

**Effective Modeling of Throughput Time in  
Semiconductor Assembly Processes**

by

José A. Cervantes

BS Mechanical Engineering

Stanford University, Palo Alto, California (1995)

Submitted to the Departments of Mechanical Engineering and Operations

Research in partial fulfillment of the requirements for the degrees of

Master of Science in Operations Research

February 1998

and

Master of Science in Mechanical Engineering

June 1998

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

© José A. Cervantes

The author hereby grants to MIT permission to reproduce and distribute publicly paper and electronic copies of this thesis document in whole or in part, and to grant others the right to do so.

Author.....

Department of Mechanical Engineering

March 11, 1998

Certified by.....

Kevin Otto, Mechanical Engineering Thesis Advisor

Robert N. Noyce Career Development Assistant Professor

.....

Stanley B. Gersky, Operations Research Thesis Advisor

Senior Research Scientist

Accepted by.....

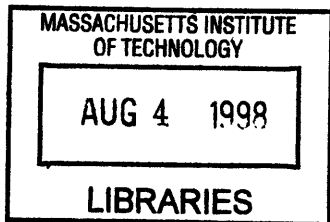
Ain A. Sonin

Chairman, Mechanical Engineering Graduate Committee

.....

Robert M. Freund

Co-Director, Operations Research Center



Eng

~~SECRET~~

# Effective Modeling of Throughput Time in Semiconductor Assembly Processes

by

José A. Cervantes

Submitted to the departments of Mechanical Engineering and Operations Research on March 11, 1998 in partial fulfillment of the requirements for the degrees of Master of Science in Mechanical Engineering and Master of Science in Operations Research at the Massachusetts Institute of Technology

## Abstract

We develop discrete event simulation (DES) models of a semiconductor assembly process capable of predicting average throughput time (TPT) within 6% of factory-floor measured average TPT. This represents a significant improvement over current DES models in the semiconductor industry, which commonly yield average TPTs that are 25% to 50% *less* than those measured on the factory floor. We accomplish this improvement by (1) constructing a separate area model of the system's constraint, (2) inputting an exact lot-starts schedule to the model, and (3) using failure parameters which are operation dependent rather than time dependent. In addition, we devise heuristics to parallelize the simulation process and significantly reduce computational expense in DES models, thereby affording the inclusion of detailed factors. We perform various sensitivity studies on the assembly's constraining process which indicate that operators and shift effects *do not* have a significant effect on average TPT. We also apply the series-parallel flow-line analytical model developed by Gershwin [8] and Burman [2], obtaining average TPTs which are 27% of factory-floor measured average TPT with numerical computational times of under a second.

Thesis Supervisor (Mechanical Engineering): Kevin N. Otto  
Title: Robert N. Noyce Career Development Assistant Professor

Thesis Supervisor (Operations Research): Stanley B. Gershwin  
Title: Senior Research Scientist



# Acknowledgments

This work was made possible by the fruitful contribution of Intel Corporation and MIT's Leaders for Manufacturing (LFM) group 6-7, *Integrated Analysis and Development Group*.

I would like to thank first, my thesis advisor Kevin Otto, for introducing me to the project early-on and guiding my research, and Sean Cunningham, who supervised my research at Intel, providing technical, moral, and emotional support. I am equally grateful to Court Hilton, the Intel liaison to LFM group 6-7, and Gene Meieran, for providing me with his group's support and allowing me to use the assembly floor as my laboratory, not to mention his mind-boggling riddles.

Expressed thanks are also due to my fellow group members at Intel: Dev Pillai for his genuine interest and indispensable guidance, Lance Solomon for his sense of humor and DES support, and Greg Mazenko, for his personal help and orientation to Intel.

I would also like to thank all of the industrial engineers who patiently answered my baffling questions about the assembly process, particularly Toby Norwood, Wells Fargo, Brad Whaley, Renee Lewis, and Don Hull. Special thanks to Letty Villegas for providing access to validation data.

Lastly, I would like to thank the following fellow students and faculty at MIT: Stan Gershwin, for his helpful guidance with analytical transfer line models and for co-advising my research, Asbjorn Bonvik for his invaluable help with DES models, and Mitchell Burman for allowing me to apply his series-parallel flow-line model.

This research was financed by Lucent Technology and AT&T Bell Labs' *Cooperative Research Fellowship Program*, Intel Corporation, and LFM group 6-7.

# Contents

<b>1.Introduction.....</b>	<b>17</b>
1.1 Background and Motivation .....	17
1.2 Thesis Objectives .....	21
1.3 Definition of Key Terms.....	22
1.3.1 Throughput Time (TPT) .....	22
1.3.2 Elements of Product Flow.....	25
1.3.3 Related Terminology, Nomenclature, and Conventions.....	27
1.4 Thesis Outline .....	32
<b>2.Problem Solving Methodology.....</b>	<b>33</b>
2.1 Possible Causes of TPT Error .....	34
2.1.1 Structural Errors.....	34
2.1.2 Data Errors .....	36
2.2 Mapping of Actual Factors to Modeled Factors .....	37
2.3 Mapping of Actual Data to DES Input Data.....	38

<b>3.Preliminary Studies: A Deterministic Processing Time Transfer Line .....</b>	<b>43</b>
3.1 Motivation of Study .....	43
3.2 System Description and Modeling Process.....	46
3.2.1 Experimental Vehicle.....	46
3.2.2 Initial Conditions and Simulation Procedure.....	48
3.2.3 Experimental Output And Response Variable Relationships .....	49
3.3 Simulation Results .....	50
3.3.1 Time For Convergence .....	55
3.3.2 Little’s Law Correlation.....	57
3.4 Conclusions of Study .....	59
<b>4. Case Study: A Microprocessor Assembly Series-Parallel Flow Line.....</b>	<b>61</b>
4.1 System Description.....	61
4.1.1 Brief Overview of Process Flow.....	63
4.1.2 Lot and Sub-lot Configurations .....	63
4.1.3 Resources and Logistics.....	64
4.2 DES Modeling .....	71
4.2.1 Background / Model Hierarchy.....	71
4.2.2 Model Abstractions and Interface Architecture .....	72
4.2.3 Description of Models.....	76
4.3 Model Validation .....	102
4.4 Conclusions.....	105
4.4.1 Results Summary .....	105



4.4.2 Recommendations for Future Action.....	105
<b>5.Analytical Approximations: Series-Parallel Flow Model.....</b>	<b>107</b>
5.1 Modeling Strategy.....	108
5.2 Two Machine Transfer Line .....	108
5.2.1 System Description .....	108
5.2.2 Model Assumptions .....	109
5.2.3 Analysis.....	110
5.2.4 Solution Technique.....	120
5.3 Decomposition Method for Multiple Server Serial Transfer Line.....	126
5.3.1 Overview of Decomposition Procedure.....	126
5.3.2 Decomposition Equations.....	128
5.3.3 Numerical Solution Methods .....	131
5.4 The Series-Parallel Model.....	132
5.4.1 Assumptions and Nomenclature .....	133
5.4.2 Solution Approach .....	134
5.4.3 Collapsed Transfer-line Parameters .....	134
5.5 Series-Parallel Model Applied to Case Study.....	137
5.5.1 Model Application .....	137
5.5.2 Input Parameters .....	139
5.5.3 Results and Discussion .....	140

<b>6. Conclusions</b> .....	<b>143</b>
6.1 Summary of Findings.....	143
6.2 Recommendations for Future Action.....	145
6.3 Concluding Remarks.....	146
<b>Appendix A: Calculations</b> .....	<b>147</b>
<b>Appendix B: Spreadsheet Models</b> .....	<b>161</b>
<b>Bibliography</b> .....	<b>155</b>

# List of Figures

Figure 1-1: Illustration of Gordon Moore’s Law for Intel microprocessors.....	18
Figure 1-2: Example of shrinking technology generations for Intel microprocessors.....	18
Figure 1-3: Product flow elements: units, lots and batches .....	25
Figure 1-4: k-machine transfer line.....	29
Figure 1-5: (a) A simple re-entrant flow line, (b) a series-parallel flow line,.....	30
Figure 2-6: TPT <i>fishbone</i> diagram .....	35
Figure 2-7: Problem Solving Methodology (a) Structural Errors .....	40
Figure 2-8: Problem Solving Methodology (b) Structure/Data Issues .....	41
Figure 2-9: Problem Solving Methodology (c) Data Issues .....	42
Figure 3-1: Average TPT scatter for various simulation lengths.....	50
Figure 3-2: Average TPT vs. number of cycles with 95% confidence interval bars .....	51
Figure 3-3: Estimates for mean and standard deviation of TPT vs. number of shifts. ....	52
Figure 3-4: Average buffer levels vs. number of cycles for buffers 1-5.....	54
Figure 3-5: Regression fit for confidence interval of average TPT vs. number of .....	56
Figure 3-6: Little's law TPT and simulated TPT vs. number of shifts.....	58

Figure 4-1: Chip manufacturing process.....	62
Figure 4-2: Assembly process flow .....	63
Figure 4-3: Downtime Classification Scheme .....	68
Figure 4-4: Model Hierarchy .....	72
Figure 4-5: Simulation Input Interface and Output Reports .....	73
Figure 4-6: Average TPT for a progressively detailed DES models .....	76
Figure 4-7: Step by step average TPT for <i>model 1</i> .....	80
Figure 4-8: Flow of data in decomposed model .....	81
Figure 4-9: Lot routing heuristic.....	82
Figure 4-10: Simulation run-time comparison for models 1 and 2.....	89
Figure 4-11: <i>Model 2</i> average TPT compared to <i>model 1</i> and actual .....	90
Figure 4-12: Typical shift schedule .....	92
Figure 4-13: Average Step TPTs for <i>model 3</i> .....	94
Figure 4-14: Operator sensitivity study results.....	95
Figure 4-15: Average TPT for MTTF and MTTR sensitivity study.....	98
Figure 4-16: Illustration of time-based versus use-based failure parameters .....	100
Figure 4-17: Step-by-step average TPTs for <i>model 3</i> , <i>model 4</i> and actual.....	101
Figure 4-18: Average TPT for <i>model 4</i> and arbitrary delays at stage 4.....	103
Figure 4-19: Average TPT for <i>model 4</i> and arbitrary delays for pre-constraint, constraint and post-constraint sections of model.....	104

Figure 5-1: Two-machine single-buffer transfer line.....	109
Figure 5-2: Markov process for the status of a single machine .....	112
Figure 5-3: Decomposition Method.....	127
Figure 5-4: “Collapsing” a series-parallel flow line into a single series flow line .....	133
Figure 5-5: Series-Parallel approximation for stage 5 process .....	138
Figure 5-6: Analytical model results compared to DES and factory-floor.....	140
Figure 5-7: Average buffer levels for analytical model (normalized) .....	141



## List of Tables

Table 4-1: Example of lot processing procedure for a stage 5 bay .....	67
Table 4-2: “Average processing rate” heuristic used for routing lots to stage 5 bays .....	85
Table 4-3: Illustration of “deterministic factory” heuristic.....	87
Table 4-4: Settings for MTTR and MTTF sensitivity studies .....	97





# Chapter 1

## Introduction

### 1.1 Background and Motivation

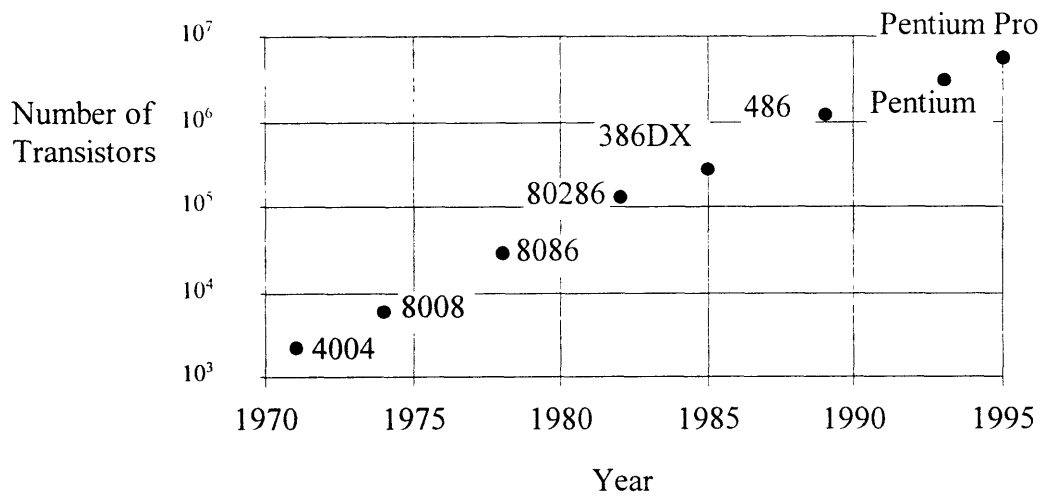
With the specter of the year 2000 looming on our doorstep, the increase in computing power is a notable exception to some of the ill-fated fantastic expectations of yesteryear. While winged cars have not become the predominant mode of transportation, few envisioned the ubiquity of the personal computer and the wealth of information available at our fingertips through the Internet. Indeed, few envisioned the prevalence of the *silicon* age over the *space* age.

The exponential increase in computer chip density, as predicted by *Gordon Moore's Law*<sup>1</sup>, (Figure 1-1) has been accompanied by equally dramatic shrinking technology

---

<sup>1</sup> *Gordon Moore's Law* states that transistor density doubles every eighteen months.

Figure 1-1: Illustration of Gordon Moore's Law for Intel microprocessors



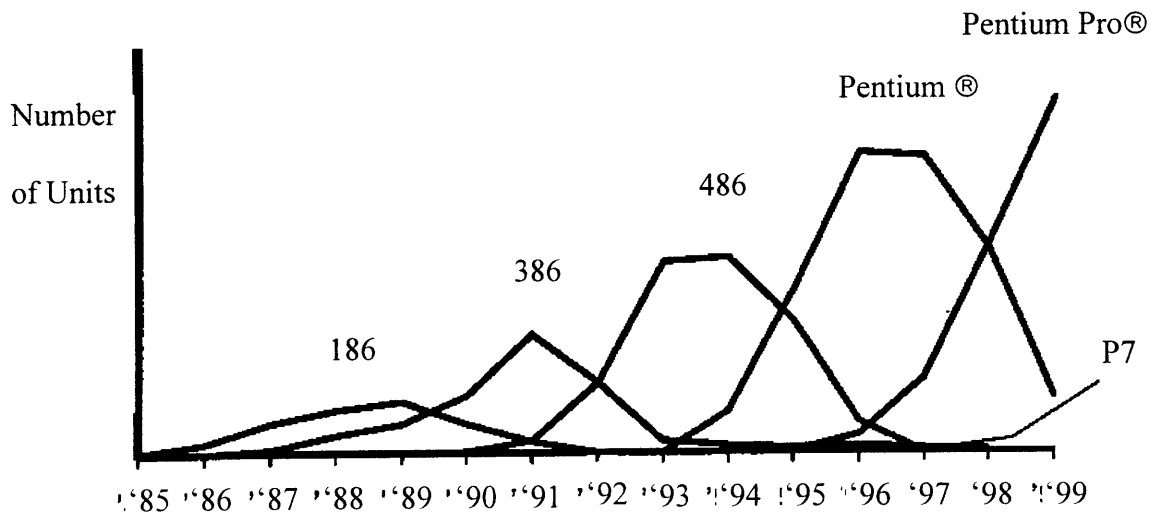
Source: *Byte*, December 1996.

generations (Figure 1-2).

With each new product generation, semiconductor manufacturers face three equally challenging tasks: (1) to design ever-faster chips, (2) to produce them in ever-greater numbers and (3) to execute (1) and (2) at a fraction of the time of the previous generation in order to keep up with customer demands and competitor technology (thus leading to shrinking technology generations). Clearly, time plays a critical role in the ability of semiconductor manufacturers to continue these trends.

In light of the challenges outlined above, there are several metrics that are used in gauging an organization's ability to meet customer demand and ramp-rate objectives. In particular, *throughput time* (TPT) is a standard metric of key importance which is used in most semiconductor manufacturing facilities. Roughly defined, TPT is the time that it takes for a lot of parts to get through a system. Such a system can include the sales and

Figure 1-2: Example of shrinking technology generations for Intel microprocessors



*Courtesy Eugene S. Meieran, Intel Corporation*

customer delivery teams, in which case the TPT is the time taken from the time a customer order is placed until the order is delivered. Thus, the average TPT is an indicator of how soon an order can be delivered to a customer. Likewise, the variance of TPT is an indicator of how accurately a customer's delivery date can be estimated. The critical importance of meeting delivery deadlines and maintaining robust delivery schedules provide strong justification for understanding the TPT distribution of manufacturing facilities.

There are various methods that have historically been used to model TPT. These can be divided into two main categories: analytical methods and discrete event simulation (DES). Analytical methods provide reasonable accuracy and have an advantage over DES models in terms of computational expense and ease of use. The disadvantage of

analytical methods is that they are only appropriate for a limited class of systems and are otherwise mathematically prohibitive. Although DES models are expensive both computationally and monetarily (relative to analytical models), they offer great promise since they can be—in theory—arbitrarily accurate. Moreover, they remain the only method of analysis for many systems.

In all of these models, there are data and structure assumptions which frequently lead to an underprediction of TPT. In particular, we have found that DES models in the semiconductor industry routinely yield TPTs that are 25% to 50% less than those measured on the factory floor.

The underprediction of TPT presents three major problems: (1) it decreases modelers' confidence that all important interactions affecting factory performance have been uncovered, (2) it undermines the credibility of models, which may otherwise be very useful in analyzing *relative* differences and predicting other metrics, and (3) since the ultimate goal of models is to serve as tools for improving process performance—e.g. reducing TPT variability and meeting ramp rate objectives—inaccurate TPT predictions are a limitation in making concrete recommendations and prescribing specific factory design improvements.

One obvious way to get simulated TPT to match the measured TPT is by incorporating arbitrary delays into the simulation to make the lots take longer to finish. This skewing of

the results, nevertheless, does not help to solve the problems listed above. It is therefore of interest to obtain accurate TPT predictions with assignable input factors.

## **1.2 Thesis Objectives**

In this thesis, we investigate the reasons why average TPT is frequently underpredicted. Our objectives are: (1) To identify the primary explanatory variables (i.e. factors which must be modeled) for accurate TPT predictions, (2) to identify factors which can typically be ignored (i.e. insignificant or non-explanatory factors) and (3) to outline inherent process limitations to the accuracy within which average TPT can be predicted. In addition, we devise analytical approximations which may be helpful in performing “what-if” analyses of transfer lines and provide cost-effective TPT modeling methods. It is our goal to outline modeling methods and procedures for creating DES models capable of predicting average TPT within 5%-10% of actual.

Concepts are tested via a case study: a chip assembly process for which we build DES and transfer-line analytical models, which are validated with factory floor data.

## 1.3 Definition of Key Terms

The terminology, nomenclature, and conventions presented in this section, although not universal, are of widespread use in industry and academia. They will be used throughout this document.

### 1.3.1 Throughput Time (TPT)

#### 1.3.1.1 Definition of TPT

The throughput time of a process step (*tpt*) is defined as the time elapsed from the instant a part or lot finishes processing at the previous operation. to the instant the part or lot finishes processing at the current operation or process step. This includes time the lot spends waiting in queue, transportation, machining, and any other operations which take place between the previous and current process steps. The throughput time of a process (TPT) is the sum of the step throughput times (*tpt*) over all the steps that constitute that process. The TPT for a lot in a process which consists of operation *a* through operation *b* is:

$$TPT_{a-b} = \sum_{i=a}^b tpt_i \quad (1.1)$$

Thus, any reference to TPT depends on the process or control volume being studied, including precise definitions of the processes *a* and *b*.<sup>2</sup>

---

<sup>2</sup>The lack of a well defined system can lead to discrepancies in TPT studies. In many simulations, for instance, TPT is calculated from the moment the first machine begins *operating* on a part. By contrast, many manufacturing lot-tracking systems calculate TPT from the moment parts enter a pre-process buffer where material is stored.

For most manufacturing systems, TPT has three main components:

(1) Transport time: the time it takes to transport a part or a lot from machine to machine (e.g. on a conveyor belt, AGV, or WIP cart). (2) Time spent in queues: the part may have to wait in a queue to get transported to a different machine or in a queue to get processed by a machine. This may be due to limited resources, including operators, tools to perform certain functions, machine failures, etc. (3) Time in process: The time taken for a part or lot to get processed by a particular machine and return to an output or transport queue, including load/unload times at processing locations

Out of these three factors, (2) is generally held attributable for most of the variability in TPT [12]. In many of the models used, particularly in semiconductor manufacturing, machine *processing times* (for definition see section 1.3.3.1) are assumed to be deterministic, since the variability in processing time is extremely small compared to the variability in queuing times and other factors [1].<sup>3</sup>

### **1.3.1.2 Distinction Between TPT and Average TPT**

There is an important distinction to be made between TPT and average TPT. These terms are sometimes used interchangeably, but throughout this thesis we will make an explicit distinction (when we refer to “TPT” alone we will assume it is the TPT for a single part or lot). The *average* TPT is calculated by taking the sample or time-series mean of the

---

<sup>3</sup> This queuing variability is largely due to large buffers used between stages in semiconductor assembly processes.

observed individual TPTs (i.e the sum of the observations divided by the number of observations). Thus the average TPT calculated for  $n$  lots across a defined system is:

$$\overline{TPT}_n = \frac{1}{n} \sum_{j=1}^n TPT_j \quad (1.2)$$

For  $n$  lots going through a system consisting of processes  $a$  through  $b$ , the average TPT is given by:

$$\overline{TPT}_n = \frac{1}{n} \sum_{j=1}^n TPT_j = \frac{1}{n} \sum_{j=1}^n \sum_{i=a}^b tpt_{ij} = \sum_{i=a}^b \frac{1}{n} \sum_{j=1}^n tpt_{ij} = \sum_{i=a}^b \overline{tpt}_m \quad (1.3)$$

That is, the average “factory” TPT is equal to the sum of the average process tpts (additive property of expected values of random variables). Form 1.3 is often useful for data calculation purposes in simulations and in validation data, since in many factories measurements are made on an operation basis rather than a lot basis.

### 1.3.1.3 TPT Variance

The variance of TPT is generally a much more complicated metric to estimate, and will only be treated in a cursory manner in this thesis. For analytical transfer line models, reasonably accurate estimates for the variance have only been developed for very simple transfer lines [4].

In our DES models, the TPT variance is calculated as the sample variance of the individual TPTs. For  $n$  parts that have gone through a system, their variance is:

$$s_n^2 = \frac{1}{n-1} \sum_{i=1}^n (\overline{TPT}_n - TPT_i)^2 \quad (1.4)$$



where  $\overline{TPT}_n$  is the sample mean for  $n$  parts, as defined in section 1.3.1.2.

Note that for both the mean and variance, these estimators are only meaningful if there exists a steady state (or near steady state) TPT distribution. This will be expounded upon in Chapter 3.

### 1.3.2 Elements of Product Flow

The following defines the elements by which product flows in a factory.

In general, there is a three level hierarchy in the flow of products: *units*, *lots*, and *batches*.

An illustration is shown in Figure 1-3.

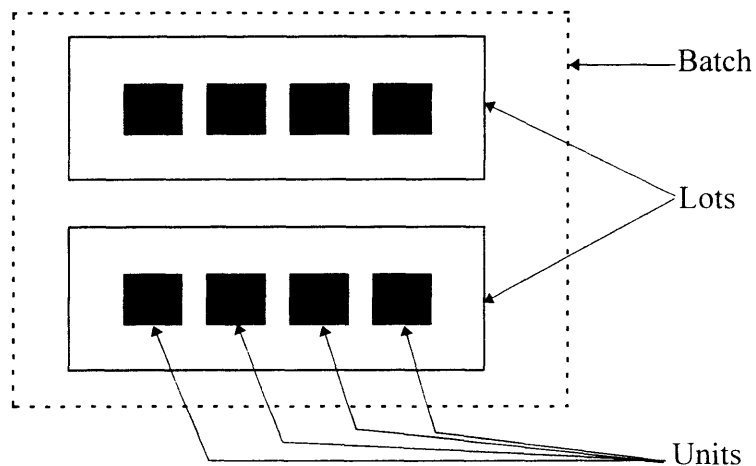


Figure 1-3: Product flow elements: units, lots and batches

*Units* are defined as the smallest entity traveling through a factory. In a semiconductor assembly process, it is a chip, since most machines operate one *chip* a time, and

furthermore, chips cannot be split into smaller elements. In a semiconductor FAB, the *unit* is a wafer, since most machines operate one wafer at a time and, within the FAB, wafers are not split.

*Lots* are defined as a group of *units* which generally travel together from process to process in a factory. Although most machines operate one unit at a time, units must wait for the remainder of their lot to finish processing at the current step before they can move on to the next process. Units are generally grouped into lots for tracking and logistic purposes.

A *batch* is defined as a group of parts or lots which are processed simultaneously on a *batching machine*. An example of a *batching machine* is an oven, where several lots can be processed simultaneously.

*Sub-lots* are created when lots are *split*. Lots are often split into sub-lots at batching machines, since batching machines may be unable to process an integral number of lots. In many assembly processes, these sub-lots are carried by *magazines*, which may be used for machine loading and unloading purposes. If *sub-lots* are not merged after a splitting operation, they become *lots*, since they now represent the elemental travel unit.

In a typical process, there is multiple splitting and batching of parts, which makes the above definitions context dependent. In the process of making a chip, for example, wafers travel through the FAB in *wafer lots*. At the assembly stage, wafer lots are split

into wafers, and further split into die. The die are then grouped into *boats* which are in turn grouped into *assembly lots*. This scheme is explained in detail in Chapter 4.

### **1.3.3 Related Terminology, Nomenclature, and Conventions**

#### **1.3.3.1 Factory Characterization, Machine States, and Flow Logistics**

The *throughput* of a manufacturing system is its production rate, that is, the number of parts that it produces per time unit.

*Work-In-Process*, or *WIP* is all of the material found in a manufacturing system, including material in machines, storage areas, transportation systems and inspection states. It is also referred to as *In-Process Inventory* or *Inventory*.

*WIP Management Strategy* is the policy used to allocate resources to WIP. This includes decisions on machine loading, batching, and lot prioritizations.

The terms *up* and *down* are used to describe the state of a machine. If the machine is *up*, it is working properly and operating (at least to a minimum threshold level). A machine is said to be *down*, if it has failed or is unavailable to perform work. Machine downtimes are further categorized into *scheduled* and *unscheduled* downtimes. An example of scheduled downtime is preventative maintenance. An example of an unscheduled downtime is a random failure that prevents the machine from operating. Note that in

most analyses of manufacturing systems, the state of the machine is discrete, in the sense that the machine cannot be *partially* down, or *partially* up.

A machine or process is said to be *blocked* if it is prevented from operating on a part due to downstream disruptions or if downstream inventory space is filled to capacity. Likewise, a machine or process is said to be *starved* if it does not have parts to work on due to upstream disruptions.

The *Processing Time* of a machine with respect to a part to be processed is the *ideal* time in which the machine can process the part. It does not include idle or set-up time, or other disruptions due to scheduled or unscheduled downtimes. Thus, the *effective* or *actual* processing time is always greater than the *ideal* processing time due to starvation, blockage, downtime, imperfect yields, etc.

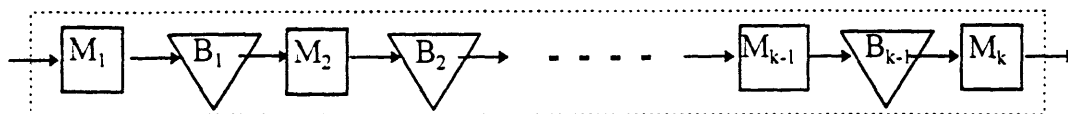
A *bottleneck* is defined by Goldratt [9] as any process which has a production rate that is less than the demand rate for the part being processed. It is also used to describe a system's *constraining* process, that is, the process step with the lowest throughput rate. An example of a bottleneck is a busy highway intersection during rush-hour traffic.

### 1.3.3.2 Representation of Manufacturing Systems

A *transfer line* is a manufacturing system consisting of a linear network of service stations or machines, separated by storage areas. Figure 1-4 (page 29) illustrates a *k*-machine transfer line. Each machine is represented by a square, and each storage area,

also referred to as a *queue* or *buffer*, is represented by a triangle. Material flows from outside the system to  $M_1$ , then to  $B_1$ , then to  $M_2$ ,  $B_2$  and so forth until it reaches  $M_k$ , after which it leaves. Note that the process sequence could have started with a buffer,  $B_0$ , or ended at another buffer  $B_{k+1}$ , depending on the system for which TPT is being measured. Such distinctions are important since the TPT is system dependent. The TPT for the system with extra buffers at the beginning and end of the transfer line, for example, is greater than the TPT for the transfer line shown in Figure 1-4. The dotted rectangle around the transfer line represents the *control volume*, which defines the system being studied.

Figure 1-4: k-machine transfer line



A *reentrant flow line*, is one in which, unlike a transfer line, lots (or fractions thereof) visit each machine more than once. Illustrations for this and other flow lines are shown in Figure 1-5. Examples of reentrant processes are systems where lots are reworked (due to imperfect yields), or lithographic layering in wafer FABs.

A *series-parallel flow line* is one which contains multiple parallel servers (or machines) at each step. Series-parallel flow lines are very common in assembly processes which are *sequence dependent*, i.e. “downstream” functions cannot be performed before “upstream” functions. Typically, the machines at each stage perform a particular function, such as

taking a chip from a wafer and attaching it onto a package. Such systems are robust in the sense that if one or several machines are down, the other machines can still perform the process step and prevent downstream machines from being starved.

Finally, we have hybrids of the systems described above. An example is a *reentrant series-parallel flow line*, shown in Figure 1-5(c). Although such systems are very complicated analytically, they are commonplace in industry. Thus, for many systems, DES remains the only viable method for modeling and analysis.

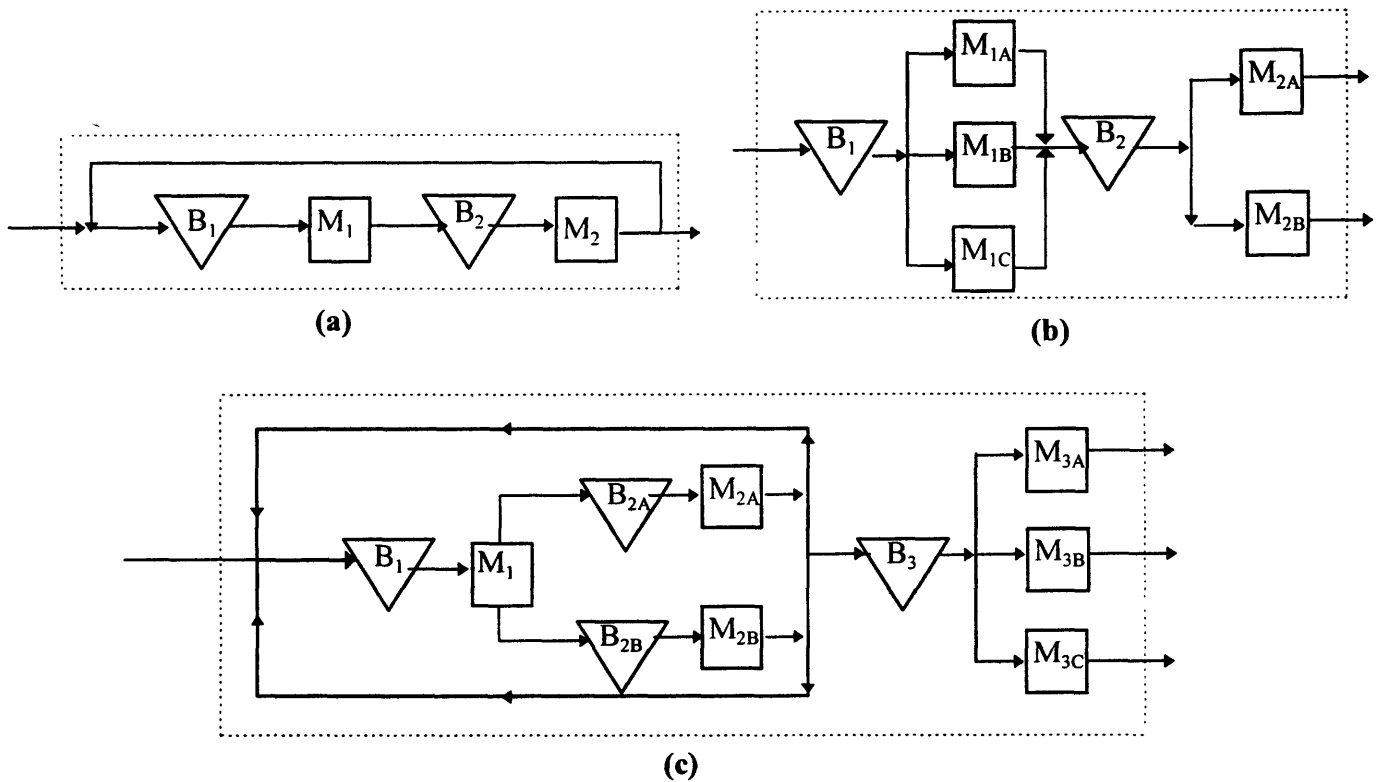


Figure 1-5: (a) A simple re-entrant flow line, (b) a series-parallel flow line, and (c) a series-parallel flow line with reentrancy

The banks of parallel machines in series-parallel flow lines, are known as *station families*, or *work centers*. As mentioned before, these are characterized by the function they perform and often contain identical (or near identical) machines.

## 1.4 Thesis Outline

In Chapter 2 we present a general problem solving methodology. An a-priori cause-and-effect *fishbone* diagram is presented. In addition, we devise a methodology which provides a possible framework for analyzing and correcting the discrepancy between modeled and factory-floor average TPT. This includes the mapping of data and structure information from the factory floor to the model.

In Chapter 3 we explore the convergence behavior of average TPT. A simplified deterministic processing time transfer line is used to study the effects of varying DES random seeds on average TPT. In addition, we study buffer level time-series behavior and agreement with *Little's Law*.

In Chapter 4 we present a real-world case study: a microprocessor assembly series-parallel flow line. Through a DES model, the effects of operators and shift schedules are assessed. The model is validated with factory floor data.

In Chapter 5, we present an analytical method for analyzing series-parallel flow lines. We use this model to obtain average TPT estimates for the case study presented in Chapter 4. Results are then compared with those from DES models.

In Chapters 6 we present a summary of conclusions and recommendations for future action.



## Chapter 2

# Problem Solving Methodology

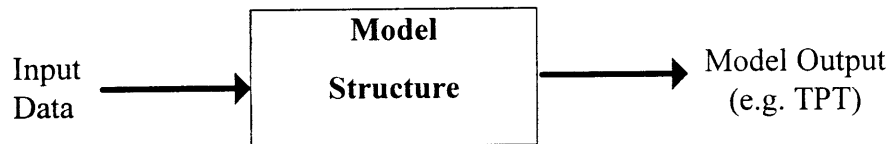
In this section, we discuss general approaches for finding the causes of TPT error. Moreover, we discuss an iterative methodology to correct possible causes of structural and data error in DES models. Although this methodology may seem overly simplistic, we found it a useful tool in focusing our area of research. In particular, due to the overwhelming number of factors that determine TPT (and therefore overwhelming number of possible *paths* in trying to correct TPT error), it is extremely useful to have a systematic error “troubleshooting” roadmap. This is presented only as a possible medium to pursue the problem, and not a guaranteed solution or exhaustive listing of possibilities.<sup>1</sup>

---

<sup>1</sup>There is a famous fable about a mouse, who is so troubled by an unrelenting cat, that he decides to go to a wiseman to try to come up with a solution to fend off the cat. After hours of careful thought, the wiseman comes back with his answer: “convert yourself into a dog.” The mouse replies to the wiseman “but how can I do this? You’ve pondered the problem at such length, only to offer me this seemingly infeasible alternative.” The wiseman replies “I come up with *solutions*. The *implementation* is up to you.”

## 2.1 Possible Causes of TPT Error

This analysis is based on the following breakdown of any model:



Thus, the errors in TPT can be attributed to errors in both input data and the model structure. Since there is a well defined separation between the data and structural portions of most DES models, this provides an effective breakdown for analyzing TPT-error. Figure 2-1(page 35) shows a *fishbone* diagram of the possible causes of TPT error. The two principal branches, consistent with the above diagram, are structural and data errors. Following are descriptions of the various branches and sub-branches.

### 2.1.1 Structural Errors

The structural errors are shown on the lower branch directly off the main TPT error branch. The two main sub-branches are *unmodeled factors* and *inaccurate modeling of factors*.

#### 2.1.1.1 Unmodeled Factors

These are items which have been left out of the model either explicitly or through implicit assumptions. Although all models--by definition--have factors which are omitted, it is important to ensure that such factors are not leading to large errors in TPT. Two

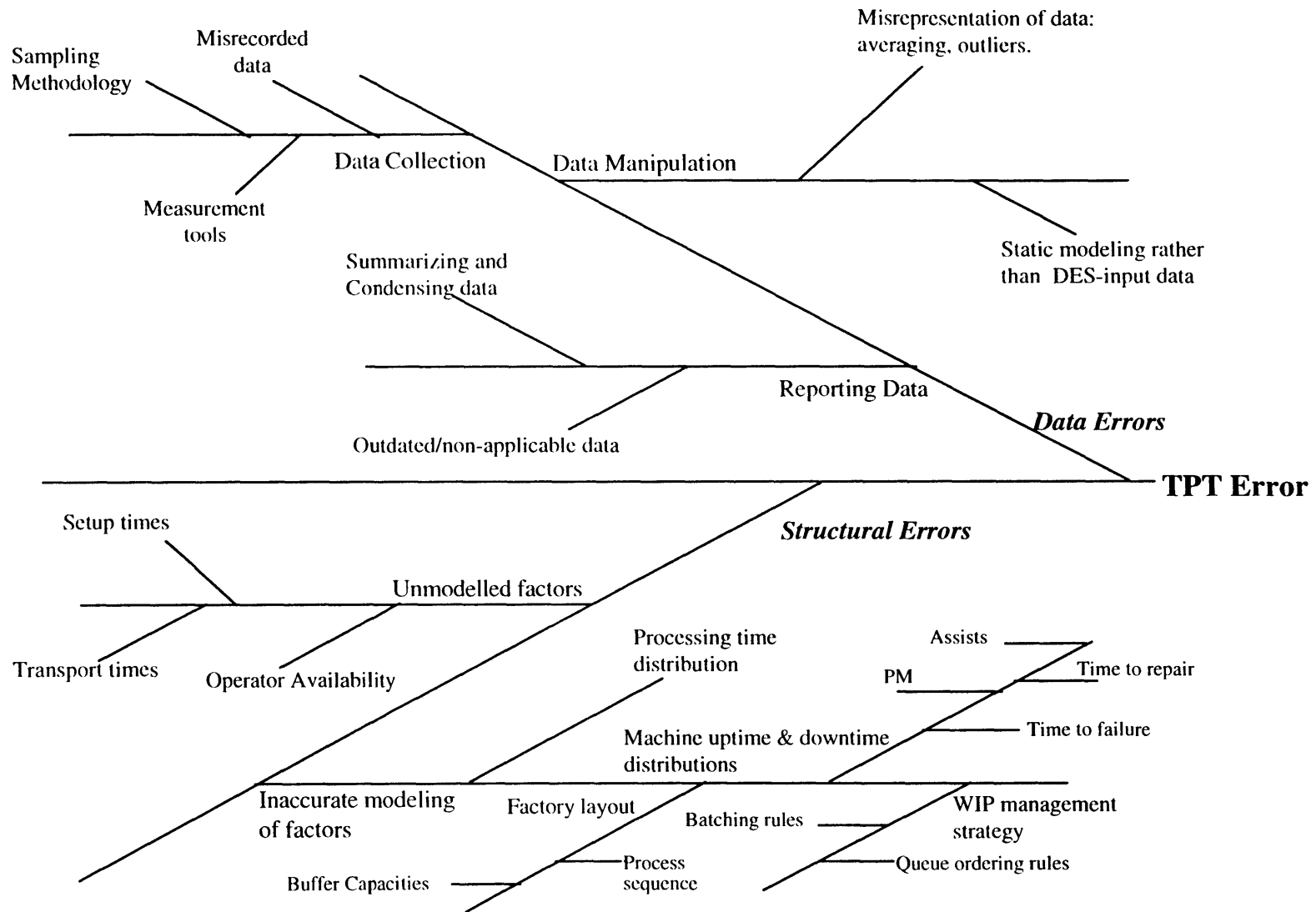


Figure 2-1: TPT fishbone diagram

examples are operator availability (which in many simulations is assumed 100% since operators are not modeled), and transport times, which are often not included in DES models.

#### **2.1.1.2 Inaccurate Modeling of Factors**

These are factors that have been modeled incorrectly due to faulty assumptions made in the modeling process. A crude example is factory layout, where items such as the machine counts, process sequence or buffer capacities are not modeled exactly. A common assumption is that all machines in a station family are assumed identical. A second example is the probability distribution of machine uptimes and downtimes. This is an area where assumptions must be made, since the actual probability distributions for machine uptimes and downtimes may be hard to describe, or, more commonly, are unknown. Moreover, when new equipment is installed in a factory, there is no available empirical performance data by which the machine can be modeled. A third example are WIP management strategies, where batching rules or other queue ordering rules may not be modeled accurately.

#### **2.1.2 Data Errors**

These are listed in the upper branch of the fishbone diagram. Modelers often obtain data from industrial engineers/factory floor personnel, rather than through independent data collection efforts. In many factories, this may be the only medium to obtain data, since data collection efforts for large factories can be prohibitively expensive. Unfortunately, the data that is collected on the factory floor is thus transformed several times before it

becomes input to models. We separate this transformation into three main processes: data collection, data manipulation (once it is collected) and the reporting of the data. Each of these processes contributes to the overall data induced TPT-error. The flow of data from the factory floor to the DES model is further described in section 2.3.

## 2.2 Mapping of Actual Factors to Modeled Factors

Figure 2-2 through 2-4 (starting on page 40) show a general problem solving methodology. This is separated into *structure* and *data* sections. In these figures, squares are "action" nodes and diamonds are "decision" nodes. Under structural issues (in step number *1s*) we list the arguments (factors) to the *actual* TPT function, and on a parallel track, we list the arguments to the *simulated* or *modeled* TPT function. In the modeling process, the actual factors ( $x_A$ ) are mapped to simulated factors ( $x_M$ ). Since, presumably, there are many more actual factors than modeled factors (indeed this is the purpose of a model), there is not a one-to-one correspondence between actual and modeled factors. Moreover, there will be actual factors which are unmodeled in the simulation, as illustrated in step *2s*. In the next step (*3s*) we assess the accuracy and assumptions made in the mapping of actual factors to modeled factors. For those factors that are unmodeled, we assess their impact on TPT, i.e. the sensitivity of TPT to these factors.

In step *4s* we take corrective actions. If this is a modeled factor which has been mismatched, we try to determine the appropriate mapping. This may be a simple case of, for instance, changing the machine downtime distribution from *Exponential* to *Gaussian*. It may also involve further investigation into lot-splitting, batching, or other WIP

management policies which have somehow been oversimplified or incorrectly modeled. If this is an unmodeled factor which is deemed significant, the factor is incorporated into the model. If the factor is unmodeled and deemed not significant (on a “vary one variable at a time” approach), we then determine whether there are interactions with other factors which may be significant. Again, if the interactions are significant, this factor is incorporated into the model; otherwise, it is appropriate to exclude this factor from the model since it is not explanatory. So far, we have discussed a methodology for correcting *structural* error sources; in the next section, we discuss a methodology for dealing with *data* error sources.

## 2.3 Mapping of Actual Data to DES Input Data

The steps under data error sources are labeled *1d* through *5d*.

The first step, *1d*, is to assess whether the data for the factor being examined has been recorded and processed accurately. In particular, we wish to examine the transformations made on this data element from the factory floor ( $v_{\text{actual}}$ ) until becomes model input ( $v_{\text{reported}}$ ).

If it is determined that there is an accurate mapping (and this factor is a source of error), we then look for other sources of error, and recheck the model *structure*. If it is determined that this factor has an inaccurate mapping, we then examine data collection and data processing procedures. We may refer back to the *fishbone* diagram (Figure 2-1) for possible sources of this mismap; examples include sampling methods, measurement

tools, and possible data manipulation. The decision steps for data collection and manipulation are shown in *3d* and *4d*. If indeed data collection practices are found to be unsound, we try to determine appropriate data collection methods; likewise if data has been oversimplified or misrepresented, we try to obtain “unprocessed” data and find appropriate input data (step *5d*).

The last step of the iteration is to incorporate the new data and structural changes into a new model, and return to beginning of the methodology for a new iteration (if deemed necessary).

Figure 2-2: Problem Solving Methodology (a) Structural Errors

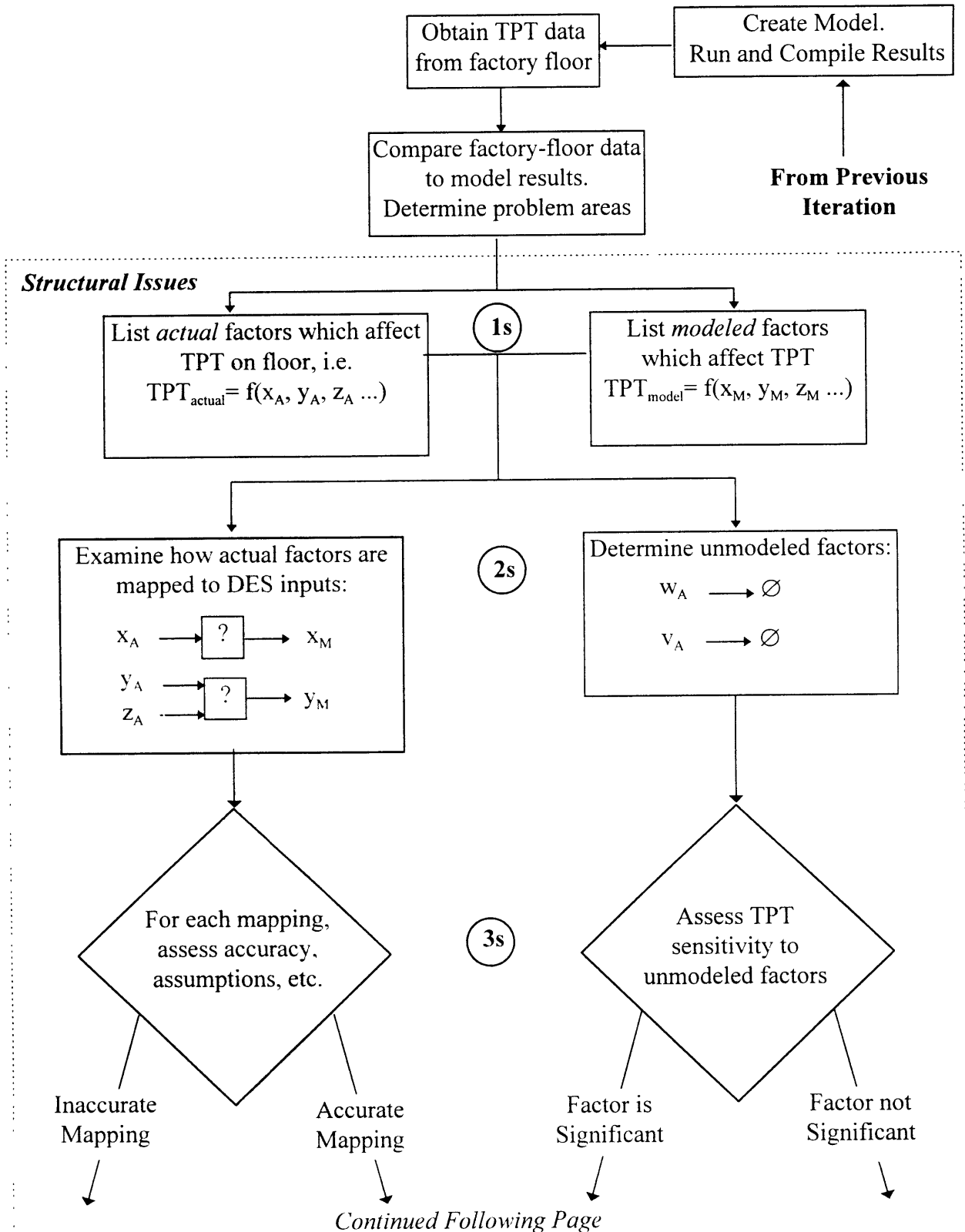
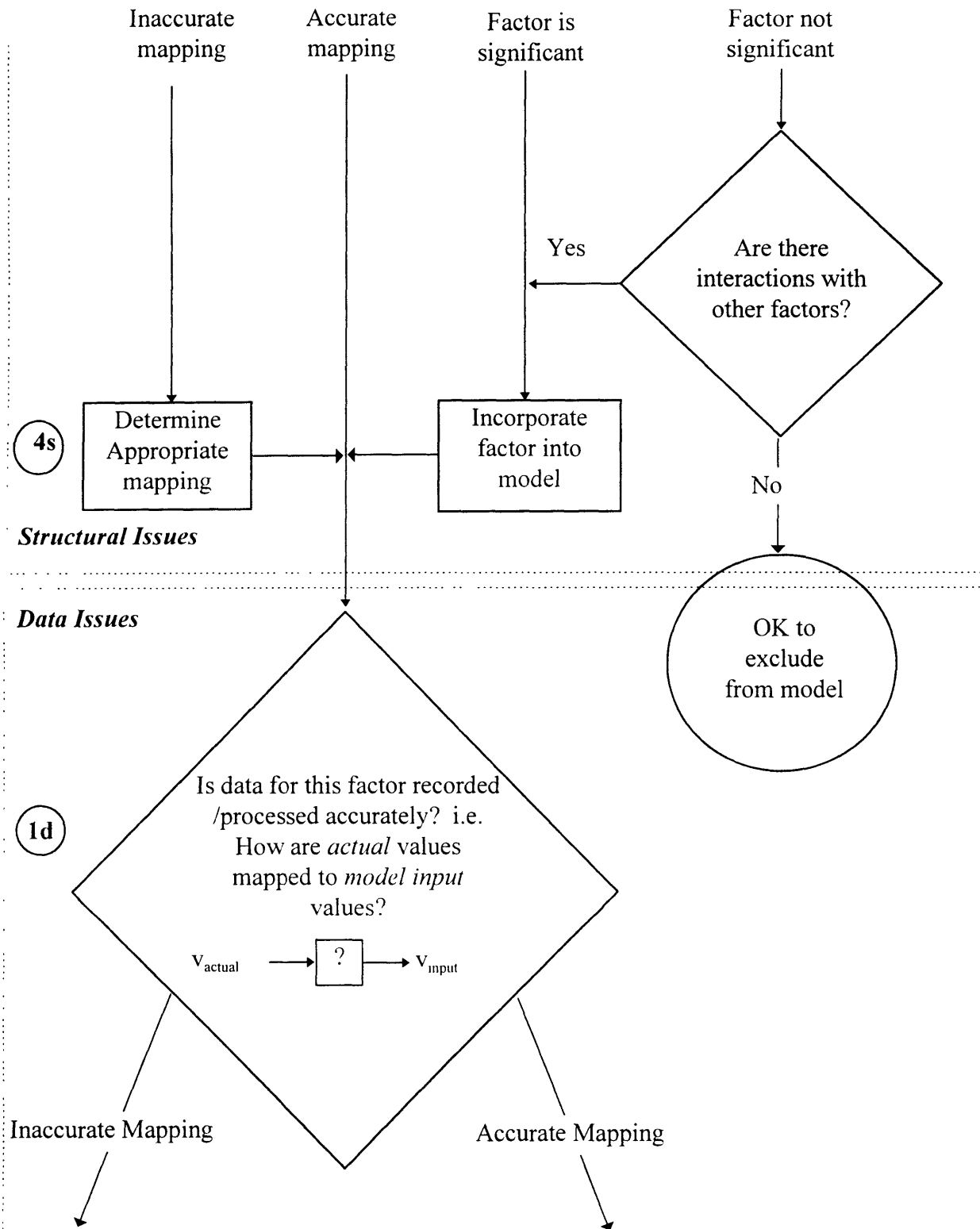


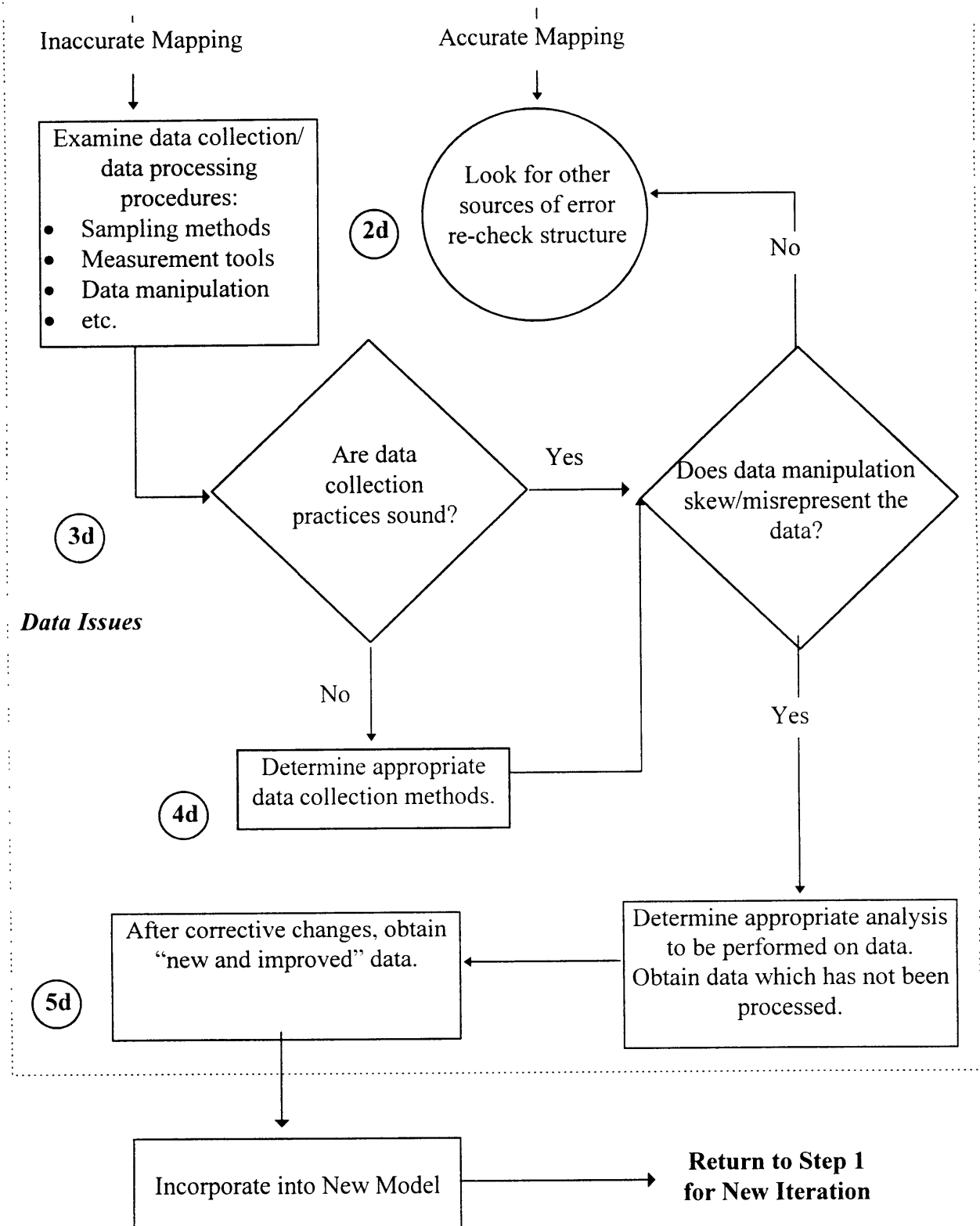


Figure 2-3: Problem Solving Methodology (b) Structure/Data Issues



Continued Following Page

Figure 2-4: Problem Solving Methodology (c) Data Issues



## Chapter 3

# Preliminary Studies: A Deterministic Processing Time Transfer Line

### 3.1 Motivation of Study

In order to answer questions about the predictability of average TPT, it is of interest to investigate its behavior with regard to any given initial conditions. In a DES, this might be a random seed that was used. On the factory floor, this might translate to slightly different lot-start schedules, WIP, or other management policies used among otherwise identical factories.

For any particular experiment (in this chapter we will refer to each DES run as an “experiment”) it can be shown that the time series data mean is a stochastically convergent measure [7]. Thus, the average TPT will always converge for *any given*

experiment as the number of cycles is increased. The average TPTs from experiment to experiment (given that experiments are *independent*), however, may or may not converge to the same value. This is true because two independent experiments (or DES runs) can yield average TPTs composed of random variables of altogether different probability distributions.

To illustrate the previous point, consider the position of a top on a surface as it is being spun. Let us assume a series of experiments are conducted in which we spin a top in two different surfaces, one being a "flat" table and the other a nicely curved bowl. For any given spin of the top on the flat table, it is certain to spend a large period of time spinning at a certain "steady state" spot before it topples over. If we spin this top several times over that same flat surface, however, we will find that *each trial* has a *different* "steady state" spot. If we conduct the same series of experiments in the bowl, however, we will find that the top will always finish spinning at the bottom of the bowl (assuming a well-defined bottom). The bowl corresponds to the case where the average TPT among different experiments, or top "spins," converges to the same value. The flat surface would correspond to a TPT which has "unstable" features or multiple steady-state modes..

Multiple references to "steady-state" and "transient" TPT are made. By steady state it is meant that the TPT distribution reaches a state which changes little from cycle to cycle (e.g. the small area or "spot" where the top settles to do most of its spinning). By transient, it is meant that the TPT distribution is still fluctuating wildly from shift to shift (the top is "wandering" about a relatively large area before stabilizing).

In sum, it is of interest to explore the convergence behavior of the TPT distribution. Based on this, we can infer on the viability of accurate simulation predictions of TPT. The discrepancy between simulated TPT and factory-floor measured TPT could be due the lack of a single steady state for average TPT (in both the simulations and the actually factories). It may also be the case that the transient TPT period (e.g. the period when the average TPT is still fluctuating widely) is more than the lifetime of a factory.

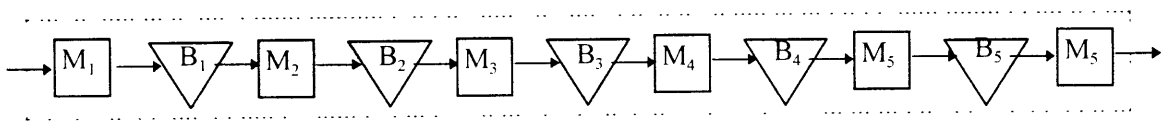
To return to the top example, both the factory and simulation must act like a top in a bowl in order for the factory-floor and simulated TPT to agree. If the actual factory floor TPT has multiple steady-state modes<sup>1</sup>, then it is very rare that the simulation will hit exactly *the* mode under which the factory is operating. If both the simulation and factory have multiple steady-state modes, the chances are even slimmer.

---

<sup>1</sup>Studies in the semiconductor industry [10] have found that relatively small changes in control policies, i.e. WIP management, lot release, etc. can have a significant effect on TPT. This may suggest that factories with identical set-ups but slightly different control policies would have different steady-state average TPTs.

## 3.2 System Description and Modeling Process

The system chosen for this study was a six machine transfer line with no reentrancy (i.e. loops). This model is a transfer line simplification of the microprocessor assembly process that will be presented in chapter 4. A diagram of the transfer line is shown in the following figure:



Thus the throughput time for this system is measured from the time a part enters  $M_1$  until it exits  $M_5$ ; there are no pre-process or post-process buffers.

### 3.2.1 Experimental Vehicle

The six step assembly and test process was modeled using a simple DES. The input to the DES were the mean time to failure (MTTF), mean time to repair (MTTR), and maximum buffer sizes. The time unit used for the simulation was the machine processing time, i.e. the time for a machine to complete an operation. This time was assumed deterministic and the same for all machines on the transfer line. The probability distributions of both failure and repair were assumed geometric with parameters  $1/\text{MTTF}$  and  $1/\text{MTTR}$ , respectively. The maximum buffer sizes were assumed to have a large limit of 1000, which is a good approximation of the assembly process. The machines were chosen to have identical MTTF and MTTR, since large variations from machine to

machine further complicate the behavior of TPT. The simulation operated on the assumption that the first machine is never starved (i.e. it has an infinite "reservoir" of parts), and that the last machine is never blocked (there is an infinite "sink" at the end of the transfer line or "infinite demand" for the part being processed). A final and important assumption is that all machines have only *operation dependent failures* (ODF), i.e. machines can only fail while working on a part.<sup>2</sup>

An additional input to the simulation was the number of shifts for which the simulation was being run and the length of each shift in terms of the machine processing time (we will also refer to each shift as one "cycle"). The length of each shift was chosen to be 480 machine processing times, which is equivalent to the number of minutes in eight hours (e.g.. if the processing time were one minute, this would correspond to eight-hour shifts, and all of the TPTs calculated by the simulation would be in time units of minutes). The shifts are assumed to run continuously for the length specified. The following table displays the settings for each of these parameters in the simulation.

TIME UNIT	MTTF*	MTTR*	Machine Processing Time*	Shift length	Buffer sizes
1 Machine processing time(mpt)	100mpt	100mpt	1mpt	480mpt	1000 parts

\* These parameters were equal for each of the six machines.

---

<sup>2</sup> This is an important assumption. For certain models, *time dependent failures* (TDF) are sometimes used. TDFs vary significantly from ODFs in that the former allow failures to occur while a machine is blocked, starved, or otherwise idle. The implications of ODFs and TDFs are further discussed in Chapter 4.

### 3.2.2 Initial Conditions and Simulation Procedure

#### 3.2.2.1 Factory Initial Condition

At the beginning of each simulation (time zero) WIP was set zero (e.g. all buffers and machines were empty) and all the machines were up.

#### 3.2.2.2 Simulation Procedure

The following number of shifts were chosen to run simulations:

Number of shifts	20	40	60	80	100	500	1K
# of simulation runs	30	30	30	30	30	30	30

Number of shifts	5K	10K	15K	50K	100K	500K
# of simulation runs	30	30	30	30	30	30

It is important to note that each simulation was started from the factory initial condition (at time zero), as the transient behavior of TPT was being studied. For example, thirty simulations of the factory were run from time=0 to time=20 shifts (9600*mpt*), then thirty simulations were run from time=0 to time=40 shifts (19,200 *mpt*), etc.

Since each simulation represents an independent experiment, 30 trials is a good number to begin making normality assumptions. Different random seeds were chosen for each simulation to ensure that runs were indeed independent. This also allowed for comparison of experiments at different simulation lengths, i.e. for a given random



number seed the simulation results are the same over any given number of cycles. For example, had the simulation run to 500 thousand cycles been stopped at shift 100, it would yield exactly the same results as the simulation ran to 100 with the same random number seed. Thus, not only do we obtain thirty data points for any of the given simulated periods, but we also have 30 independent *histories* of the throughput time over 500 thousand cycles.

### 3.2.3 Experimental Output And Response Variable Relationships

For each simulation run the following statistics were collected: sample mean and sample variance of TPT, average production rates and average queue levels (also referred to as buffer levels) for each of the machines. These figures were calculated for the entire period being simulated.

The relationship between TPT, queue levels and production rate *over long time periods* is given by Little's law:

$$L = \lambda W \quad (3-1)$$

where  $L$  is the work-in-process inventory, or the average number of parts in the system (this includes parts in queues and parts being processed),  $\lambda$  is the average production rate (throughput), or the rate at which finished parts are leaving the system, and  $W$  is the TPT.

Thus, we can see that the probability distribution for the throughput time is a function of, among other things, the levels of the queues in the system. Important insight can be gained by looking at the fluctuations of the queues. In particular, if the average buffer

levels evidence multiple steady states (i.e. the average buffer levels from simulation to simulation converge to different values), this may suggest that a single steady state does not exist for the TPT.

### 3.3 Simulation Results

Figure 3-1 through 3-3 show several "exploratory" graphs of TPT. In all of these the y-axis is the TPT and the x-axis shows the number of shifts on a *logarithmic* scale.

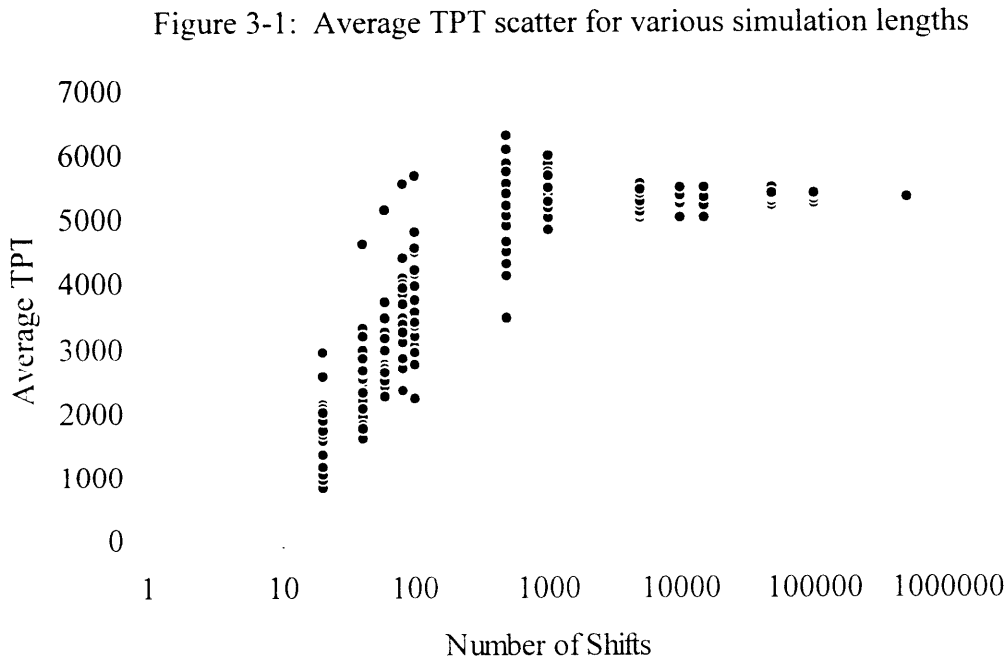
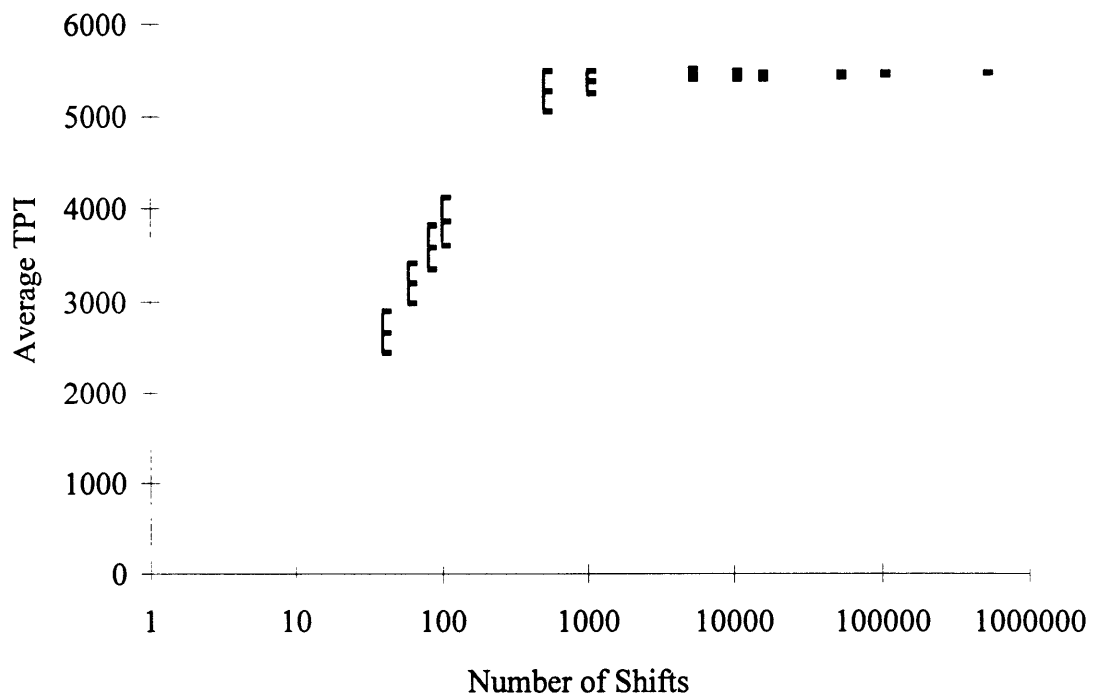


Figure 3-1 is a scatter graph of average TPT. This graph shows 30 data points for each of the simulated time periods. As can be seen, the average TPT fluctuated quite widely at low number of cycles (not surprisingly as the system is in a highly transient state), but did indeed tend to converge to a single value as the number of cycles increased. Overall, there is a well defined operating envelope, which initially increases but decreases

substantially as the number of simulated shifts increases.

Figure 3-2 shows the overall average TPT for each of the time periods (i.e. the average of the sample means for each data set). The upper and lower limits are 95% confidence intervals *on the mean* based on a sample size of 30 (Not to be confused with the  $1.96\sigma$  limits of the *underlying* TPT--they are much larger).<sup>3</sup> These intervals hold relatively constant until 500 shifts, and then decrease substantially after 1000 shifts. Also note that the x-axis here is on a *logarithmic* scale, so the tightening of confidence intervals is a slow and steady process.

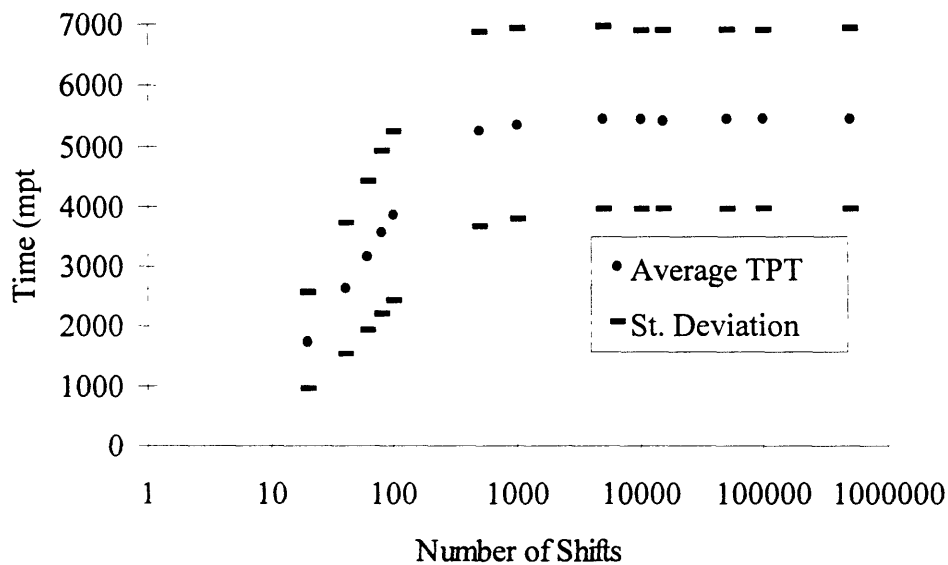
Figure 3-2: : Average TPT vs. number of cycles with 95% confidence interval bars



<sup>3</sup> The procedure used for calculating the confidence intervals is in Appendix A.

Figure 3-3 shows estimates for the expected value and standard deviation of the underlying TPT distribution. The bars around each point represent a deviation of  $1\sigma$  from the mean. The estimator used for the overall mean in a given number of shifts is the sample mean ( $\bar{x}$ ) of all the observations (each “observation” is a parts that went through the system during that simulation). The estimator for the variance ( $\sigma^2$ ) is the sample variance ( $S^2$ ) taken over all observations in that shift. Both the mean and standard deviations tended to grow with time, but leveled out after a large number of shifts.

Figure 3-3: Estimates for mean and standard deviation of TPT vs. number of shifts.



Note that these estimators are exactly that--estimates. They work well (and make sense) when all of the observations come from independent identically distributed random variables (*iidrv*). In the transient state, there is a *different* TPT distribution for every part trying to get through the system, so the distribution is changing almost continuously with

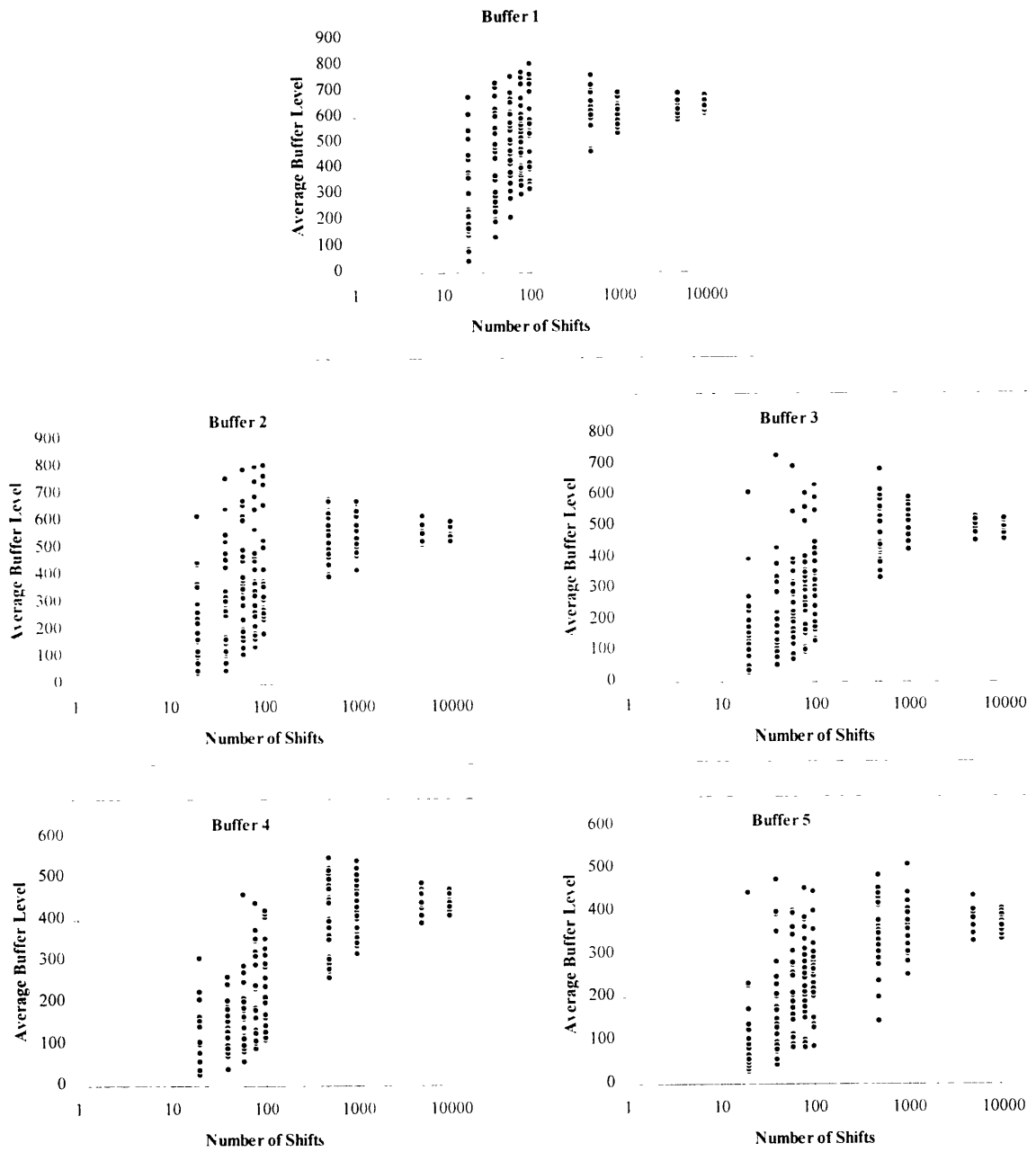
time. For example, for the very first part going through the system, we would expect the mean and variance of the TPT to be relatively small. Chances are the part will get straight through since it does not have to wait in line (although it could still get delayed if a machine fails in the process). The chances for the second part to get through the system, however, are not so optimistic, since it might have to wait in line while the first part is being processed. The 50<sup>th</sup> part will almost certainly have to wait in line, and its TPT is much more likely to be large than for parts 1 and 2. Thus we see that the TPT probability density function is dependent on the state of the factory--the queue sizes, and operation status of each machine--at the time that each part first enters the factory.

In our calculations we have only collected data at certain finite-size time intervals. The estimation errors are further augmented because the means and standard deviations have been computed cumulatively and therefore reflect the parameters of observations which do not come from *iidrv*. Nonetheless, in the steady state it appears as though the distributions converge so that our estimates for the mean and variance become more accurate. The procedure used for calculating the TPT variance can be found in Appendix A.

The average buffer levels are plotted against the number of shifts and are shown in Figure 3-4. We expect to see a correlation between buffer levels and TPT, as explained by Little's Law. These graphs display a convergence trend for the buffer levels as the number of cycles is increased. Overall, there is good agreement between the convergence of *average* buffer levels and the convergence of *average* TPT. Such a result would be

predicted by Little's Law given that the throughput remains relatively constant.

Figure 3-4: Average buffer levels vs. number of cycles for buffers 1-5.



This is further expounded upon in section 3.3.2. Note that this does not imply convergence of the buffer levels or individual TPTs from shift to shift. The buffer levels, and therefore the TPT, will probably continue to fluctuate indefinitely, as evidenced by the relatively large calculated TPT standard deviations.

### 3.3.1 Time For Convergence

One of the objectives of the study was to obtain a reasonable model for the time of convergence of average TPT. To measure this convergence, we analyzed the size of the 95% confidence interval and its decrease with time. To reiterate, this confidence interval was based on observations from 30 independent trials, so the analysis performed here is applicable only to this experimental set up (i.e given a lower number of trials the size of this confidence would be larger).

In particular, such a model would aid in answering the following question: given a desired confidence interval  $x'$ , how much simulated (or "real" factory) time,  $Y$  is needed? From looking at Figure 3-2, a reasonable regression model seems:

$$Y = \beta_0 + \beta_1 x$$

here we used the transformations:  $Y = \ln(Y)$  and  $x = \ln(x')$

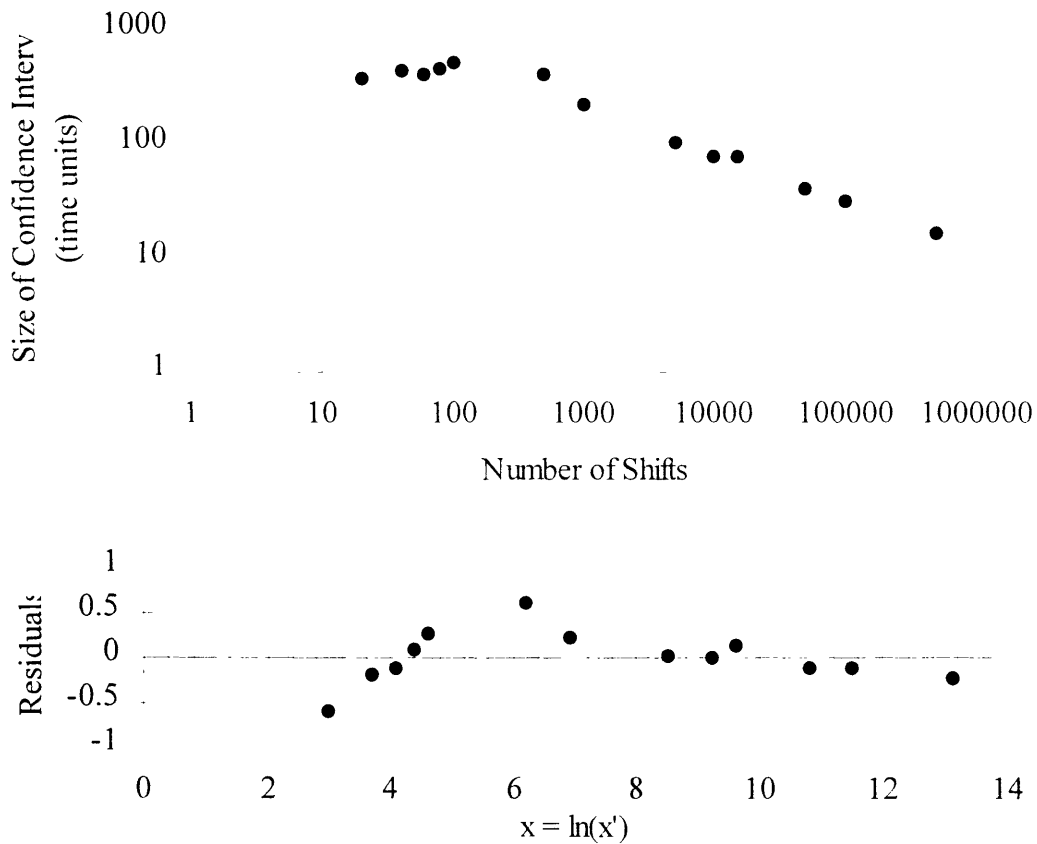
Finding the correlation coefficients by the least squares method, we obtain the regression fit shown in Figure 3-5 (page 56). To check the correlation coefficients we test the null hypothesis  $H_0: \beta_j = 0$  against  $H_1: \beta_j \neq 0$  at significance level of  $\alpha = .05$ , we obtain the following results:

Regression Parameters	Coefficients	t-Statistic	t(.025;11)	Conclusion
Intercept, $\beta_0$	7.561	35.76	2.201	<i>accept H1</i>
$x=\ln(x')$ , $\beta_1$	-0.345	-13.082	2.201	<i>accept H1</i>

Thus the number of simulation cycles is a significant explanatory variable at  $\alpha=.05$ .

The bottom of Figure 3-5 shows the residuals from this model. They do not appear random, i.e. there appears to be data "hugging". This is probably due to the fact that the

Figure 3-5: Regression fit for confidence interval of average TPT vs. number of factory shifts (based on sample size of 30)





observation readings are not independent. The means from simulation to simulation are independent, but they are coupled by the confidence interval calculation (e.g. for the confidence interval calculations, the experiment means are averaged and then the deviations are squared). Nonetheless, this model provides useful information. It is interesting to note, for example, that to decrease the size of the 95% confidence interval by a factor of  $Z$ , the simulated (or factory) time, has to be increased by a factor  $Z^{-1/\beta^1}$  (roughly  $Z^2$ ). For example, to cut the size of the 95% confidence interval in half, the simulation has to be run 7.5 times as long.

### 3.3.2 Little's Law Correlation

Here we will see how closely our obtained TPT values agree with those obtained from Little's Law. This may be a useful tool in calculating TPT, since throughput and buffer levels are often easier to measure on the factory floor than TPT.

To reiterate, Little's relates the average total inventory or WIP (denoted by  $L$ ), average TPT (denoted by  $W$ ) and average production rate or throughput (denoted by  $\lambda$ ):  $L = \lambda W$ . Solving for TPT ( $W$ ), we obtain:

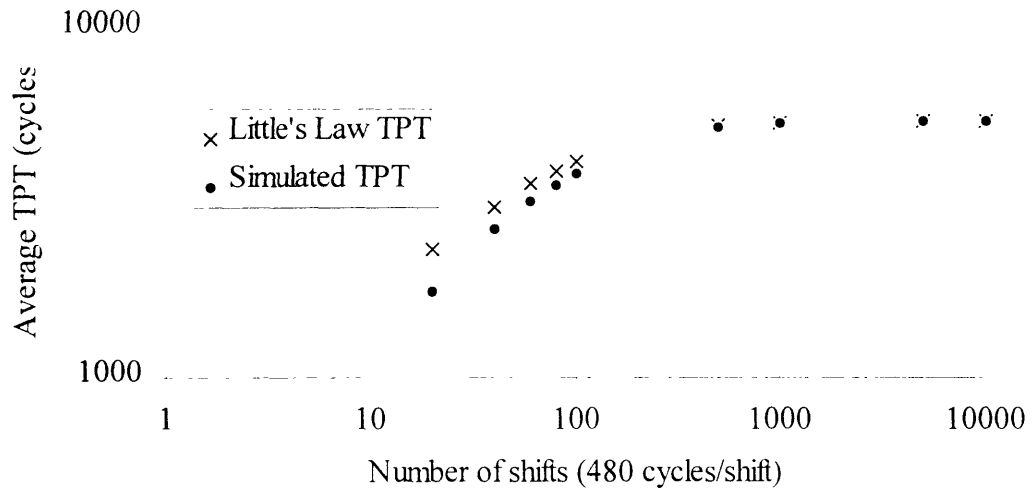
$$W = \frac{L}{\lambda} \quad (3-2)$$

As before, the WIP consists of all parts waiting in queues and parts being worked on.

The graph of Little's Law average TPT and simulated average TPT is shown in Figure 3-

6. The predominant trend in this figure is that the Little's Law average TPT agrees

Figure 3-6: Little's law TPT and simulated TPT vs. number of shifts



more closely with the simulated average TPT as the number of shifts is increased. This indicates that Little's Law is most applicable when flow has reached a steady state.

### 3.4 Conclusions of Study

We draw four major conclusions from this study. First, it is evident that both the mean and the variance of TPT increased with time but tended to converge as time increased. In steady state, the standard deviation was more than 25% of the mean. The numbers below summarize the mean and standard deviation at selected times (rounded to nearest integer).

# of Shifts	20	100	1000	10000	100000	500000
Mean	1764	3859	5362	5437	5446	5447
St. Deviation	815	1400	1573	1472	1467	1471

*Mean and standard deviation are in units of machine processing times.*

Second, the average TPT of each of the thirty experiments tended to converge to the same value (roughly 5500 machine processing times). The standard deviation for each experiment also tended to converge toward the same value (roughly 1500 machine processing times). This suggests that there *does* exist a steady state probability density function for TPT.

Third, this study showed that the average TPT tended to converge logarithmically with time. In particular, to decrease the size of the confidence interval by a factor of  $Z$ , the simulated (or factory) time has to be increased roughly by a factor  $Z^2$ .

Fourth, the agreement with Little's Law was better as the number of shifts increased and worse in the transient state. This is a plausible result, since over the beginning warm up time period the TPT displayed different probability distributions for each run, and in the long term the TPT distributions from each experiment tended to converge to the same distribution.

Based on these findings, it is reasonable to assume that simulation TPT will agree with factory-floor measured TPT *given* that the factory displays similar steady state behavior. Returning once more to the top analogy, we have shown that simulated TPT behaves like a top in a bowl since it has a single steady state spot. Therefore, the simulation has the capability to model actual TPT given that the actual TPT distribution also has a single steady state "spot."

We further explore these issues in Chapter 4 with simulation results for a microprocessor assembly factory.

## **Chapter 4**

### **Case Study: A Microprocessor Assembly**

#### **Series-Parallel Flow Line<sup>1</sup>**

In this chapter we examine the effects of various factors with the goal of obtaining simulated TPTs within 5 to 10% of “factory floor” TPT.<sup>2</sup> We present several DES models and evaluate their effectiveness with validation data. We achieve our best results by adding detail at the constraint and post-constraint processes.

#### **4.1 System Description**

In contrast to most wafer fabrication processes, the assembly process is relatively simple.

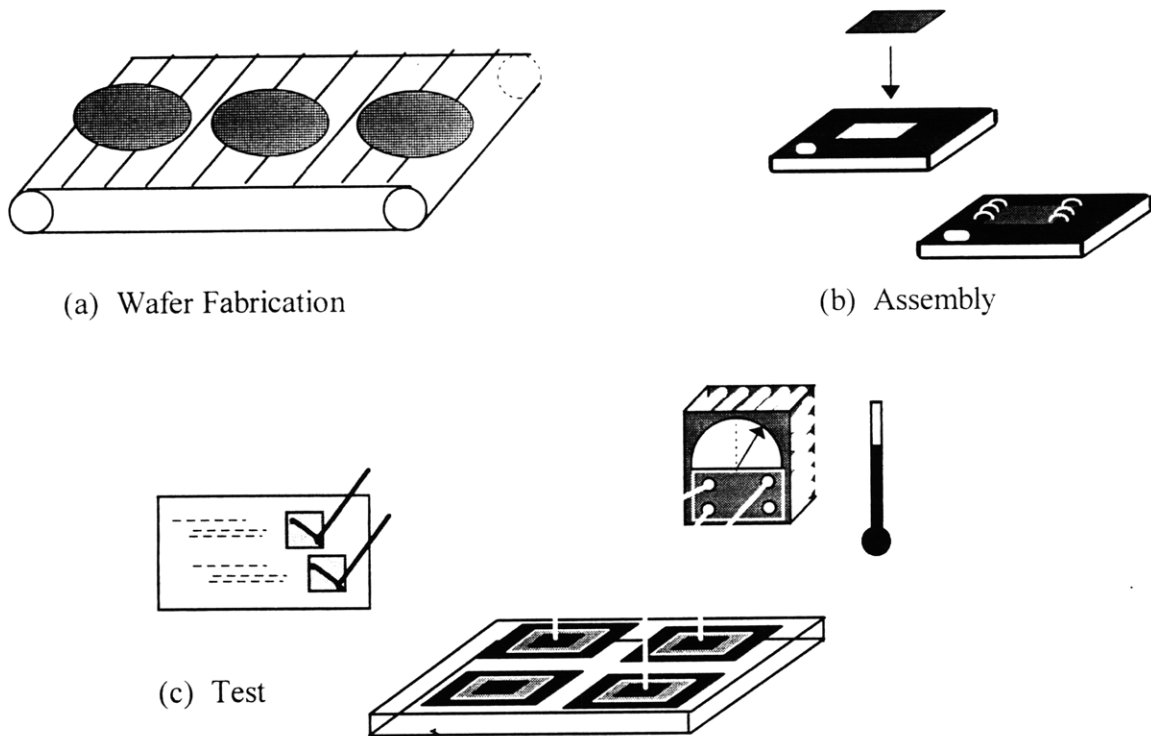
---

<sup>1</sup>This case study is part of work done at a semiconductor assembly factory. Due to the proprietary nature of information, many of the data in this section have been omitted or normalized.

<sup>2</sup>Since there are errors in any “measured” or “reported” TPT values and, furthermore, the magnitude of these errors is unknown, we will use the term “factory floor” in lieu of “actual” TPT. Evaluating the reliability of factory floor validation data is a complex task which is beyond the scope of this study.

In particular, there is no reentrancy and there is a well defined set of machines used for each product. There remain, however, modeling difficulties with respect to TPT, especially in obtaining inexpensive models that are reasonably accurate. As mentioned before, DES models in the semiconductor industry commonly yield average TPTs which are 25-50% of actual. The system studied is the standard chip-packaging assembly process used by most semiconductor manufacturers. The assembly process starts with the microprocessor chips, or *dice* (plural of *die*), on unsawn wafers and ends with the codemarking of packaged die. The entire chip manufacturing processes is illustrated in Figure 4-1.

Figure 4-1: Chip manufacturing process

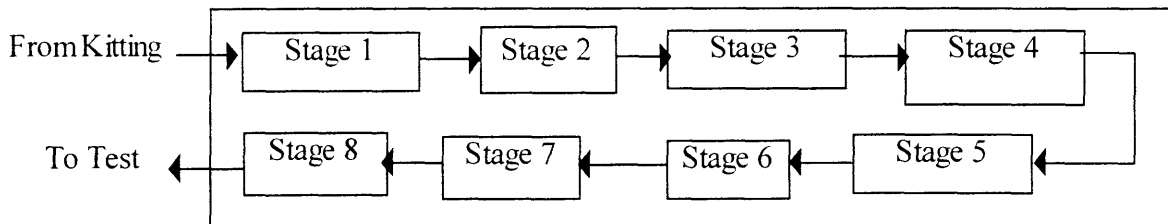


Wafers are manufactured in the FABs. Once fabricated, the wafers are sent from the FABs to a “resting” spot referred to as *kitting* (not shown in the figure). The wafers are then released to the assembly floor, where they are packaged. Next, the packaged chips are sent to test, where they undergo a series of mechanical and electrical performance tests.

#### 4.1.1 Brief Overview of Process Flow

The process flow of the assembly process is comprised of eight principal steps, as illustrated in Figure 4-2. Due to confidentiality restrictions, we will refer to these as *stage 1* through *stage 8*. Each of these stages consists of a set of parallel machines among which lots are split (e.g. similar to a job-shop configuration). Stage 5 can be considered the bottleneck of the process.

Figure 4-2: Assembly process flow



We will refer to stages 1 through 4 as the *pre-constraint* processes, stage 5 as the *constraining* process, and stages 6 through 8 as the *post-constraint* processes.

#### 4.1.2 Lot and Sub-lot Configurations

The assembly process is somewhat complex from a logistics standpoint in that the lots are

split and re-merged into other lots at several stages. In particular, the constraining stage (stage 5) is composed of a series of parallel machine centers, or bays, each of which is comprised of several machines. Each bay processes a lot, which is split among the machines in that bay. The flow logistics at the constraint and other processes are expounded in the following section

### **4.1.3 Resources and Logistics**

There are two critical types of resources in the assembly process: machines and operators. Although material is generally considered a resource, it does not represent a binding constraint since, during the period studied, there was no starvation due to lack of available material.

In the following sections we present descriptions of the functions performed by the operators, WIP management strategies and machine parameters.

#### **4.1.3.1 Operator Activities**

Most machine stations are staffed with a dedicated group of operators. The operator activities consist mainly of loading and unloading lots to machines, transporting lots from machine to machine, and inspecting parts at certain steps. Although there is also preventative maintenance performed on the machines, this is generally done by outside operators, who are considered resources external to this system.

Not only is stage 5 the system's constraint, but it is also the most complicated from a



logistics and WIP management standpoint. Stage 5 machines are separated into several bays, each bay being staffed by multiple operators. Stage 5 operator tasks are to pull each WIP cart into their respective bay and load and unload boats (fractions of a lot) to each machine. Thus, each lot is split into boats and processed by the different machines in that bay. As each boat finishes processing, the finished parts are inspected. Once inspected, the finished boats are put back onto the WIP carts. The operators then transport the finished lots to a general stage 6 buffer. Stage 5 logistics are described in greater detail in section 4.1.3.2

#### **4.1.3.2 Loads and WIP Management Strategies**

In addition to the procedures followed by the operators as described in section 4.1.3.1, the following rules are followed. All lots are processed on a first-in first-out (FIFO) basis. For the stage 3 and 5 operations the operators are generally handed an itinerary with the lots to be processed during the shift; such itineraries, nevertheless, usually follow FIFO procedures and are used primarily to insure proper lot tracking. In general there is no batching except at the stage 4. Inter-lot inspection policy is, for all intents and purposes, also equivalent to a FIFO policy.

Due to the splitting of lots across machines at stage 5, this process is the most logistically complicated in the assembly process. A new lot is pulled into each stage 5 bay as machines become available. For example, assume that a bay has sixteen machines and

that there are thirty-three boats to a lot.<sup>3</sup> Furthermore, assume that each boat takes exactly one minute to process, and that the machines are perfectly reliable. The throughput time of several lots in such a system is illustrated in Table 4-1. When the first lot goes into the system (*time 0*), it is split into thirty-three boats. Since there are sixteen free machines, sixteen boats go into processing while the remaining seventeen wait on the WIP cart. After one minute (*time 1*) the first sixteen boats finish processing (barring any failures); they are then inspected and put back onto the WIP carts. The next sixteen boats are then put into processing. After one more minute, those sixteen boats are completed and the final boat from that lot is loaded onto a machine. This leaves fifteen machines free, so a new lot is pulled into the bay, and fifteen of its lots begin processing. At *time 2*, the final boat from the first boat and fifteen boats from the second lot finish processing. The TPT for the first lot is therefore three minutes. Sixteen lots from the second lot are then loaded. When these finish, the remaining two boats from the second lot are processed along with fifteen from the third lot. The throughput time of the second lot is therefore three minutes. This loading policy is continued for all lots across all bays.

We've generalized this for the  $n$ th lot, which has  $16\left\lfloor\frac{n}{16}\right\rfloor + 1 - n$  of its boats processed at time  $2(n-1) + \left\lfloor\frac{n-1}{16}\right\rfloor$ , and the final  $16\left\lceil\frac{1-n}{16}\right\rceil + n$  boats processed at time  $2n + \left\lceil\frac{n-1}{16}\right\rceil$  (here  $\lfloor x \rfloor$  is  $x$  rounded *down* to the nearest integer and  $\lceil x \rceil$  is  $x$  rounded *up* to the nearest integer). Although not evident in this example, there is great variability in the TPT of this system because the lots do not arrive at constant and regular intervals. This

---

<sup>3</sup> These numbers were picked for illustration only and do not apply to the factory being studied.

variability is further augmented by machine failures, set-ups, and the fact that not all bays have the same number of machines and not all lots contain exactly the same number of magazines.

Table 4-1: Example of lot processing procedure for a stage 5 bay

Boats processed during a given time interval										
Time→	0	1	2	3	4	5	$2(n-1) + \left\lfloor \frac{n-1}{16} \right\rfloor$	$2n-1 + \left\lfloor \frac{n-1}{16} \right\rfloor$	$2n + \left\lfloor \frac{n-1}{16} \right\rfloor$	<i>tpt</i>
Lot 1	16	16	1							3
Lot 2			15	16	2					3
Lot 3					14	16	...			3+
Lot n							$16 \left\lfloor \frac{n}{16} \right\rfloor + 1 - n$	16	$16 \left\lfloor \frac{1-n}{16} \right\rfloor + n$	3+

#### 4.1.3.3 Resource Counts and Machine Parameters

As described before, there is a two level hierarchy for machines. Machines which all perform the same, or similar functions are grouped into *station families*. Each of these stages can be characterized by the number of machines it contains and the machine processing times for each machine.<sup>4</sup> As defined in Chapter 1, the processing is the “ideal” or “theoretical” time, which is less than the effective time due to failures, assists, starvation, etc. The variability in processing time is extremely small compared to the variability in downtimes and other parameters; for this reason, it is reasonable to assume

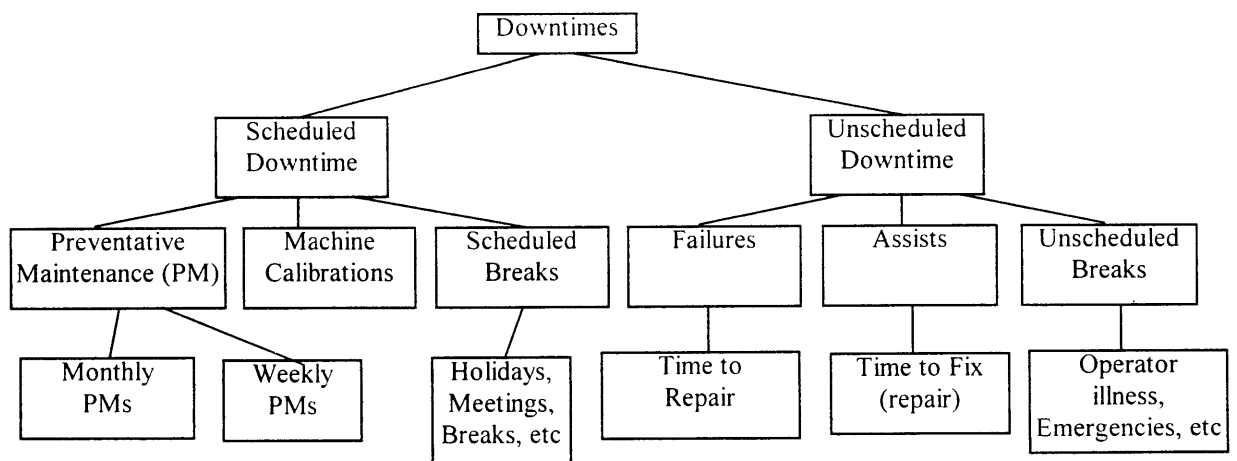
---

<sup>4</sup> Confidentiality restricts publication of these figures.

that they are constant and deterministic.

Since these machines are not perfectly reliable, it is also necessary to describe uptimes and downtimes for each machine. To account for machine downtimes, there is usually a classification scheme. A tree of a sample classification scheme is shown in Figure 4-3. This is not an exhaustive or comprehensive list, but is merely presented to show the major groups and as an example of common classification scheme. Downtimes are generally grouped into scheduled and unscheduled downtimes. Scheduled downtimes include preventative maintenance, machine calibration, and regularly scheduled operator breaks. Unscheduled downtimes are due to random failures, which range in magnitude from small breakdowns to substantive failures which could require new equipment, ordering of special parts, etc. For this reason, unscheduled downtimes are generally

Figure 4-3: Downtime Classification Scheme



much harder to describe and predict than scheduled downtimes. Idle time is not considered part of downtime. rather, idle time results from machine starvation and/or blockage. Thus the total time equation is given by the following formula:

$$\begin{aligned}
 \text{TOTAL TIME} = & \\
 \text{SCHEDULED DOWNTIME} + \text{UNSCHEDULED DOWNTIME} & \quad \} \text{ Downtimes} \\
 + \text{MACHINE IDLE TIME}^5 + \text{PROCESSING TIME} & \quad \} \text{ Uptimes}
 \end{aligned}$$

Characterizing the factory downtime is particularly useful when trying to *model* the factory. In simple analytical models, the typical parameters are buffer sizes (usually denoted by  $N$ ), a failure rate (usually denoted by  $\rho$ ), and a repair rate (usually denoted by  $r$ ). Based on these, the models assume certain probability distributions for the up and downtimes. DES models, have the advantage of allowing the user to input many more details. Uptimes and downtimes can assume actual probability mass functions given by historical data (although such data is often difficult to obtain).

Unscheduled downtimes in DES models are often divided into *failures* and *assists*. Assists are generally defined as short failures which can usually be fixed by machine operators, *failures* are defined as all other machine failures, and usually entail specialized personnel to repair. Since the distributions for failures and assists are often unknown (due to unavailability of historical data), modelers usually have assume a distribution

---

<sup>5</sup> It should be noted that for scheduling purposes, there is usually a “gap” term added to this equation, to account for small “idle” time intervals that are difficult to describe. Typically, industrial engineers will have a “desired gap value”, and schedule factories to obtain these values.

based on available parameters, namely the mean time to failure (MTTF), mean time to repair (MTTR), mean time to fail for assists (MTTF<sub>A</sub>) and mean time to assist (MTTA).

In these models we generally assume that all machines in a station family have identical failure and repair parameters. In reality, each machine has different parameters which describe its performance (i.e. there can be significant differences in the reliability of machines across a certain station family). Due to the large number of machines and unavailability of specific data, the modeling process is vastly simplified by assuming that the machines at each station are identical.

In the next two sections, we present several DES models of the assembly process we have described.

## 4.2 DES Modeling

In this section we describe a series of DES models that were created with the objective of obtaining accurate TPT predictions. The DES models were made progressively more detailed. Many of the items presented in Chapter 2 such as operator availability and shift effects are explored in detail. We begin with a background discussion of DES models.

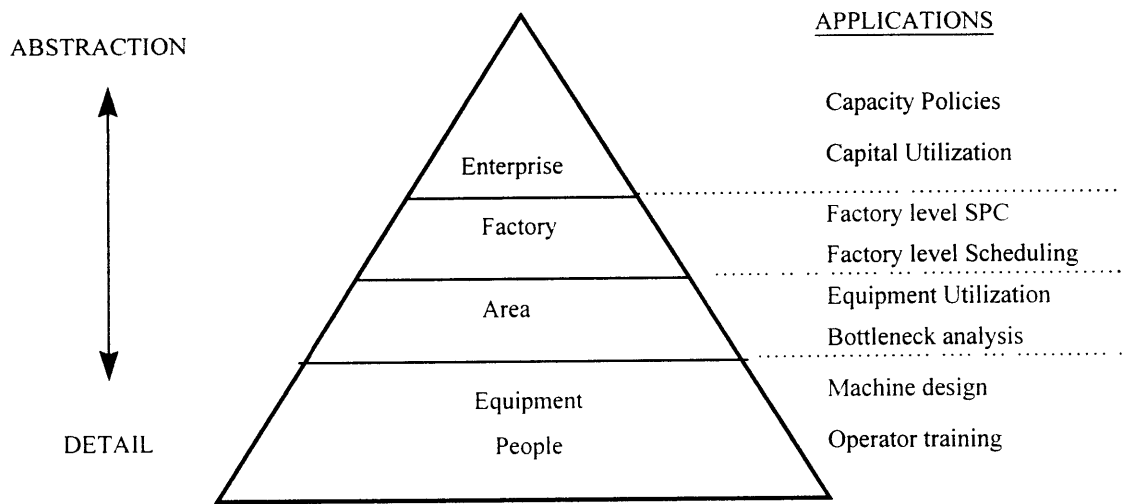
### 4.2.1 Background / Model Hierarchy

DES models have a clear advantage over analytical methods in that they can be made, in theory, arbitrarily accurate. Nonetheless, there is a clear tradeoff between accuracy and computational cost. With excessive detail, models can become prohibitively expensive; it therefore behooves the modeler to determine which factors are the most significant. Indeed, detail can be regarded as a resource which needs to be carefully allocated to each element to be modeled.

A diagram of a model hierarchy is shown in Figure 4-4. This is a conceptual diagram in which the vertical axis represents the level of detail of the model. The top of the pyramid represents the most general all-encompassing models, and the base of the pyramid represents the most specific models. The higher level models (*enterprise* or *factory*) are less detailed, but answer broader questions such as capacity policies or capacity utilization. The lower level models (*equipment* and *people*) are more detailed and help answer much more specific questions such as machine design and operator training. Examples in the *enterprise* regime are capacity policy and capital utilization models.

Examples of *equipment* models are computer aided design (CAD) or kinematic models which are capable of estimating the throughput of a single machine, exploring mechanical details of machine failure modes, etc.

Figure 4-4: Model Hierarchy



*Courtesy of Eugene S. Meiran, Intel Corporation*

Typical DES factory models are in the *factory* or *area* domain. They are suitable to give throughput and TPT estimates of factories or station families, but not detailed enough to give detailed “mechanical” information about a particular machine. DES scheduling models often only include throughput numbers and failure parameters, but do not include the physical characteristics of every machine.

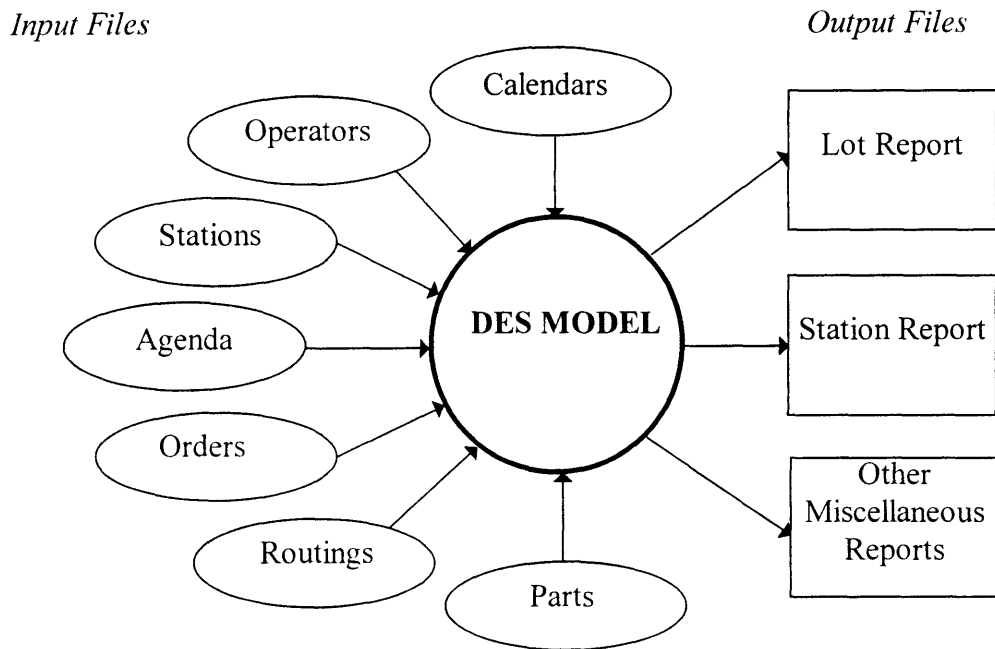
#### 4.2.2 Model Abstractions and Interface Architecture

As mentioned in the previous section, DES models generally take as input machine processing times, buffer capacities, and descriptions of the uptime and downtime



distributions rather than the detailed physical characteristics of each machine. All of the DES models in this study were constructed with the aid of *Autosched*<sup>6</sup>, a commercial DES package which is widespread in the semiconductor and other manufacturing industries. The schematic of the *Autosched* input interface (which is similar to other packages) is shown in Figure 4-5.

Figure 4-5: Simulation Input Interface and Output Reports



This is a simplified schematic which only shows the major categories of inputs and examples of possible outputs. Following is a basic description of each item.

---

<sup>6</sup> Trademark of AutoSimulations Inc.

*Calendars:* In these files the user is allowed to input “schedules” for downtimes. Exact times can be input for scheduled downtimes, while unscheduled downtimes can be modeled by probability distributions.

*Operators:* Through these files the user can define operators and assign them to certain machines or processes. In addition, the user can define certain levels of operator classification rules by which operators select tasks, etc.

*Stations:* In this file users define the various station families and the characteristics pertaining to each of the machines in the station families (except for failure and repair behavior, which is described in the *calendars* file). In addition, the user can assign rules to each machine or station by which it selects the next part to be loaded.

*Agenda:* In this file the user can specify set up states and set up times as required by different loads or part types.

*Parts:* This file allows the user to define one or several part types that are processed by the system.

*Orders:* In this file the user specifies the factory volume, i.e. the number of parts and time at which each part enters the system.

*Routings:* In this file, the user define routes for the parts defined in the *parts* file. The user inputs the machines or station families visited by each part and the processing time at each machine. In addition the user can input lot splitting and merging information.

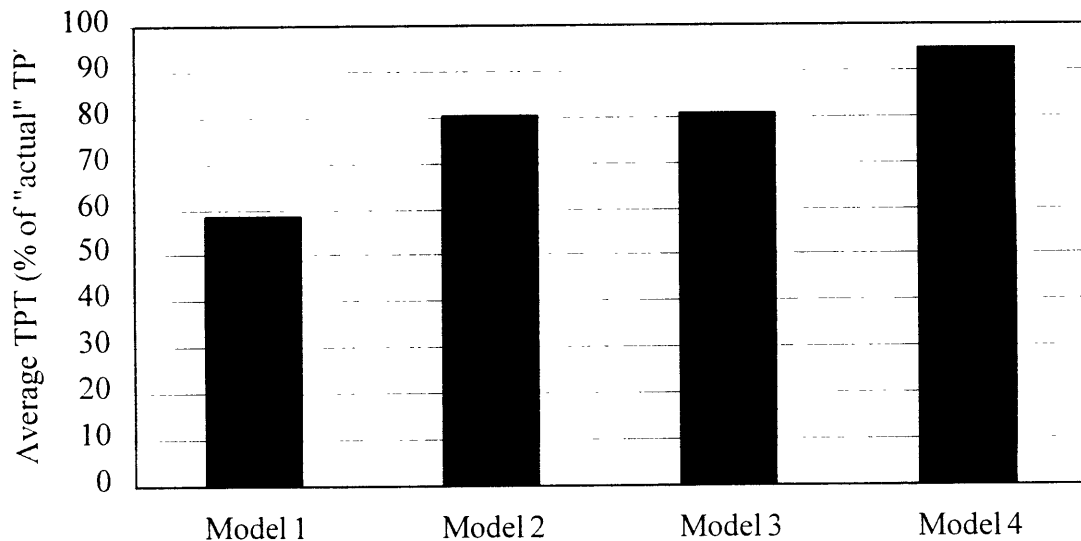
*Output Reports:* These reports can be customized to include statistics of the lots, parts, or other system characteristics. In this study, output reports were used for total average TPT ( $\overline{TPT}$ ), and step-to step average TPT ( $\overline{tpt}$ ).

In the following section we describe in detail numerous models that were built of the assembly process. All of these models were constructed with *Autosched*, using the input system described above.

### 4.2.3 Description of Models

Figure 4-6 shows the results of progressively detailed DES models. There are subsequent model iterations on the *x*-axis. The *y*-axis represents average TPT. The units of TPT are percentage of factory floor or "actual" TPT.<sup>7</sup>

Figure 4-6: Average TPT for a progressively detailed DES models



There were four main models created, with the final model having a TPT of 94% of factory floor TPT. All were created with *Autosched*. The time-series average TPT behavior of these models was similar to the model described in Chapter 3. Since the warm up period was found to be only a few days, the models were run for a time period

---

<sup>7</sup> Recall we will use the term "factory floor" in lieu of "actual" due to the unconfirmed reliability of validation data. When the word "actual" is used it will be in quotation marks.

of three months (i.e., three "factory" months were simulated). The data presented in Figure 4-6 disregards transient period data, i.e. data was collected after a model "warm up" period. We will discuss each model in detail in the following sections.

#### **4.2.3.1 Model 1**

The following were included in each of the DES input files:

*Stations:* All station families were instructed to use FIFO as their task selection rule, except for stage 5. Due to the complicated lot allocation scheme at stage 5, custom "filters" were defined to send each lot in the common queue to an available bay as it became available. In addition machines were instructed to batch lots at stage 4 as described in section 4.1.3.1. No inspection stages were included in this model.<sup>8</sup>

*Parts:* Two part types were defined for each type of package. For each, routings were defined under the routings file.

*Orders:* A rough number was obtained for the factory volume. This figure was obtained from speaking to various stage 1 and stage 2 shift managers. It should be noted that such numbers are often difficult to obtain since the factory volume changes from day to day (and these changes can be very drastic for the "ramp-up" period). Although in reality

---

<sup>8</sup> After looking at factory floor data, shown in Figure 4-7, the only significant inspection stage was found to be that immediately after the stage 5 operation.

wafer lots are released periodically throughout the day. in this model parts were *assumed* to be released every shift.

*Routings*: Two separate routes were defined for each part type. Since the primary focus of the study was average assembly TPT for one of these packages types<sup>9</sup>, the other packages were only included in the stage 1 and stage 2 process, after which point they exited the system. This is a reasonable modeling procedure, since there is no resource contention for the post stage 2 processes, i.e. the machines beyond stage 2 are entirely dedicated to working on a single package type.<sup>10</sup> All of the lots entered the modeled system as *assembly lots*, although in reality these enter the system as *wafer lots*. This was done in the interest of reducing computational expense. Transport times were also excluded from this model.

At stage 5, each lot was split into boats, and each boat was processed separately by each of the machines in a given bay. This worked in conjunction with the WIP management strategy defined through the custom filter called in *stations*.

*Calendars*: Exact preventative maintenance schedules were input for each of the machines. Unscheduled downtimes were separated into *failures* and *assists*. All of MTTF, MTTR,  $MTTF_A$  (assists), and MTTA were *assumed* to have exponential used-

---

<sup>9</sup> Confidentiality restricts descriptions of the package types

<sup>10</sup> The machines are, however, shifted to different product lines from time to time. The time horizon chosen for this study had no significant machine changes, so this is not thought to have lead to TPT error.

based distribution.

*Operators:* No operators were included in this model.

*Agenda:* Since most of the assembly line has dedicated sets of machines, there are no significant “set-up” requirements. This input file was omitted from the model.

#### **Summary of Model 1 Assumptions:**

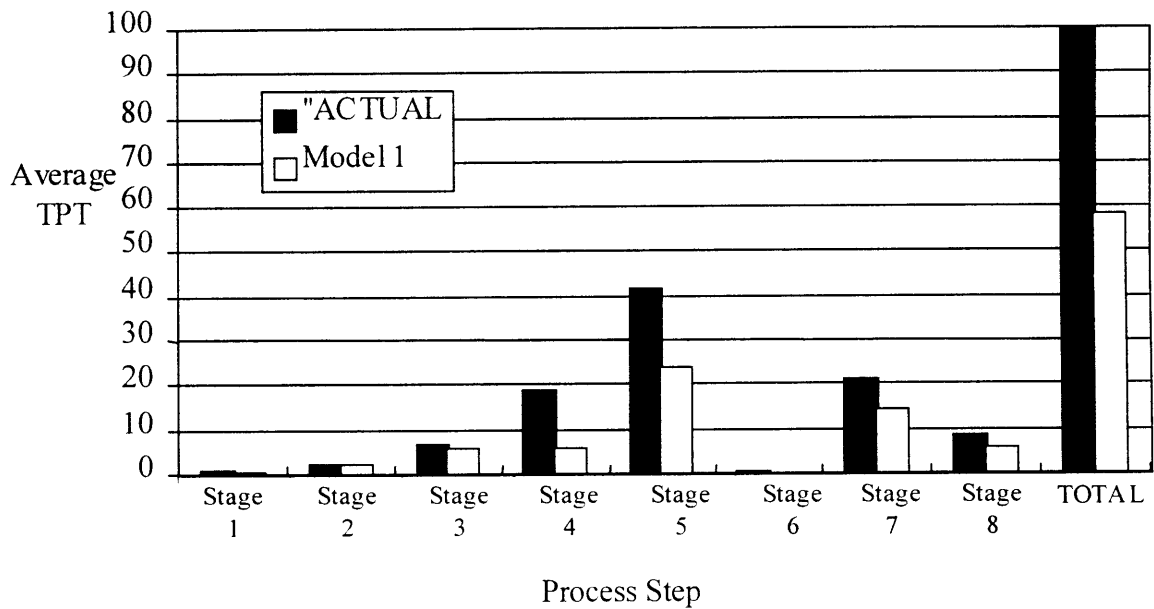
A rough number for lot releases was used. In addition, lot releases were assumed to occur on a shiftly rather than on a continuous basis. The parts flowed through stage 2 and stage 1 in *assembly* lots, rather than *wafer* lots. All uptime and downtime distributions were assumed exponential based on use. Operators and transport times were omitted from the model.

#### **4.2.3.2 Model 1 results**

This model turned out to be very expensive, with run times of nearly an hour per simulated day. The inefficiency of the model was attributed to the complicated splitting and batching scheme required at stage 5. In addition, the model underpredicted average TPT by more than 40%. The step by step average TPTs are shown in Figure 4-7. The step by step TPTs are calculated from the time a lot enters the station family queue for the process in question, until it enters the next process’ family queue. The model was very accurate in predicting average TPT for the stage 1 and stage 2 operations.

The largest TPT discrepancies were at the post die-attach operations, particularly for stage 4 and stage 5 where the average TPT is underpredicted by close to 50%.

Figure 4-7: Step by step average TPT for *model 1*



#### 4.2.3.3 Model 2

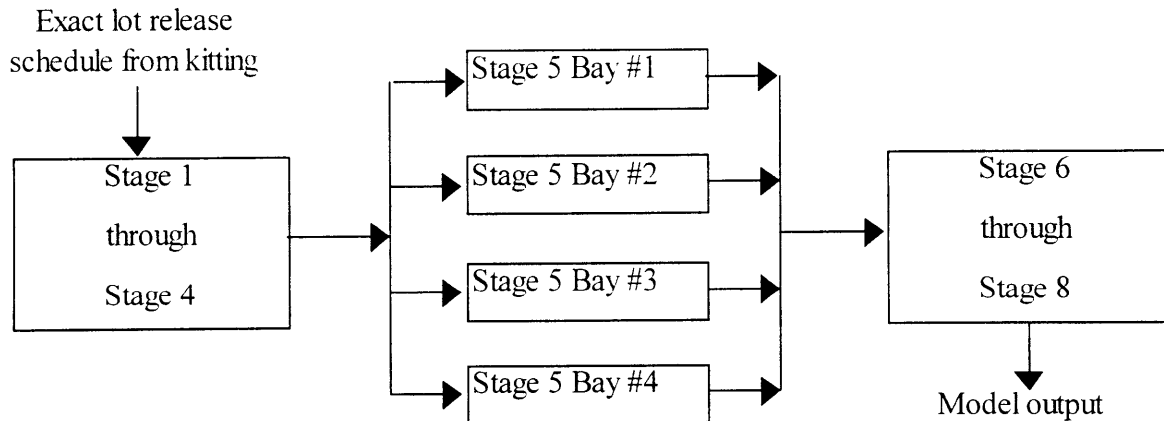
The primary weakness of the first model, in addition to the large underprediction of TPT, was the prohibitive model run-times. The first change made in the second model was the use of an exact lot-start schedule. The exact lot-start schedule provided an increase in volume of 12%. This higher volume made run-times even slower and thus motivated the decomposition of the model into three separate segments: a pre-constraint model (stage 1 through stage 4), a constraint model (stage 5), and post-constraint model (stage 7 and stage 8). This decomposition had the added benefit of making details at stage 5



affordable.

Each stage 5 bay was modeled separately, so there were a total of six separate models: one model covering stage 1 through stage 4, one model for each of the four stage 5 bays, and a final model for stage 7 and stage 8. This is illustrated in Figure 4-8. (From this point on we will refer to the stage 1 through stage 4 model as the *pre-constraint model*, and to the stage 7 and stage 8 model as the *post-constraint model*). The advantage of this was two-fold: (1) it parallelized the process, vastly reducing computational effort, and (2) it allowed for much easier manipulation of parameters at the stage 5 (e.g. inclusion of operators, different shift schedules, etc.).

Figure 4-8: Flow of data in decomposed model

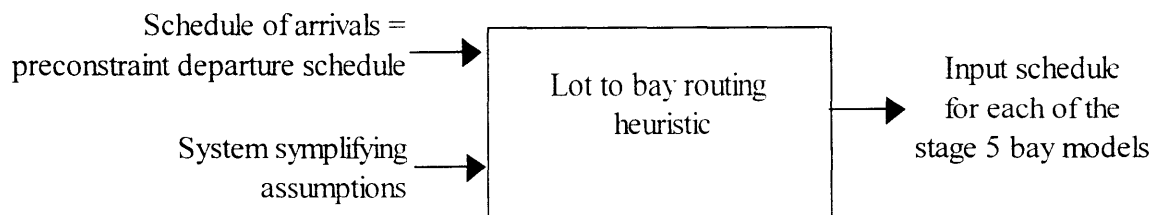


An exact lot start schedule was given as input to the first model. The output of this first model was then used as input to the stage 5 bay models. Lastly, the output from the stage 5 bay models was input to the stage 7 and stage 8 model.

## Heuristics for Lot Allocation to Stage 5 Bays

Although this decomposition allowed great flexibility and did indeed reduce run times, there were a few major challenges. In particular, a heuristic had to be developed for mapping the outputs of the first model to each of the stage 5 bay models. In reality, the bay models are interrelated, since their relative processing characteristics determine to which bay each lot is routed. For example, if the fourth bay has many machines that are malfunctioning, this will increase the loads on the first three bays. Thus, it is of interest to obtain a schedule which assigns each lot to a stage 5 bay, i.e. a lot-start schedule for each of the stage 5 bay models. Such an assignment schedule can be created based on an arrival schedule to the stage 5 process and some simplifying assumptions about the system, as illustrated in Figure 4-9.

Figure 4-9: Lot routing heuristic



One solution is to run the models in small time increments, check the input queue of each model, and reallocate lots to each model if there are large discrepancies among the queues. To illustrate this method we will take an extreme example. Assume that the model is initialized and the first four lots enter the system. All of the machines are initially empty, so each lot goes to a stage 5 bay. A new lot then arrives at the stage 5

station family queue, and the question arises as to which bay this lot should be sent. By the procedure discussed above, we could pause each of the stage 5 models at very short time intervals, and determine the first bay that finishes processing its lot. This bay would then become the recipient of the new lot. This procedure could be repeated for every lot waiting to be processed. Although this heuristic would produce close to exact behavior, it would add significant time to the modeling process, defeating the purpose of the model parallelization. To keep in the spirit of reducing computational effort and truly parallelize the process, lot allocation to each bay can be done entirely a-priori based on an estimate for the throughput rate of each bay. This suggests a second possible procedure, which we will refer to as the “average processing rate” heuristic. Assuming that all stage 5 machines have identical failure and repair rates, then, in the long term, the number of lots processed by each bay will be proportional to that bay’s average processing rate. Thus, over a given *long* time period, the average throughput rate of each bay will also be proportional to the number of machines it contains. For example, assume that there are three bays of  $x$  machines and one bay of  $y$  machines (a total of  $3x+y$  machines), then it is reasonable to assume that each of the first three bays processes  $(\frac{x}{3x+y})$  of all lots, and the last bay processes  $(\frac{y}{3x+y})$  of all lots. These fractions also determine each bay’s relative processing rate.

Based on this fact, we developed a heuristic which is illustrated in Table 4-2. Given a schedule of lot starts at stage 5, we know what fraction of the lots will be processed by

each bay. Let us assume that the throughput rate of each bay is equal to its average long-term effective processing rate, as given by the fractions above. Then, the heuristic is as follows. Assume that initially all bays are available. Then the very first lot that arrives can go to any of the bays. As a convention, when more than one bay is available, the lot will go to the lowest numbered bay. Then the first three lots will go straight to bays 1 through 4, respectively. The fifth lot will then wait in queue until there is an available bay (assuming the lot arrives while other lots are processing, as is most often the case for a bottleneck). Since we are assuming that the first three bays have identical processing rates and no failures, then the next available bay will be the first bay. Thus, the fifth lot will go into the first bay, and the first bay will again become available after it has finished processing the fifth lot. By a similar argument, the sixth lot will go to the second bay. In sum, we treat each bay as a machine with a deterministic processing rate which is proportional to the number of machines in that bay.

This heuristic can be implemented through the use of a spreadsheet, as shown in Table 4-2. The output schedule from the pre-constraint model can be equated to the arrival time schedule for the constraint model, which can be input into the second column of Table 4-2.

Initially, we assume that all bays are available, thus the first four lots can go immediately to processing. The lot departure time can be calculated as the arrival time plus the queue time (which is zero for the first four lots), plus the processing time. Based on the lot departure time, we can change the value of the dynamic variable  $tf_i$ ; namely  $tf_i$  can be set

to the departure time of the latest lot which was processed by bay  $i$ . Thus, we make lot assignments based on the first bay which will become available for a particular lot in queue, which is  $\text{argmin}(tf_i)$ . Using  $tf_i$  we can also calculate the queue time and thus the *new*  $tf_i$  becomes the arrival time plus the queue time plus the processing time of the lot in question. This can also be expressed as the *old*  $tf_i$  for that bay plus the lot processing time, i.e.

$$(tf_i)_t = (tf_i)_{t-1} + PR(B_i)$$

We can repeat this time accounting for all of the arrival times and in the end we will obtain the desired bay assignments for each of the incoming lots. We have provided a sample spreadsheet of this heuristic in Appendix B.1.

Table 4-2: “Average processing rate” heuristic used for routing lots to stage 5 bays

Lot	Arrival time (ta)	Time in queue (tq)	Bay Assignment $\text{argmin}(tf_i)^*$	Lot Departure time (td)	Time at which Bay $i$ becomes free ( $tf_i$ )			
					$tf_1$	$tf_2$	$tf_3$	$tf_4$
Lot 1	$ta_1$	0	1	$td_1 = ta_1 + 1/PR(B_1)^{**}$	$td_1$			
Lot 2	$ta_2$	0	2	$td_2 = ta_2 + 1/PR(B_2)$		$td_2$		
Lot 3	$ta_3$	0	3	$td_3 = ta_3 + 1/PR(B_3)$			$td_3$	
Lot 4	$ta_4$	0	4	$td_4 = ta_4 + 1/PR(B_4)$				$td_4$
Lot 5	$ta_5$	$tq_5 = \min(tf_i) - ta_5$	1	$td_5 = ta_5 + tq_5 + 1/PR(B_1)$	$td_5$			
Lot 6	$ta_6$	$tq_6 = \min(tf_i) - ta_6$	2	$td_6 = ta_6 + tq_6 + 1/PR(B_2)$		$td_6$		
Lot 7	$ta_7$	$tq_7 = \min(tf_i) - ta_7$	3	$td_7 = ta_7 + tq_7 + 1/PR(B_2)$			$td_7$	
Lot 8	$ta_8$	$tq_8 = \min(tf_i) - ta_8$	1	$td_8 = ta_8 + tq_8 + 1/PR(B_1)$	$td_8$			
Lot $n$	$ta_n$	$tq_n = \min(tf_i) - ta_n$	$\text{argmin}(tf_i)$	$td_n = ta_n + tq_n + 1/PR(B_n)$	†	†	†	†

NOTES: \*  $\text{argmin}(tf_i)$  is the bay corresponding to the minimum of the times  $tf_i$ .

\*\*  $PR(B_i)$ : Processing rate of bay  $i$ .  $PR(B_1) = PR(B_2) = PR(B_4) = x/(3x+y)$ ,  $PR(B_3) = y/(3x+y)$

† For all  $i$  such that a  $\text{argmin}(tf_i) \neq i$  this entry is blank (i.e. the value of  $tf_i$  is not updated). If  $\text{argmin}(tf_i) = i$  then  $tf_i$  is set to  $td_i$

Still, there are problems with this heuristic. In particular, it is incorrect to assume that no lot can go into a bay before it has finished processing the current lot. In reality each bay is processing two lots simultaneously most of the time (refer back to explanation of stage 5 WIP management strategy, illustrated in Table 4-1). This suggests a third heuristic. Assume that each individual machine is perfectly reliable. Here we will think of factory running in discrete time. We will look at the state of the stage 5 bays at a given time interval, and since we are assuming that there are no failures, the factory runs deterministically. Let us make two more assumptions: the first four lots arrive simultaneously, and the stage 5 bays are never starved or blocked. Thus, if we are looking for “interesting” events, note that all lots enter and exit each of the bays at multiples of the processing time of a boat. This is illustrated in Table 4-3. In this table, each time period is the processing time for one boat. The numbers in the upper left side of the shaded squares are the lot numbers, numbered chronologically (e.g. the first lot to enter is number one, second lot number 2, etc.). The other numbers in each shaded area are the number of boats in that lot that are processed during the given time period (time periods are shown on the first column). This yields the schedule shown on the right (under Bay assignments). Note that after thirty-three time periods (in which exactly fifty-five lots are processed), this sequence repeats, *ad infinitum*.

The implementation of this heuristic is as follows: given a schedule of pre-constraint model completion times, we can rank these in order (1-55, 56-110, etc.), and assign them to bays as dictated by Table 4-3. This heuristic has the advantage of having a repeating

structure, which is easier to implement. It is also, in some respects, more accurate than the “average processing time” heuristic in that it takes into account the fact that each bay processes several lots simultaneously.

Table 4-3: Illustration of “deterministic factory” heuristic

Time Period	Number of boats processed at each time interval			
	Bay 1	Bay 2	Bay 3	Bay 4
1	1 16	2 16	3 16	4 7
2	16	16	16	7
3	1 5 15	1 6 15	1 7 15	7
4	16	16	16	7
5	8 14	2 9 14	2 10 14	2 5 11 2
6	16	16	16	7
7	3 12 13	3 13 13	3 14 13	7
8	16	16	16	7
9	15 12	4 16 12	4 17 12	4 7
10	16	16	16	18 4 3
11	5 19 11	5 20 11	5 21 11	7
12	16	16	16	7
13	22 10	6 23 10	6 24 10	6 7
14	16	16	16	7
15	7 26 9	7 27 9	7 28 9	1 25 6
16	16	16	16	7
17	29 8	8 30 8	8 31 8	8 7
18	16	16	16	7
19	9 32 7	9 33 7	9 34 7	35 1 6
20	16	16	16	7
21	36 6	10 37 6	10 38 6	10 7
22	16	16	16	7
23	11 39 5	11 40 5	11 41 5	7
24	16	16	16	4 42 3
25	43 4	12 44 4	12 45 4	12 7
26	16	16	16	7
27	13 46 3	13 47 3	13 48 3	7
28	16	16	16	7
29	49 2	14 50 2	14 51 2	14 52 5 2
30	16	16	16	7
31	15 53 1	15 54 1	15 55 1	7
32	16	16	16	7
33	16	16	16	7

Lot Number	Bay Assignment	Lot Number	Bay Assignment
1	1	34	3
2	2	35	4
3	3	36	1
4	4	37	2
5	1	38	3
6	2	39	1
7	3	40	2
8	1	41	3
9	2	42	4
10	3	43	1
11	4	44	2
12	1	45	3
13	2	46	1
14	3	47	2
15	1	48	3
16	2	49	1
17	3	50	2
18	4	51	3
19	1	52	4
20	2	53	1
21	3	54	2
22	1	55	3
23	2		
24	3		
25	1		
26	2		
27	3		
28	4		
29	1		
30	2		
31	3		
32	1		
33	2		

For all of the results presented in this thesis, we have used the “deterministic factory” heuristic. Since there were a large number of runs, we found this to be the most practical due to its repeating structure. Moreover, the implementation of the “average processing rate” heuristic was tested for a set of incoming lots and found to have very similar behavior to that of the “deterministic factory” heuristic. Bay assignments up to the 55th lot for both the "deterministic factory" and "average processing time" heuristics are shown in Appendix B.2.

### **Creating a Lot-Start Schedule for the Post-Constraint Model**

To create a schedule of lot-starts to input into the post-constraint model, the completion times from each of the stage 5 bay models were compiled and ranked chronologically. This then became the lot-start input schedule for the post-constraint model.

### **Summary of Model 2 Assumptions**

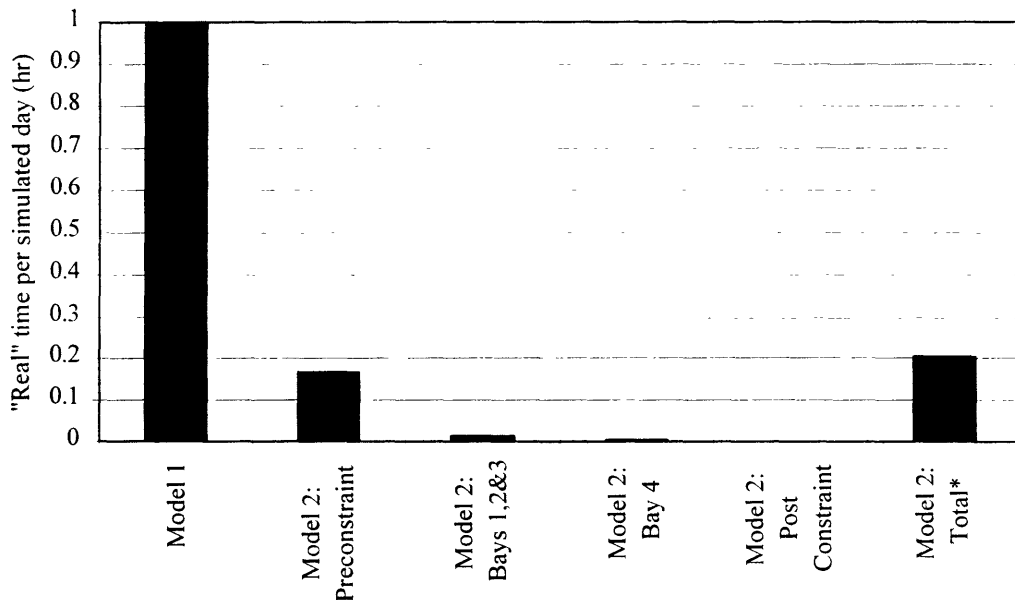
As in *model 1*, we did not include operators or shift effects (e.g. operator end of shift breaks, etc.). We have also assumed use-based exponential downtimes and uptimes. Since the model was broken down into pre-constraint, constraint, and post-constraint models, a heuristic was used to create a schedule of lot starts for each of the stage 5 bay models. This lot start schedule was created a-priori, through a predictive heuristic which assumed deterministic behavior of perfectly reliable stage 5 machines.



#### 4.2.3.4 Model 2 results

One major advantage of *model 2* over *model 1* was the simulation run time. As can be seen in Figure 4-10, the breakdown of the model vastly reduced model run times. This figure shows runtimes for each of the segments of *model 2*. On the y-axis is the runtime, expressed as number of “real-time” hours per simulated day. As can be seen, the *model 2* total is roughly one-fifth of the *model 1* total. It should be pointed out, nevertheless, that the total run times for *model 1* and *model 2* cannot be directly compared for several reasons. In particular, there is some effort that goes into taking the pre-constraint completion times, applying the heuristic to obtain a lot-start schedule for each of the

Figure 4-10: Simulation run-time comparison for models 1 and 2.

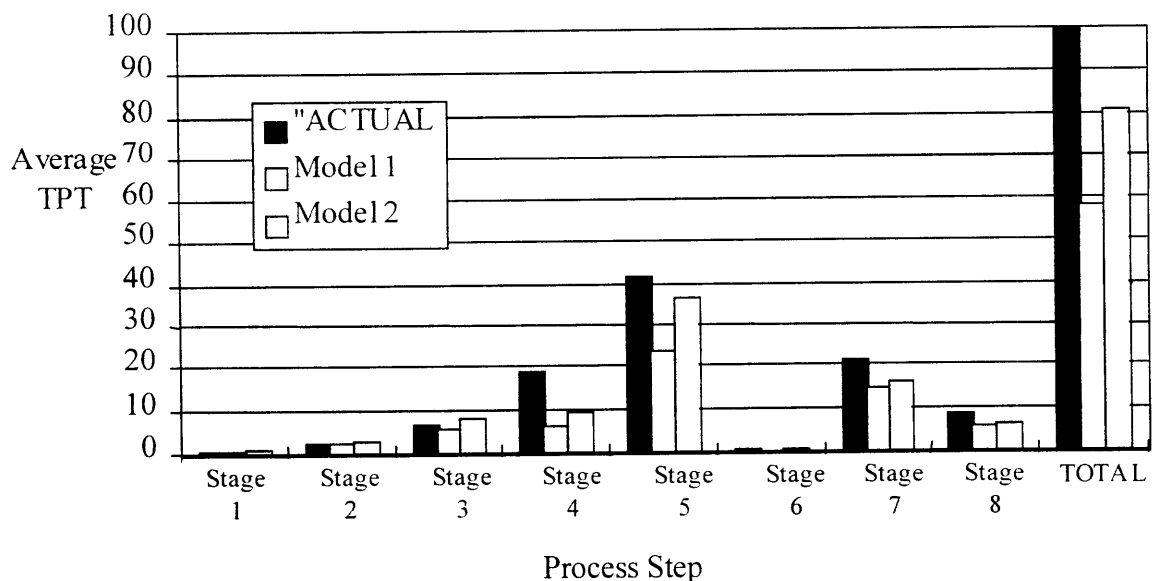


\*The total for *model 2* was calculated as the addition of the run times of the subparts

stage 5 bay models, and taking the completion times of each of the stage 5 bay models to create lot-start schedule for the post-constraint model. This effort was not quantified and would add some time to the *model 2* simulation time. Nonetheless, the *model 2* time could also be reduced by running each of the four stage 5 bay models simultaneously (on four different machines). Moreover, the heuristic used for creating lot-start schedules for the constraint and post-constraint models could be automated, making the additional effort trivial.

In terms of average TPT, *model 2* also yielded more accurate predictions. Figure 4-11 shows the step-by-step and total average TPTs of *model 2* compared to *model 1* and factory floor. *Model 2* brought the average TPT to 80% of actual, some 20% more than

Figure 4-11: *Model 2* average TPT compared to *model 1* and actual



*model 1* (refer back to Figure 4-6). This increase can be attributed to the use of an exact lot-start schedule, which also provided for a higher average factory volume than the lot start schedule used in *Model 1*. As can be seen, all of the step average TPTs increased, particularly for the stage 5 process. The average TPT is most underpredicted, proportionally, at stage 4. This will be explored further in models 3 and 4.

#### **4.2.3.5 Model 3**

Since the model decomposition greatly reduced run times, this allowed the addition of details. In *model 3*, operators and shift effects were added at stage 5. As discussed earlier, each stage 5 bay is manned by multiple operators. The operator tasks are to transport WIP carts from the stage 5 general queue into their respective queue, load and unload the boats to and from the machine's input and output platforms, and inspect the finished parts. Time studies were performed to determine the average transport, loading and inspection times.<sup>11</sup> Operators were included as resources in the *operator* DES file and the processing times were added in the *routes* file.

In addition, shift schedules and shift effects were modeled. The assembly floor generally runs in twelve hour shifts, with one set of operators for each shift. A typical shift schedule is shown in

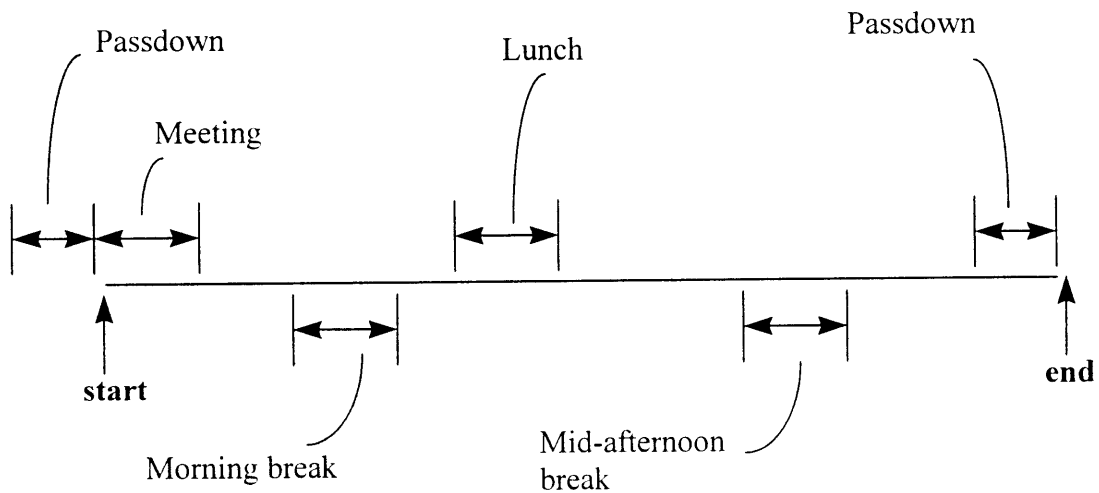
Figure 4-12. The transition between shifts is done through what are referred to as

---

<sup>11</sup> Confidentiality restricts publication of these figures.

*passdowns*. During the passdown period the operators from the ending shift meet with the incoming operators to discuss any emergencies, machines which have been problematic, and related issues. Since the operators cannot supervise the machines during this period, the general rule is that additional boats are loaded *only* if remaining shift time is sufficient to process.

Figure 4-12: Typical shift schedule



Next come *meetings*, during which the shift managers speak to the operators and stretching and relaxation exercises are performed. During this period the stage 5 bays are completely unstaffed, so all operations are paused.<sup>12</sup> Upon returning to the machines after the meetings, the operators can restart the machines exactly where they left off.

---

<sup>12</sup> The stage 5 machines have a “pause” feature which allows operator to stop the machine at any time during the stage 5 operation without having to unload any parts or backtrack on any work.

In addition, there are multiple breaks throughout a shift. In particular there are morning and midafternoon breaks, as well as a lunch break. During breaks the operator is unavailable for loading, inspection or transporting. Since breaks for any two operators in a given bay are never taken simultaneously, machines are kept running. These shift rules were implemented through the *calendar* input file.

### **Summary of Model 3 Assumptions**

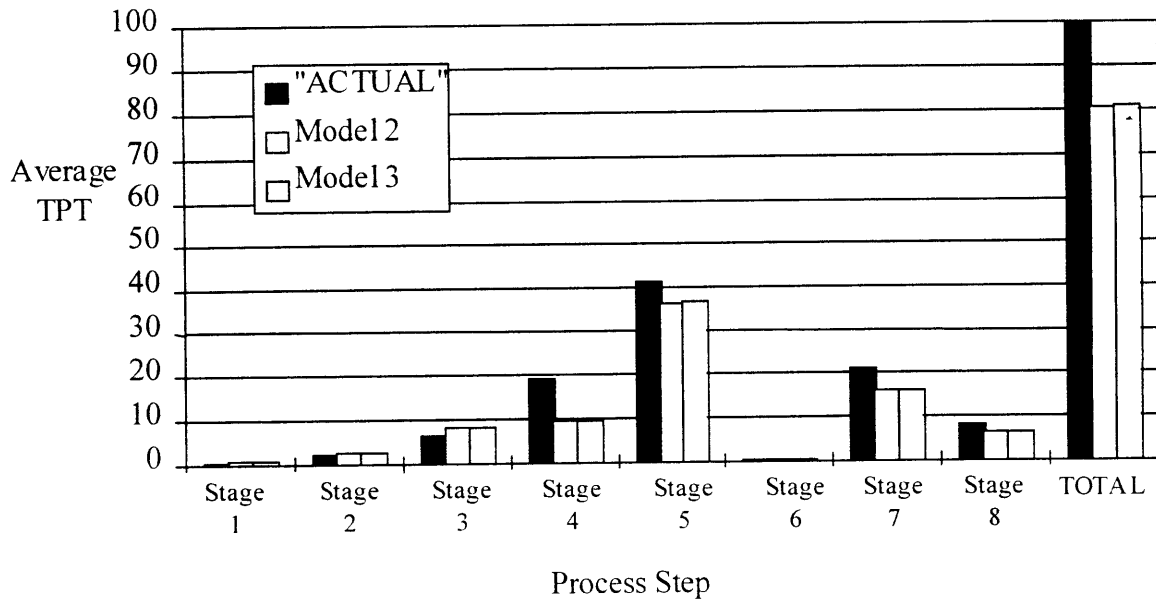
All of the assumptions made for *model 2* were also made in this model, with a few notable exceptions. In particular, operators, shift effects and transport times at each of the stage 5 models were included. There were no changes made to the lot-starts, or machine characterization parameters.

#### **4.2.3.6 Model 3 Results**

The step-by-step average TPTs for *model 3*, along with the *model 2* and actual TPTs are shown in Figure 4-13. Surprisingly, the addition of operators and shift effects at stage 5 had a minor effect on average TPT (recall that operators and shift effects were added *only* at stage 5). This can be attributed to the fact that through lunch and other breaks machines are seldomly starved, since breaks are never taken simultaneously.

In addition, the input platforms have sufficient capacity for the machines to be left running without loading for extended periods of time. These issues were further explored through a series of sensitivity studies, which are discussed in the next section.

Figure 4-13: Average Step TPTs for *model 3*



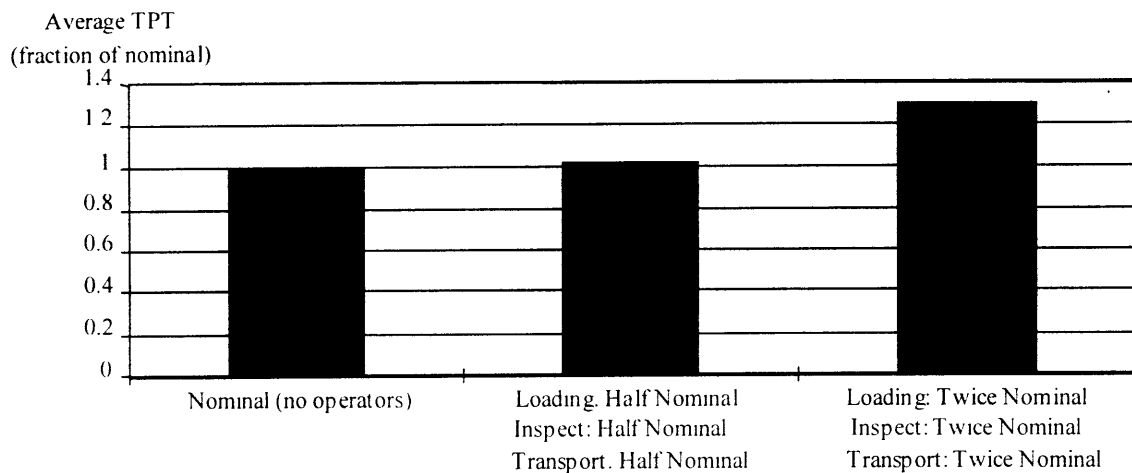
#### 4.2.3.7 Sensitivity Studies

The model decomposition reduced model run times and thus afforded the inclusion of other factors. Before model decomposition, operators and shift effects had been excluded. In addition, the accuracy of machine uptime and downtime distribution parameters was in question. This motivated a series of sensitivity studies to assess the importance of operators, transport times, and machine uptime and downtime distribution parameters. All of these sensitivity studies were performed at the stage 5 bay model, which, being the constraining process, is likely to have the greatest impact on average TPT.

## Operator Sensitivity Study

Three separate simulations were run. The times required for each of the operator activities were run at nominal, half of nominal, and twice nominal. In the first run, no operators were included (essentially this is the same stage 5 model that was used in *model 2*). In the second run, operators were included and the “measured” (i.e. thought be actual) operation times were used (these are also the settings used in *model 3*). In the third run, each of these times was doubled. The results from these runs is shown in Figure 4-14.

Figure 4-14: Operator sensitivity study results



As can be seen, the average TPT is not significantly affected until a “critical” time to perform a function is reached. In particular, since each stage 5 machine has an input queue for up to three magazines, and the operator breaks are never taken simultaneously, the machines can undergo some time period without operators before they are starved. When this “critical” time is reached, the machines begin to experience some starvation and thus the TPT increases.

### **Sensitivity to Uptime and Downtime Distribution Parameters**

There is still a great deal of research to be done in the area of characterizing machine uptimes and downtimes. Due to the lack of factory floor data, the distributions for MTTF and MTTR are often unknown. For this reason, the modeler is often forced to assume some probability distribution. Even after assuming a distribution, the statistical significance of distribution *parameters* is often unknown. Thus, the lack of factory floor data leads to both a *structural* uncertainty (the probability distributions) and a *data* uncertainty (the parameters for these distributions). If the factory being modeled contains recently acquired equipment or uses cutting-edge technology, the problem is further exacerbated by the lack of historical performance data. The lack of data forces the modeler into assuming a distribution based on theoretical or academic field work, and making “guesstimates” for the distribution parameters.

This uncertainty motivated a sensitivity study on MTTR and MTTF at stage 5. Six separate simulations were run. The values used for MTTF and MTTR for each of these runs are shown in Table 4-4. The “nominal” values were used in run 3. For the other runs, values of one-half and twice nominal MTTR were used. These were combined with MTTF values set at nominal and one-tenth of nominal.



Table 4-4: Settings for MTTR and MTTF sensitivity studies

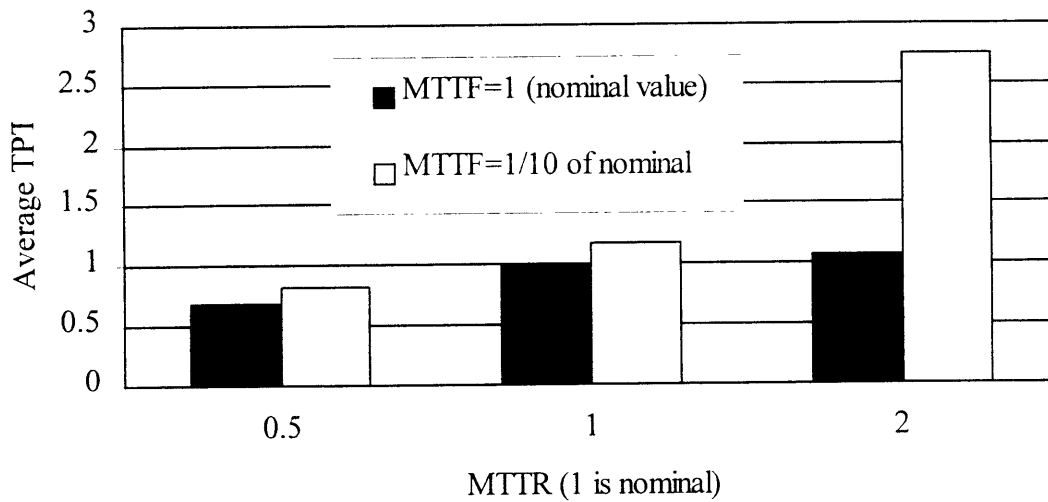
Simulation-->	Run 1	Run 2	Run 3*	Run 4	Run 5	Run 6
MTTF (CPUs)	x	x/10	x	x/10	1x	x/10
MTTR (CPUs)	y/2	y/2	y	y	2y	2y

\* The values for Run 3 are the nominal or “best estimate” values.

NOTE: x and y have no relation to previous references

The results from this study are shown in Figure 4-15. Values for MTTR are shown on the x-axis, with 1 representing nominal. On the y-axis is average TPT. For each value of MTTR, there is one column for MTTF at nominal value, and MTTF at one-tenth of nominal, as shown in the legend. As can be seen, the changes in TPT are most drastic when MTTR is at twice nominal, whereas at MTTR of half nominal there is only about a 15% jump in average TPT for an MTTF of one-tenth nominal. This jump increases as the MTTR increases. Looking at the graphs for the MTTF values separately, we see that at the nominal MTTF there is only a moderate increase in average TPT with increasing MTTR. At one-tenth of nominal MTTF, however, we see that the average TPT increases drastically as MTTR is increased. This result is intuitive: the time to repair is only a significant factor if the machine fails often. If the machine seldomly fails, then average TPT will be relatively insensitive to the time to repair.

Figure 4-15: Average TPT for MTTF and MTTR sensitivity study



#### 4.2.3.8 Model 4

This model had the same features included in *model 3* with a few changes. More specifically, this model changed the values for the failure parameter MTTF. In models 1 through 3, failures were modeled with use-based exponential MTTF distributions. The MTTF values are usually obtained from factory floor personnel, who record the date and time at which each failure occurs through a logbook or calendar.<sup>13</sup> MTTFs are typically calculated as the average “calendar” time elapsed between subsequent failures. In reality, classical failure analysis shows that failures are *operation*, rather than *time* dependent; i.e.

---

<sup>13</sup> This is under the best case scenario. Often such data is not recorded at all, or recorded selectively. Moreover, the modeler seldomly has access to raw data and is only given an average or summarizing statistic.

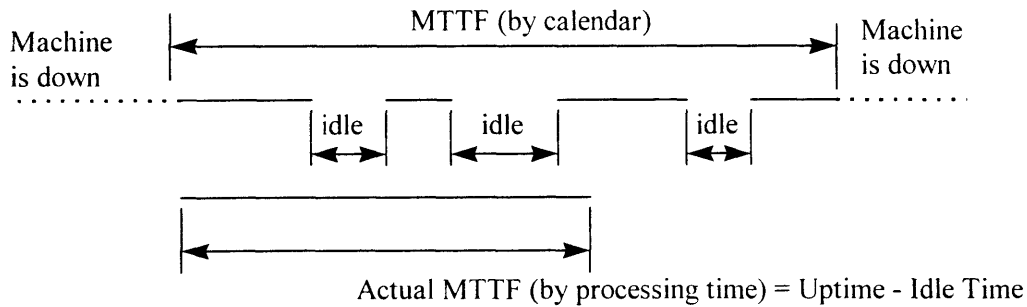
machines fail only while operating on a part.<sup>14</sup> Although there are some exceptions to this failure mode, it is applicable to a vast majority of manufacturing processes [8]. Since machines only fail while operating on a part, MTTFs should only include operational time, disregarding idle times due to blockage and starvation. To reduce ambiguity, the term "mean units [processed] between failures" (MUBF) is often used, since it necessarily implies a use-based failure parameter distribution.

In sum, the problem with models 1 through 3 is that a time-based MTTF was being used to describe a use-based distribution. The use of MTTF rather than MUBF as a failure parameter translates into failures occurring less often in the simulation than in reality, thus leading to a smaller TPT. This is illustrated in Figure 4-16. This makes sense conceptually, since machines which are not used very often tend to fail less often than those that are heavily used. Thus, it is more sensible to use failure parameters that are use-based rather than time based.

---

<sup>14</sup> Thorough studies of *operation dependent failures* are cited by Gershwin [8], including Buzacott and Hanifin [3].

Figure 4-16: Illustration of time-based versus use-based failure parameters



To correct this error, MUBF were used as failure distribution parameters for the constraint and post-constraint processes (due to lack of utilization data for the pre-constraint processes, these were MTTF were left unchanged). MUBFs were estimated by multiplying “calendar” MTTF by machine utilization and dividing by the machine processing rate, i.e.

$$\text{MUBF} = (\text{MTTF} * \text{Machine Utilization}) / \text{Processing Time}$$

#### Summary of Model 4 Assumptions

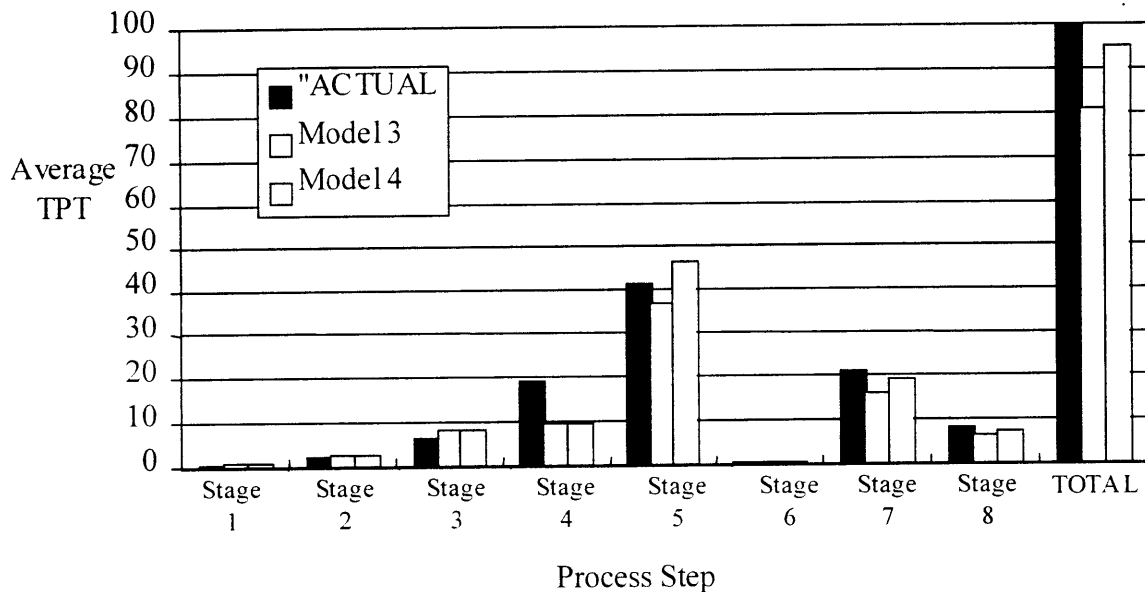
The same assumptions made in *model 3* were used. Operators and shift effects were included based on “observed” values. MTTF and  $\text{MTTF}_A$  were converted to MUBF and  $\text{MUBF}_A$  (recall that we use the subscript *A* to indicate *assist* parameters) at the constraint and post-constraint processes.

#### 4.2.3.9 Model 4 Results

The average TPT obtained for *model 4* was nearly 95% of actual. The step-by step average TPTs are shown in Figure 4-17. As can be seen, there were significant TPT

increases at the stage 5 and post-stage 5 processes (the pre-stage 5 TPTs are the same because *model 4* did not change anything in the pre-constraint model). The most interesting result is that the average stage 5 TPT from the simulation was higher than actual. The reason for this overprediction is explored in the next section. The largest TPT discrepancy for *model 4* was at stage 4, where average TPT was underpredicted by roughly 50%. The other process steps had TPTs within 80% of actual.

Figure 4-17: Step-by-step average TPTs for *model 3*, *model 4* and actual.



### 4.3 Model Validation

To try to validate the *model 4* results, especially in obtaining a plausible cause for the overprediction at stage 5, a few hypotheses were formed. A look at the factory floor record for the stage 4 processing time revealed that lots remain in machines long after they are processed. Lots appear to be “bumped” from machines, rather than removed upon process completion (as is the case in the simulation). To verify this hypothesis, arbitrary delays were added at stage 4. Two simulations were run, one with an arbitrary delay of  $x$  at stage 5, and one with a delay of  $1.5x$  at stage 5.<sup>15</sup> Each delay had two components: One which was added as processing time, and one which was added as a lot holding time.<sup>16</sup> The delay was allocated equally for these two categories.

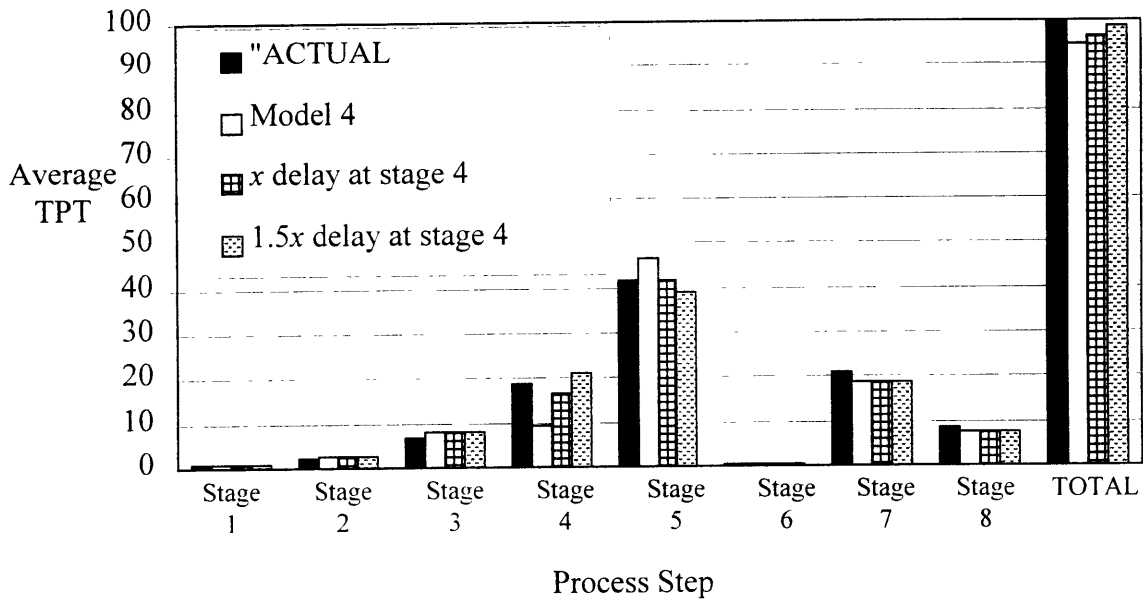
The results from these experiments are shown in Figure 4-18. As can be seen, arbitrary delays at stage 4 had the effect of reducing stage 5 TPT. This makes sense, since delays effectively add time to the processing at stage 4 and reduce queue time at stage 5 (i.e. rather than spending the time in queue, the lots spend time in the stage 4 machines).

---

<sup>15</sup> Confidentiality restricts the publication of these figures.

<sup>16</sup> To truly test this hypothesis, a procedure would be added to the simulation where lots are actually “bumped” from the machines (as was hypothesized). Due to lack of time, this procedure was not used.

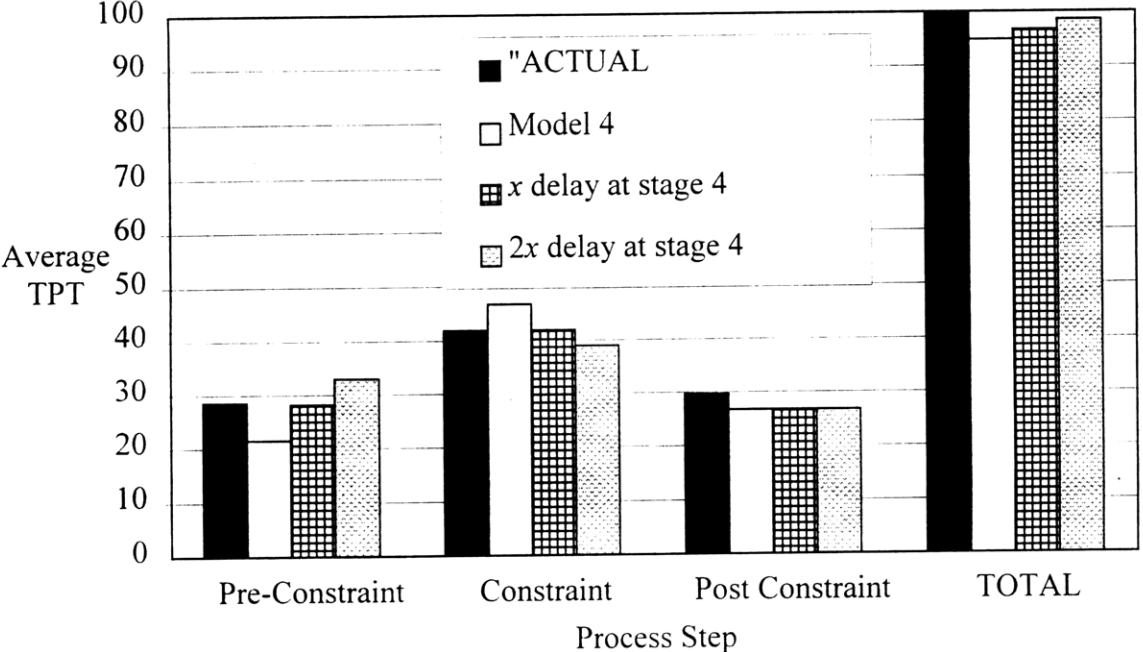
Figure 4-18: Average TPT for *model 4* and arbitrary delays at stage 4



This also explains the “over-estimation” of average TPT in *model 4*. Whereas in *model 4* the lots were immediately sent to the stage 5 queue upon processing, the lots in these experiments experience a delay before being sent to the stage 5 queue. When they do arrive at stage 5 queue, the queue is shorter than usual, since during the time they are held at stage 4, the stage 5 machines are busy processing lots. Due to infrequent starvation at stage 5, the overall TPT increases only slightly. This indicates that detail at pre-constraint processes does not significantly increase total average TPT.

This phenomenon is further elucidated by Figure 4-19. This figure shows the average TPT for the pre-constraint, constraint, and post-constraint sections of the model.

Figure 4-19: Average TPT for *model 4* and arbitrary delays for pre-constraint, constraint and post-constraint sections of model



As can be seen, the increase in average TPT at the pre-constraint model is accompanied by a decrease in average TPT at the constraint, thus leaving the total TPT the same (note that the post-constraint average TPT is virtually unaltered).



## 4.4 Conclusions

### 4.4.1 Results Summary

The results of these analyses can be summarized as:

1. The primary explanatory factors are factory volume and use of operation-dependent failure parameters.
2. Shift effects and operators *do not* significantly contribute to average TPT.
3. Given that the constraint is not starved, detail at pre-constraint processes only marginally adds to total average TPT. Thus, most significant impact will be given by detail at constraint and post-constraint processes.
4. There are limitations in determining accuracy of models since reliability of validation data is unknown.

### 4.4.2 Recommendations for Future Action

We make these recommendations based on barriers encountered during the research process. The *validation* of models proved equally challenging to the *creation* of accurate models. Recommendations for future action are summarized below:

1. Breaking up system into pre-constraint, constraint, and post-constraint models is a cost effective--and perhaps necessary--method to accurately model the system's constraint. It also makes the study of sensitive factors more affordable.
2. The reliability of validation data is yet to be assessed. This is a necessary step for robust model validation.

3. Although expensive, independent data gathering efforts may be necessary for statistically significant and reliable factory floor data. The type of data gathered by industrial engineers does not coincide with that needed for accurate DES models and/or is often difficult to obtain. This provides a barrier in obtaining both model input data and validation data which is reliable. Some key examples of input data are machine uptime and downtime distributions and WIP management rules, for which there is a dearth of *valuable* data.<sup>17</sup>

---

<sup>17</sup> In particular, we found it extremely difficult to find distributions for data, and rather were often given crude averages.

## Chapter 5

# Analytical Approximations: Series-Parallel Flow Model

While it was demonstrated that simulation can indeed predict average TPTs which are very close from factory floor, they required significant effort. For this reason, it is of interest to develop analytical models which can provide “reasonable” results at only a fraction of the effort.<sup>1</sup> In this section we discuss analytical approaches for modeling a series-parallel flow line, such as the one presented in the case study. We will use this analytical model to obtain estimates for average TPT, which we will compare to DES and factory floor results.

---

<sup>1</sup> There are many factors which go into the "total" cost of models. DES, for instance, often requires significant training and specialized knowledge which goes far beyond the knowledge required for analytical models. The author spent a substantial period of time learning DES modeling to arrive at the results presented in Chapter 4. It is hoped that the technically oriented reader will be able to understand and apply the analytical models presented here at a fraction of the time.

## 5.1 Modeling Strategy

The modeling of a series-parallel flow line will be made in three steps. First, we will analyze a simple two-machine single buffer system. This will serve as the building block for describing larger and more complex systems. Second, we will use a decomposition technique to describe the performance of series transfer lines with more than two machines. Lastly, we will use a *method of moments* estimation technique to describe series-parallel flow lines as simple series lines, to which we can then apply the decomposition technique.<sup>2</sup>

## 5.2 Two Machine Transfer Line

A two machine single-buffer transfer line is the most elemental non-trivial transfer line which can be analyzed. Although these systems are quite simple, they provide powerful building blocks for the analysis of larger systems.

### 5.2.1 System Description

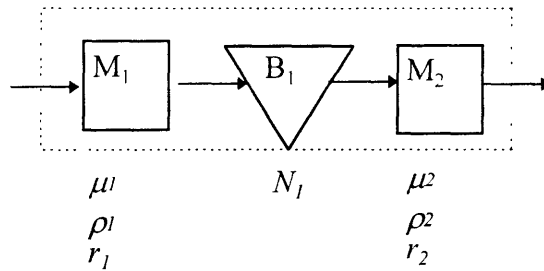
A schematic of a two machine single buffer transfer line is shown in Figure 5-1. In this model, each of the machines is described in terms of three parameters  $\mu$ ,  $\rho^3$ , and  $r$ , and each buffer is described in terms of  $N$ , where these symbols have the following definitions:

---

<sup>2</sup> The two-machine transfer line analysis comes from Gershwin [8]. The decomposition and series-parallel model come from Burman [2].

<sup>3</sup> This is notated  $p$  in [8].

Figure 5-1: Two-machine single-buffer transfer line



$\mu_i =$  { The processing rate of machine  $i$ . Therefore,  $\mu_i \delta t$  is the amount of material processed in one time unit.

$\rho_i =$  { The "failure rate" of machine  $i$ . Therefore,  $\rho_i \delta t$  is the probability that machine  $i$  will fail in the next time unit given that it is currently up and not blocked or starved by any other machine.

$r_i =$  { The "repair rate" of machine  $i$ . Therefore,  $r_i dt$  is the probability that machine  $i$  will be repaired in the next time unit given that it is currently down

$N_i =$  The capacity of buffer  $i$  ( in this case  $i=1$ , so we will refer to this as simply  $N$ )

## 5.2.2 Model Assumptions

For clarity, we will list all explicit assumptions made in this model:

- Machine processing rates are deterministic and non-homogenous. By *deterministic* it is meant that the processing time for *each machine* is constant and does not vary.<sup>4</sup> By *non-homogenous*, it is meant that the machines in the transfer line have *dissimilar* processing rates.

---

<sup>4</sup> In practice, the *deterministic processing time* assumption is made when the variance in processing time is very small, i.e. the processing time distribution is very "tight" in comparison with the variance of queuing times and other factors.

- Machines are unreliable with operation dependent failures (ODF). The ODF assumption has some rather profound consequences in analytical models. In particular, ODF implies that *machines cannot fail while they are blocked or starved*.
- The length of failures is exponentially distributed based on use (ODF).
- The length of repairs is exponentially distributed based on time.
- Failure and repair times are independent.
- Yield on all steps of the manufacturing process is 100%.
- The flow of material through the transfer line is *continuous*.<sup>5</sup> Thus, the buffer levels take on continuous values (not integers). We take this to be a reasonable approximation due to the large volume (in terms of number of parts) processed by a typical assembly factory.
- The first machine has an infinite reservoir of parts of parts to operate on. For this reason, the first machine is never starved.
- The last machine deposits parts in an infinite sink (or an infinitely large buffer). For this reason, the last machine is never blocked.

### 5.2.3 Analysis

An analytical description of this system has been developed by Gershwin [8].<sup>6</sup> This flow line can be modeled as a continuous time, mixed state Markov process. We can determine the probabilities that the system will go from one state to another and based on

---

<sup>5</sup> Similar analytical models which have discrete buffer levels and non-deterministic processing rates can be found in [8].

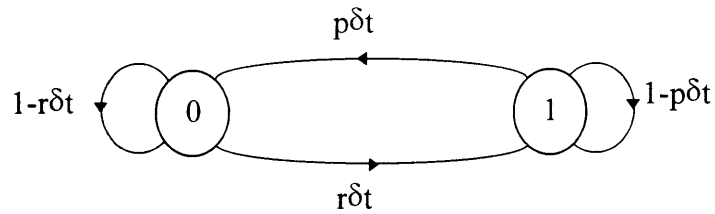
<sup>6</sup> In this section we only present a brief derivation of the needed equations. For a more comprehensive treatment, refer to [8].

this determine steady-state probabilities for each of the states. The average throughput and average WIP of the system can then easily be expressed in terms of the steady state probabilities and the performance of the system in each of the states. The state of the system at any instant in time can be described in terms of three parameters:  $x$ ,  $\alpha_1$ , and  $\alpha_2$ , where  $x$  is the buffer level,  $\alpha_1$  is the status of machine 1 (either working or under repair), and  $\alpha_2$  is the status of machine 2 (either working or under repair). For notational convenience, we let  $\alpha_i$  assume a value of 0 if machine  $i$  is down, and 1 if machine  $i$  is up. Thus we will use the three dimensional vector  $\vec{s}(t)=(x(t), a_1(t), a_2(t))$  to describe the state of the two machine system at time  $t$ , where  $0 \leq x \leq N$  and  $\alpha_i=0$  or 1 depending on whether machine  $i$  is up or down, respectively. We will usually drop the  $t$  argument for ease of notation.

To illustrate our approach, we will estimate the average throughput of a single machine. The Markov chain for the status of a single machine is shown in

Figure 5-2. If the machine is currently down (in state 0), it has a probability of  $r\delta t$  of being repaired, and a probability of remaining in the failed state of  $1 - r\delta t$  within the next  $\delta t$ . If the machine is currently up (in state 1), it has a probability of  $\rho\delta t$  of failing, and a probability of  $1 - \rho\delta t$  of not failing within the next  $\delta t$ . These probabilities are referred to as the *state transition probabilities*. The steady state probabilities for this system are  $\mathbf{p}(\alpha=0) = \rho/(r+\rho)$  and

Figure 5-2: Markov process for the status of a single machine



$\mathbf{p}(\alpha=1) = r/(r+\rho)$  (we will use the boldface  $\mathbf{p}$  to denote a probability). Thus, the long-term average throughput is the probability that the machine is up times the throughput when the machine is up,  $\mu r/(r + \rho)$ . The term  $r/(r + \rho)$  is often referred to as the *isolated efficiency*,  $e_i$ , of machine  $i$ . We will apply this same concept to a two machine transfer line by describing each of the steady-state probabilities  $\mathbf{p}(x, \alpha_1, \alpha_2)$ , and then calculating the average, or *expected value*, of the throughput, which is simply the summation of the expected throughput of each of the steady states (i.e. the state's long term probability times the throughput when the transfer line is in that state). Similarly, we can find the average WIP (or buffer level). The average TPT, can then be easily calculated by Little's Law (average TPT = [average WIP]/[average throughput]). It is important to realize that in this analysis we are not concerned with transient states, since by definition, these are states which have zero steady-state probability (i.e. transient states cannot be reached by any *recurrent* or non-transient states).

### 5.2.3.1 Description of Possible States

In a two-machine transfer line, for any given buffer level  $x$ , there are at most four possible



sets of states of the form  $(x, \alpha_1, \alpha_2)$ , namely  $(x, 1, 1)$ ,  $(x, 1, 0)$ ,  $(x, 0, 1)$ , and  $(x, 0, 0)$ <sup>7</sup>. Furthermore, we can separate these into *boundary states* where  $x = 0$  or  $x = N$ , and *internal states* where  $x$  assumes values strictly between 0 and  $N$ . These are important distinctions, since the internal behavior of the system can be described by probability *density* functions (i.e., there is a zero probability of the buffer level assuming an *exact* value), while there may be a nonzero probability (i.e. probability *mass*) of finding the system in a given boundary state. For example, if  $\mu_1 < \mu_2$ , the level of storage will tend to decrease if both machines remain operational and the buffer will eventually empty. At this point, the system will be at state  $(0, 1, 1)$ , where it will remain until one of the machines fails. If the machines are fairly reliable, this could be a non-trivial amount of time. This behavior clearly differs from internal states such as  $(x, 1, 1)$ ,  $0 < x < N$ , because there are no states which are very much more likely to be reached than other states or at which the system will spend a non-trivial amount of time. We will exploit this observation to structure the transition equations into *intermediate* and *boundary* equations.

### 5.2.3.2 Intermediate Buffer Levels

Here we derive transition probabilities for each of the internal states of the form  $(x, \alpha_1, \alpha_2)$ ,  $0 < x < N$ . We will look at the probability of a state occurring at time  $t + \delta t$  in terms of possible states at time  $t$ . For example, the probability of finding both machines

---

<sup>7</sup> We say "at most," because some of these states are transient (cannot be reached by any state with nonzero long-term probability) and can thus be ignored.

operational with a storage level between  $x$  and  $x+\delta x$  at time  $t+\delta t$  is given by  $f(x,1,1,t+\delta t)\delta x$  where

$$(5.1) \quad f(x,1,1,t+\delta t) = (1-(\rho_1+\rho_2)\delta t)f(x-\mu_1\delta t+\mu_2\delta t,1,1,t) + r_1\delta t f(x+\mu_2\delta t,0,1,t) + r_2\delta t f(x-\mu_1\delta t,1,0,t) + o(\delta t)$$

This equation was obtained by exhaustive enumeration of the possible states at time  $t$ , and the probability that the system evolved from that state to the current state. More specifically:

- If both machines were up at time  $t$ , then  $x-\mu_1\delta t+\mu_2\delta t$  was the buffer level at that time and  $1-(\rho_1+\rho_2)\delta t$  is the probability that neither machine failed in  $\delta t$ .
- If machine 1 was down and machine 2 was up at time  $t$ , then  $x+\mu_2\delta t$  was the buffer level at that time and  $r_1\delta t(1-\rho_2\delta t) = r_1\delta t - r_1\rho_2\delta t^2 \approx r_1\delta t$  (ignoring second order terms) is the probability that machine 1 is repaired and machine 2 stays up in  $\delta t$ .
- If machine 2 was down and machine 1 was up at time  $t$ , then  $x-\mu_1\delta t$  was the buffer level at that time and  $r_2\delta t(1-\rho_1\delta t) = r_2\delta t - r_2\rho_1\delta t^2 \approx r_2\delta t$  (ignoring second order terms) is the probability that machine 2 is repaired and machine 1 stays up in  $\delta t$ .
- Note that if both machines were down at time  $t$ , the probability of both getting repaired would be  $r_2\delta t r_1\delta t = r_1 r_2\delta t^2 \approx 0$  for this first-order model.

For completeness in accountability, we have added the error term  $o(\delta t)$  in equation (5.1).

Since this is a continuous time model, we wish to look at the transition probability at very

small time intervals, that is  $\frac{\partial f}{\partial t}(x,1,1)$  as  $\delta t \rightarrow 0$ . Equation (5.1) becomes:

$$\frac{\mathcal{J}}{\mathcal{A}}(x,1,1) = -(\rho_1 + \rho_2)f(x,1,1) + (\mu_2 - \mu_1)\frac{\mathcal{J}}{\mathcal{A}}(x,1,1) + r_1f(x,0,1) + r_2f(x,1,0) \quad (5.2)$$

where the  $t$  argument is suppressed for ease of notation. Note that the error term vanishes in this limit.

Using this same approach, we obtain the other three equations that describe internal storage behavior:

$$\frac{\mathcal{J}}{\mathcal{A}}(x,0,0) = -(r_1 + r_2)f(x,0,0) + \rho_1f(x,1,0) + \rho_2f(x,0,1) \quad (5.3)$$

$$\frac{\mathcal{J}}{\mathcal{A}}(x,0,1) = \mu_2\frac{\mathcal{J}}{\mathcal{A}}(x,0,1) - (r_1 + \rho_2)f(x,0,1) + \rho_1f(x,1,1) + r_2f(x,0,0) \quad (5.4)$$

$$\frac{\mathcal{J}}{\mathcal{A}}(x,1,0) = -\mu_1\frac{\mathcal{J}}{\mathcal{A}}(x,1,0) - (\rho_1 + r_2)f(x,1,0) + \rho_2f(x,1,1) + r_1f(x,0,0) \quad (5.5)$$

### 5.2.3.3 Boundary Behavior

There are eight boundary states, four states where  $x=0$  and four states where  $x=N$ , (recall  $N$  is the buffer capacity). We will refer to the former as *lower boundary* states and the latter as the *upper boundary* states. As discussed previously, there may be a nonzero probability of finding a state in a given boundary state. For each of these boundaries, we will examine three types of transitions: boundary-to-boundary, interior-to-boundary, and boundary-to interior. Since it is not clear a-priori which equations fall under each of the categories described above, we will first look at the possibilities for arriving at each of the lower boundary states, and then look at their structure to categorize them and obtain the upper boundary equations.

**Lower Boundary Equations:  $x=0$**

**Arriving at state (0,1,0).** State (0,1,0) is not a possible steady state, since if the system ever reaches this state, it leaves instantly. If the first machine is operational and the second is down, material will instantly accumulate in the buffer. Our first equation is thus,

$$\mathbf{p}(0,1,0) = 0 \quad (5.6)$$

Note that this equation also implies that state (0,1,0) cannot be reached from any interior state  $(x, \alpha_1, \alpha_2)$ . This may be regarded as our first *interior-to-boundary* equation.

**Arriving at state (0,0,0).** If the system is in state (0,0,0) at time  $t+\delta t$ , at time  $t$  it was either at (0,0,0) or (0,1,0). The (0,0,0) to (0,0,0) transition probability is  $(1-(r_1+r_2)\delta t$  (neither of the machines get repaired). The (0,1,0) to (0,0,0) transition probability is  $r_1\delta t$  (machine 1 fails). But recall that state (0,1,0) is not possible. Thus

$$\frac{d}{dt}\mathbf{p}(0,0,0) = -(r_1 + r_2)\mathbf{p}(0,0,0) \quad (5.7)$$

As can be seen, this equation involves only lower boundary states. It is therefore a *boundary-to-boundary* equation.

**Arriving at state (0,0,1).** To arrive at state (0,0,1) at time  $t+\delta t$ , the system may have been in one of four sets of states at time  $t$ . (1) It could have been in state (0,0,0) and then the second machine could have gotten repaired (with probability  $r_2\delta t$ ). (2) It could have

been in state (0,0,1) and then gone for a time interval without repair of the first machine. The second machine could not have failed since it was starved, so this sequence would occur with a probability of  $(1 - r_1\delta)$ . (3) It could have been in state (0,1,1) with the first machine failing and the second not failing. The probability of this sequence is  $\rho_1\delta(1 - \rho_2\delta) = \rho_1\delta - \rho_2\rho_1\delta^2 \approx \rho_1\delta$  (ignoring second order terms). (4) It could have been in any state coming from the *interior* of the form  $(x,0,1)$  where  $0 < x < \mu_2\delta$  if the first machine is not repaired and the second machine does not fail (recall  $\mu_2\delta$  is the amount produced by machine 2 during a small time period). The probability that this sequence occurred is  $\int_0^{\mu_2\delta} f(x,0,1,t)dx$  (recall in the *interior* there is probability density). Adding the probabilities of possibilities (1)-(4) we obtain the total probability of being at state (0,0,1):

$$\mathbf{p}(0,0,1,t + \delta t) = r_2\delta t \mathbf{p}(0,0,0,t) + (1 - r_1\delta t) \mathbf{p}(0,0,1,t) + \rho_1\delta t \mathbf{p}(0,1,1,t) + \int_0^{\mu_2\delta} f(x,0,1,t)dx \text{ or,}$$

$$\frac{d}{dt} \mathbf{p}(0,0,1) = r_2\mathbf{p}(0,0,0) - r_1\mathbf{p}(0,0,1) + \rho_1\mathbf{p}(0,1,1) + \mu_2 f(0,0,1) \quad (5.8)$$

As is evident, it is possible to arrive at this state from the interior (hence the term  $\mu_2 f(0,0,1)$ ). This is therefore an *interior-to-boundary* equation.

**Arriving at state (0,1,1).** To get to state (0,1,1) there are two possibilities. (1) If  $\mu_1 \leq \mu_2$  it is possible to get to (0,1,1) from (0,1,1) if no failures occur with probability  $(1 - \rho_1\delta)(1 - \rho_2\delta) \approx 1 - (\rho_1 + \rho_2)\delta$ ; from (0,0,1) if the first machine is repaired and the second doesn't fail (with probability  $r_1\delta(1 - \rho_2\delta) \approx r_1\delta$ ); and from  $(x,1,1)$  if  $0 < x \leq (\mu_2 - \mu_1)\delta$

and no failures occur with probability  $\int_0^{(\mu_2 - \mu_1)\delta} f(x,1,1,t)dx$ . (2) If  $\mu_1 > \mu_2$  it is not possible to get from  $(0,1,1)$  at  $t$  to  $(0,1,1)$  at  $t+\delta$  if  $\delta$  is small. This is because when both machines are working, material accumulates in the buffer. For the same reason, if the system is in state  $(0,0,1)$  at time  $t$  and the first machine is repaired during  $(t,t+\delta)$ , there will be some material in the buffer at time  $t+\delta$ . Lastly, note that  $(0,1,1)$  can be arrived at from any interior state of the form  $(x,1,1)$ , since if both machines stay up, the buffer level at time  $t+\delta$  will be greater than  $x$  (it *cannot* decrease since  $\mu_1 > \mu_2$ ). Thus for  $\mu_1 > \mu_2$  state  $(0,1,1)$  has steady-state probability of zero. To summarize:

$$\frac{d}{dt} \mathbf{p}(0,1,1) = \begin{cases} -(\rho_1 + \rho_2)\mathbf{p}(0,1,1) + r_1\mathbf{p}(0,0,1) + (\mu_2 - \mu_1)f(0,1,1). & \text{if } \mu_1 \leq \mu_2 \\ 0 \text{ (because } \mathbf{p}(0,1,1) = 0) & \text{if } \mu_1 > \mu_2 \end{cases} \quad (5.9)$$

This equation is also considered an *interior-to-boundary* equation, since this state can be reached from  $(x,1,1)$  if  $0 < x \leq (\mu_2 - \mu_1)\delta$ .

**Arriving at states  $(x,1,0)$  from the lower boundary.** Note that all such transitions are only possible if  $0 < x < \mu_1\delta$ . The only possible internal states which could arrive at  $(x,1,0)$  are  $(0,0,0)$  if the first machine is repaired, or  $(0,1,1)$  if the second machine fails. The probabilities for these transitions (ignoring second order terms as before) are  $r_1\delta$  and  $\rho_2\delta$ , respectively. The probability of being at the state  $(x,1,0)$  within a small buffer level interval  $\delta x = (\mu_1\delta - 0) = \mu_1\delta$  recall ( $0 < x < \mu_1\delta$ ), is the probability density times  $\delta x$  (since the interval  $\delta x$  is very small, the area under the density curve is simply a rectangle of length  $\delta x$  and height  $f(x,1,0) = f(0,1,0)$  since  $x \rightarrow 0$ ). Hence

$$f(x,1,0)\delta x = f(0,1,0) \mu_1 \delta t = r_1 \delta t \mathbf{p}(0,0,0) + \rho_2 \delta t \mathbf{p}(0,1,1)$$

equivalently,

$$\mu_1 f(0,1,0) = r_1 \mathbf{p}(0,0,0) + \rho_2 \mathbf{p}(0,1,1) \quad (5.10)$$

This is our second *boundary-to-interior* equation.

**Arriving at states  $(x,1,1)$  from the lower boundary.** Note that it is only possible to reach such states from the boundary if  $0 < x \leq (\mu_1 - \mu_2)\delta t$ , where  $\mu_1 > \mu_2$ . The only boundary state which can reach such a  $(x,1,1)$  is  $(0,0,1)$  if the first machine is repaired.

Using the same technique as in the derivation of (5.10), we obtain:

$$(\mu_1 - \mu_2)f(0,1,1) = r_1 \mathbf{p}(0,0,1) \text{ if } \mu_1 > \mu_2 \quad (5.11)$$

This is our third *boundary-to-interior* equation.

**Arriving at states  $(x,0,1)$  and  $(x,0,0)$  from the lower boundary.** Since both of these states have machine 1 in a failed state, these are not possible transitions. For example,  $(x,0,1)$  can clearly not be arrived at from the interior. It cannot have arrived from  $(0,1,1)$  since if the first machine fails within the next  $\delta t$  this would empty out the buffer. It could not have arrived from any  $(0,0,\alpha_2)$  since the presence of material at the buffer implies machine 1 *must* have been up some  $\delta t$  earlier. The only left possible state is  $(0,1,0)$ , which is transient and thus does not exist in the steady state. The same arguments apply to the arrival at  $(x,0,0)$ .

Thus, we have successfully described all possible transitions at the lower boundary.

Almost exactly analogous phenomena occur at the upper boundary. For this reason transition equations for the upper boundary will only be stated and not derived here.

**Upper Boundary Equations:  $x=N$**

Boundary-to-Boundary Equation:

$$\frac{d}{dt}\mathbf{p}(N,0,0) = -(r_1 + r_2)\mathbf{p}(N,0,0) \quad (5.12)$$

Interior-to-Boundary Equations

$$\frac{d}{dt}\mathbf{p}(N,1,0) = r_1\mathbf{p}(N,0,0) - r_2\mathbf{p}(N,1,0) + \rho_2\mathbf{p}(N,1,1) + \mu_1 f(N,1,0) \quad (5.13)$$

$$\frac{d}{dt}\mathbf{p}(N,1,1) = \begin{cases} -(\rho_1 + \rho_2)\mathbf{p}(N,1,1) + r_2\mathbf{p}(N,1,0) + (\mu_1 - \mu_2)f(N,1,1), & \text{if } \mu_1 \geq \mu_2 \\ 0 \text{ (because } \mathbf{p}(N,1,1) = 0) & \text{if } \mu_1 < \mu_2 \end{cases} \quad (5.14)$$

Boundary-to-Interior Equations.

$$\mathbf{p}(N,0,1) = 0 \quad (5.15)$$

$$\mu_2 f(N,0,1) = r_2 \mathbf{p}(N,0,0) + \rho_1 \mathbf{p}(N,1,1) \quad (5.16)$$

$$(\mu_2 - \mu_1)f(N,1,1) = r_2 \mathbf{p}(N,1,0) \text{ if } \mu_1 < \mu_2 \quad (5.17)$$

**5.2.4 Solution Technique**

The solutions here are presented but not proved. For any given buffer level,  $\hat{N}$ , it can easily be shown that the solutions presented here are valid by substituting them into the



state transition equations presented in sections 5.2.3.2 and 5.2.3.3. For more comprehensive treatment refer to Gershwin [8].

The obvious choice for coupled ordinary differential equations such as (5.2)-(5.5) is some exponential form. In particular, the following solution satisfies (5.2)-(5.5):

$$f(x, \alpha_1, \alpha_2) = C e^{\lambda x} Y_1^{\alpha_1} Y_2^{\alpha_2} \quad (5.18)$$

which is the general approach to solving differential equations. We can then solve for the constants  $C$ ,  $Y_i$  and  $\lambda_j$  from the following parametric equations:

$$\sum_{i=1}^2 (\rho_i Y_i - r_i) = 0 \quad (5.19)$$

$$-\mu_1 \lambda = (\rho_1 Y_1 - r_1) \frac{1 + Y_1}{Y_1} \quad (5.20)$$

$$\mu_2 \lambda = (\rho_2 Y_2 - r_2) \frac{1 + Y_2}{Y_2} \quad (5.21)$$

Because of the nature of the boundary conditions, the boundary probabilities can be categorized into three cases  $\mu_1 = \mu_2$ ,  $\mu_1 > \mu_2$  and  $\mu_1 < \mu_2$ .

**Case 1:  $\mu_1 = \mu_2$**

In this case, the steady state boundary probabilities are:

$$\mathbf{p}(0,0,1) = C \frac{\mu}{r_1 \rho_2} (r_1 + r_2) \quad (5.22)$$

$$\mathbf{p}(0,1,1) = C \frac{\mu}{\rho_2} \left( \frac{r_1 + r_2}{\rho_1 + \rho_2} \right) \quad (5.23)$$

$$\mathbf{p}(N,1,0) = C \frac{\mu}{\rho_1 r_2} e^{\lambda N} (r_1 + r_2) \quad (5.24)$$

$$\mathbf{p}(N,1,1) = C \frac{\mu}{\rho_1} e^{\lambda N} \left( \frac{r_1 + r_2}{\rho_1 + \rho_2} \right) \quad (5.25)$$

All other steady-state boundary probabilities are zero.

The constant,  $C$ , can be found the following *normalization* equation (*Law of total probability*):

$$\sum_{\alpha_1=0}^1 \sum_{\alpha_2=0}^1 \left[ \int_0^N f(x, \alpha_1, \alpha_2) dx + \mathbf{p}(0, \alpha_1, \alpha_2) + \mathbf{p}(N, \alpha_1, \alpha_2) \right] = 1 \quad (5.26)$$

### Case 2: $\mu_1 > \mu_2$

For both  $\mu_1 > \mu_2$  and  $\mu_1 < \mu_2$ , the interior probability density can be described by:

$$f(x, \alpha_1, \alpha_2) = \sum_{j=1}^2 C_j e^{\lambda_j x} Y_{1j}^{\alpha_1} Y_{2j}^{\alpha_2} \quad (5.27)$$

The  $Y_{ij}$  and  $\lambda_j$  parameters are obtained from (5.19)-(5.21). The  $C_j$  are obtained from the normalization equation (5.26) and from the following equation:

$$\sum_{j=1}^2 C_j Y_{ij} = 0 \quad (5.28)$$

The boundary steady state probabilities follow:

$$\mathbf{p}(0,1,1) = 0 \quad (5.29)$$

$$\mathbf{p}(0,0,1) = \frac{\mu_1 - \mu_2}{r_1} \sum_{j=1}^2 C_j Y_{1j} Y_{2j} \quad (5.30)$$

$$\mathbf{p}(N,1,1) = \frac{\mu_1}{\rho_1} \sum_{j=1}^2 C_j e^{\lambda_j N} Y_{2j} \quad (5.31)$$

$$\mathbf{p}(N,1,0) = \frac{\mu_1}{r_2 \rho_1} (r_1 + r_2) \sum_{j=1}^2 C_j e^{\lambda_j N} \quad (5.32)$$

All other steady-state boundary probabilities are zero.

### Case 3: $\mu_1 < \mu_2$

The analysis for this case is similar. The interior probability density and constants can be determined as in case 1. The boundary steady state probabilities follow:

$$\mathbf{p}(0,0,1) = \frac{\mu_2}{r_1 \rho_2} (r_1 + r_2) \sum_{j=1}^2 C_j \quad (5.33)$$

$$\mathbf{p}(0,1,1) = \frac{\mu_2}{\rho_2} \sum_{j=1}^2 C_j Y_{1j} \quad (5.34)$$

$$\mathbf{p}(N,1,0) = \frac{\mu_2 - \mu_1}{r_2} \sum_{j=1}^2 C_j e^{\lambda_j N} Y_{1j} Y_{2j} \quad (5.35)$$

$$\mathbf{p}(N,1,1) = 0 \quad (5.36)$$

All other steady-state boundary probabilities are zero.

#### 5.2.4.1 Average Throughput

The throughput, denoted by  $P$ , is the rate at which material leaves the second machine.

When the storage is not empty and the second machine is operational, the rate is  $\mu_2$ .

When the storage is empty but both machines are operational, the rate is  $\mu = \mu_1$ .

Otherwise, no material emerges from the system. Over the long term, the throughput can be expressed as the “effective” processing rate of machine 2 times the probability that it

is not starved, namely:

$$P_2 = \mu_2 e_2 (1 - p_S) \quad (5.35)$$

where  $p_S$  is the probability of starvation and  $e_2$  is the isolated efficiency of machine 2,  $r_1/(r_1 + \rho_1)$ . Due to conservation of flow, the production rate of machine 1 is equal to the production rate of machine 2.<sup>8</sup> Thus, the average throughput of the system is also given by the effective processing rate of machine 1 times the probability that it is not blocked:

$$P_1 = P_2 = P = \mu_1 e_1 (1 - p_b) \quad (5.36)$$

Recall that the probability of blocking  $p_b$  is the probability that a machine is unable to operate because its downstream buffer is full. Likewise, the probability of starvation  $p_S$  is the probability that a machine is unable to operate because its upstream buffer is full. These probabilities are given by the following equations:

$$p_b = \mathbf{p}(N, 1, 0) + \left(1 - \frac{\mu_2}{\mu_1}\right) \mathbf{p}(N, 1, 1) \quad (5.37)$$

$$p_S = \mathbf{p}(0, 0, 1) + \left(1 - \frac{\mu_1}{\mu_2}\right) \mathbf{p}(0, 1, 1) \quad (5.38)$$

#### 5.2.4.2 Average WIP

The average WIP is given by:

$$\bar{x} = \sum_{l=1}^2 C_l \left(\frac{1 + Y_{1,l}}{\lambda_l}\right) \left(\frac{1 + Y_{2,l}}{\lambda_l}\right) [e^{\lambda_l \Lambda} (\lambda_l N - 1) + 1] + N(\mathbf{p}(N, 1, 0) + \mathbf{p}(N, 1, 1)) \quad (5.39)$$

where the constants and probabilities are given by equations (5.18)-(5.36), as applicable.

---

<sup>8</sup> This fact is fairly intuitive but not proved in this thesis. A rigorous proof can be found in [8].

### 5.2.4.3 Average TPT

The average TPT can be found by Little's Law, using the average throughput and WIP described in sections 5.2.4.1 and 5.2.4.2. Restating equation (3-2) with the notation used in this chapter:

$$\overline{TPT} = \frac{\bar{x}}{P} \quad (5.40)$$

## 5.3 Decomposition Method for Multiple Server Serial Transfer Line

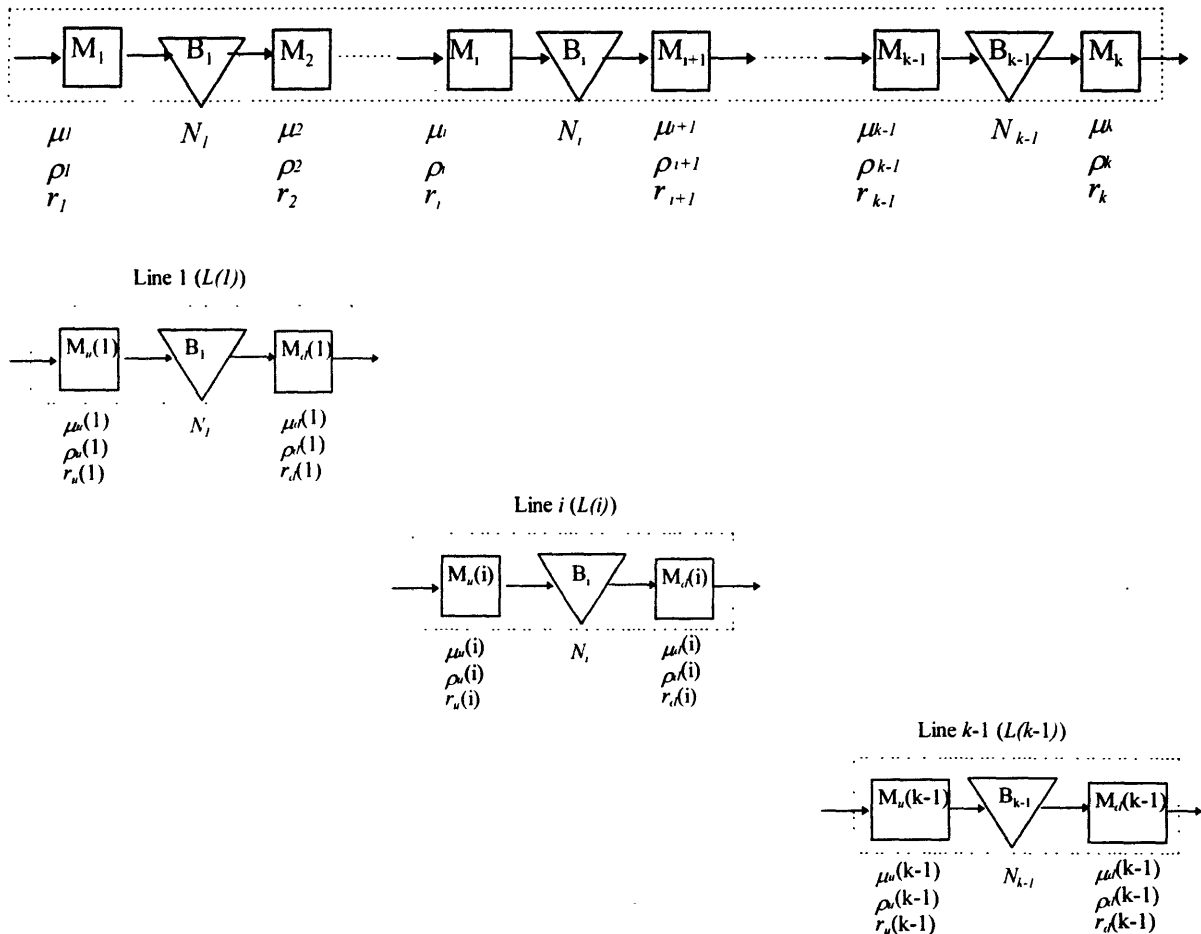
Here we will present a decomposition technique for multiple server series transfer lines, as developed by Burman [2]. The decomposition method allows for serial lines with multiple servers to be described in terms of simple two-machine systems such as the one developed in the previous section.

### 5.3.1 Overview of Decomposition Procedure

Transfer line decomposition methods work as follows. Given a transfer line with  $k$  machines and  $k-1$  buffers, this system can be approximated as  $k-1$  two-machine single buffer systems. This is illustrated in Figure 5-3. To explain this analysis, suppose there is an observer at buffer  $i$  who sees parts entering and leaving the buffer, but cannot see any machines upstream from machine  $i$ , or any machines downstream from machine  $i+1$ . An observer from this perspective cannot tell the difference between a two machine transfer line with machines  $M_u(i)$ ,  $M_d(i)$  (here we use the subscripts  $u$  and  $d$  for *upstream* and *downstream*, respectively), and the original transfer line with  $k$  servers. If we apply this analysis to all such  $k-1$  intermediate systems, we can successfully describe the flow behavior of the transfer line through the use of  $k-1$  two-machine transfer lines such as the one analyzed in section 5.2. We can then calculate the throughput as the production rate of any one of the two-machine systems (recall production rate across a serial line is equal

for all machines due to conservation of flow). We can also calculate the total average WIP as the sum of the average WIP for all such two machine systems. With this, we can then calculate average TPT through Little's Law. In Figure 5-3, we have labeled each of the two-machine lines as  $L(1), \dots, L(i), \dots, L(k-1)$ . In the coming sections, we will refer to these as "virtual" transfer lines and to the upstream and downstream machines as "virtual" machines. The original transfer line components will be referred to as "real."

Figure 5-3: Decomposition Method



### 5.3.2 Decomposition Equations

The object of decomposition is to be able to describe each two-stage line in terms of the original parameters of the transfer line. Since each two-machine line,  $L(i)$ , has six unknown parameters ( $r_u(i)$ ,  $r_d(i)$ ,  $\rho_u(i)$ ,  $\rho_d(i)$ ,  $\mu_u(i)$ , and  $\mu_d(i)$ ),  $6(k-1)$  equations are needed for a  $k$ -machine line. We will present these equations, but an exact derivation is best found in Burman [2].

The variables used in the following equations, are defined below (refer back to Figure 5-3):

- $\rho_u(i)$  and  $\rho_d(i)$  are the failure rates of “virtual” machines  $M_u(i)$  and  $M_d(i)$ , respectively.
- $r_u(i)$  and  $r_d(i)$  are the repair rates of “virtual” machines  $M_u(i)$  and  $M_d(i)$ , respectively.
- $\mu_u(i)$  and  $\mu_d(i)$  are the production rates of “virtual” machines  $M_u(i)$  and  $M_d(i)$ , respectively.
- $e_u(i)$  and  $e_d(i)$  are the isolated efficiencies of “virtual” machines  $M_u(i)$  and  $M_d(i)$ , respectively. Recall that the isolated efficiency of a machine  $i$ ,  $e_i$ , is  $r_i/(r_i + \rho_i)$
- $P(i)$  is the steady state production rate (or average throughput) of “virtual” line  $L(i)$ .
- $\mathbf{p}_i(x, \alpha_1, \alpha_2)$  is the probability that the “virtual” line  $L(i)$  is in state  $(x, \alpha_1, \alpha_2)$ .
- $\rho_i$  is the failure rate of the “real” machine  $i$ .
- $r_i$  is the repair rate of the “real” machine  $i$ .
- $e_i$  is the isolated efficiency of the “real” machine  $i$ .



### 5.3.2.1 Interruption of Flow Equations

The first set of equations is the so called *interruption of flow* (IOF) equations. They provide the failure rates of  $M_u(i)$  and  $M_d(i)$ . These equations are complicated by the fact that than an upstream failure, from the perspective of  $B_i$ , could be caused by either  $M_i$  actually failing or a starvation caused by  $B_{i-1}$  being empty and  $M_u(i-1)$  being down simultaneously. The IOF equations account for both of these possibilities. These equations are obtained in a similar manner as for the two-machine Markov process: they enumerate the possibilities in which an upstream failure could occur (as described above), and assign probabilities to each of these possibilities. The failure rates of  $M_u(i)$  and  $M_d(i)$ , respectively, are given by the following two equations:

$$\rho_u(i) = \rho_i \left( 1 + \frac{\mathbf{p}_{i-1}(0,1,1)\mu_u(i)}{P(i) - \mathbf{p}_i(N_i,1,1)\mu_u(i)} \left( \frac{\mu_u(i-1)}{\mu} - 1 \right) \right) + \left( \frac{\mathbf{p}_{i-1}(0,0,1)\mu_u(i)}{P(i) - \mathbf{p}_i(N_i,1,1)\mu_u(i)} \right) r_u(i-1)$$

$$i=2, \dots, k-1. \quad (5.41)$$

$$\rho_d(i) = \rho_{i-1} \left( 1 + \frac{\mathbf{p}_{i+1}(N_{i+1},1,1)\mu_d(i)}{P(i) - \mathbf{p}_i(0,1,1)\mu_u(i)} \left( \frac{\mu_d(i+1)}{\mu_{i+1}} - 1 \right) \right) + \left( \frac{\mathbf{p}_{i+1}(N_{i+1},1,0)\mu_d(i+1)}{P(i) - \mathbf{p}_i(0,1,1)\mu_u(i)} \right) r_d(i+1)$$

$$i=1, \dots, k-2. \quad (5.42)$$

### 5.3.2.2 Resumption of Flow Equations

The second set of the decomposition equations is the *resumption of flow* (ROF) equations, which provide the repair rates of  $M_u(i)$  and  $M_d(i)$ . Similar to the IOF equations, the ROF equations are derived by the realization that, from the perspective of

buffer  $i$ , a failure can be caused by either machine  $i$  failing or a starvation caused by buffer  $i-1$  being empty and  $M_d(i-1)$  being down simultaneously. The ROF equations, like their IOF counterparts, are derived by enumerating the states that will lead to recovery (from both of the conditions listed above) and the finding transition probabilities. The repair rate equations are given below.

$$r_u(i) = r_u(i-1) \frac{\mathbf{p}_{i-1}(0,0,1)r_u(i)\mu_u(i)}{\rho_u(i)P(i)} + r_i \left( 1 - \frac{\mathbf{p}_{i-1}(0,0,1)r_u(i)\mu_u(i)}{\rho_u(i)P(i)} \right) \\ i=2, \dots, k-1. \quad (5.43)$$

$$r_d(i) = r_d(i+1) \frac{\mathbf{p}_{i+1}(N_{i+1},1,0)r_d(i)\mu_d(i)}{\rho_d(i)P(i)} + r_{i+1} \left( 1 - \frac{\mathbf{p}_{i+1}(N_{i+1},1,0)r_d(i)\mu_d(i)}{\rho_d(i)P(i)} \right) \\ i=1, \dots, k-2. \quad (5.44)$$

### 5.3.2.3 Processing Rate Equations

The final set of decomposition equations are derived for the processing rate of the upstream and downstream machines  $M_u(i)$  and  $M_d(i)$ . These equations are derived by *conservation of flow*, and *flow-rate idle-time* relationships in Gershwin [8]. They are presented here in their final form.

$$\mu_u(i) = \frac{1}{e_u(i)} \left\{ \frac{1}{\frac{1}{P(i)} + \frac{1}{e_i\mu_i} - \frac{1}{e_d(i-1)\mu_d(i-1)}} \right\} \quad i = 2, \dots, k-1 \quad (5.45)$$

$$\mu_u(i) = \frac{1}{e_d(i)} \left\{ \frac{1}{\frac{1}{P(i)} + \frac{1}{e_{i+1}\mu_{i+1}} - \frac{1}{e_u(i+1)\mu_u(i+1)}}} \right\} \quad i = 1, \dots, k-2 \quad (5.46)$$

#### 5.3.2.4 Boundary Conditions

These are the final six equations, which added to our previous  $6(k-2)$  give the need  $6(k-1)$  equations (recall we have  $6(k-1)$  unknowns). These so called *boundary conditions* are the most straightforward. Note that the upstream machine for "virtual" line 1 is simply the machine 1 of the original line, and the downstream machine for "virtual" line  $k-1$  is simply machine  $k$  of the original line. This gives us the following equations:

$$r_u(1) = r_1 \quad (5.47)$$

$$\rho_u(1) = \rho_1 \quad (5.48)$$

$$\mu_u(1) = \mu_1 \quad (5.49)$$

$$r_u(k-1) = r_k \quad (5.50)$$

$$\rho_u(k-1) = \rho_k \quad (5.51)$$

$$\mu_u(k-1) = \mu_k \quad (5.52)$$

#### 5.3.3 Numerical Solution Methods

There are several methods which have been developed to solve decomposition-type equations. Among, these the Dallery-David-Xie algorithm [5] is the quickest and most reliable. Burman [2] developed the Accelerated Dallery-David-Xie algorithm (ADDX) specifically for continuous material decomposition equations. We do not present the algorithm or convergence proofs here, but they are treated in detail by Burman [2] and

## 5.4 The Series-Parallel Model

We are now at the final stage of our analytical model. The technique proposed in this section, which was developed by Burman [2], approximates each *stage* (i.e. station family) of a series-parallel flow line as a single unreliable machine. Once this is done, the system “collapses” into a conventional series transfer line which we can analyze through decomposition methods. The simplification of each stage in the flow line as a single machine is done through a *method of moments* approach. That is, each machine in the “collapsed” line is designed so that it matches the first and second moments of throughput (i.e. the expected value and variance) of the original station family.

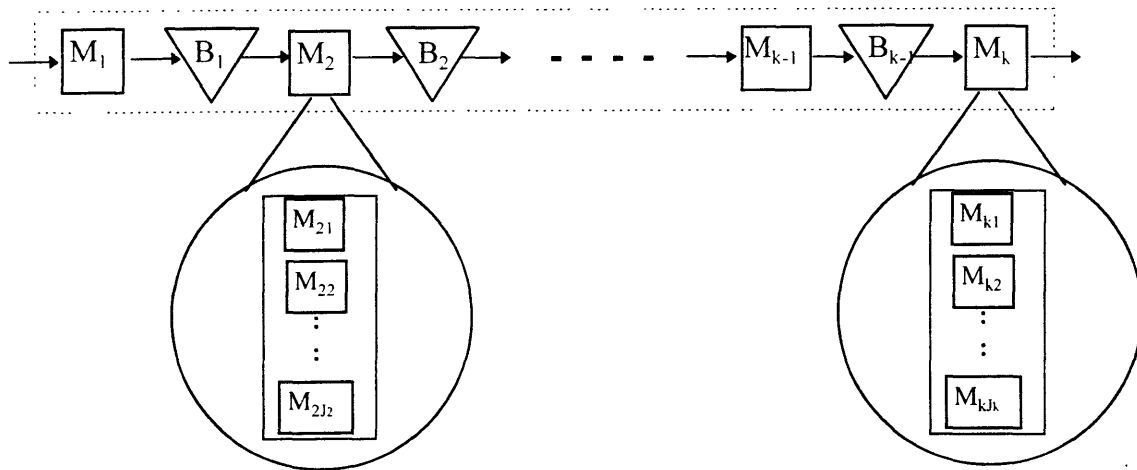
A typical example of a series-parallel flow line is the system presented in the case study.<sup>10</sup> If we view each station family as a single machine, this becomes a simple series line. An illustration is shown in Figure 5-4. Each set of parallel machines (or station family) is represented as a single machine. The resulting system can be described through the decomposition method. Our objective is, therefore, to obtain parameters  $r$ ,  $\rho$ ,  $\mu$  for each of the collapsed machines from the original parameters of the machine.

---

<sup>9</sup>The ADDX algorithm, which is used in our analysis of the case study, can be found in [2].

<sup>10</sup>In reality, the semiconductor assembly process contains an embedded parallel flow line at stage 5, i.e. there is a set of parallel *bays*, which in turn have a set of parallel *machines*. This, however, is only a simple extension of the analysis presented here.

Figure 5-4: “Collapsing” a series-parallel flow line into a single series flow line



### 5.4.1 Assumptions and Nomenclature

Our assumptions about the series-parallel system are the same as those for the continuous material two-machine model presented in section 5.2. We assume that machine processing rates are non-homogeneous and machines are unreliable with ODFs.

We will refer to each set of parallel machines or station family as a *work center*. Work center  $i$  will be denoted by  $M_i$ , and having  $J_i$  parallel machines. Each machine  $ij$ , in work center  $i$  ( $M_{ij}$ ) is capable of producing material at a rate of  $u_{ij}$ . It will process at that rate unless impeded by a failure, starvation or blockage. When a work center becomes either partially blocked or partially starved, each machine produces material at a rate proportional to its production capacity as a percentage of total capacity of the work center. We will let  $\rho_{ij}$  and  $r_{ij}$  denote the failure and repair parameters, respectively, of machine  $ij$ .

The "virtual" or collapsed machine that represents work center  $i$  will be denoted by  $M_i'$ . Likewise, the parameters of machine  $M_i'$  will be  $\rho_i'$ ,  $r_i'$ , and  $\mu_i'$ .

An important assumption we will make is that when a set of parallel machines operates in isolation, a failure of one machine does not affect the performance of the other machines, i.e. each machine performs independently. This clearly differs from reality because a failure of a machine in a work center that is either partially blocked or partially starved places a disproportionate load on the other machines, thus increasing the probability that they will fail (we are assuming ODF).

### **5.4.2 Solution Approach**

For each set of parallel machines at work center  $i$ , we must generate estimates for  $\rho_i'$ ,  $r_i'$ , and  $\mu_i'$ . To estimate these parameters, we will use a variant of the *method of moments*, i.e. we will make the expected value and variance of throughput of  $M_i'$  be the same as the expected value and variance of throughput for work center  $i$ . As a third equation (since we have three parameters), we will set the maximum production rate of machine  $M_i$  to be equal to the maximum production rate of work center  $i$ .

### **5.4.3 Collapsed Transfer-line Parameters**

#### **5.4.3.1 Equation 1: Variance of Throughput**

We can approximate the variability of the output of the set of parallel machines over  $t$  time units by first predicting the variance of output of each machine  $M_{ij}$  (refer to [2] for

the derivation<sup>11</sup>).

$$V_{ij}(t) = \mu_{ij}^2 \left( (e^{-(r_{ij} + \rho_{ij})t} - 1) \frac{2r_{ij}\rho_{ij}}{(r_{ij} + \rho_{ij})^4} + \frac{2r_{ij}\rho_{ij}}{(r_{ij} + \rho_{ij})^3} t \right) \quad (5.53)$$

To find  $V_i(t)$ , the variance of the combined set of parallel machines at work center  $i$ , we will use the fact that the variance of the sum of a set of independent random variables is the sum of the variances [7]. Recall that we are *assuming* that the machines operate independently. Therefore,

$$V_i(t) = \sum_{j=1}^{J_i} \mu_{ij}^2 \left( (e^{-(r_{ij} + \rho_{ij})t} - 1) \frac{2r_{ij}\rho_{ij}}{(r_{ij} + \rho_{ij})^4} + \frac{2r_{ij}\rho_{ij}}{(r_{ij} + \rho_{ij})^3} t \right) \quad (5.54)$$

We will now equate the variance of our virtual machine  $V_i'(t)$  to that of work center  $i$ ,  $V_i(t)$ .

$$\begin{aligned} \mu_i^2 \left( (e^{-(r_i + \rho_i)t^*} - 1) \frac{2r_i\rho_i}{(r_i + \rho_i)^4} + \frac{2r_i\rho_i}{(r_i + \rho_i)^3} t^* \right) = \\ \sum_{j=1}^{J_i} \mu_{ij}^2 \left( (e^{-(r_{ij} + \rho_{ij})t} - 1) \frac{2r_{ij}\rho_{ij}}{(r_{ij} + \rho_{ij})^4} + \frac{2r_{ij}\rho_{ij}}{(r_{ij} + \rho_{ij})^3} t \right) \end{aligned} \quad (5.55)$$

Here  $t^*$  is some appropriate length of time. Since we are interested in long-term, or steady state production, we will allow  $t^*$  to very large (but finite). When  $t^*$  is very large, notice that all terms that do not have  $t^*$  as a factor become insignificant, the resulting equation is:

---

<sup>11</sup>Carrascosa [4] does a comprehensive treatment of the variance of output for two-machine lines.

$$\mu_i^2 \left( \frac{2r_i \rho_i}{(r_i + \rho_i)^3} \right) = \sum_{j=1}^{J_i} \mu_{ij}^2 \left( \frac{2r_{ij} \rho_{ij}}{(r_{ij} + \rho_{ij})^3} \right) \quad (5.56)$$

where we have divided both sides by  $t^*$ . Note that this is an *approximation*, not a limit.

#### 5.4.3.2 Equation 2: Expected Value of Throughput

In an analogous manner, we equate the expected throughput of the equivalent machine  $M_i^l$  to the expected throughput of the set of parallel machines at work center  $i$ . Here we use the fact that the expected value of the sum of a set of random variables is the sum of the expected values [2].<sup>12</sup> The resulting equation is:

$$\mu_i^l \left( \frac{r_i}{r_i + \rho_i} \right) = \sum_{j=1}^{J_i} \mu_{ij} \left( \frac{r_{ij} \rho_{ij}}{r_{ij} + \rho_{ij}} \right) \quad (5.57)$$

#### 5.4.3.3 Equation 3: Maximum Processing Rate

We define the maximum isolated processing rate as the maximum possible rate at which a stage can produce material. When all parallel machines are working, this is simply the sum of all the individual isolated processing rates. The resulting equation is

$$\mu_i^l = \sum_{j=1}^{J_i} \mu_{ij} \quad (5.58)$$

Thus we can now solve for  $\rho_i^l$ ,  $r_i^l$ , and  $\mu_i^l$  with equations (5.56), (5.57), (5.58). We can

---

<sup>12</sup>We also demonstrated this result for average TPT in Chapter 1. Note that independence is *not* a necessary condition for this result.



then apply the decomposition method to the reduced line and calculate the average TPT with equation 5.40.

## 5.5 Series-Parallel Model Applied to Case Study

### 5.5.1 Model Application

The assembly process presented in the case study has a number of important properties. Most importantly, for any given work center, all of the parallel machines are fairly similar. This assumption was made in our DES model and provided reasonable results. It is therefore sensible to also apply this assumption to our analytical model. When we assume machines at each work center are identical, equations (5.56), (5.57), (5.58) reduce to:

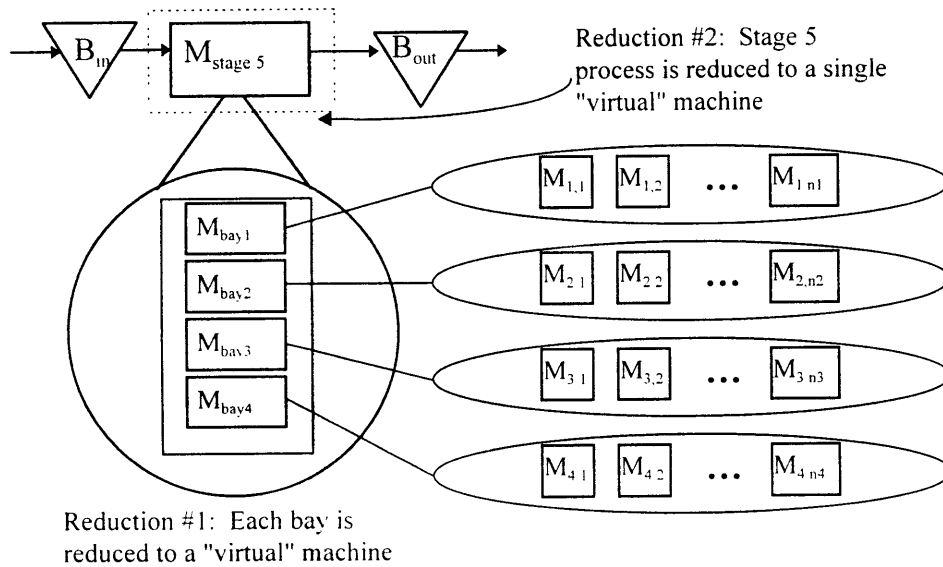
$$\begin{aligned}\rho_i^1 &= J_i \rho_i \\ r_i^1 &= J_i r_i \\ \mu_i^1 &= J_i \mu_i\end{aligned}\tag{5.59}$$

which is a surprisingly elegant result, considering that the analysis performed was non-trivial.

There are notable difference between the series-parallel analytical model and the assembly line of the case study. In particular, the stage 5 process does not have simple series-parallel structure, since it includes an imbedded set of parallel machines within each bay. Using the same line of reasoning employed in the derivation of our model,

however, there is a fairly straight-forward manner for analyzing such a system. If each bay is considered as a "virtual" machine, then the series-parallel approximation can be applied to each bay and then *reapplied* to each of the virtual machines which represents a bay. This is illustrated in Figure 5-5.

Figure 5-5: Series-Parallel approximation for stage 5 process



Thus, for the stage 5 process, we will apply equation (5.59) to obtain the first set of parameters for the first set of "virtual" machines (the bays). We will then have to apply equations (5.56), (5.57), (5.58) to obtain parameters for the final virtual machine (we cannot directly apply equation 5.59 since the bays are dissimilar).

A second significant difference is that in our model we assumed that all failures and repairs could be represented by the parameters  $r$  and  $\rho$  (and recall that all of our analytical

models have assumed use-based exponentially distributed failures and repairs). In reality, of course, there are scheduled and unscheduled downtimes, and, furthermore, unscheduled downtimes are divided into repairs and assists. Since we have assumed that all downtimes are exponential, we can treat both failures and repairs as Poisson processes with parameters  $1/\text{MTTF}$  and  $1/\text{MTTF}_A$ . Due to the additive property of the Poisson process, the resulting failure distribution parameter will be  $\rho = 1/\text{MTTF} + 1/\text{MTTF}_A$ .<sup>13</sup> Likewise, the repair parameter will be  $r = 1/\text{MTTR} + 1/\text{MTTA}$ .

## 5.5.2 Input Parameters

The resulting parameters  $\mu$ ,  $\rho$  and  $r$  can be calculated as shown in section 5.5.1.<sup>14</sup> In addition, the model requires that we assign values to the buffer capacities,  $N_i$ . This turns out to be difficult, since buffer capacities are not clearly defined as part of the factory WIP management policy. The only *clear* upper bound which we can put on buffer size is the physical limitation, i.e. there is only a limited amount of space on the floor for WIP carts. The buffer capacities used in our analytical model are based on the physical space available for WIP lots.

---

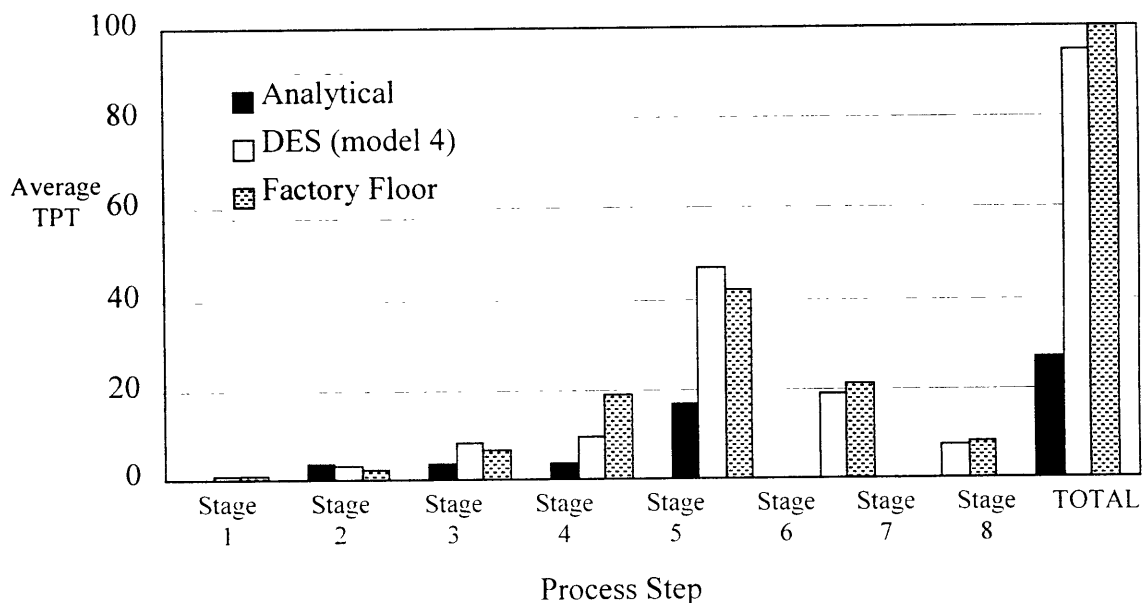
<sup>13</sup> To account for ODF, we have multiplied MTTF and  $\text{MTTF}_A$  by machine utilization.

<sup>14</sup> Confidentiality restricts publication of processing, repair and failure, and buffer limit parameters.

### 5.5.3 Results and Discussion

Figure 5-6 shows average TPT for the analytical model compared to DES and factory-floor. Total average TPT was around 27% percent of factory floor TPT. Although results for mount through die-attach were reasonable, average TPT for stage 7 and stage 8 were vastly underpredicted. Still, the overall result is quite good, considering that DES models routinely underpredict average TPT by 25% to 30%.

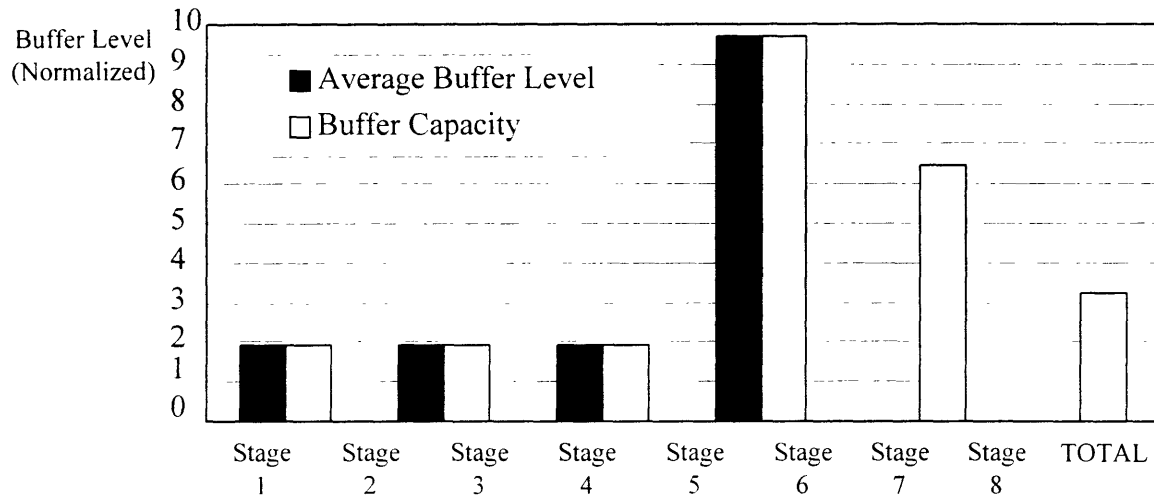
Figure 5-6: Analytical model results compared to DES and factory-floor



The underprediction could be due to a number of plausible reasons. One clear reason is that scheduled downtimes, including PMs, were not taken into account. In addition, the analytical system constantly feeds the system (infinite reservoir for of parts), thus

assuring a fairly uniform rate at which parts arrive in the system. In reality (and in the simulation), there are "globs" or "lumps" of parts which clog the bottlenecks and increase the variance in TPT. The small TPTs for the post-stage 5 processes can be attributed to the fact that, in our analytical model, the queues tend to be quite small, thus most of the parts get straight-through once they get past the bottleneck. Figure 5-7 shows the average buffer levels (and buffer capacities) for the analytical model.

Figure 5-7: Average buffer levels for analytical model (normalized)



The most salient features of this graph are the near-capacity average queue levels for the pre-constraint buffers and the extremely small queue sizes for the post-constraint processes. These results are reasonable, since the pre-constraint processes will often be blocked, while the post-constraint processes will often be starved. In reality the post-constraint processes are less reliable than in the model, thus leading to buffer accumulation and larger TPTs at the post-constraint processes.



# Chapter 6

## Conclusions

### 6.1 Summary of Results

In this thesis, we have made four significant advances. First, in Chapter 3, we found that multiple DES runs with different random seeds and otherwise identical setups, converged to the same value of average TPT over *long*<sup>1</sup> time periods. In particular, this assures that if a given DES model is run for an *appropriately long* time interval, the resulting average TPT will be very close that obtained by averaging the results of multiple DES runs (disregarding transient or "warm-up" periods). In sum, DES models of transfer-lines such as the one presented in Chapter 3, do not appear to have multiple steady states *or final classes*.<sup>2</sup>

---

<sup>1</sup> It is important to emphasize long time periods, since over the "transient" period, average TPT will differ significantly among runs with dissimilar random seeds.

<sup>2</sup> We can only make his claim for simple serial-flow lines such as the one presented in Chapter 3. We have not yet studied convergence behavior for re-entrant and other flow-lines.

Second, in Chapter 4 we showed that DES models that are accurate within 6% of factory floor average TPT can be created by: (1) allocating detail at the constraining process. (2) adjusting failure distribution parameters to reflect operation dependent failures and (3) using an exact lot-starts schedule. For the case study presented in Chapter 4, detail at the constraining process necessitated the decomposition of the model in to three separate sections (pre-constraint, constraint, and post-constraint) in order to accurately model flow behavior.<sup>3</sup> Arbitrary delays that were added at pre-constraint processes did not significantly add to the total average TPT, thus showing that detail at pre-constraint processes is not necessary to obtain relatively accurate results.

Third, we showed that operator and shift effects at the constraining process do not significantly impact average TPT in the assembly process.<sup>4</sup> This is due to the fact that the input and output buffers at stage 5 are large enough to prevent any blocking or starvation of machines. Starvation and blocking is also prevented because operator breaks (excluding beginning-of-shift meetings) are never taken simultaneously; thus, stage 5 bays are seldomly left unattended. Moreover, in experimental studies performed, we showed that there is a "critical" loading time after which the machines begin to experience blocking and starvation. At the current loading and unloading rate, there appears to be an ample safety margin to prevent starvation and blocking at the stage 5 machines.

---

<sup>3</sup> Recall that a heuristic was used to create a lot-start schedule for each of the stage 5 bay models. A detailed description of this heuristic can be found in Chapter 4 and an is illustrated in Appendix B.

<sup>4</sup> Again, this is a case-specific recommendation which may not be applicable to other processes.



Fourth, in Chapter 5 we applied the analytical models developed by Gershwin [8] and Burman [2] to the assembly process with a resulting average TPT that was 27% of factory-floor TPT. This is a surprisingly good result, considering the practicality of analytical models and that DES models commonly yield average TPTs in the range of 25% to 50% of actual.

## **6.2 Recommendations for Future Action**

Under the current uncertainty of both DES input and validation data, it is unrealistic to pursue modeling methods that will yield better results than the ones found in our model of the assembly process. Although no studies were performed on the "noise" of input and validation data, general opinion (especially among industrial engineers involved in the assembly process) is that it is nontrivial. Moreover, the propagation of input data has not been examined. Accordingly, we recommend that the reliability of DES input data, namely machine uptime and downtime probability distributions, as well as validation data, be closely examined before improved modeling of "structural" factors is pursued.

To reach this objective, independent data collection efforts may be necessary. An increasingly viable data collection medium is satellite lot-tracking. Since the modeler would be able to track lots in real time and be aware of a lot's physical location at all times, satellite lot tracking would be an extremely accurate method for measuring TPT. Due to the high accuracy of each reading, the modeler would only need to collect a reasonably sized sample to reach the desired statistical significance. The reliability of such data would be much higher than the current measure, which takes the average of a

large sample of inaccurate readings.

For the analytical model, results would likely be improved by dividing the "reduced" transfer line into pre-constraint, constraint, and post-constraint in a similar manner as the DES model. In particular, the double reduction performed at stage 5 may have largely contributed to modeling error. Due to the cutting-edge nature of the analytical model presented in Chapter 5, there is a large amount of research to be done in effective implementation of these models, including heuristics for dealing with lot splitting and changing lot sizes during the process flow.

### **6.3 Concluding Remarks**

Although the primary focus of this research was to obtain accurate TPT results from models, the underlying motivation is to exploit the power of models when designing new facilities and making factory process improvements. To this end, it is hoped our research increased the "credibility" of DES and analytical models. We feel that at the state of the art, including advances made in our research, the biggest contributor to modeling error lies in the lack of robust input data, *not* in the lack of sound modeling techniques. Likewise, model results are often discarded as "inaccurate" but the reliability of validation data is seldomly examined.

To continue the exponential increase in microprocessor speed and ever-increasing ramp rate objectives, models will undoubtedly be an *indispensable* tool not only in the analysis, but also in the *design* and *creation* of semiconductor manufacturing systems.

# Appendix A

## Calculations

### A.1 Calculating TPT Variance for Multiple DES Runs (Chapter 3)

Define following parameters:

$n$  = number of parts “produced” over the number of cycles for experiment being run.

$m$  = number of “experiments” or simulations.

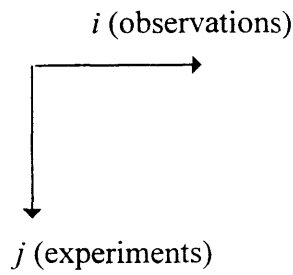
$t_{ij}$  = individual TPT of part  $i$  during experiment  $j$ .

$\hat{s}_i^2$  = our estimator for TPT variance

$\sigma_i^2$  = actual variance of underlying TPT distribution

	<u>Observations</u>	<u>Sample Variance</u>
$i$ (observations)		
	$[t_{11} \dots t_{i1} \dots t_{n1}]$	$\frac{1}{n-1} \sum_{i=1}^n (\overline{t_{i=1}} - t_{i1})^2$
$j$ (experiments)		$\vdots$
		$\vdots$
		$\vdots$

Continued...

	<u>Observations</u>	<u>Sample Variance</u>
	[ $t_{1j} \dots t_{ij} \dots t_{nj}$ ]	$\frac{1}{n-1} \sum_{i=1}^n (\bar{t}_j - t_{ij})^2$
		.
		.
		.
	[ $t_{1m} \dots t_{im} \dots t_{nm}$ ]	$\frac{1}{n-1} \sum_{i=1}^n (\bar{t}_{j=m} - t_{im})^2$

“Best” estimate for variance of TPT is sample variance for entire matrix of readings, i.e.

$$s_i^2 = \frac{1}{mn-1} \sum_{j=1}^m \sum_{i=1}^n (\bar{t}_j - t_{ij})^2 \quad (\text{Where } \bar{t}_j \text{ is the overall sample mean})$$

Estimate presented in Chapter 3 was mean of sample variances (i.e. mean of variances shown above):

$$\hat{s}_i^2 = \frac{1}{m} \sum_{j=1}^m \frac{1}{n-1} \sum_{i=1}^n (\bar{t}_j - t_{ij})^2 = \frac{1}{mn-m} \sum_{j=1}^m \sum_{i=1}^n (\bar{t}_j - t_{ij})^2$$

This approximation is valid when  $n$  is large, i.e.

$$\hat{s}_i^2 \rightarrow s_i^2 \text{ when } mn \gg m \text{ and } \bar{t}_j \text{ s are close to } \bar{t}_j$$

Note that for a transient density function  $s_i^2 \rightarrow \sigma_i^2$  (“actual” variance) as  $n$  gets small and  $m$  gets large. Note that as  $s_i^2 \rightarrow \sigma_i^2$ ,  $\hat{s}_i^2$  becomes a bad estimator. The procedure used here, therefore, is not recommended to obtain parameters for transient density function.

## A.2 Confidence Interval Calculation<sup>1</sup>

To calculate a confidence interval for a given variable, we first calculate the sample expected value and sample variance for the quantity of interest. For each trial  $i$  of simulation of a given line simulated  $n$  times, we obtain an estimate for a given parameter  $x$  that we will call  $x_i$ . For example,  $x$  might be the throughput rate for a line simulated for 40,000 time units. We first calculate for random variable  $x$

$$\sigma^2 = E[x^2] - \{E[x]\}^2 \quad \text{where } E[y] \text{ is the expected value of random variable } y$$

$$S^2 = \frac{\sum_{i=1}^n x_i^2 - \left\{ \sum_{i=1}^n x_i \right\}^2}{n-1}$$

However, we are interested in the confidence interval for  $\bar{x}$  (average  $x$ ). We estimate the expected value of  $\bar{x}$  with the following:

$$\hat{\bar{x}} = \frac{\sum_{i=1}^n x_i}{n}$$

and the sample variance of  $\bar{x}$  is

$$\begin{aligned} S_{\bar{x}}^2 &= \frac{nS_x^2}{n^2} = \frac{S_x^2}{n} \\ &= \frac{\sum_{i=1}^n x_i^2 - \left\{ \sum_{i=1}^n x_i \right\}^2}{n(n-1)} \end{aligned}$$

which leads to a sample standard deviation of  $\bar{x}$  of

$$S_{\bar{x}} = \sqrt{\frac{\sum_{i=1}^n x_i^2 - \left\{ \sum_{i=1}^n x_i \right\}^2}{n(n-1)}}$$

---

<sup>1</sup> Parts of this section were taken from Burman [2].

To determine the confidence interval, we invoke the Law of Large Numbers (Drake [7]). Since  $N > 30$ , we make the common assumption that  $x$  follows a normal distribution. Therefore, we can use the following to estimate the confidence interval around this value:

$$95\% \text{ Confidence Interval} = \hat{x} \pm 1.96 \sqrt{\frac{\sum_{i=1}^n x_i^2 - \left\{ \sum_{i=1}^n x_i \right\}^2}{n(n-1)}}$$

# **Appendix B**

## **Spreadsheet Models**

## B.1 Average Processing Time Heuristic Spreadsheet

Shown below is the spreadsheet for the "average processing time" heuristic for a sample arrival schedule. Here we have assumed a lot in bays 1-3 takes 60 time units to process and a lot in Bay 4 takes 137.14 time units to process (based on the relative capacity of each bay). Refer to Table 4-2 for field formulas.

Lot	Arrival time (ta)	Time in queue (tq)	Bay (Bl) argmin(tfi)	Processing Time	Lot Departure time (td)	Bay1	Bay2	Bay3	Bay4
						tf1	tf2	tf3	tf4
Lot 1	0.0	0.0	1	60.00	60.0	60.0	0.0	0.0	0.0
Lot 2	2.5	0.0	2	60.00	62.5	60.0	62.5	0.0	0.0
Lot 3	21.6	0.0	3	60.00	81.6	60.0	62.5	81.6	0.0
Lot 4	72.0	0.0	4	137.14	209.1	60.0	62.5	81.6	209.1
Lot 5	75.2	0.0	1	60.00	135.2	135.2	62.5	81.6	209.1
Lot 6	100.1	0.0	2	60.00	160.1	135.2	160.1	81.6	209.1
Lot 7	113.5	0.0	3	60.00	173.5	135.2	160.1	173.5	209.1
Lot 8	120.8	14.4	1	60.00	195.2	195.2	160.1	173.5	209.1
Lot 9	126.6	33.5	2	60.00	220.1	195.2	220.1	173.5	209.1
Lot 10	148.8	24.6	3	60.00	233.5	195.2	220.1	233.5	209.1
Lot 11	162.0	33.3	1	60.00	255.2	255.2	220.1	233.5	209.1
Lot 12	168.0	41.2	4	137.14	346.3	255.2	220.1	233.5	346.3
Lot 13	181.7	38.5	2	60.00	280.1	255.2	280.1	233.5	346.3
Lot 14	189.3	44.2	3	60.00	293.5	255.2	280.1	293.5	346.3
Lot 15	207.4	47.8	1	60.00	315.2	315.2	280.1	293.5	346.3
Lot 16	213.8	66.4	2	60.00	340.1	315.2	340.1	293.5	346.3
Lot 17	231.5	62.0	3	60.00	353.5	315.2	340.1	353.5	346.3
Lot 18	236.3	78.9	1	60.00	375.2	375.2	340.1	353.5	346.3
Lot 19	255.2	84.9	2	60.00	400.1	375.2	400.1	353.5	346.3
Lot 20	268.4	77.9	4	137.14	483.4	375.2	400.1	353.5	483.4
Lot 21	288.2	65.3	3	60.00	413.5	375.2	400.1	413.5	483.4
Lot 22	309.2	66.0	1	60.00	435.2	435.2	400.1	413.5	483.4
Lot 23	328.4	71.8	2	60.00	460.1	435.2	460.1	413.5	483.4
Lot 24	415.3	0.0	3	60.00	475.3	435.2	460.1	475.3	483.4
Lot 25	417.7	17.6	1	60.00	495.2	495.2	460.1	475.3	483.4
Lot 26	437.9	22.2	2	60.00	520.1	495.2	520.1	475.3	483.4
Lot 27	457.8	17.5	3	60.00	535.3	495.2	520.1	535.3	483.4
Lot 28	460.8	22.7	4	137.14	620.6	495.2	520.1	535.3	620.6
Lot 29	480.6	14.6	1	60.00	555.2	555.2	520.1	535.3	620.6
Lot 30	488.2	32.0	2	60.00	580.1	555.2	580.1	535.3	620.6
Lot 31	567.5	0.0	3	60.00	627.5	555.2	580.1	627.5	620.6
Lot 32	570.9	0.0	1	60.00	630.9	630.9	580.1	627.5	620.6
Lot 33	588.8	0.0	2	60.00	648.8	630.9	648.8	627.5	620.6
Lot 34	602.0	18.5	4	137.14	757.7	630.9	648.8	627.5	757.7

Continued...



Lot	Arrival time (ta)	Time in queue (tq)	Bay (Bl) argmin(tfi)	Processing Time	Lot Departure time (td)	Bay1	Bay2	Bay3	Bay4
						tf1	tf2	tf3	tf4
Lot 35	614.9	12.6	3	60.00	687.5	630.9	648.8	687.5	757.7
Lot 36	636.5	0.0	1	60.00	696.5	696.5	648.8	687.5	757.7
Lot 37	639.7	9.0	2	60.00	708.8	696.5	708.8	687.5	757.7
Lot 38	657.5	29.9	3	60.00	747.5	696.5	708.8	747.5	757.7
Lot 39	662.2	34.2	1	60.00	756.5	756.5	708.8	747.5	757.7
Lot 40	688.1	20.6	2	60.00	768.8	756.5	768.8	747.5	757.7
Lot 41	693.3	54.1	3	60.00	807.5	756.5	768.8	807.5	757.7
Lot 42	711.1	45.4	1	60.00	816.5	816.5	768.8	807.5	757.7
Lot 43	762.3	0.0	4	137.14	899.5	816.5	768.8	807.5	899.5
Lot 44	764.9	3.9	2	60.00	828.8	816.5	828.8	807.5	899.5
Lot 45	785.7	21.8	3	60.00	867.5	816.5	828.8	867.5	899.5
Lot 46	878.4	0.0	1	60.00	938.4	938.4	828.8	867.5	899.5
Lot 47	901.7	0.0	2	60.00	961.7	938.4	961.7	867.5	899.5
Lot 48	907.9	0.0	3	60.00	967.9	938.4	961.7	967.9	899.5
Lot 49	927.4	0.0	4	137.14	1064.6	938.4	961.7	967.9	1064.6
Lot 50	937.0	1.5	1	60.00	998.4	998.4	961.7	967.9	1064.6
Lot 51	952.8	8.9	2	60.00	1021.7	998.4	1021.7	967.9	1064.6
Lot 52	971.6	0.0	3	60.00	1031.6	998.4	1021.7	1031.6	1064.6
Lot 53	976.0	22.5	1	60.00	1058.4	1058.4	1021.7	1031.6	1064.6
Lot 54	999.4	22.2	2	60.00	1081.7	1058.4	1081.7	1031.6	1064.6
Lot 55	1015.8	15.7	3	60.00	1091.6	1058.4	1081.7	1091.6	1064.6

Note that unlike the "deterministic factory" heuristic, this heuristic depends on the arrival schedule and therefore has a non-repeating pattern. Here we have calculated lot assignments up to the 55th lot.

## B.2 Bay Assignment Comparison for Average Processing Rate and Deterministic Factory Heuristics

Lot Number	Bay Assignment		Lot Number	Bay Assignment	
	Heuristic 1	Heuristic 2		Heuristic 1	Heuristic 2
1	1	1	34	3	4
2	2	2	35	4	3
3	3	3	36	1	1
4	4	4	37	2	2
5	1	1	38	3	3
6	2	2	39	1	1
7	3	3	40	2	2
8	1	1	41	3	3
9	2	2	42	4	1
10	3	3	43	1	4
11	4	1	44	2	2
12	1	4	45	3	3
13	2	2	46	1	1
14	3	3	47	2	2
15	1	1	48	3	3
16	2	2	49	1	4
17	3	3	50	2	1
18	4	1	51	3	2
19	1	2	52	4	3
20	2	4	53	1	1
21	3	3	54	2	2
22	1	1	55	3	3
23	2	2			
24	3	3			
25	1	1			
26	2	2			
27	3	3			
28	4	4			
29	1	1			
30	2	2			
31	3	3			
32	1	1			
33	2	2			

**Heuristic 1:**  
**"Deterministic Factory"**  
 From Table 4-3

**Heuristic 2:**  
**"Average Processing Time"**  
 From Appendix B.1

# Bibliography

- [1] A. Bonvik, *Performance Analysis of Manufacturing Systems Under Hybrid Control Policies*, PhD thesis, Massachusetts Institute of Technology, 1996.
- [2] Burman, *New Results in Flow Line Analysis*, PhD thesis, Massachusetts Institute of Technology, 1995.
- [3] J. Buzacott and L. Hanifin. Models of automatic transfer lines with inventory banks—A review and comparison. *IIE Transactions*, 10:197-207, 1978.
- [4] M. Carrascosa. *Variance of Throughput in Two-machine Transfer-Line*. MS thesis, Massachusetts Institute of Technology, 1995.
- [5] R. Dallery, Y David, and. X. Xie. An efficient algorithm for analysis of transfer lines with unreliable machines and finite buffers. *IIE Transactions*, 20:280-283, 1988.
- [6] R. Dallery, Y David, and. X. Xie. Approximate analysis of transfer lines with unreliable machines and finite buffers. *IIE Transactions on Automatic Control*, AC-34:943-953, 1989.
- [7] A. Drake. *Fundamentals of Applied Probability Theory*. McGraw-Hill Publishing Company, Inc., 1988.

[8] S.B. Gershwin. *Manufacturing Systems Analysis*. Prentice Hall Inc., New Jersey, 1994.

[9] E. Goldratt and J. Cox (1986), *The Goal*, North River Press, 1986.

[10] Intel Corporation technical report.\*

[11] Intel Corporation technical report.\*

[12] Intel Corporation technical report.\*

---

\*These sources could not be disclosed for proprietary concerns. Contact Court Hilton at Intel Corporation for additional information.