UNIVERSITY OF THESSALY

DEPARTMENT OF MECHANICAL ENGINEERING

Diploma Thesis

IMPLEMENTATION OF A QUEUEING NETWORK MODELLING METHOD FOR ANALYZING SEMICONDUCTOR MANUFACTURING SYSTEMS

by

PAPADIMITROPOULOS KONSTANTINOS

Submitted in partial fulfillment of the

requirements for the Diploma

in Mechanical Engineering

December, 2018

Η έγκριση της διπλωματικής εργασίας από το Τμήμα Μηχανολόγων Μηχανικών της Πολυτεχνικής Σχολής του Πανεπιστημίου Θεσσαλίας δεν υποδηλώνει αποδοχή των απόψεων του συγγραφέα (Ν. 5343/32 αρ. 202 παρ. 2).

The Diploma Thesis of Konstantinos Papadimitropoulos has been examined and approved by a three-member Examination Committee as satisfactory for the thesis requirement for the Mechanical Engineering degree.

Approved by:

First Examiner (Supervisor)	Dr. George Liberopoulos Professor, Department of Mechanical Engineering, University of Thessaly
Second Examiner	Dr. Dimitrios Pandelis Associate Professor, Department of Mechanical Engineering, University of Thessaly
Third Examiner	Dr. George Saharidis Assistant Professor, Department of Mechanical Engineering, University of Thessaly

Acknowledgements

I feel the need to express my gratitude to everyone who supported me throughout the course of this thesis and all years of study, for the moral and knowledge support.

First of all, I thank God that blessed me to complete successfully my studies, as he is the sponsor of my efforts. Additionally, I would like to thank my family and especially my parents for their love and their endless support and encouragement by believing and instilling confidence in me.

I am indebted to my advisor, Dr. George Liberopoulos, for his inspiring guidance and his significant contribution in my research work. I would like to thank him for suppling me with a great feedback of knowledge and awareness for my future career.

I am also grateful to the PhD candidate and my friend Michalis Deligiannis, who gave me real help in my project and with his approaches and ideas made the research work enjoyable.

I express my warm thanks to my friends from my student years with whom I share my interactions and discussions. Special thanks go to my friend student George Logothetis whose involvement was essential in the start phase of the thesis course.

Lastly, I want to refer to my passion for the sports club of AEK Athens which gave me joy.

Abstract

The study of a semiconductor manufacturing system is very difficult in practice due to the highly complexity of its re-entrant behavior. The complexity arises also from the fact that the basic entities of a fabrication facility, which is the jobs composed of wafers, have stochastic routing flows. Not only this, but the jobs might be scrapped and reworked, partially or entirely. So, the size of a job may be change during the process. Furthermore, a wafer fabrication facility (wafer fab) consists of several different types of tools grouped into distinct tool groups. Each tool type presents specific characteristics in conjunction with the incapacitation events that occur on this. All the above factors increase the variability of such a production line making the control of it challenging.

Thereafter, with these in mind, for performance evaluation of manufacturing systems, queueing models are extensively used. Especially, we present an open queueing network model developed by Connors et al. [3] for rapid performance analysis of semiconductor manufacturing facilities. This model differs from other queueing models in the detailed ways in which the effect of rework and scrap on wafer job sizes is characterized and different tool groups found in semiconductor wafer fab are modeled.

In the analysis of the theoretical model of a Queueing Network, the Decomposition Approach is implemented to interpret more efficient the performance features of the tool groups, represented by individual queues. The variability is incorporated in the first two moments of the inter-arrival time even the service time and for that reason coefficients of variation are defined. Hence, from the traffic rates and the time requirements of the jobs at each of the tool groups the performance measures of utilization and queueing delay are affected. Considering that, we analyze two different types of tool groups, single-wafer tools and batch tools. For the batch tools a greedy policy is occasionally described. The model achieves, in the end, to estimate a mean cycle time for the products of the system.

Our main contribution to the based research area of analysis of production control methods is the development of a computational model based on Connors et al. [3]

theoretical analysis, modelling every needed aspect of this and implementing various examples in order to compare the accuracy and the reliability of the model. The bibliographic model that is thoroughly used is the Mini-Fab model introduced by Kempf [6]. Implementing this in our computational model we derived reasonable and encouraging results. The performance measures of mean utilization and mean queueing delay at each tool group type and the average cycle time for all released products are verified from related literature and simulated examples. From that we conclude that our implementation approach is reliable.

Contents

Table of Figures

List of Tables

Chapter 1

Introduction

1.1 General

In the last decade, the electronics industry has become one of the largest and fastest growing industries in the world. Semiconductors became the leaders of the electronics revolution due to the rapid development of technology and especially the fast-changing market demands for networking, storage components, telecommunications/wireless, consumer, computer, and storage systems that have become necessary tools for today. On its basis, the semi-conductor material is the raw silicon or gallium, mainly, the raw silicon, which is identified as the dominant material for semiconductor fabrication. Silicon is the main-substrate that is used to manufacture integrated circuits (ICs). An integrated circuit (IC) is a device made of interconnected electronic components that can hold millions of circuits that are capable of performing a wide range of computing operations at high speeds. Most importantly, silicon is abundantly available in nature and has very special properties making it an inexpensive raw material, which is extremely affordable and appealing. It acts as a semiconductor, wherein it conducts electricity under some conditions and alternatively acts as an insulator in others. These properties have enabled ICs to be extensively used in electronic products like computers, transistors and indispensable “smart” devices, the mobile phones, in which they have been incorporated advanced technologies ([13]).

The Semiconductor Manufacturing comprises four (4) basic stages: the *Wafer Fabrication (Wafer-Fab)*, the *Probe* or *Sort*, the *Assembly* and finally the *Testing*, as it is illustrated in Figure 1.1. These manufacturing processes can be grouped into two essential categories: "Front-End" and "Back-End Operations". Wafer Fabrication and Probe/Sort are included in "Front-End Operations" while Assembly and Testing belong to "Back-End Operations".

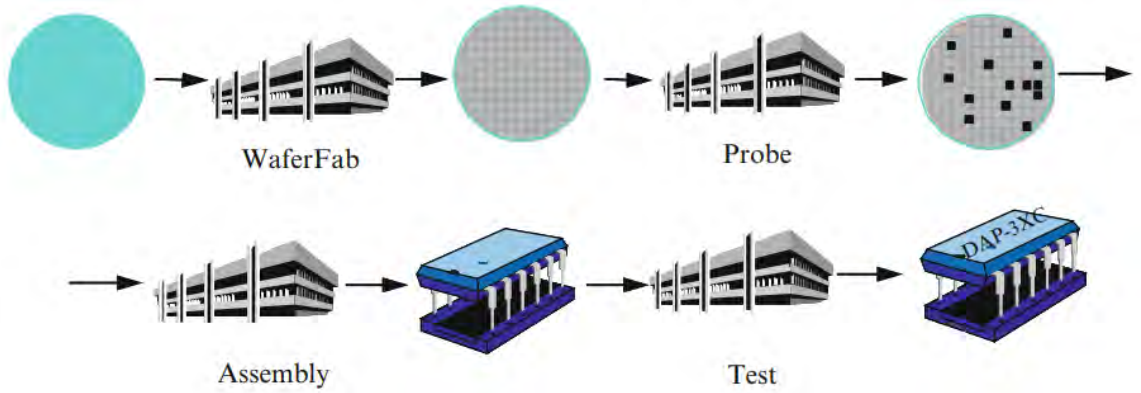

Figure 1.1: Stages of Semiconductor Manufacturing (source: Mönch et al., [9])

In this thesis we are going to deal with the first stage of the Semiconductor Manufacturing and most crucial part of wafer processing, the Wafer Fabrication. Wafer Fabrication, as we mentioned, is the manufacturing of integrated circuits (ICs or chips) on thin silicon discs (wafers) and, accordingly, it is at the heart of this industry. However, the terminologies "Front-End" and "Back-End Operations" of the Semiconductor Manufacturing do not work to build the model on this thesis project.

In comparison to discrete manufacturing facilities, the unique challenges that semiconductor fabrication processes faces are ([10]):

- Large number of process steps;
- Complex and reentrant process flows;
- Intermixture of lots and single wafers in tool queues;
- Batching of different lots having common processing recipes and times;
- Different rework routes for every product; and

- Regular use of metrology tools for parameter measurement and to judge the health of the wafer.

As a result from the above challenges, the fabrication of ICs on silicon wafers is arguably the most complex manufacturing process in existence. Also, it is the most time-consuming and the most costly step. This complexity is caused by many factors including multiple products, routes with several hundred process steps, and a large number of machines (tools). More specifically, wafer fabrication is characterized by the following process conditions:

- A mix of different process types, for example, batch processes, i.e., several jobs, can be processed simultaneously on the same machine vs. single wafer processes;
- Unrelated parallel machines that are often highly unreliable or require considerable preventive maintenance to keep them reliable;
- Sequence-dependent setup times that can in some cases take considerably longer than the time to process a job;
- Variety of products with a changing product mix; and
- Customer due dates that are very aggressive.

Afterwards, refer at [10], a vast amount of research has gone into optimizing the wafer fabrication process since it is the most expensive phase of semiconductor manufacturing. This stage involves the addition of layers of circuits on the silicon wafer through a sequence of 300-600 intricate steps. As we said, the process flow is highly reentrant, where many of the processing steps are repeated for every layer. Different sequences of steps are required for different circuits and some steps can include sub-operations on different tools. Processing steps also vary in the quantity of wafers that are processed at one time. These quantities can be single wafers, wafer lots or batches of wafer lots. Typically a lot consists of 24-48 wafers, while a batch consists of 6 lots. Each lot or job contains a fixed number of wafers. These characteristics of semiconductor manufacturing make it different from traditional manufacturing. Due to the reentrant nature of the flow, wafers at different stages of production queue up in front of the same tool a number of times.

A simplification of the typical reentrant flow of a wafer fab in a semiconductor manufacturing facility is shown in Figure 1.2, where the different work areas within a wafer fab are also summarized.

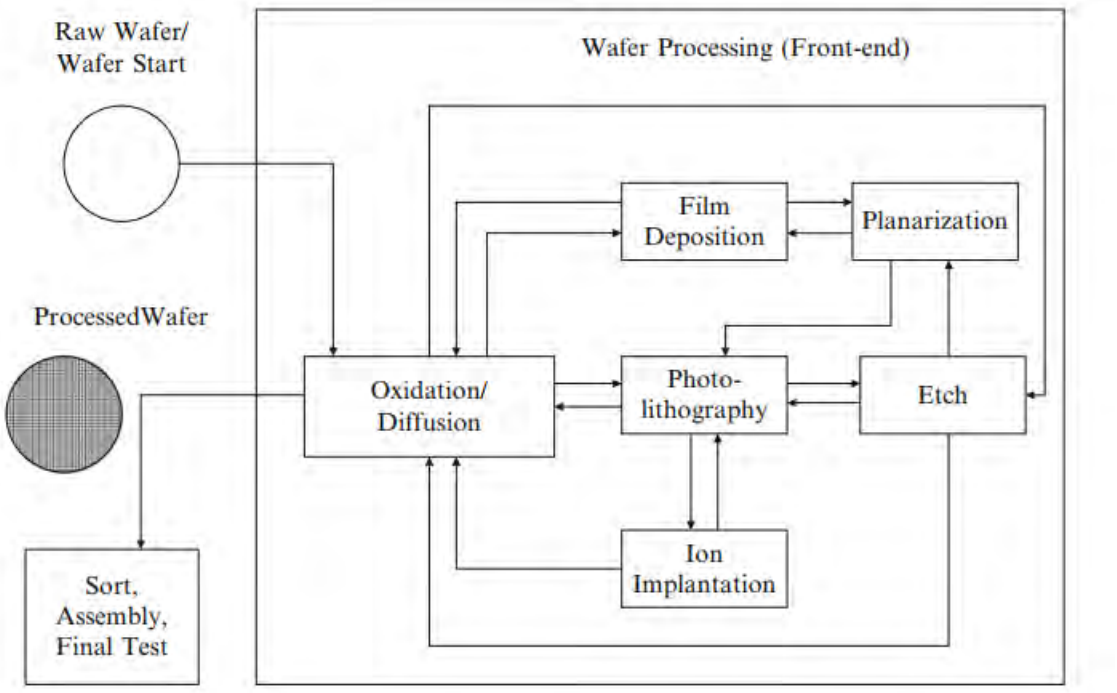

Figure 1.2: Operations in a wafer fab (source: Mönch et al., [9])

Referring to [9], in the production process every semiconductor manufacturing starts with raw wafers and building the electronic circuit layer-by-layer on each wafer can be made up to thousand identical chips. Next, the wafers are sent to sort or probe, where electrical tests identify the individual dies that are not likely to be good when packaged. The probed wafers are sent to an assembly facility where the dies with a reasonable quality are put into an appropriate package. Finally, the packaged dies are sent to a test facility where they are tested in order to ensure that only good products are sent to customers. At certain process steps, it can happen that jobs, wafers, or dies are processed in a way that they become damaged. In some situations, rework is possible to repair the wafer. When rework is not allowed, the useless wafers are called scrapped material.

As a result, the performance measures of Semiconductor Manufacturing Systems (SMSs) are the percentage (yield) of dies that meet their electrical specifications and the cycle time that is defined as the time needed for a lot of wafers (a job) to travel through the semiconductor manufacturing system including queue time, processing time, and transit time. Therefore, apart from implementing changes in the products and the fabrication procedures, improvement in productivity is achieved by increasing utilization of resources on the fab floor, effectively managing the work-in-process (WIP) and reducing cycle times.

1.2 Overview of the Thesis

Here, in this thesis's project, we are engaged in research of controlling average cycle times in interconnection with the utilization of tool groups. Our goal is to compute the average cycle time for a product family crossing the flow line from the start through the finish area. This is done by implementing Queueing Theory methods to model a re-entrant flow line of a discrete-event system, Chapter 2. In addition, the manufacturing environment is highly stochastic whereas the number of wafers in a job may change as the job moves through the fab. Also, tool breakdowns, preventive maintenance, operator unavailability, setup requirements and other factors combine to make the problem stochastic. Consequently, the first method related to decision-making for stochastic problems is queueing theory. We will analyze its aspects like the use of the traffic equations (traffic rates & variability) and the characteristics of different tool types in Chapter 3.

In our attempts to implement a computational model adapted to queueing theory, we were relied on the Connors et al.'s research work ([3]) of Chapter 3, and we succeeded in modelling every aspect of the theoretical analysis required and incorporated into analytical models. The computational model and some techniques and methods in its coding are analyzed extensively in Chapter 4. The model was developed in the programming environment of MATLAB-MathWorks software.

The computational model that we developed in Chapter 4 has to reflect the physical structure of a real-time re-entrant semiconductor manufacturing facility illustrating the complexity of the semi-conductor industry. This is succeeded implementing a few variants of the Mini-Fab model introduced by Kempf [6]. The significant input data that we are need are extracted from this Five-Machine Six Step Mini-Fab case. Not only the data, but also the structure of our computational model is influenced from the Mini-Fab case. This simulation model suggested by researchers from Intel Corporation and is described as an example of simulation model that contains typical features of a wafer fab (fabrication facility). Hence, from the derived results we evaluate our computational model in Chapter 5.

1.3 Motivation

The main literature in which this thesis's project is based on is the scientific paper of Daniel P. Connors, Gerald E. Feigin, and David D. Yao that was published in *IEEE Transactions on Semiconductor Manufacturing* scientific magazine in August 1996. It is dealt with a Queueing Network Model for Semiconductor Manufacturing. Specifically, an open queueing network model is being developed for rapid performance analysis of semiconductor manufacturing facilities, but it is differentiated from other approaches and researches in the detailed ways that the different tool groups, which are found in a semiconductor wafer fabrication, are modeled. The paper's research was my graduate work, but the perspective of this is described below.

The Production Organization Laboratory of the Department of Mechanical Engineering at University of Thessaly participates in the research project Productive4.0 "Electronics and ICT as an enabler for digital industry and optimized supply chain management covering the whole product lifecycle". This project started on May 1, 2017, has duration of three years and it is funded by the Consortium Electronic Components and Systems for European Leadership Joint Undertaking (ECSEL JU) under project contract No. 737459. This consortium is funded by the European Research and Innovation Program H2020 and National Funding Authorities. Project coordinator is

Infineon Technologies and many other industrial and research partners are involved. Particularly our Production Organization Lab cooperates under the guidance of the Robert Bosch GmbH (Bosch) Company. The main purpose of the project is a significant improvement in digitalization of European industry through electronic and Information and Communication Technology (ICT). This project is the largest European research effort to date under the initiative of automation and data exchange in Industry4.0 industrial technologies ([1]).

So, in this context, the development of our model has the prospect to be implemented in the semiconductor supply chain of Bosch in its factories in Reutlingen (Germany). With the corresponding adjustments, there is strong potentiality that the physical system's structure of the Frontend part of its semiconductor plant can be described better and more thoroughly with our implementation model in order to manage significant impact on the improvement of the line's performance.

Chapter 2

Queueing Theory for Semiconductor Manufacturing Systems

Before we continue to analyze the Connors et al.'s [3] model we will make a reference in this Chapter to the theoretical background of the Queueing Theory that implemented in semiconductor manufacturing systems (SMSs). A literature review is conducted at first in Section 2.1 in order to make a connection with the basic Queueing Models for Toolsets which are introduced in Section 2.2. In Section 2.3 the Queueing Network and the implemented Decomposition method with its properties are discussed.

2.1 Literature Review

Performance measures evaluation is very crucial part during the analysis of high-complex semiconductor manufacturing systems, such as semiconductor fabs, especially for the thorough modeling and management of such stochastic systems. Queueing theory is the most well-known method to analyze stochastic networks in conjunction with the different types of toolsets there have been in semiconductor fabs.

Our focus in this Chapter is on analytical methods, which are primarily based on queueing models. One of the earliest applications of queueing network models for the performance evaluation of semiconductor fabs is Connors et al. [3]. They incorporated

various tool models as well as the functions of scrapping and rework into the fab queueing network. Compared with simulation, they showed that the predictability of their model was accurate in that 47 out of 72 toolsets falling in the 95% confidence interval. In this thesis work, we describe the methodology of Connors et al. [3] and report on our experience from implementing it.

Since the early work of Connors et al. [3], other queueing network approaches for modeling Semiconductor Manufacturing Systems (SMSs) have been developed. Much of the work on queueing network modeling for the performance evaluation of large-scale manufacturing systems has relied on decomposition approaches, because they are the only realistic methods for analyzing such systems. The first decomposition method, called queueing network analyzer (QNA), was developed by Whitt [16]. He identified three basic network operations to capture the propagation of variance of flow in the network as his (QNA) model characterizes the arrival processes by two or three parameters analyzing the individual nodes separately. Meng et al. [8] extended the decomposition model to incorporate batch tools with operation-specific batch size instead of machine-specific batch size. Their experiments showed that significant improvement in cycle time estimation was achieved compared with the QNA model (Whitt, [16]).

Despite the development presented above, practical applications of queueing theory in SMSs are rather limited, as Shanthikumar et al. [12] pointed out in their review paper. They suggested that in order to improve the quality of queueing network modeling approaches, researchers should pay more attention to the collection and analysis of fab data, revisit the independence assumptions regarding the sequence of interarrival and service times, incorporate control policies with specific tool configurations, and explore diffusion approximation methods of reentrant queueing networks to accurately quantify the performance of control policies that address instabilities in reentrant process flows. Also, Grosbard et al. [4] developed a decomposition without aggregation (DWOA) method to model a Semiconductor Manufacturing Queueing Network (SMQN) and yield cycle time approximations.

2.2 Queueing Models for Toolsets

In order to translate the fab environment into a queuing network, it is firstly modelled lots (jobs) as the basic customers in the network, so all lots of the same customer class (product family) follow the same route (process sequence). This is helping in order now each toolset in the fab to be modeled as a queue with multiple and every operation to be described as a different customer service at a queue with its own service time distribution. A queue may perform multiple operations yet every operation can be performed on a single queue. The tool groups are going to be characterized in the fab as queueing systems with their queue properties because this is going to help viewing the production line as a queueing network upon which the appropriate network decomposition process will be applied.

To designate a queue model suited for a toolset in a fab they are quoted some characterization properties as they displayed in [4]:

- Service Process: The actual time taken to perform a single process on a job/wafer is typically constant. However, different operations performed on the same tool may require different processing times and may require setup times between them. As a result the processing time distribution is discrete and the queue service distribution is of a general distribution, G.
- Arrival/Departure Processes: Lots (jobs) arrive at a toolset follow a deterministic production route. Rework and metrology inspection operations divert some of the lots from the main route, requiring probabilistic routing modeling. Due to the highly re-entrant nature of fabs, arrivals to a single toolset may come from various other toolsets requiring superposition of different arrival processes. Hence the inter-arrival process distribution is considered general, G, or more specifically ∑G.
- Service Disciplines: The service policy is First-Come-First-Served (FCFS), or First-In-First-Out (FIFO).
- Number of Servers: The number of servers at each queue is the number of tools or machines within the represented toolset or tool group. In my work, we come

across with one and multiple machines in the tool groups that service customers (jobs).

- Incapacitation Events: The incapacitation of tools in a toolset, due to breakdowns and Preventive Maintenance (PM), is a primary factor for a large portion of the production flow variance. Considering incapacitation events only in the approximation of queue's performance measures fails to estimate the impact of these on the arrival variance to downstream toolsets in fabs. The incapacitation events of each toolset are modelled as arrivals of different "incapacitating" type customers to the queue, occupying a single server for the duration of its service time. To avoid dealing with truncated service, server incapacitation events are modeled as a non-preemptive high-priority customer class.

General, service processes on toolsets are subject to high variation. Usually, fabs are constructed as job-shop systems, whereby similar tools are grouped together as one toolset to perform similar processes. High variation is introduced through multiple products and operations on the same toolset, accompanied with requirements of cascading and setups. Scheduled and nonscheduled downtimes also add to the toolset unavailability and variability. Moreover, the lot arrival process of a toolset is a stochastic process with inter-arrival times usually following certain distributions. Based on tool configuration and process requirements, processing times are also stochastic. Lots that have already arrived but cannot be processed immediately are placed in the waiting queue. So, Queueing theory takes information of the arrival process and the service process and predicts the average waiting time or queue size in steady states.

From existing literature models answered in semiconductor manufacturing (refer to [12]) the average cycle time can be calculated exactly as

$$CT = s + t_q \qquad (\mathbf{2.1})$$

where s is the average service time, λ is the arrival rate, $\rho(= \lambda * s)$ denotes the tool utilization, t_q is the average waiting time in queue, and CT is the average cycle time. But, for realistic queueing models, exact analytical solutions are very difficult to achieve. Researchers usually use approximations to estimate cycle time because of the stochasticity of process and queue times. Accuracy of approximation models depends

greatly on the actual distribution of the inter-arrival times and the service times. It also assumed that the arrival process and the service process are independent.

At next level of analyzation, the second term of $(\mathbf{2.1})$ can be analyzed for its easy comprehension. Various approximations may exist for the same queueing model, although the most widely used approximation developed by Kingman is the $G/G/1$ model where the service time has general distribution. It is given by

$$t_q = \left(\frac{\rho * \left({C_a}^2 + {C_s}^2\right)}{2 * (1-\rho)}\right) * s \qquad (\mathbf{2.2})$$

where ${C_s}^2$ is the squared coefficient of variation (SCV) of the service times and ${C_a}^2$ the SCV of the inter-arrival times. Similar notation for the Squared Coefficient of Variation (SCV) is υ, υ_s for the SCV of service times, υ_a for the SCV of inter-arrival times.

Continuing at the multi-machine station types, with given distribution, any $G/G/m$ system, where m means the number of identical machines, can theoretically be estimated with exact analytical formula. Based on Kingman's $G/G/1$ approximation and the $G/G/m$ approximation, Hopp and Spearman developed the following approximation

$$t_q = \left(\frac{{C_a}^2 + {C_s}^2}{2}\right) * \left(\frac{\rho^{\sqrt{2*(m+1)}-1}}{1-\rho}\right) * s \qquad (\mathbf{2.3})$$

In general, the higher the variance in either interarrival times or service times, the longer the waiting time. If the number of machines increases, even though the average utilization of each machine keeps the same, the average waiting time will decrease dramatically. It is also worth mentioning that $G/G/m$ assumption is reasonably close to reality for some toolsets of SMSs. The approximation that is used in my work such as at Connors et al. [3] is relatively accurate but much more complicated.

Finally, the last model of toolset type is for the batch tools. $G^X/G/m$ models where X represents the batch size distribution of arrivals presented. Specifically for a SMS the Connors et al.'s approach uses a model with flexible and upper limited batch sizes. This last category of tool type is important to be modelled because up to one-third of the steps in semiconductor manufacturing involve batch processing. We proceed now

in the description of the structure that the model is discernible and the method that it is used to solve such systems.

2.3 Queueing Network and Decomposition Approach

Toolsets in SMSs are not independent from each other. The arrival process of a toolset is constructed from the departure processes of its upstream toolsets. In fact, all toolsets are linked together stochastically by process flows. The system can be modeled as a queueing network, in which the toolsets are the nodes and the flows are the arcs. The toolsets and lots are modeled as servers and customers, respectively.

Here is a rough description of the model as Whitt presents in [16]: There is a network of nodes and directed arcs. The nodes represent service facilities (servers) and the arcs represent flows of customers (jobs or wafers). There is also one external node, which is not a service facility, representing the outside world. Customers enter the network on directed arcs from the external node to the internal nodes, move from node to node along the internal directed arcs, and eventually leave the system on one of the directed arcs from an internal node to the external node. The flows of customers on the arcs are assumed to be random so that they can be represented as stochastic processes. If all servers are busy at a node when a customer arrives, then the customer joins a queue and waits until a server is free. When there is a free server, that customer begins service, which is carried out without interruption. Successive service times at each node are assumed to be random variables, which may depend on the type of customer but which otherwise are independent of the history of the network and are mutually independent and identically distributed. After the customer completes service, he goes along some directed arc from that node to another node. The customer receives service in this way from several internal nodes and then eventually leaves the network. A picture of a typical network (without the external node) is given in Figure 2.1.

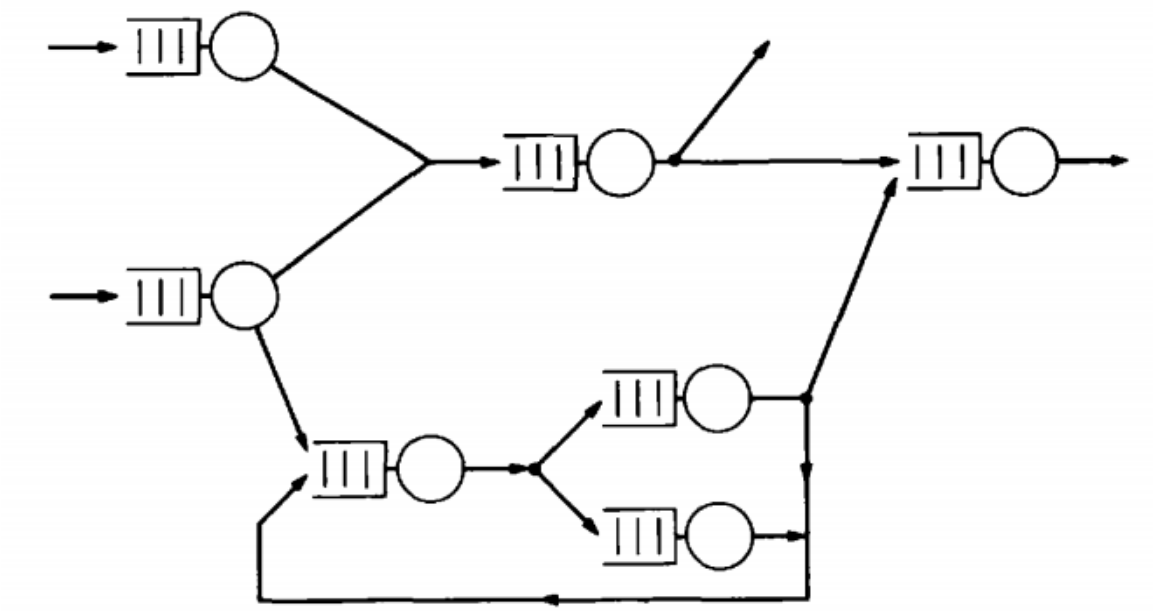

Figure 2.1: An open network of queues (source: Whitt, [16])

An important feature of the model is that there may be flows from node j to node i, as well as flows from node i to node j. Flows from node j to node i represent different customers than the customers that flow from node i to node j. The network is open rather than closed because customers come from outside, receive service at one or more nodes, and eventually leave the system with replenishment after finishing the last processing step. This was an assumption about the model as well as that there are no capacity constraints. There is no limit on the number of customers that can be in the entire network and each service facility has unlimited waiting space.

The general approach is to represent all the arrival processes and service-time distributions by a few parameters. The congestion at each facility is then described by approximate formulas that depend only on these parameters. The parameters for the internal flows are determined by applying an elementary calculus that transforms the parameters for each of the three basic network operations: superposition (merging), thinning (splitting), and flow through a queue (departure). These basic operations are displayed in Figure 2.2.

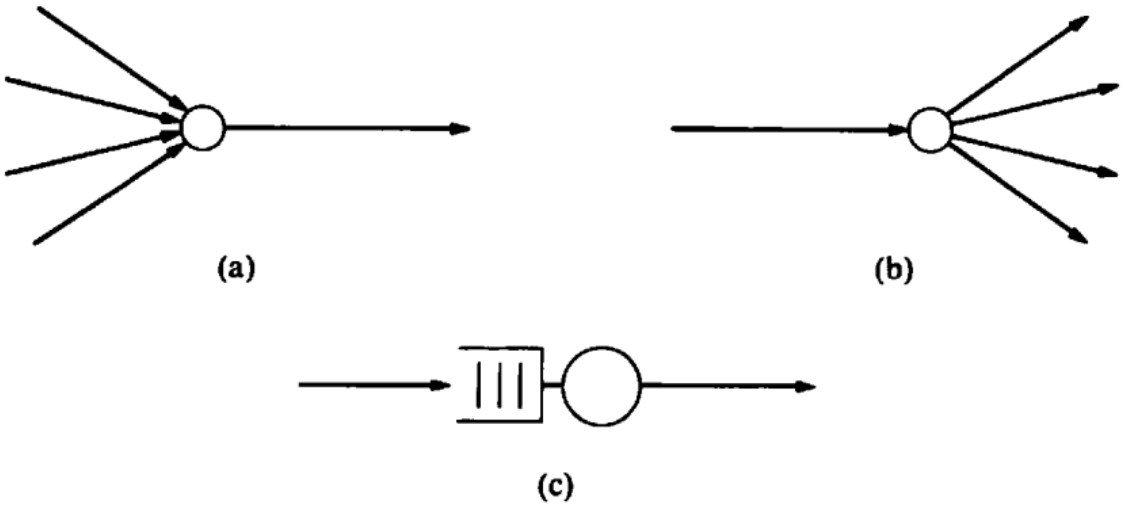

Figure 2.2: Basic network operations, (a) Superposition or merging, (b) Decomposition or splitting, (c) Departure or flow through a queue (source: Whitt, [16])

In simple words, the arrival processes and the service times are characterized from two parameters, one to describe the rate and the other to describe the variability. For the service times, the two parameters are the first two moments. For the arrival processes, the parameters are associated with renewal-process approximations. The first two parameters are equivalent to the first two moments of the renewal interval (interval between successive points) in the approximating renewal process. The equivalent parameters we use are the arrival rate λ, which is the reciprocal of the renewal-interval mean, and the squared coefficient of variation C^2, which is the variance of the renewal interval divided by the square of its mean. For the calculation of these first-two moment's quantities, we describe, in the next paragraphs, how the approximated SCVs are estimated from decomposition method processes.

Primarily, researchers have extended Jackson's model to incorporate general service times and inter-arrival times as the renewal process with general distributions. Such model can be analyzed by the decomposition method. The method assumes: (a) the nodes (toolsets) can be treated as being stochastically independent and (b) the input to each queue is the renewal process characterized by the mean and variation of the lot inter-arrival time distributions. The input and output of each node is linked through lot routings. The following processes at each node are considered: queue output process,

output splitting, and output merging. The traffic and variations are then carried over through the network structure. So, queuing network decomposition method works by decomposing the queuing network into a set of separate queues having independent arrival processes and approximating the performance measures of those queues. The process is comprised of three steps (the second step is the core of network decomposition method):

1) Calculation of the first moment of the inter-arrival durations to each queue in the network.
2) Approximation of the second moment of the inter-arrival durations to each queue in the network.
3) Approximation of queue performance measures based on the above first two moments.

The decomposition method can be classified into two classes, decomposition with aggregation (DWA) and decomposition without aggregation (DWOA) (refer to [4]). In order to obtain approximations of the SCVs of the general arrival process at each queue in the network, DWA methods apply the three basic linear operators of superposition, departure and splitting, as they mentioned above, that approximate the SCV of the aggregated inter-arrival, inter-departure distribution to and from the queue and every departure process from the queue, respectively. The three operators are used to create a set of linear equations that yields an approximation of SCV of the inter-arrival distribution of the aggregated arrival process at each queue (aggregated over all arrival processes to the queue). To sum up, my research work focuses on the DWA method as Connors et al. and Whitt ([3], [16]) have done in their models and so we are going to analyze the Sections of traffic variability in this scheme.

Chapter 3

Analysis and Description of the Queueing Network Model

In this Chapter we are going to present you the theoretical basics of the Connors et al. paper's research ([3]) on which we supported the work we have done on our computational model of the next Chapter 3. The paper describes an open queueing network model that is specifically developed to provide reliable performance estimates for complex semiconductor fabs, as the use of queueing models for estimating performance of manufacturing systems is not new. The chapter begins with the basic differentiation in analysis of the paper's research work (Section 3.1) in comparison to similar research works. Next, in Section 3.2 we format-standardize the operations of a process sequence and we discuss the details of the scrap and rework processes and their impact on routing and job-size distributions. Then, in Section 3.3, the relative traffic equations are listed which will help us understand the flow rates between the operations and tool-groups. In Section 3.4, the characteristics of the tools' types, such as the mean queueing delay and the utilization of each toolset type, as the analyzation of the incapacitation events, which is equally important, are presented. Last but not least, we give a formula for the estimation of average cycle time for the products in Section 3.5.

.

3.1 Semiconductor-Specific Model Features

Considering that a flow line is composed from toolsets in a row and its toolset has its own queue that accepts customers (lots or jobs, they represent the same entities in the system) with some arrival process, then this system can be decomposed in such systems, which are the toolsets (or tool-groups, it's the same term). So, in order to analyze the queueing network we will follow the decomposition based approximation approach. That is, each node in the network is analyzed separately, with a set of renewal input processes, characterized by the first two moments of their respective inter-arrival times. These two moments capture the interdependency among the nodes, and they are obtained through solving two sets of linear equations, for traffic rates and traffic variability, respectively. It is fair to say that this decomposition approach has by now become a standard and well-accepted technique at the network level. How well it works depends largely on how well it captures the details of the specific system under study, in particular, how well the individual queueing models at the node level capture the operating features of the manufacturing processes. It is exactly in this regard that this model differs quite significantly from what exists in the literature.

The model of Connors et al. [3] includes the following features:

- A detailed analysis of the scrap and rework processes to capture the effect of variable job sizes on the workload and utilization of the tool groups;
- A detailed modeling of different types of tools that are typical in semiconductor fabs, including the use of batch-service queues and tandem queues;
- A careful treatment of the "incapacitation" events, those that disrupt the normal process at the tools, using priority queueing results and other related techniques.

These are the main differences among other research works. Briefly, this model is taking into account the rework characteristics and the special treatment of the incapacitation events at the node level, in conjunction with the features of the different tool types and its properties in different cases.

In this graduate work I have been involved with two types of tool groups, the single wafer tools, including the simple server (single tool - one machine) and multiple

tools (parallel identical machines), and the batch tools. In aim to calculate the average cycle time of each product family that flow in the line and process in each tool type, we first have to estimate the mean queueing delays for these different tool types. More precisely, we use exact priority queueing formula for the single wafer tools with a simple server and an approximate analysis based on modified service times for single-wafer tools with multiple machines. For the batch tools we distinguish two cases from which we calculate queueing delays and utilizations. In constructing the model, the characteristic quantities of mean and variance of processing time and utilization of each toolset type are used.

3.2 Probabilistic Model Formulation

Because the individual customer's routes that are represented by the job's routings in a semiconductor network fab are not deterministic, but random (stochastic) routes depending on probabilities, the derivation of the job-size distribution is essential to be determined in order to have a knowledge of the lot size in wafers that flow in the network in each state of the system. So, in Section Part 3.2.1, we have an introduction of what configurations and properties of semiconductor fabs we have to be aware of in order in Section Part 3.2.2 to set the appropriate probabilities and finally in Section Part 3.2.3 to define the wafers in a job (distribution) that flows in the fab.

3.2.1 Pre-Analysis

The first step of our analysis is to format the operations in a semiconductor fab into those of a queueing network. We associate each tool group in the fab with a node in the queueing network. Recall that wafers circulate in the fab in groups referred as jobs, so, wafers and jobs are the basic entities that travel among the nodes and are processed by the tools there. In general, there is a set of distinct product families and a set of distinct toolsets. Each product family corresponds to the distinct set of wafers and jobs and has its own process sequence of operations. A process sequence consists of a set of process

steps-operations that are to be performed on jobs belonging to that product family. According to the model, the operations have an ascending sequence, i.e., the first operations matches with the first product family, the second's with the 2^{nd} product family, etc. In this part of analysis the main study structure is the operations because of the need to define how many wafers or fraction of the job continue to the next process step of process sequence without problems. This work's plan will help us determine the size's distribution of jobs that are scrapped or goes to the next process step and additionally the job-size distribution that are sent for rework operations among with the times of reworking.

The process sequence consists of a nominal part and a set of rework sequences. The nominal process sequence is an ordered list of operations to be performed on the job, if no rework needs to be performed on it. The total operations of a product family f are denoted by N^f. More precisely, the rework sequences specify the operations that are to be performed when a job is sent for rework. After any operation we evaluate the status of the job, i.e. a job may be entirely or partially scrapped or can be sent for rework (entirely or partially). When a job or part of a job is sent for rework, a set of operations is performed on it and then it reenters the nominal process sequence at or before the point at which it was sent for rework. If only a portion of a job is sent for rework, the remaining portion waits for the rework portion to complete its rework. When the rework portion of the job eventually returns to the point at which it was sent for rework, the two jobs are merged into one and this merged job continues along its nominal process sequence. Note that it is possible for a job or a portion of a job to be sent for the same rework sequence multiple times. In general, then, every operation has at most one nominal successor operation and at most one rework successor operation.

The above of course is much in theory as well as there are differences in practice. The formulation that was displayed previous cannot be implemented in absolute degree in the supply chains of the manufacturing industries. That is because companies have standardized their operations in their production lines and it is very difficult to change some characteristics of the common facilities. Additionally, it has to be examined the

effect changes that will be occurred of these implementations to existing line's performance. Nevertheless, this is time consuming as well money investments required.

For instance, in contradiction with the model's formalism, there are some important differences in characteristics of existing flow lines in big companies-industries, like Bosch. Characteristically, their product families contain products with common process sequences and each process of the sequence has a recipe with certain process phases. Usually, there are some preparatory operations before or at the end of the main sequence while 10-15% of the total operations correspond to Testing Operations. Testing operations are the last operations in the recipe of a sequence's process. For the Testing, it is used one wafer every 10 wafers as a sample. Checking only the sample wafer, it is decided if the lot (job) will proceed in next station or will be scrapped or it is needed rework. Obviously, if the sample wafer is ok the lot will proceed, even though bad parts will proceed too. Moreover, for the rework operations, depending on the symptom-defect that is detected from the control check the wafers follow specific rework process sequence. If one wafer goes for rework then the whole job goes for rework. As you notice, the formulation that develop in a theoretical model differentiate from existing supply chains and production lines, so, do not be surprised of disagreements between them at the rest of the paper's research work.

3.2.2 Scrap and Rework Probabilities

In this Section Part we are going to define the probabilities that an individual wafer is scrapped or sent for rework after each operation. Calculating these probabilities (P_{ws}, P_{wrw}) it will help us to specify the routing probabilities in the network and, as a result, to find the unknown job-size's distributions. Recall that P_{ws} is the probability that an individual wafer is scrapped after an operation and P_{wrw} is the probability that an individual wafer is sent for rework after an operation.

After a job completes an operation, it may either be partially or completely scrapped, partially or completely sent for rework, or may proceed to the next operation in its nominal process sequence unaffected by scrap and rework. By being scrapped, we

mean that wafers in the job are eliminated from the job and discarded. If a job is completely scrapped, then all wafers in that job are discarded. If it is partially scrapped, then a subset of the wafers in the job is lost. By the same token, if a job is completely reworked, all wafers in the job are sent for rework, whereas if a job is partially reworked, a subset of the wafers in the job is sent for rework.

It is assumed that scrapping and rework decisions are made sequentially. More specifically, after each nominal operation, it is decided in the following order:

1) whether or not the entire job should be scrapped;
2) which, if any, of the wafers in a job are to be scrapped;
3) whether or not the entire job, with all the remaining wafers in it, should be sent to rework;
4) which, if any, individual wafers need rework.

In the scrapping and rework decision hierarchy, we always examine the job first, before examining any individual wafer. This is because all the wafers in the same job often share a common quality problem (in addition to their individual problems). For instance, the entire job has to be reworked or scrapped because of contamination or physical damage to the container that carries the job. Note that an entire job can be scrapped or sent to rework for reasons other than that all the individual wafers in the job have to be scrapped or reworked. Hence, the following probabilities for each operation in the nominal process sequence are identified,

P_{js} = probability a job is scrapped after an operation

$P_{es|js}$ = probability a job is entirely scrapped given it is scrapped

$P_{ps|js}$ = probability a job is partially scrapped given it is scrapped, $P_{ps|js} = 1 - P_{es|js}$

$P_{ws|ps}$ = probability a wafer is scrapped given it .is in a job that is partially scrapped

$P_{rw|nes}$ = probability a job is reworked given it is not entirely scrapped

$P_{erw|rw}$ = probability a job is entirely reworked given it is reworked

$P_{prw|rw}$ = prob. a job is partially reworked given it is reworked, $P_{prw|rw} = 1 - P_{erw|rw}$

$P_{wrw|prw}$ = probability a wafer is reworked given it is in a job that is partially reworked.

From the above probabilities, Connors et al. [3] calculate the probabilities that a wafer is scrapped or sent for rework after a given operation.

$$P_{ws} = P_{js} * \left[P_{es|js} + P_{ps|js} * P_{ws|ps}\right] \quad (\mathbf{3.1})$$

$$P_{wrw} = \left[1 - P_{js} * P_{es|js}\right] * \left[P_{rw|nes}\right] * \left[P_{erw|rw} + P_{prw|rw} * P_{wrw|prw}\right] \quad (\mathbf{3.2})$$

To understand how expressions $(\mathbf{3.1})$ and $(\mathbf{3.2})$ are derived, we constructed three probability tree diagrams shown in Figures 3.1-3.3. The middle diagram in Figure 3.2 is used to derive an expression for the auxiliary probability that a job is not entirely scrapped, $(P_{nes} = 1 - P_{js} * P_{es|js})$, which is used as an input for the derivation of P_{wrw} shown in Figure 3.3.

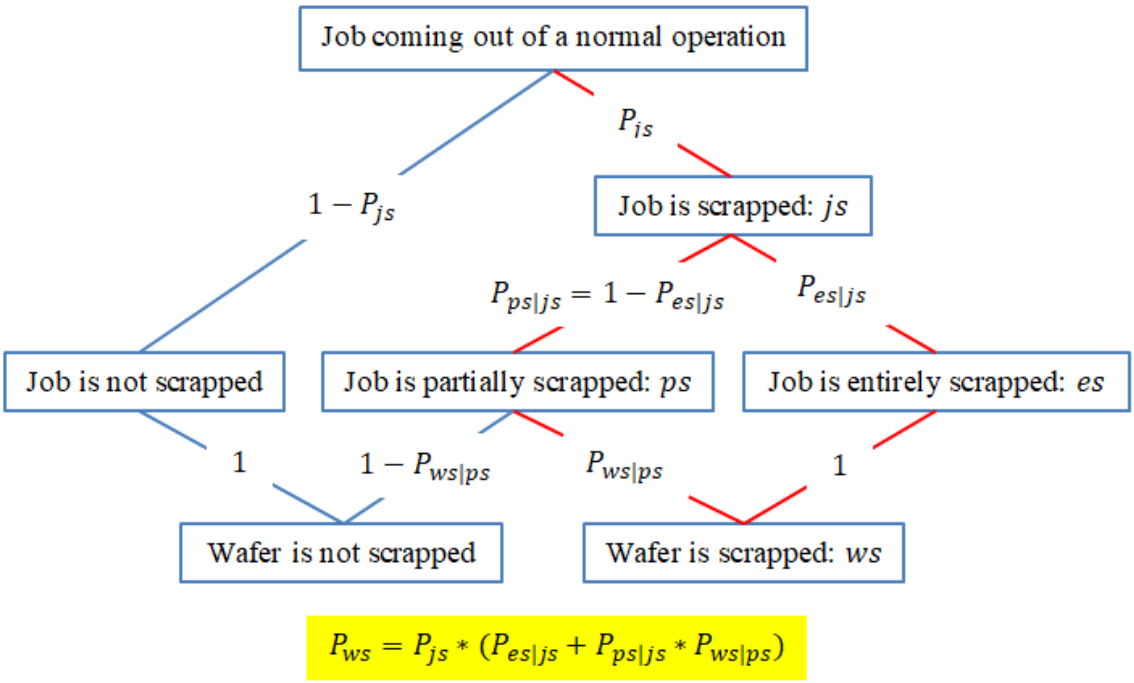

Figure 3.1: Derivation of probability P_{ws}

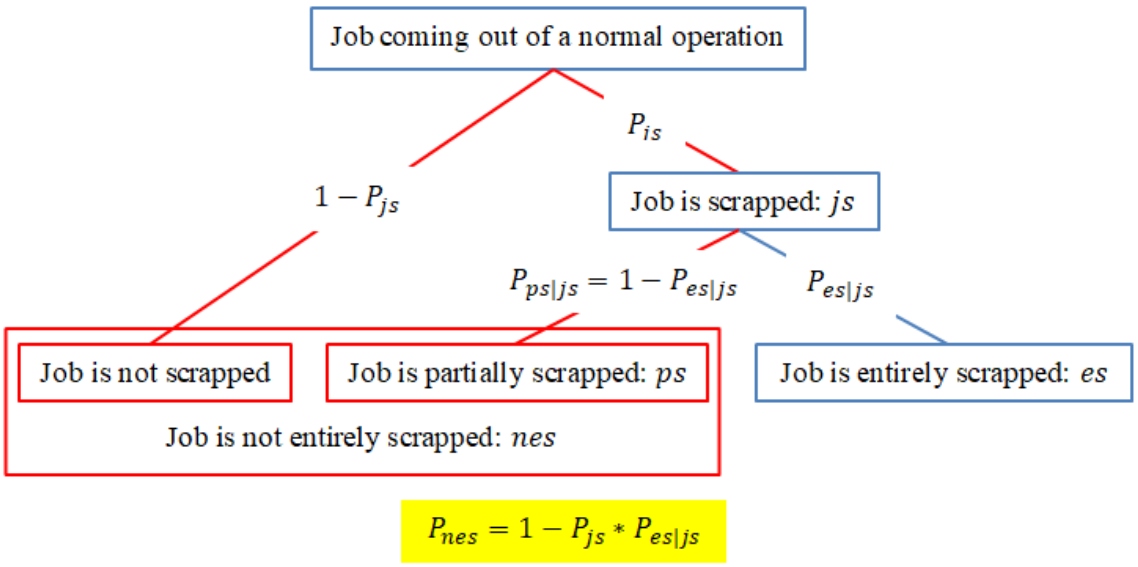

Figure 3.2: Derivation of probability P_{nes}

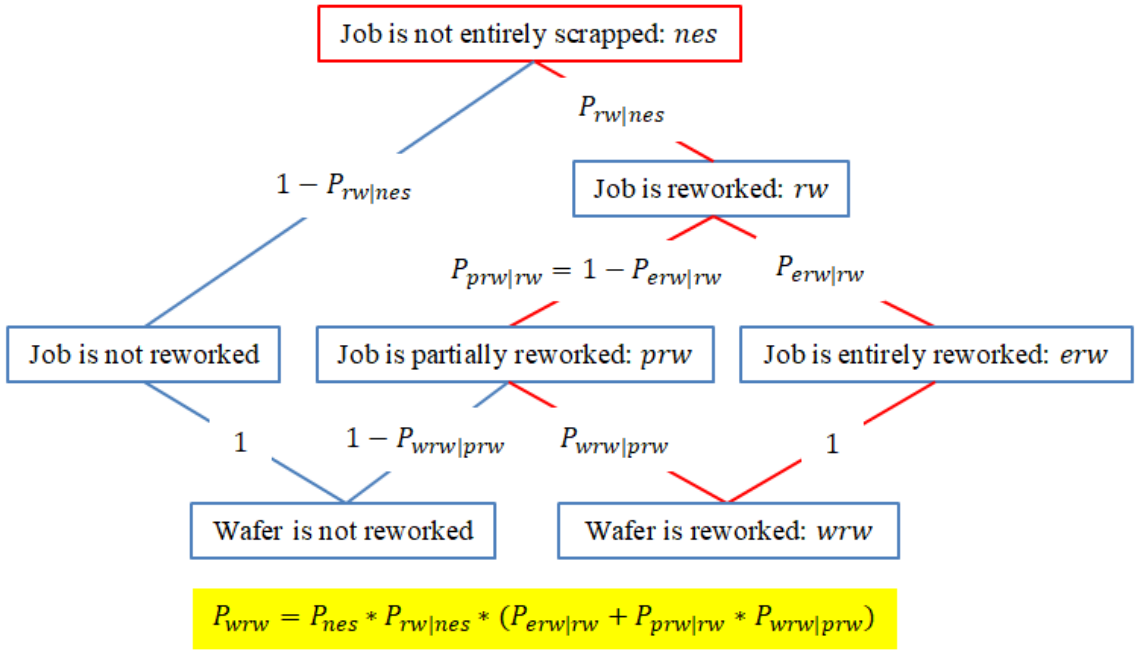

Figure 3.3: Derivation of probability P_{wrw}

Observing more carefully the second equation $(\mathbf{3.2})$ for the calculation of P_{wrw} we find out that this is wrong. This is because the first pair of brackets, which specifies the probability that a wafer is not scrapped is partially correct. It is implied that a wafer may be sent for rework if the job it belongs to is not entirely scrapped. We believe that

this is only partially correct and results in overestimating P_{wrw}, because a wafer may be sent for rework if the wafer itself (not only the jobs it belongs to) is not scrapped. Alternatively, according to the situations that a job or a wafer can pass or end up, the derivation type of P_{wrw} includes the concept of the state that a job is not entirely scrapped. From this state it is evidenced that a wafer can be sent for rework but also it can be scrapped. The latest case is not taken into account in type $(\mathbf{3.2})$ and therefore we think we should replace the probabilistic term of (P_{nes}) with $(1 - P_{ws})$ which incorporates the occasion that a wafer can be scrapped when a job is not entirely scrapped. Thus, given the expression $(\mathbf{3.1})$ for the probability that a wafer is not scrapped, we modify the equation $(\mathbf{3.2})$ that gives the probability that a wafer is sent for rework (P_{wrw}) into the following expression:

$$P_{wrw} = \left[1 - P_{js} * \left(P_{es|js} + P_{ps|js} * P_{ws|ps}\right)\right] * \left[P_{rw|nes}\right] * \left[P_{erw|rw} + P_{prw|rw} * P_{wrw|prw}\right] \Rightarrow$$

$$P_{wrw} = [1 - P_{ws}] * \left[P_{rw|nes}\right] * \left[P_{erw|rw} + P_{prw|rw} * P_{wrw|prw}\right] \qquad (\mathbf{3.3})$$

3.2.3 Job-Size Distributions

Due to scrapping and rework, the number of wafers contained in each job circulating in the fab is a random variable. For our analysis, we need to calculate the job-size distribution at each operation. Connors et al. [3] supposed the following random variables with their distributions,

- $J \sim F(k)$: Distribution of size of jobs arriving to an operation k
- $\tilde{J} \sim F_{\sigma}(k)$: Distribution of size of jobs arriving to the nominal successor operation of the operation k
- $\hat{J} \sim F_{\sigma_{rw}}(k)$: Distribution of size of jobs arriving to the rework operation of the nominal operation k
- n_{max}: maximum number of wafers in a job

(**Note**: $\sigma(k)$ = the nominal successor operation of operation k, $\sigma_{rw}(k)$ = the rework successor operation k, $P(J = n) =$ the probability that a job arriving to a certain operation has n wafers in it).

To calculate the $F_{\sigma}(k)$ distribution given $F(k)$ we use the following expressions:

$$P(\tilde{J} = n) = P(J = n) * \left(1 - P_{js}\right) + P_{js} * P_{ps|js} * \sum_{i=n}^{nmax} P(J = i) * P(exactly\ i - n\ wafers\ scrapped\ from\ job)\ , n = 1, \dots, n_{max} \quad (\mathbf{3.4})$$

$$P(\tilde{J} = 0) = P_{js} * P_{es|js} + P_{js} * P_{ps|js} * \sum_{i=1}^{nmax} P(J = i) * P(exactly\ i\ wafers\ scrapped\ from\ job) \quad (\mathbf{3.5})$$

Because this procedure is an iterative algorithm, the above types can be written in a more readable form in order to be emerged the retrospective process which is implied. This is accomplished by expunging the second random variable $\tilde{J}$ with distribution F_{σ}, using only the random variable J with distribution F and letting k be the index of repetitions that correspond to each operation.

Suppose that wafers are scrapped according to a Bernoulli trial with parameter $p \equiv P_{ws|ps}$. In the sum, the probability density function of the Binomial distribution form is used.

$$P(exactly\ (i - n)\ \text{of}\ i\ \text{wafers are scrapped}) = \binom{i}{i - n} p^{i-n}(1 - p)^{n}$$

The above distribution for $\tilde{J}$ holds if jobs with no wafers continue to circulate in the fab. However, since empty jobs are removed from the fab, we are really interested in the conditional distribution

$$P(\tilde{J} = n \mid \tilde{J} > 0) = \frac{P(\tilde{J} = n)}{1 - P(\tilde{J} = 0)}$$

Now, to calculate the distribution $F_{\sigma_{rw}}$ for jobs that are sent for rework, we proceed in a similar manner, except we use F_{σ} as the a priori distribution. The reason is

that, as we mentioned earlier, the decision to send a job or a portion of a job for rework comes after the scrapping decisions are made.

$$P(\hat{J} = n) = P(\tilde{J} = n) * P_{rw|nes} * P_{erw|rw} + P_{rw|nes} * P_{prw|rw} * \sum_{i=n}^{nmax} P(\tilde{J} = i) * P(exactly\ i\ wafers\ sent\ for\ rework), n = 1, \dots, n_{max} \quad (\mathbf{3.6})$$

$$P(\hat{J} = 0) = P_{rw|nes} * P_{prw|rw} * \sum_{i=1}^{nmax} P(\tilde{J} = i) * P(exactly\ 0\ of\ i\ wafers\ sent\ for\ rework) \quad (\mathbf{3.7})$$

where it is assumed that wafers are sent for rework according to a Bernoulli trial with $p \equiv P_{wrw|prw}$, i.e.,

$$P(exactly\ n \text{ of } i \text{ wafers are scrapped}) = \binom{i}{n} * p^n * (1-p)^{i-n}$$

$$P(0 \text{ of } i \text{ wafers are sent for rework}) = (1-p)^i$$

Again, since no rework jobs contain zero wafers, we are really interested in the conditional distribution

$$P(\hat{J} = n \mid \hat{J} > 0) = \frac{P(\hat{J} = n)}{1 - P(\hat{J} = 0)}.$$

The second formula for $P(\hat{J} = 0)$ specifies that any of the wafers in the reworked job will go for rework. This means that despite the control for the job and the decision to go for rework, though any of the wafers in it are about to go for rework after the check, i.e. none of the wafers in a job sent for rework must need rework, given that the job needs to be partially reworked. Therefore, all wafers of the reworked job are not reworked and they will proceed to the nominal operation without rework. Under this condition, we believe that the calculation type of $P(\hat{J} = 0)$, equation $(\mathbf{3.7})$, is wrong, or else, is partially correct and results in underestimating $P(\hat{J} = 0)$ because the number of wafers in a job sent for rework can also be zero if the entire job is not sent for rework to begin with.

The probability of this event is $(1 - P_{rw|nes})$. With this in mind, we modify equation $(\mathbf{3.7})$ into the following expression:

$$P(\hat{J} = 0) = (1 - P_{rw|nes}) + P_{rw|nes} * P_{prw|rw} * \sum_{i=1}^{nmax} P(\tilde{J} = i) * P(exactly\ 0\ wafers\ sent\ for\ rework) \qquad (\mathbf{3.8})$$

The above analysis assumes jobs are sent for rework at a given operation a maximum of once. It can be easily modified by repeatedly applying the above formulas for up to a maximum allowed number of times that a job can be sent for rework. To carry out the computations we also need the distribution of the sizes of jobs initially released in the fab. Normally, they are deterministic.

3.3 Traffic Equations

Traffic equations characterize the arrival processes to the tool groups, in terms of rates and variability. They form a critical link in the decomposition approach relating the overall network to the individual nodes. Therefore, in this Section we indicate the calculus of the parameters which characterize the internal (and external) flows of the network. The calculus is linear for each network operation, so that the internal flow parameters are determined simply by solving systems of linear equations. In Section Part 3.3.1 we focus on the flow rates, which are obtained via the traffic rate equations, just as with the Markov models. In Section Part 3.3.2 we display the corresponding system of linear equations yielding the variability parameters. The analyzation of Traffic Equations will help to understand the complexity in modelling the wafer fab.

3.3.1 Traffic Rates

In this step the total arrival rate to each node is calculated where the main structure of study is the operations and not the tool groups. Let α_k denote the rate at which wafers are released exogenously into the fab (in wafers per time unit) to operation

k. Let λ_k be the overall rate at which wafers arrive to operation k, including both exogenous arrivals and internal transitions. These quantities are the unknowns in the traffic equations. Let $q_{j,k}$ denote the probability that a wafer after completing operation j is routed to operation k. From the analysis in Section Part 3.2.2 we can write an expression for the probability that a wafer departing operation j arrives at operation k, for all operations j and k. So, this expression is:

$$q_{j,k} = \begin{cases} P_{wrw}(j) & if\ k = \sigma_{rw}(j) \\ 1 - P_{wrw}(j) - P_{ws}(j) & if\ k = \sigma(j) \\ P_{ws}(j) & otherwise \end{cases} \quad (\mathbf{3.9})$$

Therefore, given the arrival rates of the exogenous arrival process, the traffic rates λ_k, for all operations k, can be calculated by solving the following set of linear equations:

$$\lambda_k = \alpha_k + \sum_j q_{j,k} * \lambda_j \quad (\mathbf{3.10})$$

The solution to the traffic equations gives us the arrival rate of wafers to each operation. As we mentioned earlier, however, wafers travel through the fab in jobs. With the job-size distribution derived in Section Part 3.2.3, the wafer arrival rate is translated into a job arrival rate by dividing the wafer arrival rate by the average job size. Letting Λ_k be the arrival rate of jobs to operation k, we have

$$\Lambda_k = \frac{\lambda_k}{E[J_k]} \quad (\mathbf{3.11})$$

where $E[J_k]$ is the average job size of jobs arriving to operation k.

3.3.2 Traffic Variability

The reciprocal of the arrival rate to each operation gives the mean of the inter-arrival times. Connors e al. [3] gives a second-order characterization of the inter-arrival times, in terms of specifying the squared coefficient of variation (SCV), i.e., the ratio of the variance (of the inter-arrival times) to the square of the mean. Their analysis has two simplifications:

- All arrivals to each tool group are aggregated into a single process, instead of treating the more detailed breakdown of different product types.
- The departure processes from batch tools (i.e., tool groups that process wafers in batches) is not explicitly dealt with.

Before proceeding in the linear equations of variability, we provide some introductory and auxiliary entities which interpret with clarity the equation's parameters. Denote Γ_g as the total arrival rate of jobs to tool group g, i.e., $\Gamma_g \equiv \sum_{k \epsilon T_g} \Lambda_k$. Also, denote γ_{gh} as the total arrival rate of jobs to tool group h from tool group g. When $g = 0$, γ_{gh} denotes the rate of exogenous arrivals to h. Let c_g denote the number of tools or machines at tool group g. Let ρ_g denote the utilization of tool group g, this input parameter we will calculate it in the later section's parts. Let r_{gh} denote the proportion of jobs leaving tool group g that proceed directly to tool group h. Finally, let $\theta_{gh} \equiv \gamma_{gh}/\Gamma_h$. θ_{gh} represents the proportion of arrivals to tool group h that came from tool group g.

As input parameters for the variability equations are also needed the SCV of the aggregate job arrival process to tool group $g \epsilon G$, ${v_g}^a$, which is the merged stream of all job types arriving to the tool group including both exogenous arrivals and internal transitions, the SCV of the aggregate exogenous arrival processes to tool group g, ${v_g}^e$ and the SCV of the service time at station g for a "generic job" arriving to tool group g, ${v_g}^s$. The calculation of ${v_g}^s$ is specified in the Section 3.4 for each tool type. Therefore, ${v_g}^a$ are the unknowns in the equations that we are going to list below.

Generally, the starting point for studying flows is the arrival of jobs to a single workstation (tool group). At a basic stage, the departures from this workstation will be arrivals to other workstations. Therefore, once we have described the variability of arrivals to one workstation and determined how this affects the variability of departures from that workstation (and hence arrivals to other workstations), we will have characterized the flow variability for the entire line. This is the philosophy of the Decomposition approach. The essential step is to characterize the departures from a workstation as it analyzed in the Factory Physics [5]. In a serial production line without yield loss or rework, the departure features of an (i) workstation corresponds to the

arrival features of the downstream $(i + 1)$ station. So, as it is depicted in Figure 3.4, for the coefficient of variations it holds that:

$$C_a(i+1) = C_d(i)$$

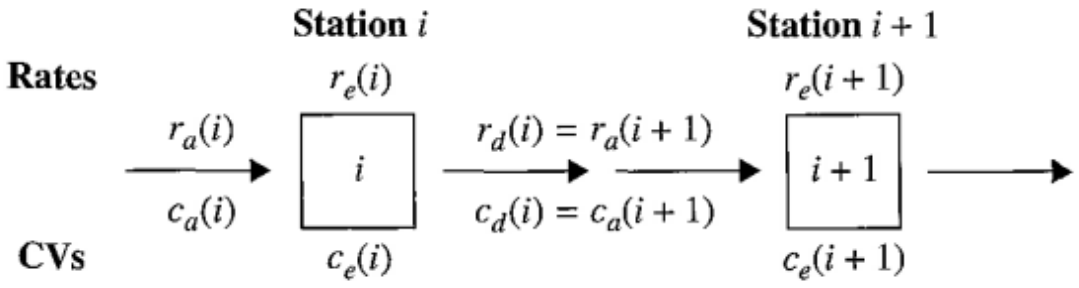

Figure 3.4: Propagation of variability between workstations in series (source: Factory Physics, [5])

However, the main issue to resolve concerning flow variability is how to characterize the variability of departures from a station in terms of information about the variability of arrivals and process times. Specifically, variability in departures from a station is the result of both variability in arrivals to the station and variability in the process times. The relative contribution of these two factors depends on the utilization of the workstation. This relationship can be imprinted by the following expression:

$$\upsilon_h{}^a = \left(1 - \rho_g{}^2\right) * \upsilon_g{}^a + \rho_g{}^2 * \upsilon_g{}^s$$

Hence, following standard techniques that approximate the SCV of merging and splitting point processes, using parameter's variables and additional factors to modify different cases of the wafer fab processes, aggregately, the set of linear equations with $\upsilon_g{}^a$ unknowns is given as follows:

$$\upsilon_h{}^a = a_h + \sum_{g \,\epsilon\, G} \upsilon_g{}^a * b_{gh} \quad \forall\, g, h \,\epsilon\, G \qquad (\mathbf{3.12})$$

where

$$a_h = 1 + w_h * ((\theta_{0h} * {v_h}^e - 1) + \sum_{g \in G} \theta_{gh} * [(1 - r_{gh}) + r_{gh} * {\rho_g}^2 * x_g])$$

$$b_{gh} = w_h * \theta_{gh} * r_{gh} * (1 - {\rho_g}^2)$$

$$w_h = [1 + 4 * (1 - \rho_h)^2 * (v_h - 1)]^{-1}$$

$$v_h = \left[\sum_{g \in G} {\theta_{gh}}^2\right]^{-1}$$

$$x_g = 1 + \frac{\max({v_g}^s, 0.2) - 1}{\sqrt{c_g}}$$

3.4 Tool-Types Characteristics

We have obtained, the rates of jobs arrive to the operations as well as the job-size (in wafers) distributions. In this Section we decompose the network into G isolated queues, one for each tool group. To analyze each queue, we first model the features of the incapacitation events (Section Part 3.4.1) and then we distinguish the two different types of tool groups that we got involved with in our model: the single-wafer tools (Section Part 3.4.2) and the batch tools (Section Part 3.4.3). Following the standard decomposition-based approximations in queueing networks, one key assumption we make in our analysis is that jobs arrive to each tool group following a renewal process with known mean and SCV. The mean is taken to be the reciprocal of the job arrival rate following the traffic rate equations and the SCV comes from solving the traffic variability equations. For each type of tool group, our goal is to derive approximate formulas for the utilization and the mean queueing delay. To this end, we need to derive the following quantities: the first two moments of the processing time for each operation, and the long-run proportion of time that the tool group is incapacitated.

3.4.1 Incapacitation Events

Before proceeding with our analysis for each type of tool group, Connors et al. [3] describe the different types of incapacitation events that can affect the utilization of each tool and define some of the parameters associated with these events. There are three types of incapacitation events that disrupt the operation of a tool: breakdown, preventative maintenance, and major setup. We model each of these events as non-preemptive priority jobs that arrive to each tool group according to renewal processes with known distributions.

Under this assumption, it is conceivable that multiple incapacitation events of the same type could be in queue at the same time. To justify the non-preemptive assumption, Connors et al. [3] argue that a breakdown does not necessarily mean that a tool comes to a sudden halt but rather that it has drifted outside of strict process control specifications. When this happens, the job in process will often finish processing before the tool is taken out of service. To justify the queueing of multiple similar incapacitation events, they argue that the relative infrequency of these events makes the probability of there being more than one incapacitation event in queue at any time rather small.

In general, suppose that there is a set B_g of incapacitation events that affect tools at tool-group g. Denote by Λ_b the arrival rate for incapacitation events of type $b \in B_g$ for tools at tool-group g. Associated with each incapacitation event is a random variable S_b with a known distribution that represents the duration of the incapacitation event. According to the way we model incapacitation events, the steady state proportion of time each tool at tool group g spends incapacitated, ${\rho_b}^{inc}$ is given by

$$ {\rho_b}^{inc} = \sum_{b \in B_g} \frac{\Lambda_b * E[S_b]}{c_g} \qquad (\mathbf{3.13}) $$

This quantity plays decisive role in analysis because it is used extensively in the computational type for the mean queueing delay for each tool type.

3.4.2 Single-Wafer Tools

The single-wafer tools carry out operations wafer by wafer - one wafer at a time. We will analyze the tool groups that consist of single-wafer tools distinguishing two cases for that: the single tools consisting of one server-machine in a tool group ($c_g = 1$) (Case 3.4.2 A) and the multiple tools ($c_g > 1$) (Case 3.4.2 B). The reason for this distinction is that in the case of $c_g = 1$ there exists exact priority queueing formulas for the mean queueing delay, whereas in the case $c_g > 1$, we must resort to an approximate analysis based on modified service times. Notice that, our analysis is proceeding from the node level of operations at the node level of tools.

The procedure of processing a job at a single-wafer tool consists of four distinct steps: minor setup, load, serially processing of all the wafers in the job and unload. Minor setup is the one step that is not necessarily performed for each job and it can be claimed that its time is negligible comparing to the other time's steps. If a minor setup is performed for a specific job depending on its operation, the operation of the job will be processed immediately before it. In our steady-state analysis, we suppose that the minor setup having a deterministic setup time is performed with a given probability for each operation. The load and unload times are each assumed to be are independent and identically distributed (i.i.d.) random variables with known distributions. The process time of a job is the sum of the process times of the individual wafers in the job, which are also assumed to be i.i.d. random variables. So, the total processing time for a job at operation k of a single-wafer tool is calculated by:

$$S_k = M_k + L_k + \sum_{n=1}^{J_k} W_k(n) + U_k \qquad (\mathbf{3.14})$$

M_k denotes the amount of time required for a minor setup, L_k and U_k give the time needed to load and unload a job at operation k and W_k represents the processing time per wafer for a job at operation k.

However, what we need to define the characteristic tool measures of utilization and delay time are the first two moments of the total processing time S_k, the mean and

the variance, which are the most significant quantities in our further analysis. These two moments are specified by the following expressions:

$$E[S_k] = E[M_k] + E[L_k] + E[J_k] * E[W_k] + E[U_k] \quad (\mathbf{3.15})$$

$$V[S_k] = V[M_k] + V[L_k] + E[J_k] * V[W_k] + E^2[W_k] * V[J_k] + V[U_k] \quad (\mathbf{3.16})$$

At next, the utilization of the tool group that measures the long-run proportion of time that a tool in a tool group spends processing jobs (i.e., neither idle nor incapacitated) is calculated by:

$$\rho_g = \sum_{k \in T_g} \frac{\Lambda_k * E[S_k]}{c_g} \quad (\mathbf{3.17})$$

To calculate the mean queueing delay at a single-wafer tool group, it is further assumed that jobs and incapacitation events arrive following Poisson processes. The queueing delay formulas are then modified to account for more general arrival processes. Regarding that, we specialize our analysis in two cases considering the case of a single tool ($c_g = 1$) in the tool group and the case of multiple tools ($c_g > 1$). For the first case, there exist exact priority queueing formulas, in contrast with the second case where approximate expressions based on modified service times are used due to the lack of analytical types for general arrival processes.

Case 3.4.2 A: Single Tool

Because of the Poisson arrivals of operations even for incapacitation events we give the exact type for $M/G/1$ queue with nonpreemtive priority jobs. The following expression for the mean queueing delay for jobs at tool group g is at the same form and it is sharing the equal idea as the type $(\mathbf{2.2})$ that we gave describing Queueing Models for Toolsets in Section 2.2.

$$E[D_g] = \frac{\sum_{b \in B_g} \Lambda_b * E[{S_b}^2] + \sum_{k \in T_g} \Lambda_k * E[{S_k}^2]}{2 * (1 - {\rho_b}^{inc}) * (1 - {\rho_b}^{inc} - \rho_g)} \quad (\mathbf{3.18})$$

The generalized formula of $(\mathbf{3.18})$ incorporates the first two moments of inter-arrival and service times of jobs using their squared coefficients of variation.

The SCV of the inter-arrival processes are derived from the linear equations of variability $(\mathbf{3.12})$ while the SCV of the "generic" service times of jobs is extracted from the expression:

$$v_g{}^s = \frac{E\left[S_g{}^2\right] + E^2\left[S_g\right]}{E^2\left[S_g\right]} \qquad (\mathbf{3.19})$$

where $S_g \equiv S_k(w.p.\frac{\Lambda_k}{\Gamma_g})$ for each operation k that is performed at tool group g. The first two moments of S_g are given by the following types:

$$E\left[S_g\right] = \sum_{k\in T_g} \frac{\Lambda_k * E[S_k]}{\Gamma_g} \qquad E\left[S_g{}^2\right] = \sum_{k\in T_g} \frac{\Lambda_k * E\left[S_k{}^2\right]}{\Gamma_g} \qquad (\mathbf{3.20})$$

Before we give the expression of the mean queueing delay, Connors et al. [3] specifies two correction factors contributing the term variability of general processes. The one describes the arrival and service processes of jobs and the second which corresponds to the incapacitation events.

$$\varphi_g = \frac{v_g{}^s + v_g{}^a}{v_g{}^s + 1} \qquad \varphi_b = \frac{v_b{}^s + v_b{}^a}{v_b{}^s + 1} \qquad (\mathbf{3.21})$$

The approximation for the mean queueing delay at tool group g, for non-Poisson (general) inter-arrival times, is

$$E\left[D_g\right] = \frac{\sum_{b\in B_g} \Lambda_b * E\left[S_b{}^2\right] * \varphi_b + \varphi_g * (\sum_{k\in T_g} \Lambda_k * E\left[S_k{}^2\right])}{2 * (1 - \rho_b{}^{inc}) * \left(1 - \rho_b{}^{inc} - \rho_g\right)} \qquad (\mathbf{3.22})$$

The extraction of this approximation was motivated (have strong influences) from the well-known approximation for $G/G/1$ queues, as you can prove setting $m = 1$ at the expression $(\mathbf{2.3})$. It is evident from the approximations $(\mathbf{3.22})$, $(\mathbf{2.3})$ that

$$\mathrm{E[D]_{G/G/1}} \approx \mathrm{E[D]_{M/G/1}} * \left(\frac{v^s + v^\alpha}{v^s + 1}\right) \approx \left(\frac{\rho^2 * (v^s + v^\alpha)}{2 * \lambda * (1-\rho)}\right) \approx \left(\frac{\rho * (v^s + v^\alpha)}{2 * (1-\rho)}\right) * s$$

Case 3.4.2 B: Multiple Tools

In this case we want to model theoretically the tool groups which consist of more than one machine-server. The exception from the previous case of a single machine is that the formulas for the mean queueing delay are estimations rather than exact analytical

types, even for Poisson and non-Poisson (general) arrivals. To doing so, we develop two approximation schemes. The first one adjusts the service times to account for incapacitation, and then uses a (nonpriority) multi-server queue model. The second approach is to simply convert the model of the tool group with multiple tools into a single-server model like the one in Case 3.4.2 A, in which the single tool operates c_g times faster. We will explain them in next paragraphs.

Specifically, analyzing the first approach, Connors et al. [3] created the random variable of the adjusted service time at a tool group g, Z_g, which is the sum of the generic service time aggregated over all operations for the specific tool group, S_g, and the variable defined as,

$$B_g = \begin{cases} S_b & w.p.\ \dfrac{\Lambda_b}{\Gamma_g}, b \in B_g \\ 0 & w.p.\ 1 - \displaystyle\sum_{b \in B_g} \frac{\Lambda_b}{\Gamma_g} \end{cases}$$

The first two moments of the adjusted service time that they will be used to calculate the SCV of the adjusted service time are given from the following expressions:

$$E[Z_g] = E[S_g] + E[B_g] \qquad (\mathbf{3.23})$$

$$E[{Z_g}^2] = E[{S_g}^2] + E[{B_g}^2] + 2 * E[S_g] * E[B_g] \qquad (\mathbf{3.24})$$

where

$$E[B_g] = \textstyle\sum_{b \in B_g} E[S_b] * \frac{\Lambda_b}{\Gamma_g} \quad and \quad E[{B_g}^2] = \sum_{b \in B_g} E[{S_b}^2] * \frac{\Lambda_b}{\Gamma_g} \qquad (\mathbf{3.25})$$

It is implicitly assumed that $\sum_{b \in B_g} \Lambda_b \leq \Gamma_g$ is applied. This equation shows that the incapacitation events (of all types) occur at each tool group less frequently than job arrivals (of all operations). This assumption certainly holds in the applications that we are concerned with.

Hence, the SCV for the adjusted service time, ${v_g}^z$, is

$${v_g}^z = \frac{E[{Z_g}^2] - E^2[Z_g]}{E^2[Z_g]} \qquad (\mathbf{3.26})$$

and the approximation for the mean queueing delay is

$$E[D_g] \approx \mathrm{E[D]}_{\mathrm{M/M/c}} * \left(\frac{{v_g}^z + {v_g}^a}{2}\right) \qquad (\mathbf{3.27})$$

$\mathrm{E[D]}_{\mathrm{M/M/c}}$ is the mean queueing delay for an "equivalent" M/M/c queue, with arrival rate of $\Gamma \equiv \Gamma_g$, the service rate is equal to $\mu = 1/E[Z_g]$, and $c \equiv c_g$,

$$\mathrm{E[D]}_{\mathrm{M/M/c}} = \left(\frac{P(D>0)}{c*\mu - \Gamma}\right) \qquad (\mathbf{3.28})$$

where $$P(D>0) = \left(\frac{\frac{(c\rho)^c}{c!(1-\rho)}}{\sum_{i=0}^{c-1}\frac{(c\rho)^c}{i!} + \frac{(c\rho)^c}{c!(1-\rho)}}\right) \quad \text{and} \quad \rho = \frac{\Gamma}{\mu * c} \qquad (\mathbf{3.29})$$

This is an Erlang's loss formula that represents an approximation of the probability that all servers are busy.

The second approaching method replaces the c_g tools by a single tool that operates c_g times faster. This is tantamount to scaling (dividing) all service times S_k and incapacitation times S_b by a factor c_g. For instance, for Poisson arrivals (of both jobs and incapacitation events), the expected queueing delay can be computed easily following the $E[D_g]$ expression ($\mathbf{3.22}$) with no change but only divide the result by ${c_g}^2$. Though, this method will result in overestimating the queueing delay because replacing a set of multiple servers by a single, faster server often results in a longer queueing delay. One way to compensate for this is to include an adjustment factor $\left({\rho_g} + {\rho_b}^{inc}\right)^{\sqrt{c_g - 1}}$. So, the mean queueing delay, for non-Poisson arrivals using the adjustment factors φ_b, φ_g, is

$$E[D_g] = \frac{\left({\rho_g} + {\rho_b}^{inc}\right)^{\sqrt{c_g - 1}}}{{c_g}^2} * \frac{\sum_{b \in B_g} \Lambda_b * E[{S_b}^2] * \varphi_b + \varphi_g * \left(\sum_{k \in T_g} \Lambda_k * E[{S_k}^2]\right)}{2 * (1 - {\rho_b}^{inc}) * \left(1 - {\rho_b}^{inc} - \rho_g\right)} \qquad (\mathbf{3.30})$$

To summarize, in Connors et al.'s research paper ([3]), it is underlined that from numerical experience in this field indicates that both approaches presented at this case of multiple tools work quite well, with the second approach slightly better for tool groups with a small number of tools.

3.4.3 Batch Tools

Simple batch tools process wafers in batches. For these tools, the processing of a job consists of the same four distinct steps: minor setup, load, parallel process of all the wafers in the job, and unload. For the analysis two distinct cases are considered: the case where the maximum batch size, ${\beta_g}^{max}$, is less than or equal to the average job size and the case where the maximum batch size is greater than the average job size. By average job size, we mean the average over all jobs arriving to a tool group. It is defined as

$$\bar{J_g} \equiv E[J_g] = \frac{\sum_{k \in T_g} \Lambda_k * E[J_k]}{\Gamma_g} \qquad (\mathbf{3.31})$$

Case 3.4.3 A: ${\beta_g}^{max} \leq \bar{J_g}$

In this case, the average size of a job exceeds the capacity of a single batch, so the jobs must typically be split into two or more batches. It is defined the number of batches required to process a job at operation k that has J_k wafers in it.

$$\beta_k \equiv \left\lceil \frac{J_k}{{\beta_g}^{max}} \right\rceil \qquad (\mathbf{3.32})$$

Since β_k batches need to be processed to complete the processing of a job at operation k, the total processing time of the job is

$$Y_k = \sum_{i=1}^{\beta_k} S_k(i)$$

where the $S_k(i)$'s are i.i.d. random variables, each following the same distribution as S_k. The mean and variance of Y_k are

$$E[Y_k] = E[\beta_k] * E[S_k] \quad , \quad V[Y_k] = E[\beta_k] * V[S_k] + E^2[S_k] * V[\beta_k] \qquad (\mathbf{3.33})$$

The calculation of the utilization and mean queueing delay is done similarly to the single-wafer tool case with the difference that we use the $E[Y_k]$ and $V[Y_k]$ of types

($\mathbf{3.33}$) instead of $E[S_k]$, $V[S_k]$. In the above case the assumption that different jobs are not batched together is hold.

Case 3.4.3 B: ${\beta_g}^{max} > \overline{J_g}$

In second case where the maximum batch size is greater than the average job-size a batch-service queueing model is used. Particularly, Connors et al. [3] follows a "greedy" policy of what is mostly used in practice. they assume that a "greedy" rule is followed to load batches, i.e., as soon as a tool (machine) becomes available it begins to process a new batch if at least one job is waiting and it batches as many jobs as are available. To model this type of tool, it was supposed that operations with different process codes can be batched together.

Since a job does not require more than one batch to be processed, its processing time at an operation k is just S_k. Then, an adjusted service time model is constructed to account for incapacitation events as it was done for the case of multiple single-wafer tools in Case 3.4.2 B. The same method is followed in order to define the first two moments of the adjusted service time of a batch. Next, we modify the maximum batch size so to correlate the number of wafers into a maximum batch size and the average number of jobs. We define the maximum batch size in jobs, ${\omega_g}^{max}$, to be

$$ {\omega_g}^{max} \equiv \left\lfloor \frac{{\beta_g}^{max}}{E[J_g]} \right\rfloor \qquad (\mathbf{3.34}) $$

To calculate the queueing delay and the utilization of this case of batch tools it is necessary to compute some probabilities that identify the busy servers of a $M/M^{{\omega_g}^{max}}/c_g$ batch-service queue. Let $\pi_{m,n}$ denote the probability that m servers are busy and there are n customers in queue. Define $\varphi_g \equiv \Gamma_g * E[Z_g]$, so, from Connors et al. [3] we list the equations for the probabilities, types ($\mathbf{3.35}$):

$$ \left(\pi_{0,0}\right)^{-1} = \frac{{\varphi_g}^{c_g}}{c_g!} * \left(1 - \frac{1}{x}\right)^{-1} + \sum_{i=0}^{c_g - 1} \frac{{\varphi_g}^{i}}{i!} \qquad (\mathbf{3.35\ a}) $$

$$\pi_{m,0} = \pi_{0,0} * \frac{{\varphi_g}^m}{m!}, m = 1, \dots, c_g - 1 \qquad (\mathbf{3.35\ b})$$

$$\pi_{c_g,0} = \pi_{0,0} * \frac{{\varphi_g}^{c_g}}{c_g!} * \left(\frac{1}{x}\right)^n, n = 0,1, \dots \qquad (\mathbf{3.35\ c})$$

where x is the single root, lying in the interval $\left(1, c_g * \frac{{\omega_g}^{max}}{\varphi_g}\right)$ of the following equation

$$\frac{\varphi_g}{c_g} * x^{({\omega_g}^{max}+1)} - \left(1 + \frac{\varphi_g}{c_g}\right) * x^{{\omega_g}^{max}} + 1 = 0 \qquad (\mathbf{3.36})$$

Concluding, the approximation of the mean queueing delay when arrivals and service times follow general distributions is given by

$$D_{G/G{\omega_g}^{max}/c_g} \approx D_{M/M{\omega_g}^{max}/c_g} * \left(\frac{{\upsilon_g}^a + {\upsilon_g}^z}{2}\right) = \left(\frac{\pi_{c_g,0} * x}{\Gamma_g * (x-1)^2}\right) * \left(\frac{{\upsilon_g}^a + {\upsilon_g}^z}{2}\right) (\mathbf{3.37})$$

To calculate the utilization of the batch tool, Connors et al. [3] give the calculation type of the utilization included incapacitation events, $\tilde{\rho}_g$, from the adjusted service time model because of the true utilization of the tool is unknown.

$$\tilde{\rho}_g \equiv \sum_{i=1}^{c_g-1} \pi_{i,0} * \frac{i}{c_g} + \sum_{i=0}^{\infty} \pi_{c_g,i} = \pi_{0,0} * \sum_{i=1}^{c_g-1} \frac{i * {\varphi_g}^i}{c_g * i!} + \pi_{0,0} * \left(\frac{{\varphi_g}^{c_g}}{c_g!} * \frac{x}{x-1}\right) \qquad (\mathbf{3.38})$$

The quantity $\tilde{\rho}_g$ includes time spent "serving" incapacitation events and so is greater than the proportion of time the tool spends processing jobs. To obtain the actual tool's utilization ρ_g that spends only processing an adjustment of type ($\mathbf{3.38}$) is needed. Setting ${\tilde{\rho}_g}^{inc}$ the proportion of time the tool spends "serving" incapacitation events (in general, ${\tilde{\rho}_g}^{inc}$ is less than the actual proportion of time the tool spends incapacitated), and then, combining the following relations, we can resolve with unknown the requested actual tool utilization.

$$\tilde{\rho}_g = \rho_g + {\tilde{\rho}_g}^{inc}$$

$$\frac{{\tilde{\rho}_g}^{inc}}{\rho_g} = \frac{E[B_g]}{E[S_g]} \Rightarrow \rho_g = \frac{\tilde{\rho}_g}{1 + \frac{E[B_g]}{E[S_g]}} \qquad (\mathbf{3.39})$$

Now, we proceed to end up the theoretical analysis with the expressions of the Cycle Time Estimation.

3.5 Cycle Time Estimation

Once the mean queueing delay at each tool group and the mean processing time of each operation are computed, the average cycle time for each product family in the can be estimated using the family routing information.

Lettting N_f denote the set of norninal operations for product family f and let $R(j)$ denote the set of operations in the rework sequence initiated immediately after operation j, the expected nominal cycle time for product family f, ${\Psi_{nom}}^f$, and the expected cycle time of the rework sequence initiated after operation j, $\Psi_{rw}(j)$, are approximated as follows:

$$ {\Psi_{nom}}^f \approx \sum_{k \in N_f} \left(E\left[D_{t(k)}\right] + E[S_k] \right) \qquad (\mathbf{3.40}) $$

$$ \Psi_{rw}(j) \approx \sum_{k \in R(j)} \left(E\left[D_{t(k)}\right] + E[S_k] \right) \qquad (\mathbf{3.41}) $$

where $E\left[D_{t(k)}\right]$ denotes the approximate mean queueing delay at tool group $t(k)$ and $E[S_k]$ denotes the mean total processing time of operation k. Recall that for an operation k, $P_{rw|nes}(k)$ denotes the probability that a job is sent for rework after the operation k. By definition, if $P_{rw|nes}(k) = 0$ then $\Psi_{rw}(j) = 0$. Letting R_{max} denote the maximum number of times that a job can be sent on any rework loop then the expected cycle time, Ψ^f, for the product family f is given by

$$ \Psi^f \approx {\Psi_{nom}}^f + \sum_{k \in N_f} \sum_{i=1}^{R_{max}} P_{rw|nes}(k)^i * \Psi_{rw}(k) \qquad (\mathbf{3.42}) $$

With cycle time estimation we close this Chapter 3 of the theoretical background that we have to be aware of and of course it is supported and implemented from our computational model of the next Chapter 4.

Chapter 4

Implementation of the Queueing Network Model

The current Chapter represents our efforts to implement Connors et al.'s [3] approach in a reliable computational model. Here, we will describe the implemented techniques that were used developing the computational model while in parallel we will analyze the code, step by step, making references in the applied calculations types and methods of the theoretical model of Chapter 3. At first, we display the input variables that are necessary for the computational model (Section 4.1). These variables have the possibility to be changing manually or automatically at every different example will be executed. At Section 4.2, we initialize the probabilities that they will be used to find the distributions of the size of the jobs in the next Section 4.3. The next step is to solve the traffic equations of the rates (Section 4.4). This is happen by creating the routing matrix and solving a linear system of the traffic rates (Section Part 4.4.1). Also, some additional quantities based on these rates are defined (Section Part 4.4.2). At the next Section 4.5 we manipulate a function so that to calculate the first two moments of the tools time requirements of the processing steps as well the features of the incapacitated events. Based on the theoretical model, we calculate the tool's utilization and the SCV of the service times in the same complex frame of code (Section 4.6), since, in Section 4.7, we solve the traffic variability equations. Lastly, we compute the mean queueing delay for each toolgroup and the mean cycle time for all product families (Section 4.8).

4.1 Input Variables for the Computational Model

The most important and necessary part in coding is to aggregate at the top of the code all the input quantities or other variables that usually change in each different example that the computational model runs. This is a technique of modelling elegantly since the user would focus on the beginning of the code and not in the main code's body searching the quantities that are to be changed. Thus, it is helpful for the user's convenience and the optimization of the code, whichever parameter you want to change in a various executable instance to be at the beginning of the code.

Firstly, we determine the number of tool groups, G, the number of product families, F, and the number of operations per product family, N^f. Moreover, the sequence of the tool groups where each operation is performed per product family are given as input data as well the number of machines at each tool group, c_g.

We also specify the basic entity which circulates in a fab system namely the job-size, i.e. the number of wafers that a lot is composed of. The maximum number of wafers in a job (n_{max}) will be very helpful in the main body of the code because it is a variable that participate in calculations and control commands of the developed code. The exogenous arrival rate of wafers per week is used as the input rate for our modelling system and it is defined for each product family that enters into the fab. We assume it deterministic for our examples in an average of a week. The exogenous arrival rates are reversed in minutes (7days/24hours/60mins) because the basic time-unit of the model as in other similar literature examples is the minute. Another decisive quantity for our model is the maximum batch size at each toolgroup. This could be predefined in lots or wafers as a ration of the maximum number of wafers in a lot (n_{max}) or an independent arbitrary number.

Our computational model also requires some time's specifications as input. The run processing time (in minutes) for a lot (job) of each process step without load, unload and setup times is known from the literature and other simulated models as well as the load and unload times at the machines of each tool group. We suppose that these time

durations of Load, effective-Processing and Unload processes are the mean values of a uniform distribution that they follow. The Loading, effective-Process and Unload times are determined for each lot.

Additionally, the number of incapacitation events B has to be fixed in advance. In the model that we are coding, these are the preventative maintenance, the breakdown and the major setup. For these incapacitation events we give their arrivals at each toolgroup and their mean processing time in each tool group. We assume that the time an incapacitation event spends in a toolgroup follows the uniform distribution with mean equals to the price we give as input. Especially, we import the arrivals and the time durations of incapacitation events in $(B \times G)$ matrices where G the number of tool groups, as we mentioned in previous paragraph, and B the number of incapacitation events. The times are determined in minutes while the arrivals are reported in arrivals per minute.

Some auxiliary variables are also added as inputs that contribute to the economy of some calculations in the main body of the code. For example, the maximum allowable number of times that a job is sent for rework after each operation (*max_mmax*) is important for the creation of the probability that a job is sent for rework an exactly number of times after each operation. There also are percentages of dispersions that characterize the tool's time requirements and the time requirements of incapacitation events. These percentages define the intervals of the uniform distributions affecting the variability of the system.

Lastly, we also derive at the beginning the necessary quantities of the total number of operations N, the product family of each operation $u(f)$, the first operation of each product family $s(f)$, even, the nominal successor operation as well the rework successor operation of each operation for all operations. After the determination of the sequence of tool groups for each product family, now we define the tool group where each operation is performed for all operations, nominal and rework. All the above quantities are calculated from automated commands in MATLAB-MathWorks with the help of the pre-defined input quantities.

Having gathered the variable quantities that change in each example in the top of our computational model is much easier to simulate much more cases because it doesn't need to modify the main body of the code, but just the parameters you need.

4.2 Initializations of Scrap-Rework Probabilities

Here, we initialize the main probabilities, their meaning analyzed in Section Part 3.2.2, in order to run the code and calculate the appropriate distributions of the next Section 4.3. Under normal circumstances these probabilities would be found from statistical analyzations of semiconductor industrial datasets. Experts monitor and collect data from the production line in order to provide the correct elements of their analysis in other production sectors that they are involved in different kinds of analyzes. However, due to the lack of this information we can generate randomly the values of the probabilities which are independent each one from the other, or more specifically, we can initialize the probabilities in a specific value level, e.g. 10 or 15 percent.

We refer to the scrapping probabilities $P_{js,}$ $P_{es|js}$ and $P_{ws|ps}$, also the rework probabilities $P_{rw|nes}$, $P_{erw|rw}$ and $P_{wrw|prw}$. For greater ease, we use one parameter probability for scrapping, p_s, and another one for the rework probabilities, p_r. At these two ancillary probabilities we set the value level of the above scrapping and rework probabilities that we want. The p_s is weighted with the probabilities $P_{js,}$ $P_{es|js}$, $P_{ws|ps}$ and the p_r with the corresponding probabilities of rework, $P_{rw|nes}$, $P_{erw|rw}$, $P_{wrw|prw}$. So, we initialize vectors of probabilities (population of individual numbers) illustrating a probability for each operation. After we have initialized our main probabilities, we calculate the two probabilities that an individual wafer go for scrapping and go for rework (P_{ws}, P_{wrw} respectively) which are functions of the previous ones.

The next task is to create the distribution of the probability that a job is sent for rework an exactly number of times after each operation, P_{rwm}. the distribution of this probability is created placing random probability's numbers lying in the interval [0,1] in a delimited space defined by the maximum allowable number of times that a job is sent for rework after each operation, *max_mmax,*. Then, we normalize this distribution of P_{rwm}.

Finally, we initialize the requested distributions of the job-size for jobs are not scrapped and jobs go for rework, declaring a value for each operation of each probability. Vectors of auxiliary resources are also pre-initialized for testing different computational methods. Particularly, the probabilities that we initialize with zeros at first are:

- $Probin(f,n)$ = Probability that a job arriving to first operation of product f, $k = s(f)$, has n wafers
- $Probnom(k,n)$ = Probability that a job departing from operation k has n wafers
- $Probrewm(k,n,m)$ = Probability that a job arriving at m_{th} rework operation after operation k has n wafers (m: the number of rework times)
- $Probrew(k,n)$ = Average probability that a job arriving at rework operation after operation k has n wafers
- $Diffnom$, $Diffrewm$ = Vectors that store the numerical difference between two computational methods (determine the convergence of two methods).

$Probin$ is the distribution of wafers (job-size) in the entrance of the system, $Probnom$ is the distribution of wafers of a nominal operation, i.e., the distribution of the size of the job that leaves an operation and directed to its nominal operation (the next in the nominal sequence). $Probrew$ is the average weighted distribution of the $Probrewm$ probabilities' distributions. $Probrewm$ declares the distribution of the job-size for a job arriving to a rework operation for the m_{th} time.

4.3 Computation of Job-Size Distributions

After we have initialized the probabilities we can implement the types $(\mathbf{3.4})$-$(\mathbf{3.7})$ of Section Part 3.2.3 in order to calculate the distributions of the job-size for the nominal successor operations and the rework successor operations. The process is almost similar for both calculations if we take notice of these computational formulas. They are retrospective algorithms that are based on the precedent repetition of the process, i.e. they use the distribution of the job-size of the previous operation to calculate the job-size distribution of the current operation.

Firstly, we calculate the job-size distribution of jobs that are not being scrapped because in the next step this distribution will be used to find the job-size distribution of jobs sent to rework. The size in wafers of a job in the entrance of the system is concrete and deterministic. So, for each product family, the size for a job (lot) is determined from the given maximum number of wafers, n_{max}. Hence, the distribution of the job-size for the first nominal operation of the model, $Probin$, is degenerate and contains n_{max} wafers with absolute probability equal to one without detriment of reality.

The probability that a job has n wafers in it and it is directed to the nominal successor operation from the operation it was in the previous repetition is calculated from the iterative algorithm $(\mathbf{3.4})$ for $n \geq 1$. The calculation is achieved with auxiliary temporary variables. One stores in a vector named "$Prob$" the job-size distribution of the previous step-operation $(k-1)$, since the repetitive steps are the operations counting from the first to the last operation of each product family. The second auxiliary vector named "$Pexact$" stores the values of the binomial distribution of the current repetition. We use the type of the probability density function (pdf) of the binomial distribution with adjusted coefficients due to the form of the sums. Therefore, with this method we build step by step the distribution of job-size for the nominal operations. The probability that a job is departed from an operation with no wafers in it, i.e. all of the contained wafers are scrapped, can be calculated analytically from the Connors et al. formula $(\mathbf{3.5})$, or approximately from the type:

$$Probnom(k,0) = 1 - \sum_{i=1}^{n_{max}} Probnom(k,i)$$

where k is the current repetition (operation).

In the end, we can compare the two solutions of calculation probability a job contains zero wafers after completed an operation, $P(J=0)$, with the quantity $Diffnom$. This vector calculates the difference between the two methods of computing the probability $P(J=0)$. If the resulted value is very small, approximately equal to zero, this means that both computational methods give correct, identical results. Certainly the most effective solution with less computational time and reliable results is the second one with the approximation type and not the Connors's et al. formula $(\mathbf{3.5})$. As a last step, we normalize the completed probability's distribution.

In order our analysis to be more accurate we calculate the job-size distribution assuming jobs can be sent for rework after each operation an arbitrary number of times, up to a maximum allowed, $mmax$. Following the above methodology, we first calculate the conditional job-size distribution given that a job has been sent for rework for m_{th} time, $Probrewm$. This can be done by repeatedly applying the calculation presented above. We use again the auxiliary temporary variables "$Prob$" and "$Pexact$", in rework terms in this case, and we continue by calculating the distribution that we interested in by the type $(\mathbf{3.6})$. To compute the probability that the job arriving for rework containing zero wafers after a nominal operation, two methods can be used to determine it, as it was done previous for the $Probnom$ distribution. We can use Connors et al. formula $(\mathbf{3.7})$ or the following simply type:

$$Probrewm(k,0,m) = 1 - \sum_{i=1}^{n_{max}} Probrewm(k,i,m)$$

where $m = mmax(k)$: the number of rework times at operation k. Accordingly, we use the variable $Diffrewm$ to compare the resulted probability $P(J=0)$ from the two computational methods presented above. The difference proves that both computational methods yield to the same values of $Probrewm(J=0)$.

Having calculated and normalized the distribution $Probrewm$ distribution for each value of m up to the allowed maximum, we use it to find the average job-size distribution of jobs sent to rework for up to the m_{th} time at each operation. In this way, we combine the completed distributions $Probrewm(k,n,m)$ of all values of m, i.e. the times that a job is went for rework, into a single distribution $Probrew(k,n)$ for each operation by weighting the distribution for rework exactly m times by the probability that a job is sent for rework exactly m times and summing over all values of m. In a code manner, the distribution of the job-size for a job that goes for rework exactly m times, $Probrewm(k,n,m)$, is multiplied by the probability that the job is sent for rework exactly m times, P_{rwm} and in a repeat pattern, we sum all values of m for each operation by keeping the current cost and adding the next value up to this. The probability P_{rwm} is assumed as a known probability which we initialize it as an input distributional probability at Section 4.2.

So, before continue to calculate the arrival rates we have computed the distributions of the size of the jobs that are not scrapped and proceed to its nominal operation and also for the jobs that are sent for rework after a nominal operation. The computational methods of the above are certified in Appendix A, in which we quote the model codes and the applied commands in the environment of MATLAB-MathWorks.

4.4 Solving Traffic Rates Equations

In calculating the traffic rates between nodes we use the analytical types of Section Part 3.3.1. At first we create the routing matrix which corresponds to the probabilities that a job is directed from one situation (operation) to another and then we solve a linear system to calculate the arrival rates (Section Part 4.4.1). At the Section Part 4.4.2, we compute other important resources related to the traffic rates which are needed as inputs at the calculation system of the Squared Coefficients of Variation (SCV) of the aggregate job arrival processes to each node.

4.4.1 Linear System via Routing Matrix

The routing matrix, called Q, is integrated in the types that solve the traffic rates. In particular, type $(\mathbf{3.10})$, which is the linear system of the traffic equations, uses the matrix Q correlating the probability that a wafer departing from one operation arrives at a second operation with the arrival rate of the first one. Before continue to construct the matrix Q we have to identify its dimensions. In this part of the code, naming the operations we also mean the states. They are identical meanings.

Except from the nominal operations of the system, N at the total, we also have N rework operations, one for each nominal operation, based on the theoretical model that it is presented in Chapter 3. Additionally, we have to model the exit of the system as an individual state (+1 operation = exit state) and in analogous way, we do the same for the scrapping situation (+1 operation = state of scrap). So, the $[(N + N) + 1]$ is the state of

exit for all product families and the $[(N+N)+2]$ is the state of scrapping for the system.

Recall that Q is the matrix of probabilities that a wafer after completing operation j is routed to operation k. So, in order to build the matrix Q we assign values of probabilities to its matrix cells depending on the type ($\mathbf{3.9}$). We assume that the probability a wafer left the rework operation and directed to the nominal operation is equal to one, i.e., whatever goes for rework (of wafers) returns to the nominal sequence. Furthermore, we set with zero the probability that the wafers from exit stage go to a line operation, as well as no wafers from stage of scrapping go to a line operation.

Having found the routing matrix Q we can easily solve the linear system: $\lambda = (I - Q^T)^{-1} * A$. The implemented type referred by Connors et al. [3] implies that the matrix $I - Q^T$ is invertible and it also uses the vector A of exogenous arrival rates to calculate the arrival rates λ. The solution of the above system results in the rate of wafers at each nominal and rework operation of the system in conjunction with the rates of system's exit. By the expression type ($\mathbf{3.11}$) we calculate the arrival rates in jobs for the nominal and rework operations. Obviously, when the arrival rate of wafers is zero then also the arrival rate of jobs at the specific operation is zero.

4.4.2 Ancillary Resources for Calculation of Arrivals' SCVs

The linear equation system defined by the types ($\mathbf{3.12}$) calculates the Squared Coefficient of Variation (SCV) of the aggregated arrival processes to each tool group. These linear equations demand as input quantities the total arrival rate of jobs to each toolgroup, the total arrival rate of jobs from toolgroup to toolgoup, the proportion of arrivals between the tool groups and the proportion of jobs that leave one toolgroup and proceed directly to another.

First of all, we find the set of operations (T_g) performed at each toolgroup and we compute the total arrival rate of jobs to a toolgroup g, Γ_g, from the expression $\Gamma_g = \sum_{k \epsilon T_g} \Lambda_k$. These two entities have essential contribution in the remaining body of the code. Subsequently, we use a repeatable structure of the order ***for*** to calculate the total

arrival rate of jobs to toolgroup h from toolgroup g (traffic rates between toolgroups), γ_{gh}, and the proportion of arrivals to toolset h came from toolgroup g, θ_{gh}. The latest two quantities are combining in the expression $\theta_{gh} = \gamma_{gh}/\Gamma_h$.

A check for our computational code is considered if we quantify an additional state, $(G + 1)$ state, which corresponds to the outside world (entrance or/and exit). So, the sum of arrival rates of all the states/toolgroups at one toolgroup including the extra state of entrance/exit must equal to the quantity of the specific Γ_g. Likewise, the sum of proportions of the arrivals to a specific toolgroup that come from all toolgroups (plus the state of outside) must equal to one, as these proportions represent probabilities that a job leave toolgroup g and directed to toolgroup h. At last, a secondary quantity can be computed. The rate of jobs that are going for scrapping is assumed as the deduction (quantitative difference) of all the rates from toolgroup g that directed only to all other toolgroups, not outside, abstracted from all the arrival rates to that toolgroup g.

Continuing in the same idea, we are able to compute the proportion of jobs leaving toolgroup g that proceed directly to toolgroup h, r_{gh}, with the similar repetitive structure in terms of operations. We also have a check point (valuation) to the code for this computation by summing the values of proportions r_{gh} for all h from one specific g that must equal to one in according to all possibilities from which the jobs come to toolgroup g (law of flow conservation).

4.5 Processing & Other Time Requirements for Tools

At the next Section Parts they are defined the most important quantities of time requirements that they will be used to calculate utilizations and SCVs of service and arrival times at each tool group. We will see which characteristic features of the incapacitation events will be calculated (Section Part 4.5.1), such as the proportion of time each tool spends incapacitated (“utilization” of incapacitation events). In the same context, the first moments of the time distributions of loading, effective process and

unloading will be extracted (Section Part 4.5.2), which are prerequisites entities even for the last step of the Cycle Time calculation and have a preliminary role for the calculation Sections of the tool's utilization and queueing delay.

4.5.1 Features for Incapacitation Events

We have already imported at the model the arrival rate and the average duration for each incapacitation event at each tool group. Review that we model three incapacitation events: preventative maintenance, breakdown and major setup.

Before we proceed we have to underline this: in the input matrices of arrival rates Λ_b and time durations S_b of the incapacitation events, referred in the code as "$Lamdainc$" and "$ESinc$", respectively, the rows indicate the types of incapacitation events (1st row: preventative maintenance, 2nd row: breakdown, 3rd row: major setup) while the tool group which is affected from each incapacitation event is showed from each column (1st column: tool group 1, 2nd column: tool group 2, 3rd column: tool group 3). The matrices are in a $(B \times G)$ form which is more readable and understanding for the reader.

For the user it is much more effectiveness to reconstruct the matrices of arrivals and duration times of the incapacitation events in vectors so to have dimensions of a column or row. Up to this view, we use a two-level repeatable structure with three indicators. The one runs the rows and the second runs the columns of the input matrices of the incapacitation events. The third one increases continuously from one to the number $(B \times G)$ which is the maximum number of combinations between B number of types of incapacitation events and G number of affected toolgroups, $(i, j) \rightarrow k$. So, running the third indicator k specifies a specific position in the vector-table related to each incapacitation event of specific toolgroup. With this technique we decomposed the matrices in single vectors of one dimension.

The main quantities that are being calculated in each repetition are the first two moments of the time duration of the incapacitation events, the SCV of the "service time"

of each type of incapacitation events, a correction factor φ_b, the "utilization" of the incapacitation events for each tool group and finally two secondary variables.

We assume that the input data of the time durations for the incapacitation events represent the mean values of a uniform distribution that the times follow. Therefore, in order to find the second moments of these time distributions and more specifically the variance $V[S_b]$ as well the mean of the squared values $E[{S_b}^2]$, we created a ***function*** named "moments", which calculates the above quantities of the uniform distribution. The function accepts two variables as input, the mean value and a percentage of dispersion, while she outputs three quantities: the mean value, the variance and the mean of the squared times. The function uses the percentage of dispersion to create two bounds, a low (lb) and a high (ub) bound, around the mean value and then calculates the variance of the uniform distribution from the closed form: $(ub - lb)^2/12$. The mean of the squared values is calculated by the expression: $E[{S_b}^2] = V[S_b] + E^2[S_b]$.

The SCV of the "service time" of the incapacitation events is calculated from the type $(\mathbf{3.19})$ with a pointer b indicating the terms of incapacitation events. To compute the correction factor φ_b, except from the SCV of service times, we also need the SCV of the arrival rates of incapacitation events. In absence of data, we hypothesize that the SCV of the arrival rates of incapacitated arrivals is zero, i.e., we know exactly (deterministically) when each incapacitation event occur in the system. So, we calculate the correction factor from the expression $(\mathbf{3.21})$ for the incapacitation events.

Finally, at each repetition we build the incapacitated "utilization" and an important entity consisting of three multiplication parameters, called "$LbES2b$" in the code. The "utilization" uses the first moment of S_b, i.e. the mean value, and the arrival rate of each incapacitation event at each toolgroup, while the entity "$LbES2b$" uses the second moment of S_b, i.e. the mean of the squared values, the arrival rate and the correction factor for each incapacitation event at each tool group. However, these two quantities are computed for each toolgroup. The values of the three incapacitation events corresponding for each tool group are summed together forming a single value of the "utilization" and "$LbES2b$" quantities for each toolgroup.

The last quantities that are calculated are the first two moments of the variable B_g as they are given in the formulas $(\mathbf{3.25})$.

4.5.2 Time Requirements for Tools

Continuing from the previous Section Part, we are going to enhance the technique that was followed to obtain the incapacitation time parameters, in order to compute the necessary processing time requirements and other variables which are incorporated in the calculations. The aim of this frame of code is to calculate the first two moments of the total processing time S_k from the types $(\mathbf{3.15})$ and $(\mathbf{3.16})$. We compute the individual pieces of the above types by calculating the first two moments of each time distribution of the time requirements of a tool in a repetitive pattern for all operations.

To begin with, it is considered that minor's setup time is negligible comparing to the other time's steps. In addition, it is not sure that it is necessarily performed for each job. If it is performed for a specific job depending on its operation, we calculate the first two moments of its time distribution. Given a nonnegative random probability for each operation, and a minimum deterministic setup time close to zero, we can calculate the mean and the variance of this time variable. Specifically, $E[M_k] = m_k * p_k^{ms} + 0 * (1 - p_k^{ms})$, $V[M_k] = (p_k^{ms} * M_k^2) - (E^2[M_k])$. At the next distinct steps, we compute the second moments of the Loading and Unloading times for a job and run-Processing times for the wafers at each operation. The approach is the same as we did to find the second moments of the time durations of the incapacitation events in the previous Section Part 4.5.1. In particular, we use the ***function*** "moments" to calculate the variance of the uniformly time distribution that the above processes follow, setting as input the mean value of its time that equals to the input value that we gave at the beginning of the code.

However, there is an exception in finding the second moments of the run-Processing times of the wafers W_k with the above technique of calculating the second moments of Load and Unload processes. It is needed to use the average of the Job-Size distribution for each operation. Particularly, the data that we have for the Processing procedures represent the effective processing times per lot (job) without load, unload and

minor setup times. The effective processing time per lot is given from the expression $E[J] * E[W]$. Consequently, to compute the total processing time for a job S_k it is essential to define the average of the processing time per wafer, $E[W] = (E[J] * E[W])/E[J]$. Thus, before we proceed to calculate the second moments of the processing time per wafer at each operation, we calculate the average job-size $E[J]$ and then we use the mean processing time per wafer $E[W]$ in the function "moments" in order to calculate $V[W]$ and $E[W^2]$.

Therefore, having computed the individual terms of the types $(\mathbf{3.15})$, $(\mathbf{3.16})$, we calculate the first two moments of the Total Processing Time for a job at each operation, $E[S_k]$, $V[S_k]$. We also calculate the mean of the squared values of the uniformly distributed S_k, $E[S_k{}^2]$, which will be helpful to calculate the SCV of job's service times.

At the end of this code part, we compute characteristic quantities of batch tools. Referring to the first case where the average job's size exceeds the maximum batch size, i.e. $\beta_g{}^{max} \leq \bar{J}_g$, we calculate the mean and the variance of the defined quantity of the number of batches required to process a specific job at each operation, $E[\beta_k]$, $V[\beta_k]$. Then, the first two moments of the total processing time (Y_k) are also determined (see types $(\mathbf{3.33})$).

All the above procedure that it was followed in order to conclude in the calculation of the first two moments of the total processing time of a job at a nominal operation, it is also repeated for the rework operations.

4.6 Calculation of Utilization and SCV of Service Time

The current Section it may be the most significant part of the model's coding. We compute all necessary quantities that they will be used to solve the linear equations in order to find the SCV of the arrival processes and finally to calculate the queueing delays of the toolgroups. Almost every aspect of the tool-types characteristics are modeling in this Section.

Primarily, we compose a structure of repetitions for each tool group $g \in G$, in which several ***if*** statements are nested. The ***if*** statements check in which case we are placed at each time, i.e. what kind of tool-type characteristics are analyzed and computed at each frame of code. At the first level of control, the tool types of tool groups are separated accordingly to their batch size. If the batch size (in wafers) is equal to one, then the single-wafer tools are analyzed, or else if the batch size is greater than one, the properties of the batch tools are computed below this control. For the single-wafer tools, there is a control if the arrival rate of incapacitation events Λ_b in that type of tool group is smaller or equal to the total arrival rate of jobs coming to a tool group g, Γ_g. Instead, for the batch tools ($bmax(g) > 1$), we have a second level of control and more specifically, we compare the maximum batch size $bmax(g)$ with the average job-size, or else the average amount of all jobs in wafers, $AvWaf(g)$, performed to that specific tool group. If $bmax(g) \leq AvWaf(g)$ we follow a similar modelling as we done for the single-wafer tools, while, if $bmax(g) > AvWaf(g)$, we follow the approach of the "greedy" policy which was presented in Case 3.4.3 B of Chapter 3.

With these in mind, we compute first the average job-size by the type ($\mathbf{3.31}$). Then, in the code part that is referred for single-wafer tool groups, we calculate the utilization of that type of toolgroup and its overall utilization (total) including incapacitation events. We apply the expression ($\mathbf{3.17}$) in calculating the utilization of the single-wafer tool. The total utilization ($util(g) = utiltool(g) + utilinc(g)$) results by adding the "utilization" for the incapacitation events, i.e. the time that the tool spent incapacitated with the pure processing utilization of the toolgroup. Furthermore, we calculate the mean value and the mean squared value of S_g, as it is given in types ($\mathbf{3.20}$). So, now we can calculate the squared coefficient of variation (SCV) of the service times of jobs at the single-wafer tool implementing the type ($\mathbf{3.19}$).

In case we have multiple, parallel, tools ($c_g > 1$), the above technique in calculation of the utilizations and the first two moments of S_g is not changed. However, we have a control point that check the arrival rate of incapacitation events Λ_b. It is not permitted according to the theory the total of arrivals from incapacitation events occurred to a specific toolgroup to be greater than the total arrival rate of jobs in the tool group

($\sum_{b\in B_g} \Lambda_b \leq \Gamma_g$). We compute the quantities of the mean value and the mean squared value of the adjusted service times, $E[Z_g]$ and $E[{Z_g}^2]$, from the types ($\mathbf{3.23}$), ($\mathbf{3.24}$) in order to calculate the SCV of the adjusted service times. We also give an alternative type for the total utilization of the toolgroup, $util(g) = \Gamma_g * \frac{E[Z_g]}{c_g}$.

Having computed any necessary feature of the single-wafer tools we proceed in the batch-tool's characteristic quantities. We use the entity of average job-size, $AvWaf$, to distinguish the two methods as they presented at Section Part 3.4.3. In the case where $bmax(g) \leq AvWaf(g)$, the same commands in the same order as they were followed in the latest two paragraphs for the calculations of the single-wafer tools are used in coding. Only one key modification is needed in order for the batch tool's type the appropriate equations to be executed. In particular, this change lies in using the first two moments of S_k where we modify the equations in order to use the first two moments of Y_k. The first two moments of the total processing time Y_k were calculated at the end of the previous Section Part 4.5.2.

In this frame of code, the types ($\mathbf{3.33}$) of the theoretical model are used and accordingly they are incorporated in the equations for single tools. Remember that in the equations of calculation of the first two moments of Y_k, the number of batches required to process a job at an operation k, β_k, has strong contribution on them. Thus, we modify the equations ($\mathbf{3.17}$) and ($\mathbf{3.20}$) in order to take into account the first moments of the modified total time Y_k. The utilization and the SCV of the service times are calculated based on the first two moments of Y_k, $E[Y_k], E[{Y_k}^2]$ and $E[Y_g], E[{Y_g}^2]$ and then the technique for the calculation of the adjusted service time's moments $E[Z_g]$, $E[{Z_g}^2]$ is implemented to calculate the adjusted SCV of service.

The last case that we model is the case when $bmax(g) > AvWaf(g)$. At the beginning, we calculate the first two moments of S_g that help us to define the SCV of service times and secondly we find the adjusted SCV of service from the first two moments of Z_g. Finishing, it is needed to model the approach that uses steady state probabilities for servers as well the roots of a polynomial equation to calculate the utilization of the second case of batch tools. This approach called "greedy" policy holds

when the batch size for the tool group is greater than the average job-size, as it was presented in Case B of Section Part 3.4.3.

The "greedy" method says that first we specify the factor $\varphi_g = \Gamma_g * E[Z_g]$, and then we initialize the polynomial coefficients with standard values based on the quantities ${\omega_g}^{max}$, φ_g and c_g. The maximum batch size ${\omega_g}^{max}$ is given from the type $(\mathbf{3.34})$. After defining these, we solve the polynomial equation $(\mathbf{3.36})$ using the command ***roots*** of MATLAB-MathWorks. This command solves the polynomial equation and finds all its roots in a complex form of the imaginary plus the real part of the root. The method instead uses the single root of the solution which lying in the interval $(1, c_g * {\omega_g}^{max} / \varphi_g)$. So, we keep the root which has zero imaginary part from the pre-referred interval. This root is used as an input for the types of the steady state probabilities as it is presented in types $(\mathbf{3.35}\ \boldsymbol{a})$, $(\mathbf{3.35}\ \boldsymbol{c})$ where x corresponds to the single root.

Lastly, we compute the "adjusted" utilization $\tilde{\rho}_g$ by the analytical type of $(\mathbf{3.38})$ after we have computed the appropriate steady state probabilities. But, the actual utilization for the tool group, i.e. the proportion of time the tool spends processing jobs without the effect of the incapacitation events is calculated by the relation $(\mathbf{3.39})$. Note that the "adjusted" utilization is the total utilization including time spent "serving" incapacitation events.

4.7 Linear Equations for the SCV of Arrivals

In this Section we are going to calculate the Squared Coefficients of Variation (SCVs) of the aggregate job arrival processes to each toolgroup, ${\upsilon_g}^a$, solving the linear equations of the analytical model equations $(\mathbf{3.12})$ of the Section Part 3.3.2.

The variables that are assumed as inputs for these equations are the SCV of the aggregate exogenous arrival processes to each tool group ${\upsilon_g}^e$, the SCV of the service times ${\upsilon_g}^s$, the overall utilization (total), and especially the auxiliary resource quantities that we have computed in Section Part 4.4.2. Well, the ancillary resource's parameters are the proportion of arrivals to toolgroup h came from other toolgroup g, θ_{gh}, and the

proportion of jobs departure from toolgroup g and proceed directly at a toolgroup h, r_{gh}. The individual coefficients of ${v_g}^a$ that are incorporated in linear equation's model of $(\mathbf{3.12})$ are the a_h, b_{gh}, w_h, v_h and x_g for two general tool groups $g, h \in G$.

We assume that the SCV of the aggregate exogenous arrival process to a tool group is zero (deterministic arrival procedures) while the total utilization has been calculated in the previous Section 4.6. Therefore, implementing the types of the linear equation system it results in finding the SCVs of the arrival processes. Underline that in the equations we use the total utilization and not the pure utilization of processing for a toolgroup because we want to find the SCV of the total arrival processes to the toolgroup affected by jobs and incapacitation arrivals. Finally, we calculate and the correction factor of the single-wafer tool groups and the batch tools for which it is applied the first case where $bmax(g) \leq AvWaf(g)$.

4.8 Calculation of Mean Queueing Delay and Average Cycle Time

At the last Section we are going to code the calculation types of the model's quantities that support the main results of our implementation. We succeed in calculating the queueing delay at each tool group of the system and an estimation of the cycle time for all product families that we import in the system.

About calculating the queueing delay, we implement the same code modelling construction with the commands ***for*** and ***if***, as we developed in Section 4.6 for the calculation of the utilization and the SCV of service time. Especially, for single-wafer tools, we implement the method that specified from the expressions $(\mathbf{3.28})$ and $(\mathbf{3.29})$ in case the inequality $\sum_{b \in B_g} \Lambda_b \leq \Gamma_g$ is valid and we have $c_g \geq 5$ machines at these types of tool groups. On the other hand, if these conditions are not true, then we calculate the mean queueing delay from the general type $(\mathbf{3.30})$, as it is presented in Case B of Section Part 3.4.2. We have to note that these approaches are implemented either we have one machine/tool ($c_g = 1$) or we have multiple tools ($c_g > 1$) in the tool group. The final

delay with non-Poisson processes is calculated using the SCVs of the adjusted service times in the first case or the SCV of the actual service times in the second case in conjunction with the SCV of the arrivals. If the second, general, approach is followed the above SCVs of the service times are incorporated in the correction factors of types $(\mathbf{3.21})$.

For the batch tools, in the case that $bmax(g) \leq AvWaf(g)$ we repeat the code for the single-wafer tools with the detailed assumption that we refer to batch tools of the specific instance. This is because in the term of service rate, $\mu = 1/E[Z_g]$, the features of the total processing time Y_k, which takes account the batching processing, are implied. Practically, we replace the $E[{S_k}^2]$ with $E[{Y_k}^2]$. In the case where $bmax(g) > AvWaf(g)$ we implement the approximation $(\mathbf{3.37})$ in which the SCV of the service time is the adjusted.

In the final point of modelling, in order to calculate the average cycle time for each product family, we first compute the expected nominal cycle time for each product family, as it is given from the approximation $(\mathbf{3.40})$, and the cycle time for the rework sequence after each operation, see type $(\mathbf{3.41})$. We code these calculation types with the Connors et al. [3] assumption that a rework operation after an operation k is performed in the same equipment with the same processing time requirements. In the end, the total estimated cycle time is calculated building a sum of the maximum number of times $(mmax)$ that a job can be sent on any rework loop. In Chapter of the theoretical model, the quantity $mmax$ is referred as R_{max}. After all, we arrived at the point to calculate an estimation of the mean cycle time from the type $(\mathbf{3.42})$.

The next Chapter exhibits the results which are derived from corresponding examples of the literature as well with our own applications.

Chapter 5

Numerical Results

In this Chapter we present the derived results of various numerical examples in which we coded and implemented Connors et al.'s [3] approach. These are variants of a five-machine six-step model of a wafer fab initially suggested by Kempf ([6]) at Intel Corporation, described by Spier and Kempf ([14]). This model which is often referred to as Mini-Fab, has small complexity but contains many of the typical features of a wafer fab with respect to the basic system and process. At first, we describe the Mini-Fab model and two main variants of it that we analyzed (Section 5.1), as well as the results of our analysis on the two nominal examples (Section 5.2). Afterwards, we list an amount of different scenarios created by changing one parameter that we choose of the original model of Mini-Fab each running time (Section 5.3). At last, we give the results of these applications discussing probable correlations and ratios between the scenarios (Section 5.4).

5.1 The Mini-Fab model

The basic flow line of the Mini-Fab model consists of three machine groups where multi-product production and batch-processing are performed. A general diagram of the Mini-fab is shown in Figure 5.1. It is assumed that the Mini-Fab operates 24 hours per day, 7 days per week. Each day of operations is composed of two shifts of 12 hours.

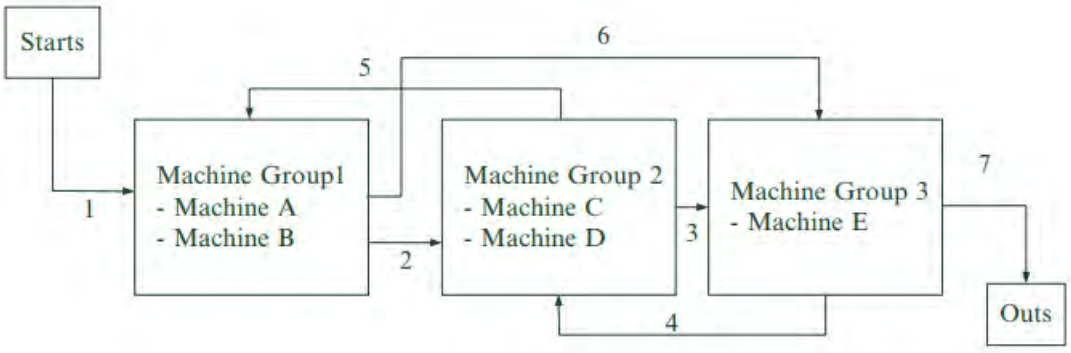

Figure 5.1: Process flow of the Mini-Fab model (source: Mönch et al., [9])

Suppose that the machines of each workstation represent in reality the diffusion, ion implantation and lithography machines with the corresponding processes of the wafer fabrication, respectively. So, the three machine groups, as indicated in Figure 5.1, provide the following features:

- Machine Group 1 (Diffusion) has two identical machines (Machines A and B). Each machine batches 3 lots at a time and requires 75 minutes per day of preventive (scheduled) maintenance.
- Machine Group 2 (Ion Implantation) has two machines (Machines C and D). Each machine processes one lot at a time, requires 120 minutes of preventive maintenance per 12-hour shift, and is down for emergency (unscheduled) maintenance between 12 and 16 hours per week.
- Machine Group 3 (Lithography) has of one machine (Machine E) that processes one lot at a time, requires 30 minutes of preventive maintenance per shift, and requires time-consuming setup changes when converting between process steps and/or product lots. Specifically, there is a 10-min setup on each step change, a 5-min setup on each product family change, and a 12-min setup on each step and product family change.

The Mini-Fab produces three different families of products, including a test product utilized for monitoring and production tracking. Hereafter, we refer to these products as Product A, Product B, and Test wafers. The test product is used to monitor

the accuracy of the production process and the other two types are commercial products to be sold to customers.

The basic product unit is the lot. There is no lot-sizing specification, so to be consistent with the framework of Connors et al. [3], we assume that each lot contains one wafer. The nominal lot starts (i.e., the lots releasing rates) are as follows:

- Product A: 51 lots per week
- Product B: 30 lots per week
- Test wafers: 3 lots per week

The released products follow a predefined production sequence visualized in Figure 5.1. So, the fabrication of each product considered in the model is completed following the next sequence of six processing steps where in each workstation the products are entered twice. These steps are as follows:

- Start
- Step 1: Diffusion (Machine Group 1)
- Step 2: Ion Implantation (Machine Group 2)
- Step 3: Lithography (Machine Group 3)
- Step 4: Ion Implantation (Machine Group 2)
- Step 5: Diffusion (Machine Group 1)
- Step 6: Lithography (Machine Group 3)
- Exit

Before starting the processing at each station, the wafer lots are held in six different buffers at the beginning of each step. In addition, in Station 1 there is a batching process that takes place before the batches are placed in buffers of Machine Group 1 for the process steps of 1 and 5. The capacity in the buffers of the system is limited to 18 lots (6 batches) in both buffers of workstation 1 and 12 lots in the remaining 4 buffers of workstations 2 and 3.

There are also 2 equipment operators and 1 maintenance technician assigned to each shift. Each operator gets two one-hour breaks per shift and has an additional 60

minutes for meetings and training. The technician has 30 minutes per shift for meetings and training and gets two 45-minute breaks.

The re-entrant mapping between processing steps, machines, and personnel is shown in Table 5.1, along with loading, processing, and unloading times in minutes.

Table 5.1: Processing steps, machines, and operator assignments (LT = Load Time, PT = Process Time, UT = Unload Time, OD = Operator Designation).

	Machine Group 1: Machines A and B Maximum batch size: 3				Machine Group 2: Machines C and D Maximum batch size: 1				Machine Group 3: Machine E Maximum batch size: 1			
Step	LT	PT	UT	OD	LT	PT	UT	OD	LT	PT	UT	OD
1	20	225	40	1								
2					15	30	15	1,2				
3									10	55	10	2
4					15	50	15	1,2				
5	20	255	40	1								
6									10	10	10	2

The factory has a cellular layout and an automated material handling system. The linear layout consists of five cells: the starting material warehouse on the extreme left, Machine Group 1 in the next cell, Machine Group 3 in the middle, machine Group 2 in the fourth production cell, and the finished product warehouse on the extreme right. Personnel work only in the production cells (machine groups) taking 1 minute to move from cell to cell or 2 minutes to traverse the length of the three cells. The material handling system can carry only one lot at a time and requires one minute to load and one minute to unload. It visits all five cells, taking 4 minutes to move from cell to cell or 16 minutes to traverse the length of the factory.

Finally, some restrictions are incorporated in the Mini-Fab model especially for the batching processing. In the diffusion machines the formation of a batch can be done with any combination of lots of Products A or B, but no more than one lot of Test Wafers. Also, the batch must contain three products of equal production step and less than two test product lots. At last, at the processing step 5, the batches cannot contain a mix of commercial products (A and B). In addition, in multi-machine workstations, test

products are processed once by both machines. So, when a test product enters the diffusion or the implantation workstation for the second time it can only be processed on the machine it was not processed on the first time.

In the two variants and the different scenarios of the Mini-Fab model that we analyzed in the next Sections, we made the following assumptions:

1) No buffer sizes are modeled for the tool stations.
2) Neither maintenance technicians nor operators are modeled.
3) No transport limits are imposed.
4) Batches in Machine Group 1 can be formed by mixing lots in any combination of products.
5) Each machine in Machine Group 2 (Machines C and D) is down for emergency (unscheduled) maintenance 14 hours per week instead of between 12 and 16 hours per week assumed in the original Mini-Fab model.
6) Scrapping and rework probabilities are as follows: $P_{js} = P_{es|js} = P_{ws|ps} = p_s$ and $P_{rw|nes} = P_{erw|rw} = P_{wrw|prw} = p_r$, for some values of p_s and p_r.
7) A lot may be sent for rework only once after each nominal operation. The rework operation following each nominal operation as a repetition of the nominal operation, i.e., it is performed on the same equipment and takes the same total processing time as the nominal operation.

As far as the scrapping and rework probabilities are concerned, we distinguish between four cases shown in Table 5.2.

Table 5.2: Four cases of scrapping and rework probabilities.

Case	Description	p_s	p_r
1	Neither scrapping nor rework	0	0
2	Rework but no scrapping	0	0.15
3	Scrapping but no rework	0.1	0
4	Both scrapping and rework	0.1	0.15

It should be noted that in the original Mini-Fab model, neither scrapping nor rework is modelled; therefore, the original model corresponds to case 1.

To evaluate the analytical queueing network methodology of Connors et al. [3] we compute the following performance metrics.

- Mean utilization and queueing delay for each machine group;
- Mean cycle time of each product family.

5.2 Two Nominal examples

In this Section, we describe the two of the examples that we studied, which are variants of the Mini-Fab model, and we discuss the resulting Mini-Fab model for the four cases shown in Table 5.2.

5.2.1 Example 1

In Example 1, we consider a variant of the Mini-Fab model described above with the following modifications in the assumptions:

- There are no setups.
- The loading, process, and unloading times are uniformly distributed with means equal to those shown in Table 5.1. The intervals of the uniform distributions are very tight and equal to only 1‰ of the means on either side; therefore, practically, the process times are close to deterministic.

The results are shown in Table 5.3 and Table 5.4, respectively.

Table 5.3: Mean utilization (MU) and mean queuing delay (MD) (in min) per machine group for Example 1.

Machine Group	Case 1		Case 2		Case 3		Case 4	
	MU	MD	MU	MD	MU	MD	MU	MD
1	0.9793	153.1077	0.9849	190.6113	0.9735	128.2448	0.9794	149.8951
2	0.8333	59.7915	0.8576	74.0378	0.8099	48.9940	0.8328	58.9394
3	0.9167	276.5681	0.9531	481.9089	0.8703	178.2601	0.9041	242.8269

Table 5.4: Mean cycle time (in hrs) per product family for Example 1.

Product family	Case 1	Case 2	Case 3	Case 4
A	30.3989	44.8139	25.9333	33.5095
B	30.3989	44.8139	25.9333	33.5095
Test wafers	30.3989	44.8139	25.9333	33.5095

From the results of Example 1 we can make the following observations.

When the rework probability p_r is increased from zero to a positive value (compare case 1 vs. case 2 and case 3 vs. case 4), the mean utilization, mean queuing delay and mean cycle time increase. This is expected because with rework, the system becomes more utilized and more congested. Instead, when the scrapping probability p_s is increased from zero to a positive value (compare case 1 vs. case 3 and case 2 vs. case 4), the mean utilization, mean queuing delay and mean cycle time decrease. This is also expected because with scrapping the system is less utilized and less congested. The downside of scrapping, of course, is the decrease in throughput.

When both the rework probability p_r and the scrapping probability p_s are increased from zero to positive values (compare case 1 vs. case 4), both effects take place, i.e., there is a tendency for the mean utilization, mean queuing delay and mean cycle time to both increase and decrease. For the set of parameters that we considered, it turns out that while the mean utilization and mean queuing delay decrease, the mean cycle time increases. A reason to that could be that the probability we are going for rework is a bit greater than the probability of scrapping and so the cycle time is bigger comparing to the case 1 where we have neither scrapping nor rework.

According to Connors et al. [3], the mean utilization for each machine is calculated as the percentage of the total time that the machine is being utilized either processing jobs or doing other tasks like setups scheduled/unscheduled maintenance, etc. Kempf [6] calculates the utilization as the ratio of the time that the machine is processing jobs over the time that the machine is available to process jobs, where by available we mean that it is not doing other tasks like setups scheduled/unscheduled maintenance, etc. Once the utilization defined in Kempf [6] is converted to the utilization defined by Connors et al. [3], the mean utilization values for Machine Groups 2 and 3 in Table 5.3

agree with the utilization values calculated in Kempf [6]. For Machine Group 1, the utilization displayed in Table 5.3, which is calculated based on Connors et al. [3], is higher than that calculated in Kempf ([6]). The reason for this is that in Connors et al. [3] the machines in Machine Group 1 are assumed to operate under a greedy batching policy, which means that each machine can start processing a batch of lots even if the maximum batch size has not been reached. The calculation of the utilization in Kempf [6], on the other hand, assumes that the size of all batches is equal to the maximum batch size.

All product families have the same mean cycle time values. This is expected, since they all have the same processing sequence and processing times.

5.2.2 Example 2

In Example 2, we consider another variant of the Mini-Fab model described above with the following modifications in the assumptions:

- There is no preventive (scheduled) maintenance in any of the machines.
- Machine E in Machine Group 3 requires a 10-min setup twice per shift. This value is close to the value of a 24-min setup per shift assumed in van den Berk [15].
- The loading, process, and unloading times are uniformly distributed with means equal to those shown in Table 1, except for the process time of step 5 in Machine Group 1 which has a mean of 225 min instead of 255 min and the unload times of steps 1 and 5 in Machine Group 1 which have a mean of 20 min instead of 40 min. The intervals of the uniform distributions are very tight and equal to only 1‰ of the means on either side; therefore, practically, the process times are close to deterministic.

The results of the analysis are shown in Table 5.5 and Table 5.6.

Table 5.5: Mean utilization (MU) and mean queuing delay (MD) (in min) per machine group for Example 2.

Machine Group	Case 1		Case 2		Case 3		Case 4	
	MU	MD	MU	MD	MU	MD	MU	MD
1	0.9557	84.1840	0.9629	92.2022	0.9487	77.2387	0.9561	83.2401
2	0.6667	20.4343	0.6909	23.6485	0.5599	17.6496	0.6661	20.2133
3	0.9028	238.3171	0.9392	377.0837	0.8286	160.9358	0.8902	212.8048

Table 5.6: Mean cycle time (in hrs) per product family for Example 2.

Product family	Case 1	Case 2	Case 3	Case 4
A	24.3478	33.7500	21.4441	26.9774
B	24.3478	33.7500	21.4441	26.9774
Test wafers	24.3478	33.7500	21.4441	26.9774

From the results we can make the same observations as those we made for Example 1 as well as the following additional observation.

The mean utilization, queuing delay and cycle time for all the cases of Example 2 are lower than their respective values in Example 1. The reason for this is that, while the characteristics of the emergency (unscheduled) maintenance are the same in both examples, the times of certain tasks that use up machine capacity are lower in Example 2 than they are in Example 1. More specifically, the preventive (scheduled) maintenance times in Example 2 are zero whereas they are positive in Example 1, and the mean process time of step 5 and the unload times of steps 1 and 5 in Machine Group 1 in Example 2 are lower than their respective values in Example 1. The only task that takes longer in Example 2 than it does in Example 1 is the setup of Machine E in Machine Group 3, but this time is not high enough to cause higher machine utilization, delays and cycle time in Example 2.

5.3 Variants of the Example 1

Taking occasion of the previous variants of the original Mini-Fab model, we decided to execute many different cases (scenarios) of the Mini-Fab, as it was presented

in Section 5.1, by changing each running time a specific parameter. This strategy aims to notice how each factor affects the basic model.

The basic model we agreed to be the original Mini-Fab model including the assumptions we made in Section 5.1 and corresponds to the case that we have neither scrapping nor rework. Then, we chose eight different parameters of the model to change with alternative case values for each parameter. In particular, the parameters we chose to change are:

1. The scrapping probabilities, $P_{js} = P_{es|js} = P_{ws|ps} = p_s$, which are quantified in probability levels of 0%, 10% or 20%.
2. The rework probabilities, $P_{rw|nes} = P_{erw|rw} = P_{wrw|prw} = p_r$, which are also quantified with the same three case-values of 0%, 10% and 20%.
3. The time durations of the incapacitation events (setup, scheduled and unscheduled maintenance) we assume that are uniformly distributed with means equal to those presented for each Machine Group at Section 5.1. The intervals of these uniform distributions are specified by a percentage of the means noted by $e1$. This percentage changes to 1‰, 10‰ and 100‰ each time, reflecting either a tight, practically deterministic, interval or a slack interval.
4. The loading, process, and unloading times are uniformly distributed with means equal to those shown in Table 5.1. The intervals of the uniform distributions are also specified by a percentage of the means, $e2$. This percentage equals as well to 1‰, 10‰ and 100‰ reflecting either a tight, practically deterministic, interval or a slack interval.
5. We suppose that from the three cases of setup changes in Machine Group 3, only one setup change is modelled with setup time equals to 10 minutes. We distinguish between two cases where no setup is performed, i.e. setup with zero time, or a 10-min setup is conducted.
6. The arrival rate of setups is changed depending in the possibility if setups is actually performed on that specific occasion. When no setups are performed, then the arrivals of setup-changes are zero. Instead, when it is decided the 10-min setups to be performed on the machine E, then three distinct cases of arrival rates

are modelled for the setup-changes, e.g. a low level of 1 arrival of setup-changes per shift, a median level of 2 arrivals per shift and a high level of 3 arrivals per shift.

7. We assumed that each machine in Machine Group 2 (Machines C and D) is down for emergency (unscheduled) maintenance 14 hours per week. Now, we model an alternative case in which the emergency maintenance (breakdown) lasts 7 hours per half-week. (This sentence is modelled as two different parameters, one for the time duration of the breakdown and the second that represents the arrival rate). In the mean time the two cases are identical but have a different effect on the system.

Briefly, the alternative cases of the parameters are summarized in the following two Tables.

Table 5.7: Cases of scrapping and rework probabilities, cases of the percentage of the means that determine the intervals.

p_s	p_r	$e1$	$e2$
0	0	1‰	1‰
0.1	0.1	10‰	10‰
0.2	0.2	100‰	100‰

Table 5.8: Setup time (ST) (in min) and frequency of setups (FS) (in arrivals per shift) at each machine of machine group 2, time duration of unscheduled maintenance (DM) (in hours) and frequency of unscheduled maintenances (FM) (in arrivals per week) at each machine of machine group 3.

Setup		Unscheduled Maintenance	
ST	FS	DM	FM
0	0	14	1
10	1	7	2
10	2		
10	3		

Hence, combining all the above mentioned instances of changes, they are extracted a total of 216 different scenarios of the basic Mini-Fab model. (We list the 216 different cases at Appendix B). The first scenario (scenario 1) is our basic Mini-Fab model.

It is noted that the two different cases of the unscheduled maintenance are modeled for our own convenience as follows:

- Failures arrive at each machine in Machine Group 2 (Machines C and D) twice a week and each machine requires a repair time of 420 min (7 hrs), instead of 14 hours per week for each machine that an emergency (unscheduled) maintenance was lasted;
- In the same manner, for the second case, failures arrive at each machine in Machine Group 2 (Machines C and D) twice per half-week and each machine requires a repair time of 210 min (3.5 hrs), instead of a breakdown with 7 hours per half-week for each machine.

5.4 Discussion of the Results

The results of the various scenarios that were presented in the previous Section are listed in Appendix C. Here, we make some observations on how each parameter affects the performance metrics of the system. First of all, the observations that we made in Section Part 5.2.1 for the first example, which is our basic case (Scenario 1), also apply to this Section for the different scenarios.

As far for the influence of the probabilities is concerned, when we increase the value level of the rework probability the mean utilization, mean queuing delay and mean cycle time are also increased. In contrast, when the scrapping probability is increased from zero to a positive value the mean utilization, mean queuing delay and mean cycle time decrease. In the interim cases, when both rework and scrapping probabilities are increased from zero to positive values the performance measures have the tendency either to increase or decrease, depending on which probability is greater than the other. A significant observation is made out when both probabilities have the same positive value. In these scenarios (compare scenario 5 vs. scenario 9) the mean utilization, mean queuing delay and mean cycle time are decreased compared to the basic case (scenario 1) where we have zero probabilities for rework and scrapping processes. This can be justified from a view that, lots removed due to scrap would reduce the traffic of the downstream toolsets

so less jobs would go for rework, although the probability for scrapping and rework is the same. The above tendencies are repeated in each change cases of the other parameters.

Changing the percentage of the means used to determine the intervals of the uniform distribution, which is followed from the load, process and unload processes, also as well the incapacitation processes, it is observed that the increase of this percentage forces the intervals to widen. So, the metrics of mean queuing delay and mean cycle time have the tendency to increase, while the mean utilization is stable. For 10‰ of $e1$ and $e2$, the difference in the increase is not great (compare scenario 10 vs. scenario 1), but for 100‰ of $e1$ and $e2$, the increase in the mean queuing delay and mean cycle time is obvious (compare scenario 19 vs. scenario 1). This tendency can be evaluated comparing scenarios in which we have changed other parameter's characteristics, such as for setup or/and unscheduled maintenance (compare scenario 37 vs. scenario 28 and scenario 46 vs. scenario 28, etc.).

The incapacitation event of setup has a great impact on the performance metrics of machine group 3. More specifically, increasing the setup time from zero to 10 minutes and simultaneously setting the arrivals processes of setups at the low level of 1 arrival per 12-hour shift, we observe that the mean utilization and mean queueing delay is increasing only for the machine group 3. This is expected because setups are performed only on the machines of tool group 3 and so that the utilization and the mean queueing delay of the other two machines groups are not affected at all (compare scenario 28 vs. scenario 1). Moreover, the mean cycle time of the three products is increased. This is also logical because the machine group 3 is delaying the whole system.

Next, increasing the value of the mean rate of arrivals of setups to the median level of 2 arrivals per shift and the high level of 3 arrivals per shift, we come across with some important results. Firstly, the mean utilization and the mean queueing delay of the machine group 3 and the mean cycle time of the product families are increased for all scenarios based on the different values of p_s, p_r and $e1$, $e2$. However, there are scenarios where the cycle times present negative values as well the same happens for the mean queueing delay of machine group 3. This fact is confirmed by the observation that the mean utilization at machine group 3 exceeds the one. This value for utilization is equal

strange and wrong, because it is not feasible for a machine to operate over 100% of its production capacity. These specific scenarios are determined by the state of the system in which the rework probability p_r equals to the high value level of 20% as the scrapping probability p_s is zero, i.e. the processed jobs have only the possibility to go for rework but cannot be scrapped.

The last sentence may explain the unfeasible results, since the system becomes more utilized and more congested. In combination with the tendency of increase of the performance metrics caused by the increase of the arrival rate of setups, the negative results imply that the system require more time to process all amount of nominal and reworked jobs. This results in negative cycle times for the products due to lack of extra time for processing in the machine group 3 (negative queueing delay). That is why the machine group 3 works in greater production capacity of what it can provide (utilization over one). All the observations denoted in this paragraph can be verified comparing scenarios 57, 66, 75, 84, 93 and 102 vs. scenarios 3, 12, 21, 30, 39, 48.

To prove the problem of the over-congestion that is mentioned previously, we ran some extra scenarios in which the probability of rework is 15 percent instead of 20 percent while at the same time the scrapping probability is zero. Practically, we ran the same scenarios which have the problem with the negative values in delay and cycle times with the difference that the rework probability p_r is 15%. From this analysis, we conclude that the system with probability of rework equals to 15% and setup arrivals of two and three arrivals per shift operates to full. This is happening since the utilization of machine group 3 is 98% at least at these scenarios, the mean queueing delay at the machine group 3 is very high and therefore the mean cycle times are very big (compare scenarios 57A, 66A, 75A, 84A, 93A, 102A vs. scenarios 3, 12, 21, 30, 39, 48). However, the system seemed to be over-congested even when the arrivals of setups in machine group 3 were only one per shift (see scenarios 30, 39, 48). In these scenarios, the mean utilization of machine group 3 is approximately 99.4%, the mean queueing delay in this tool group takes the extravagant time of 3260-3268 minutes, while the mean cycle time of the products is about to 160 hours. We understand that if any of the factors increase, the system would be infeasible.

Ending up, we observe that the system in the above scenarios with rework probability $p_r = 15\%$ is working to its maximum capacity, especially referred to machine group 3, and if we increase a little bit this probability by a 5% ($15 \rightarrow 20\%$) then the productivity of machine group 3 and the total time that the system offers for the products, including processing and queue time, are insufficient. (compare scenarios 57, 66, 75, 84, 93, 102 vs. scenarios 57A, 66A, 75A, 84A, 93A and 102A).

Finally, the last parameter that is changing is the frequency of the failures which occur at machine group 2. Comparing the scenarios where the machines C and D (machine group 2) are down for 14 hours in a week in one hand, or in the other hand they are down for 7 hours per half-week, we conclude based on the results that the values of mean queueing delays and cycle times are not differentiate almost at all, (the utilizations are identical). Of course, this is expected and something else would be wrong because the two cases of modelling the unscheduled maintenance have the same effect on the system in an average of time. So, there is no disagreement in the performance measures, since their mean values are calculated. (compare scenario 109 vs. scenario 1, scenario 136 vs. scenario 28, etc.).

Chapter 6

Conclusions

6.1 Summary

We presented the open queueing network model developed by Connors et al. [3] for rapid performance analysis of semiconductor manufacturing facilities. This model differs from other queueing models for performance evaluation of manufacturing systems in the detailed ways in which the effect of rework and scrap on wafer lot sizes is characterized and different tool groups found in semiconductor wafer fabrication are modeled.

Especially, a high-volume semiconductor wafer fabrication facility consists of several hundred tools grouped into dozens of distinct tool groups. Collections of wafers (jobs) move from tool group to tool group undergoing hundreds of operations before completion. Each job follows a process sequence that corresponds to the product family to which the job belongs. The process sequence is specified by a consecutive list of operations to be performed on the wafers in the job. The highly reentrant flow of jobs through the fab causes jobs make multiple visits to tool groups. The jobs, or portions of jobs, may be scrapped or/and go for rework during the course of being processed. Thus, the number of wafers in a job is changing as the job moves through the fab. So, the first basic step in our analyzation was the determination of the job-size distribution for the jobs that are not being scrapped, also for the jobs that go for rework.

Then, the decomposition approach is implemented at the queueing model of Connors et al. [3] to decompose the complex system's network into individual queues. The decomposition method uses the first moment traffic equations in order to interpret more efficient and accurate the characteristics of the tool groups, represented by the individual queues. The first moment traffic equations are formulated and solved at the operation level instead of the tool level and only then summed for every queue.

The first two moments of the effective arrival rates at each node are obtained by solving two systems of linear equations, which synthesize the effects of the three basic network operations (superposition, splitting and departure) on the first and second moments of the arrival rates, respectively. In simple words, a system of linear equations that is just the familiar traffic rate equations occurring in the Jackson network of M/M/m queues (Markov models) is used for the rates, and after having obtained the rates, we obtain the variability parameters (the squared coefficients of variations) of the internal flows by solving another system of linear equations.

In the main part of the analysis, we specify the features of three different tool types: single-wafer tools with one machine and multiple (parallel) machines and batch tools. Basically, we define the SCV of the service time by calculating the first two moments of the service time at each node as the weighted combination of the service times of all the operations performed on this node (machine group). After obtaining the first two moments of the effective arrival rates and service time of all the nodes, the performance measures of the mean utilization and mean waiting time in the queue are calculated from exact and approximated methods. In one case of batch tools analysis, a "greedy" policy was followed. It is noted that in our modelled system the incapacitations events of the tool breakdowns, the preventative maintenance and the setup requirements, which make the manufacturing environment highly stochastic, are modelled as a non-preemptive priority jobs arriving at the tool groups. Finally, we calculate the average cycle time of each product family considering that the rework operations are performed on the same tool group and takes the same total processing time as its nominal operation.

To implement the analysis of Connors et al. [3] for a semiconductor manufacturing queueing network model, we developed a computational model at the

environment of the software of MATLAB-MathWorks. The results that were derived from executing the model in many different variants of the Mini-Fab model introduced by Kempf [6] seem reasonable. When scrapping probabilities increase then the performance metrics of average utilization and mean queueing delay at each of the toolgroup as well the mean cycle time for each product family decrease due to the decongestion of the system. Instead, when the rework probabilities increase the system becomes more utilized and thus cycle time and queueing delay of the toolsets would increase.

Comparing our results with other literature examples, we observe that our results for the single-wafer tool types agree with the second ones, though the modelling of the batch tools seem problematic. A reason that the performance metrics for the batch tools are higher than from corresponding bibliographic examples is that Connors et al. [3] assumes a greedy batching policy. Finally, we ran a lot of variants of the Mini-Fab case in order to clarify the effectiveness of each parameter in the system.

6.2 Expectations & Future research

Thorough understanding of the fab operation was critical to queueing network modeling. Based on the results that were obtained from the study, it can be concluded that the model did verify its expected behavior. Considering the practical manufacturing process scenarios of the semiconductor industry, the dissertation model encapsulates the real IC fabrication process, given the accuracy of the data that was available. The data used to model the fabrication process was obtained from literature models assuming that the time requirements follow the uniform distribution. Hence the model could have been more accurate if access to real data were available, or if real data were fitted to a probability distribution. Since this model instantiates the fabrication process of three products at one time in a representative fab, this model could be used in the industry to simulate the fabrication process by modifying the model components to match the architecture of the fab. Several inferences like the analysis of the interactions between the

factors that participate in the system and machine failures on cycle times can be made using this model.

Also, model type effects on machine utilization, throughput, lot sizing, work-in process could be studied in detail using these models. But, the biggest challenge in gauging and improving operational performance is the nature of the semiconductor manufacturing process itself – characterized by complex reentrant flows, large variations in raw process times, differences in batch sizes for tool sets, cascading tool sets, and a rich set of complexity in the tools themselves. This results in a significant portion of lead time being non-productive queue time (waiting to be serviced) and a well-known tradeoff between reducing cycle time and increasing equipment utilization sometimes known as the operating curve. Its research analysis would provide the expectations of a reduced cycle time, which is always highly desirable, in order to enable faster yield learning, resulting in an increase in good chips per wafer earlier in a development cycle.

To end up, the results from the implementation of Connors et al. [3] approach seem reasonable and promising, something that make us to ascertain that we developed a reliable computational model. Therefore, the findings and conclusions from this research thesis provide opportunities for further research in the same area, although, how to extract the input parameters for the queueing network analyzer from real factory data remains a formidable challenge.

Bibliography

[1] http://www.mie.uth.gr/n_news_one.asp?id=295.

[2] Brown S. M., Hanschke T., Meents I., Wheeler B. R., Zisgen H. "Queueing Model Improves IBM's Semiconductor Capacity and Lead-Time Management" (2010). Article by Institute for Operations Research and the Management Sciences (INFORMS). Interfaces Vol.40, No.5, pp 397-407.

[3] Connors D. P., Feigin G. E., Yao D. D. "A Queueing Network Model for Semiconductor Manufacturing" (1996). IEEE Transactions on Semiconductor Manufacturing Vol.9, No.3, pp 412-427.

[4] Grosbard D., Kalir A., Tirkel I., Rabinowitz G. "A Queuing Network Model for Wafer Fabrication Using Decomposition without Aggregation" (2013). IEEE International Conference on Automation Science and Engineering (CASE), pp 717-722.

[5] Hopp W. J., Spearman M. L. "Factory Physics - Foundations of Manufacturing Management" (second edition) (2000). Chapter 8.

[6] Kempf K. "Intel Five-Machine Six Step Mini-Fab Description" (1994). Available online via http://aar.faculty.asu.edu/research/intel/papers/fabspec.html.

[7] Li H., Ramirez-Hernandez J. A., Fernandez E., McLean C. R., Leong S. "A Framework for Standard Modular Simulation: Application to Semiconductor Wafer Fabrication" (2005). Proceedings of the Winter Simulation Conference, pp 1-17.

[8] Meng G., Heragu S. S. "Batch size modeling in a multi-item, discrete manufacturing system via an open queuing network" (2002). IIE Transactions 36, pp 743–753.

[9] Mönch L., Fowler J. W., Mason S. J. “Production Planning and Control for Semiconductor Wafer Fabrication Facilities - Modeling, Analysis, and Systems” (2013). Chapters 2, 3.

[10] Ramamurthi V. “Analysis of production control methods for semiconductor research and development fabs using simulation” (2004). Thesis, Rochester Institute of Technology, Chapter 1.

[11] Ramirez-Hernandez J. A., Fernandez E. “A Simulation-Based Approximate Dynamic Programming Approach for the Control of the Intel Mini-Fab Benchmark Model” (2009). Proceedings of the 2009 Winter Simulation Conference, pp 1634-1645.

[12] Shanthikumar G. J., Shengwei D., Zhang M. T. “Queueing Theory for Semiconductor Manufacturing Systems: A Survey and Open Problem” (2007). IEEE Transactions on Automation Science and Engineering Vol.4, No.4, pp 513-522.

[13] Shinde A. “Modeling and simulation of a Semiconductor Manufacturing Fab for cycle time analysis” (2018). Master of Science in Systems Engineering, University of Maryland.

[14] Spier J., Kempf K. “Simulation of Emergent Behavior in Manufacturing Systems” (1995). IEEE/SEMI Advanced Semiconductor Manufacturing Conference, pp 90-94.

[15] van den Berk J.P.A. “Analysis of the Intel Five-Machine Six Step Mini-Fab” (2004). Master Thesis, Eindhoven University of Technology.

[16] Whitt W. “The Queueing Network Analyzer” (1983). The Bell System Technical Journal Vol.62, No.9, pp 2779-2815.

[17] Zisgen H., Meents I., Wheeler B. R., Hanschke T. “A Queueing Network Based System to Model Capacity and Cycle Time for Semiconductor Fabrication” (2008). Proceedings of the Winter Simulation Conference, pp 2067-2074.

Appendices

Appendix A: Code of the Computational Model

```
clear all
close all
clc

InputData=xlsread('DATA.xlsx','InputData','B2:I217');
Cases=size(InputData,1);
for Case=1:Cases

rng(1);                    % initialize random number generator seed

G=3;                       % number of tool groups
F=3;                       % number of product families
Nf=[6 6 6];                % number of operations per product family
seq=[1 2 3 2 1 3];         % sequence of tool groups per product family where each operation is performed
c=[2 2 1];                 % number of machines at each toolgroup

EJEWt=[225 30 55 50 255 10];      % run processing time (in minutes) for a lot(job) of each process step -
                                  % per product family without load, unload times and setup times
Load=[20 15 10];                  % load times (in minutes) for each tool group
Unload=[40 15 10];                % unload times (in minutes) for each tool group

nmax=1;                           % maximum number of wafers in a job
A=[51*nmax 30*nmax 3*nmax];       % exogenous arrival rates of wafers per time-unit(week) for each -
                                  % product family into the fab
A=A/(7*24*60);                    % exogenous arrival rates per minute (7days/24hours/60mins)
bmax=[3*nmax 1 1];                % maximum batch size of wafers at each toolgroup
max_mmax=1;                       % maximum allowable number of times that a job is sent for rework -
                                  % after each operation
nnmax=4;

B=3;                                          % number of incapacitation events (preventative -
                                              % maintenance, breakdown and major setup)
Lamdainc=[2/(24*60) 2/(12*60) 1/(12*60);      % arrivals of incapacitation events at each toolgroup
      0 InputData(Case,8) 0
      0 0 InputData(Case,6)];
ESinc=[75 120 30;                             % mean processing times of incapacitation events
    0 InputData(Case,7) 0;
    0 0 InputData(Case,5)];
```

```
e1=InputData(Case,3);          % percentage of dispersion from the mean value for incapacitation's -
                               % events time requirements
e2=InputData(Case,4);          % percentage of dispersion from the mean value for tool's time -
                               % requirements

N=sum(Nf);                     % total number of operations
u=repelem(1:F,Nf);             % product family of each operation
s=cumsum(Nf)-Nf+1;             % first operation of each product family
nom=(1:N)+1;                   % nominal successor operation of operation k for all operations
nom(1,cumsum(Nf))=0;           % nominal successor operation of last operation of each product family
rew=(sum(Nf)+1:sum(Nf)+N);     % rework successor operation of operation k for all operations (each has -
                               % the value k+N)
t=[seq seq seq];               % tool group where each nominal operation is performed
t(rew)=[seq seq seq];          % tool group where each rework operation is performed
EJEW=[EJEWt EJEWt EJEWt];
EJEW(rew)=[EJEWt EJEWt EJEWt];

%%

% Scrap and Rework Probabilities section (Initializations)

ps=InputData(Case,1);
pr=InputData(Case,2);

Pjs= ps*ones(1,N);             % prob that a job is scrapped after op k
Pesjs= ps*ones(1,N);           % prob that a job is entirely scrapped given it is scrapped after op k
Ppsjs=1-Pesjs;                 % prob that a job is partially scrapped given it is scrapped after op k
Pwsps= ps*ones(1,N);           % prob that a wafer is  scrapped given it is in a job that is partially -
                               % scrapped after op k
Prwnes= pr*ones(1,N);          % prob that a job is reworked given that it is not entirely scrapped after op k
Perwrw= pr*ones(1,N);          % prob that a a job is entirely reworked given it is reworked after op k
Pprwrw=1-Perwrw;               % prob that a a job is partially reworked given it is reworked after op k
Pwrwprw= pr*ones(1,N);         % prob that a wafer is reworked given it is in a job that is partially reworked -
                               % after op k

Pws=Pjs.*(Pesjs+Ppsjs.*Pwsps);                          % prob that a wafer is scrapped after op k
Pwrw=(1-Pws).*Prwnes.*(Perwrw+Pprwrw.*Pwrwprw);         % prob that a wafer is reworked after op k

% Creation of Probability Distribution Prwm that a job is sent for rework exactly m times after operation k
mmax = randi(max_mmax,1,N);
max_mmax = max(mmax);
Prwm = zeros(N, max_mmax);
for k = 1:N
   Prwm(k,1:mmax(k)) = rand(1,mmax(k));
   Prwm(k,1:mmax(k)) = Prwm(k,1:mmax(k))/sum(Prwm(k,1:mmax(k)));    % Normalized Prwm
end

% Initializations of our main Prob Distributions
Probin=zeros(F,nmax+1);                % Prob(f,n) = Prob that a job arriving to first op of product f, -
                                       % k=s(f), has n wafers
Probnom=zeros(N,nmax+1);               % Prob(k,n) = Prob that a job departing from op k has n wafers
Probrewm=zeros(N,nmax+1,max_mmax);     % Prob(k,n) = Prob that a job arriving at mth rework op after
```

```
                                        % op k has n wafers
Probrew=zeros(N,nmax+1);                % Prob(k,n) = Average prob that a job arriving at rework op
                                        % after op k has n wafers

% Initializations of auxiliary resources for testing different computational methods
Probnom1=zeros(1,N);
Diffnom=zeros(1,N);
Probrewm1=zeros(N,max_mmax);
Diffrewm=zeros(N,max_mmax);

%%

% Job-Size Distributions section

% Calculate job-size distribution of jobs not scrapped
for f=1:F
   Probin(f,1:nmax)=0;
   Probin(f,nmax+1)=1;

   for k=s(f):s(f)+Nf(f)-1
      p=Pwsps(k);
      for n=2:nmax+1
         Prob(1:n-1)=zeros;
         Pexact=zeros(1,nmax);
         ntemp=n-1;
         if k==s(f)
            Prob(n:nmax+1)=Probin(f,n:nmax+1);
            Prob1=Probin(f,2:nmax+1);
         else
            Prob(n:nmax+1)=Probnom(k-1,n:nmax+1);
            Prob1=Probnom(k-1,2:nmax+1);
         end
         for i=ntemp:nmax
            Pexact(i)=binopdf(i-ntemp,i,p);
         end
         Probnom(k,n)=Prob(n)*(1-Pjs(k))+Pjs(k)*Ppsjs(k)*sum(Prob(n:nmax+1).*Pexact(ntemp:nmax));
      end
      Probnom(k,1)=1-sum(Probnom(k,2:nmax+1));
      Probnom(k,2:nmax+1)=Probnom(k,2:nmax+1)/(1-Probnom(k,1));          % normalization
%        Probnom(k,2:nmax+1)=Probnom(k,2:nmax+1)/sum(Probnom(k,2:nmax+1));

      % Compute Prob(J=0), Probability job contains zero wafers at operation k, using Connors et al. -
      % formula
      Pexact1=zeros(1,nmax);
      for i=1:nmax
         Pexact1(i)=p^i;
      end
      Probnom1(k)=Pjs(k)*Pesjs(k)+Pjs(k)*Ppsjs(k)*sum(Prob1.*Pexact1);
      Diffnom(k)=Probnom1(k)-Probnom(k,1);    % Checking that both computational formulas yield to the -
                                              % same Probnom(J=0)

   end
```

```
end

% Calculate job-size distribution of jobs sent to rework for the m_th time
for k=1:N
   Prob=Probnom(k,:);
   Prob1=Probnom(k,2:nmax+1);
   p=Pwrwprw(k);
   for m=1:mmax(k)
      for n=2:nmax+1
         Pexact=zeros(1,nmax);
         ntemp=n-1;
         for i=ntemp:nmax
            Pexact(i)=binopdf(ntemp,i,p);
         end
         Probrewm(k,n,m)=Prob(n)*Prwnes(k)*Perwrw(k)+Prwnes(k)*Pprwrw(k)*sum(Prob(n:nmax+1)
                              .*Pexact (ntemp:nmax));
      end
      Probrewm(k,1,m)=1-sum(Probrewm(k,2:nmax+1,m));
      Probrewm(k,2:nmax+1,m)=Probrewm(k,2:nmax+1,m)/(1-Probrewm(k,1,m));          % normalization
%       Probrewm(k,2:nmax+1,m)=Probrewm(k,2:nmax+1,m)/sum(Probrewm(k,2:nmax+1,m));

      % Compute Prob(J=0), Probability job senting for rework contains zero wafers after op k, using
      % Connors et al. formula
      Pexact1=zeros(1,nmax);
      for i=1:nmax
         Pexact1(i)=(1-p)^i;
      end
      Probrewm1(k,m)=(1-Prwnes(k))+Prwnes(k)*Pprwrw(k)*sum(Prob1.*Pexact1);
      Diffrewm(k,m)=Probrewm1(k,m)-Probrewm(k,1,m);    % Checking that both computational formulas
                                                       % yield to the same Probrewm(J=0)

      Prob=Probrewm(k,:,m);
      Prob1=Probrewm(k,2:nmax+1,m);
      Probrew(k,:)=Probrew(k,:)+Probrewm(k,:,m)*Prwm(k,m);  % Running average job-size distribution of -
                                                            % jobs sent to rework for up to the mth -
                                                            % time
   end
end

%%

% Traffic Rates section

% [N nominal operations],[N rework operations],[1 exit operation],[1 scrap operation]
% (N+N)+1 : state of exit for each product family, (N+N)+2 : state of scrapping
% Operations===States

% Q(matrix of probabilities that a wafer after completing operation j is routed to operation k)
% Qf(routing matrix for each product family)
Q=zeros;    % Q = [ (N+N)+2 , (N+N)+2 ]
Qf=zeros;   % Qf(f) = [ 2*Nf(f)+1 , 2*Nf(f)+1 ]

len=zeros;
```

```
lenold=0;
i=0;
for f=1:F
   % Create matric Q
   for k=s(f):s(f)+Nf(f)-2
      Q(k,rew(k))=Pwrw(k);              % Prob wafers go for rework
      Q(k,nom(k))=1-Pwrw(k)-Pws(k);     % Prob wafers go to nominal successor
      Q(k,rew(N)+2)=Pws(k);             % Prob wafers go for scrapping
      Q(rew(k),nom(k))=1;               % Prob=1 after rework go to nominal operation
   end
   k=s(f)+Nf(f)-1;                      % last operation of each product family f
   Q(k,rew(k))=Pwrw(k);
   Q(k,rew(N)+1)=1-Pwrw(k)-Pws(k);
   Q(k,rew(N)+2)=Pws(k);
   Q(rew(k),rew(N)+1)=1;

   Q(rew(k)+1,:)=0;                     % Prob wafers from exit stage go to a line operation
   Q(rew(k)+2,:)=0;                     % Prob wafers from stage of scrapping go to a line operation

   % Build matrix Qf from Q
   len(f)=0;
   for k=s(f):s(f)+Nf(f)-1
      i=i+1;
      Qf(i,1:Nf(f))=Q(k,s(f):s(f)+Nf(f)-1);
      Qf(i,Nf(f)+1:2*Nf(f))=Q(k,N+s(f):N+s(f)+Nf(f)-1);
      Qf(i+Nf(f),1:Nf(f))=Q(rew(k),s(f):s(f)+Nf(f)-1);
      Qf(i+Nf(f),Nf(f)+1:2*Nf(f))=Q(rew(k),N+s(f):N+s(f)+Nf(f)-1);
   end
   k=s(f)+Nf(f)-1;
   Qf(i,2*Nf(f)+1)=Q(k,rew(N)+1);
   Qf(i+Nf(f),2*Nf(f)+1)=Q(rew(k),rew(N)+1);
   Qf(i+Nf(f)+1,:)=0;

   len(f)=length(Qf);
   len(f)=len(f)-lenold;
   lenold=length(Qf);
   i=length(Qf);
end

% Calculate arrival rates to nominal operations of each product family (+ traffic rate of exit)

PF=zeros;
Pf=zeros;
lamda=zeros;
Lamda=zeros;
lenold=1;
Nfold=0;
for f=1:F
   I=eye(len(f),len(f));
   A(2:Nf(f)+1,:)=zeros;

%    [Pf] : Auxiliary invertible matrix with dimension (Nf(f)+1)x(Nf(f)+1), Nf(f)+1:state of exit
%    [PF] : PF=inv(I-transp(Qf))
```

```
   PF(1:len(f),1:len(f),f)=inv(I-(Qf(lenold:lenold+len(f)-1,1:len(f)))');
   Pf(1:Nf(f)+1,1:Nf(f)+1,f)=[PF(1:Nf(f),1:Nf(f),f) , PF(1:Nf(f),len(f),f) ; PF(len(f),1:Nf(f)+1,f)];   % formation -
                                                                                                        % by segments -
                                                                                                         % of matrix PF
   Pf(Nf(f)+1,Nf(f)+1,f)=PF(len(f),len(f),f);

   % Calculation of traffic rates of wafers to operation k
   lamda(1,s(f):s(f)+Nf(f))=Pf(:,:,f)*A(:,f);
%    lamda(1,1:Nf(f)+1,f)=Pf(:,:,f)*A(:,f);
   lamda(2,s(f)+Nf(f)-1)=lamda(1,s(f)+Nf(f));     % exit rate of product family f

   lamda=lamda(:,1:f*Nf(f));
   Nfold=Nf(f);
   lenold=lenold+len(f);

%    Compute arrival rates of JOBS to operation k of product family f
   for k=s(f):s(f)+Nf(f)-1
      if k==s(f)
         Lamda(1,k)=lamda(1,k)/sum((1:nmax).*Probin(f,2:nmax+1));
      else
         Lamda(1,k)=lamda(1,k)/sum((1:nmax).*Probnom(k-1,2:nmax+1));
      end
   end
   Lamda(2,k)=lamda(2,k)/sum((1:nmax).*Probnom(k,2:nmax+1));     % exit rate of JOBS of product family f
end

I2=eye(2*N+2,2*N+2);
A2=zeros(2*N+2,1);
for f=1:F
   A2(s(f))=A(1,f);
end
lamda2=(inv(I2-Q'))*A2;
lamda2=lamda2';

Lamda2=zeros;
for f=1:F
   for k=s(f):s(f)+Nf(f)-1
      if k==s(f)
         Lamda2(k)=lamda2(k)/sum((1:nmax).*Probin(f,2:nmax+1));
         if lamda2(rew(k))==0
            Lamda2(rew(k))=0;
         else
            Lamda2(rew(k))=lamda2(rew(k))/sum((1:nmax).*Probrew(k,2:nmax+1));
         end
      else
         Lamda2(k)=lamda2(k)/sum((1:nmax).*Probnom(k-1,2:nmax+1));
         if lamda2(rew(k))==0
            Lamda2(rew(k))=0;
         else
            Lamda2(rew(k))=lamda2(rew(k))/sum((1:nmax).*Probrew(k,2:nmax+1));
         end
      end
   end
```

```
end
Lamda2(rew(N)+1)=sum(Lamda(2,:));

 % Auxiliary Resources for traffic Variability equations

T=zeros;
Gamma=zeros;
for g=1:G
   T(1:length(find(t==g)),g)=find(t==g);       % Set of nominal operations performed at each toolgroup
   Gamma(g)=sum(Lamda2(1,T(:,g)));          % Total arrival rate of jobs to each toolgroup
end
% Gamma(G+1)=sum(Lamda(2,cumsum(Nf)));
Gamma(G+1)=Lamda2(rew(N)+1);

%{
 Calculate the total arrival rate of jobs to toolgroup h from toolgroup g and
 the proportion of arrivals to tool h came from toolgroup g.
 (G+1):state of outside(entrance or/and exit)
%}
gamma=zeros(G+1,G+1);    % traffic rates between toolgroups
thita=zeros(G+1,G+1);    % proportion of arrivals at toolgroups
for f=1:F
   for k=s(f):s(f)+Nf(f)-1
      if k==s(f)
         gamma(G+1,t(k))=gamma(G+1,t(k))+Lamda2(1,k)+Lamda2(1,rew(k));
         thita(G+1,t(k))=gamma(G+1,t(k))/Gamma(t(k));
      else
         gamma(t(k-1),t(k))=gamma(t(k-1),t(k))+Lamda2(1,k)+Lamda2(1,rew(k));
         thita(t(k-1),t(k))=gamma(t(k-1),t(k))/Gamma(t(k));
      end
   end
   gamma(t(k),G+1)=gamma(t(k),G+1)+Lamda(2,k);
   thita(t(k),G+1)=gamma(t(k),G+1)/Gamma(G+1);
end
ScrapRate=zeros;
for g=1:G
   ScrapRate(g)=Gamma(g)-sum(gamma(g,1:G+1));
end

% Compute the proportion of jobs leaving toolgroup g that proceed directly to tool h
r=zeros;
for f=1:F
   for k=s(f):s(f)+Nf(f)-1
      if k==s(f)
         r(G+1,t(k))=gamma(G+1,t(k))/sum(gamma(G+1,:));
      else
         r(t(k-1),t(k))=gamma(t(k-1),t(k))/sum(gamma(t(k-1),:));
      end
   end
   r(t(k),G+1)=gamma(t(k),G+1)/sum(gamma(t(k),:));
end

%%
```

```
% Processing & Other Time Requirements for Tools section

% incapacitation events: preventative maintenance, breakdown and major setup
corfinc=zeros;
SCVbs=zeros;
SCVba=zeros;
Lamdab=zeros;
Sb=zeros(B*G,3);
ESb=zeros(1,B*G);
ES2b=zeros(1,B*G);
LbES2b=zeros(1,G);
utilinc=zeros(1,G);
EB=zeros(1,G);
EB2=zeros(1,G);
k=0;
for i=1:B
   for j=1:G
      k=k+1;
      SCVba(k)=0;
      Lamdab(k)=Lamdainc(i,j);
      e=e1;
      EX=ESinc(i,j);
      Sb(k,:)=moments(EX,e);
      ESb(k)=Sb(k,1);
      ES2b(k)=Sb(k,3);
      if ESb(k)==0
         SCVbs(k)=0;
      else
         SCVbs(k)=(ES2b(k)-(ESb(k)^2))/(ESb(k)^2);
      end
      corfinc(k)=(SCVbs(k)+SCVba(k))/(SCVbs(k)+1);

      LbES2b(j)=LbES2b(j)+Lamdab(k)*ES2b(k)*corfinc(k);
      utilinc(j)=utilinc(j)+(Lamdab(k)*ESb(k))/c(j);          % Utilization of incapacitation events

      EB(j)=EB(j)+ESb(k)*(Lamdab(k)/Gamma(j));
      EB2(j)=EB2(j)+ES2b(k)*(Lamdab(k)/Gamma(j));
   end
end

pms=zeros;
M=zeros;
EM=zeros;
VM=zeros;
L=zeros(2*N,3);
EL=zeros;
VL=zeros;
U=zeros(2*N,3);
EU=zeros;
VU=zeros;
W=zeros(2*N,3);
EW=zeros;
```

```
VW=zeros;
EJ=zeros;
VJ=zeros;
ES=zeros;
VS=zeros;
ES2=zeros;
Eb=zeros;
Vb=zeros;
EY=zeros;
VY=zeros;
EY2=zeros;
for f=1:F
   for k=s(f):s(f)+Nf(f)-1
      pms(k)=rand();
      M(k)=randi([0 0],1);            % minor setup time(min) for a job
      EM(k)=pms(k)*M(k);
      VM(k)=(pms(k)*(M(k)^2))-(EM(k)^2);

      e=e2;
      EX=Load(t(k));
      L(k,:)=moments(EX,e);
      EL(k)=L(k,1);
      VL(k)=L(k,2);
      EX=Unload(t(k));
      U(k,:)=moments(EX,e);
      EU(k)=U(k,1);
      VU(k)=U(k,2);

      EJ(k)=sum(Probnom(k,2:nmax+1).*(1:nmax));
      VJ(k)=sum(Probnom(k,2:nmax+1).*((1:nmax).^2))-(EJ(k)^2);

      EW(k)=EJEW(k)/EJ(k);

      if bmax(t(k))>1            % Diffusion tool group(furnace) : processing times of wafers are the same
         e=0;
         EX=EW(k);
         W(k,:)=moments(EX,e);
      else
         e=e2;
         EX=EW(k);
         W(k,:)=moments(EX,e);
      end
      EW(k)=W(k,1);
      VW(k)=W(k,2);

      ES(k)=EM(k)+EL(k)+EJ(k)*EW(k)+EU(k);    % Mean Processing Time (total) for jobs
      VS(k)=VM(k)+VL(k)+EJ(k)*VW(k)+(EW(k)^2)*VJ(k)+VU(k);
      ES2(k)=VS(k)+(ES(k)^2);

      if k==s(f)
         Eb(k)=sum((ceil((1:nmax)./bmax(t(k)))).*Probin(f,2:nmax+1));
         Vb(k)=sum(((ceil((1:nmax)./bmax(t(k)))).^2).*Probin(f,2:nmax+1))-(Eb(k)^2);
      else
```

```
        Eb(k)=sum((ceil((1:nmax)./bmax(t(k)))).*Probnom(k-1,2:nmax+1));
        Vb(k)=sum(((ceil((1:nmax)./bmax(t(k)))).^2).*Probnom(k-1,2:nmax+1))-(Eb(k)^2);
      end
      EY(k)=Eb(k)*ES(k);
      VY(k)=Eb(k)*VS(k)+Vb(k)*(ES(k)^2);
      EY2(k)=VY(k)+(EY(k)^2);

    end
  end
  for f=1:F
    for k=s(f):s(f)+Nf(f)-1
      pms(rew(k))=rand();
      M(rew(k))=randi([0 0],1);
      EM(rew(k))=pms(rew(k))*M(rew(k));
      VM(rew(k))=(pms(rew(k))*(M(rew(k))^2))-(EM(rew(k))^2);
      e=e2;
      EX=Load(t(rew(k)));
      L(rew(k),:)=moments(EX,e);
      EL(rew(k))=L(rew(k),1);
      VL(rew(k))=L(rew(k),2);
      EX=Unload(t(rew(k)));
      U(rew(k),:)=moments(EX,e);
      EU(rew(k))=U(rew(k),1);
      VU(rew(k))=U(rew(k),2);
      if lamda2(rew)==0
        EJ(rew(k))=0;
        VJ(rew(k))=0;
        EW(rew(k))=0;
      else
        EJ(rew(k))=sum(Probrew(k,2:nmax+1).*(1:nmax));
        VJ(rew(k))=sum(Probrew(k,2:nmax+1).*((1:nmax).^2))-(EJ(rew(k))^2);
        EW(rew(k))=EJEW(rew(k))/EJ(rew(k));
      end
      if bmax(t(rew(k)))>1
        e=0;
        EX=EW(rew(k));
        W(rew(k),:)=moments(EX,e);
      else
        e=e2;
        EX=EW(rew(k));
        W(rew(k),:)=moments(EX,e);
      end
      EW(rew(k))=W(rew(k),1);
      VW(rew(k))=W(rew(k),2);
      ES(rew(k))=EM(rew(k))+EL(rew(k))+EJ(rew(k))*EW(rew(k))+EU(rew(k));
      VS(rew(k))=VM(rew(k))+VL(rew(k))+EJ(rew(k))*VW(rew(k))+(EW(rew(k))^2)*VJ(rew(k))+VU(rew(k));
      ES2(rew(k))=VS(rew(k))+(ES(rew(k))^2);
      Eb(rew(k))=sum((ceil((1:nmax)./bmax(t(rew(k))))).*Probrew(k,2:nmax+1));
      Vb(rew(k))=sum(((ceil((1:nmax)./bmax(t(rew(k))))).^2).*Probrew(k,2:nmax+1))-(Eb(rew(k))^2);
      EY(rew(k))=Eb(rew(k))*ES(rew(k));
      VY(rew(k))=Eb(rew(k))*VS(rew(k))+Vb(rew(k))*(ES(rew(k))^2);
      EY2(rew(k))=VY(rew(k))+(EY(rew(k))^2);
    end
```

```
end

%%

% Calculation of the utilization of each toolgroup and the SCV of service & inter-arrival time section

utiltool=zeros;
AvWaf=zeros;
ESg=zeros;
ES2g=zeros;
EYg=zeros;
EY2g=zeros;
EZ=zeros;
EZ2=zeros;
SCVs=zeros;
SCVz=zeros;
wmax=zeros;
corf=zeros;
utiladj=zeros;
util=zeros;
for g=1:G
   AvWaf(g)=sum(Lamda2(1,T(:,g)).*EJ(T(:,g)))/Gamma(g);
   if bmax(g)==1
      utiltool(g)=sum((Lamda2(1,T(:,g)).*ES(T(:,g)))/c(g));          % Utilization of single-wafer tools
      util(g)=utiltool(g)+utilinc(g);
      ESg(g)=sum((Lamda2(1,T(:,g)).*ES(T(:,g)))/Gamma(g));
      ES2g(g)=sum((Lamda2(1,T(:,g)).*ES2(T(:,g)))/Gamma(g));
      SCVs(g)=(ES2g(g)-(ESg(g)^2))/(ESg(g)^2);                       % SCV of the service time at station g for a -
                                                                     % "generic job" arriving to toolgroup g
      if any(Lamdainc(:,g)<=Gamma(g)) && c(g)>=5
         EZ(g)=ESg(g)+EB(g);
         EZ2(g)=ES2g(g)+EB2(g)+2*ESg(g)*EB(g);
         SCVz(g)=(EZ2(g)-(EZ(g)^2))/(EZ(g)^2);
         util(g)=Gamma(g)/((1/EZ(g))*c(g));
      end
   elseif bmax(g)>1
      if bmax(g)<=AvWaf(g)
         utiltool(g)=sum((Lamda2(1,T(:,g)).*EY(T(:,g)))/c(g));       % Utilization of batch tools
         util(g)=utiltool(g)+utilinc(g);
         EYg(g)=sum((Lamda2(1,T(:,g)).*EY(T(:,g)))/Gamma(g));
         EY2g(g)=sum((Lamda2(1,T(:,g)).*EY2(T(:,g)))/Gamma(g));
         SCVs(g)=(EY2g(g)-(EYg(g)^2))/(EYg(g)^2);
         if any(Lamdainc(:,g)<=Gamma(g)) && c(g)>=5
            EZ(g)=EYg(g)+EB(g);
            EZ2(g)=EY2g(g)+EB2(g)+2*EYg(g)*EB(g);
            SCVz(g)=(EZ2(g)-(EZ(g)^2))/(EZ(g)^2);
            util(g)=Gamma(g)/((1/EZ(g))*c(g));
         end
      elseif bmax(g)>AvWaf(g)
         ESg(g)=sum((Lamda2(1,T(:,g)).*ES(T(:,g)))/Gamma(g));
         ES2g(g)=sum((Lamda2(1,T(:,g)).*ES2(T(:,g)))/Gamma(g));
         SCVs(g)=(ES2g(g)-(ESg(g)^2))/(ESg(g)^2);
         EZ(g)=ESg(g)+EB(g);
```

```
        EZ2(g)=ES2g(g)+EB2(g)+2*ESg(g)*EB(g);
        SCVz(g)=(EZ2(g)-(EZ(g)^2))/(EZ(g)^2);

        corf(g)=Gamma(g)*EZ(g);
        wmax(g)=floor(bmax(g)/AvWaf(g));
        pp=zeros(1,wmax(g)+2);
        clear roots
        pp(1)=corf(g)/c(g);
        pp(2)=-(1+corf(g)/c(g));
        pp(wmax(g)+2)=1;
        roots=roots(pp);
        sroots=roots(imag(roots)==0);
        sroot=sroots(single(sroots)>1 & single(sroots)<(c(g)*wmax(g)/corf(g)));
        StBatProb00=(((corf(g)^c(g))/factorial(c(g)))*((1-(1/sroot))^(-1))+sum((corf(g).^
                    (0:c(g)-1))./factorial(0:c(g)-1)))^(-1);
        StBatProbm0=StBatProb00*((corf(g).^(1:c(g)-1))./factorial(1:c(g)-1));
        StBatProbcn=StBatProb00*((corf(g)^c(g))/factorial(c(g)))*((1/sroot).^(0:nnmax));
        utiladj(g)=StBatProb00*(sum(((1:c(g)-1).*(corf(g).^(1:c(g)-1))./(c(g).*factorial(1:c(g)-1)))+
                (((corf(g)^c(g))*sroot)/(factorial(c(g))*(sroot-1))));
        utiltool(g)=utiladj(g)/(1+(EB(g)/ESg(g)));
        util(g)=utiladj(g);

      end
    end
end

% Calculate Squared Coefficients of Variation (SCV) of the aggregate job arrival processes to each -
% toolgroup solving linear equations.
SCVa=zeros(G,1);
SCVe=zeros;
x=zeros;
v=zeros;
w=zeros;
b=zeros(G,1);
a=zeros;
for g=1:G
   SCVe(g)=0;         % SCV of the aggregate exogenous arrival process to tool group g

   x(g)=1+(max(SCVs(g),0.2)-1)/sqrt(c(g));
   v(g)=(sum(thita(:,g).^2))^(-1);
   w(g)=(1+4*(1-util(g)^2)*(v(g)-1))^(-1);
   b(:,g)=w(g)*(thita(1:G,g).*r(1:G,g))*(1-util(g)^2);
   a(g)=1+w(g)*((thita(G+1,g)*SCVe(g)-1)+sum(thita(1:G,g).*((1-r(1:G,g))+r(1:G,g).*((util(g)^2)*x(g)))));

   SCVa(g)=a(g)+sum(SCVa(:).*b(:,g));

   if bmax(g)==1 || bmax(g)<=AvWaf(g)
      corf(g)=(SCVs(g)+SCVa(g))/(SCVs(g)+1);    % correction factor
   end
end

%%
```

```
% Mean Queueing Delay and Average Cycle Time section

adjust=zeros;
Delay=zeros;
for g=1:G
   if bmax(g)==1
      if any(Lamdainc(:,g)<=Gamma(g)) && c(g)>=5
         Gg=Gamma(g);
         mi=1/EZ(g);
         cc=c(g);
         ro=Gg/(mi*cc);
         ProbD=(((cc*ro)^cc)/(factorial(cc)*(1-ro)))/(sum(((cc*ro).^(0:cc-1))./
                  factorial(0:cc-1))+(((cc*ro)^cc)/(factorial(cc)*(1-ro))));
         DelayMMc=ProbD/((cc*mi)-Gg);
         Delay(g)=DelayMMc*((SCVz(g)+SCVa(g))/2);
      else
         adjust(g)=((utiltool(g)+utilinc(g))^(sqrt(c(g)-1)))/(c(g)^2);
         Delay(g)=adjust(g)*((LbES2b(g)+corf(g)*sum(Lamda2(1,T(:,g)).*ES2(T(:,g))))/
                     (2*(1-utilinc(g))*(1-utilinc(g)-utiltool(g))));
      end
   elseif bmax(g)>1
      if bmax(g)<=AvWaf(g)
         if any(Lamdainc(:,g)<=Gamma(g)) && c(g)>=5
            Gg=Gamma(g);
            mi=1/EZ(g);
            cc=c(g);
            ro=Gg/(mi*cc);
            ProbD=(((cc*ro)^cc)/(factorial(cc)*(1-ro)))/(sum(((cc*ro).^(0:cc-1))./
                     factorial(0:cc-1))+(((cc*ro)^cc)/(factorial(cc)*(1-ro))));
            DelayMMc=ProbD/((cc*mi)-Gg);
            Delay(g)=DelayMMc*((SCVz(g)+SCVa(g))/2);
         else
            adjust(g)=((utiltool(g)+utilinc(g))^(sqrt(c(g)-1)))/(c(g)^2);
            Delay(g)=adjust(g)*((LbES2b(g)+corf(g)*sum(Lamda2(1,T(:,g)).*EY2(T(:,g))))/
                        (2*(1-utilinc(g))*(1-utilinc(g)-utiltool(g))));
         end
      elseif bmax(g)>AvWaf(g)
         Delay(g)=((StBatProbcn(1)*sroot)/(Gamma(g)*((sroot-1)^2)))*((SCVa(g)+SCVz(g))/2);
      end
   end
end

CTnom=zeros;
CTrw=zeros;
CT=zeros(1,F);
for f=1:F
   CTnom(f)=sum(Delay(t(s(f):s(f)+Nf(f)-1))+ES(s(f):s(f)+Nf(f)-1));
   CTrw(s(f):s(f)+Nf(f)-1)=Delay(t(rew(s(f)):rew(s(f))+Nf(f)-1))+ES(rew(s(f)):rew(s(f))+Nf(f)-1);
   for k=s(f):s(f)+Nf(f)-1
      CT(f)=CT(f)+sum((Prwnes(k).^(1:mmax(k))).*CTrw(k));
   end
   CT(f)=CT(f)+CTnom(f);
end
```

```
xlswrite('DATA.xlsx',[util,Delay,CT/60],'Results',strcat('B',num2str(Case+1)));
end

%%-------------------------------------------------------------------------------------------------------------------------------------------

% Function for the calculation of the first moments of the uniform distribution

function [moments]= moments(EX,e)

lb=EX-e*EX;
ub=EX+e*EX;
VX=((ub-lb)^2)/12;
EX2=VX+(EX)^2;
moments=[EX,VX,EX2];

end
```

Appendix B: Input Data

Table B: Input values for the different scenarios (p_s, p_r: probabilities of scrapping and rework, respectively, $e1, e2$: percentages of the means that determine the intervals, ST: Setup Time in min, FS: Frequency of Setups in arrivals per shift, DM: Duration of the unscheduled Maintenance in min, FM: Frequency of unscheduled Maintenances in arrivals per week)

In our analysis we modified the frequency of setups and unscheduled maintenances in order to specify the arrivals in minutes, for example:

- FS $= 1\ arrival\ per\ shift\ \equiv \frac{1}{12*60} = 0.001388889$/min,

  $= 2\ arrivals\ per\ shift \equiv \frac{2}{12*60} = 0.002777778$/min, etc.
- FM $= 4\ arrivals\ per\ week \equiv \frac{4}{7*24*60} = 0.000396825$/min, etc.

Scenarios	p_s	p_r	$e1$	$e2$	ST (min)	FS (arrivals per shift)	DM (min)	FM (arrivals per week)
1	0	0	0.001	0.001	0	0	420	4
2	0	0.1	0.001	0.001	0	0	420	4
3	0	0.2	0.001	0.001	0	0	420	4
4	0.1	0	0.001	0.001	0	0	420	4
5	0.1	0.1	0.001	0.001	0	0	420	4
6	0.1	0.2	0.001	0.001	0	0	420	4
7	0.2	0	0.001	0.001	0	0	420	4
8	0.2	0.1	0.001	0.001	0	0	420	4
9	0.2	0.2	0.001	0.001	0	0	420	4
10	0	0	**0.01**	**0.01**	0	0	420	4
11	0	0.1	0.01	0.01	0	0	420	4
12	0	0.2	0.01	0.01	0	0	420	4
13	0.1	0	0.01	0.01	0	0	420	4
14	0.1	0.1	0.01	0.01	0	0	420	4
15	0.1	0.2	0.01	0.01	0	0	420	4
16	0.2	0	0.01	0.01	0	0	420	4
17	0.2	0.1	0.01	0.01	0	0	420	4
18	0.2	0.2	0.01	0.01	0	0	420	4
19	0	0	**0.1**	**0.1**	0	0	420	4
20	0	0.1	0.1	0.1	0	0	420	4
21	0	0.2	0.1	0.1	0	0	420	4
22	0.1	0	0.1	0.1	0	0	420	4

23	0.1	0.1	0.1	0.1	0	0	420	4
24	0.1	0.2	0.1	0.1	0	0	420	4
25	0.2	0	0.1	0.1	0	0	420	4
26	0.2	0.1	0.1	0.1	0	0	420	4
27	0.2	0.2	0.1	0.1	0	0	420	4
28	0	0	**0.001**	**0.001**	**10**	**1**	420	4
29	0	0.1	0.001	0.001	10	1	420	4
30	0	0.2	0.001	0.001	10	1	420	4
31	0.1	0	0.001	0.001	10	1	420	4
32	0.1	0.1	0.001	0.001	10	1	420	4
33	0.1	0.2	0.001	0.001	10	1	420	4
34	0.2	0	0.001	0.001	10	1	420	4
35	0.2	0.1	0.001	0.001	10	1	420	4
36	0.2	0.2	0.001	0.001	10	1	420	4
37	0	0	**0.01**	**0.01**	10	1	420	4
38	0	0.1	0.01	0.01	10	1	420	4
39	0	0.2	0.01	0.01	10	1	420	4
40	0.1	0	0.01	0.01	10	1	420	4
41	0.1	0.1	0.01	0.01	10	1	420	4
42	0.1	0.2	0.01	0.01	10	1	420	4
43	0.2	0	0.01	0.01	10	1	420	4
44	0.2	0.1	0.01	0.01	10	1	420	4
45	0.2	0.2	0.01	0.01	10	1	420	4
46	0	0	**0.1**	**0.1**	10	1	420	4
47	0	0.1	0.1	0.1	10	1	420	4
48	0	0.2	0.1	0.1	10	1	420	4
49	0.1	0	0.1	0.1	10	1	420	4
50	0.1	0.1	0.1	0.1	10	1	420	4
51	0.1	0.2	0.1	0.1	10	1	420	4
52	0.2	0	0.1	0.1	10	1	420	4
53	0.2	0.1	0.1	0.1	10	1	420	4
54	0.2	0.2	0.1	0.1	10	1	420	4
55	0	0	**0.001**	**0.001**	10	**2**	420	4
56	0	0.1	0.001	0.001	10	2	420	4
57A	0	0.15	0.001	0.001	10	2	420	4
57	0	0.2	0.001	0.001	10	2	420	4
58	0.1	0	0.001	0.001	10	2	420	4
59	0.1	0.1	0.001	0.001	10	2	420	4
60	0.1	0.2	0.001	0.001	10	2	420	4
61	0.2	0	0.001	0.001	10	2	420	4
62	0.2	0.1	0.001	0.001	10	2	420	4
63	0.2	0.2	0.001	0.001	10	2	420	4
64	0	0	**0.01**	**0.01**	10	2	420	4
65	0	0.1	0.01	0.01	10	2	420	4
66A	0	0.15	0.01	0.01	10	2	420	4
66	0	0.2	0.01	0.01	10	2	420	4

67	0.1	0	0.01	0.01	10	2	420	4
68	0.1	0.1	0.01	0.01	10	2	420	4
69	0.1	0.2	0.01	0.01	10	2	420	4
70	0.2	0	0.01	0.01	10	2	420	4
71	0.2	0.1	0.01	0.01	10	2	420	4
72	0.2	0.2	0.01	0.01	10	2	420	4
73	0	0	**0.1**	**0.1**	10	2	420	4
74	0	0.1	0.1	0.1	10	2	420	4
75A	0	0.15	0.1	0.1	10	2	420	4
75	0	0.2	0.1	0.1	10	2	420	4
76	0.1	0	0.1	0.1	10	2	420	4
77	0.1	0.1	0.1	0.1	10	2	420	4
78	0.1	0.2	0.1	0.1	10	2	420	4
79	0.2	0	0.1	0.1	10	2	420	4
80	0.2	0.1	0.1	0.1	10	2	420	4
81	0.2	0.2	0.1	0.1	10	2	420	4
82	0	0	**0.001**	**0.001**	10	**3**	420	4
83	0	0.1	0.001	0.001	10	3	420	4
84A	0	0.15	0.001	0.001	10	3	420	4
84	0	0.2	0.001	0.001	10	3	420	4
85	0.1	0	0.001	0.001	10	3	420	4
86	0.1	0.1	0.001	0.001	10	3	420	4
87	0.1	0.2	0.001	0.001	10	3	420	4
88	0.2	0	0.001	0.001	10	3	420	4
89	0.2	0.1	0.001	0.001	10	3	420	4
90	0.2	0.2	0.001	0.001	10	3	420	4
91	0	0	**0.01**	**0.01**	10	3	420	4
92	0	0.1	0.01	0.01	10	3	420	4
93A	0	0.15	0.01	0.01	10	3	420	4
93	0	0.2	0.01	0.01	10	3	420	4
94	0.1	0	0.01	0.01	10	3	420	4
95	0.1	0.1	0.01	0.01	10	3	420	4
96	0.1	0.2	0.01	0.01	10	3	420	4
97	0.2	0	0.01	0.01	10	3	420	4
98	0.2	0.1	0.01	0.01	10	3	420	4
99	0.2	0.2	0.01	0.01	10	3	420	4
100	0	0	**0.1**	**0.1**	10	3	420	4
101	0	0.1	0.1	0.1	10	3	420	4
102A	0	0.15	0.1	0.1	10	3	420	4
102	0	0.2	0.1	0.1	10	3	420	4
103	0.1	0	0.1	0.1	10	3	420	4
104	0.1	0.1	0.1	0.1	10	3	420	4
105	0.1	0.2	0.1	0.1	10	3	420	4
106	0.2	0	0.1	0.1	10	3	420	4
107	0.2	0.1	0.1	0.1	10	3	420	4
108	0.2	0.2	0.1	0.1	10	3	420	4

109	0	0	**0.001**	**0.001**	**0**	**0**	**210**	**8**
110	0	0.1	0.001	0.001	0	0	210	8
111	0	0.2	0.001	0.001	0	0	210	8
112	0.1	0	0.001	0.001	0	0	210	8
113	0.1	0.1	0.001	0.001	0	0	210	8
114	0.1	0.2	0.001	0.001	0	0	210	8
115	0.2	0	0.001	0.001	0	0	210	8
116	0.2	0.1	0.001	0.001	0	0	210	8
117	0.2	0.2	0.001	0.001	0	0	210	8
118	0	0	**0.01**	**0.01**	0	0	210	8
119	0	0.1	0.01	0.01	0	0	210	8
120	0	0.2	0.01	0.01	0	0	210	8
121	0.1	0	0.01	0.01	0	0	210	8
122	0.1	0.1	0.01	0.01	0	0	210	8
123	0.1	0.2	0.01	0.01	0	0	210	8
124	0.2	0	0.01	0.01	0	0	210	8
125	0.2	0.1	0.01	0.01	0	0	210	8
126	0.2	0.2	0.01	0.01	0	0	210	8
127	0	0	**0.1**	**0.1**	0	0	210	8
128	0	0.1	0.1	0.1	0	0	210	8
129	0	0.2	0.1	0.1	0	0	210	8
130	0.1	0	0.1	0.1	0	0	210	8
131	0.1	0.1	0.1	0.1	0	0	210	8
132	0.1	0.2	0.1	0.1	0	0	210	8
133	0.2	0	0.1	0.1	0	0	210	8
134	0.2	0.1	0.1	0.1	0	0	210	8
135	0.2	0.2	0.1	0.1	0	0	210	8
136	0	0	**0.001**	**0.001**	**10**	**1**	210	8
137	0	0.1	0.001	0.001	10	1	210	8
138	0	0.2	0.001	0.001	10	1	210	8
139	0.1	0	0.001	0.001	10	1	210	8
140	0.1	0.1	0.001	0.001	10	1	210	8
141	0.1	0.2	0.001	0.001	10	1	210	8
142	0.2	0	0.001	0.001	10	1	210	8
143	0.2	0.1	0.001	0.001	10	1	210	8
144	0.2	0.2	0.001	0.001	10	1	210	8
145	0	0	**0.01**	**0.01**	10	1	210	8
146	0	0.1	0.01	0.01	10	1	210	8
147	0	0.2	0.01	0.01	10	1	210	8
148	0.1	0	0.01	0.01	10	1	210	8
149	0.1	0.1	0.01	0.01	10	1	210	8
150	0.1	0.2	0.01	0.01	10	1	210	8
151	0.2	0	0.01	0.01	10	1	210	8
152	0.2	0.1	0.01	0.01	10	1	210	8
153	0.2	0.2	0.01	0.01	10	1	210	8
154	0	0	**0.1**	**0.1**	10	1	210	8

155	0	0.1	0.1	0.1	10	1	210	8
156	0	0.2	0.1	0.1	10	1	210	8
157	0.1	0	0.1	0.1	10	1	210	8
158	0.1	0.1	0.1	0.1	10	1	210	8
159	0.1	0.2	0.1	0.1	10	1	210	8
160	0.2	0	0.1	0.1	10	1	210	8
161	0.2	0.1	0.1	0.1	10	1	210	8
162	0.2	0.2	0.1	0.1	10	1	210	8
163	0	0	**0.001**	**0.001**	10	**2**	210	8
164	0	0.1	0.001	0.001	10	2	210	8
165A	0	0.15	0.001	0.001	10	2	210	8
165	0	0.2	0.001	0.001	10	2	210	8
166	0.1	0	0.001	0.001	10	2	210	8
167	0.1	0.1	0.001	0.001	10	2	210	8
168	0.1	0.2	0.001	0.001	10	2	210	8
169	0.2	0	0.001	0.001	10	2	210	8
170	0.2	0.1	0.001	0.001	10	2	210	8
171	0.2	0.2	0.001	0.001	10	2	210	8
172	0	0	**0.01**	**0.01**	10	2	210	8
173	0	0.1	0.01	0.01	10	2	210	8
174A	0	0.15	0.01	0.01	10	2	210	8
174	0	0.2	0.01	0.01	10	2	210	8
175	0.1	0	0.01	0.01	10	2	210	8
176	0.1	0.1	0.01	0.01	10	2	210	8
177	0.1	0.2	0.01	0.01	10	2	210	8
178	0.2	0	0.01	0.01	10	2	210	8
179	0.2	0.1	0.01	0.01	10	2	210	8
180	0.2	0.2	0.01	0.01	10	2	210	8
181	0	0	**0.1**	**0.1**	10	2	210	8
182	0	0.1	0.1	0.1	10	2	210	8
183A	0	0.15	0.1	0.1	10	2	210	8
183	0	0.2	0.1	0.1	10	2	210	8
184	0.1	0	0.1	0.1	10	2	210	8
185	0.1	0.1	0.1	0.1	10	2	210	8
186	0.1	0.2	0.1	0.1	10	2	210	8
187	0.2	0	0.1	0.1	10	2	210	8
188	0.2	0.1	0.1	0.1	10	2	210	8
189	0.2	0.2	0.1	0.1	10	2	210	8
190	0	0	**0.001**	**0.001**	10	**3**	210	8
191	0	0.1	0.001	0.001	10	3	210	8
192A	0	0.15	0.001	0.001	10	3	210	8
192	0	0.2	0.001	0.001	10	3	210	8
193	0.1	0	0.001	0.001	10	3	210	8
194	0.1	0.1	0.001	0.001	10	3	210	8
195	0.1	0.2	0.001	0.001	10	3	210	8
196	0.2	0	0.001	0.001	10	3	210	8

197	0.2	0.1	0.001	0.001	10	3	210	8
198	0.2	0.2	0.001	0.001	10	3	210	8
199	0	0	**0.01**	**0.01**	10	3	210	8
200	0	0.1	0.01	0.01	10	3	210	8
201A	0	0.15	0.01	0.01	10	3	210	8
201	0	0.2	0.01	0.01	10	3	210	8
202	0.1	0	0.01	0.01	10	3	210	8
203	0.1	0.1	0.01	0.01	10	3	210	8
204	0.1	0.2	0.01	0.01	10	3	210	8
205	0.2	0	0.01	0.01	10	3	210	8
206	0.2	0.1	0.01	0.01	10	3	210	8
207	0.2	0.2	0.01	0.01	10	3	210	8
208	0	0	**0.1**	**0.1**	10	3	210	8
209	0	0.1	0.1	0.1	10	3	210	8
210A	0	0.15	0.1	0.1	10	3	210	8
210	0	0.2	0.1	0.1	10	3	210	8
211	0.1	0	0.1	0.1	10	3	210	8
212	0.1	0.1	0.1	0.1	10	3	210	8
213	0.1	0.2	0.1	0.1	10	3	210	8
214	0.2	0	0.1	0.1	10	3	210	8
215	0.2	0.1	0.1	0.1	10	3	210	8
216	0.2	0.2	0.1	0.1	10	3	210	8

Appendix C: Results for the Different Scenarios

Table C: Mean utilization (Util) and mean queueing delay (Delay) (in min) at each of the machines groups 1,2,3, mean cycle time (CT) (in hours) for each of the product families A,B,Test Wafers for all the different scenarios.

Scenarios	Util 1	Util 2	Util 3	Delay 1 (min)	Delay 2 (min)	Delay 3 (min)	CT A (hrs)	CT B (hrs)	CT Test (hrs)
1	0.9793	0.8333	0.9167	153.10767	59.79154	276.5681	30.39891	30.39891	30.39891
2	0.982	0.8444	0.9333	167.80701	65.75416	343.66908	36.65678	36.65678	36.65678
3	0.9887	0.8753	0.9797	236.39666	87.83824	1069.60711	72.65368	72.65368	72.65368
4	0.9735	0.8099	0.8703	128.24485	48.99405	178.26009	25.9333	25.9333	25.9333
5	0.9762	0.8204	0.8857	137.07690	53.22373	203.2899	29.92332	29.92332	29.92332
6	0.9833	0.8495	0.9288	172.83246	68.06622	325.66649	39.56261	39.56261	39.56261
7	0.9561	0.7484	0.752	91.71345	30.39625	87.21201	21.06072	21.06072	21.06072
8	0.9591	0.7572	0.7645	94.8774	32.3158	93.01378	23.56592	23.56592	23.56592
9	0.967	0.7817	0.7994	105.60747	38.50045	112.71343	27.17285	27.17285	27.17285
10	0.9793	0.8333	0.9167	153.10796	59.79555	276.57443	30.39927	30.39927	30.39927
11	0.982	0.8444	0.9333	167.80734	65.75853	343.67713	36.65724	36.65724	36.65724
12	0.9887	0.8753	0.9797	236.39715	87.84397	1069.63483	72.65504	72.65504	72.65504
13	0.9735	0.8099	0.8703	128.24509	48.99743	178.26399	25.93355	25.93355	25.93355
14	0.9762	0.8204	0.8857	137.07717	53.22737	203.2944	29.92363	29.92363	29.92363
15	0.9833	0.8495	0.9288	172.83281	68.07077	325.67407	39.56311	39.56311	39.56311
16	0.9561	0.7484	0.752	91.71362	30.39854	87.21383	21.06087	21.06087	21.06087
17	0.9591	0.7572	0.7645	94.87755	32.31821	93.01572	23.56609	23.56609	23.56609
18	0.967	0.7817	0.7994	105.60768	38.50325	112.71581	27.17307	27.17307	27.17307
19	0.9793	0.8333	0.9167	153.13739	60.19659	277.20717	30.43471	30.43471	30.43471
20	0.982	0.8444	0.9333	167.84020	66.19589	344.48197	36.704	36.704	36.704
21	0.9887	0.8753	0.9797	236.44603	88.41711	1072.40711	72.79081	72.79081	72.79081
22	0.9735	0.8099	0.8703	128.26939	49.33552	178.654	25.95863	25.95863	25.95863
23	0.9762	0.8204	0.8857	137.10361	53.59131	203.74502	29.95446	29.95446	29.95446
24	0.9833	0.8495	0.9288	172.86791	68.52606	326.43275	39.61307	39.61307	39.61307
25	0.9561	0.7484	0.752	91.73062	30.62771	87.39510	21.07511	21.07511	21.07511
26	0.9591	0.7572	0.7645	94.8954	32.55938	93.20985	23.58270	23.58270	23.58270
27	0.9667	0.7817	0.7994	105.62843	38.78309	112.95425	27.19463	27.19463	27.19463
28	0.9793	0.8333	0.9306	153.10767	59.79154	330.24682	32.18820	32.18820	32.18820
29	0.982	0.8444	0.9472	167.80701	65.75416	429.98355	39.82164	39.82164	39.82164
30	0.9887	0.8753	0.9936	236.39666	87.83824	3259.88817	160.2649	160.2649	160.2649
31	0.9735	0.8099	0.8842	128.24485	48.99405	200.15214	26.66303	26.66303	26.66303
32	0.9762	0.8204	0.8996	137.07690	53.22373	231.57305	30.96037	30.96037	30.96037
33	0.9833	0.8495	0.9427	172.83246	68.06622	401.41536	42.59256	42.59256	42.59256
34	0.9561	0.7484	0.7659	91.71345	30.39625	93.27461	21.26281	21.26281	21.26281
35	0.9591	0.7572	0.7784	94.87738	32.3158	99.7543	23.81307	23.81307	23.81307
36	0.967	0.7817	0.8133	105.60747	38.50045	122.04375	27.54607	27.54607	27.54607

37	0.9793	0.8333	0.9306	153.10796	59.79555	330.25456	32.18860	32.18860	32.18860
38	0.982	0.8444	0.9472	167.80734	65.75853	429.99391	39.82219	39.82219	39.82219
39	0.9887	0.8753	0.9936	236.39715	87.84397	3259.97731	160.2687	160.2687	160.2687
40	0.9735	0.8099	0.8842	128.24509	48.99743	200.1566	26.66330	26.66330	26.66330
41	0.9762	0.8204	0.8996	137.07717	53.22737	231.57827	30.96070	30.96070	30.96070
42	0.9833	0.8495	0.9427	172.83281	68.07077	401.42497	42.59314	42.59314	42.59314
43	0.9561	0.7484	0.7659	91.71362	30.39854	93.27657	21.26296	21.26296	21.26296
44	0.9591	0.7572	0.7784	94.87755	32.31821	99.7564	23.81325	23.81325	23.81325
45	0.967	0.7817	0.8133	105.60768	38.50325	122.04636	27.54629	27.54629	27.54629
46	0.9793	0.8333	0.9306	153.13738	60.19659	331.02851	32.22875	32.22875	32.22875
47	0.982	0.8444	0.9472	167.84020	66.19589	431.02992	39.87742	39.87742	39.87742
48	0.9887	0.8753	0.9936	236.44603	88.41711	3268.89062	160.6502	160.6502	160.6502
49	0.9735	0.8099	0.8842	128.26939	49.33552	200.60189	26.69023	26.69023	26.69023
50	0.9762	0.8204	0.8996	137.10361	53.59131	232.1012	30.99419	30.99419	30.99419
51	0.9833	0.8495	0.9427	172.86791	68.52606	402.38561	42.65118	42.65118	42.65118
52	0.9561	0.7484	0.7659	91.73062	30.62771	93.47246	21.27769	21.27769	21.27769
53	0.9591	0.7572	0.7784	94.8954	32.55938	99.96682	23.83046	23.83046	23.83046
54	0.967	0.7817	0.8133	105.62843	38.78309	122.3076	27.56876	27.56876	27.56876
55	0.9793	0.8333	0.9444	153.10767	59.79154	409.36806	34.82558	34.82558	34.82558
56	0.982	0.8444	0.9611	167.80701	65.75416	575.15732	45.14468	45.14468	45.14468
57A	0.9849	0.8576	0.9809	190.61134	74.03781	1133.42143	69.78854	69.78854	69.78854
57	0.9887	0.8753	1.0074	236.39666	87.83824	-2683.09244	-77.4543	-77.4543	-77.4543
58	0.9735	0.8099	0.8981	128.24485	48.99405	227.66935	27.58027	27.58027	27.58027
59	0.9762	0.8204	0.9135	137.07690	53.22373	268.41829	32.31136	32.31136	32.31136
60	0.9833	0.8495	0.9566	172.83246	68.06622	523.38155	47.47121	47.47121	47.47121
61	0.9561	0.7484	0.7797	91.71345	30.39625	100.08692	21.48989	21.48989	21.48989
62	0.9591	0.7572	0.7923	94.87738	32.3158	107.37349	24.09244	24.09244	24.09244
63	0.967	0.7817	0.8272	105.60747	38.50045	132.81497	27.97692	27.97692	27.97692
64	0.9793	0.8333	0.9444	153.10796	59.79555	409.37793	34.82605	34.82605	34.82605
65	0.982	0.8444	0.9611	167.80734	65.75853	575.17165	45.14538	45.14538	45.14538
66A	0.9849	0.8576	0.9809	190.6117	74.04269	1133.451	69.78988	69.78988	69.78988
66	0.9887	0.8753	1.0074	236.39715	87.84397	-2683.17109	-77.4572	-77.4572	-77.4572
67	0.9735	0.8099	0.8981	128.24509	48.99743	227.6745	27.58057	27.58057	27.58057
68	0.9762	0.8204	0.9135	137.07717	53.22737	268.42448	32.31173	32.31173	32.31173
69	0.9833	0.8495	0.9566	172.83281	68.07077	523.39448	47.47192	47.47192	47.47192
70	0.9561	0.7484	0.7797	91.71362	30.39854	100.08904	21.49004	21.49004	21.49004
71	0.9591	0.7572	0.7923	94.87755	32.31821	107.37578	24.09262	24.09262	24.09262
72	0.967	0.7817	0.8272	105.60768	38.50325	132.81784	27.97715	27.97715	27.97715
73	0.9793	0.8333	0.9444	153.13738	60.19659	410.36423	34.87327	34.87327	34.87327
74	0.982	0.8444	0.9611	167.84020	66.19589	576.60457	45.21516	45.21516	45.21516
75A	0.9849	0.8576	0.9809	190.6499	74.53074	1136.428	69.92418	69.92418	69.92418
75	0.9887	0.8753	1.0074	236.44603	88.41711	-2691.03533	-77.7469	-77.7469	-77.7469
76	0.9735	0.8099	0.8981	128.26939	49.33552	228.19043	27.60984	27.60984	27.60984
77	0.9762	0.8204	0.9135	137.10361	53.59131	269.04327	32.34873	32.34873	32.34873
78	0.9833	0.8495	0.9566	172.86791	68.52606	524.68702	47.54324	47.54324	47.54324
79	0.9561	0.7484	0.7797	91.73062	30.62771	100.30151	21.50533	21.50533	21.50533

80	0.9591	0.7572	0.7923	94.8954	32.55938	107.60481	24.11052	24.11052	24.11052
81	0.9667	0.7817	0.8272	105.62843	38.78309	133.10572	28.00069	28.00069	28.00069
82	0.9793	0.8333	0.9583	153.10767	59.79154	538.81836	39.14059	39.14059	39.14059
83	0.982	0.8444	0.975	167.80701	65.75416	875.75566	56.16662	56.16662	56.16662
84A	0.9849	0.8576	0.9948	190.6113	74.03781	3990.651	179.3157	179.3157	179.3157
84	0.9887	0.8753	1.0213	236.39666	87.83824	-867.86865	-4.84535	-4.84535	-4.84535
85	0.9735	0.8099	0.9119	128.24485	48.99405	263.3708	28.77032	28.77032	28.77032
86	0.9762	0.8204	0.9274	137.07690	53.22373	318.57199	34.15033	34.15033	34.15033
87	0.9833	0.8495	0.9705	172.83246	68.06622	755.73778	56.76546	56.76546	56.76546
88	0.9561	0.7484	0.7936	91.71345	30.39625	107.79369	21.74678	21.74678	21.74678
89	0.9591	0.7572	0.8062	94.87738	32.3158	116.05174	24.41065	24.41065	24.41065
90	0.967	0.7817	0.8411	105.60747	38.50045	145.38734	28.47981	28.47981	28.47981
91	0.9793	0.8333	0.9583	153.10796	59.79555	538.83177	39.14118	39.14118	39.14118
92	0.982	0.8444	0.975	167.80734	65.75853	875.77837	56.16762	56.16762	56.16762
93A	0.9849	0.8576	0.9948	190.6117	74.04269	3990.761	179.3201	179.3201	179.3201
93	0.9887	0.8753	1.0213	236.39715	87.84397	-867.89663	-4.84622	-4.84622	-4.84622
94	0.9735	0.8099	0.9119	128.24509	48.99743	263.37689	28.77065	28.77065	28.77065
95	0.9762	0.8204	0.9274	137.07717	53.22737	318.5795	34.15075	34.15075	34.15075
96	0.9833	0.8495	0.9705	172.83281	68.07077	755.75716	56.76643	56.76643	56.76643
97	0.9561	0.7484	0.7936	91.71362	30.39854	107.79601	21.74694	21.74694	21.74694
98	0.9591	0.7572	0.8062	94.87755	32.31821	116.05425	24.41083	24.41083	24.41083
99	0.967	0.7817	0.8411	105.60768	38.50325	145.39054	28.48006	28.48006	28.48006
100	0.9793	0.8333	0.9583	153.13738	60.19659	540.17277	39.20022	39.20022	39.20022
101	0.982	0.8444	0.975	167.84020	66.19589	878.04977	56.26815	56.26815	56.26815
102A	0.9849	0.8576	0.9948	190.6499	74.53074	4001.834	179.7647	179.7647	179.7647
102	0.9887	0.8753	1.0213	236.44603	88.41711	-870.69422	-4.93324	-4.93324	-4.93324
103	0.9735	0.8099	0.9119	128.26939	49.33552	263.98609	28.80303	28.80303	28.80303
104	0.9762	0.8204	0.9274	137.10361	53.59131	319.33127	34.19263	34.19263	34.19263
105	0.9833	0.8495	0.9705	172.86791	68.52606	757.6949	56.86355	56.86355	56.86355
106	0.9561	0.7484	0.7936	91.73062	30.62771	108.02742	21.76286	21.76286	21.76286
107	0.9591	0.7572	0.8062	94.8954	32.55938	116.30469	24.42951	24.42951	24.42951
108	0.967	0.7817	0.8411	105.62843	38.78309	145.70989	28.50486	28.50486	28.50486
109	0.9793	0.8333	0.9167	153.10767	59.79153	276.5681	30.39891	30.39891	30.39891
110	0.982	0.8444	0.9333	167.80701	65.75415	343.66908	36.65678	36.65678	36.65678
111	0.9887	0.8753	0.9797	236.39666	87.83822	1069.60711	72.65368	72.65368	72.65368
112	0.9735	0.8099	0.8703	128.24485	48.99404	178.26009	25.9333	25.9333	25.9333
113	0.9762	0.8204	0.8857	137.07690	53.22372	203.28989	29.92332	29.92332	29.92332
114	0.9833	0.8495	0.9288	172.83246	68.06621	325.66649	39.56261	39.56261	39.56261
115	0.9561	0.7484	0.752	91.71345	30.39624	87.21201	21.06072	21.06072	21.06072
116	0.9591	0.7572	0.7645	94.87738	32.3158	93.01378	23.56592	23.56592	23.56592
117	0.967	0.7817	0.7994	105.60747	38.50044	112.71343	27.17285	27.17285	27.17285
118	0.9793	0.8333	0.9167	153.10796	59.79457	276.57443	30.39923	30.39923	30.39923
119	0.982	0.8444	0.9333	167.80734	65.75748	343.67713	36.65720	36.65720	36.65720
120	0.9887	0.8753	0.9797	236.39715	87.8426	1069.63483	72.65498	72.65498	72.65498
121	0.9735	0.8099	0.8703	128.24509	48.9966	178.26399	25.93352	25.93352	25.93352
122	0.9762	0.8204	0.8857	137.07717	53.22648	203.2944	29.9236	29.9236	29.9236

123	0.9833	0.8495	0.9288	172.83281	68.06967	325.67407	39.56306	39.56306	39.56306
124	0.9561	0.7484	0.752	91.71362	30.39796	87.21383	21.06085	21.06085	21.06085
125	0.9591	0.7572	0.7645	94.87755	32.31761	93.01572	23.56607	23.56607	23.56607
126	0.967	0.7817	0.7994	105.60768	38.50255	112.71581	27.17304	27.17304	27.17304
127	0.9793	0.8333	0.9167	153.13738	60.09937	277.20717	30.43146	30.43146	30.43146
128	0.982	0.8444	0.9333	167.84020	66.09036	344.48197	36.70013	36.70013	36.70013
129	0.9887	0.8753	0.9797	236.44603	88.28058	1072.40711	72.78535	72.78535	72.78535
130	0.9735	0.8099	0.8703	128.26939	49.25266	178.654	25.95587	25.95587	25.95587
131	0.9762	0.8204	0.8857	137.10361	53.50251	203.74502	29.95121	29.95121	29.95121
132	0.9833	0.8495	0.9288	172.86791	68.41631	326.43275	39.60868	39.60868	39.60868
133	0.9561	0.7484	0.752	91.73062	30.56988	87.3951	21.07319	21.07319	21.07319
134	0.9591	0.7572	0.7645	94.8954	32.49875	93.20985	23.58048	23.58048	23.58048
135	0.967	0.7817	0.7994	105.62843	38.71347	112.95425	27.19185	27.19185	27.19185
136	0.9793	0.8333	0.9306	153.10767	59.79153	330.24682	32.1882	32.1882	32.1882
137	0.982	0.8444	0.9472	167.80701	65.75415	429.98355	39.82164	39.82164	39.82164
138	0.9887	0.8753	0.9936	236.39666	87.83822	3259.8882	160.2649	160.2649	160.2649
139	0.9735	0.8099	0.8842	128.24485	48.99404	200.15214	26.66303	26.66303	26.66303
140	0.9762	0.8204	0.8996	137.0769	53.22372	231.57305	30.96037	30.96037	30.96037
141	0.9833	0.8495	0.9427	172.83246	68.06621	401.41536	42.59256	42.59256	42.59256
142	0.9561	0.7484	0.7659	91.71345	30.39624	93.27461	21.26281	21.26281	21.26281
143	0.9591	0.7572	0.7784	94.87738	32.3158	99.7543	23.81307	23.81307	23.81307
144	0.967	0.7817	0.8133	105.60747	38.50044	122.04375	27.54607	27.54607	27.54607
145	0.9793	0.8333	0.9306	153.10796	59.79457	330.25456	32.18857	32.18857	32.18857
146	0.982	0.8444	0.9472	167.80734	65.75748	429.99391	39.82215	39.82215	39.82215
147	0.9887	0.8753	0.9936	236.39715	87.8426	3259.9773	160.2687	160.2687	160.2687
148	0.9735	0.8099	0.8842	128.24509	48.9966	200.1566	26.66328	26.66328	26.66328
149	0.9762	0.8204	0.8996	137.07717	53.22648	231.57827	30.96067	30.96067	30.96067
150	0.9833	0.8495	0.9427	172.83281	68.06967	401.42497	42.5931	42.5931	42.5931
151	0.9561	0.7484	0.7659	91.71362	30.39796	93.27657	21.26294	21.26294	21.26294
152	0.9591	0.7572	0.7784	94.87755	32.31761	99.7564	23.81322	23.81322	23.81322
153	0.967	0.7817	0.8133	105.60768	38.50255	122.04636	27.54626	27.54626	27.54626
154	0.9793	0.8333	0.9306	153.13738	60.09937	331.02851	32.22551	32.22551	32.22551
155	0.982	0.8444	0.9472	167.8402	66.09036	431.02992	39.87355	39.87355	39.87355
156	0.9887	0.8753	0.9936	236.44603	88.28058	3268.8906	160.6447	160.6447	160.6447
157	0.9735	0.8099	0.8842	128.26939	49.25266	200.60189	26.68746	26.68746	26.68746
158	0.9762	0.8204	0.8996	137.10361	53.50251	232.1012	30.99093	30.99093	30.99093
159	0.9833	0.8495	0.9427	172.86791	68.41631	402.38561	42.64679	42.64679	42.64679
160	0.9561	0.7484	0.7659	91.73062	30.56988	93.47246	21.27576	21.27576	21.27576
161	0.9591	0.7572	0.7784	94.8954	32.49875	99.96682	23.82824	23.82824	23.82824
162	0.967	0.7817	0.8133	105.62843	38.71347	122.3076	27.56598	27.56598	27.56598
163	0.9793	0.8333	0.9444	153.10767	59.79153	409.36806	34.82558	34.82558	34.82558
164	0.982	0.8444	0.9611	167.80701	65.75415	575.15732	45.14468	45.14468	45.14468
165A	0.9849	0.8576	0.9809	190.6113	74.0378	1133.421	69.78854	69.78854	69.78854
165	0.9887	0.8753	1.0074	236.39666	87.83822	-2683.092	-77.4543	-77.4543	-77.4543
166	0.9735	0.8099	0.8981	128.24485	48.99404	227.66935	27.58027	27.58027	27.58027
167	0.9762	0.8204	0.9135	137.0769	53.22372	268.41829	32.31136	32.31136	32.31136

168	0.9833	0.8495	0.9566	172.83246	68.06621	523.38155	47.47121	47.47121	47.47121
169	0.9561	0.7484	0.7797	91.71345	30.39624	100.08692	21.48989	21.48989	21.48989
170	0.9591	0.7572	0.7923	94.87738	32.3158	107.37349	24.09244	24.09244	24.09244
171	0.967	0.7817	0.8272	105.60747	38.50044	132.81497	27.97692	27.97692	27.97692
172	0.9793	0.8333	0.9444	153.10796	59.79457	409.37793	34.82602	34.82602	34.82602
173	0.982	0.8444	0.9611	167.80734	65.75748	575.17165	45.14534	45.14534	45.14534
174A	0.9849	0.8576	0.9809	190.6117	74.04152	1133.451	69.78984	69.78984	69.78984
174	0.9887	0.8753	1.0074	236.39715	87.8426	-2683.171	-77.4573	-77.4573	-77.4573
175	0.9735	0.8099	0.8981	128.24509	48.9966	227.6745	27.58054	27.58054	27.58054
176	0.9762	0.8204	0.9135	137.07717	53.22648	268.42448	32.3117	32.3117	32.3117
177	0.9833	0.8495	0.9566	172.83281	68.06967	523.39448	47.47188	47.47188	47.47188
178	0.9561	0.7484	0.7797	91.71362	30.39796	100.08904	21.49002	21.49002	21.49002
179	0.9591	0.7572	0.7923	94.87755	32.31761	107.37578	24.0926	24.0926	24.0926
180	0.967	0.7817	0.8272	105.60768	38.50255	132.81784	27.97712	27.97712	27.97712
181	0.9793	0.8333	0.9444	153.13738	60.09937	410.36423	34.87003	34.87003	34.87003
182	0.982	0.8444	0.9611	167.8402	66.09036	576.60457	45.21129	45.21129	45.21129
183A	0.9849	0.8576	0.9809	190.6499	74.41362	1136.428	69.91969	69.91969	69.91969
183	0.9887	0.8753	1.0074	236.44603	88.28058	-2691.035	-77.7524	-77.7524	-77.7524
184	0.9735	0.8099	0.8981	128.26939	49.25266	228.19043	27.60708	27.60708	27.60708
185	0.9762	0.8204	0.9135	137.10361	53.50251	269.04327	32.34548	32.34548	32.34548
186	0.9833	0.8495	0.9566	172.86791	68.41631	524.68702	47.53885	47.53885	47.53885
187	0.9561	0.7484	0.7797	91.73062	30.56988	100.30151	21.5034	21.5034	21.5034
188	0.9591	0.7572	0.7923	94.8954	32.49875	107.60481	24.10829	24.10829	24.10829
189	0.967	0.7817	0.8272	105.62843	38.71347	133.10572	27.9979	27.9979	27.9979
190	0.9793	0.8333	0.9583	153.10767	59.79153	538.81836	39.14059	39.14059	39.14059
191	0.982	0.8444	0.975	167.80701	65.75415	875.75566	56.16662	56.16662	56.16662
192A	0.9849	0.8576	0.9948	190.6113	74.0378	3990.651	179.3157	179.3157	179.3157
192	0.9887	0.8753	1.0213	236.39666	87.83822	-867.8687	-4.84535	-4.84535	-4.84535
193	0.9735	0.8099	0.9119	128.24485	48.99404	263.3708	28.77032	28.77032	28.77032
194	0.9762	0.8204	0.9274	137.0769	53.22372	318.57199	34.15033	34.15033	34.15033
195	0.9833	0.8495	0.9705	172.83246	68.06621	755.73778	56.76546	56.76546	56.76546
196	0.9561	0.7484	0.7936	91.713452	30.39624	107.79369	21.74678	21.74678	21.74678
197	0.9591	0.7572	0.8062	94.87738	32.3158	116.05174	24.41065	24.41065	24.41065
198	0.967	0.7817	0.8411	105.60747	38.50044	145.38734	28.47981	28.47981	28.47981
199	0.9793	0.8333	0.9583	153.10796	59.79457	538.83177	39.14114	39.14114	39.14114
200	0.982	0.8444	0.975	167.80734	65.75748	875.77837	56.16758	56.16758	56.16758
201A	0.9849	0.8576	0.9948	190.6117	74.04152	3990.761	179.3201	179.3201	179.3201
201	0.9887	0.8753	1.0213	236.39715	87.8426	-867.8966	-4.84628	-4.84628	-4.84628
202	0.9735	0.8099	0.9119	128.24509	48.9966	263.37689	28.77062	28.77062	28.77062
203	0.9762	0.8204	0.9274	137.07717	53.22648	318.5795	34.15072	34.15072	34.15072
204	0.9833	0.8495	0.9705	172.83281	68.06967	755.75716	56.76639	56.76639	56.76639
205	0.9561	0.7484	0.7936	91.71362	30.39796	107.79601	21.74692	21.74692	21.74692
206	0.9591	0.7572	0.8062	94.87755	32.31761	116.05425	24.41081	24.41081	24.41081
207	0.967	0.7817	0.8411	105.60768	38.50255	145.39054	28.48003	28.48003	28.48003
208	0.9793	0.8333	0.9583	153.13738	60.09937	540.17277	39.19698	39.19698	39.19698
209	0.982	0.8444	0.975	167.8402	66.09036	878.04977	56.26428	56.26428	56.26428

210A	0.9849	0.8576	0.9948	190.6499	74.41362	4001.834	179.7603	179.7603	179.7603
210	0.9887	0.8753	1.0213	236.44603	88.28058	-870.6942	-4.9387	-4.9387	-4.9387
211	0.9735	0.8099	0.9119	128.26939	49.25266	263.98609	28.80027	28.80027	28.80027
212	0.9762	0.8204	0.9274	137.10361	53.50251	319.33127	34.18937	34.18937	34.18937
213	0.9833	0.8495	0.9705	172.86791	68.41631	757.6949	56.85916	56.85916	56.85916
214	0.9561	0.7484	0.7936	91.73062	30.56988	108.02742	21.76093	21.76093	21.76093
215	0.9591	0.7572	0.8062	94.8954	32.49875	116.30469	24.42729	24.42729	24.42729
216	0.967	0.7817	0.8411	105.62843	38.71347	145.70989	28.50207	28.50207	28.50207