takes advantage of the replication feature embedded in the simulation software. This feature forces the model to run the same stochastic model multiple times to calculate the average values required to produce statistical reports. Moreover, simulation optimization provides the analyst with the opportunity to experiment with the model's elements such as raw materials, resources, etc. to attain the optimal design along with the optimal flow of material in the system to increase productivity. To sum up, the simulation optimization process starts by changing the input parameters of the base model and provides a family of suggested solutions. The process carries on until the suggested solution satisfies the previously-defined objective function. However, conducting experiments for each possible solution along with the replication runs would take a tremendous amount of time and computation effort. Therefore, researchers have developed algorithms such as *Tabu* and *Scatter Search* to reduce the number of experiments based on the results of previous runs. Despite the fact that the optimization engine is not related to the simulation model, the simulation model affects the result of the optimization output. In other words, in order for the optimization engine to produce the correct result, the designer must ensure that the model reflects a valid system design. Moreover, without the correct object function the solution is useless. Wrong assumptions can cause the optimization engine to proceed down the wrong path, which will in turn yield inaccurate results.

2.4 Bottleneck Analysis

This section will describe the techniques that are currently available as bottleneck detection methods. Due to the limitation of the previous approaches, bottleneck analysis was investigated, and has been selected to be the fundamental approach for locating the resource that restricts throughput within a system. For this reason, bottleneck analysis has been proven to be the best performance measurement for throughput (Bukchin, 1998). This section highlights the weaknesses and limitations of the bottleneck detection methods to determine if they can be automated and used by the decision support system provided in this study.

The concept of bottleneck analysis was first revealed in the book “The Goal” by Dr. Eliyahu M. Goldratt, who introduced the concept of “theory of constraint” (TOC), considered to be a management tool designed to assist decision-makers to decrease inventory and increase throughput, whether in a manufacturing facility or in the service industry (Mabin & Balderstone, 2000). The main concept of this theory is that every production line has a resource that limits the system’s productivity. There have been several success stories for implementing TOC. For example, TOC’s 5-step method helped Virginia Semiconductor Inc. to become more customer-oriented and more flexible, with increased responsiveness to market change (Miller, 2000). The five steps outlined by Dr. Goldratt are:

1. Identify the system constraint.
2. Exploit the constraint.
3. Subordinate the constraint.
4. Elevate the constraint.
5. If the constraint is “broken”, return to step 1.

A need arose to establish a framework to truly evaluate the outcome of these steps. Therefore, “Step 0” was added, which identified a measured objective before continuing with the remaining steps (Dan Trietsch, 2005). A survey was conducted to evaluate the performance of companies that had implemented TOC, JIT, both, or neither. The conclusion was that companies that have adopted TOC had the highest increase in performance among all other companies who participated in the survey (Sale & Inman, 2003).

2.4.1 Percentage Utilization

In this study, utilization would be defined as when a resource is not idle due to a lack of parts (Hopp & Spearman, 2000), at which point it would be considered to be not utilized. Moreover, the blockage of a resource by the downstream resources would prevent that resource from working on the incoming parts, and hence would not be utilized (Hopp & Spearman, 2000). Nevertheless, this methodology can be applied to any resource in the model, e.g. machines, laborers, conveyors, etc. Table 2-1 depicts the state of the machine, as well as instances when it is considered both utilized and non-utilized. Even though there is no clear consensus from the research community on what should be considered as utilized state, this study will follow the definition provided by Hopp & Spearman (2000) to determine the utilized state. Therefore, readers should be aware of how the simulation packages define resources utilization in order to reach to an accurate conclusion. ProModel, for example, is a simulation package that considers blockage as part of the utilized state, and hence the simulation does not use Equation 2-1 (D’Souza, 2004).

Table 2-1 Percentage Utilization Resource States

Resource	Utilized	Non-utilized
Processing Machine	o working, o in repair, o changing tools, o serviced	o waiting for part, o blocked
AGV	o moving to a pickup location, o moving to a drop off location, o recharging, o being repaired	o waiting, o to a waiting area
Worker (human)	o working, o recovering	o waiting

$$\text{Utilization } (\rho) = \text{Arrival Rate } (\lambda) * \text{Effective Production Rate } (\mu)$$

Equation 2-1 Resource Utilization (Hopp & Spearman, 2000)

The resource with the highest utilization percentage is considered to be the bottleneck (Law & Kelton, 1991). The calculations of ideal and block time can be easily observed within the statistics that are gathered by default at the end of the simulation run in most simulation packages. Therefore, percentage utilization method is easily implemented and automated. The disadvantage comes to light when there are multiple resources with close utilization percentage, which would render it difficult to pinpoint the true bottleneck (Roser et al., 2001). To illustrate, a model used for testing consisted of eight serial machines, with every machine having a buffer of size 3. The model ran for 130 hours in addition to one hour warm-up time. The simulation output showed that machines #2 and #4 had a utilization of 97% and 99% respectively. The authors argued that there was insufficient data to clearly determine which machine caused the bottleneck (Roser et al., 2001).

2.4.2 Waiting Queue

This method identifies the bottleneck by monitoring the buffers that feed the machines in the model. The part’s waiting time in the buffers is measured. The machine that pulls a part from a buffer that has the longest waiting time would be determined to be the bottleneck (Law & Kelton, 1991). However, this method can produce inaccurate results if the buffer has a limited size. In other words, the parts would be blocked and prevented from entering the buffer if the maximum size was reached. Also, if the batch size of the parts arriving at the resources varies in the model, this would also cause an incorrect determination of the bottleneck (Roser et al., 2002) (Roser et al., 2003). In addition, the methodology cannot be implemented if the bottleneck is not a processing element (e.g. human resource or vehicle), since those types of resources do not usually have buffers (Roser et al., 2001).

There is also another viewpoint for this concept. Rather than observing the waiting time, queue length can be an indicator of the bottleneck. According to Lawrence and Buss (1995), the machine that acquires its parts from a buffer with the longest queue length would be a candidate to be identified as a bottleneck. Similar to the percentage utilization method, measuring and viewing the buffers’ size, as well as the amount of time spent on every buffer is a standard output for simulation packages nowadays. However, most real-life models have a pre-defined buffer size, which means that this is not a practical method of detecting bottlenecks.

2.4.3 Throughput-based Method

D’Souza (2004) proposed the throughput-based methodology to identify bottlenecks. The methodology focuses on measuring and comparing the throughput for every resource added to the model. Therefore, the throughput for the first run will be the raw material entering the system minus the number of parts produced by the end of the simulation run with the first resource only. The next step would be introducing the second resource to the model for the second simulation run and the new throughput will be calculated. After running the model n times, where the model contains n resources, the i^{th} workstation, which was added to the model with the highest drop-in throughput (DIT), would be considered as the bottleneck (D'Souza, 2004). The method uses Equation 2-2 to calculate DIT. The resource (i+n) is actually the resource (n) positioned downstream to resource (i) in the system. Usually, n is equal to one, however, in cases where the system consists of a concurrent process, n can be greater than one. Figure 2-1 is a flow chart diagram that explains the process of implementing the method.

$$|DIT| = \text{Throughput } (i) - \text{Throughput } (i+n)$$

Equation 2-2 Drop-in Throughput Equation (D'Souza, 2004)

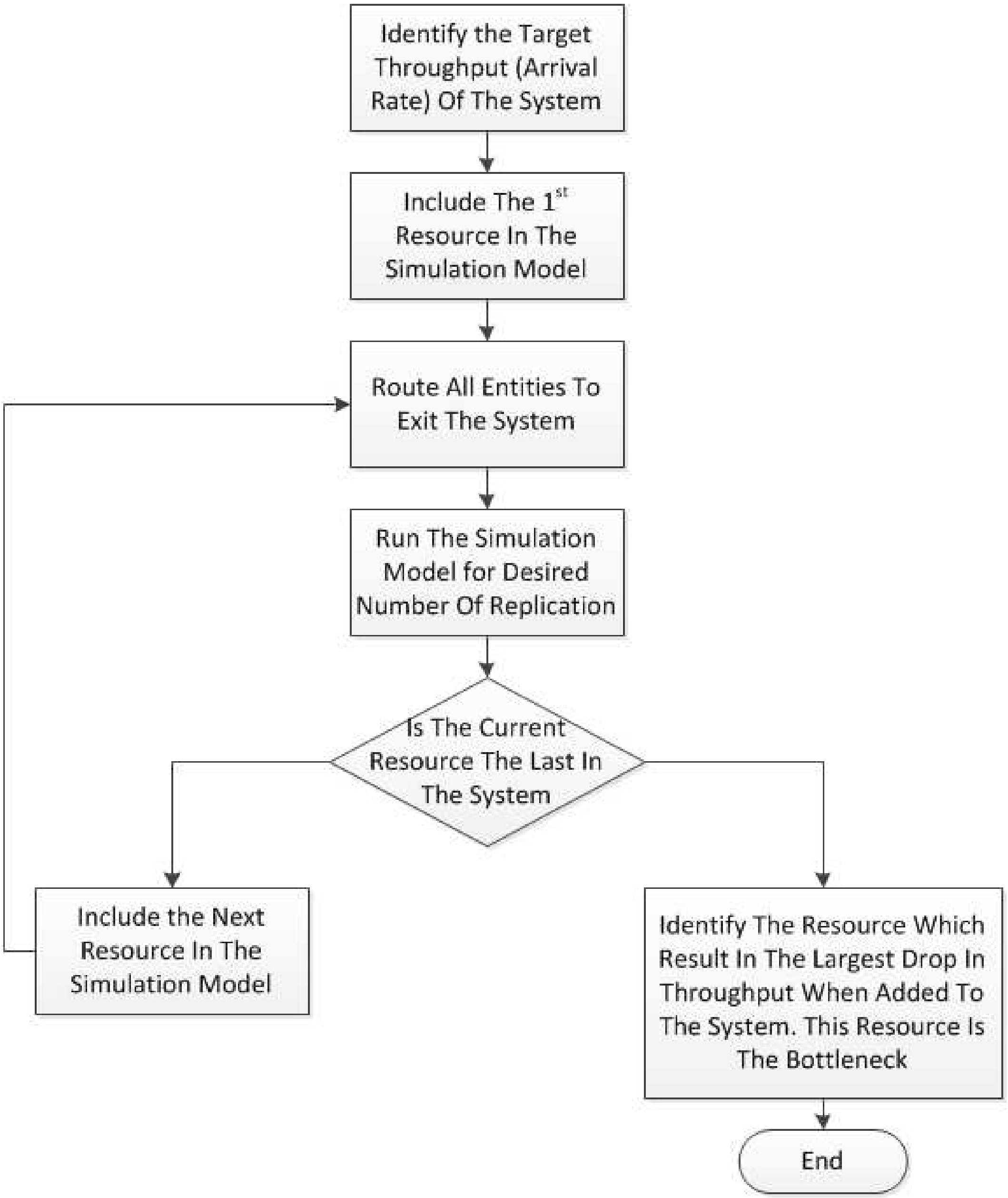


Figure 2-1 Throughput-Based Flow Chart (D'Souza, 2004)

As was pointed out by the author, the proposed method contains flaws that affect the ability to accurately detect the bottleneck resource. Firstly, the method fails to correctly identify the bottleneck if the arrival rate of the input part is high. No matter the location of the bottleneck, the method always identified the bottleneck resource as the first resource in the model. Secondly, the method did not detect all bottleneck resources in the model when the service time for the bottleneck resources had the same deterministic value (D'Souza, 2004). Our observation of this method is that it is very difficult to automate. The calculations of the throughput require manual intervention to set up the recursive models that start with one resource and finish by adding all the resources. The method depends on the simulation runs to calculate the throughput every time a resource is added. For that reason, obtaining useful information will require an extensive amount of time to compute. Moreover, the methodology becomes very difficult to implement if dynamic resources, such as labor, exist in the model. The example shown by the author placed the dynamic resources at the end of the model, which might not be the case in a real-world situation. Also, the dynamic resource was not part of the throughput calculations until all the other resources (machines) were added and their throughout was computed. However, the author had not shown how the algorithm could be applied if the dynamic resource was located in the middle of the model. As in the case of split and joint, the analyst must know when to add or ignore the branch paths. Hence, implementing this method can be expensive in terms of time, especially if the wrong assumptions were made.

2.4.4 Simulation-based Method

The simulation-based methodology proposed by Kasemset and Kachitvichyanukul in 2007 is based on the theory of constraints concept. They defined the bottleneck as the resource that has a production rate equal to or less than the required demand. The technique calculates three parameters: process utilization factor, resource utilization, and bottleneck rate. The resource with a high process utilization factor, high resource utilization, and low bottleneck rate would be a candidate for a bottleneck (Kasemset & Kachitvichyanukul, 2007). The first parameter, resource utilization, is defined as when the resource is busy. Unlike the percentage utilization method, the authors did not specifically define the word "busy". This raises the question, "*Would the resource be considered to be 'busy' if it was in the setup state*?" However, we have assumed that the authors considered the busy state only when the resource is actually processing the part, and nothing else. Also, the authors failed to provide an equation to calculate the resource utilization. The authors' quote with regards to resource utilization is as follows: "Utilization of machine / process: These data can be directly collected from simulation model. Processes or machines having high utilization are selected to be bottleneck candidates." The second parameter, utilization factor (ρ) (Equation 2-3), was defined as the arrival time (λ) divided by the departure time (μ). Bottleneck rate, Equation 2-4, is the highest long-term utilization for a resource. In other words, the number of parts processed divided by the operation time would yield the bottleneck rate (Kasemset & Kachitvichyanukul, 2007).

$$\text{Utilization factor } (\rho) = \lambda / \mu$$

Equation 2-3 Resource Utilization According to Simulation-based Method (Kasemset & Kachitvichyanukul, 2007)

$$\text{Bottleneck Rate} = \text{Number of Parts Processed} / \text{Operation Time}$$

Equation 2-4 Bottleneck Rate (Kasemset & Kachitvichyanukul, 2007)

The objective of this methodology is to improve the productivity of the bottleneck resource to be equal to or slightly higher than the demand. This methodology can be implemented as follows:

1. Identify the system constraint.
2. Exploit the system constraint.
3. Subordinate the system constraint.
4. Elevate the system constraint.
5. Repeat the process from the first step if the constraint is violated.

The above-mentioned method has been improved to the point where it can be used to identify the true bottleneck in systems with multiples bottlenecks (Kasemset & Kachitvichyanukul, 2008). The algorithm starts by calculating the three parameters: resource utilization, utilization factor, and bottleneck rate. Based on the output of the three parameters, bottleneck candidates will be identified. Several simulation experiments must be conducted to identify the true bottleneck in order to rectify the problem, thereby increasing the throughput of the system. The simulation experiments would be divided into scenarios. Each scenario would increase the capacity of each individual or pair of bottleneck candidates. Based on the simulation's output, the analyst would judge which resource(s) is/are the true bottleneck(s). After isolating the true bottleneck(s), the correct buffer size must be determined. Again, multiple simulation experiments with different buffer sizes are tested in order to correctly choose the buffer size that will increase the throughput to the target requirement. Moreover, the method relies on human interpretation of the result along with trial-and-error approach. In the illustrative example provided by Kasemset & Kachitvichyanukul (2007), Resource E, was identified as a bottleneck candidate because it satisfied all the conditions. However, increasing Resource E did not result in system improvement. The authors had to therefore re-conduct the experiment with an additional resource to find the co-bottleneck. In the end, Resource A also satisfied all the conditions, and therefore they concluded that both Resources A and E were the true bottlenecks.

2.4.5 Maximum Average Active Duration Method

This method was developed by Toyota in 2001. It classifies the resources' states as as shown in Table 2-2, with the intention of measuring the duration of uninterrupted activity state for each resource. As outlined in Figure 2-2, the resource would be considered as being in an uninterrupted activity state as long as it remains in one of the active states identified in Table 2-2, and never switches to the inactive state. The bottleneck resource would be the machine that has the longest time in an uninterrupted activity state with the minimum number of interruptions (Roser et al., 2001). The method can be summarized using the formulas listed in Equation 2-5 and Equation 2-6 to calculate the average active time. The value (n) would be the number of states switched during the simulation run.

$$A_i = \{a_{i,1}, a_{i,2}, \ldots\ldots, a_{i,n}\}$$

Equation 2-5 Set of Active Time Duration (Roser et al., 2001)

$$\overline{a}_i = \sum_{j=1}^{n} ai,j / n$$

Equation 2-6 Calculating the Average Active Duration Time (Roser et al., 2001)

The method is simple to implement as it works independently from the simulation model. In other words, and as the authors argued, that method can be applied regardless of the model structure, i.e., using either serial or parallel models. The method requires an analysis of only the log file that contains data about the resource name, duration time, and states. In addition, it has the capability of identifying bottleneck resources of any kind (i.e., machines, AGV, worker, etc.). The weakness in this technique surfaces when the model has resources with the same severity, wherein it would be difficult to identify the bottleneck with a high level of certainty (Leporis & Králová, 2010). In addition, Tamilselvan, Krishnan and Cheraghi (2010) believed that the active duration method did not always detect the correct bottleneck. They argued that due to the fact that the method focused only on the average of the longest active resource, it ignored the fundamental definition of bottleneck resource. The fundamental definition of bottleneck resource, in their opinion, is that the bottleneck resource causes the upper resources to be blocked and the downstream resource to starve.

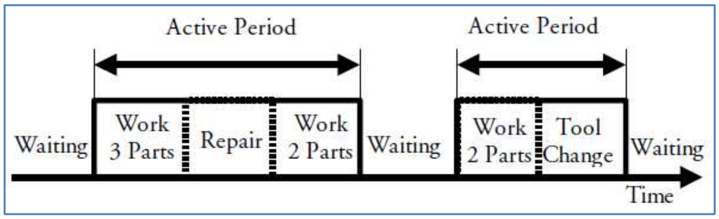


Figure 2-2 Active Duration Example (Roser et al., 2001)

Table 2-2 States Definition (Roser et al., 2001)

Machine	Active	Inactive
Processing Machine	o working, o in repair, o changing tools, o serviced	o waiting for part, o waiting for service, o blocked
AGV	o moving to a pickup location, o moving to a drop off location, o recharging, o being repaired	o waiting, o moving to a o waiting area
Human Worker	o working, o recovering	o waiting
Supply	o obtaining new part	o blocked
Output	o removing a part form the system	o waiting
Computer	o calculating	o idle
Phone Operator	o servicing customer	o waiting

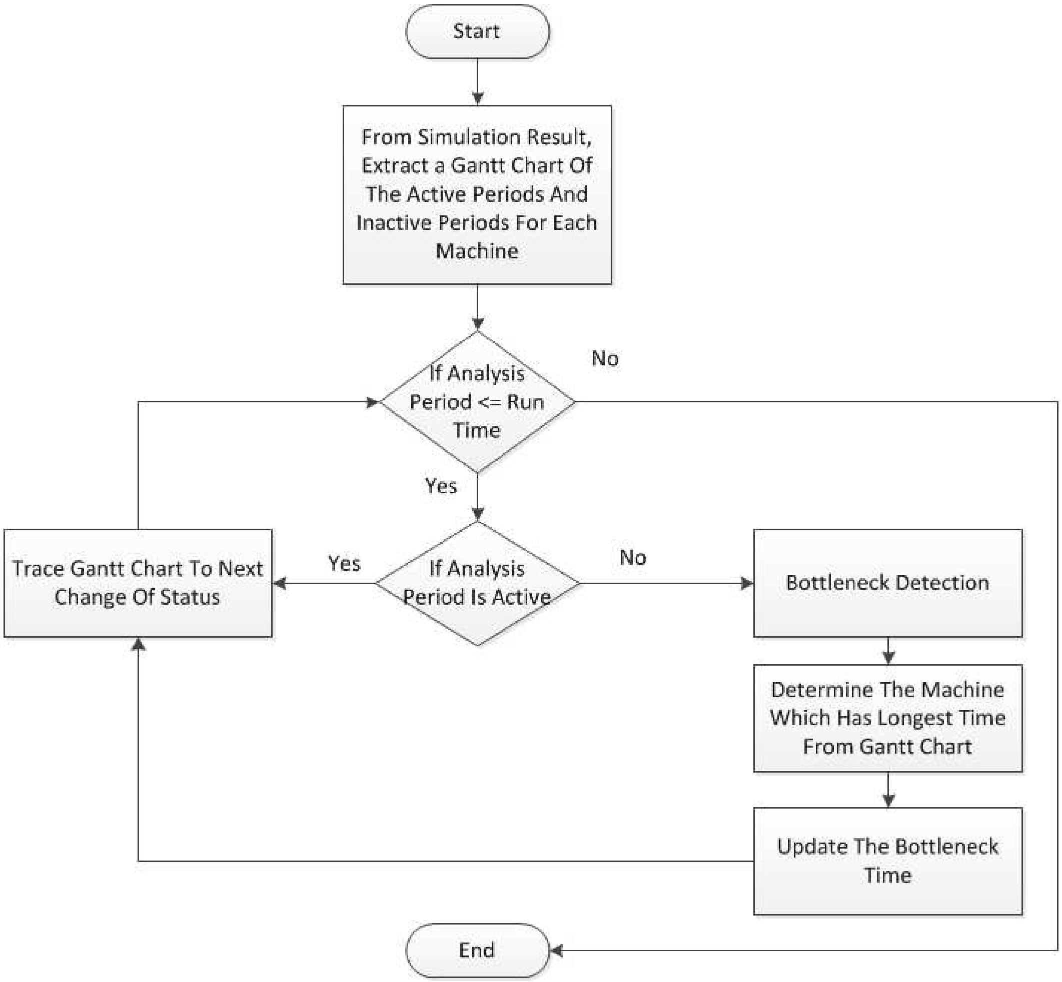


Figure 2-3 Active Duration Flow Chart (Tamilselvan , 2007)

2.4.6 Shifting Bottleneck Detection Method

This technique is a continuation of Toyota's method, active duration time, which has the has the ability to define three types of resources: non-bottleneck, shifting bottleneck, or sole bottleneck. The definition of the active and inactive states of the resources will be the same as the one described in

Table 2-2, the active duration time method, which pays close attention to the relationship between the interconnected resources. In other words, it focuses on observing and analyzing the system as a whole rather than considering a single resource compared to the percentage utilization method, for example (Roser et al., 2002). Figure 2-4 illustrates the concept behind the shifting bottleneck detection method. At the conclusion of the simulation run, the method calculates the percentage of shifting or sole bottleneck for every resource. The resource with the highest percentage of sole bottleneck would be labeled as the primary bottleneck. According to the authors, improving the sole bottleneck would definitely improve the throughput of the model. However, if the cost of improving the sole bottleneck is high, it could be more economical to enhance the throughput of the model by improving the shifting bottlenecks first, or by reducing the flow of the non-bottleneck resources (Roser et al., 2003).

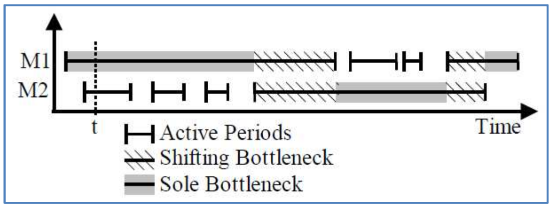


Figure 2-4 Shifting Bottleneck Example (Roser et al., 2002)

2.4.7 Arrow-based Method

This methodology works only with serial production lines (Chiang et al., 2000). The algorithm calculates the starving (ST_i) and blocking (BL_i) probabilities for each resource. In order to calculate the probabilities, the following facts should be pointed out:

- The serial production line consists of M machines and (M – 1) limited size buffers between every machine;

- Up and down are the only states for each machine. The machine would be considered as up if it produces one part per one unit of time. Otherwise, the resource would be considered as down;
- Uptime and downtime are distributed exponentially with parameters p_i and r_i respectively;
- The first machine is never starved and the last machine is never blocked. If machine m_{i-1} is not able to deliver a part to buffer b_{i-1}, then machine m_i will be considered to be starved. If machine m_{i+1} is not able to pull part from buffer b_i and buffer b_i is full, then machine m_i will be considered as blocked.

From the above information, the formulas for the starving and blockage probabilities are shown in Equation 2-7. After calculating the starving and blockage probabilities, an arrow is drawn from machine m_i to m_{i-1} if BL_i is greater than ST_{i+1}, and vice versa. The resources without arrows would be considered as the bottleneck (Chiang et al., 2000). If multiple resources are identified as the bottleneck, the resource with the largest severity would be the candidate (Biller et al., 2008). The severity can be calculated as shown in Equation 2-8.

msi = Prob ({mi-1 fails to put a part into bi-1 at time t} ∩ {bi-1 is empty at time t} ∩ {mi is up at time t})

mbi = Prob ({mi is up at time t} ∩ {bi is full at time t} ∩ {mi+1 fails to take part from bi at time t})

Equation 2-7 Starving and Blockage Probability Equations

$$S_1 = |ST_2 - BL_1|$$

$$Si = |ST_{i+1} - BL_i| + |ST_i - BL_{i-1}|, \ i=2, \ldots\ldots., M-1$$

$$SM = |ST_M - BL_{M-1}|$$

Equation 2-8 Determination of Bottleneck Severity

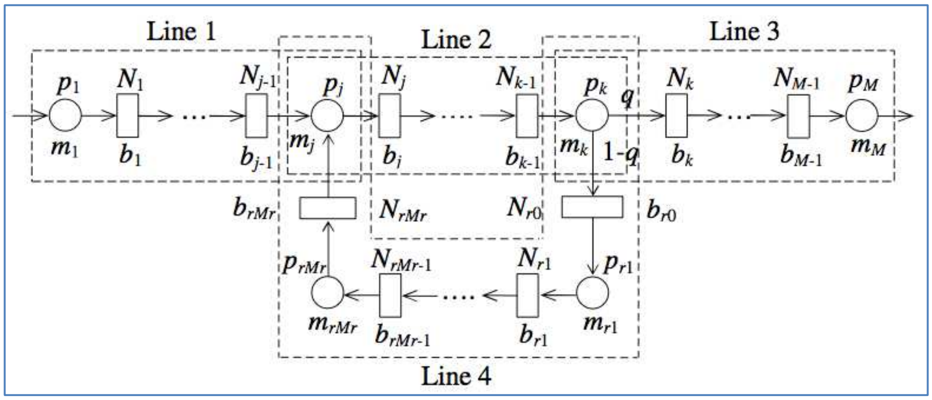


Figure 2-5 Arrow-based Method with Re-Work Segments (Biller et al., 2008)

As was stated by the authors, the arrow-based method lacks the ability to determine the true bottleneck in models other than in serial production lines. However, recent studies have been conducted to enhance the method to be applied to a production line with re-work (Biller et al., 2008). The basic idea is to divide the production line into four segments as shown in Figure 2-5. The arrow-based method will be applied to each segment, as each is considered to be a serial production line. The four identified bottlenecks will be called local bottlenecks (LBN). According to the authors, there are two rules that can generate a quick conclusion. Firstly, if the LBN was an overlap of a resource that existed in three out of the four segments, it confirms that it is definitely the global bottleneck (GBN) for the model. Secondly, if the LBN is an overlapping resource that existed in only one segment, it is definitely not the GBN. If those conditions are not satisfied, then the throughput of the model needs to be measured based on the bottleneck candidates' LBNs. The candidate with the most effect on the throughput would be identified as the GBN.

In summary, this method is not capable of detecting labor or AGV bottlenecks. Moreover, it can be applied only to serial production lines. There is a need for manual intervention, which would be difficult to automate, and a requirement to divide the model into segments in order to allow the method to work successfully with models involving re-work. For those reasons, this study will not pursue this method as it is unsuitable for analyzing real-world cases.

2.4.8 Turning Point Method

The turning point method examines the bottleneck resources as the machine that usually blocks the upstream resources and forces the downstream resources to be idle, i.e., waiting on parts. Therefore, the bottleneck resource is the busiest resource compared to those adjacent to it (Li et al., 2009). The busiest resource can be identified if it satisfies the formulas in Equation 2-9. TB_j represents the blockage time for the j^{th} machine, whereas TS_j is the starvation time for the j^{th} machine. Moreover, j-1 represents the index of the nearest upstream machine, and j+1 is the index of the nearest downstream machine.

$$(TBi - TSi) > 0 \text{ where } 1 < i < j-1$$

$$(TBi - TSi) < 0 \text{ where } j+1 < i < n$$

$$TBj + TSj < TBj\text{-}1 + TSj\text{-}1 \text{ where } j \mathrel{!}= 1 \text{ or } n$$

$$TBj + TSj < TBj+1 + TSj+1 \text{ where } j \mathrel{!}= 1 \text{ or } n$$

$$\text{If } j=1\text{: } (TB1 - TS1) > 0 \ \& \ (TB2 - TS2) < 0 \ \& \ (TB1 + TS1) < (TB2 + TS2)$$

$$\text{If } j=n\text{: } (TBn\text{-}1 - TSn\text{-}1) > 0 \ \& \ (TBn - TSn) < 0 \ \& \ (TBn + TSn) < (TBn\text{-}1 + TSn\text{-}1)$$

Equation 2-9 Turning Point Definition (Li, 2009)

The article written by Li et al., (2009) provides examples of applying the method on a serial production line without labor or AGV. The advantage of this method is that it can quickly pinpoint the bottleneck by either acquiring data from the simulation run or by real-time observation of a facility, since the method is concerned with only starving and blockage time (Li, 2009) (Leporis & Králová, 2010). Also, the method uses an indexing mechanism to rank the bottleneck candidates in the event that the method identifies several bottlenecks. The method was improved to solve the problem of having concurrent and feedback loops in the system (Li et al., 2009). In order to solve the concurrent process problem, the authors have combined them and represented them as a single virtual resource. Therefore, the new production line will be translated into a serial production line with virtual resources, as shown in Figure 2-6. After the conversion, the turning point method can be applied to the new serial system and the method will have the capability of identifying the bottleneck. A bottleneck identified as the new virtualized resource signifies that the bottleneck exists among those concurrent processes, which must be analyzed further. As of feedback quality loop, the system path can be identified as either the main path or the feedback path. The user must consider the location of the resources that links the main path to the feedback path, as shown in Figure 2-7. The method starts by analyzing the main path. If results show that the joint resources are not the bottleneck for the main path, this means that the feedback loop has less impact on the throughput of the entire system and thus the main path would be the focus of the solution. On the other hand, if the joint resources happen to be the bottleneck in the main path, then the feedback loop has an impact on the throughput of the entire system and hence requires enhancement in order to improve the throughput of the system (Li, 2009).

As mentioned above, most of the analysis depends on the analyst's ability to group the concurrent resources and to define the joint resources between the main path and the feedback loops, rendering this process very difficult to automate. Moreover, any incorrect assumptions could lead to a wrong conclusion. In addition, the method is not capable of finding labor or AGV bottlenecks. For those reasons, we have again elected not to pursue this method to detect bottlenecks if the model has concurrent process, labor, or feedback.

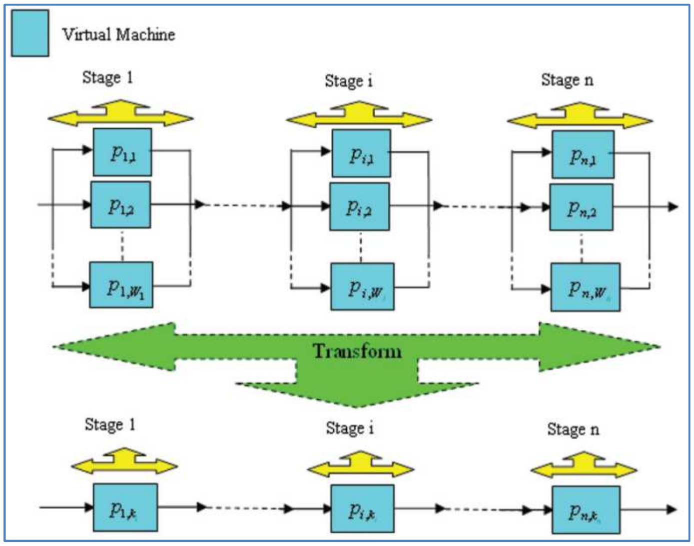


Figure 2-6 Concurrent Process Transformation (Li, 2009)

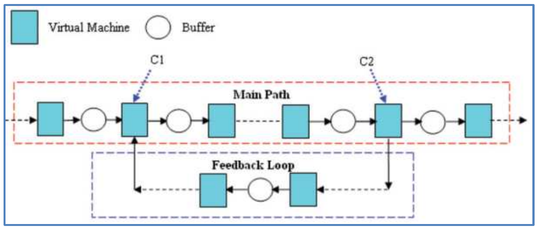


Figure 2-7 Feedback Example (Li, 2009)

2.4.9 Critical Indicator Method

Leporis and Králová (2010) define a bottleneck as the resource that restricts the productivity of the whole system. Their approach is based on finding the “critical indicator” for each resource, which would be calculated based on the averages of utilization, starving, blocking, and labor waiting rate of the entire system. Those averages can be obtained from most of the output reports obtained from the simulations. Equation 2-10 shows how the critical indicator KR_i is calculated. The variables’ notation would be as follows: B_i would be the average utilization rate for the i^{th} machine (busy); I_i would be the average starvation rate for the i^{th} machine (idle); Bl_i would be the average blocking rate for the i^{th} machine (blocked); and L_i would be the average waiting rate for workers (labor) for the i^{th} machine.

After computing the critical indicator for every resource, the bottleneck would be identified as the resource with the lowest indicator value among all the resources (Leporis & Králová, 2010). According to the authors, this method is capable of detecting the true bottleneck regardless of the model (serial or concurrent) or the resource type (machine, AGV, or labor). However, the example in their paper lacks the details necessary to replicate the model for further analysis. Moreover, Leporis and Králová (2010) claimed that the method is simple to automate.

$$KR_i = [\,(\sum B_i / n) - B_i] + [I_i - (\sum I_i/n)] + [Bl_i - (\sum Bl_i/n)] + [L_i - (\sum L_i/n)]$$

Equation 2-10 Critical Indicator Equation

2.4.10 Inactive Duration Method

This methodology is the opposite of the active duration and the shifting bottleneck methods that were discussed earlier. Tamilsevan et al. (2010) argued that active duration and shifting bottleneck methods occasionally failed to detect the correct bottleneck. Their methodology uses the same mechanism in acquiring the needed data to make the decision, i.e., the definition of the active/inactive states for the resources. This method defines the bottleneck as the resource which forces the downstream resources to starve and the upstream to be blocked. Once the simulation run is complete, a processing time chart is required to detect the bottleneck resource based on comparing the active and inactive states between the resources in the model (Karthikeyan, 2010). The algorithm’s procedure is shown in Figure 2-8.

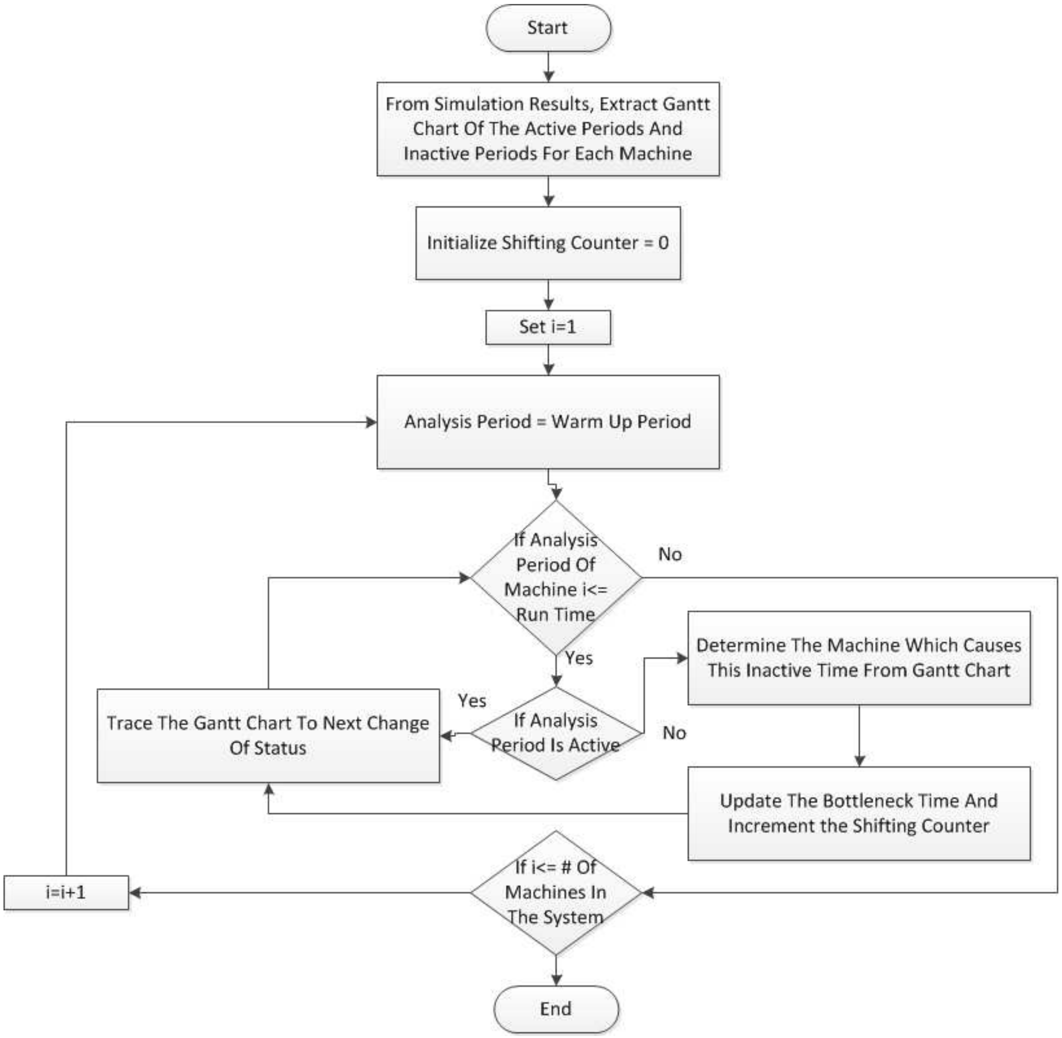


Figure 2-8 Inactive Duration Procedure (Tamilselvan , 2007)

There are also five characteristic measurements to be considered while conducting the analysis. The first four were presented in Tamilselvan et al., (2010). Firstly, the bottleneck time ratio (α), shown in Equation 2-11, calculates the relationship between the bottleneck time and the total run time for the simulation run. Secondly, the bottleneck ratio (□), shown in Equation 2-12, indicates the percentage of the bottlenecks' machines in the model compared to the total number of machines in the model. Thirdly, the bottleneck resource can shift from one resource to another, which can be caused by the random variation in the system. For that reason, bottleneck shifting frequency (Φ) measurement was introduced, which sums up the total number of times that the bottleneck shifts from one resource to another, as shown in Equation 2-13. Finally, bottleneck severity ratio (X) measures the relation between the number of times a bottleneck occurs and the total number of inactive states, as outlined in Equation 2-14. The purpose of those ratios is to help the analyst understand the system. For instance, a high value of bottleneck severity ratio with a bottleneck shifting frequency of 0 indicates that there is a sole bottleneck every time a machine is in the inactive state. Additionally, if the bottleneck severity ratio is low, this is an indication that the system is not fully utilized (Tamilselvan et al., 2010).

$$\alpha = \text{Bottleneck Time} / \text{Total Run Time}$$

Equation 2-11 Bottleneck Time Ratio

$$\square = \text{Number of Bottleneck Machines} / \text{Total Number of Machines}$$

Equation 2-12 Bottleneck Ratio

$$\Phi = 1 – (\text{Total Number of Bottleneck Machines} / \text{Total Number of Bottleneck Shifts})$$

Equation 2-13 Bottleneck Shifting Frequency

$$X = \text{Number of Inactive States with Bottleneck} / \text{Total Number of Inactive States}$$

Equation 2-14 Bottleneck Severity Ratio

$$\beta = 1 – (C_v / \sqrt{n})$$

Equation 2-15 Shiftiness Measure

In addition to the four characteristic measures, Tamilsevan et al. (2010) utilized one additional measure to enhance their understanding of the model. The bottleneck shifting measure is used to indicate the number of bottlenecks that exist in the model. If the ratio (β) is zero, this signifies that there is only one bottleneck resource. On the other hand, a ratio of one indicates that all the resources in the model are bottlenecks. C_v, in Equation 2-15, represents the coefficient variance for bottleneck probability for resources in the model, the number of which is represented by n (Lawrence & Buss, 1994).

2.4.11 Inter-departure Time

The proposed methods studied machines only, and classify the resource's states into four categories:

1) Cycle state (busy)
2) Blocked-up state (idle)
3) Blocked-down state (blocked)
4) Fail state

The procedure to identify and rank the bottleneck starts by collecting the inter-departure time for each resource for a specific period of time. Secondly, the failure time is noted for each resource. Next, the failure cycle is eliminated from the data collected in the first step, and finally, the blocked-up time is added to the blocked-down time. The resource with the lowest blocked-up and blocked-down time would be considered as the bottleneck. Based on this rule, the remaining resources can be ranked as the secondary bottlenecks (Sengupta et al., 2008). The authors argued that measuring the blocked-up and blocked-down percentages will produce better results compared to measuring the cycle and failed percentages. The supporting numerical example presented in their article provided a base model with four configurations, and the results were presented in two different formats: The blocked-up and blocked-down results were presented in minutes, whereas the cycle and fail results were presented in percentages. For example, the bottleneck in the first configuration was M-4 (fourth machine), which had 96.1% in cycle and fail states. For 102.2 minutes, M-4 was in blocked-up and blocked-down states. Based on these findings during a simulation run time of 600 minutes, M-4 spent 17% of its time in blocked-up and blocked-down states. This violates the first rule quoted by the authors “*The sum of percentage residence time in different states must be equal to 100%.”* Moreover, it was not possible to replicate the job shop model referred to in the paper as the deterministic cycle times of the machines were not mentioned in the article. Since the discussed method lacks the ability to detect resources other than machines, it was decided that this method would not be pursued in this study.

2.5 Financial Engineering Concepts

One of the basic concepts of economy is to compare apples to apples, wherein projects must be evaluated using unbiased tools. In this study, the influence of the development project is assumed to be financial only, with no other effects. For example, raising the capacity of a certain production line may affect marketing and logistics, and could also cause price elasticity. This study will assume that the annual production rate and operating costs would be the same in all years.

In order to construct an analysis to determine the economic benefits of a project, calculating equity cash flow and minimum annual revenue requirements would be the first step. After that, further analysis could be conducted by calculating net present value (NPV).

The definition of cash flow is *“the algebraic sum of money and its estimated flow in and out of a company over a certain period of time as a result of a particular project”* (Stevens, 1994). There are three types of net cash flow: total, equity and operating. For practicality, this dissertation will apply equity cash flow throughout the analysis. To calculate the equity cash flow, there are values that must be provided, as shown in Equation 2-16. The parameter's explanation is as follows (Stevens, 1994):

- G_j represents the yearly gross income.
- I_j represents the interest payment for year j.
- D_j represents the tax depreciation amount for year j.
- T represents the incremental tax rate.
- K_j represents the value of the total capital expenditure in year j.
- L_j represents the salvage value that can be obtained at the end of the depreciation period.
- P_j represents the principal payment of the money borrowed for year j.
- W_j represents the net increase (or decrease) in working capital.
- V_j represents the investment tax credit in year j.

$$X_{ej} = (G_j - C_j - I_j) - (G_j - C_j - I_j - D_j)\,T - K_j + L_j - P_j +- W_j + V_j$$

Equation 2-16 Equity Cash Flow Equation

NPV is the sum of the net cash flow discounted at the minimum acceptable rate of return (MARR) to year zero. The value of MARR is related to the best practices within each industry. For decision-making purposes, the project can be considered desirable if the NPV is greater than or equal to zero (Michel, 2001).

The second evaluation tool that will be used in this study is the minimum annual revenue requirement, an approach that considers the unrecovered capital investment. For decision-making purposes, the project can be considered desirable if the NPV is positive (Stevens, 1994). Equation 2-17, Equation 2-18, Equation 2-19, and Equation 2-20 show the equations used to calculate the minimum annual revenue requirements and the subsequent equations. Equation 2-21 depicts how the net present value can be calculated to evaluate a project. Badiru and Russell (1987) stated that the project would be desirable if the gross income was greater than the minimum annual revenue requirement. Using this logic, the minimum selling price can be determined, which would maintain profitability. This can be achieved by substituting NPV_x with 0 in Equation 2-21, while knowing the quantity of production. Using this combined information and rearranging the equation, determining the selling price can be easily calculated. This dissertation will be using the approach in Equation 2-21 to suggest a selling price for the marketing department. The parameter's explanation for the equations are as follows (Stevens, 1994):

- R_j represents the minimum annual revenue requirement for year j.
- D_{bj} represents the total capital recovered in year j.
- F_{ej} represents the return on equity in year j.
- I_j represents the debt interest cost in year j.
- C_j represents the annual cost in year j.
- t_j represents the tax paid in year j.
- K_e represents the minimum required return.
- B_{j-1} represents the book value for year j-1.
- c represents debt ratio.
- k_d represents cost debt capital.
- T represents incremental tax rate.

$$R_j = D_{bj} + F_{ej} + I_j + C_j + t_j$$

Equation 2-17 Minimum Annual Revenue Requirement Equation

$$F_{ej} = (1 - c)\,(k_e)\,(B_{j-1})$$

Equation 2-18 Return on Equity Equation

$$I_j = (c)\ (k_d)\ (B_{j-1})$$

Equation 2-19 Debt Interest Cost Equation

$$t_j = T/(1-T)\ (D_{bj} + Fe_j - D_j)$$

Equation 2-20 Tax Paid Equation

$$NPV_x = (1-T) \sum (G_j - R_j)\ (P/K\ k_x,j)$$

Equation 2-21 Net Present Value for Minimum Annual Revenue Requirement

Chapter 3

Implementation Of Bottleneck Detection Methods

The literature review showed that the analytical approach lacks the ability to solve complex real-life models. It also indicated that the simulation-based approach does not follow a solid methodology to detect bottlenecks, and depends solely on the analyst's interpretation to locate the bottleneck. The shortcoming of the simulation optimization approach rises from the concern of the computational resources and the time spent in solving a wide range of solutions, as well as the noise associated with it. The literature review concerning locating the bottleneck concluded that the theory of constraints would be the most suitable approach (Chang et al., 2007) (Roser et al., 2003). In addition, eleven methods related to the bottleneck analysis approach were identified: percentage utilization (Law & Kelton, 1991), waiting time queue (Law & Kelton, 1991), throughput-based indicator (D'Souza, 2004), simulation-based procedure (Kasemset & Kachitvichyanukul, 2007), maximum average active duration (Roser et al., 2001), shifting bottleneck (Roser et al., 2002), inactive duration (Tamilselvan , 2007), arrow-based indicator (Kuo et al., 1996), turning point (Li et al., 2009), critical indicator (Leporis & Králová, 2010), and inter-departure time (Sengupta et al., 2008). Based on the strengths and weaknesses identified in the literature review, a decision support system that applies these algorithms was developed. As for financial justification, two financial measurement tools were identified. The net present value of the equity cash flow, and minimum revenue requirement, would indicate the desirability of implementing the proposed solution. The financial aspect of the problem will be discussed in details in Chapter 6.

The objective of this chapter is to demonstrate the process path that was followed to implement the bottleneck detection methods. Based on the conclusion arrived at in chapter two, the following methods have been automated: percentage utilization, waiting queue, turning point, maximum average active duration, inactive duration, critical indicator, shifting bottleneck, and simulation-based methods. In order to evaluate these methods, nine production line models were constructed:

1) Serial Production Line With Bottleneck Located at the Beginning.
2) Serial Production Line With Bottleneck Located at the End.
3) Serial Production Line With Bottleneck Located in the Middle.
4) Serial Production Line With Single Labor Bottleneck.
5) Serial Production Line With Multiple Bottlenecks.
6) Serial Production Line With Multiple Labor Bottlenecks.
7) Production Line With Concurrent Process.
8) Production Line With Feedback - Bottleneck Located in Main Path.
9) Production Line With Feedback - Bottleneck Located in Feedback Path.

3.1 Serial Production Line With Bottleneck Located at the Beginning

The objective of this scenario is to consider the first machine in the model as the bottleneck. The objective can be achieved by configuring the first machine with the highest cycle time. This configuration will force downstream machines to starve and the first buffer to fill up. The model consists of four single machines connected in a series, with unlimited raw material to supply it. The cycle time for each machine is 15, 10, 10, and 10 minutes respectively. The inter-arrival rate of the raw material is one part per one minute. In addition, no labor exists in the model. For simplicity, the model does not have setup, downtime, or warm-up time. The simulation run time is 100 hours and the model has four buffers of size ten. The first buffer was added to model in order to be able to count the number of parts waiting to be processed by Machine001, and to note the duration of time in the waiting queue (an essential component of the waiting queue method). The system will reach the steady state after 45 minutes. Therefore, running the model for 100 hours will produce a realistic result.

Table 3-1 Specs for Model Where Bottleneck is Located at the Beginning

Element	Description
Raw Material Arrival Rate	One part per minute
Machine	Four single machines, Cycle times: 15, 10, 10, and 10 minutes, respectively.
Buffer	Four buffers of size 10
Setup	N/A
Breakdown	N/A
Labor	N/A
Simulation Run Time	100 hours
Warm-up Time	N/A

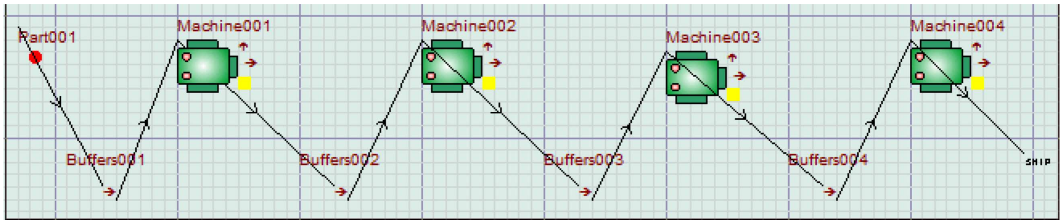


Figure 3-1 Model Where Bottleneck is Located at the Beginning

Table 3-2 Machines Statistics

Resource	%Idle	%Busy	%Blocked	%Waiting On Labor
Machine001	0.00	100.00	0	0.00
Machine002	33.50	66.50	0	0.00
Machine003	33.58	66.42	0	0.00
Machine004	33.67	66.33	0	0.00

Table 3-3 Buffers Statistics

Buffer Name	Average Size	Average Time
Buffers001	9.92	145.23
Buffers002	0	0
Buffers003	0	0
Buffers004	0	0

The system produced 398 parts in 100 hours. The percentage utilization method declared that Machine001 is clearly the bottleneck, Table 3-2. Also, Buffers001 has the maximum size and part waiting time, Table 3-3. Therefore, the waiting queue method suggests that the bottleneck is Machine001, since Buffers001 is its sole input source. The critical indicator was able to identify the correct bottleneck as Machine001, which returned the lowest KR_i value, Table 3-4. The average active duration method also identified Machine001 as the bottleneck, since it was the most active machine during the simulation run, Table 3-5. According to the simulation-based method, Machine001 had the highest utilization, highest utilization factor, and lowest bottleneck rate, Table 3-6, and therefore, Machine001 was the bottleneck. Because Machine001 was the least inactive machine among all the resources, the inactive duration method also confirmed that it was the bottleneck. The shifting bottleneck method showed that Machine001 entered the active state at time 0 and did not exit that state until the end of the simulation run. As for turning point, the conditions for the bottleneck at the beginning of the model are: $(TB_1 – TS_1) > 0$, $(TB_2 – TS_2) < 0$, and $(TB_1 + TS_1) < (TB_2 + TS_2)$. In this scenario, the values for TB_1, TB_2, and TS_1 are all 0 and TS_2 is 33.2, as shown, in Table 3-2, and hence, the first two conditions were violated. The method returned a false result as there is no equal sign in the inequalities.

Table 3-4 Critical Indicator Method Summary

Resource	**Idle**	**Busy**	**Blocked**	**Waiting on Labor**	**KR_i**
Machine001	-25.19	-25.19	0.0	0.0	-50.38
Machine002	8.31	8.31	0.0	0.0	16.62
Machine003	8.40	8.40	0.0	0.0	16.80
Machine004	8.50	8.50	0.0	0.0	17.00

Table 3-5 Active Duration Method Summary

Resource	Total Active Time	Number of Switches	Average Active Time
Machine001	6000.00	1	6000.00
Machine002	3990.00	400	9.99
Machine003	3985.00	399	9.99
Machine004	3980.00	398	10.00

Table 3-6 Simulation-based Method Summary

Resource	% Utilization	Utilization Factor	Bottleneck Rate
Machine001	100	0.07	0.07
Machine002	66.50	0.07	0.10
Machine003	66.42	0.07	0.10
Machine004	66.33	0.07	0.10

Table 3-7 Inactive Duration Method Summary

Resource	Total Inactive Time	Number of Switches	Average Inactive Time
Machine001	0.00	1	0.00
Machine002	2010.00	400	9.99
Machine003	2015.00	399	5.05
Machine004	2020.0	399	5.06

3.2 Serial Production Line With Bottleneck Located at the End

In this scenario, the bottleneck will be placed at the end of the production line by configuring the last machine in the model with the highest cycle time. This configuration will force the upstream machines to be blocked. The model has four single machines connected in serial fashion, and there is unlimited raw material to supply the model. The cycle times for the machines are 10, 10, 10, and 15 minutes respectively. In addition, the model does not have labor, setup, downtime, or warm-up time, and has a run time of 100 hours. The model has three buffers of size 10. The system is required to run for 45 minutes to produce the first part.

Table 3-8 Specs for Model Where Bottleneck is Located at the End

Element	Description
Raw Material Arrival Rate	Unlimited supply
Machine	Four single machines, Cycle times: 10, 10, 10, and 15 minutes, respectively.
Buffer	Three buffers of size 10
Setup	N/A
Breakdown	N/A
Labor	N/A
Simulation Run Time	100 hours
Warm-up Time	N/A

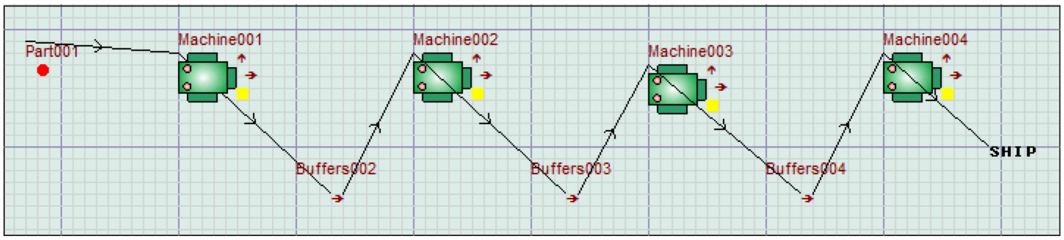


Figure 3-2 Model Where Bottleneck is Located at the End

Table 3-9 Machines Statistics

Resource	%Idle	%Busy	%Blocked	%Waiting On Labor
Machine001	0.00	71.83	28.17	0.00
Machine002	0.17	70.00	29.83	0.00
Machine003	0.33	68.17	31.50	0.00
Machine004	0.50	99.50	0.00	0.00

Table 3-10 Buffers Statistics

Buffer Name	Average Size	Average Time
Buffers002	8.70	121.11
Buffers003	9.20	131.43
Buffers004	9.70	142.30

The system produced 398 parts in 100 hours. The percentage utilization method concluded that Machine004 was clearly the bottleneck, Table 3-9. Also, Buffers004 has the maximum size and part waiting time, Table 3-10 and therefore, the method suggests that the bottleneck is Machine004, since it is the only resource that is fed by Buffers004. The critical indicator was able to correctly identify the bottleneck as Machine004, which returned the lowest KR_i value (results shown in Table 3-11). The average active duration method also points to Machine004 as the bottleneck, as it was the most active machine during the simulation run, Table 3-12. According to the simulation-based method, Machine004 has the highest utilization, highest utilization factor, and lowest bottleneck rate, Table 3-14, again confirming that Machine004 is the bottleneck. Machine004 was the least inactive machine among all the resources, therefore, it is the bottleneck according to the inactive duration method, Table 3-14. As for the shifting bottleneck method, although Machine004 entered the active state after 30 minutes and never changed its state, it caused the other machines to be blocked. For that reason, Machine004 was identified as the sole bottleneck in the model. According to the turning point method, the conditions are for the bottleneck located at the end of the model: $(TB_{n-1} – TS_{n-1}) > 0$, $(TB_n – TS_n) < 0$, and $(TB_n + TBS_n) < (TB_{n-1} + TS_{n-1})$. In this scenario, $TB_{n-1}=31.50$, $TB_n=0.00$, $TS_{n-1}=0.33$, and $TS_n=0.50$ as shown in Table 3-9. Hence, all conditions were satisfied and the method returned correct results.

Table 3-11 Critical Indicator Method Summary

Resource	Idle	Busy	Blocked	Waiting on Labor	KR_i
Machine001	-0.25	5.54	5.79	0.00	11.08
Machine002	-0.08	7.38	7.46	0.00	14.75
Machine003	0.08	9.21	9.13	0.00	18.42
Machine004	0.25	-22.13	-22.38	0.00	-44.25

Table 3-12 Active Duration Method Summary

Resource	Total Active Time	Number of Switches	Average Active Time
Machine001	4310.00	339	12.71
Machine002	4200.00	359	11.70
Machine003	4090.00	379	10.80
Machine004	5970.00	1	5970.00

Table 3-13 Simulation-based Method Summary

Resource	% Utilization	Utilization Factor	Bottleneck Rate
Machine001	71.83	0.07	0.10
Machine002	70.00	0.07	0.10
Machine003	68.17	0.07	0.10
Machine004	99.50	0.07	0.07

Table 3-14 Inactive Duration Method Summary

Resource	Total Inactive Time	Number of Switches	Average Inactive Time
Machine001	1690.00	339	5.00
Machine002	1800.00	359	5.01
Machine003	1910.00	379	5.04
Machine004	30.0	1	30.0

3.3 Serial Production Line With Bottleneck in the Middle

Machine002 was chosen to have the highest cycle time in this scenario, which caused the upstream machine to be blocked and the downstream machines to starve. The model is comprised of four single machines with cycle times of 10, 15, 10 and 10 minutes respectively, and the rate of the raw material flow is unlimited. The model has three buffers of size 10 units, and does not include labor, setup, breakdown, or warm-up time. The simulation run time of 100 hours.

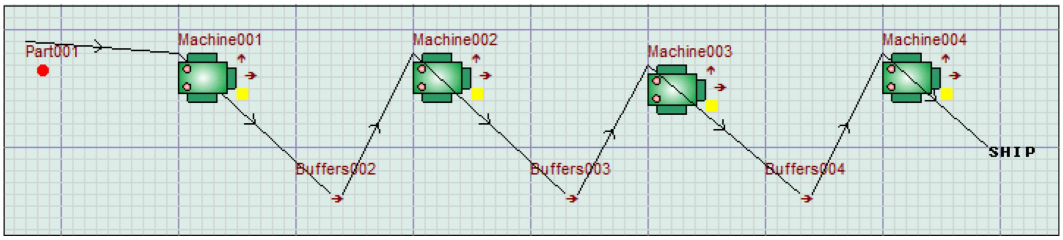


Figure 3-3 Model Where Bottleneck is Located in the Middle

Table 3-15 Specs for Model Where Bottleneck is Located in the Middle

Element	Description
Raw Material Arrival Rate	Unlimited supply
Machine	Four single machines, Cycle times: 10, 10, 10, and 15 minutes, respectively.
Buffer	Three buffers of size 10
Setup	N/A
Breakdown	N/A
Labor	N/A
Simulation Run Time	100 hours
Warm-up Time	0

Table 3-16 Machines Statistics

Resource	Idle	Busy	Blocked	Waiting On Labor
Machine001	0.00	68.42	31.58	0.00
Machine002	0.17	99.83	0.00	0.00
Machine003	33.58	66.42	0.00	0.00
Machine004	33.67	66.33	0.00	0.00

Table 3-17 Buffers Statistics

Buffer Name	Average Size	Average Time
Buffers002	9.73	142.44
Buffers003	0.00	0.00
Buffers004	0.00	0.00

The system produced 398 parts in 100 hours. The percentage utilization method identified Machine002 as clearly being the bottleneck, Table 3-16. Also, Buffers002 has the maximum size and the part waiting time, Table 3-17, which is indicative of the fact that the bottleneck is Machine002, since it is the only resource that is fed by Buffers002. The critical indicator was able to correctly identify the bottleneck as Machine002, since it returned the lowest KR_i value, Table 3-18. The average active duration method also concluded that Machine002 was the bottleneck as it was the most active machine during the simulation run, Table 3-19. According to the simulation-based method, Machine002 had the highest utilization, highest utilization factor, and lowest bottleneck rate, Table 3-20; therefore, Machine002 is declared as the bottleneck. Machine002 was the least inactive machine among all the resources, therefore, it is deemed to be the bottleneck according to the inactive duration method, Table 3-21. The shifting bottleneck method also identified Machine002 as the bottleneck, as it was active for 5990 minutes. Equation 2-9 shows the conditions for determining the bottleneck for the turning point method, in which all conditions were satisfied and hence, Machine002 is the bottleneck.

Table 3-18 Critical Indicator Method Summary

Resource	**Idle**	**Busy**	**Blocked**	**Waiting on Labor**	**KR_i**
Machine001	-16.85	6.83	23.69	0.00	13.67
Machine002	-16.69	-24.58	-7.90	0.00	-49.12
Machine003	16.73	8.83	-7.90	0.00	17.67
Machine004	16.81	8.92	-7.90	0.00	17.83

Table 3-19 Active Duration Method Summary

Resource	Total Active Time	Number of Switches	Average Active Time
Machine001	4105.00	380	10.80
Machine002	5990.00	1	5990.00
Machine003	3985.00	399	9.99
Machine004	3980.00	398	10.00

Table 3-20 Simulation-based Method Summary

Resource	% Utilization	Utilization Factor	Bottleneck Rate
Machine001	68.42	0.07	0.10
Machine002	99.83	0.07	0.07
Machine003	66.42	0.07	0.10
Machine004	66.33	0.07	0.10

Table 3-21 Inactive Duration Method Summary

Resource	Total Inactive Time	Number of Switches	Average Inactive Time
Machine001	1895.00	380	4.99
Machine002	10.00	1	10.00
Machine003	2015.00	399	5.05
Machine004	2020.0	399	5.06

3.4 Serial Production Line With Single Labor Bottleneck

In this scenario, labor will be a shared resource among all machines. Workers must be present at the machine in order to process a part, which will confirm that the bottleneck will be labor. If a sufficient number of laborers are working throughout the system, waiting on labor time will be significantly reduced. The model consists of four single machines connected in a series. There is unlimited supply for raw material. The cycle time for all the machines is 10 minutes, and there are three buffers of size 10 units, as shown in Figure 3-4. There is no setup or breakdown in the model, and the simulation run time is 100 hours, with no warm-up time required.

Table 3-22 Specs for Model Where Bottleneck Is Labor

Element	Description
Raw Material Arrival Rate	Unlimited supply
Machine	Four single machines, Cycle time: 10 minutes for all machines
Buffer	Three buffers of size 10
Setup	N/A
Breakdown	N/A
Labor	One shared Laborer
Simulation Run Time	100 hours
Warm-up Time	N/A

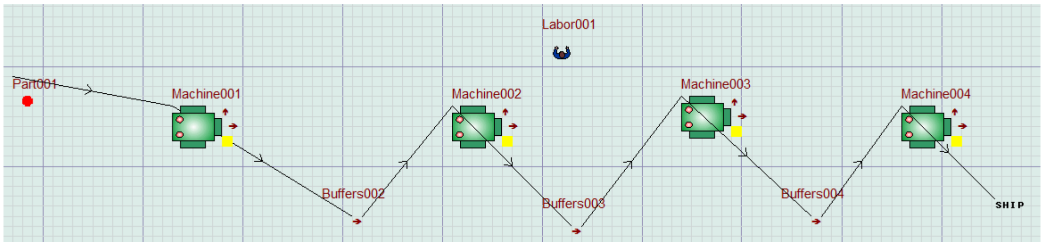


Figure 3-4 Model Where Bottleneck Is One Single Laborer

Table 3-23 Machines Statistics

Resource	%Idle	%Busy	%Blocked	%Waiting On Labor
Machine001	0.00	25.17	0.00	74.83
Machine002	25.17	25.00	0.00	49.83
Machine003	25.17	25.00	0.00	49.83
Machine004	25.50	24.83	0.00	49.67

Table 3-24 Buffers Statistics

Buffer Name	Average Size	Average Time
Buffers002	0.00	0.00
Buffers003	0.00	0.00
Buffers004	0.00	0.00

Table 3-25 Labor Statistics

Labor Name	Busy	Idle
Laboor001	100	0

The system produced 149 parts in 100 hours. The percentage utilization method determined that Machine001 and Labor001 were 100% utilized,

Table 3-23 and that all buffers have zero average waiting time and average size, Table 3-24. Therefore, the method suggests that there is no bottleneck in the model. The critical indicator was able to identify the bottleneck as Machine001, which returned the lowest KR_i value, Table 3-26. The critical indicator method mentioned that it was worth analyzing labor associated with any bottleneck machine that was identified as a potential bottleneck. The average active duration method also concluded that Machine001 and Labor001 were the bottlenecks because they were the most active resources during the simulation run, Table 3-27. The reader should keep in mind that the method considers 'Waiting On Labor' as a utilized time. According to the simulation-based method, Machine001 is the bottleneck, because it had the highest utilization since all the resources have equal values for the other parameters, Table 3-28. Labor001 and Machine001 were the least inactive among all the resources, therefore they are considered to be shared bottlenecks according to the inactive duration method, Table 3-29. According to bottleneck definition provided by Roser, Nakano, and Tanaka (2002), waiting on labor is considered active time, and consequently, Machine001 and Labor001 are both considered 100% active. The shifting bottleneck method, therefore, also identified Machine001 and Labor001 as shared bottlenecks. Turning point identified Machine001 as the bottleneck, however as was stated in Chapter 2, turning point lacks the ability to identify labor as a bottleneck.

Table 3-26 Critical Indicator Method Summary

Resource	Idle	Busy	Blocked	Waiting on Labor	KR_i
Machine001	-18.96	-0.17	0.0	18.79	-0.33
Machine002	6.21	0.0	0.0	-6.21	0
Machine003	6.21	0.0	0.0	-6.21	0
Machine004	6.54	0.17	0.0	-6.37	0.33

Table 3-27 Active Duration Method Summary

Resource	Total Active Time	Number of Switches	Average Active Time
Machine001	6000.00	1	6000.00
Machine002	4490.00	151	29.74
Machine003	4490.00	150	29.93
Machine004	4470.00	150	29.80
Labor001	6000.00	1	6000.00

Table 3-28 Simulation-based Method Summary

Resource	% Utilization	Utilization Factor	Bottleneck Rate
Machine001	100.00	0.025	0.10
Machine002	74.83	0.025	0.10
Machine003	74.83	0.025	0.10
Machine004	74.50	0.025	0.10

Table 3-29 Inactive Duration Method Summary

Resource	Total Inactive Time	Number of Switches	Average Inactive Time
Machine001	0.00	1	0.00
Machine002	1510.00	151	10.00
Machine003	1510.00	151	10.00
Machine004	1530.00	150	10.20
Labor001	0.00	1	0.00

3.5 Serial Production Line With Multiple Bottlenecks

In this scenario, the two middle machines will have the highest cycle time among all machines in the model. This will ensure that the upstream machine will be blocked, whereas the downstream machine will starve. Both bottleneck machines need to be enhanced in order to improve the production line. The model has four single machines, and the cycle time for the machines are 10, 15, 15, 10 minutes respectively. There are three buffers of size 10. Furthermore, there is no setup time, breakdown, warm-up, or labor as shown in Table 3-30.

Table 3-30 Specs for Model With Multiple Bottlenecks

Element	**Description**
Raw Material Arrival Rate	Unlimited Supply
Machine	Four single machines, Cycle times: 10, 15, 15, and 10 minutes respectively
Buffer	Three buffers of size 10
Setup	N/A
Breakdown	N/A
Labor	N/A
Simulation Run Time	100 hours
Warm-up Time	N/A

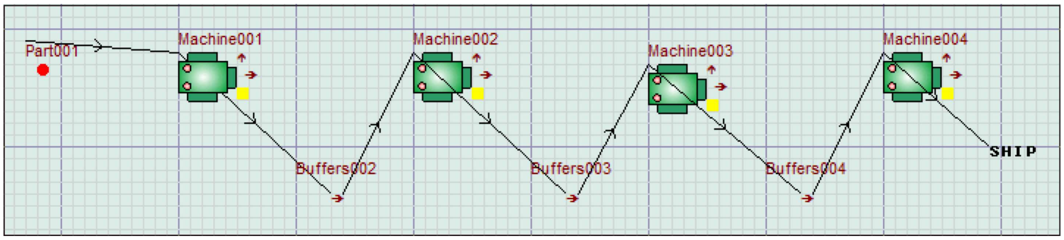


Figure 3-5 Model With Multiple Bottlenecks – Base

Table 3-31 Machines Statistics

Resource	%Idle	%Busy	%Blocked	%Waiting On Labor
Machine001	0.00	68.42	31.58	0.00
Machine002	0.17	99.83	0.00	0.00
Machine003	0.42	99.58	0.00	0.00
Machine004	33.75	66.25	0.00	0.00

Table 3-32 Buffers Statistics

Buffer Name	Average Size	Average Time
Buffers002	9.73	142.44
Buffers003	0.00	0.00
Buffers004	0.00	0.00

The system produced 398 parts in 100 hours. Machine002 and Machine003 have a very a very similar utilization, Table 3-31, and according to the percentage utilization method, these two resources are co-bottlenecks in the model. Also, Buffer002 has the maximum size and part waiting time,

Table 3-32, which suggests that the bottleneck is solely Machine002. It was concluded that the waiting queue suggested Machine002 first, as it is the first bottleneck in the model. The parts did not pile up in front of Machine003 since it was able to handle the flow coming from Machine002. Once Machine002 is enhanced, we anticipated that the production line would jam in front of Machine003 without improving the overall throughput. The critical indicator method was able to identify Machine002, as it returned the lowest KR_i values, Table 3-33. The method specifically stated that the bottleneck is the resource with smaller KR_i, however, Machine003 is very close compared to Machine002. The average active duration method also linked Machine002 and Machines003 as the bottlenecks. Machine002 and Machine003 were the most active machines during the simulation run, Table 3-34. According to the simulation-based method, Machine002 and Machine003 have the highest utilization, high utilization factor, and lowest bottleneck rate, Table 3-35. Because both Machine002 and Machine003 have identical ratios, they are considered as bottleneck candidates. It should be noted that neither Machine002 nor Machine003 had the highest utilization factor, however, according to the method, they are the bottleneck candidates since two conditions were satisfied. Machine002 was the least inactive machine among all the resources, therefore, Machine002 is the bottleneck according to inactive duration method, Table 3-36. Machine002 and Machine003 were 99.83% and 99.58% active, respectively, during the simulation run, and are therefore considered to be the largest shared bottleneck resources in the model. Based on the turning point condition, Equation 2-9, Machine002 satisfied all the conditions, and was consequently concluded to be the bottleneck. As for Machine003, condition $(TB_i – TS_i) > 0$ was violated, TB_i=0.00 and TS_i=0.17, hence Machine003 was not detected as a bottleneck.

Table 3-33 Critical Indicator Method Summary

Resource	Idle	Busy	Blocked	Waiting on Labor	KR_i
Machine001	-8.58	15.10	23.69	0.00	30.21
Machine002	-8.42	-16.31	-7.80	0.00	-32.62
Machine003	-8.17	-16.06	-7.90	0.00	-32.12
Machine004	25.17	17.27	-7.90	0.00	34.54

Table 3-34 Active Duration Method Summary

Resource	Total Active Time	Number of Switches	Average Active Time
Machine001	4100.00	380	10.80
Machine002	5985.00	1	5985.00
Machine003	5970.00	1	5970.00
Machine004	3970.00	398	9.97

Table 3-35 Simulation-based Method Summary

Resource	% Utilization	Utilization Factor	Bottleneck Rate
Machine001	68.42	0.0683	0.10
Machine002	99.83	0.0665	0.0667
Machine003	99.58	0.0663	0.0667
Machine004	66.25	0.0662	0.10

Table 3-36 Inactive Duration Method Summary

Resource	Total Inactive Time	Number of Switches	Average Inactive Time
Machine001	1895.00	380	4.99
Machine002	10.00	1	10.00
Machine003	25.00	1	25.00
Machine004	2025.00	398	5.09

Since most methods identified Machine002 as the bottleneck and the model was configured to have two bottlenecks, let's consider the model when only Machine002 was enhanced.

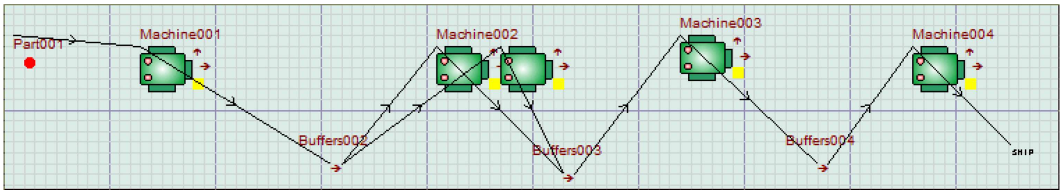


Figure 3-6 Model With Multiple Bottlenecks – 1st Enhancement

Table 3-37 Machines Statistics

Resource	%Idle	%Busy	%Blocked	%Waiting On Labor
Machine001	0.00	70.25	29.75	0.00
Machine002	1.50	51.33	47.17	0.00
Machine002	1.58	51.25	47.17	0.00
Machine003	0.42	99.58	0.00	0.00
Machine004	33.75	66.25	0.00	0.00

Table 3-38 Buffers Statistics

Buffer Name	Average Size	Average Time
Buffers002	9.18	130.88
Buffers003	9.71	142.42
Buffers004	0.00	0.00

The system produced 397 parts in 100 hours. The percentage utilization method declared Machine003 is the bottleneck in the model, Table 3-37. Because Buffers003 has the maximum size and part waiting time, Table 3-38, it was concluded by the waiting queue that Machine003 is the true bottleneck since it blocked Machine002 and caused it to fill Buffers002. This can be shown by comparing Table 3-31 to Table 3-37. The critical indicator method was able to identify the correct bottleneck as Machine003, which returned the lowest KR_i values, Table 3-39. The average active duration method also showed Machines003 as the bottleneck, as it was the most active machine during the simulation run, Table 3-40. According to the simulation-based method, Machine003 had the highest utilization, high utilization factor, and lowest bottleneck rate, Table 3-41, which confirms Machine003 as the bottleneck candidate. Machine003 was the least inactive machine among all the resources, therefore, it is the bottleneck according to the inactive duration method, Table 3-42.Machine003 was 99.58% active during the simulation run, and according to the shifting bottleneck method, it is considered to be the largest sole bottleneck resources in the model. Based on the turning point condition in Equation 2-9, Machine003 satisfied all the conditions, therefore, it was concluded to be the bottleneck.

Table 3-39 Critical Indicator Method Summary

Resource	Idle	Busy	Blocked	Waiting on Labor	KR_i
Machine001	-7.44	-2.51	4.93	0.00	-5.03
Machine002	-5.94	16.40	22.35	0.00	32.80
Machine002	-5.86	16.48	32.96	0.00	32.96
Machine003	-7.03	-31.84	-24.81	0.00	-63.69
Machine004	26.30	1.48	-24.81	0.00	2.96

Table 3-40 Active Duration Method Summary

Resource	Total Active Time	Number of Switches	Average Active Time
Machine001	4210.00	358	11.75
Machine002	170.00	33	5.15
Machine003	5970.00	1	5970.0
Machine004	3970.00	398	9.97

Table 3-41 Simulation-based Method Summary

Resource	% Utilization	Utilization Factor	Bottleneck Rate
Machine001	70.25	0.07	0.10
Machine002	51.25	0.03	0.06
Machine002	51.33	0.03	0.06
Machine003	99.58	0.06	0.06
Machine004	66.25	0.06	0.10

Table 3-42 Inactive Duration Method Summary

Resource	Total Inactive Time	Number of Switches	Average Inactive Time
Machine001	1785.00	358	4.98
Machine002	5825.00	34	171.3
Machine003	25.00	1	25.00
Machine004	2025.00	398	5.08

3.6 Serial Production Line With Multiple Labor Bottlenecks

The objective of this scenario is to evaluate the methods that will be able to detect multiple labor bottlenecks. The same scenario used in section 3.4 will be used here. The only difference is that Machine001 and Machine002 will be operated by only one worker. A second worker will be responsible for Machine003 and Machine004, as shown in Figure 3-7.

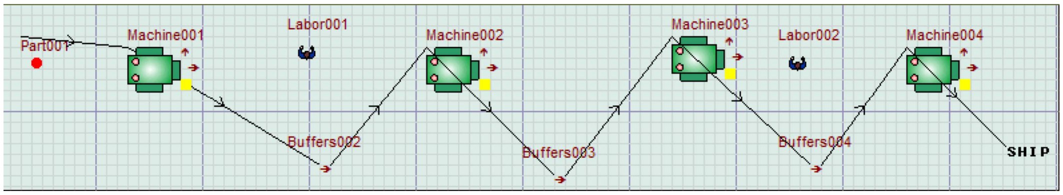


Figure 3-7 Multiple Labor Bottleneck Model

Table 3-43 Specs for Model With Multiple Laborers as Bottleneck

Element	Description
Raw Material Arrival Rate	Unlimited supply
Machine	Four machines, Cycle times: 10, 10, 10, and 10 minutes respectively
Buffer	Three buffers of size 10
Breakdown	N/A
Setup	N/A
Labor	Two labor resources
Simulation Run Time	100 hours
Warm-up Time	N/A

Table 3-44 Machines Statistics

Resource	%Idle	%Busy	%Blocked	%Waiting On Labor
Machine001	0.00	50.00	0.00	50.00
Machine002	50.00	50.00	0.00	0.00
Machine003	50.17	49.83	0.00	0.00
Machine004	50.17	49.83	0.00	0.00

Table 3-45 Buffers Statistics

Buffer Name	Average Size	Average Time
Buffers002	0.00	0.00
Buffers003	0.00	0.00
Buffers004	0.00	0.00

Table 3-46 Labor Statistics

Labor Name	Busy	Idle
Laboor001	100	0
Laboor002	99.67	0.33

The system produced 299 parts in 100 hours. The percentage utilization method noted that Machine001 and Labor001 were 100% utilized, Table 3-44 and Table 3-46. All buffers have zero average waiting time and average size, and therefore the method concluded that no bottleneck exists in the model. The critical indicator method identified the bottleneck as Machine001 and Machine002, which returned the lowest KR_i value, Table 3-47. The author of the method indicated that the labor associated with a resource with the smallest KR_i could be a factor of being the bottleneck resource in the model. The average active duration method also shows Machine001 and Labor001 as the bottlenecks as they were the most active resources during the simulation run, Table 3-48. According to the simulation-based method, Machine001 is the bottleneck, because it has the highest utilization since all the resources have equal values for the other parameters,

Table 3-49. Labor001 and Machine001 were the least inactive among all the resources, therefore, they are the bottlenecks according to the inactive duration method, Table 3-50. Machine001 and Labor001 were 100% shared bottleneck resources during the simulation. Therefore, those two resources will be considered as the true bottlenecks in the model. The turning point method also identified Machine001 as the bottleneck, however the method is not designed to locate labor-related bottlenecks.

Table 3-47 Critical Indicator Method Summary

Resource	Idle	%Busy	Blocked	Waiting on Labor	KR_i
Machine001	-37.58	-0.08	0.0	37.50	-0.17
Machine002	12.42	-0.08	0.0	-12.50	-0.17
Machine003	12.58	0.08	0.0	-12.50	0.17
Machine004	12.58	0.08	0.0	-12.50	0.17

Table 3-48 Active Duration Method Summary

Resource	Total Active Time	Number of Switches	Average Active Time
Machine001	6000.00	1	6000.00
Machine002	3000.00	1	3000.00
Machine003	4490.00	150	29.93
Machine004	4470.00	150	29.80
Labor001	6000.00	1	6000.00
Labor002	5980.00	1	5980.00

Table 3-49 Simulation-based Method Summary

Resource	% Utilization	Utilization Factor	Bottleneck Rate
Machine001	100.00	0.05	0.10
Machine002	50.00	0.05	0.10
Machine003	49.83	0.05	0.10
Machine004	49.83	0.05	0.10

Table 3-50 Inactive Duration Method Summary

Resource	Total Inactive Time	Number of Switches	Average Inactive Time
Machine001	0.00	1	0.00
Machine002	3000.00	301	9.97
Machine003	3010.00	300	10.03
Machine004	3010.00	300	10.03
Labor001	0.00	1	0.00
Labor002	20.00	1	20.00

3.7 Production Line With Concurrent Process

The objective of this scenario is to have a bottleneck resource located in a parallel process in the production line. The system has unlimited supply of raw material that feeds directly to the first machine. Machine001 processes the parts and places them in the first buffer. Machine002 and Machine005 will be pulling parts from Buffers001, as shown in Figure 3-8, and will push the parts ahead to the next machine. Machine003 and Machine006 will send the processed parts to Buffers004. Finally, Machine004 will pick up the parts and prepare them for shipment. It should be noted that the size of the buffers is 10 units, and the cycle times are 10, 25, 25, 10, 50, and 10 minutes respectively.

Table 3-51 Specs for Model With Concurrent Process

Element	Description
Raw Material Arrival Rate	Unlimited Supply
Machine	Six machines, Cycle times:10, 25, 25, 10, 50, and 10 minutes respectively
Buffer	Four buffers of size 10
Setup	N/A
Breakdown	N/A
Labor	N/A
Simulation Run Time	100 hours
Warm-up Time	N/A

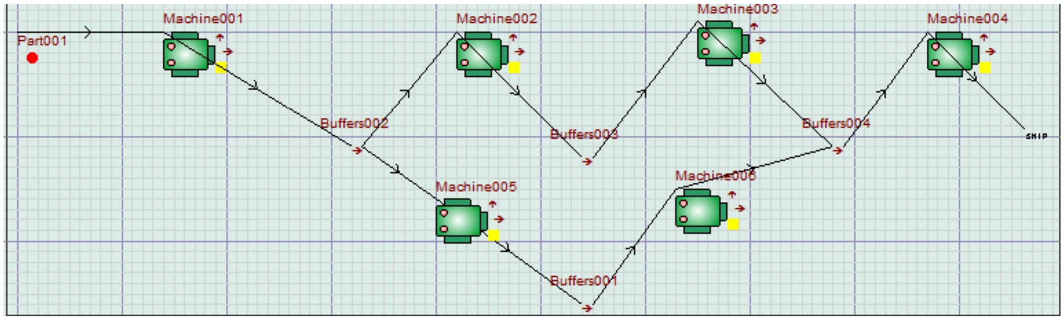


Figure 3-8 Model With Concurrent Process

Table 3-52 Machines Statistics

Resource	%Idle	%Busy	%Blocked	%Waiting On Labor
Machine001	0.00	61.75	38.25	0.00
Machine002	0.33	99.67	0.00	0.00
Machine003	0.75	99.25	0.00	0.00
Machine004	40.58	59.42	0.00	0.00
Machine005	0.17	99.83	0.00	0.00
Machine006	80.17	19.83	0.00	0.00

Table 3-53 Buffers Statistics

Buffer Name	Average Size	Average Time
Buffers001	0.00	0.00
Buffers002	9.77	158.45
Buffers003	0.00	0.00
Buffers004	0.20	3.33

The system produced 356 parts in 100 hours. The percentage utilization method showed that Machine005 had the highest utilization in the model, Table 3-25. It should be noted that the utilization of Machine002 and Machine003 were very close to that of Machine005, illustrating one of the limitation of the percentage utilization method. In this study, only the resource with the highest utilization was selected. Because Buffer002 had the maximum size and part waiting time, Table 3-53, the method suggests that the bottleneck is either Machine002 or Machine005, as Buffers002 is the input source for both machines. Critical indicator was able to identify the bottleneck as Machine005, as it returned the lowest KR_i value, Table 3-54. The average active duration method also pointed to Machine005 as the bottleneck as it was the most active machine during the simulation run, Table 3-55. According to the simulation-based method, Machine005 has the highest utilization and lowest bottleneck rate, Table 3-56, and was consequently the bottleneck candidate. It should be noted that Machine005 did not have the highest utilization factor and based on this fact, the bottleneck candidate might not satisfy all conditions. Machine001 was the least inactive machines among all the resources. Therefore, Machine001 was identified as the bottleneck, Table 3-57. Machine005 was the sole bottleneck in the model ten minutes after the simulation ran until the end, confirming that Machine005 is the bottleneck in the model. The result of turning point was ignored since the method requires a manual intervention to create virtual machines, which is not part of the scope of this study.

Table 3-54 Critical Indicator Method Summary

Resource	Idle	Busy	Blocked	Waiting on Labor	KR_i
Machine001	-20.33	11.54	31.88	0.00	23.08
Machine002	-20.00	-26.38	-6.38	0.00	-52.75
Machine003	-19.58	-25.96	-6.38	0.00	-51.92
Machine004	20.25	13.87	-6.38	0.00	27.75
Machine005	-20.17	-26.54	-6.38	0.00	-53.08
Machine006	59.83	53.46	-6.38	0.00	106.92

Table 3-55 Active Duration Method Summary

Resource	Total Active Time	Number of Switches	Average Active Time
Machine001	3135.00	287	10.92
Machine002	5975.00	1	5975.00
Machine003	5975.00	1	5975.00
Machine004	3560.00	238	14.96
Machine005	5985.00	1	5985.00
Machine006	1190.00	119	10.00

Table 3-56 Simulation-based Method Summary

Resource	% Utilization	Utilization Factor	Bottleneck Rate
Machine001	61.75	0.06	0.10
Machine002	99.67	0.04	0.04
Machine003	99.25	0.04	0.04
Machine004	59.42	0.06	0.10
Machine005	99.83	0.02	0.02
Machine006	19.83	0.02	0.10

Table 3-57 Inactive Duration Method Summary

Resource	Total Inactive Time	Number of Switches	Average Inactive Time
Machine001	2860.00	288	9.93
Machine002	20.00	1	20.00
Machine003	45.00	1	45.00
Machine004	2435.00	238	10.23
Machine005	10.00	1	10.00
Machine006	4805.00	120	40.04

3.8 Production Line With Feedback – Bottleneck In Main Path

The model was configured, as shown in Table 3-58. The objective of the model is to have the bottleneck placed in the main path of the production line. It consists of five machines and five buffers. Machine001 has two feeds; the first feed comes from the new raw materials arriving at the system, and the second feed comes from Buffers005, which holds parts that were re-worked. Machine003 inspects the parts. If a part passes inspection, it will be sent to Buffers004. If it fails, it will be sent to the re-work line, as shown in Figure 3-9. It is expected that 10% of the parts will require re-work.

Table 3-58 Specs for Model With Feedback

Element	Description
Raw Material Arrival Rate	Unlimited Supply
Machine	Five single machines, Cycle times: 10, 60, 10, 10, 10 minutes respectively.
Buffer	Five buffers of size 10
Rework	10%
Setup	N/A
Breakdown	N/A
Labor	N/A
Simulation Run Time	100 hours
Warm-up Time	N/A

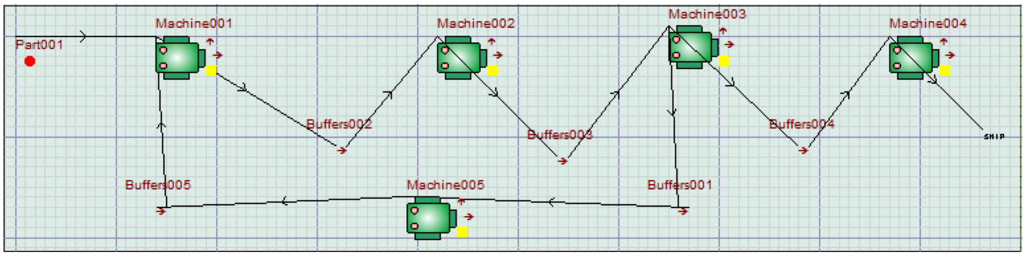


Figure 3-9 Model With Feedback Process (Bottleneck In Main Path)

Table 3-59 Machines Statistics

Resource	%Idle	%Busy	%Blocked	%Waiting On Labor
Machine001	0.00	18.50	81.50	0.00
Machine002	0.17	99.83	0.00	0.00
Machine003	83.50	16.50	0.00	0.00
Machine004	85.83	14.17	0.00	0.00
Machine005	97.67	2.33	0.00	0.00

Table 3-60 Buffers Statistics

Buffer Name	Average Size	Average Time
Buffers001	0.00	0.00
Buffers002	9.88	539.09
Buffers003	0.00	0.00
Buffers004	0.00	0.00
Buffers005	0.09	40.00

The system produced 85 parts in 100 hours. The percentage utilization method identified identified Machine002 as the bottleneck in the model,

Table 3-59. Buffers002 has the maximum size and part waiting time, Table 3-60 and as a result, the method suggests that the bottleneck is solely Machine002. Critical indicator was able to identify the bottleneck as Machine002 since it returned the lowest KR_i value, Table 3-61. The average active duration method also showed Machine002 as the bottleneck, because it was the most active machine during the simulation run, Table 3-62. According to the simulation-based method, Machine002 had the highest utilization, highest utilization factor, and lowest bottleneck rate, Table 3-63, and is therefore positively the bottleneck. Machine002 was the least inactive machine among all the resources, and is identified to be the bottleneck according to the inactive duration method, Table 3-64. At the beginning, Machine001 and Machine002 were shared bottlenecks. However, 140 minutes into the run time, Machine002 became the sole bottleneck in the system. Therefore, Machine002 would be the bottleneck in the system. The output of turning point showed that Machine002 satisfied the conditions of the bottleneck. Since Machine002 is not a joint point to the feedback loop, the feedback loop has less impact to enhance the model according to the turning point method.

Table 3-61 Critical Indicator Method Summary

Resource	Idle	Busy	Blocked	Waiting on Labor	KR_i
Machine001	-53.43	11.76	65.20	0.00	23.53
Machine002	-53.26	-69.56	-16.30	0.00	-139.13
Machine003	30.06	13.76	-16.30	0.00	27.53
Machine004	32.39	16.09	-16.30	0.00	32.19
Machine005	44.23	27.93	-16.30	0.00	55.86

Table 3-62 Active Duration Method Summary

Resource	Total Active Time	Number of Switches	Average Active Time
Machine001	3060.00	146	20.95
Machine002	5960.00	1	5960.00
Machine003	140.00	14	10.00
Machine004	990.00	99	10.00
Machine005	850.00	85	10.00

Table 3-63 Simulation-based Method Summary

Resource	% Utilization	Utilization Factor	Bottleneck Rate
Machine001	18.50	0.01	0.10
Machine002	99.83	0.01	0.01
Machine003	16.50	0.01	0.10
Machine004	14.16	0.01	0.10
Machine005	2.33	0.00	0.10

Table 3-64 Inactive Duration Method Summary

Resource	Total Inactive Time	Number of Switches	Average Inactive Time
Machine001	2910.00	147	19.79
Machine002	10.00	1	10.00
Machine003	4980.00	100	49.80
Machine004	5120.00	86	59.53
Machine005	5830.00	15	388.66

3.9 Production Line With Feedback – Bottleneck in Feedback Path

The model was configured as shown in Figure 3-10. The objective of the model was to have the bottleneck placed in the feedback loop path of the production line, which consists of five machines and five buffers. Machine001 has two feeds; the first feed comes from the new raw materials coming to the system, and the second feed arrives from Buffers005, which holds parts that were re-worked. Machine003 inspects the parts; if the part passes the inspection, it will be sent to Buffers004, but if it does not pass, it will be sent to the re-work line as shown in Figure 3-9. We expect to have 10% of the parts re-worked.

Table 3-65 Specs for Model With Feedback

Element	Description
Raw Material Arrival Rate	Unlimited Supply
Machine	Five single machines, Cycle times: 10, 10, 10, 10, 30 minutes respectively.
Buffer	Five buffers of size 10
Rework	10%
Setup	N/A
Breakdown	N/A
Labor	N/A
Simulation Run Time	100 hours
Warm-up Time	N/A

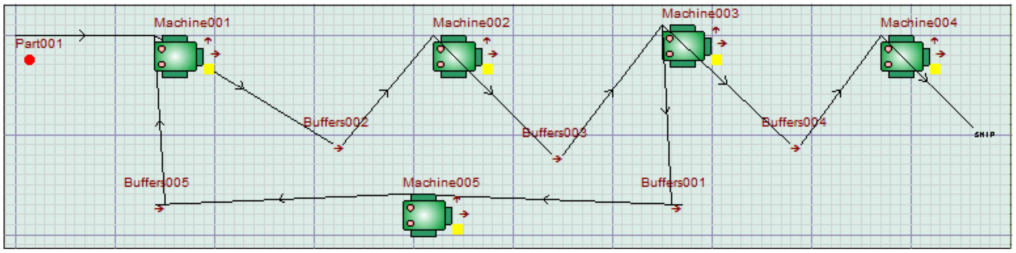


Figure 3-10 Model With Feedback Process (Bottleneck In Feedback Path)

Table 3-66 Machines Statistics

Resource	%Idle	%Busy	%Blocked	%Waiting On Labor
Machine001	0.00	89.00	11.00	0.00
Machine002	0.17	87.17	12.67	0.00
Machine003	0.33	85.33	14.33	0.00
Machine004	49.00	51.00	0.00	0.00
Machine005	1.17	98.83	0.00	0.00

Table 3-67 Buffers Statistics

Buffer Name	Average Size	Average Time
Buffers001	8.34	244.00
Buffers002	7.55	84.81
Buffers003	8.16	93.63
Buffers004	0.00	0.00
Buffers005	0.00	0.00

The system produced 306 parts in 100 hours. The Percentage Utilization method labeled Machine005 as the bottleneck, Table 3-66. Also, Buffers001 has the maximum size and part waiting time, Table 3-67, and thus the method suggests that the only bottleneck is Machine005. The critical indicator method was able to identify the bottleneck as being Machine005 because it returned the lowest KR_i values, Table 3-68. The average active duration method also shows Machine005 as the bottleneck because it was the most active machine during the simulation run, Table 3-69. According to the simulation-based method, Machine005 has the highest utilization and lowest bottleneck rate, Table 3-70, thus it is the bottleneck candidate. Although Machine005 did not have the highest utilization factor, it is still considered as a bottleneck candidate. Machine001 was the least inactive machine among all the resources, therefore, it is the bottleneck according to the inactive duration method, Table 3-71. After the first 190 minutes into simulation run, Machine005 became the sole bottleneck in the model. The result from turning point was ignored since the conditions are not applicable to this configuration; the conditions assume that there are multiple machines in the feedback loop. However, in this configuration, there is only one machine in the feedback loop path, which is Machine005, and therefore, the result was ignored. This proves to be another limitation of the turning point method. In summery turning point and the inactive duration methods failed to identify the true bottleneck.

Table 3-68 Critical Indicator Method Summary

Resource	Idle	Busy	Blocked	Waiting on Labor	KR_i
Machine001	-10.13	-6.73	3.40	0.00	-13.46
Machine002	-9.96	-4.90	5.06	0.00	-9.80
Machine003	-9.80	-3.06	6.73	0.00	-6.13
Machine004	38.86	31.26	-7.60	0.00	62.53
Machine005	-8.96	-16.56	-7.60	0.00	-33.13

Table 3-69 Active Duration Method Summary

Resource	Total Active Time	Number of Switches	Average Active Time
Machine001	5340.00	43	124.18
Machine002	5000.00	52	96.15
Machine003	4910.00	60	81.83
Machine004	3060.00	121	25.28
Machine005	5930.00	2	2965.00

Table 3-70 Simulation-based Method Summary

Resource	% Utilization	Utilization Factor	Bottleneck Rate
Machine001	89.00	0.08	0.10
Machine002	87.16	0.08	0.10
Machine003	85.33	0.08	0.10
Machine004	51.00	0.05	0.10
Machine005	98.83	0.03	0.03

Table 3-71 Inactive Duration Method Summary

Resource	Total Inactive Time	Number of Switches	Average Inactive Time
Machine001	660.00	43	15.34
Machine002	1000.00	52	19.23
Machine003	1090.00	60	18.16
Machine004	2940.00	121	24.29
Machine005	70.00	2	35.00

3.10 Evaluation of the Algorithms

Table 3-72 is a summary of the studies conducted in this chapter. Based on the result, most of the bottleneck detection methods are capable of detecting the true bottleneck in serial, concurrent, and feedback production lines. The challenge occurs when the model includes labor. Applying the percentage utilization method to labor in the model can identify the busiest labor resource in the system. Since “waiting on labor” is considered utilized time for machines, the utilization percentage for machines will be high. The question then becomes, “*Would the labor be considered the bottleneck, or would it be the machine*?” Active duration, inactive duration, and shifting bottleneck methods also define “waiting on labor” as active time. This raises the same concern of what should be considered first: the highest utilized labor, or the machine?

There is also the challenge of selecting the correct bottleneck when multiple resources have similar or close results that identify them as the bottleneck. Should only the resource with highest/lowest value be enhanced at a time, or should they all be? Which one should be considered first? The simulation-based method, for example, has three evaluation parameters: Which problem is the most critical? These interpretation will be difficult to automate. For that reason, simulation-based method will not be part of the prototype of the decision support system.

Shifting bottleneck and inactive duration methods are derivative of the active duration method and provide the same result. The only difference is that they are designed to detect bottleneck for specific period of time, momentum bottleneck. Since this study is analyzing the system for the full simulation run time and to minimize the computational process, these two methods will not be part of the decision support system.

Based on the results obtained from the previous sections with regards to turning point, an equal sign is recommended to the inequalities, Equation 2-9. It was concluded that the method failed to identify the correct bottleneck in the concurrent process model due to the fact that the method requires a manual intervention to substitute the machines in concurrent process with virtual machines. Moreover, the turning point method cannot work properly with a model that consists of resources other than machines. Therefore, it failed to identify the correct bottlenecks in models that required labor resource. Based on the performance measurements matrix, it was decided that the turning point method will be eliminated from the decision support system.

Again, the models used in this chapter are simple and straight forward, although the results could be biased. For that reason, models used in previous literature will be extracted and examined in chapter 5 to verify our findings, and we will implement the automated process.

Table 3-72 Performance Measurement Matrix

	Bottleneck Location in The Model (Machine)			**Non-Machine Bottleneck**	**Multiple Bottlenecks**				
Bottleneck Analysis Method	**Beginning**	**Middle**	**End**	**Labor/ AGV**	**Machines**	**Labors/ AGVs**	**Concurrent Process**	**Feedback Main Path**	**Feedback loop**
Percentage Utilization	Yes	Yes	Yes	Yes	Yes - Yes	Yes	Yes	Yes	Yes
Waiting Time Queue	Yes	Yes	Yes	No	Yes – Yes	No	Yes	Yes	Yes
Simulation–based Procedure	Yes	Yes	Yes	No	Yes – Yes	No	Yes	Yes	Yes
Maximum Average Active Duration	Yes	Yes	Yes	Yes	Yes – Yes	Yes	Yes	Yes	Yes
Turning Point	No	Yes	Yes	No	Yes – Yes	No	No	Yes	No
Shifting Bottleneck	Yes	Yes	Yes	Yes	Yes – Yes	Yes	Yes	Yes	Yes
Inactive Duration	Yes	Yes	Yes	Yes	Yes – Yes	Yes	Yes	Yes	Yes
Critical Indicator	Yes	Yes	Yes	No	Yes - Yes	No	Yes	Yes	Yes

- Yes - the method is able to detect the correct bottleneck in the case scenario.
- No - the method is not able to detect the correct bottleneck in the case scenario.

Chapter 4

Decision Support System

This chapter will be dedicated to explaining the process of the decision support system. Figure 4-1 shows the decision support system's flow chart. The process starts by loading the simulation model into the decision support system. The user should also specify the simulation run time. It should be noted that the user has to know about the warm-up time and add it to the time allocated to gather statistics, Figure 4-2. For example, assume that there is a need to fill the production line with parts for 30 days, and then gather the statistics for 60 days. The user should input 90 days in the run time field text. The base time unit in the developed DSS is expressed in minutes. The DSS assumes that there are 525,949 minutes in a year. The DSS will convert the time unit selected to minutes automatically. This will help to forecast the annual production quantity and to calculate the gross income, as explained further in this chapter.

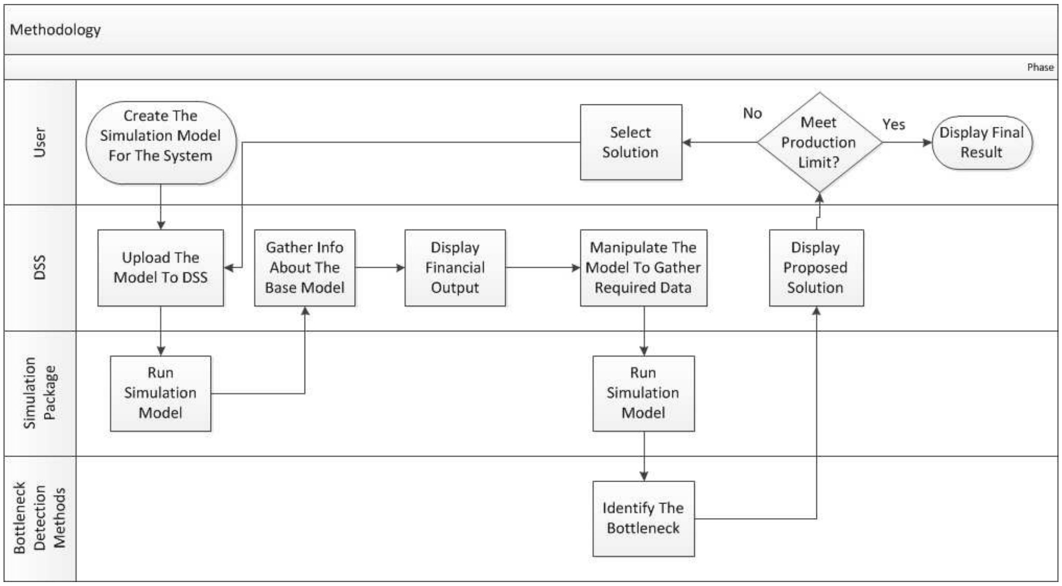
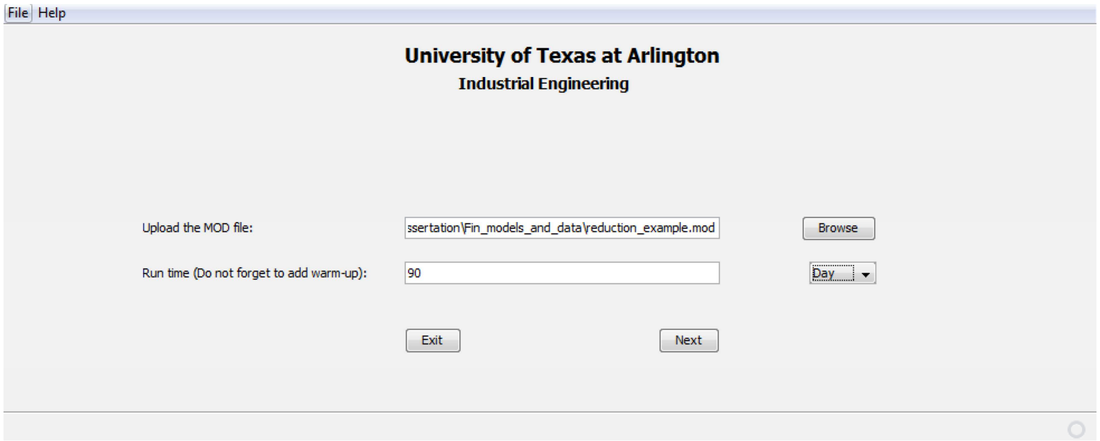


Figure 4-1 Methodology Flow Chart

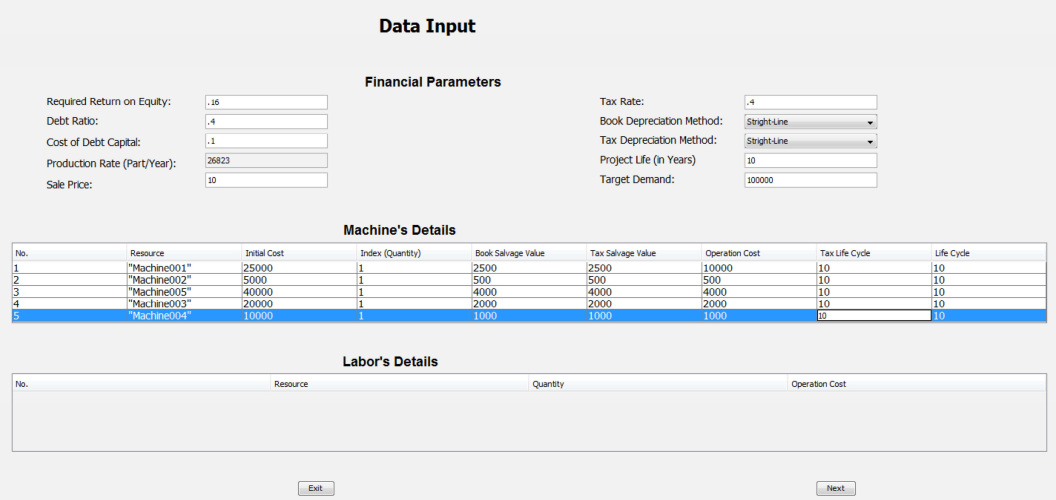


Upload the MOD file: ssertation\Fin_models_and_data\reduction_example.mod

Run time (Do not forget to add warm-up): 90 Day

Figure 4-2 Loading Simulation Model to DSS

The methodology in this study recommends to construct the simulation model with the minimum number of resources. the objective of the DSS is to guide the analyst with proper configuration of the model to reach the desired target. Once the simulation model is loaded, the DSS will pass the simulation model to the simulation package to run the model to gather relevant information and to construct the financial input form. The DSS will detect the model's machines and labor resources, as well as their quantities and will display them to the user. Furthermore, the DSS will forecast the annual production quantity based on the simulation run time entered. In addition, the selling price is also input, as it is required to calculate the gross income generated for each year, if the equity cash flow is needed. The initial cost of machines, book salvage value, tax salvage value, operating costs, tax life cycle, and book life cycle should also be entered to the system, as well as the financial parameters discussed in Chapter 2. The project life entry is used in case the life cycle of the machines are different. However, the project life cycle should be equal to the longest life cycle resource in the model. As for labor resources, the operating costs must be entered. Figure 4-3 contains an example of the financial data entry to DSS.



Data Input

Financial Parameters

Required Return on Equity: .16
Debt Ratio: .4
Cost of Debt Capital: .1
Production Rate (Part/Year): 26823
Sale Price: 10
Tax Rate: .4
Book Depreciation Method: Stright-Line
Tax Depreciation Method: Stright-Line
Project Life (in Years) 10
Target Demand: 100000

Machine's Details

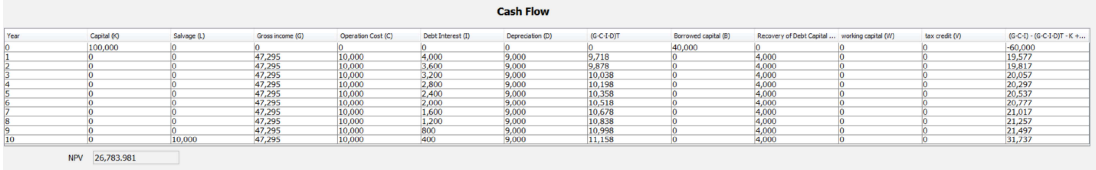
No.	Resource	Initial Cost	Index (Quantity)	Book Salvage Value	Tax Salvage Value	Operation Cost	Tax Life Cycle	Life Cycle
1	"Machine001"	25000	1	2500	2500	10000	10	10
2	"Machine002"	5000	1	500	500	500	10	10
3	"Machine005"	40000	1	4000	4000	4000	10	10
4	"Machine003"	20000	1	2000	2000	2000	10	10
5	"Machine004"	10000	1	1000	1000	1000	10	10

Labor's Details

No.	Resource	Quantity	Operation Cost

Figure 4-3 Financial Data Input Screen

Once all required fields are entered, the cash flow and/or the minimum annual revenue requirements will be calculated and displayed as requested. The DSS will also display the net present values for the cash flow, Figure 4-4, and the minimum annual revenue requirement, Figure 4-5. If those values are positive, the project will be deemed attractive. If not, the project will be considered less desirable. Based on the minimum annual revenue requirements, a new selling price will be suggested as shown in Figure 4-5. The analyst will decide whether to use the suggested new price, or to continue using the original price.

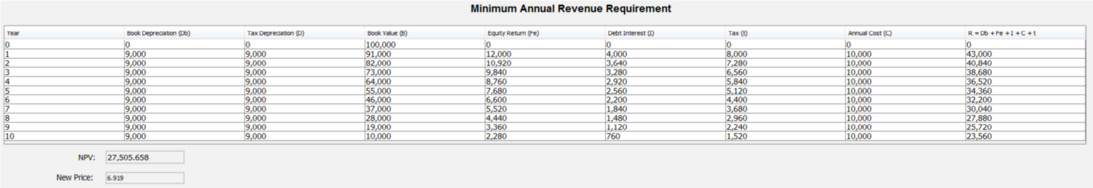


Cash Flow

Year	Capital (K)	Salvage (L)	Gross income (G)	Operation Cost (C)	Debt Interest (I)	Depreciation (D)	(G-C-I-D)T	Borrowed capital (B)	Recovery of Debt Capital ...	working capital (W)	tax credit (V)	(G-C-I) - (G-C-I-D)T - K +...
0	100,000	0	0	0	0	0	0	40,000	0	0	0	-60,000
1	0	0	47,295	10,000	4,000	9,000	9,718	0	4,000	0	0	19,577
2	0	0	47,295	10,000	3,600	9,000	9,878	0	4,000	0	0	19,817
3	0	0	47,295	10,000	3,200	9,000	10,038	0	4,000	0	0	20,057
4	0	0	47,295	10,000	2,800	9,000	10,198	0	4,000	0	0	20,297
5	0	0	47,295	10,000	2,400	9,000	10,358	0	4,000	0	0	20,537
6	0	0	47,295	10,000	2,000	9,000	10,518	0	4,000	0	0	20,777
7	0	0	47,295	10,000	1,600	9,000	10,678	0	4,000	0	0	21,017
8	0	0	47,295	10,000	1,200	9,000	10,838	0	4,000	0	0	21,257
9	0	0	47,295	10,000	800	9,000	10,998	0	4,000	0	0	21,497
10	0	10,000	47,295	10,000	400	9,000	11,158	0	4,000	0	0	31,737

NPV 26,783.981

Figure 4-4 Equity Cash Flow



Minimum Annual Revenue Requirement

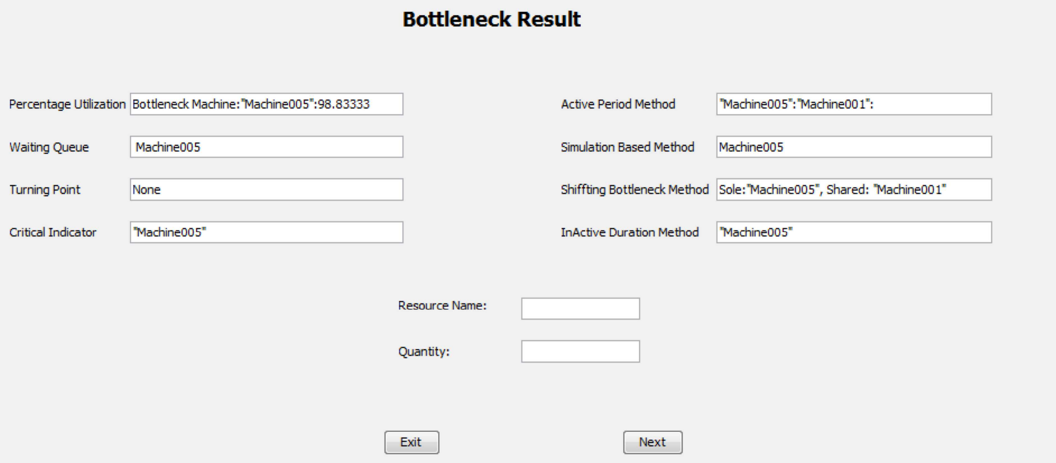
Year	Book Depreciation (Db)	Tax Depreciation (D)	Book Value (B)	Equity Return (Fe)	Debt Interest (I)	Tax (t)	Annual Cost (C)	R = Db + Fe + I + C + t
0	0	0	100,000	0	0	0	0	0
1	9,000	9,000	91,000	12,000	4,000	8,000	10,000	43,000
2	9,000	9,000	82,000	10,920	3,640	7,280	10,000	40,840
3	9,000	9,000	73,000	9,840	3,280	6,560	10,000	38,680
4	9,000	9,000	64,000	8,760	2,920	5,840	10,000	36,520
5	9,000	9,000	55,000	7,680	2,560	5,120	10,000	34,360
6	9,000	9,000	46,000	6,600	2,200	4,400	10,000	32,200
7	9,000	9,000	37,000	5,520	1,840	3,680	10,000	30,040
8	9,000	9,000	28,000	4,440	1,480	2,960	10,000	27,880
9	9,000	9,000	19,000	3,360	1,120	2,240	10,000	25,720
10	9,000	9,000	10,000	2,280	760	1,520	10,000	23,560

NPV: 27,505.658

New Price: 6.919

Figure 4-5 Minimum Annual Revenue Requirement

Finally, the DSS will modify the simulation model to produce the output needed by each method. Once the required data is generated, the bottleneck detection algorithms will be applied. The next screen will show the bottleneck suggested by each method, as shown in Figure 4-6.



Bottleneck Result

Percentage Utilization: Bottleneck Machine:"Machine005":98.83333

Waiting Queue: Machine005

Turning Point: None

Critical Indicator: "Machine005"

Active Period Method: "Machine005":"Machine001":

Simulation Based Method: Machine005

Shiffting Bottleneck Method: Sole:"Machine005", Shared: "Machine001"

InActive Duration Method: "Machine005"

Resource Name:

Quantity:

Figure 4-6 Bottleneck Detection Methods Result

Chapter 5

Bottleneck Detection Methods Evaluation

Information from several previous case studies was extracted in order to investigate and evaluate the eight bottleneck detection methods that were selected in this dissertation. However, our research has found that there were three problems with the extracted models. Firstly, most case studies are simple and do not represent real-world scenarios. Secondly, there was not enough information provided by the authors to replicate those models and produce the same results. Lastly, most literature had applied its own method against the case study model presented in the literature. In our opinion, this leads to biased result. On the contrary, this study aims at applying all the bottleneck detection methods mentioned earlier against case study models extracted from previous literature. To achieve this goal, this paper identifies eight scenarios that will help to truly evaluate the bottleneck detection methods. The case studies scenarios are:

1) Serial Production Line With Bottleneck Located at the Beginning.
2) Serial Production Line With Bottleneck Located at the End.
3) Serial Production Line With Bottleneck Located in the Middle.
4) Serial Production Line With Single AGV or Labor Bottleneck.
5) Serial Production Line With Multiple Bottlenecks.
6) Serial Production Line With Multiple AGV’s/Labor Bottlenecks.
7) Production Line With Concurrent Process.
8) Production Line With Feedback.

To ensure the accuracy of the bottleneck detection algorithms, a manual process will be conducted to measure throughput. In other words, the quantities for each resource in the model will be increased and the throughput will be measured. The resource(s) with the highest throughput will be deemed to be the true bottleneck. This approach will also validate the bottleneck identified in the paper from which the models were extracted.

At the end of this chapter, a summary will be provided to highlight the strengths and weaknesses of each technique. The summary will be used to determine which algorithm can be automated and used in the decision support system tool provided by this study.

5.1 Serial Production Line With Bottleneck Located at the Beginning

The model was extracted from Kasemset & Kachitvichyanukul (2007), and illustrates the concept of the bottleneck being located at the beginning of the model. As the model was intended for simple clarification, no concise specification was provided. Analysis in this study was conducted based on the specification and assumption shown in Table 5-1. The model has three single machines connected in serial fashion. The inter-arrival rate for the input part is 20 parts per every one minute. The cycle time for each machine is 1/10, 1/2, and 1/15 minutes respectively, and the model does not include labor. Moreover, no setup, downtime, or warm-up time was specified. The simulation run time was assumed to be 100 hours. To obtain useful result, three buffers of size 1 were added to the model.

Table 5-1 Specs for Model Where Bottleneck is Located at the Beginning

Element	**Description**
Raw Material Arrival Rate	20 parts per minute
Machine	Three single machines, Cycle times: 1/10, 1/12, and 1/15 minutes, respectively.
Buffer	Three buffers of size 1
Setup	N/A
Breakdown	N/A
Labor	N/A
Simulation Run Time	100 hours
Warm-up Time	N/A

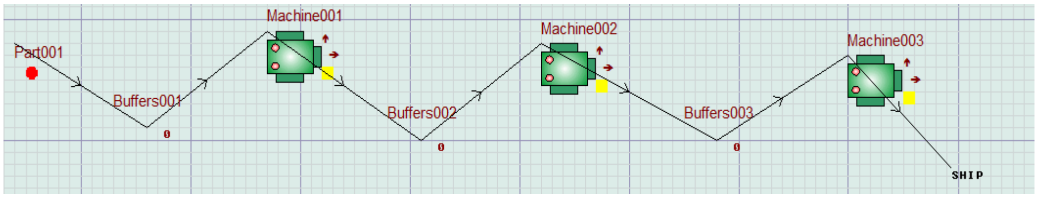


Figure 5-1 Model Where Bottleneck is Located at the Beginning

Table 5-2 Production Quantities

Machine Added To The Model	**Production Quantity**
Base Model	59,998
Machine001	71,998
Machine002	59,998
Machine003	59,998

The simulation run conducted in this research revealed that 59,998 parts were produced. The resources' utilization was: 100%, 83.33%, and 66.66% respectively. Adding Machine001 as an additional machine to the model began the manual verification process. This alteration allowed the production line to increase productivity to 71,998 parts. When Machine002 was added, the production line reverted back to producing 59,998 parts. The same result was observed when Machine003 was introduced. This proves that Machine001 is the bottleneck for this model, Table 5-2.

All methods successfully identified the true bottleneck except the turning point method because that method failed to satisfy the condition $(TB_1\text{-}TS_1) > 0$. The actual values for TB_1 and TS_1 are 0. However, there is no equal sign in the inequality provided by the original author, and thus the condition was not satisfied, nor did it return a result.

It should be noted that even though the buffer size is pre-defined, the waiting queue method was able to detect the bottleneck. The average waiting time in the first buffer was .09 and the average size was .86. Those values are the highest among all the buffers.

5.2 Serial Production Line With Bottleneck Located in the Middle

The model was extracted from Roser, Nakano and Tanaka (2002) and consists of four single serial production machines. The input rate for the raw material is one part per every 1.25 seconds. The cycle time for each machine is 1, 1, 1.1, and 1 second respectively. No labor, setup, or breakdown time was specified in the model. The simulation run time is 120,000 seconds, which includes 20,000 seconds of warm-up time. As the reader might anticipate, the bottleneck is clearly the third machine. It should be noted that the model presented in this paper was extracted from Lawrence and Buss (1994), applying a slight modification. Because neither of these two papers mentioned buffer specifications, this study has added four buffers of size 1000 in front of every machine.

Table 5-3 Specs for Model Where Bottleneck is Located in the Middle

Element	**Description**
Raw Material Arrival Rate	Exponential - 1 part every 1.25 seconds
Machine	Four single machines, Exponential cycle times: 1, 1, 1.1, and 1 second respectively
Buffer	Four buffers of size 1000
Setup	N/A
Breakdown	N/A
Labor	N/A
Simulation Run Time	120,000 seconds
Warm-up Time	20,000 seconds

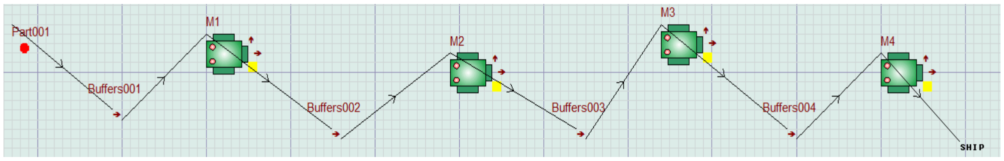


Figure 5-2 Model Where Bottleneck is Located in the Middle

Table 5-4 Production Quantities

Machine Added To The Model	Production Quantity
Base Model	112,539
M1	112,539
M2	112,543
M3	112,545
M4	112,539

The simulation trial conducted in this research found that the resources' utilization for the base model is 80.25, 80.69, 88.24, and 80.46 respectively. This is quite similar to the results presented in the paper, which were: 80.1%, 80.2%, 88.0%, and 80.0% respectively. It should be noted that the difference is due to the exponential distribution used in the model, as well as the fact that different simulation software was used in the two experiments. The base model produced 112,539 parts. The same result was obtained in the experiments when M1 and M4 were added, separately. When M2 was added, the production line produced 112,543 parts. Finally, when M3 was added, the production line produced 112,545 parts, Table 5-4. The paper indicated that the bottleneck in the model is M3.

All methods successfully identified the bottleneck. Turning point was the only exception, as it was unable to identify the true bottleneck due to condition $(TB_2 – TS_2) > 0$, which failed to be satisfied. The actual values for TB_2 and TS_2 were 0 and 19.81 respectively.

It is important to note that even though the buffer size was pre-defined, the waiting queue method was able to detect the bottleneck. The average waiting time and average size for the third buffer were .14 and 6.89 respectively.

5.3 Serial Production Line With Bottleneck Located at the End

The model was extracted from Sengupta, Das, Vantil (2008). The model includes four single machines with cycle times of 0.80, 0.83, 0.87 and 0.90 minutes respectively, with an unlimited rate of raw material flow. The model has three buffers of size 8 units. No setup time or labor had been specified in the model. The time to repair (TTR) and time between failure (TBF) are shown in Table 5-5. The simulation run time is 600 minutes with no warm-up time. The author has identified the bottleneck resource as the fourth machine.

Table 5-5 Specs for Model Where Bottleneck is Located at the End

<table>
<tr><th>Element</th><th colspan="3">Description</th></tr>
<tr><td>Raw Material Arrival Rate</td><td colspan="3">Unlimited</td></tr>
<tr><td>Machine</td><td colspan="3">Four single machines,
Exponential cycle times: 0.80, 0.83, 0.87 and 0.90 minutes respectively</td></tr>
<tr><td>Buffer</td><td colspan="3">Three buffers of size 8</td></tr>
<tr><td>Setup</td><td colspan="3">N/A</td></tr>
<tr><td rowspan="5">Breakdown</td><td>Machine</td><td>TTR (minutes)</td><td>TBF (minutes)</td></tr>
<tr><td>M1</td><td>EXPO(3)</td><td>EXPO(120)</td></tr>
<tr><td>M2</td><td>EXPO(3)</td><td>EXPO(90)</td></tr>
<tr><td>M3</td><td>EXPO(3)</td><td>EXPO(60)</td></tr>
<tr><td>M4</td><td>EXPO(3)</td><td>EXPO(25)</td></tr>
<tr><td>Labor</td><td colspan="3">N/A</td></tr>
<tr><td>Simulation Run Time</td><td colspan="3">600 minutes</td></tr>
<tr><td>Warm-up Time</td><td colspan="3">N/A</td></tr>
</table>

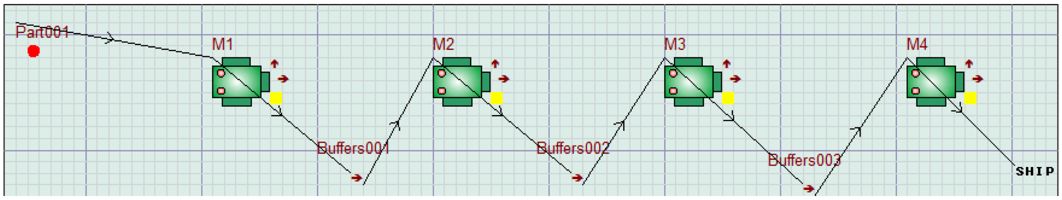


Figure 5-3 Model Where Bottleneck is Located at the End

Table 5-6 Production Quantities

Machine Added to the Model	Production Quantity
Base Model	585
M1	585
M2	585
M3	585
M4	623

The base model produced 585 parts. The result obtained in the experiments when M1, M2, and M3 were added to the system is shown in Table 5-6. On the other hand, when Workstation4 was added to the system, production increased to 623 parts. This is another indication that M4 is the bottleneck.

Table 5-7 Resources Statistics (Sengupta et al., 2008)

Machine	Blocked-up + Blocked-down (Minute)	Cycle + Failure (Percentage)
M1	155.2	72.57%
M2	150.2	79.34%
M3	151.5	75.72%
M4	102.2	96.1%

Table 5-8 Resources Statistics Obtained

Machine	%Busy	%Idle	%Blocked	%Broken
M1	81.61	0.00	17.42	0.98
M2	83.72	0.13	8.73	7.42
M3	86.14	1.84	6.88	5.14
M4	87.76	2.63	0.00	9.61

The output obtained from the article is shown in Table 5-7. The sum of the busy and the failure percentages for M2 is 79.34%, whereas, the sum of the blocked-up and blocked-down times is 150.2 minutes. The runtime was 600 minutes, which imples that M2 spent 25% of its time in blocked-up and blocked-down states. Adding the two percentages exceeds 100%. This conclusion contradicts the first rule to identify the data element errors, which was proposed by the authors. Nevertheless, M4 is indeed the bottleneck, as pointed out by the majority of the bottleneck detection methods.

The waiting queue method identified the bottleneck as M2, and was the only method that was unable to identify the correct bottleneck. The average waiting time in Buffers002, which feeds M3, was 7.61 minutes. The average waiting time in Buffers003, which feeds M4, was 6.76 minutes. Moreover, the average number of parts waiting in Buffers002 is 7.76 parts, compared to 6.70 in Buffers003. These variances are due to the fact that the size of the buffers were finite. Therefore, waiting queue was not able to pinpoint the true bottleneck.

5.4 Serial Production Line With Single AGV or Labor Bottleneck

None of the previous papers have considered a model with a single labor bottleneck. Therefore, a model was constructed with the following specifications shown in Table 5-9. The model consists of four single machines connected in a series, with an unlimited supply of raw material. The cycle times for the machines are: 5, 6, 7, and 8 minutes respectively. There are three buffers of size 10 units as shown in Figure 5-4. Neither setup nor breakdown is specified in the model. The laborer must be positioned beside the machine in order to process a part. The simulation run time is 100 hours, with no warm-up time. Since the labor-shared resource works on every machine to process the part, labor is considered to be the bottleneck in the model.

Table 5-9 Specs for Model Where Bottleneck is Labor

Element	Description
Raw Material Arrival Rate	Unlimited supply
Machine	Four single machines, Cycle times: 5, 6, 7, and 8 minutes respectively
Buffer	Three buffers of size 10
Setup	N/A
Breakdown	N/A
Labor	One Shared Laborer
Simulation Run Time	100 hours
Warm-up Time	N/A

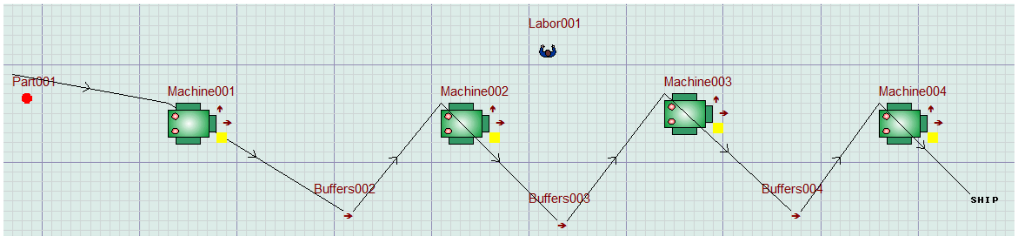


Figure 5-4 Model Where Bottleneck Is One Single Laborer

Table 5-10 Production Quantities

Machine Added to the Model	Production Quantity
Base Model	230
Machine001	228
Machine002	230
Machine003	230
Machine004	230
Labor	460

The base model produced 230 parts. The same result was obtained when Machine002, Machine003, and Machine004 were added to the experiment. However, when Machine001 was added, the production decreased to 228 parts. Finally, when Labor001 was added, the production line produced 460 parts, Table 5-10. This is an indication that the true bottleneck is Labor001.

The utilization percentage for labor, average active duration, shifting bottleneck, and inactive duration are the methods that were able to identify the true bottleneck. The waiting queue method was not able to identify any bottleneck in the system, because during the simulation run, none of the parts waited in the buffers. The remaining methods identified Machine001 as the bottleneck. We believe that the reason those bottleneck methods failed to identify the true bottleneck is due to how the bottleneck resource was defined. Most of the bottleneck detection methods identified “waiting on labor” as busy and utilized time. According to the machine statistic report generated by the simulation package, Machine001 is 100% utilized. This is shown in Figure 5-5, as “waiting on labor” is considered busy time.

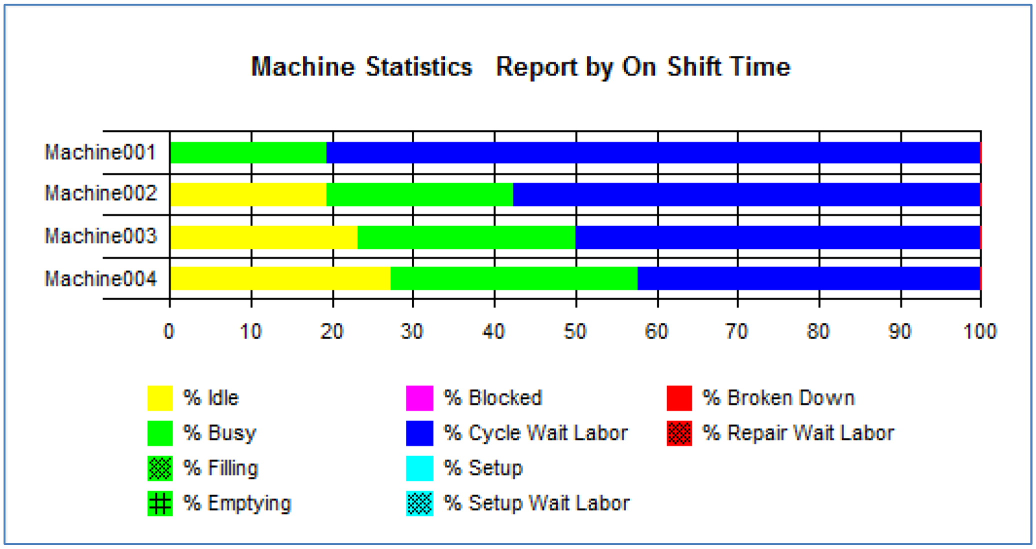


Figure 5-5 Machine Statistics for Model with Single Labor Bottleneck

Adding Machine001 to the system decreased productivity due to a blockage that was produced in the system, as shown in Figure 5-6. On the other hand, when the correct bottleneck was added (Labor001), production doubled and the flow was smoother.

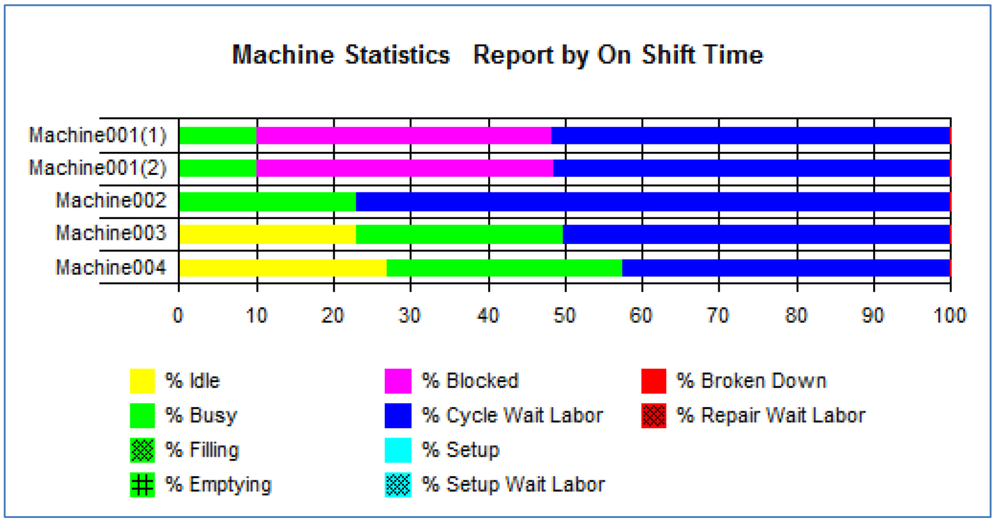


Figure 5-6 Machine Statistics With Wrong Bottleneck Increased

5.5 Serial Production Line With Multiple Bottlenecks

The model was extracted from Li, Chang, and Ni (2009). The model consisted of five single machines, with each machine having a cycle time of 10 minutes. Also, there were four buffers of size 50, with no setup time or breakdown, as shown in Table 5-11. The authors indicated that the bottlenecks in the model were the second and fourth machines.

Table 5-11 Specs for Model With Multiple Bottlenecks

<table>
<tr><th>Element</th><th colspan="3">Description</th></tr>
<tr><td>Raw Material Arrival Rate</td><td colspan="3">Unlimited Supply</td></tr>
<tr><td>Machine</td><td colspan="3">Five single machines, Exponential cycle times: 10, 10, 10, 10, and 10 seconds respectively</td></tr>
<tr><td>Buffer</td><td colspan="3">Four buffers of size 50</td></tr>
<tr><td>Setup</td><td colspan="3">No</td></tr>
<tr><td rowspan="6">Breakdown</td><td>Machine</td><td>MTTR(min)</td><td>MTBF(min)</td></tr>
<tr><td>1</td><td>2</td><td>31</td></tr>
<tr><td>2</td><td>2</td><td>23</td></tr>
<tr><td>3</td><td>0</td><td>4800</td></tr>
<tr><td>4</td><td>2</td><td>27</td></tr>
<tr><td>5</td><td>2</td><td>30</td></tr>
<tr><td>Labor</td><td colspan="3">No</td></tr>
<tr><td>Simulation Run Time</td><td colspan="3">1 week</td></tr>
<tr><td>Warm-up Time</td><td colspan="3">None</td></tr>
</table>

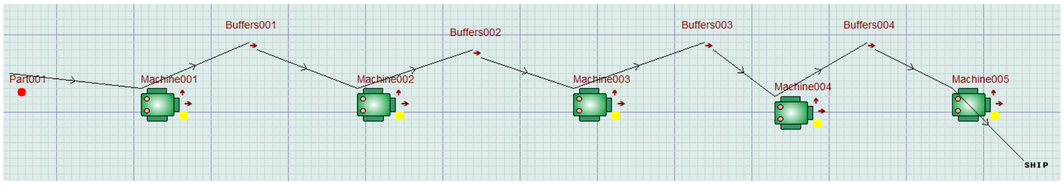


Figure 5-7 Model With Multiple Bottlenecks

Table 5-12 Production Quantities

Machine Added To The Model	Production Quantity
Base Model	55,208
Machine001	55,208
Machine002	55,976
Machine003	55,208
Machine004	55,208
Machine005	55,208

The model ran for one week to conduct the initial assessment of this study. The production line produced 55,208 parts for the base model. The same result was obtained when Machine001, Machine003, Machine004, and Machine005 were added to the system. When Machine002 was added, the system produced 55,976 parts. The production line produced 56,419 parts when Machine002 and Machine004 were added to the model. As a sanity check, Machine001 and Machine003 were added to the system, which then produced only 55,208 parts. This verifies that the co-bottlenecks are Machine002 and Machine004. The buffer statistics and utilizations presented in the paper are shown in Table 5-13 and Figure 5-8. The outputs obtained from this study are shown in Table 5-14 and Table 5-15.

Table 5-13 Buffer Statistics (Li et al., 2009)

Buffer	Capacity	Average contents	Maximum contents	Minimum contents
B1	50	45.33	50	38
B2	50	6.09	50	0
B3	50	24.55	49	2
B4	50	5.18	12	0

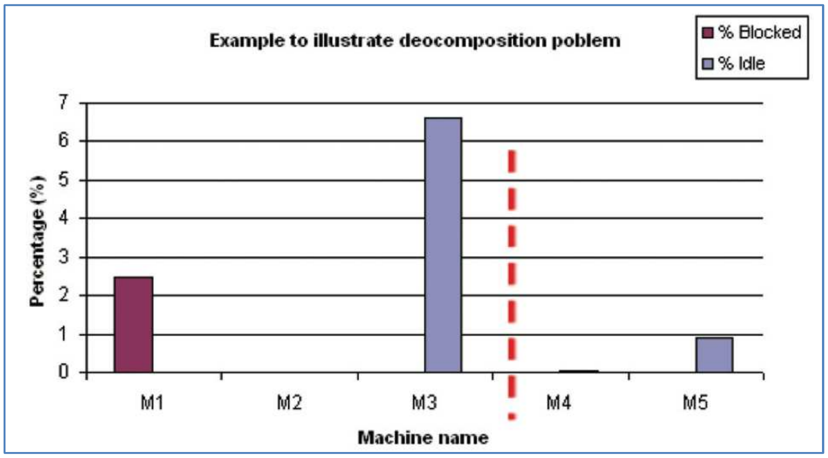


Figure 5-8 Resources Utilization (Li et al., 2009)

Table 5-14 Buffers Statistics

Buffer	Capacity	Average contents	Maximum contents	Minimum contents
B1	50	29.45	50	0
B2	50	0.00	1	0
B3	50	4.71	13	0
B4	50	5.70	13	0

Table 5-15 Resources Statistics

Resource	%Idle	%Busy	%Blocked	%Broken
M1	0.00	92.99	0.76	6.25
M2	0.03	91.22	0.00	8.75
M3	8.82	91.18	0.00	0.00
M4	1.77	90.73	0.00	7.50
M5	2.64	90.69	0.00	6.67

The true bottlenecks are identified as being Machine002 and Machine004. In the initial run, all methods identified Machine002 as the bottleneck, but we believe that occurred because it is the first bottleneck resource in the model. However, increasing Machine002 did not improve productivity. After increasing Machine002, all the methods successfully identified the second co-bottleneck resource as Machine004. Increasing both resources improved productivity.

5.6 Serial Production Line With Multiple AGV's/Labor Bottlenecks

The model presented by D'Souza (2004) consists of four machines. There are also four laborers who transfer the parts between three machines in a circular fashion, as shown in Figure 5-9. In other words, Workstation1 starts the process and pushes the part to Buffer1. A laborer picks up the part from Buffer1 and moves it to Workstation2. Once the part is processed at Workstation2, the same laborer will transfer it to Workstation3, if it is available. If not, the part will be placed in Buffer2. Once the part is processed at Workstation3, the laborer will transfer it to Workstation4. Again, if Workstation4 is unavailable, the part will be placed in Buffer3. Finally, when the part is processed at Workstation4, it leaves the system and the laborer returns to Buffer1 to pick up a new part and begins a new cycle. The author has identified the laborer as the bottleneck in the model.

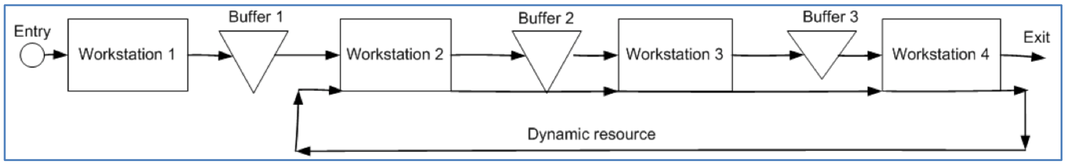


Figure 5-9 Multiple Labor Bottleneck Model (D'Souza, 2004)

Table 5-16 Specs for Model With Multiple Laborers As Bottleneck

Element	**Description**
Raw Material Arrival Rate	16 jobs per hour
Machine	Four machines, Exponential cycle times: 3, 3.5, 2, and 2 minutes respectively
Buffer	Three buffers of size 15
Breakdown	N/A
Setup	N/A
Labor	Four labor resources
Simulation Run Time	700 hours
Warm-up Time	120 hours

Several solution scenarios were conducted to manually identify the bottleneck. The first scenario involved adding Workstation1 to the base model. The model then ran for 820 hours with the production quantity noted, and so forth. Surprisingly, the true bottleneck turned out to be Workstation2, Table 5-17. As the quantity of Workstation2 was increased to two, productivity jumped to 13,466 units. Whereas, when labor was increased from 4 to 10, the system produced only 11,852 parts. In our opinion, the true bottleneck is Workstation2, as opposed to the labor resource.

Table 5-17 Production Quantities

Machine Added To The Model	Production Quantity
Base Model	10,550
Workstation1	10,550
Workstation2	13,466
Workstation3	11,033
Workstation4	11,002
One Labor	5,575
Ten Labors	11,852

The statistical outputs presented by D'Souza (2004) are shown in Table 5-18 and Table 5-19. The statistical outputs obtained in this study are shown in Table 5-20 and Table 5-21. As the results show, the two experiments are almost identical.

Table 5-18 Workstation Statistics for Multi-Bottleneck Labor (D'Souza, 2004)

Workstation	%Busy	%Setup	%Idle	%Waiting	%Blocked	%Down
1	74.20	0.00	0.00	0.00	25.80	0.00
2	86.95	0.00	0.01	13.03	0.00	0.00
3	50.08	0.00	49.92	0.00	0.00	0.00
4	49.71	0.00	50.29	0.00	0.00	0.00

Table 5-19 Labor Statistics for Multi-Bottleneck Labor (D'Souza, 2004)

Resource	%In Use	%Travel to Use	%Travel to Park	%Idle	%Down
Labor001	65.56	0.42	1.66	32.37	0.00

Table 5-20 Workstation Statistics for Multi-Bottleneck Labor

Workstation	%Busy	%Setup	%Idle	%Waiting	%Blocked	%Down
1	75.52	0.00	0.00	0.00	24.48	0.00
2	87.12	0.00	12.88	0.00	0.00	0.00
3	50.38	0.00	49.62	0.00	0.00	0.00
4	49.82	0.00	50.18	0.00	0.00	0.00

Table 5-21 Labor Statistics for Multi-Bottleneck Labor

Resource	%In Use	%Travel to Use	%Travel to Park	%Idle	%Down
Labor001	64.57	N/A	N/A	35.43	N/A

Because the true bottleneck in the model turned out to be a machine, the methods were not truly evaluated to detect a model with multiple laborers who were determined to be the bottlenecks. Therefore, the model was adjusted to have only minimum resources. In other words, the new base model will have only a quantity of one of each resource that existed in the original model extracted from D'Souza (2004). Table 5-22 shows the model configuration for the new base model. The base model produced 5,575 parts. Table 5-23 and Table 5-24 show the workstations and labor statistics. Based on these results, it was concluded that labor is the bottleneck for the new base model.

Table 5-22 New Base Model

Element	Description
Raw Material Arrival Rate	16 jobs per hour
Machine	Four machines, Exponential cycle times: 3, 3.5, 2, and 2 minutes respectively
Buffer	Three buffers of size 15
Breakdown	N/A
Setup	N/A
Labor	one labor resource
Simulation Run Time	700 hours
Warm-up Time	120 hours

Table 5-23 Workstations Statistics for New Base Model

Workstation	%Busy	%Setup	%Idle	%Waiting	%Blocked	%Down
1	39.30	0.00	0.00	0.00	60.70	0.00
2	47.01	0.00	52.99	0.00	0.00	0.00
3	26.89	0.00	73.11	0.00	0.00	0.00
4	26.10	0.00	73.90	0.00	0.00	0.00

Table 5-24 Labor Statistics for New Base Model

Resource	%In Use	%Travel to Use	%Travel to Park	%Idle	%Down
Labor001	100.00	N/A	N/A	0.00	N/A

The result of the new configuration revealed that the bottleneck is the single labor resource. Once another laborer was added, production jumped to 8,224 parts. The new bottleneck is Workstation2. The result of the experiment is shown in Table 5-25 and Table 5-27. It was concluded that the new bottleneck is Workstation2, and therefore another Workstation2 was added to the system, resulting in production increasing to 9,812 parts. Since the labor resources are utilized 100% and the blockage is on Workstation1, it is evident that the true bottleneck is the labor resource. Adding a third laborer will definitely increase productivity. Therefore, the methods in question can be evaluated with configuration presented in Table 5-28.

Table 5-25 Model Configuration When Labor is Added to the New Based Model

Element	Description
Raw Material Arrival Rate	16 jobs per hour
Machine	Four machines, Exponential cycle times: 3, 3.5, 2, and 2 minutes respectively
Buffer	Three buffers of size 15
Breakdown	N/A
Setup	N/A
Labor	Two labor resources
Simulation Run Time	700 hours
Warm-up Time	120 hours

Table 5-26 Resource Statistics When Labor is Added to the New Based Model

Workstation	%Busy	%Setup	%Idle	%Waiting	%Blocked	%Down
1	59.11	0.00	0.00	0.00	40.89	0.00
2	68.30	0.00	31.70	0.00	0.00	0.00
3	39.06	0.00	60.94	0.00	0.00	0.00
4	39.06	0.00	60.94	0.00	0.00	0.00

Table 5-27 Labor Statistics When Labor is Added to the New Based Model

Resource	%In Use	%Travel to Use	%Travel to Park	%Idle	%Down
Labor001	83.93	N/A	N/A	16.07	N/A

Table 5-28 Model Configuration When Workstation2 is Added to the Model

Element	Description
Raw Material Arrival Rate	16 jobs per hour
Machine	Five machines Exponential cycle times: 3, 3.5, 2, and 2 minutes respectively
Buffer	Three buffers of size 15
Breakdown	N/A
Setup	N/A
Labor	Two labor resources
Simulation Run Time	700 hours
Warm-up Time	120 hours

Table 5-29 Production Quantities

Machine Added To The Model	Production Quantity
Base Model	9,812
Workstation1	9,811
Workstation2	9,812
Workstation3	10,499
Workstation4	10,482
Labor	12,270

Table 5-30 Resource Statistics When Workstation2 is Added to the Model

Workstation	%Busy	%Setup	%Idle	%Waiting	%Blocked	%Down
1	70.24	0.00	0.00	0.00	29.76	0.00
2(1)	40.67	0.00	59.33	0.00	0.00	0.00
2(2)	40.44	0.00	59.56	0.00	0.00	0.00
3	46.91	0.00	53.09	0.00	0.00	0.00
4	46.39	0.00	53.61	0.00	0.00	0.00

Table 5-31 Labor Statistics When Workstation2 is Added to the Model

Resource	%In Use	%Travel to Use	%Travel to Park	%Idle	%Down
Labor001	100.00	N/A	N/A	0.00	N/A

Table 5-29 and Table 5-31 show that the true bottleneck is labor. Only the active duration, percentage utilization, inactive duration, and shifting bottleneck methods were able to identify Labor001 as the bottleneck.

5.7 Production Line With Concurrent Process

This model was created by D’Souza (2004), consisting of seven machines. The raw material feeds directly to the first machine, which splits the raw material into two parts. Each part will be directed to a different buffer, which is the source for two separate parallel processes. Each process will have two machines separated by a buffer. At the end of the parallel process is a buffer that collects the processed parts from the two lines. The sixth machine will join the two parts coming from the two buffers and feed them to the seventh machine. Figure 5-10 shows the diagram of the model. It should be noted that the size of the buffers is 15 units, and the cycle time is exponentially distributed as 3, 2, 5, 2, 2, 2, and 2 minutes respectively. The arrival rate for the raw material is 20 parts per every hour, and the simulation run time is 700 hours in addition to 120 hours of warm-up time. Finally the author indicated that the bottleneck is the third machine (Machine003). The model specifications are shown in Table 5-32.

Table 5-32 Specs for Model With Concurrent Process

Element	**Description**
Raw Material Arrival Rate	20 parts every hour
Machine	Seven machines, exponentially distributed as 3, 2, 5, 2, 2, 2, and 2 minutes respectively
Buffer	Six buffers of size 15
Setup	N/A
Breakdown	N/A
Labor	N/A
Simulation Run Time	700 hours
Warm-up Time	120 hours

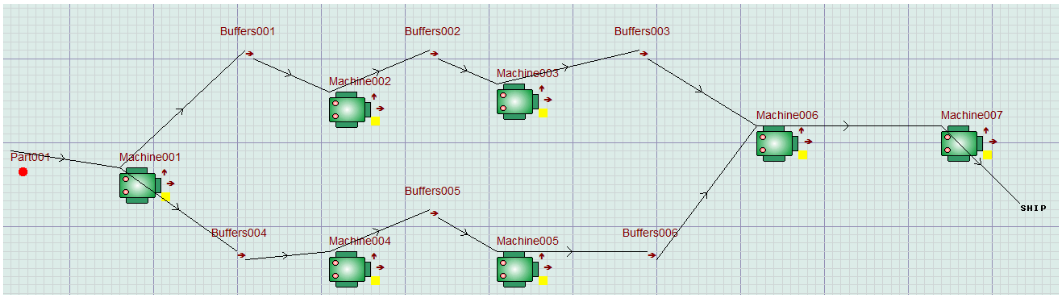


Figure 5-10 Model With Concurrent Process

Table 5-33 Production Quantities

Machine Added To The Model	Production Quantity
Base Model	8405
Machine001	8405
Machine002	8405
Machine003	13732
Machine004	8405
Machine005	8405
Machine006	8405
Machine007	8405

The base model that was created for this study produced 8,405 parts. The same number of parts was obtained when Machine001, Machine002, Machine004, Machine005, Machine006, and Machine007 were added to the model. On the other hand, when Machine003 was added, the production line produced 13,732 parts. This indicates that Machine003 is the bottleneck.

The resources' utilization of the resources extracted from the paper is shown in Table 5-34. The output obtained through this study was close, as shown in Table 5-35. It was concluded that the slight difference is due to the exponential distribution, as well as the fact that different simulation packages were utilized in the two experiments.

The statistical output for the buffers is shown in

Table 5-36. According to the waiting queue method, Machine006 is the bottleneck, which is corroborated. It was concluded that the method failed to identify the correct bottleneck because the buffers had a pre-defined size.

Turning Point was also unable to identify the bottleneck because it failed to satisfy the condition (TB1-TS1) > 0. The actual values for TB1 and TS1 are 0, however there is no equal sign in the inequality provided by the original author. Therefore, the condition was not satisfied. Turning point is also not designed to detects bottleneck with system that has parallel processes. For those reasons, the method did not return a result which proves that turning point requires manual intervention to create virtual machines for the parallel process.

Table 5-34 Resources Statistics (D'Souza, 2004)

Workstation	%Busy	%Setup	% Ideal	%Waiting	%Blocked	%Down
1	59.73	0.00	0.00	0.00	40.27	0.00
2	40.13	0.00	27.95	0.00	31.92	0.00
3	99.96	0.00	0.04	0.00	0.00	0.00
4	39.70	0.00	1.52	0.00	58.79	0.00
5	40.20	0.00	0.24	0.00	59.56	0.00
6	40.16	0.00	59.68	0.14	0.02	0.00
7	40.17	0.00	59.83	0.00	0.00	0.00

Table 5-35 Resources Statistics

Workstation	%Busy	%Setup	% Ideal	%Waiting	%Blocked	%Down
1	60.29	0.00	0.00	0.00	39.71	0.00
2	39.92	0.00	0.00	0.00	60.08	0.00
3	100.00	0.00	0.00	0.00	0.00	0.00
4	39.89	0.00	17.76	0.00	42.35	0.00
5	40.23	0.00	0.00	0.00	59.77	0.00
6	39.59	0.00	46.70	0.14	13.71	0.00
7	40.45	0.00	59.55	0.00	0.00	0.00

Table 5-36 Buffers Statistics

Buffer	**Average Part in Buffer**	**Average Time in Buffer**
Buffers001	14.102	70.40105
Buffers002	14.72068	73.47219
Buffers003	0.6089016	3.042697
Buffers004	0.4704333	2.351047
Buffers005	14.33938	71.54361
Buffers006	14.79936	73.82104

5.8 Production Line With Feedback

The model was extracted from Li (2009). It was outlined in Table 5-37, and consisted of nine machines and two buffers. Machine C1 has two feeds. The first feed comes from the new raw materials arriving from Machine M1. The second feed comes from Machine F3, which processes the re-worked parts. Machine C2 inspects the parts. If the part passes the inspection, it will be forwarded to the second buffer. If the part fails, it will be sent to re-work line as shown in Figure 5-11. It should be mentioned that the simulation run time was not specified in the paper.

Table 5-37 Specs for Model With Feedback

Element	Description			
Raw Material Arrival Rate	Unlimited Supply			
Machine	Nine single machines			
Buffer	Two buffers of size 20			
Rework	N/A			
Setup	N/A			
Breakdown	Machine	MTTR (min)	MTBF(min)	Cycle Time
	M1	2.1	31.2	29.7
	C1	2.9	25.4	30.2
	M2	1.9	28.1	29.6
	M3	2.0	26.2	28.8
	C2	3.1	21.3	27.7
	M4	2.1	29.1	28.2
	F1	1.1	178.2	15.4
	F2	1.2	179.1	15.1
	F3	0.9	181.3	15.2
Labor	N/A			
Simulation Run Time	N/A			
Warm-up Time	N/A			

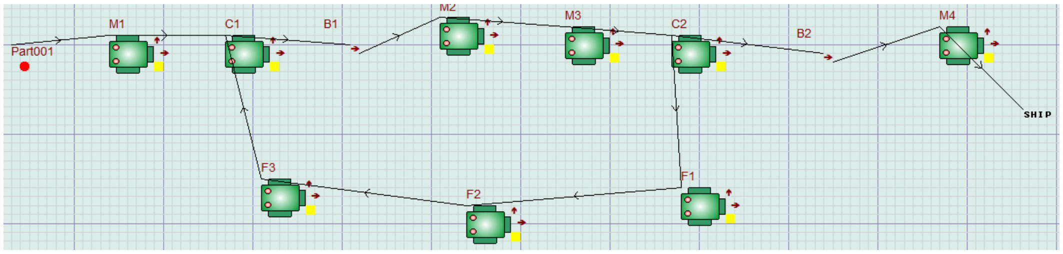


Figure 5-11 Model With Feedback Process

Table 5-38 Production Quantities

Machine Added To The Model	Production Quantity
Base Model	8,305
M1	8,117
M2	8,708
M3	8,305
M4	8,305
C1	8,305
C2	8,611
F1	8,305
F2	8,305
F3	8,305

The author has indicated that the bottleneck in the main path is C2; in the feedback loop path, it is F1. Therefore, the true bottleneck is C2 according to the author. The starving and blockage percentages presented in the paper are shown in Table 5-39. Table 5-40 shows the blockage and starvation that was obtained from the experiment conducted in this study. The base model produced 8,305 parts. When C1, M3, M4, F1, F2, and F3 were added, the model produced 8,305 parts. Production decreased to 8,117 parts when M1 was added, as the system froze after 5857.38 minutes. When C2 was added, the system produced 8,611 parts. Surprisingly, when M2 was added, the system produced 8,708 parts. The result obtained in this study conflicts with the conclusion stated in the original paper. In other words, we have concluded that the true bottleneck is M2.

Turning point and inactive duration methods indicate that the bottleneck is C2. Moreover, the inactive duration method showed that adding M2 to the system could also increase productivity. All the methods failed to identify the true bottleneck. It was concluded that the methods were not able to identify the correct bottleneck due to the configuration of the model. Normally, the bottleneck resource would cause the upstream resources to be blocked and the downstream resources to be idle. Table 5-39 shows that all the resources have been blocked and starved at the same time. This contradicts with the normal bottleneck definition. In addition, adding M2 will increase the blockage on M2 as shown in Figure 5-12 while C2 is never blocked . This is a very interesting model that needs to be further investigated.

Table 5-39 Resources Statistics (Li, 2009)

Machine	% Blocked	%Starved
M1	26.59	0
C1	7.67	4.66
M2	18.66	0.09
M3	13.69	7.07
C2	0.28	16.67
M4	0	30.36
F1	0.73	94.84
F2	2.17	93.47
F3	12.27	83.36

Table 5-40 Resources Statistic

Machine	% Blocked	%Starved
M1	24.56	0
C1	9.13	5.80
M2	21.20	0.02
M3	14.23	8.07
C2	0.05	17.99
M4	0.00	27.72
F1	0.19	97.25
F2	0.53	96.91
F3	5.19	92.41

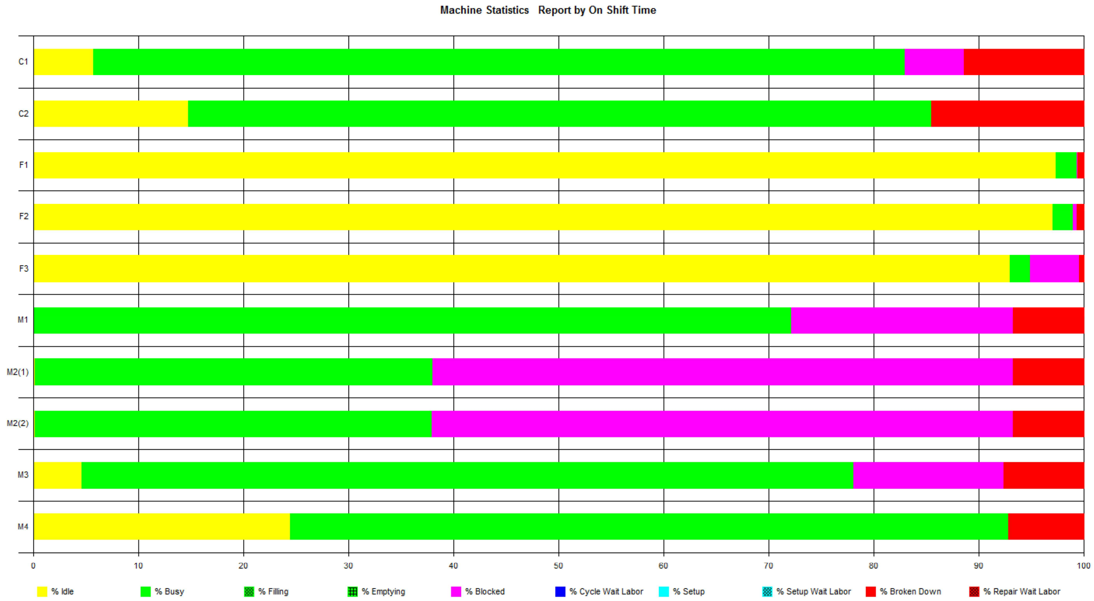


Figure 5-12 Resources Utilization After Adding M2

5.9 Evaluation of the Algorithms

Table 5-41 shows strong methods such as percentage utilization, maximum average duration, shifting bottleneck, inactive duration, and critical indicator methods. Also it shows challenging case study scenarios such as models with shared resources and feedback. As stated earlier, the extracted model to present the feedback scenario was challenging and our finding contradict with the result found in the literature. Based on the result obtained from Table 5-41 and Table 3-72 it was concluded that percentage utilization, maximum average duration, and critical indicators will be elected to be used in the decision support system.

Table 5-41 Performance Measurements Matrix

	Bottleneck Location in the Model (Machine)			**Non-Machine Bottleneck**	**Multiple Bottlenecks**			
Bottleneck Analysis Method	**Beginning**	**Middle**	**End**	**Labor/AGV**	**Machines**	**Labor/AGVs**	**Concurrent Process**	**Feedback**
Percentage Utilization	Yes	Yes	Yes	Yes	Yes - Yes	Yes	Yes	No
Waiting Time Queue	Yes	Yes	No	No	Yes – Yes	No	No	No
Simulation–based Procedure	Yes	Yes	Yes	No	Yes – Yes	No	Yes	No
Maximum Average Active Duration	Yes	Yes	Yes	Yes	Yes – Yes	Yes	Yes	No
Turning Point	Yes	No	Yes	No	Yes – Yes	No	No	Yes
Shifting Bottleneck	Yes	Yes	Yes	Yes	Yes – Yes	Yes	Yes	No
Inactive Duration	Yes	Yes	Yes	Yes	Yes – Yes	Yes	Yes	Yes
Critical Indicator	Yes	Yes	Yes	No	Yes - Yes	No	Yes	No

- Yes - the method is able to detect the correct bottleneck in the case scenario.
- No - the method is not able to detect the correct bottleneck in the case scenario.

Chapter 6

Financial Analysis

This chapter demonstrates the process of justifying the cost associated with two examples. The first experiment will be enhancing the system to meet an increase in the market demand. The second experiment will be for tuning the system to meet the target of the market demand.

6.1 Increase In Market Demand

A model extracted from D'Souza (2004) will be used as an example of a production line, details of which are shown in Table 6-1 and Figure 6-1. The cost justification methodology starts by calculating the equity cash flow followed by the calculation of the net present value. If the net present value is positive, the enhancement is deemed desirable. Otherwise, it is not justifiable. Stevens (1994) describes the financial problem in the following scenario: A company is considering a project that has an initial expenditure of $100,000, with a salvage value of $10,000 at the end of ten years. It is estimated that this initial expenditure will increase annual gross income and costs by $40,000 and $10,000 respectively for the next ten years. The company's tax rate is 40% and for tax purposes, the initial expenditure will be depreciated on a straight-line basis assuming no working capital consideration or investments tax credit. It was assumed that the debt ratio is 40% and the debt is to be paid in ten equal principle installments with interest assessed on the unpaid balance at the rate of 10%. In addition, the minimum acceptable rate of return (MARR) would be assumed to be 16%. The study, also, assumes that the capital investment will start in the first year, and the tax depreciation for the capital expenditure will start immediately after the capital has been allocated.

Table 6-1 Model Spec

Element	Description
Raw Material Arrival Rate	1 part every 2 minutes
Machine	Four single machines, Exponential cycle times: 3, 2, 2, and 5 minutes respectively
Buffer	Three buffers of size 15
Setup	N/A
Breakdown	N/A
Labor	N/A
Simulation Run Time	700 hours
Warm-up Time	120 hours

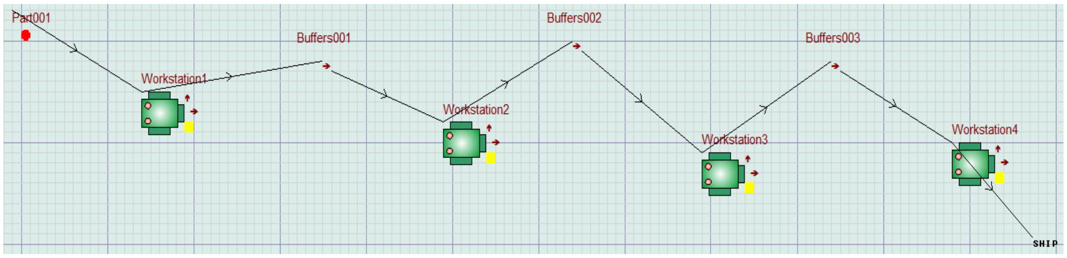


Figure 6-1 Model Diagram

Table 6-2 Resources Cost

No	Machine Name	Initial Cost	Salvage Value	Operating Cost	Life (years)
1	Workstation1	50,000	5,000	5,000	10
2	Workstation2	10,000	1,000	1,000	10
3	Workstation3	15,000	1,500	1,500	10
4	Workstation4	25,000	2,500	2,500	10

Table 6-3 Financial Parameters

Element	Description
Tax Rate	40%
Depreciation Method	Straight-line
Debt Ratio	40%
MARR	16%
Cost of Debt Ratio	10%

The base model produced 114,811 units with a target demand of 400,000 units. The sales price provided was $0.44 per unit. The equity cash flow for the base model is outlined in Table 6-4. Based on the bottleneck analysis conducted on the model, the following resources need to be added to the production system in order to reach the target demand: M4, M1, M4, M3, M2, M4, M1 and M4. The equity cash flow for each enhancement is also shown in the tables below. The NPV for each enhancement is positive, which is an indication that the enhancement is financially desirable. The summary of the manufacturing quantities produced and the net present values for the needed enhancement of the project lifetime is shown in Figure 6-2.

The x-axis of the graph represents the enhancement suggested by the decision support system. The left y-axis represents the production quantities, whereas, the right y-axis represents the NPV of the cash flow. The graph reveals that the first enhancement will increase the production line throughput, which in turn will be translated into higher cash flow. The cost of second enhancement, on the other hand, is higher in terms of capital investment. This change will lower the return of the capital investment even though the production is going up.

Moreover, it was noted that going from point 3 to point 4 did not increase throughput or profitability. Therefore, an experiment was conducted to skip the fourth enhancement and proceed directly from point 3 to point 5. The system produced 220,717 units, which is less than what the DSS originally estimated, which indicates that in order to reach the target, the enhancements must be followed in a specific order.

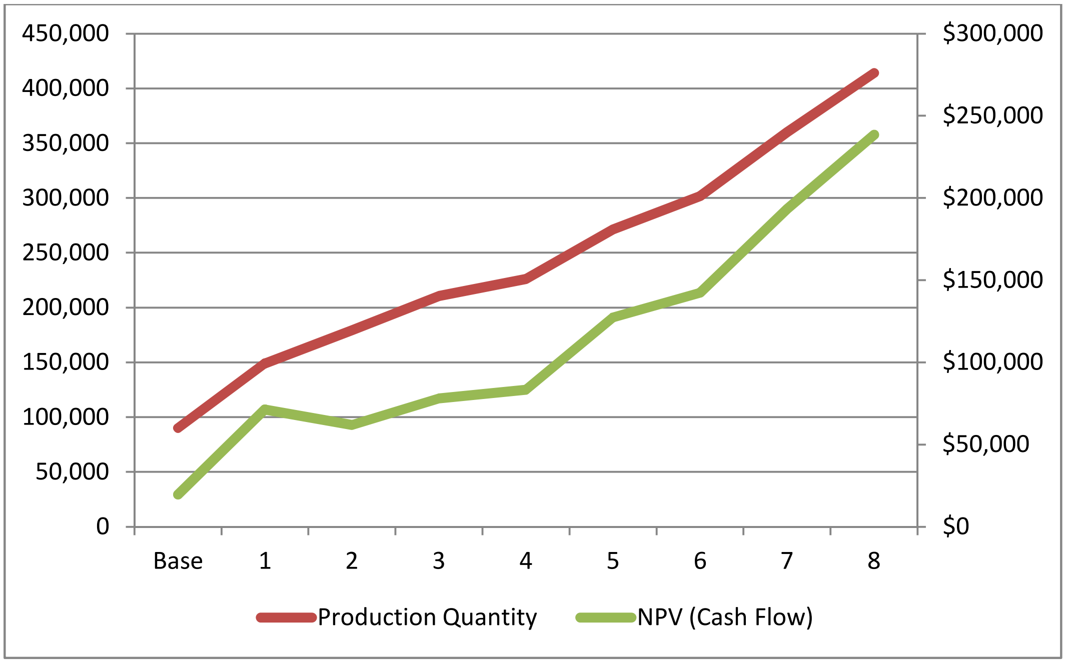


Figure 6-2 Production Quantity, NPV (Cash Flow) vs. Enhancements

Table 6-4 Equity Cash Flow for Base Model

Year	Initial Cost	Salvage	Gross Income	Annual Cost	Debt Interest	Tax Dep	B_0	P	G-C-I	(G-C-I-D)T	Equity Cash Flow	16%
0	100,000	0					40,000	0	0	0	**-60,000**	-60,000
1		0	40,000	10,000	4,000	9,000	0	4,000	26,000	6,800	**15,200**	13,103
2		0	40,000	10,000	3,600	9,000	0	4,000	26,400	6,960	**15,440**	11,474
3		0	40,000	10,000	3,200	9,000	0	4,000	26,800	7,120	**15,680**	10,046
4		0	40,000	10,000	2,800	9,000	0	4,000	27,200	7,280	**15,920**	8,792
5		0	40,000	10,000	2,400	9,000	0	4,000	27,600	7,440	**16,160**	7,694
6		0	40,000	10,000	2,000	9,000	0	4,000	28,000	7,600	**16,400**	6,731
7		0	40,000	10,000	1,600	9,000	0	4,000	28,400	7,760	**16,640**	5,888
8		0	40,000	10,000	1,200	9,000	0	4,000	28,800	7,920	**16,880**	5,149
9		0	40,000	10,000	800	9,000	0	4,000	29,200	8,080	**17,120**	4,502
10		10,000	40,000	10,000	400	9,000	0	4,000	29,600	8,240	**27,360**	6,202
											NPV	19,581

Table 6-5 Equity Cash Flow Adding M4 - 1st Enhancement

Year	**Initial Cost**	**Salvage**	**Gross Income**	**Annual Cost**	**Debt Interest**	**Tax Dep**	**B_0**	**P**	**G-C-I**	**(G-C-I-D)T**	**Equity Cash Flow**	**16%**
0	125,000	0					50,000	0	0	0	**-75,000**	-75,000
1		0	66,142	12,500	5,000	11,250	0	5,000	48,642	14,957	**28,685**	24,728
2		0	66,142	12,500	4,500	11,250	0	5,000	49,142	15,157	**28,985**	21,541
3		0	66,142	12,500	4,000	11,250	0	5,000	49,642	15,357	**29,285**	18,762
4		0	66,142	12,500	3,500	11,250	0	5,000	50,142	15,557	**29,585**	16,339
5		0	66,142	12,500	3,000	11,250	0	5,000	50,642	15,757	**29,885**	14,229
6		0	66,142	12,500	2,500	11,250	0	5,000	51,142	15,957	**30,185**	12,389
7		0	66,142	12,500	2,000	11,250	0	5,000	51,642	16,157	**30,485**	10,786
8		0	66,142	12,500	1,500	11,250	0	5,000	52,142	16,357	**30,785**	9,390
9		0	66,142	12,500	1,000	11,250	0	5,000	52,642	16,557	**31,085**	8,174
10		12,500	66,142	12,500	500	11,250	0	5,000	53,142	16,757	**43,885**	9,948
											NPV	71,286

Table 6-6 Equity Cash Flow Adding M1 - 2nd Enhancement

Year	**Initial Cost**	**Salvage**	**Gross Income**	**Annual Cost**	**Debt Interest**	**Tax Dep**	**B_0**	**P**	**G-C-I**	**(G-C-I-D)T**	**Equity Cash Flow**	**16%**
0	175,000	0					70,000	0	0	0	**-105,000**	-105,000
1		0	79,511	17,500	7,000	15,750	0	7,000	55,011	15,704	**32,307**	27,851
2		0	79,511	17,500	6,300	15,750	0	7,000	55,711	15,984	**32,727**	24,321
3		0	79,511	17,500	5,600	15,750	0	7,000	56,411	16,264	**33,147**	21,236
4		0	79,511	17,500	4,900	15,750	0	7,000	57,111	16,544	**33,567**	18,539
5		0	79,511	17,500	4,200	15,750	0	7,000	57,811	16,824	**33,987**	16,181
6		0	79,511	17,500	3,500	15,750	0	7,000	58,511	17,104	**34,407**	14,122
7		0	79,511	17,500	2,800	15,750	0	7,000	59,211	17,384	**34,827**	12,323
8		0	79,511	17,500	2,100	15,750	0	7,000	59,911	17,664	**35,247**	10,751
9		0	79,511	17,500	1,400	15,750	0	7,000	60,611	17,944	**35,667**	9,379
10		17,500	79,511	17,500	700	15,750	0	7,000	61,311	18,224	**53,587**	12,147
											NPV	61,849

Table 6-7 Equity Cash Flow Adding M4 - 3rd Enhancement

Year	Initial Cost	Salvage	Gross Income	Annual Cost	Debt Interest	Tax Dep	B_0	P	G-C-I	(G-C-I-D)T	Equity Cash Flow	16%
0	200,000	0					80,000	0	0	0	**-120,000**	-120,000
1		0	93,441	20,000	8,000	18,000	0	8,000	65,441	18,976	**38,464**	33,159
2		0	93,441	20,000	7,200	18,000	0	8,000	66,241	19,296	**38,944**	28,942
3		0	93,441	20,000	6,400	18,000	0	8,000	67,041	19,616	**39,424**	25,257
4		0	93,441	20,000	5,600	18,000	0	8,000	67,841	19,936	**39,904**	22,039
5		0	93,441	20,000	4,800	18,000	0	8,000	68,641	20,256	**40,384**	19,228
6		0	93,441	20,000	4,000	18,000	0	8,000	69,441	20,576	**40,864**	16,772
7		0	93,441	20,000	3,200	18,000	0	8,000	70,241	20,896	**41,344**	14,629
8		0	93,441	20,000	2,400	18,000	0	8,000	71,041	21,216	**41,824**	12,757
9		0	93,441	20,000	1,600	18,000	0	8,000	71,841	21,536	**42,304**	11,124
10		20,000	93,441	20,000	800	18,000	0	8,000	72,641	21,856	**62,784**	14,232
											NPV	78,140

Table 6-8 Equity Cash Flow Adding M3 - 4th Enhancement

Year	**Initial Cost**	**Salvage**	**Gross Income**	**Annual Cost**	**Debt Interest**	**Tax Dep**	**B_0**	**P**	**G-C-I**	**(G-C-I-D)T**	**Equity Cash Flow**	**16%**
0	215,000	0					86,000	0	0	0	**-129,000**	-129,000
1		0	100,218	21,500	8,600	19,350	0	8,600	70,118	20,307	**41,211**	35,526
2		0	100,218	21,500	7,740	19,350	0	8,600	70,978	20,651	**41,727**	31,010
3		0	100,218	21,500	6,880	19,350	0	8,600	71,838	20,995	**42,243**	27,063
4		0	100,218	21,500	6,020	19,350	0	8,600	72,698	21,339	**42,759**	23,615
5		0	100,218	21,500	5,160	19,350	0	8,600	73,558	21,683	**43,275**	20,604
6		0	100,218	21,500	4,300	19,350	0	8,600	74,418	22,027	**43,791**	17,974
7		0	100,218	21,500	3,440	19,350	0	8,600	75,278	22,371	**44,307**	15,677
8		0	100,218	21,500	2,580	19,350	0	8,600	76,138	22,715	**44,823**	13,672
9		0	100,218	21,500	1,720	19,350	0	8,600	76,998	23,059	**45,339**	11,922
10		21,500	100,218	21,500	860	19,350	0	8,600	77,858	23,403	**67,355**	15,268
											NPV	83,331

Table 6-9 Equity Cash Flow Adding M2 - 5th Enhancement

Year	**Initial Cost**	**Salvage**	**Gross Income**	**Annual Cost**	**Debt Interest**	**Tax Dep**	**B_0**	**P**	**G-C-I**	**(G-C-I-D)T**	**Equity Cash Flow**	**16%**
0	230,000	0					92,000	0	0	0	**-138,000**	-138,000
1		0	120,379	23,000	9,200	20,700	0	9,200	88,179	26,992	**51,987**	44,817
2		0	120,379	23,000	8,280	20,700	0	9,200	89,099	27,360	**52,539**	39,045
3		0	120,379	23,000	7,360	20,700	0	9,200	90,019	27,728	**53,091**	34,013
4		0	120,379	23,000	6,440	20,700	0	9,200	90,939	28,096	**53,643**	29,627
5		0	120,379	23,000	5,520	20,700	0	9,200	91,859	28,464	**54,195**	25,803
6		0	120,379	23,000	4,600	20,700	0	9,200	92,779	28,832	**54,747**	22,471
7		0	120,379	23,000	3,680	20,700	0	9,200	93,699	29,200	**55,299**	19,567
8		0	120,379	23,000	2,760	20,700	0	9,200	94,619	29,568	**55,851**	17,036
9		0	120,379	23,000	1,840	20,700	0	9,200	95,539	29,936	**56,403**	14,831
10		23,000	120,379	23,000	920	20,700	0	9,200	96,459	30,304	**79,955**	18,125
											NPV	127,334

Table 6-10 Equity Cash Flow Adding M4 - 6th Enhancement

Year	**Initial Cost**	**Salvage**	**Gross Income**	**Annual Cost**	**Debt Interest**	**Tax Dep**	**B_0**	**P**	**G-C-I**	**(G-C-I-D)T**	**Equity Cash Flow**	**16%**
0	255,000	0					102,000	0	0	0	**-153,000**	-153,000
1		0	133,858	25,500	10,200	22,950	0	10,200	98,158	30,083	**57,875**	49,892
2		0	133,858	25,500	9,180	22,950	0	10,200	99,178	30,491	**58,487**	43,465
3		0	133,858	25,500	8,160	22,950	0	10,200	100,198	30,899	**59,099**	37,862
4		0	133,858	25,500	7,140	22,950	0	10,200	101,218	31,307	**59,711**	32,978
5		0	133,858	25,500	6,120	22,950	0	10,200	102,238	31,715	**60,323**	28,720
6		0	133,858	25,500	5,100	22,950	0	10,200	103,258	32,123	**60,935**	25,010
7		0	133,858	25,500	4,080	22,950	0	10,200	104,278	32,531	**61,547**	21,777
8		0	133,858	25,500	3,060	22,950	0	10,200	105,298	32,939	**62,159**	18,960
9		0	133,858	25,500	2,040	22,950	0	10,200	106,318	33,347	**62,771**	16,506
10		25,500	133,858	25,500	1,020	22,950	0	10,200	107,338	33,755	**88,883**	20,148
											NPV	142,318

Table 6-11 Equity Cash Flow Adding M1 - 7th Enhancement

Year	**Initial Cost**	**Salvage**	**Gross Income**	**Annual Cost**	**Debt Interest**	**Tax Dep**	**B_0**	**P**	**G-C-I**	**(G-C-I-D)T**	**Equity Cash Flow**	**16%**
0	280,000	0					112,000	0	0	0	**-168,000**	-168,000
1		0	159,786	28,000	11,200	25,200	0	11,200	120,586	38,154	**71,231**	61,406
2		0	159,786	28,000	10,080	25,200	0	11,200	121,706	38,602	**71,903**	53,436
3		0	159,786	28,000	8,960	25,200	0	11,200	122,826	39,050	**72,575**	46,496
4		0	159,786	28,000	7,840	25,200	0	11,200	123,946	39,498	**73,247**	40,454
5		0	159,786	28,000	6,720	25,200	0	11,200	125,066	39,946	**73,919**	35,194
6		0	159,786	28,000	5,600	25,200	0	11,200	126,186	40,394	**74,591**	30,615
7		0	159,786	28,000	4,480	25,200	0	11,200	127,306	40,842	**75,263**	26,630
8		0	159,786	28,000	3,360	25,200	0	11,200	128,426	41,290	**75,935**	23,162
9		0	159,786	28,000	2,240	25,200	0	11,200	129,546	41,738	**76,607**	20,144
10		28,000	159,786	28,000	1,120	25,200	0	11,200	130,666	42,186	**105,279**	23,865
											NPV	193,404

Table 6-12 Equity Cash Flow Adding M4 - 8th Enhancement

Year	**Initial Cost**	**Salvage**	**Gross Income**	**Annual Cost**	**Debt Interest**	**Tax Dep**	**B_0**	**P**	**G-C-I**	**(G-C-I-D)T**	**Equity Cash Flow**	**16%**
0	305,000	0					122,000	0	0	0	**-183,000**	-183,000
1		0	183,661	30,500	12,200	27,450	0	12,200	140,961	45,404	**83,356**	71,859
2		0	183,661	30,500	10,980	27,450	0	12,200	142,181	45,892	**84,088**	62,491
3		0	183,661	30,500	9,760	27,450	0	12,200	143,401	46,380	**84,820**	54,341
4		0	183,661	30,500	8,540	27,450	0	12,200	144,621	46,868	**85,552**	47,250
5		0	183,661	30,500	7,320	27,450	0	12,200	145,841	47,356	**86,284**	41,081
6		0	183,661	30,500	6,100	27,450	0	12,200	147,061	47,844	**87,016**	35,715
7		0	183,661	30,500	4,880	27,450	0	12,200	148,281	48,332	**87,748**	31,048
8		0	183,661	30,500	3,660	27,450	0	12,200	149,501	48,820	**88,480**	26,989
9		0	183,661	30,500	2,440	27,450	0	12,200	150,721	49,308	**89,212**	23,459
10		30,500	183,661	30,500	1,220	27,450	0	12,200	151,941	49,796	**120,444**	27,303
											NPV	238,535

6.2 Reduction of System Resources to Meet Target Demand

The model discussed in 5.6 will be used in this section to demonstrate how the DSS developed can be used to tune production line. The model was adjusted to have the minimum number of resources as shown in Table 6-13. The cost justification methodology, demonstrated in section 2.5, starts by calculating the minimum annual revenue requirements, followed by the calculation of the net present value. If the net present value is positive, the enhancement is deemed desirable. Otherwise, it is not justifiable. The financial problem description as follows: A company is considering a project that has an initial expenditure of \$100,000, with a salvage value of \$10,000 at the end of ten years. It is estimated that the annual costs is 10% of the initial expenditure for the next ten years. The company's tax rate is 40% and for tax purposes, the initial expenditure will be depreciated on a straight-line basis assuming no working capital consideration or investments tax credit. It was assumed that the debt ratio is 40% and the debt is to be paid in ten equal principle installments with interest assessed on the unpaid balance at the rate of 10%. In addition, the minimum acceptable rate of return (MARR) would be assumed to be 20%. The study, also, assumes that the capital investment will start in the first year, and the tax depreciation for the capital expenditure will start immediately after the capital has been allocated.

Table 6-13 Model Spec

Element	Description
Raw Material Arrival Rate	16 jobs per hour
Machine	Four machines, Exponential cycle times: 3, 3.5, 2, and 2 minutes respectively
Buffer	Three buffers of size 15
Breakdown	N/A
Setup	N/A
Labor	One labor resources
Simulation Run Time	700 hours
Warm-up Time	120 hours

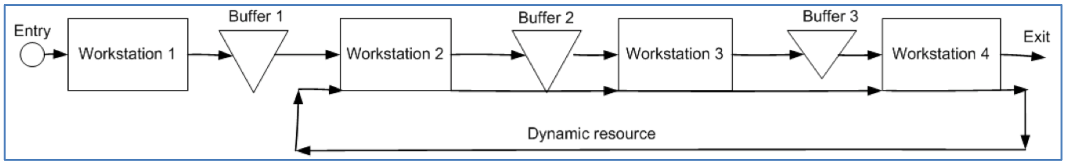


Figure 6-3 Model Diagram

Table 6-14 Resources Cost

No	Machine Name	Initial Cost	Salvage Value	Operating Cost	Life (years)
1	Machine001	50,000	5,000	5,000	10
2	Machine002	10,000	1,000	1,000	10
3	Machine003	15,000	1,500	1,500	10
4	Machine004	25,000	2,500	2,500	10
5	Labor	0	0	100,000	10

Table 6-15 Financial Parameters

Element	Description
Tax Rate	40%
Depreciation Method	Straight-line
Debt Ratio	40%
MARR	20%
Cost of Debt Ratio	10%

The base model produced 10,550 unites per 700 hours. The target market demand was set to 140,000 units per 700 hours. The objective of this exercise is to increase the productivity to reach the market demand. It is required to acquire a return on the capital investment after making the proper enhancement to reach the target market demand. If the return on the capital investment is not possible, the production quantity should be reduced. For example, adding the first and second labor will have acquire a return on the investment as long as the minimum sale price is $2.29 and $2.69 respectively as shown in Table 6-16 and Figure 6-4. If the market demand changes, Table 6-16 and Figure 6-4 can also be used to determine the number of resources needed to meet the new target market demand.

Table 6-16 Production Quantity For Each Enhancement

No.	Level	Production	Minimum Sale Price
1	Base	59,597	3
2	Adding Labor	87,915	2.288
3	Adding Labor	103,479	2.689
4	Adding Labor	112,780	3.251
5	Adding Machine002	143,952	3.869

Figure 6-4 Production Quantity vs. Minimum Sale Price

Table 6-17 Minimum Annual Revenue Requirement – New Base Model

Year	Book Depreciation	Tax Depreciation	Book Value	Equity Return	Debt Interest	Tax	Annual Cost	R
0			100,000					**0**
1	9,000	9,000	91,000	12,000	4,000	8,000	110,000	**143,000**
2	9,000	9,000	82,000	10,920	3,640	7,280	110,000	**140,840**
3	9,000	9,000	73,000	9,840	3,280	6,560	110,000	**138,680**
4	9,000	9,000	64,000	8,760	2,920	5,840	110,000	**136,520**
5	9,000	9,000	55,000	7,680	2,560	5,120	110,000	**134,360**
6	9,000	9,000	46,000	6,600	2,200	4,400	110,000	**132,200**
7	9,000	9,000	37,000	5,520	1,840	3,680	110,000	**130,040**
8	9,000	9,000	28,000	4,440	1,480	2,960	110,000	**127,880**
9	9,000	9,000	19,000	3,360	1,120	2,240	110,000	**125,720**
10	9,000	9,000	10,000	2,280	760	1,520	110,000	**123,560**
							NPV	106,733

Table 6-18 Minimum Annual Revenue Requirement – 1st Enhancement Adding Second Labor

Year	**Book Depreciation**	**Tax Depreciation**	**Book Value**	**Equity Return**	**Debt Interest**	**Tax**	**Annual Cost**	**R**
0			100,000					**0**
1	9,000	9,000	91,000	12,000	4,000	8,000	210,000	**243,000**
2	9,000	9,000	82,000	10,920	3,640	7,280	210,000	**240,840**
3	9,000	9,000	73,000	9,840	3,280	6,560	210,000	**238,680**
4	9,000	9,000	64,000	8,760	2,920	5,840	210,000	**236,520**
5	9,000	9,000	55,000	7,680	2,560	5,120	210,000	**234,360**
6	9,000	9,000	46,000	6,600	2,200	4,400	210,000	**232,200**
7	9,000	9,000	37,000	5,520	1,840	3,680	210,000	**230,040**
8	9,000	9,000	28,000	4,440	1,480	2,960	210,000	**227,880**
9	9,000	9,000	19,000	3,360	1,120	2,240	210,000	**225,720**
10	9,000	9,000	10,000	2,280	760	1,520	210,000	**223,560**
							NPV	68,885

Table 6-19 Minimum Annual Revenue Requirement – 2nd Enhancement Adding Third Labor

Year	Book Depreciation	Tax Depreciation	Book Value	Equity Return	Debt Interest	Tax	Annual Cost	R
0			100,000					**0**
1	9,000	9,000	91,000	12,000	4,000	8,000	310,000	**343,000**
2	9,000	9,000	82,000	10,920	3,640	7,280	310,000	**340,840**
3	9,000	9,000	73,000	9,840	3,280	6,560	310,000	**338,680**
4	9,000	9,000	64,000	8,760	2,920	5,840	310,000	**336,520**
5	9,000	9,000	55,000	7,680	2,560	5,120	310,000	**334,360**
6	9,000	9,000	46,000	6,600	2,200	4,400	310,000	**332,200**
7	9,000	9,000	37,000	5,520	1,840	3,680	310,000	**330,040**
8	9,000	9,000	28,000	4,440	1,480	2,960	310,000	**327,880**
9	9,000	9,000	19,000	3,360	1,120	2,240	310,000	**325,720**
10	9,000	9,000	10,000	2,280	760	1,520	310,000	**323,560**
							NPV	-65,210

Table 6-20 Minimum Annual Revenue Requirement – 3rd Enhancement Adding Forth Labor

Year	Book Depreciation	Tax Depreciation	book Value	Equity Return	Debt Interest	Tax	Annual Cost	R
0			100,000					**0**
1	9,000	9,000	91,000	12,000	4,000	8,000	410,008	**443,008**
2	9,000	9,000	82,000	10,920	3,640	7,280	410,008	**440,848**
3	9,000	9,000	73,000	9,840	3,280	6,560	410,008	**438,688**
4	9,000	9,000	64,000	8,760	2,920	5,840	410,008	**436,528**
5	9,000	9,000	55,000	7,680	2,560	5,120	410,008	**434,368**
6	9,000	9,000	46,000	6,600	2,200	4,400	410,008	**432,208**
7	9,000	9,000	37,000	5,520	1,840	3,680	410,008	**430,048**
8	9,000	9,000	28,000	4,440	1,480	2,960	410,008	**427,888**
9	9,000	9,000	19,000	3,360	1,120	2,240	410,008	**425,728**
10	9,000	9,000	10,000	2,280	760	1,520	410,008	**423,568**
							NPV	-246,589

Table 6-21 Minimum Annual Revenue Requirement – 4th Enhancement Adding Second Machine002

Year	Book Depreciation	Tax Depreciation	book Value	Equity Return	Debt Interest	Tax	Annual Cost	R
0			110,000					**0**
1	9,900	9,900	100,100	13,200	4,400	8,800	411,000	**447,300**
2	9,900	9,900	90,200	12,012	4,004	8,008	411,000	**444,924**
3	9,900	9,900	80,300	10,824	3,608	7,216	411,000	**442,548**
4	9,900	9,900	70,400	9,636	3,212	6,424	411,000	**440,172**
5	9,900	9,900	60,500	8,448	2,816	5,632	411,000	**437,796**
6	9,900	9,900	50,600	7,260	2,420	4,840	411,000	**435,420**
7	9,900	9,900	40,700	6,072	2,024	4,048	411,000	**433,044**
8	9,900	9,900	30,800	4,884	1,628	3,256	411,000	**430,668**
9	9,900	9,900	20,900	3,696	1,232	2,464	411,000	**428,292**
10	9,900	9,900	11,000	2,508	836	1,672	411,000	**425,916**
							NPV	-20,477

Chapter 7

Real World Manufacturing Facility Analysis

This chapter will summarize the analysis of a real world model. The models used in Chapter 3 and Chapter 5 were trivial in terms of size and complexity. Therefore, there is a need to evaluate the decision support system developed in this study against a real-world system. The simulation model differentiate itself from the trivial simulation model by having more sophisticated configuration such as multi-cycle time machines, timed buffers, complex input/output rules, etc.

The model used in this study is one that was developed to simulate, study, and analyze a silicon wafer production facility. The example will illustrate how the decision support system developed in this study can be used to improve productivity and can provide cost justification associated with such an improvement. The company's target is to be capable of producing 3 Megawatts per year. According to the manufacturer, it is safe to assume that each produced cell will generate 1.1 watt. Figure 7-1 shows the simulation model for the facility, and appendix A contains the facility processing description. The cost of the resources required for the base model is also included in the appendix. The base model developed for the facility has a production forecast of 4,359 packages per year, with each package containing 100 cells. The financial parameters used are as follows: tax rate is 40%, straight-line depreciation, no working capital consideration or investments tax credit, debt ratio of 40%, debt is to be paid in ten equal principle installments, and debt cost is 10%. In addition, the minimum acceptable rate of return (MARR) is assumed to be 20%. Furthermore, it was presumed that the labor resources will always be available, and hence no working shift will be configured.

The decision support system developed in this study has automatically added variables that would capture the data needed by the bottleneck detection methods as explained in Chapter 4. Due to the complexity of the model presented in this chapter, it was recognized that the decision support system is not able to automate the process of inserting the variables to the simulation model due to the lack of automatically knowing the relationship between the shared resources and the machines in the simulation model. Therefore, the variable to capture the data for the shared resources were added manually.

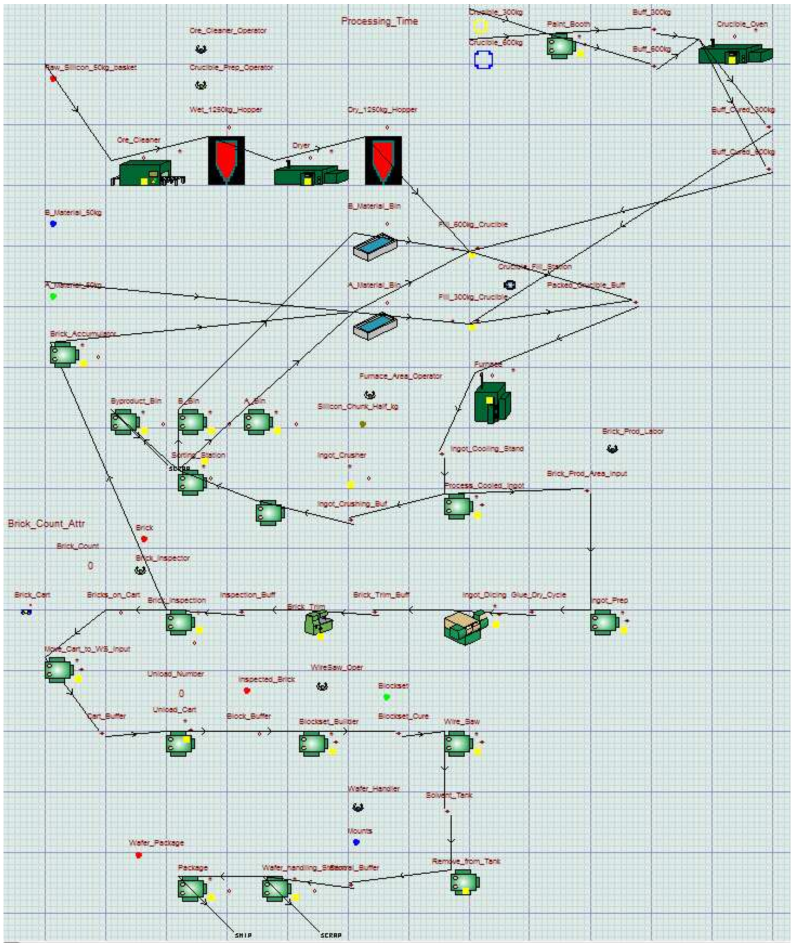


Figure 7-1 Silicon Wafer Production Facility Simulation-based Model

7.1 Use of the Decision Support System To Enhance Throughput

There were 13 enhancements conducted against the base simulation model to reach to the target market demand. Figure 7-1 shows the base simulation model that produces 4,358 units per year. Figure 7-2 shows the final simulation model that will produce 30,680 units per year. According to the methodology proposed by this study, there have been 13 enhancements to reach to the target market demand. All the enhancement are discussed in details in appendix B. this section will highlight the major finding in this implementation.

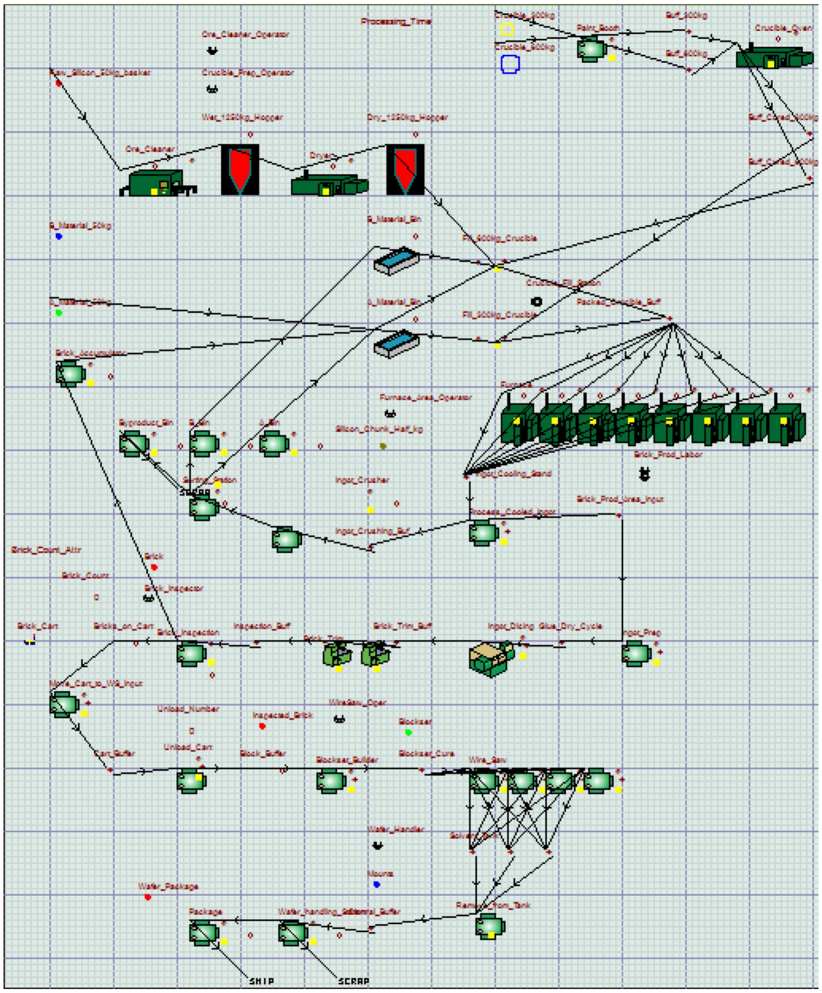


Figure 7-2 Production Line After 14 Enhancements

The first challenge occurred in the process was when the critical indicator method failed to identify the correct bottleneck in the second enhancement, appendix B section 1.2. The true bottleneck was the Wire_Saw when the critical indicator method was point at the furnace. The reason the critical indicator method failed to identify the correct the bottleneck was due to the fact there was a period of the time where the Wire_Saw was inactive due to breakdown. The critical indicator method does not consider breakage time in the equation to determine the bottleneck, Equation 2-10, and hence was not able to detect the bottleneck. Another flaw with Equation 2-10 was discussed in appendix B section 1.6. In this scenario the true bottleneck was the labor resource Brick_Prod_Labor. The reason is that Ingot_Dicing spent more time waiting on labor rather than being busy. The equation for the critical indicator, Equation 2-10, subtracts the average waiting on labor for the entire system from the individual waiting on labor for each machine. Since Ingot_Dicing spent more time waiting on labor than the length of time that the furnaces were blocked, the critical indicator method returned a false result.

Most of the bottleneck detection methods analyze machines and shared resources only. However, other elements in the simulation model can be the bottleneck. During this analysis, the timed buffer resource “Solvent_Tank” was identified as the true bottleneck, details can be found in appendix B section 1.3 and 1.10. Solvent_Tank buffer was invisible to the bottleneck detection methods. Even after converting the timed buffer resource “Solvent_Tank” to a machine, none of the bottleneck detection methods were able to identify it as the bottleneck. Based on the analysis and finding of this analysis we have concluded that the bottleneck detection methods failed to identify the correct bottleneck due to the unique behavior of the simulation model in those type of configuration. In these two scenarios the upstream and the downstream resources suffered from blockage. This is a strange behavior since in a normal case the upstream resources will suffer from blockage, whereas, the downstream resources will be starving. These finding indicate that there is still a need for new bottleneck detection methods that can overcome these challenges.

7.2 Use of the Decision Support System For Cost Justification

This section will follow the illustration contained in Chapter 6. The tax rate will be assumed to be 40% and for taxation purposes, the initial expenditure will be depreciated on a straight-line basis, assuming there is no working capital consideration or investment tax credit. It was also be assumed that the debt ratio is 40% and the debt is to be paid in ten equal principle installments with interest assessed on the unpaid balance at 10%. In addition, the minimum acceptable rate of return (MARR) is assumed to be 20%, which is typical in the solar cell manufacturing market.

The following tables show the cost justification for the enhancements in terms of net present value of the equity cash flow. Figure 7-3 illustrates the effect of each enhancement on the productivity and the net present value generated from equity cash flow. The x-axis represents each enhancement step needed to reach the target market demand. The left y-axis represents the NPV for the cash flow generated for each enhancement. The right y-axis represents the production quantities for each enhancement. Until the point of the sixth enhancement, the project seemed to be undesirable, as we are not receiving a return on the investment. After the sixth enhancement, the company was able to produce a sufficient number of products to generate positive net present value of the equity cash flow. Interestingly, even though on the 12th enhancement the desired demand was not reach, the NPV of the cash flow is higher than the NPV of the cash flow on the 14th enhancement, where the demand was met. It is up to the management to accept the production quantity on the 12th enhancement and acquire higher return on the investment rather than invest more capital in order to increase productivity and reduce the return on the investment. This highlights the importance and of considering the bottleneck detection approach with the financial justification.

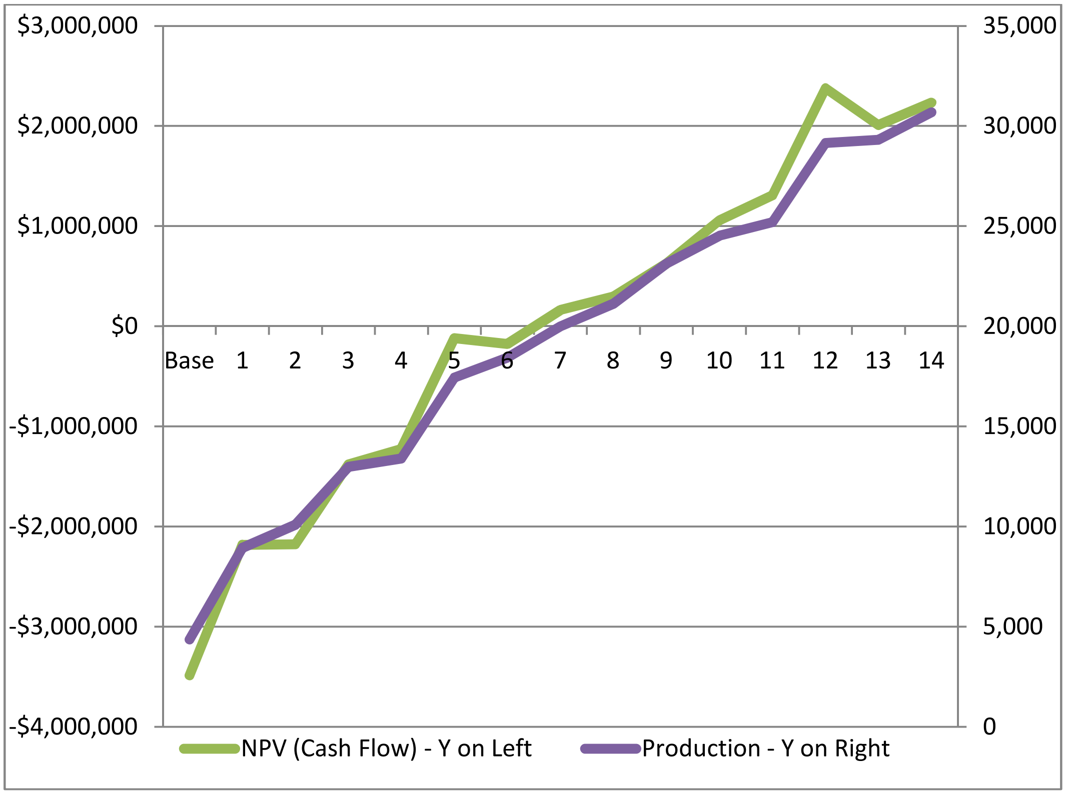


Figure 7-3 NPV of Equity Cash Flow and Productivity for Each Enhancement

Chapter 8

Conclusion And Future Study

8.1 Conclusions

Industry needs a capital investment analysis tool that can integrate production capacity and financial justification methods that will identify the least cost production configurations that can simultaneously meet production capacity and return on investment requirements. A review of the current literature has indicated that simulation-based facility analysis methods that utilize the theory of constraints analysis techniques have proven useful in analyzing production analysis problems of real-world size and complexity. Several authors have proposed bottleneck detection algorithms capable of identifying the capacity constraining resources within a simulated model of a production or service system. These system analysis methods can identify the resource that is limiting the capacity of the entire system. What we did not find, however, was any analysis methods that tightly integrated both capacity constraint production analysis and financial justification methods.

An impetus for this research was the desire for an analysis tool that can automate the identification of the most efficient mix of resources required to meet a production target and simultaneously tell the analyst if the proposed system will satisfy minimum acceptable rate of return requirements established by management. Rather than evaluating a single arbitrary system configuration, the envisioned tool would tell the analyst the most efficient sequence of system configurations that are possible to increase capacity from the minimum prototype production configuration, which contains to minimum set of resources needed to produce a least one product, to an arbitrarily larger system configuration capable of reaching a desired production capacity. At each step in this process, the tool would provide the economic impact of each addition of production capacity. This research will attempt to determine if it is possible to create an automated tool with these capabilities.

To achieve the answer to this research question, a series of tasks had to be completed. We first had to determine if it was possible to automate constraint-based system analysis methods for an arbitrary set of discrete-event production or service scenarios. These automated analysis methods had to be robust enough to analyze industrial scale system configurations. A review of the current literature revealed a number of bottleneck detection methods that could possibly be automated. Once identified, each method was evaluated to see if it was suitable for automation. In Chapter 2, throughput-based, arrow-based, and inter-departure time methods were eliminated from further consideration because the types of system configurations that these methods can analyze are limited. In Chapter 3, each automated bottleneck detection method was developed and then tested on a standard set of production system models to determine if they provided reliable results. The simulation-based method was eliminated because it provided multiple bottleneck candidates thus requiring the analyst to manually interpret the result. In addition, the shifting bottleneck and the inactive duration methods are derivatives of the active duration method. These three methods are computationally intensive and provide similar result. Therefore, we elected to use the active duration method because it provides the results slightly faster than the other two methods. The turning point method was also excluded because it does not support the analysis of shared resources such as labor and cannot easily support the analysis of concurrent processing.

Given the number of bottleneck detection methods proposed in the literature it was clear that no single method could identify the bottleneck resource in every possible system configuration. It was also clear that it would be impractical to implement every method in the tool. It was therefore necessary to identify the smallest possible sub-set of methods that needed to be implemented that would maximize the likelihood that the true bottleneck resource can be identified. To accomplish this task a strategy had to be devised to evaluate the effectiveness of the various algorithms over a representatively broad range of system configurations. A systemized attempted to evaluate the relative performance of each of these methods across a standard set of production scenarios was missing from the literature. Each paper tended to demonstrate how a proposed algorithm could successfully identify the bottleneck in a single system configuration. If alternative methods were compared at all they were compared against the author's method on the model proposed by the author. This research has attempted to identify the most promising algorithms by comparing the performance of all of the automatable bottleneck detection algorithms on all of the proposed production scenario simulations presented by the various authors in the literature. The result of this analysis can be found in Table 5-41. From this additional analysis we have selected the methods: percentage utilization, waiting queue, active duration, and critical indicator as the final methods needed to adequately identify the bottlenecks within our proposed decision support system. The active duration method has been found as the most robust and reliable bottleneck detection method for machines and labor. One drawback with this method is it is computationally complex which requires longer computing time to get the result. The critical indicator is a fast, reliable method. A potential drawback of this method is that it identifies the resource that needs the labor as the bottleneck whereas in reality the labor is the actual bottleneck.

A prototype capital investment decision support tool was designed and implemented that accepted an externally defined discrete-event simulation model and identify the capacity limiting constraint in the model. In addition to identifying the capacity limiting constraint, or bottleneck, the tool had to also provide appropriate financial measures for the system configuration defined in the simulation model. To accomplish this, the decision support system (DSS) first reads in and runs the system's simulation model to determine an estimate for the production capacity of the modeled system. The DSS then prompts the user for the relevant financial values needed to describe the system. With this information, the user of the DSS can select to calculate one of two financial performance measures: the minimum per unit sales price to meet the Minimum Annual Revenue Requirements for the proposed system, or the Net Present Value (NPV) of the system design based on equity cash flows. Both types of financial analysis require that the user to provide the Minimum Acceptable Rate of Return based on the cost of equity. To calculate the NPV the user must be able to provide the anticipated sale price in order to calculate the yearly gross revenue that the system will produce.

After calculating these financial performance measures, the DSS will dynamically add sets of variables to the simulation model that are needed to record the data required by the automated bottleneck detection methods. This expanded simulation model is stored and run by the DSS in order to perform the bottleneck detection task. The output of the DSS will indicate the bottleneck resource identified by each of the bottleneck detection methods. Using this data as a guide, the user will then select the system resource in the model that needs to be increased, or decreased if the desire is to reduce production. The DSS will make the required changes to the system's simulation model and the cycle is then repeated until the target production level is reached. Details of the decision support system can be found in Chapter 4.

One of the limitations of existing bottleneck detection literature is that the example system models used to verify the methods were almost trivial in size and complexity. Actual production or service systems are typically much larger than the models presented in the existing literature. To determine the practical application of this research, we needed to determine if the proposed DSS could support the analysis of projects of the scale and complexity that is typical of real-world systems. To determine the practical application of this research, we used the DSS prototype to analyze an existing system model that represents the actual production processes of a photovoltaic cell production facility. The simulation model was designed to faithfully represent the production processes found in that industry. Chapter 7 presents the results of our efforts to validate that the proposed DSS design can support the analysis of systems of practical size and complexity.

8.2 Future Work

Through the process of completing this research potential areas of future research have been identified. The focus of the proposed research was not to develop new bottleneck detection methods but to identify the best and most automatable methods available in the literature. In Chapter 5, during the evaluation of the bottleneck methods' ability to identify the bottleneck in system models found in the literature, there was one model that we feel could not be appropriately analyzed by the existing bottleneck detection methods (Li, 2009). This model appears to be quite challenging to analyze even manually. For example, the resource we identified as the bottleneck resource does not match the resource identified by the original author as the bottleneck resource. Of the eight bottleneck detection methods applied to this model only two methods, the original authors and one other, identify the author's selection of the bottleneck resource. None of the seven methods point to the resource we feel is the actual bottleneck resource in this model. We propose that this model will provide an interesting case study example for future bottleneck detection method development efforts.

In the extended systems model presented in Chapter 7 we also identified one system configuration in which all five of the strongest bottleneck detection methods failed to identify the true bottleneck resource discovered through manual analysis. These two examples of models that appear to defy our existing bottleneck detection methods indicate that there is still a need for new bottleneck detection methods.

While working with the case study model presented in Chapter 7, we recognized the need for bottleneck detection methods capable of handling multiple shared resource that must be present to perform a given activity. Labor is classic example of this type of shared resource. The simplified system models presented in the literature did not include complex shared resource scenarios like multi-shift labor or the presence of several different types of labor resources in the model. This is understandable given the complexity of automatically analyzing the model structure of a system model with these complex shared resource relationships and automatically generating the data collection infrastructure needed to support bottleneck detection. Additional research is needed to develop bottleneck detection methods that can detect shared resource constraints and can cope with the complexities of multiple shift environments.

The Decision Support System (DSS) prototype presented in Chapter 4 and used to support our analysis in Chapters 5, 6 and 7, has demonstrated that it is possible to develop a capital investment analysis tool that can integrate production capacity and financial justification methods that will identify the least cost production configurations that can simultaneously meet production capacity and return on investment requirements. It is important to state, however, that at this point, the tool is a prototype not a finished and fully tested software product.

As the size of the system models grew, we noticed that the amount of data needed to support the Active Duration Bottleneck Detection appeared to grow significantly. For example, when using the DSS prototype on the system model presented in Chapter 7 the tool generated eight millions event records over 62 days of simulated product time. The system simulation component of the DSS had to generate these eight million records within the simulation tool and write the records to a flat file. This activity added several minutes to the system simulation time. The time required to perform the Active Duration Bottleneck Detection analysis of the Chapter 7 case study took as long as four hours to process and analyze the eight million event records generated over 62 days of simulated production. This represents the time invested in a single iteration of our proposed multi-step analysis method. Fourteen iterations of the DSS cycle were performed to generate the analysis presented in Chapter 7. Access to stronger computer resources can improve this processing time. For example running the DSS on a dual-core processor with 4 megabytes of RAM took approximately 9 hours to process. Running the same analysis on a quad-core system with 8 megabytes of RAM took 4 hours to complete. Even with stronger computing hardware the analysis of real-world systems of significant size and complexity will require a significant amount of time. Additional research is needed to see if the Active Duration Bottleneck Detection algorithm can be made more computationally efficient. Failure to find a faster method to do this analysis may limit the practical use of this bottleneck detection method on large-scale system models.

We have proved that it is possible to automatically modify the system simulation model to support the automatic system data extraction required for our automatic bottleneck detection methods. All of the models analyzed using the prototype DSS in Chapters 3 and 5 used this automatic model enhancement process. The complexity of the system scenario used in Chapter 7 could not easily be accommodated by our prototype automatic model enhancement techniques. Our model enhancement techniques had difficulty supporting the shared resource (labor) allocation rules and the complex entity routing logic required to capture the behavior of this realistic manufacturing scenario. Additional research is required to expand the capability of our automatic model enhancement capabilities.

Our current financial analysis methods focus on the determination of the system’s Net Present Value generated using the Equity Cash Flow Method if we can estimate the Gross Revenue that will be generated by the system. We have also proposed a method to calculate the minimum sales price per unit measure derived from the Minimum Annual Revenue Requirements method. Future work could include to the addition of Operating and Total Cash Flow methods. The ability to add Internal Rate of Return calculation methods could also be added to the tool.

At this time our financial analysis methods assume that all of our capital investments are made at one time in the beginning of the project (Year Zero). The DSS tool can be expanded to support the analysis of projects that require capital investments in multiple years of the project. This feature will significantly complicate the automatic generation of the cash flow tables. This problem will be even more difficult if we attempt to support the automatic generation of annual cash flow models when multiple pieces for capital equipment are removed from service at different times. Capturing the tax implications associated with removing a capital asset for service before it is fully depreciated for tax purposes can be complex. Although interesting, these capabilities were thought to be beyond the scope of the DSS prototype proposed for this research. These financial analysis features can be added in the future as needed.

Appendix A

Overall Process Flow For Wafer Production

1. Preparing Source Silicon Material

Although the mined silicon material provides a very good precursor material for the polycrystalline wafer production process, it requires cleaning and purification before it can be used for silicon wafer production. The raw silicon will be delivered to the factory in haulage trucks. The raw silicon will be stored in hoppers, which deliver the crushed silicon ore to the factory's ore purification facility. For the purpose of this study, we can assume that there will always be a supply of raw silicon material. The objective of the ore purification process is to remove mineral contaminants and oxides from the raw silicon. This is achieved by first cleaning the silicon ore through a chemical process and then placing the clean ore inside industrial ovens, where it is heated to the point where it is transformed into a molten state. As the ore melts, it releases impurities which tend to congregate at the top or bottom of the molten material. The molten material is left to cool and solidify into a large ingot of material. A hydraulic press is used to fracture the ingot into hand-sized chunks, which are then manually inspected and sorted into three material grades: A-grade silicon, B-grade silicon, and by-product. The A-grade silicon can be used directly as source material for the wafer production process. The B-grade silicon is recycled through the ore purification process. The purification by-products are removed from the factory and hauled back to the mine, where they are used as fill material. The yield from this purification process are 43% A-grade, 36% B-grade, and 21% by-product. The details of the purification process are provided below.

1.1 Raw Silicon Cleaning

The raw silicon material from the mine arrives by truck and dumped into a source hopper, which feeds the material down to the ore cleaning facility. There, the silicon ore is emptied from the hopper into ore cleaning baskets with a holding capacity of 50 kg each. One operator is needed to oversee the basket-filling operation. Once filled, the baskets are conveyed to the ore cleaning system. The ore cleaner uses a chemical process to remove surface contaminants, advancing it through a series of five chemical and rinsing stations: primary rinse, acid bath, base rinse, base bath, and deionized water rinse. Each ore cleaning basket spends 5 minutes in each of the five stations. The baskets cannot be delayed in any of the stations beyond this 5-minute period. Once a basket starts the cleaning process, it must complete all five steps in sequence. The transfer time between each of the five stations is included within the 5-minute station processing time. After the deionized water rinse, the contents of the ore cleaning baskets are dumped into a 1250 kg capacity hopper, where the cleaned ore awaits the drying process. The residual moisture contained in the ore must be dried off before being placed in the high temperature furnaces. This is accomplished by placing 500 kg batches of ore in industrial dryers for 45 minutes. It takes 7 minutes to load and unload the dryer into another 1250 kg hopper, where the ore waits its turn in the high temperature furnaces. An operator will supervise each ore cleaning system that is brought online. If the system is not operating, the operator is free to perform other duties in the crucible preparation area.

1.2 Receiving Crucibles from a Local Vendor and Preparing it for Use

A local vendor would have a contract to provide crucibles for the material purification process and the silicon ingot production process. Two different size crucibles will be delivered by the vendor: a 600 kg capacity crucible (to purify the raw silicon material), and a 300 kg capacity crucible (to produce the silicon ingot used as the solar cell raw material). The vendor delivers the crucibles on a just-in-time basis. We will assume that crucibles of both sizes will be available to the NWE facility as needed. Each crucible must be sealed, and sprayed with a release agent. An operator is responsible for all the crucible preparation activities as described below.

Each crucible must be sprayed with a sealing agent in a paint booth. The 600 kg purification crucible requires 30 minutes to seal, whereas the 300 kg production crucible takes only 20 minutes. The NWE facility has a buffer capacity of three of each type of crucible. After the crubicle has been sprayed with sealant, it must be baked in an oven for 8 hours. The crucible oven can hold four crucibles at a time. Various types of crucibles can be baked in the ovens. There is storage capacity for six of each type of baked crucible.

1.3 Filling the Purification Crucible

Prior to being filled, each crucible must be sprayed with a release agent. A total of 30 minutes is required for the operator to apply chemical to the 600 kg purification crucible. The operator will then fill the purification crucible with 50 kg of A-grade silicon and 550 kg of raw silicon ore and/or B-grade material. The A-grade silicon chunks promote crystal formation in this phase of the purification process. Assume that 500 kg of A-grade material will be available at the facility when operations first begin, and will be stored in a 2000 kg A-grade material bin. The operators will first use as much of the B-grade material as is available before using the raw silicon ore. This material is stored in a 2000 kg B-grade material bin. Since the A and B-grade materials are in the form of irregular-shaped chunks of silicon, we will assume that it takes an operator 5 minutes to pack 50 kg of silicon (of any type) into the crucible. Once packed, the purification crucibles are transferred to a buffer (with a capacity of 5 crucibles of any size) where they wait in sequence to be processed in the furnaces.

1.4 Silicon Purification Cycle

The furnaces can process both purification crucibles and production crucibles. Since a larger mass of material needs to be melted and held in a molten state for longer, the purification crucibles must remain in the furnaces for a cycle time of eighty hours. A furnace area operator requires an average of 30 minutes to load the purification crucible and set the furnace controls. The furnace runs automatically for 80 hours, and then an operator spends approximately 20 minutes safely shutting it down, removing the crucible, and placing it on a cooling stand. The operator is now free to load or unload another furnace. The crucible must remain in the cooling station for 10 hours before it can be further processed. During this cooling cycle, the silicon hardens into a solid ingot, causing the crucible shell to fracture and separate from the ingot.

After the 10-hour cool-down period, the operator returns and spends the next 15 minutes cleaning up the broken crucible pieces and transferring the 600 kg silicon ingot to the crushing station input buffer.

1.5 Silicon Prep Ingot Crushing

The crucible prep operators are responsible for removing 600 kg silicon ingots from the buffer and loading them into a hydraulic crushing station, a 5-minute process. The crushing station fractures the ingot into fist-sized chunks that are automatically dumped onto a conveyor leading to a sorting station. We will assume that it takes about 4 seconds to grade and sort one chunk of silicon in one of three sorting bins, and that the average chunk weighs 0.5 kg. Each sorting station has three sorting bins: an A-grade bin, a B-grade bin, and a by-product bin, each with a capacity of 100 kg of material. Once the A-grade or B-grade bins are filled, the operator dumps their contents into the respective 2000 kg material holding bin, which takes about 10 minutes. The by-product is deposited into a large dumpster, where it is collected and trucked back to the mine. The yield from this purification process are 43% A-grade, 36% B-grade, and 21% by-product.

2. Producing a 300 kg Production-Grade Ingot

The next phase of the process will generate production-grade ingots of polycrystalline silicon that can be sliced into wafers and converted into photovoltaic cells.

2.1 300 kg Production Crucible Preparation

A crucible preparation operator will spend 15 minutes pulling one of the heat-treated 300 kg production crucibles out of its buffer and treating it with the release compound so that its shell cleanly separates from the ingot during the cool-down process. The crucible is then hand-packed with 300 kg of A-grade material. Since the A-grade material is in the form of irregular-shaped chunks of silicon, we will assume that it takes an operator 5 minutes to pack 50 kg of silicon into the crucible. Once packed, the crucible is placed in the furnace area input buffer.

2.2 Production-Grade Ingot Processing

The production crucibles must remain in the furnaces for a sixty (60)-hour cycle time, which includes an extended and controlled solidification cycle to promote maximum crystalline structure formation. A furnace operator needs an average of 30 minutes to load the production crucible and set the furnace controls. The furnace runs automatically for 60 hours, after which the operator spends 20 minutes safely shutting it down, removing the crucible, and placing it onto a cooling stand. The operator is now free to load or unload another furnace. The crucible must remain in the cooling station for 10 hours before it can be further processed. During the cooling cycle, the silicon continues to cool as a solid ingot and the crucible shell fractures and falls away from the ingot. After the 10-hour cool-down period, a furnace operator returns and spends the next 15 minutes cleaning up the broken crucible pieces, then moves the 300 kg silicon ingot to the brick production area input buffer.

3. Silicon Brick Production

The next phase of the process will generate “bricks” of production-grade polycrystalline silicon that can be sawed into silicon wafers. This is achieved through a series of process steps: ingot preparation, ingot dicing, brick trimming, brick inspection, and sizing.

3.1 Ingot Preparation

An operator in the silicon brick production area will use a chain hoist to pick up the 300 kg production ingot from the input buffer. The ingot will then be glued into a saw dicing pallet designed to hold the ingot firmly in place while it is cut into 4” x 4” x 8” to 11” bricks. It takes 17 minutes for the operator to prepare the pallet, lift, and properly position the ingot. Once in place, the ingot must remain on the pallet for 16 hours to allow the glue to dry before it can be placed in the dicing saw. The ingots will wait in a FI/FO queue until the glue is cured.

3.2 Ingot Dicing

An operator will pull an ingot out of the queue and load it into the dicing saw. The operator will set the saw’s controls and initiate the cutting process, which takes 12 minutes.

The saw will then operate unattended for the next 8 hours. During this time, the saw, which is programmed to remove and square up the four sides of the ingot, will cut the ingot into a 12 x12 matrix of bricks while they are still glued to the pallet. An operator will take 5 minutes to remove the pallet containing the brick matrix from the saw and place it in a buffer in front of the brick trimming area.

3.3 Brick Trimming

An operator will pull a brick matrix pallet from the queue and take it to a trimming station, where he/she will use a simple piece of wire with two handles to saw through the glue that holds each brick to the pallet's surface, and will release the bricks one at a time. Once a brick is freed from the pallet, the operator will inspect it to determine the amount of valuable polycrystalline silicon contained within. Impurities in the silicon tend to accumulate at the top and bottom of the production crucibles, forming a crust on each end of the bricks. The bricks situated in the center of the ingot have thinner crusts than those cut from the edges of the crucible. The variance between good quality and unacceptable material can easily be witnessed by the operator, who will use a diamond tipped circular saw to remove those impurities. It takes 13 minutes (per brick) to remove the bricks from the pallet and cut off each end. Once freed and trimmed, they are stacked onto a cart so that they can be conveyed to the brick inspection area. A single cart is designed to handle all bricks from one ingot. The carts are placed in an output buffer, where they will be collected by a brick inspector.

3.4 Brick Inspection and Sizing

An inspector will collect a cart of bricks from the trimming area and take it to an inspection station. The inspector will grade the brick's overall quality to predict the efficiencies of the cells produced from it. Inspectors also look for signs of damage created during the cutting and trimming processes, and for evidence of internal stress fractures that could be present in the brick due to the ingot cooling process. On average, 7% of bricks are rejected and sent back to the crucible preparation area, where they are recycled as A-grade silicon material (at this point the bricks weigh approximately 2 kg). Each brick that passes inspection is measured for length. Data for each individual brick is entered into an information system, as well as the ingot's overall yield data. The inspected bricks are again placed onto the cart, but this time there is no guarantee of getting 144 bricks. The cart is then placed in the input buffer of the wire saw area. We will assume the cart transfer time is short compared to the inspection and data entry time. The average time required to inspect and measure a brick is about 20 seconds. Inspectors are to be treated as their own labor class.

4. The Wire Saw Area

The next phase of the process involves cutting the bricks into stacks of wafers using a wire saw. Brick length data is used to assemble sets of 16 blocks, which are used to construct four matched block sets. Each block set contains four blocks that are matched by length across the block sets. All sixteen blocks will be cut simultaneously, therefore it is important for the total length of material in each of the four block sets to be as closely matched as possible. If the block sets are not balanced, there exists an increased probability that the cutting wire in the wire saw will break. Before the blocks can be cut, they must be assembled onto a saw mount assembly, consisting of a metal base, a glass substrate, and the four blocks. The glass is glued to the metal base and blocks are glued to the glass. The glass is a sacrificial material that ensures that the blocks are completely cut, while ensuring that the metal bases are not. The wire saw will cut all the way through the blocks and partially through the glass. For the purpose of this study, we will assume that on average, after being trimmed, the bricks are 7" long. That means that for each run of the wire saw, 112 inches of bricks are cut into 0.05"-thick wafers, simultaneously. During this process, 0.025 inch of material is wasted between each wafer due to the cutting action of the saw. According to my calculations, that means that 1,490 wafers are produced from each wire cutter cycle; each cycle of the wire cutter takes 7.4 hours. When the block sets are removed from the saw, they are placed into a solvent tank where hot solvents are used to loosen the glue that secures the wafers to the glass substrate. The block set must remain immersed in the tanks for a minimum of 6.5 hours.

4.1 Block Set Production

An operator from the wire saw area will retrieve a full cart of inspected blocks from the output buffer when the inventory in his/her block set preparation station drops below 16. The operator will unload all blocks from the cart and into the stations block buffer (could be less than 144 due to rejected bricks). The operator will then examine all available blocks and choose 16 that, when combined across four different block mounts, provide close to equal length on all four mounts. It takes approximately 3 minutes to locate 16 suitable bricks. The operator spends the next 45 minutes assembling the set of four block mounts. The glue on the blocks must be allowed to cure for a minimum of 3 hours before loading the block sets in the saw. This curing time takes place in an output buffer associated with each block set preparation station. The maximum buffer size is three block sets (containing 4 block mounts each).

4.2 Wire Saw Operation

When a wire saw is idle, an operator from the wire saw area will retrieve a mount set from one of the block set production station output buffers and load it on a wire saw, which takes 20 minutes. The wire saw then runs automatically for 7.4 hours. The machine is then unloaded by a wire saw operator within 10 minutes. The block sets are placed inside an empty solvent tank, where they must soak for a minimum of 6.5 hours.

The wire placed in the wire saw must be changed after every 200 hours of operation. If caught before the wire snaps, the wire change-over takes 4 hours.   However, the wire placed in the wire saw would break with a mean time to failure of 150 hours based on a negative exponential distribution.   If this occurs, it will take approximately 7 hours to get the wire saw back online. A wire saw area operator is trained to perform the wire change maintenance process, and can also handle a wire break machine failure.

5. Wafer Separation and Packing

In this part of the facility, wafer handlers are responsible for removing the wafers from the sliced block mounts. These individual wafers are cleaned and inspected. In this inspection process, 4% of all wafers are discovered to be cracked or damaged, and therefore get discarded. After 100 quality wafers have been cleaned, they are bundled together in a stack, inventoried, and placed into a storage box for later use. Monthly production is based on the number of storage boxes of full cells that can be shipped from the facility per month.

The wafer separation and packing area has a central buffer and potentially a number of wafer handling stations. The central buffer contains block mounts that have completed their soak times. If the buffer is empty, a wafer handler will attempt to locate a block set that has completed its soak time in one of the solvent tanks. If a block set is located the operator will empty the solvent tank and transport it to the central buffer. It takes 10 minutes to move a block set from a solvent tank to the central buffer. When wafer handlers have run out of wafers to inspect, clean and pack, they will remove one of the four block mounts that are placed in the central buffer each time it is filled from a solvent tank. The wafer handler will remove each wafer to clean and inspect them on the wafer mount. It takes a wafer handler 4 seconds to inspect each wafer. There is a 4% chance that the wafer will be defective and if so, it will be scrapped. If deemed acceptable, it will be set aside in a “wafer stack” until 100 satisfactory wafers have been accumulated, at which point the wafer handler will take 1 minute to package the stack and log it into inventory. This is the end of the simulated process.

Resource Type	Unit	Cost ($)
8 Labors	Operator	800K
Buffer Space for Ingot or Cart	1	50K
Furnace	1	600K
Wire Saw	1	400K
Soak Tank	1	50K
Dicing Saw	1	300k
Crucible Packing Station	1	50K
Crucible Bake Oven	1	300k
Paint Booth	1	200k
Ingot Crusher & Conveyor	1	300k
Silicon Chunk Sorting Station	1	50K
Brick Inspection Station	1	50K
Brick Trim Station	1	100k
Raw Silicon Cleaning System	1	400K
Raw Silicon Dryer	1	300k
1250 kg Hopper	1	100k
2000 kg Material Bin	1	150k
Wafer Inspection & Packing Station	1	50k
Block Set Production Station	1	100k
Glue Cure Time Buffer Capacity (mount)	1	10k
Ingot Prep - Glue Application	1	50k
Production Crucible Prep. Station	1	50k
Purification Crucible Prep. Station	1	50K

Table 0-1 Resources Cost for Silicon Wafer Production Facility

Appendix B

Use of the Decision Support System to Enhance Throughput For Real-world System

This section will represent the result obtained from the decision support system to enhance the production line. Four algorithms will be applied to the model to provide suggestions to determine what section of the facility that requires enhancement. The four algorithms are: percentage utilization, waiting queue, average active duration method, and critical indicator. In addition, cost justification analysis will be applied to each enhancement. To fully analyze the case study, a manual analysis will be conducted to ensure that the DSS is leading to the right decision.

The base model was loaded into the decision support system with a run time of 90 days. The warm-up time for the model was determined to be 28 days, which will ensure that the system reaches the steady state. The production forecast was calculated as 4,358 units. Figure 0-1 shows that the bottleneck is the furnace, since it is the only resource that is 100% utilized. The furnace also causes the upstream resources to be blocked and the downstream resources to starve. Therefore, in order to increase the throughput, an additional furnace must be added. The DSS applied the four algorithms and all methods pointed to the furnace as the bottleneck, Table 0-1. Waiting queue considers the furnace as the bottleneck because it is the only resource that uses the buffer “Packed_Crucible_Buff“ as an input buffer. The percentage utilization of labor method pointed out that the most utilized labor in resource in the model is Brick_Prod_Labor. The DSS displays the most utilized labor resource, and it is up to the user to decide whether or not the labor resource is the bottleneck. In this scenario, Brick_Prod_Labor was busy 21.35% of the time, and hence it is not considered to be the bottleneck.

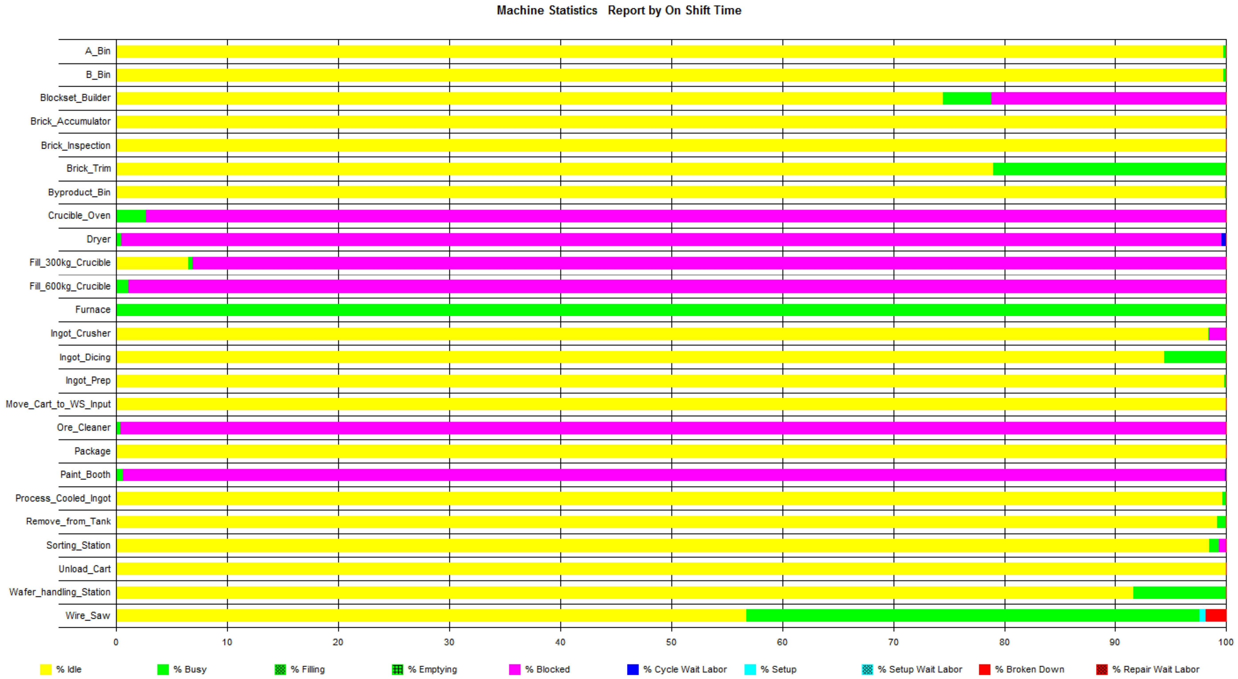


Figure 0-1 Machines Utilization – Base Model

Table 0-1 Bottleneck Detection Methods' Output

No.	Method	Bottleneck
1	Percentage Utilization (Machines)	Furnace
2	Percentage Utilization (Labor)	Brick_Prod_Labor
3	Waiting Queue	Packed_Crucible_Buff
4	Critical Indicator	Furnace
5	Average Active Duration	Furnace

1.1 First Enhancement:

Based on the output of the manual validation and the DSS output outlined previously, an additional furnace was added to the system. The enhancement increased the productivity to 8,940 packages. Figure 0-2 illustrates that the furnaces are highly utilized, which caused the upstream resources to be blocked and the downstream resources to starve. The DSS output, Table 0-1, shows that all the methods correctly identified the bottleneck as the furnaces. Again, Brick_Prod_Labor was utilized 40.41% of time and hence it is not considered to be the bottleneck. Therefore, another furnace needs to be added to the system to increase the throughput.

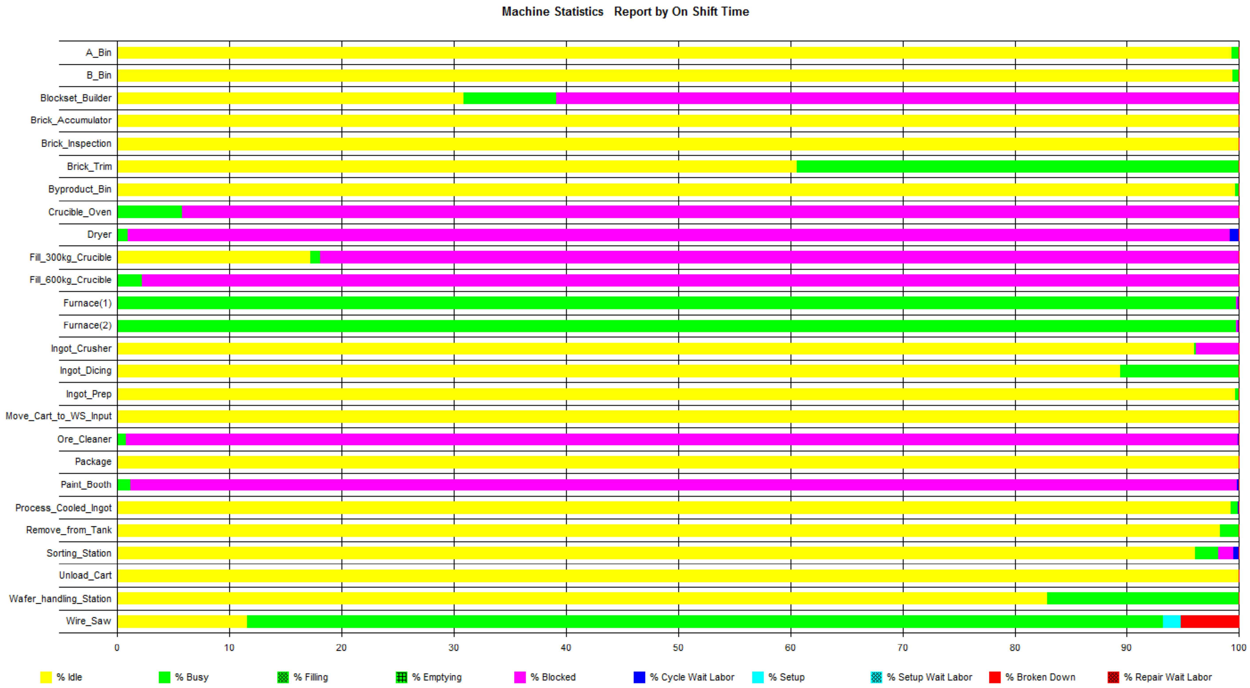


Figure 0-2 Machines Utilization - 1st Enhancement (2 Furnaces)

Table 0-2 Bottleneck Detection Methods' Output

No.	Method	Bottleneck
1	Percentage Utilization (Machines)	Furnace
2	Percentage Utilization (Labor)	Brick_Prod_Labor
3	Waiting Queue	Packed_Crucible_Buff
4	Critical Indicator	Furnace
5	Average Active Duration	Furnace

1.2 Second Enhancement:

Based on the output of the manual validation and DSS, a third furnace was added to the system. The enhancement increased productivity to 10,089 packages. Figure 0-3 shows that the bottleneck is shifting to Wire_Saw, which was highly utilized, with the remaining time allocated to either setup or downtime. As a result, the upstream’s resources were blocked and the downstream's resources starved. The DSS output, Table 0-3, indicates that all methods except critical indicator and waiting queue identified the correct bottleneck as the Wire_Saw. Critical indicator failed to identify the correct the bottleneck as there was a period of the time where Wire_Saw was inactive due to a breakdown. Critical indicator did not consider breakage time in the equation to determine the bottleneck, Equation 2-10. Waiting queue failed to identify the bottleneck because the size of the buffer Blockset_Cure is smaller than Packed_Crucible_Buff. For that reason, buffer Packed_Crucible_Buff can hold more parts, and the parts can wait inside the buffer for a longer period of time. However, in Blockset_Cure, the parts were rejected when Blockset_Cure reached its maximum capacity. Consequently, the waiting queue method returned a false result. At the end, in order to increase the throughput, another Wire_Saw would need to be added to the system.

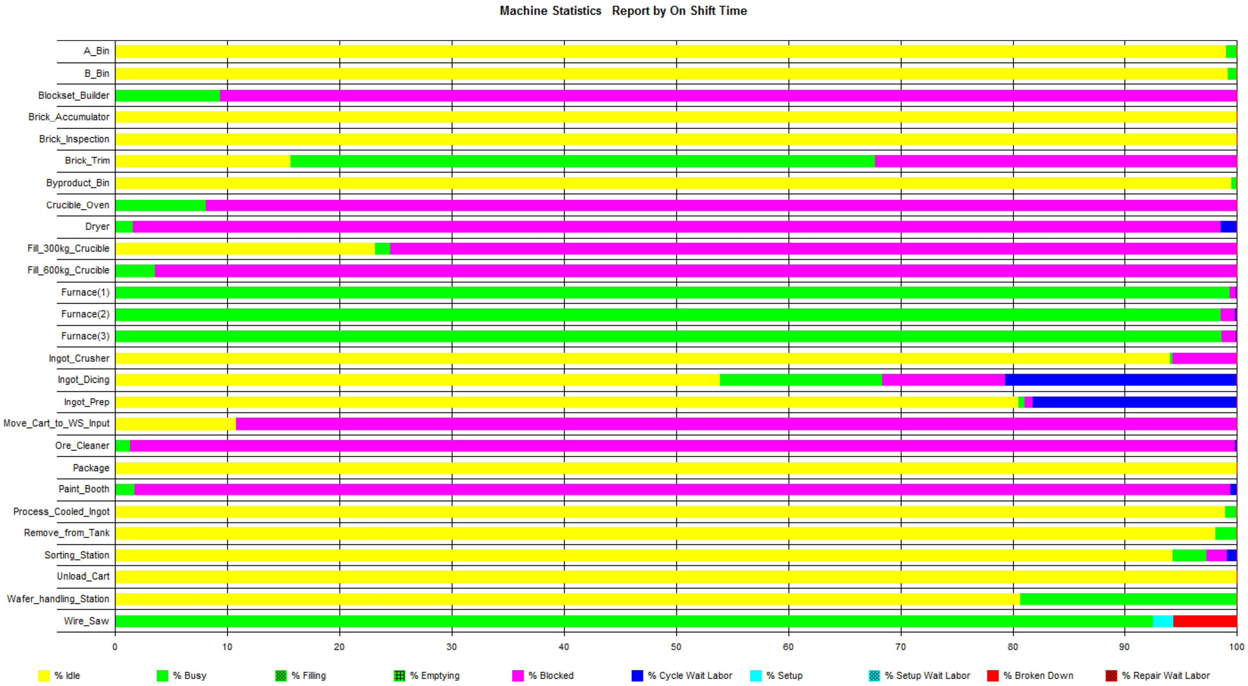


Figure 0-3 Machines Utilization - 2nd Enhancement (3 Furnaces)

Table 0-3 Bottleneck Detection Methods' Output

No.	Method	Bottleneck
1	Percentage Utilization (Machines)	Wire_Saw
2	Percentage Utilization (Labor)	Brick_Prod_Labor
3	Waiting Queue	Packed_Crucible_Buff
4	Critical Indicator	Furnace
5	Average Active Duration	Wire_Saw

1.3 Third Enhancement:

Based on the output of the manual validation and DSS, a new Wire_Saw was added to the system. The enhancement increased annual production to 12,986 packages. Figure 0-4 shows that the bottleneck shifted back to the furnace, since they were highly utilized and blocked for some time. Surprisingly, the upstream and the downtream resources were also blocked. Wire_Saw, for example, is a downstream resource that is blocked, whereas previously, it was idle. This implies that the buffers' capacity that holds the parts produced by the Wire_Saw needs to be increased. On the other hand, furnaces were quite busy most the time. These findings contradict some of the definitions of bottleneck discussed in Chapter 2. For those reasons, increasing furnace quantity will not improve the throughput. Instead, it will lead to a reduction in the production line. As a result, the true bottleneck is the capacity size of the buffer that waits after the Wire_Saw machine. Waiting queue indicated that the buffer with the largest average size and waiting time is Packed_Crucible_Buff, whereas our manual analysis shows that the buffer that requires enhancement is Solvent_Tank. The reason for this conflict is the fact that the automated algorithm assumed that all buffers in the system would be of equal size. Since the sizes are different, the algorithm selected the wrong buffer. To prove our point, the timed buffer “Solvent_Tank” was reconfigured in the model as a machine. Figure 0-4 shows that “Solvent_Tank_m”, which is the machine representation of the timed buffer, is about 97.85% utilized. Even though, “Solvent_Tank_m” is the second highest utilized resource in the model, the bottleneck detection methods still pointing to the furnace as the bottleneck. This is a challenge for this study as none of the investigated method were able to identify the true problem. Rather, the methods were leading towards a reduction in the production line.

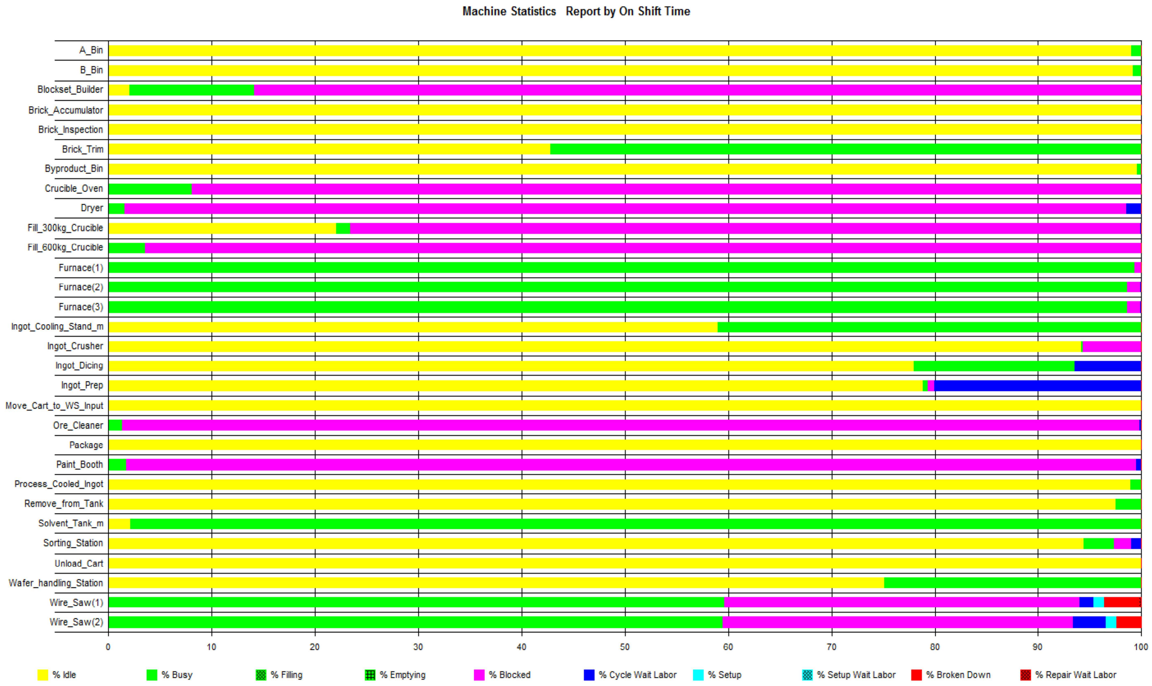


Figure 0-4 Machines Utilization - 3rd Enhancement (2 Wire_Saw)

Table 0-4 Bottleneck Detection Methods' Output

No.	Method	Bottleneck
1	Percentage Utilization (Machines)	Furnace
2	Percentage Utilization (Labor)	Brick_Prod_Labor
3	Waiting Queue	Packed_Crucible_Buff
4	Critical Indicator	Furnace
5	Average Active Duration	Furnace

1.4 Fourth Enhancement:

The result obtained from the DSS, Table 0-4, did not lead to the correct bottleneck. However, the quantity of Solvent_Tank was increased to two, which led to an annual increase in throughput to 13,392 packages. Moreover, this change allowed the system to behave normally. The upstream resource of the bottleneck is now blocked and the downstream resources are starved. Figure 0-5 and Table 0-5 both demonstrate that all bottleneck detection methods agree that the true bottleneck is the furnace.

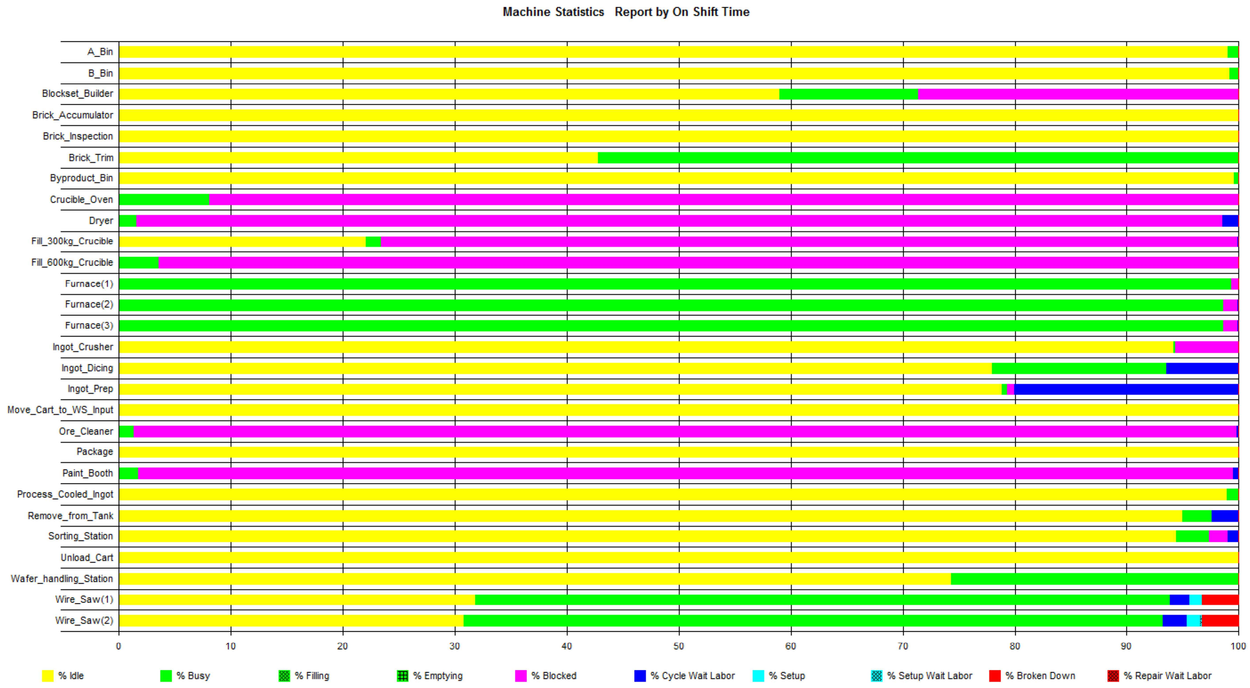


Figure 0-5 Machines Utilization - 4th Enhancement (2 Solvent_Tanks)

Table 0-5 Bottleneck Detection Methods' Output

No.	Method	Bottleneck
1	Percentage Utilization (Machines)	Furnace
2	Percentage Utilization (Labor)	Brick_Prod_Labor
3	Waiting Queue	Packed_Crucible_Buff
4	Critical Indicator	Furnace
5	Average Active Duration	Furnace

1.5 Fifth Enhancement:

The result obtained from the DSS, Table 0-5, led to the correct bottleneck. The quantity of furnaces was increased to four, which increased throughput to 17,450 packages per year.

Figure 7-7 shows that the bottleneck is yet again the furnaces. Table 0-6 demonstrates that all bottleneck detection methods agree that the true bottleneck is the furnace. It should be noted that there is a significant “wait on labor” time on the downstream resource of the furnaces, which is an indication that the labor resource “Brick_Prod_Labor” might become a problem when the furnaces start to process more parts.

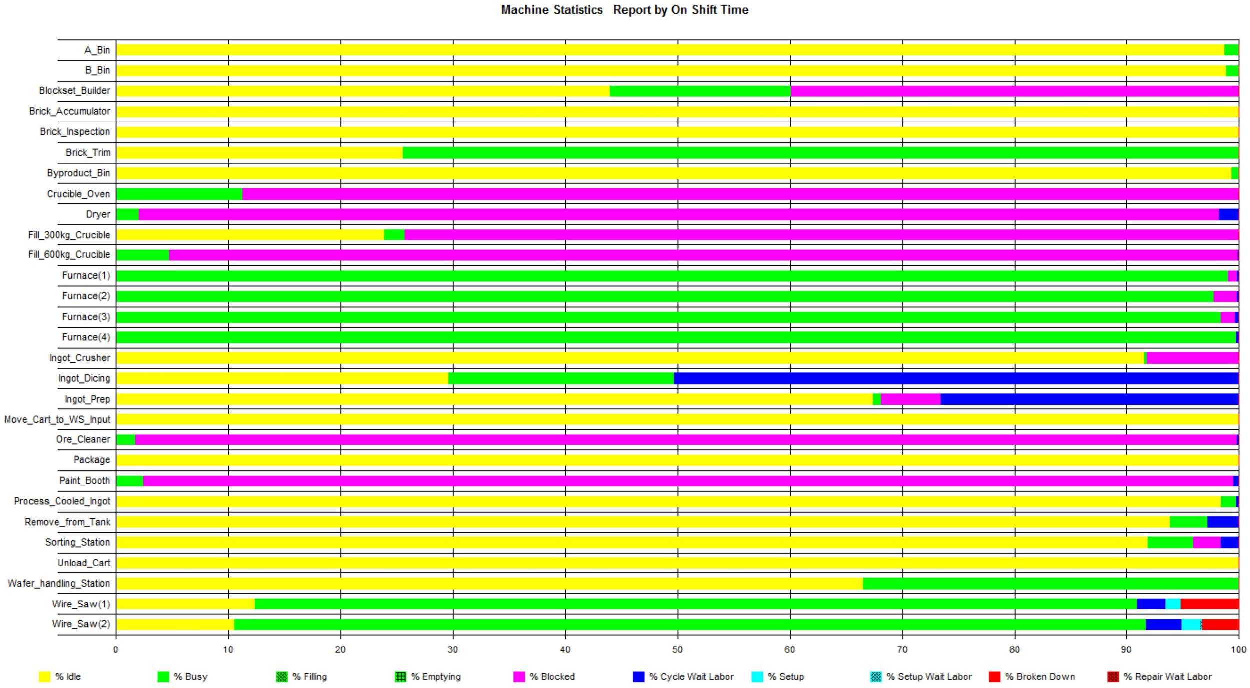


Figure 0-6 Machines Utilization - 5th Enhancement (4 Furnaces)

Table 0-6 Bottleneck Detection Methods' Output

No.	Method	Bottleneck
1	Percentage Utilization (Machines)	Furnace
2	Percentage Utilization (Labor)	Brick_Prod_Labor
3	Waiting Queue	Packed_Crucible_Buff
4	Critical Indicator	Furnace
5	Average Active Duration	Furnace

1.6 Sixth Enhancement:

The result obtained from the DSS, Table 7-7, led to the correct bottleneck. The quantity of furnaces was increased to five, a change that augmented throughput to 18,433 packages per year. Following that change, as shown in Figure 0-7, the downstream resources from the furnace are suffering from ‘waiting on labor’ state, “Brick_Prod_Labor”, as was anticipated in section 7.1.5. We believe labor “Brick_Prod_Labor” is the bottleneck at this stage and needs be increase to two. This was concluded as the other resources are blocked in the upstream resources. Such a change should release some of the blockage in the upstream resources. The only method that identified the bottleneck as “Brick_Prod_Labor” directly was the percentage utilization of labor. A few of the other methods identified the machines associated with Brick_Prod_Labor. It was mentioned in those methods that if the resource identified as the bottleneck requires a laborer, it would be useful to look closely to labor as a potential bottleneck. Ingot_Dicing is a resource that is associated with Brick_Prod_Labor. In addition, most of the bottleneck detection methods pointed to Ingot_Dicing as the bottleneck. Furthermore, the definition of bottleneck is also an important factor. Percentage utilization of machines, for example, considers “waiting on labor” as busy time. Hence, the method identified Ingot_Dicing as the bottleneck. Table 0-8, illustrates the methods and the identified bottlenecks. Critical indicator neither identified Brick_Prod_Labor nor Ingot_Dicing as the bottlenecks. The reason is that Ingot_Dicing spent more time waiting on labor rather than being busy. The equation for the critical indicator, Equation 2-10, subtracts the average waiting on labor for the entire system from the individual waiting on labor for each machine. Since Ingot_Dicing spent more time waiting on labor than the length of time that the furnaces were blocked, the critical indicator method returned a false result. Glue_Dry_Cycle is the input source for the Ingot_Dicing and hence the waiting queue method points to Ingot_Dicing as the bottleneck resource in the system.

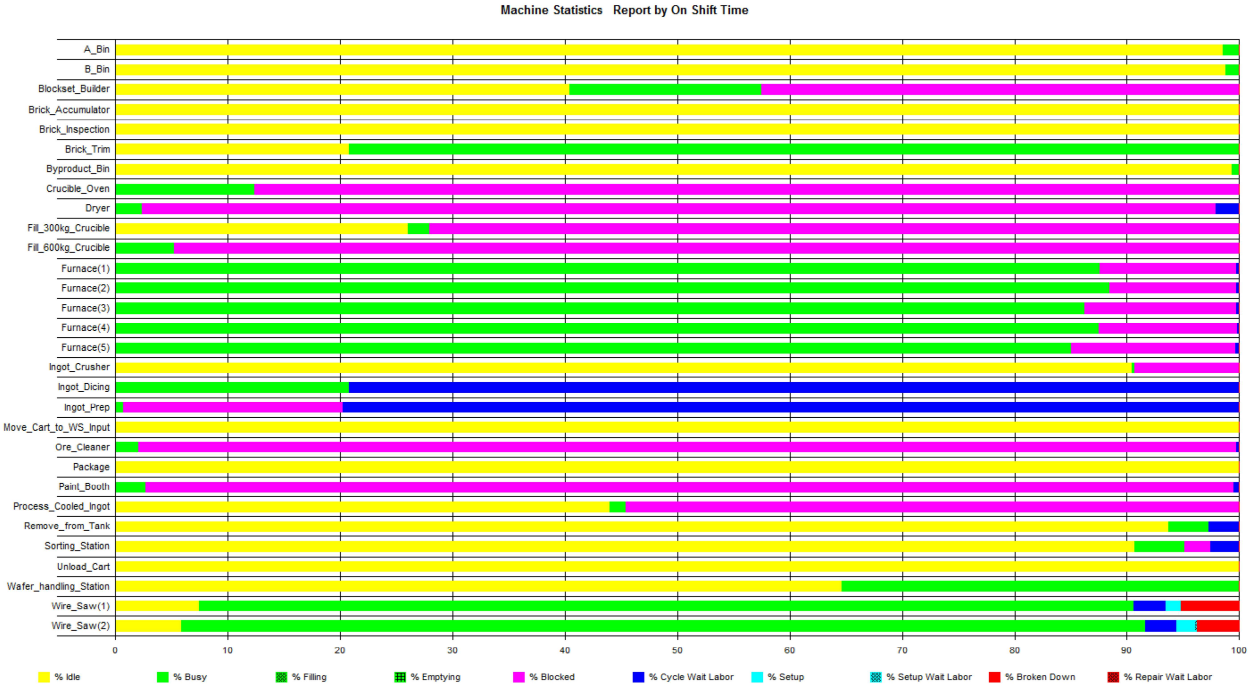


Figure 0-7 Machines Utilization - 6th Enhancement (5 Furnaces)

Table 0-7 Bottleneck Detection Methods' Output

No.	Method	Bottleneck
1	Percentage Utilization (Machines)	Ingot_Dicing
2	Percentage Utilization (Labor)	Brick_Prod_Labor
3	Waiting Queue	Glue_Dry_Cycle
4	Critical Indicator	Wire_Saw
5	Average Active Duration	Ingot_Dicing

1.7 Seventh Enhancement:

Adding a new labor resource “Brick_Prod_Labor” increased yearly throughput to 19,999 packages. Figure 0-8 shows that the Wire_Saw resources are highly utilized and cause blockage in the upstream resources and starvation in the downstream resources. Table 0-8 shows that most of the bottleneck detection methods point to Wire_Saw. The same reasons were explained in sections 7.1.2 and 7.1.3, which prevented waiting queue from detecting the correct bottleneck. Even though Brick_Prod_Labor was increased to two, percentage utilization of labor shows that Brick_Prod_Labor is the most utilized labor in the system.

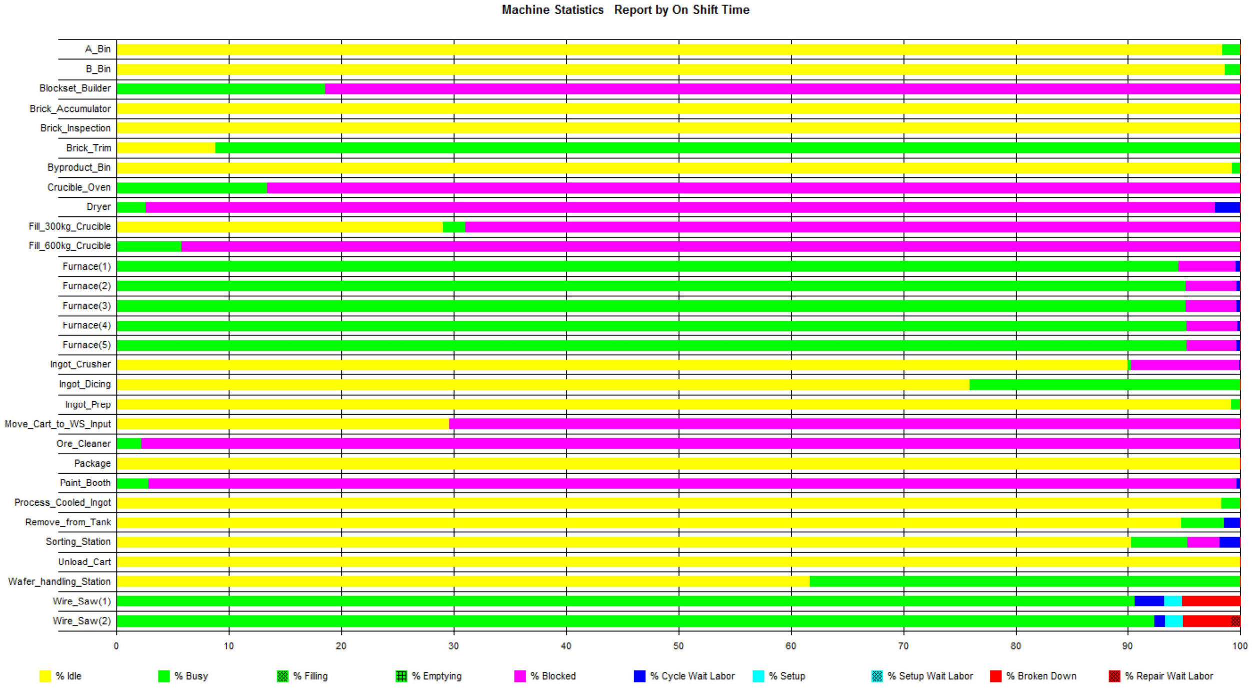


Figure 0-8 Machines Utilization - 7th Enhancement (2 Brick_Prod_Labor)

Table 0-8 Bottleneck Detection Methods' Output

No.	Method	Bottleneck
1	Percentage Utilization (Machines)	Wire_Saw
2	Percentage Utilization (Labor)	Brick_Prod_Labor
3	Waiting Queue	Packed_Crucible_Buff
4	Critical Indicator	Wire_Saw
5	Average Active Duration	Wire_Saw

1.8 Eighth Enhancement:

Adding a new Wire_Saw increased the throughput to 21,131 packages per year. Figure 0-9 shows that the bottleneck switched back to the furnaces, which are highly utilized and caused the upstream resources to become blocked and the downstream resources to starve. Table 0-9 shows that most of the bottleneck detection methods point to the furnaces. There is a slight blockage associated with the furnaces, which can be resolved by adding more capacity to the buffers beyond the furnaces. Since the blockage is only a small percentage, it will be ignored. It should also be noted that Wire_Saw is suffering from blockage as well. So why not increase the buffer size as was illustrated in section 7.1.3? The difference in this case is that Wire_Saw has been idle for some time. Blockage occurred because the downstream resources were not able to keep up with the flow due to “waiting on labor” constraint. In addition, the timed buffer “Solvent_Tank” was represented in the model as machine. Figure 0-9 shows that the “Solvent_Tank_m” is not highly utilized and hence cannot be the bottleneck. Those are indications that the buffer size is not the issue in this case.

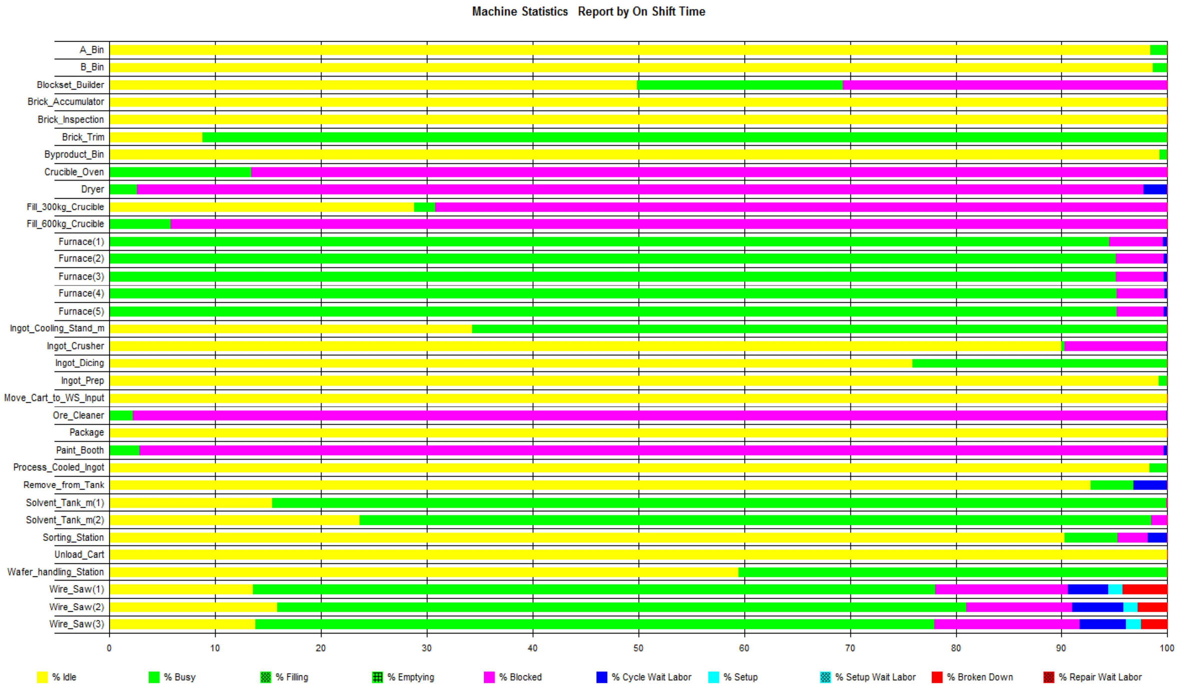


Figure 0-9 Machines Utilization - 8th Enhancement (3 Wire_Saw)

Table 0-9 Bottleneck Detection Methods' Output

No.	Method	Bottleneck
1	Percentage Utilization (Machines)	Furnace
2	Percentage Utilization (Labor)	Brick_Prod_Labor
3	Waiting Queue	Packed_Crucible_Buff
4	Critical Indicator	Furnace
5	Average Active Duration	Furnace

1.9 Ninth Enhancement:

Installing an extra furnace increased throughput to 23,132 packages per year. Figure 0-10 shows that the bottleneck is the Brick_Trim. It is 100% utilized and causes the upstream resources to be blocked and the downstream resources to starve. Table 0-10 illustrates that most of the bottleneck detection methods are pointing to the Brick_Trim as the bottleneck. Waiting queue shows that Brick_Trim_Buff and Glue_Dry_Cycle have the highest average waiting time in the system. Brick_Trim_Buff is the input source for Brick_Trim resource.

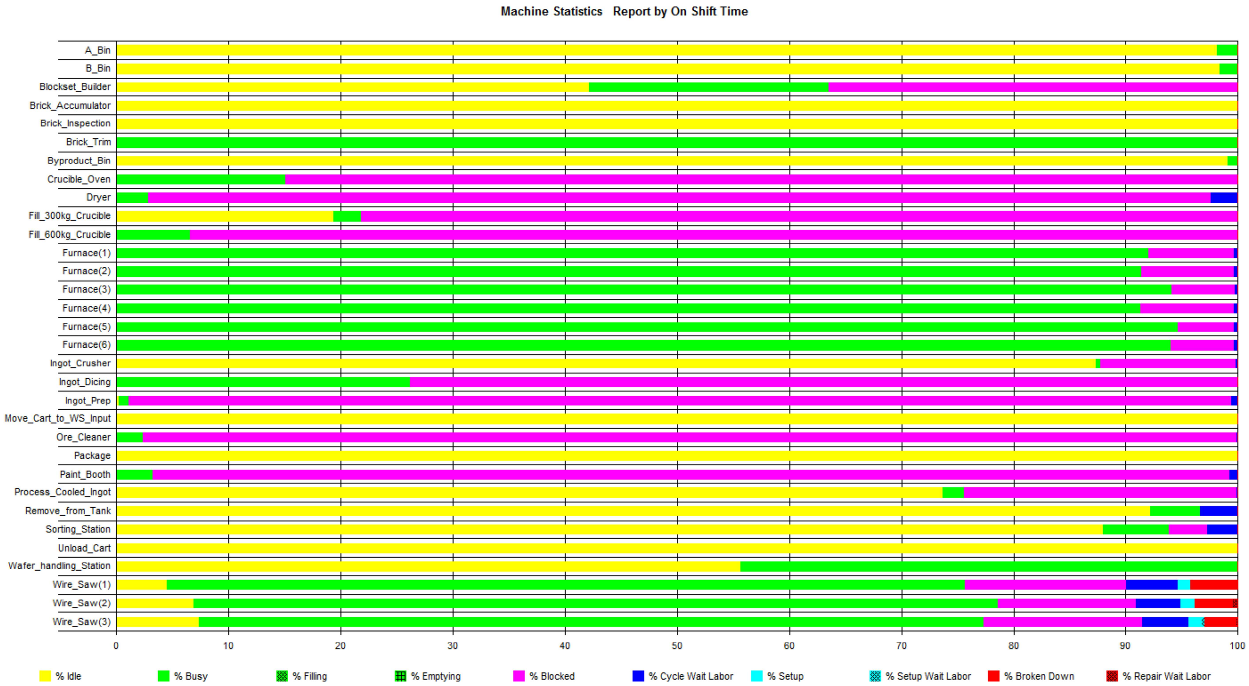


Figure 0-10 Machines Utilization - 9th Enhancement (6 Furnace)

Table 0-10 Bottleneck Detection Methods' Output

No.	Method	Bottleneck
1	Percentage Utilization (Machines)	Brick_Trim
2	Percentage Utilization (Labor)	Brick_Prod_Labor
3	Waiting Queue	Glue_Dry_Cycle and Brick_Trim_Buff
4	Critical Indicator	Brick_Trim
5	Average Active Duration	Brick_Trim

1.10 Tenth Enhancement:

Adding a new Brick_Trim increased the throughput to 24,520 packages annually. Figure 0-11 shows that the furnaces are highly utilized. However, Wire_Saw, which is in the downstream of the furnaces, is suffering from blockage. In addition, the Wire_Saw was not idle, indicating that the capacity of Solvent_Tank requires enlargement. It should be pointed out that adding a furnace instead of enlarging the Solvent_Tank will decrease the throughput. All bottleneck detection methods point to the furnaces as the bottleneck, hence leading to reduction to the production line. Even though the timed buffer “Solvent_Tank” was changed to a machine in the mode, none of the bottleneck detection method was able to identify it as the bottleneck.

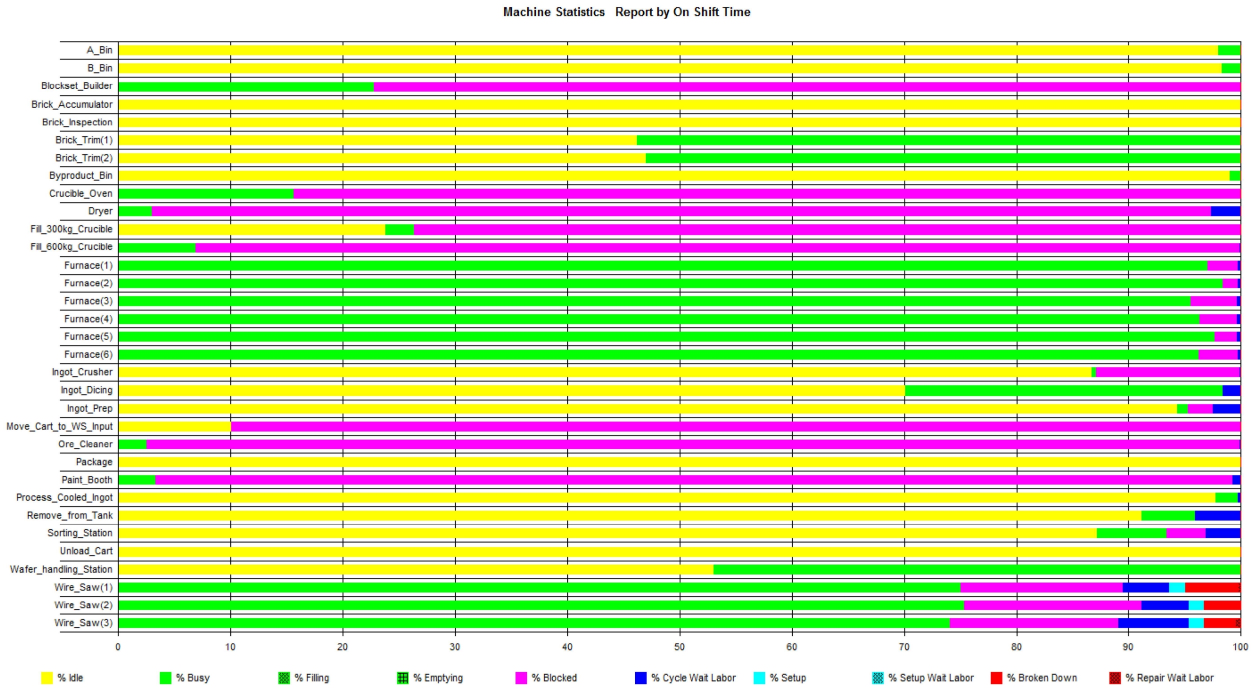


Figure 0-11 Machines Utilization - 10th Enhancement (2 Brick_Trim)

Table 0-11 Bottleneck Detection Methods' Output

No.	Method	Bottleneck
1	Percentage Utilization (Machines)	Furnace
2	Percentage Utilization (Labor)	Brick_Prod_Labor
3	Waiting Queue	Packed_Crucible_Buff
4	Critical Indicator	Furnace
5	Average Active Duration	Furnace

1.11 Eleventh Enhancement:

Increasing the capacity of Solvent_Tank increased throughput to 25,185 packages per year. year. Figure 0-12 shows that the furnaces are highly utilized. The model is behaving normally now, as the upstream resource of the furnaces are blocked and the downstream resources are starving.

Table 0-12 outlines the bottleneck detection methods and their recommendations. The manual analysis is in line with the DSS output,

Table 0-12.

Figure 0-12 Machines Utilization - 11th Enhancement (3 Solvent_Tanks)

Table 0-12 Bottleneck Detection Methods' Output

No.	Method	Bottleneck
1	Percentage Utilization (Machines)	Furnace
2	Percentage Utilization (Labor)	Brick_Prod_Labor
3	Waiting Queue	Packed_Crucible_Buff
4	Critical Indicator	Furnace
5	Average Active Duration	Furnace

1.12 Twelfth Enhancement:

Adding the 7th furnace to the system increased the throughput to 29,154 packages per year. Figure 0-13 shows that the Wire_Saw is the most utilized resource in the system. Table 0-13 also, shows that the majority of the detection methods are pointing to Wire_Saw. Surprisingly, increasing the Wire_Saw quantity will decrease production. Increasing the furnace quantity, on the other hand, will increase productivity. This is due to the fact that even though

Wire_Saw is the most active resource in the model, there are a slight period of time when the resource is not active, i.e., idle and blocked. The idle time indicates that there is a lack of parts flowing to the system. The blockage indicates that the resources downstream cannot keep up to the production of the Wire_Saw. It should be noted that these percentages are very small (less than 1%), hence, the bottleneck is not the Wire_Saw. The methods that were able to identify the true bottleneck were waiting queue and critical indicator. The percentage utilization of labor shows that the Wafer_Handler labor was utilized only 61.38% of the time. This labor resource is the reason that the Wire_Saw suffers from blockage.

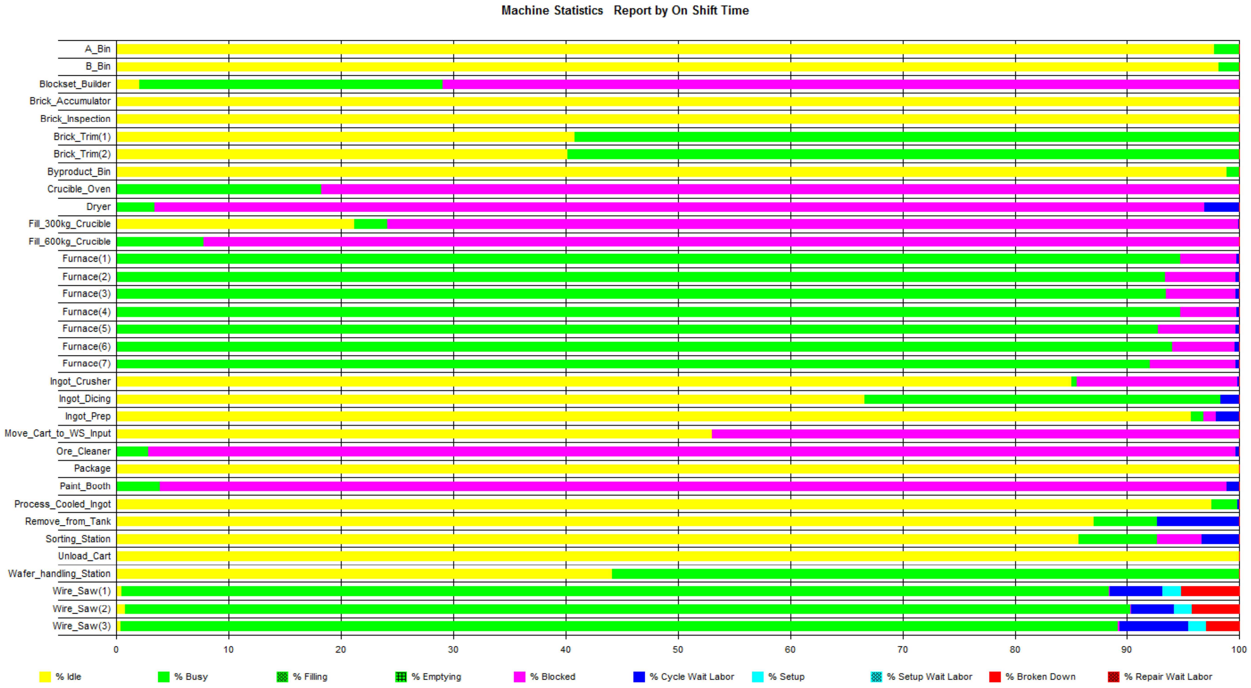


Figure 0-13 Machines Utilization - 12th Enhancement (7 Furnaces)

Table 0-13 Bottleneck Detection Methods' Output

No.	Method	Bottleneck
1	Percentage Utilization (Machines)	Wire_Saw
2	Percentage Utilization (Labor)	Wafer_Handler
3	Waiting Queue	Packed_Crucible_Buff
4	Critical Indicator	Furnace
5	Average Active Duration	Wire_Saw

1.13 Thirteenth Enhancement:

Adding an eighth furnace to the system increased the annual throughput to 29,309 packages. Figure 0-14 shows that the Wire_Saw is the most utilized resources in the system, and that the Wire_Saw resources were blocked for a very short period of time; however, there was no idle time. Table 0-14 shows that the majority of the detection methods point to Wire_Saw. Again, for the same reasons discussed in sections 7.1.2 and 7.1.3, the waiting queue method was unable to detect the correct bottleneck.

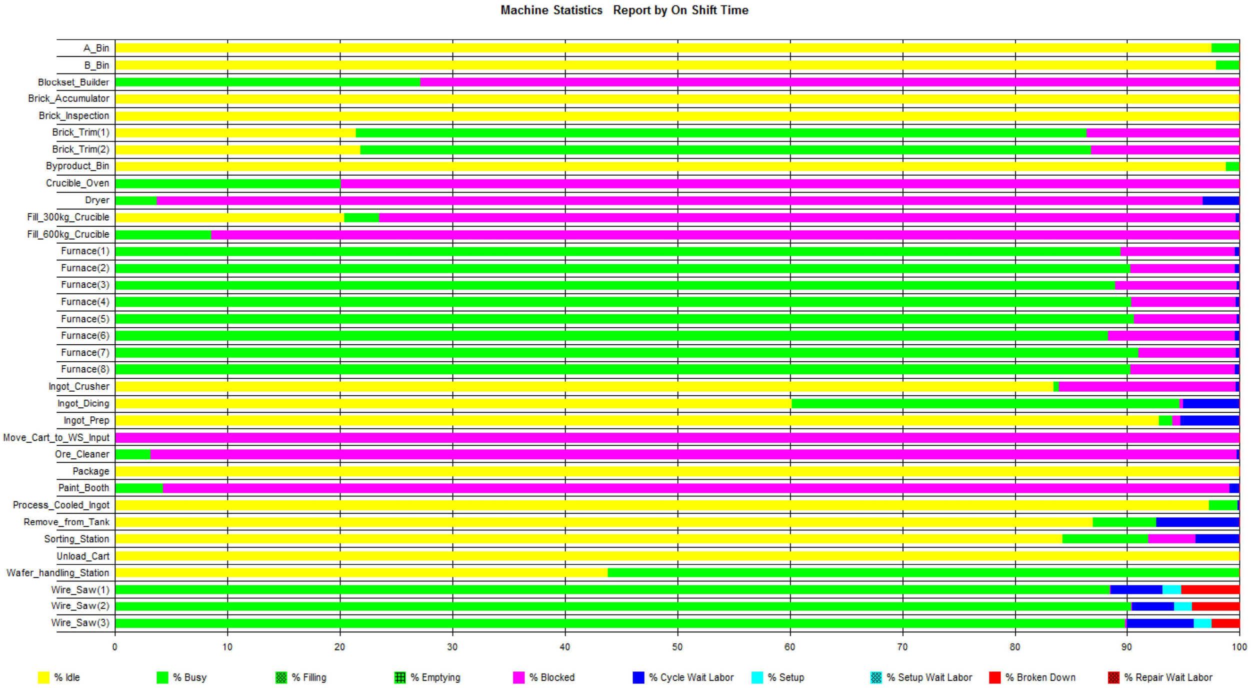


Figure 0-14 Machines Utilization - 13th Enhancement (8 Furnaces)

Table 0-14 Bottleneck Detection Methods' Output

No.	Method	Bottleneck
1	Percentage Utilization (Machines)	Wire_Saw
2	Percentage Utilization (Labor)	Brick_Prod_Labor
3	Waiting Queue	Packed_Crucible_Buff
4	Critical Indicator	Wire_Saw
5	Average Active Duration	Wire_Saw

1.14 Fourteenth Enhancement:

The addition of the fourth Wire_Saw to the system increased the throughput to 30,680 packages per year. This is the target demand on which this study has decided, and hence, no further investigation was conducted. Section 7.2 will show the cost justification of the enhancements.

References

Better, M., Glover, F., Kochenberger, G. & Wang, H., 2008. Simulation Optimization: Application In Risk Management. *International Journal of Information Technology & Decision Making*, pp.571-87.

Biller, S., Li, J. & Marin, S.P., 2008. Bottlenecks in Production Lines with Rework: A Systems Approach. *Proceedings of the 17th World Congress The International Federation of Automatic Control*, pp.14888-93.

Bukchin, J., 1998. A comparative study of performance measures for throughput of a mixed model assembly line in a JIT environment. *International Journal of Production Research*, 36(10), pp.2669-2685.

Chang, et al., 2007. Supervisory factory control based on real-time production. *ASME Transaction, Journal of Manufacturing Science and Engineering*, pp.653–60.

Chen, H.-C., Dai, L., Chen, C.-H. & Enver, Y., 1997. New development of Optimal Computing Budget Allocation for Discrete Event Simulation. *Winter Simulation Conference*, pp.334-41.

Chiang, S.-Y., Kuo, C.-T. & Meerkov, S.M., 2000. DT-Bottlenecks in Serial Production Lines: Theory and Application. *IEEE Transactions on Robotics and Automation*, pp.567-80.

Dan Trietsch, , 2005. Why a Critical Path by Any Other name Would Smell less Sweet? Towards a Holistic Approach to Pert/Cpm. *Project Management Institute*, 36 (1), pp.27-36.

D’Souza, R., 2004. *A Throughput-based Technique for Identiying Production System Bottleneck*. Mississippi: Masters Thesis, Mississippi State University.

Fu, M.C., 2002. Optimization for Simulation: Theory vs. Practice. *INFORMS Journal on Computing*, pp.192–215.

Goldratt, M., 1992. *The Goal: A Process of Ongoing Improvement*. North River Press.

Gross, D. & Thompson, J.M., 2008. *Fundamentals of Queueing Theory*. 4th ed. Wiley.

Hopp, & Spearman, , 2000. *Factory Physics*. McGraw-Hill/Irwin.

Karthikeyan, A.P., 2010. *Detection of Bottlenecks for Multiple Products and Mitigation Using Alternative Process Plans*. Wichita State University, College of Engineering, Department of Industrial and Manufacturing Engineering.

Kasemset, C. & Kachitvichyanukul, V., 2007. Simulation-Based Procedure for Bottleneck Identification. *AsiaSim*, pp.46–55.

Kasemset, C. & Kachitvichyanukul, V., 2008. Simulation-based Procedure for Implementing Theory of Constraints: Extension for Cases with Multiple Bottlenecks. *Asia Pasific Industrial Engineering & Management Systems Conference*, pp.1803-11.

Kelton, W.D., Sadowski, R.P. & Swets, N.B., 2010. *Simulation with Arena*. McGraw-Hill.

Kolker, A., 2010. Queuing Analytic Theory and Discrete Events Simulation for Healthcare: Right Application for the Right Problem. *Handbook of Research on Discrete Event Simulation*.

Kuo, C.T., Lim, J.T. & Meerkov, S.M., 1996. Bottlenecks in serial production lines: A system-theoretic approach. *Mathematical Problems in Engineering*, 2(3), pp.233-76.

Law, M. & Kelton, W., 1991. *Simulation Modeling and Analysis*. McGraw-Hill.

Law, A.M. & McComas, 2002. Simulation-based Optimization. *Winter Simulation Conference*, pp.41-44.

Lawrence, S.R. & Buss, A.H., 1994. ShiftingProductionBottlenecks:Causes, Cures, and Conundrums. *Production and Operations Management*, 3(1), pp.21-37.

Leporis, & Králová, , 2010. A Simulation Approach To Production Line Bottleneck Analysis. *International Conference CYBERNETICS AND INFORMATICS*, pp.1-10.

Li, , 2009. Bottleneck Detection of Complex Manufacturing Systems Using a Data-driven Method. *International Journal of Production Research*, pp.6929–40.

Li, L., Chang, Q. & Ni, J., 2009. Data driven bottleneck detection of manufacturing systems. *International Journal of Production Research*, pp.5019–36.

Mabin, J. & Balderstone, S.J., 2000. *The World of the Theory of Constraints*. Boca Raton , FL, USA: CRC Press.

Michel, R.G., 2001. Net Present Value Analysis: A Primer for Finance Officers. *GOVERNMENT FINANCE REVIEW*.

Miller, B., 2000. Applying TOC in the real world. *IIE Solutions*, 32 (5), pp.49-53.

Rahman, S.-u., 1998. Theory of Constraints A Review of The Philosophy and Its Applications. *International Journal of Operations & Production Management*, 18(4), pp.336-55.

Roser, , Nakano, & Tanaka, , 2001. A Practical Bottleneck Detection Method. *Proceedings of the 2001 Winter Simulation Conference*, pp.949-53.

Roser, , Nakano, & Tanaka, , 2002. Shifting Bottleneck Detection. *Proceedings of the 2002 Winter Simulation Conference*, pp.1079-86.

Roser, , Nakano, & Tanaka, , 2003. Comparision of Bottleneck Detection Methodsfor AGV Systems. *Proceedings of the 2003 Winter Simulation Conference*, pp.1192-98.

Sale, M.L. & Inman, R.A., 2003. Survey based comparison of performance and change in performanc of firms using raditional manufacturing, JIT, and TOC. *IJPR* , 41 (4), pp.829-44.

Sengupta, S., Das, K. & VanTil, R.P., 2008. A New Method For Bottleneck Detection. *Proceedings of Winter Simulation Conference* , pp.1741-45.

Stevens, G.T., 1994. *The Economic Analysis of Capital Expenditure for Managers and Engineers*. Boston: Pearson Custom Publishing.

Svancara, J. & Kralova, Z., 2009. Simulation Based Performance Analysis of The Multi-product Manufacturing System. Štrbské Pleso, Slovakia, 2009.

Taha, A., 2008. *Operations research: an introduction*. Pearson Education.

Tamilselvan , P., 2007. *BOTTLENECK SHIFTING IN PRODUCTION LINES IMPACT OF VARIABILITY, NEW DETECTION METHOD AND CONTROL STRATEGIES*. India : Anna University.

Tamilselvan, P., Krishnan, K. & Cheraghi, H., 2010. Measurement of Shifting Bottlenecks in a Non-Buffer Production System. *IIE Annual Conference. Proceedings*.

Wood, D.D., 2010. Hiring New Producer - A Capital Budgeting Decision. *CPCU eJournal*, pp.1-9.

Biographical Information

The author received his doctorate in Industrial Engineering from The University of Texas at Arlington. He received his M.S degree in Engineering Management from The University of Texas at Arlington and B.S. degree from King Fahd University of Petroleum and Minerals in Computer Engineering.

The author participated as a consultant for several projects in the areas of Telecommunication, Airlines industry, and manufacturing. He designed and implemented solutions for high availability, disaster recovery, and cloud environments. The author is currently interested in acquiring knowledge of simulation, financial engineering, operation research, and data analysis.