Generating Control Policies for Timed Discrete-Event Systems through Efficient State Space Exploration

BY

FRANCESCA SCHULER
B.S., University of Illinois, Urbana-Champaign, 1995
M.S., University of Illinois, Chicago, 1997
M.B.A., DePaul University, Chicago, 2000
M.S., Illinois Institute of Technology, Chicago 2004

THESIS

Submitted as partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Industrial Engineering and Operations Research
in the Graduate College of the
University of Illinois at Chicago, 2016

Chicago, Illinois

Defense Committee:

Houshang Darabi, Chair and Advisor
Thomas Babin, Charter Dura-Bar
Ugo Buy, Computer Science
Julius Gyorfi, Motorola Mobility
David He, Mechanical and Industrial Engineering

ACKNOWLEDGEMENTS

I would like to thank my thesis committee – Dr. Thomas Babin, Dr. Ugo Buy, Dr. Julius Gyorfi, Dr. David He and my advisor Dr. Houshang Darabi for their support and assistance. They provided guidance in various areas that helped me accomplish my research goals.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS ii

LIST OF TABLES iv

LIST OF FIGURES v

SUMMARY vii

1. INTRODUCTION 1

2. LITERATURE REVIEW 11

   2.1 Petri net software tools 11

   2.2 State space and state space reduction 26

   2.3 Line balancing and buffer allocation optimization 27

   2.4 Queuing networks performance and blocking 30

   2.5 Flexible manufacturing 31

3. EXACT METHODS FOR DETERMINING STATE SPACE PARAMETERS 34

   3.1 Model 1: N server, N+1 sequential line model with unity capacity servers and no buffer capacity restriction 34

   3.2 Model II: N server, N+1 sequential line with reduced capacity for one buffer 43

4. OPTIMIZATION FRAMEWORK FOR BUFFER CLUSTERING POLICY 50

5. APPLYING MODELS TO INDUSTRY EXAMPLES 58

   5.1 Applying Chapter 3 results to distribution center example 58

   5.2 Applying Chapter 4 results to the manufacturing center 62

6. CONCLUSIONS AND FUTURE WORK 69

CITED LITERATURE 72

VITA 79

APPENDIX I: PETRI NET TOOLS ASSESSED 82

LIST OF TABLES

Table 1. Case study distribution center processes with process times 3

Table 2. Case study manufacturing production line process steps and process times.............. 3

Table 3. Two-way variable table for MB_4 62

Table 4. W_j buffer cluster sets and BB_j values for each buffer cluster set 63

Table 5. Buffer cluster sets and buffer storage savings 65

Table 6. W_j buffer cluster sets and BB_j values for each buffer cluster set ($T_2 = 3s$) 67

Table 7. Buffer cluster sets and buffer storage savings with T_2 at 2 and 3 seconds..................67

Table 8. Number of time steps for required buffer size computations..68

Table 9. Calculation and computation time savings varying K1 68

LIST OF FIGURES

Figure 1-1. Distribution center operational workflow 1

Figure 1-2. Manufacturing operational workflow 1

Figure 1-3. Distribution line 2

Figure 1-4. Manufacturing production line 2

Figure 1-5. Distribution center increasing cyclic pattern and stable cyclic pattern 7

Figure 1-6. N-Server, N+1 Buffer sequential line 7

Figure 1-7. Work cells with one or more serial stations in between 9

Figure 1-8. Work cells using buffer cluster concept with serial stations in between 9

Figure 2-1. Number of tools versus observed first and last year of release 21

Figure 2-2. Number of tools versus support type 22

Figure 2-3. Number of Petri net tools meeting multiple criteria 23

Figure 3-1. Frequency of arrivals to S_2 when $T_2 > T_1$ 37

Figure 3-2. Frequency of arrivals to S_2 when $T_1 > T_2$ 37

Figure 3-3. Frequency of arrivals to S_2 when $T_1 = T_2$ 37

Figure 3-4. Frequency of arrivals to S_i when $T_i > MT_{i-1}$ 38

Figure 3-5. Frequency of arrivals to S_i when $MT_{i-1} > T_i$ 39

Figure 3-6. Time defined for $SA_i(t)$ from 1 through $K1^{th}$ arrival 40

Figure 3-7. Maximum number of entities buffer B_i experiences given inventory K1 42

Figure 3-8. Buffer behavior when B_v capacity set to L_v 44

Figure 3-9. Rules for determining impacted arrivals and departures for B_i and S_i 45

Figure 3-10. Time defined for $BD_i(t)$ from 1 through $K1^{th}$ departure 47

Figure 4-1. Illustration of $K2_i$, $K3_i$, t1Bi, t2Bi and MB_i -1. 50

Figure 4-2. Non-sequential and sequential clusters maintaining operation sequence 52

Figure 4-3. Buffer profiles of B_k and B_p and time intervals 1 through 3 for p = k + 1 54

Figure 5-1. Shift completion vs. demand 60

Figure 5-2. MB_i vs. pre flash service time (T_3) 61

Figure 5-3. Serial production line 65

Figure 5-4. Production line with buffer clusters...65

SUMMARY

Production systems are event driven and require knowledge of timing of events across all elements of the system particularly when developing policies for production control. A suite of time-dependent models are derived for a class of discrete systems, in particular, an N-Server, N+1-Buffer sequential line. First, time-dependent models for buffers and servers are derived. From the time-dependent server and buffer arrival and departure models the maximum number of entities any buffer in the serial line will experience across all times is derived. One assumption is then relaxed, thereby reducing the capacity of a queue. The transition and block times for all queues and servers are derived and time-dependent models for server and queue arrivals and departures are developed. Lastly, a buffer cluster concept is proposed and a time based parametric model is derived that determines the sizing of the buffer cluster. A reduced time space for which to search for the buffer cluster sizing is derived and a model that determines an optimal buffer clustering policy is presented. Real world production examples are used to illustrate where the models are utilized and discuss several additional applications and benefits of the models.

1. INTRODUCTION

The motivation for this research stems from real world production lines of mobile devices including cellular phones and land mobile radios. The manufacturing facility and distribution center provides the critical data required for the research. The data consists of the operational workflows, process times, number of stations per process, and operator skill sets. The operational workflow of the distribution assembly line consists of nine process steps as shown in Figure 1-1. The operational workflow of the manufacturing production line consists of the seven process steps as shown in Figure 1-2. Production demand varies from shift to shift.

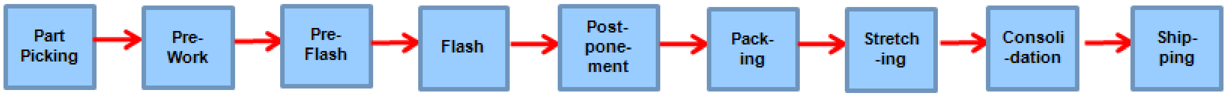


Figure 1-1. Distribution center operational workflow

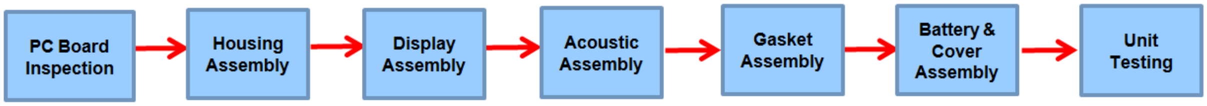


Figure 1-2. Manufacturing operational workflow

Other production lines are present, therefore space constraints exist. Buffer sizes cannot get too large that they infringe upon on other production line activities. In the sequential line, depending on the location of the incoming and outgoing process steps, the buffer in between could be a pallet or a box.

Figure 1-3 and Figure 1-4 show the serial lines the distribution center and manufacturing facility had in place. In these figures, the squares without a grid pattern represent server stations where a process step occurs. The serving stations may be manual, semi-automated or fully automated. The squares with a grid pattern represent buffers. Buffer B_1 holds the initial inventory that needs to be processed by the line. Buffers B_{10} (for distribution) and B_8 (for manufacturing) holds the final product inventory. A description of each process along with service times for the manufacturing and distribution lines respectively are available in Table 1 and Table 2.

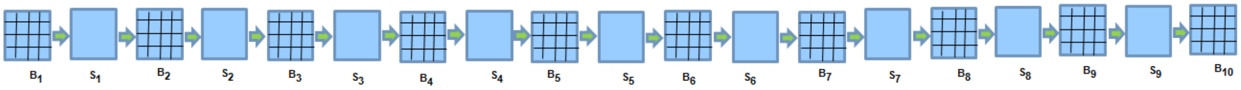


Figure 1-3. Distribution line

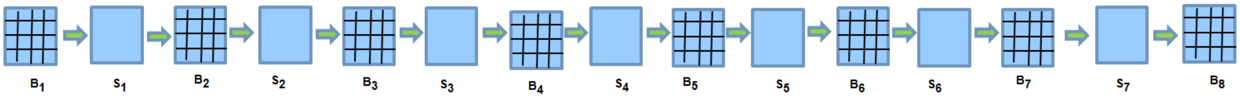


Figure 1-4. Manufacturing production line

Table 1. Case study distribution center processes with process times

Element	Name	Description	Process Time per Product Unit
S_1	Part Picking	Part Picking objective is to retrieve phones from Storage so that they can be preprocessed based on production demand.	40 seconds
S_2	Pre-Work	Pre-work objective is to perform appropriate operations to the phone prior to Pre-Flash.	240 seconds
S_3	Pre-Flash	Pre-flash objective is to update specific firmware on phone prior to Flash.	60 seconds
S_4	Flash	Flash process objective is to put secondary firmware on phone prior to Postponement.	520 seconds
S_5	Postponement	Postponement objective is to perform a customized process by which a generic or family product requires (e.g. carrier specific).	300 seconds
S_6	Packing	Packing objective is to pack product based on orders (in an over pack, with user guide, transformer, etc.).	160 seconds
S_7	Consolidation	Consolidation objective is to consolidate product based on orders, time frame and deliveries and occurs prior to Stretching.	640 seconds
S_8	Stretching	Stretching objective is to wrap product deliveries appropriately and occurs prior to Shipping.	240 seconds
S_9	Shipping	Shipping objective is to prepare product for delivery.	80 seconds

Table 2. Case study manufacturing production line process steps and process times

Element	Name	Description	Process Time Per Product Unit (sec)
S_1	PC Board Inspection	Inspects PC board	1
S_2	Housing Assembly	Assembles printed circuit board into pre-assembled housing	2
S_3	Display Assembly	Assembles display onto housing	4
S_4	Acoustic Component Assembly	Assembles acoustic components onto housing	5
S_5	Gasket Assembly	Assembles gaskets onto housing	14
S_6	Battery and Cover Assembly	Assembles battery and cover onto housing	10
S_7	Unit Testing	Tests production units	19

There are several problem areas the distribution center and manufacturing process teams desired to investigate as a part of this research. First of all, the distribution center staff identified key parameters to investigate: (1) the maximum buffer size allowed such that no buffer experiences blocking during a shift (2) the shift where a buffer's capacity should change to meet the production demand changes (3) the number of units processed by a bottleneck station when time of blocking occurs (4) the buffer transition and block time when a buffer size is reduced (5) the reaction time to route a resource (operator or workstation) to a bottleneck process to prevent impact to other processes (6) the time a failed machine must recover by as not to impact the production line or the time interval which to route a resource (operator or workstation) to a failed process (7) the maximum demand that the serial line can support given the limited shift time and buffer sizes (8) ability to determine production line behavior with varying process times.

The distribution center team was also interested in the exploration of the deployment of sensing and tracking technology to drive control policies as illustrated in (1) through (8) above. For example, a control policy to route a resource to a bottleneck process within the required reaction time to prevent impact to other processes. Or similarly, a policy is created to route a resource to a failed machine when the machine does not re-start within the required recovery time.

The last area of interest for the distribution center was directed toward leveraging flexible processes in their serial line. For example, the Postponement process was flexible in that it could occur after Pre-Flash or after the Flash process. A control policy was developed that assessed the buffer size at the Flash process and determined whether product should get routed to Postponement [1]. This area is discussed in detail as part of the Petri net survey in Chapter 2.

The manufacturing process team identified that their production lines could gain efficiencies such as increased throughput or reduced work in progress by utilizing specific configurations while maintaining the chronological order of operations. Some of the configurations, such as work cell or U-shaped production lines that have groups of buffers, often increase the space utilization. Therefore, the manufacturing center could not take advantage of the configuration efficiencies that a work cell or U-shaped production line provide. To solve this problem, the concept of a buffer cluster is introduced. A time based parametric model that determines the sizing of the buffer cluster is derived that also provides a reduced time space for which to search for the buffer cluster sizing. The model then determines an optimal buffer clustering policy that can be applied to any N-server, N+1 buffer sequential line configuration. This solution minimizes the buffer storage space utilized while ensuring no overflows or underflows occur in the buffer.

We explored the potential deployment of sensing and tracking technologies to enable control policies to address the key parameters (1) through (8) cited above which would require a model of the distribution center. Before creating the model, the appropriate modeling language and tool needs to be selected. Petri nets have been used to model discrete event systems and develop techniques for design, control and system measurement of discrete event systems. Petri nets are thought to be the ideal formalism to create the distribution center model. As there are several Petri net software tools available, the majority developed by university based academic teams and some by industry, a Petri net selection process is conducted. First, Petri net software selection criteria is developed in order to select an appropriate tool based on the goals of the distribution center team and expansion of research into new areas. The criteria for selecting the Petri net tool software are identified. Criteria included Petri Nets with Time, Performance Analysis, Token Attribute Setting (Colored Petri nets), Import/Export Capability, GUI: Editor and simulator, tool support and year of the last software tool release, operating system and scalability. After identifying the criteria, a detailed Petri net survey consisting of over fifty Petri net software tools is conducted that is discussed in detail along with detailed descriptions of the selection criteria in the literature review of Chapter 2.

As a result of the survey, the ability to scale the distribution center model to include the level of detail of interest including process times, operator skill sets, transportation times, and over five hundred batches of units per shift, required a Petri net tool to support scalability of 500+ tokens and 500+ Petri net elements (places and transitions). The tools in the market did not provide reliable results. Therefore, other discrete tools are assessed and a discrete event tool by Mathworks, named SimEvents is identified and a distribution center model is created. A control policy is created based on the flash buffer threshold as a trigger for re-routing product. The analysis requires over 150 simulations to identify the optimal control policy [1]. Although an optimal control policy is found, the system state space is not well understood as to why some control policies work better or worse than others. This leads us to the research at hand. The research put forward in the upcoming chapters provides methods for determining the state space parameters in the system which would enable us to address the distribution center and manufacturing facility areas of interest. For example, the maximum buffer capacity a buffer will experience, the numbers of arrivals and departures from a server or buffer at any given time, the time of transition or blocking of a buffer, the time of impact of a blocked buffer on other buffers and servers in the system, the time or size of a buffer or server to execute a control policy are all derived without running hundreds of simulations. This work was then leveraged to derive a model for the buffer cluster sizing and policy to enable alternate configurations of the sequential lines in the manufacturing center.

While working with the simulation model of the distribution center, cyclic patterns are observed for the buffers and servers, a sample which is shown in Figure 1-5. The observations of the cyclic patterns were utilized in Chapter 3 in deriving the exact methods to calculate key parameters (1) through (8) identified by the distribution center team. However, the distribution center process was abstracted such that it could expand or contract as new products are introduced. Thus, an N-Server, N+1 Buffer sequential line shown in Figure 1-6 is considered. Exact methods are developed that enable us to extract the parametric state space across every workstation and buffer at a given time identifying the key parameters (1) through (8) above without using simulation.

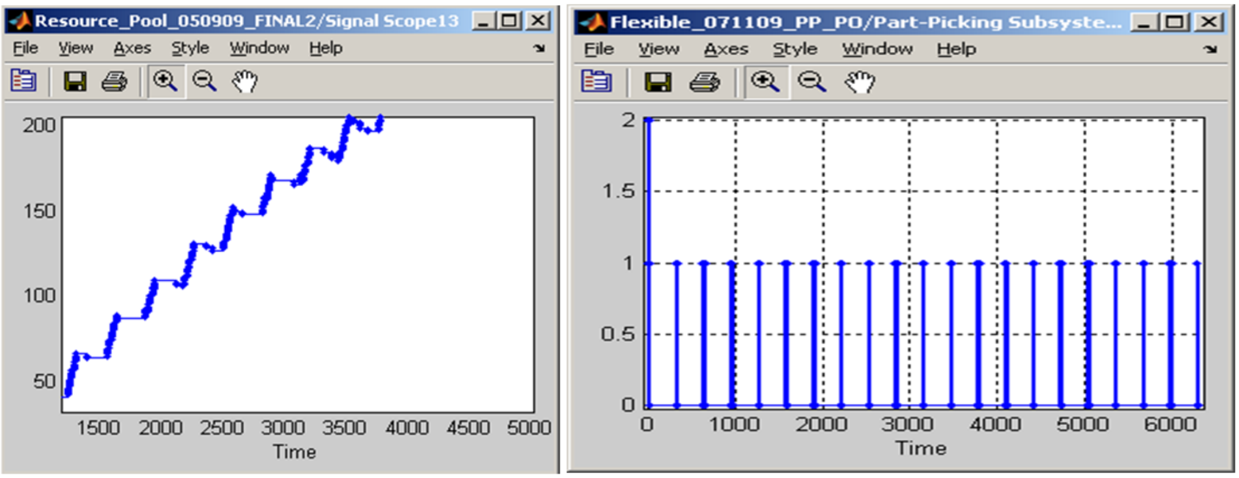


Figure 1-5. Distribution center increasing cyclic pattern and stable cyclic pattern

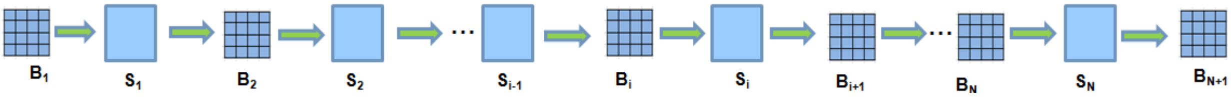


Figure 1-6. N-Server, N+1 Buffer sequential line

We first assume each server has a capacity of unity and each buffer has a capacity equal to or greater than the starting inventory. From this first scenario, we derive the number of arrivals and departures at a given time for each server and buffer. We also derive the number of entities in the buffer at any given time and the maximum capacity a buffer will experience such that a bottleneck does not occur.

Next, we select one buffer and determine the maximum capacity required and then reduce the level of capacity of that buffer to create a bottleneck. We derive the equations for determining the block time and transition times across all of the elements and determine again the server and buffer arrival and departure models at any given time. We also present a decision tree for determining which buffers and which servers are impacted by the reduced capacity buffer and a pointer to the models to use for each case.

In Chapter 4, we use the results of Chapter 3 (related to Figure 1-6) and apply them to a different model, a manufacturing work cell (related to Figure 1-7). In this scenario, because the buffers were sized separately (as dedicated buffers) with respect to the serial line and grouped in the center of the work cell, the grouped buffers were not leveraging available space in the neighboring buffers during the production shift. Therefore, the work cell exceeded the typical spacing between production lines due to the buffer storage space. A buffer cluster concept is proposed transitioning the grouped dedicated buffers in the center of the work cell to a single buffer cluster which enables increased buffer utilization and reduces the size of the grouped dedicated buffers. This allows the facility to benefit from efficiencies (e.g., increased throughput, work in progress reduction) by use of alternate configurations while reducing the buffer storage space.

In the case of this manufacturing facility, as with many facilities globally, the use of a bar code or radio frequency identification (RFID) is utilized which aids in facilitating the buffer cluster concept. Each time product moves from one station to another, the product bar code or RFID is scanned to ensure the prior processes are completed. Only if the prior processes are completed is that product picked from the buffer for the operator to perform the process at that station. Once the process is completed, the operator scans the product to inform the system that this process has been completed and puts the product back into the buffer. The buffer cluster may be partitioned and marked such that each station has a core area utilized by only that station and a shared area. Operators first focus on filling their core area and then move to shared area if needed. In an automated environment the shared area is a bin that may automatically move product to the core areas when space becomes available.

The groups of buffers in Figure 1-7 may vary in the number of buffers within the work cell and the number of serial stations in between the work cells as shown. The authors proposed transitioning the grouped buffers in the center of the work cell to a single buffer cluster shown in

Figure 1-8 which enables increased buffer utilization and reduces the size of the grouped buffers, reducing the buffer storage space. This allows the facility to benefit from efficiencies (e.g., increased throughput, work in progress reduction) by use of alternate configurations. In addition, in the case study in Chapter 5, we will discuss how sensitivity analysis of the buffer cluster size can be conducted using the models derived herein varying parameters such as the production demand and process times.

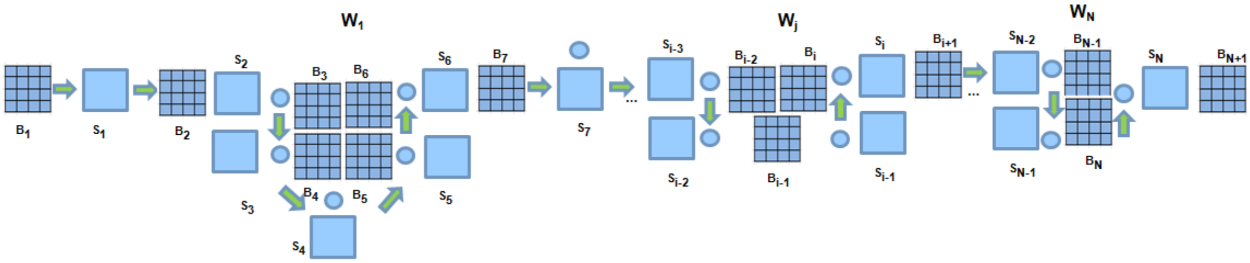


Figure 1-7. Work cells with one or more serial stations in between

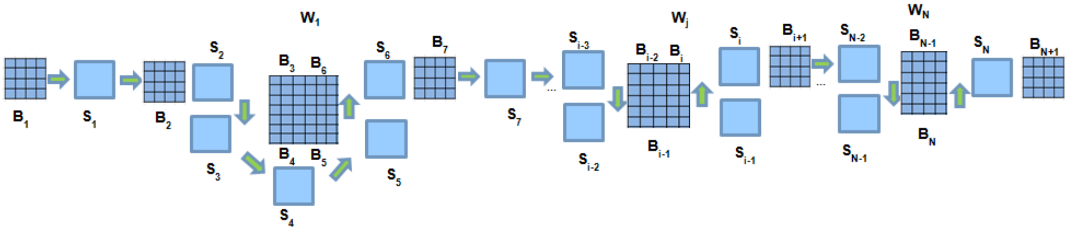


Figure 1-8. Work cells using buffer cluster concept with serial stations in between

Once the buffer clustering policy is identified for a production line, an activity relationship chart is created for the buffers and stations in the production line and the amount of space assigned to each activity is determined. From the space relationship diagram, one or more feasible layout concepts are generated. The optimal production line layout is then selected.

In Chapter 5, we apply all of the models created in the previous chapters to the real world distribution and manufacturing center examples. In the final chapter, Chapter 6, we discuss conclusions.

2. LITERATURE REVIEW

The class of discrete event systems of interest, in particular, the serial line as shown in Figure 1-6, has been studied quantitatively for years with numerous publications available. In this chapter a review of relevant literature is presented. Section 2.1 is devoted to the survey of Petri net software tools as referenced in the Introduction in Chapter 1. Section 2.2 covers state space research. Section 2.3 presents the quantitative analysis of production lines including the line balancing problem, the buffer allocation problem and queuing network and performance blocking. Section 2.4 of the literature also discusses concepts in 2.3 explored with the addition of flexible manufacturing systems with varying configurations (U-shaped, work-cells). We assess the literature in these areas and discuss how the research differs from the prior art.

2.1 Petri net software tools

2.1.1 Background

Discrete event systems consist of interacting components that are associated with a function that the system is intended to perform [2]. Discrete event systems are prevalent in several real-world applications including manufacturing, supply chain, healthcare and retail. Individuals in industry and academia have sought to model discrete event systems and develop techniques for design, control and system measurement of discrete event systems[2]. Such formalisms as Petri nets have their roots as a means for providing a common mathematical language to represent and model event driven networks. Petri nets were invented in August 1939 for the purpose of describing chemical processes [3]. Petri nets are assembled from places and transitions. Places represent resources that can be available or not. Individual resources are abstractly referred to as tokens[4]. In the early 1960s Petri nets were recognized as being the most sufficient method for modeling and analyzing concurrent processes and resource sharing. After exploring simple models manually, there came a need to do analysis of more complex models and assess the model's performance. As such, there was a need for computer software tools to enable and analyze models for larger and more complex systems. Computer software tools for Petri net Discrete Event Simulation and Non-Petri net Discrete Event Simulation evolved soon after and today are applied in several environments including manufacturing and distribution [5] healthcare applications[6], customer order and workflow systems [7] and stress or reliability analysis of a system [8] .

Although Petri nets have been used for decades as a means for modeling, simulating and analyzing concurrent systems in several environments, the practical usage of Petri nets is limited by the lack of computer tools which handle large and complex nets in a comfortable way[9]. Because of the lack of computer tools, the adoption rate in industry does not compare to that of non-Petri net discrete event system computer tools. Non-Petri net discrete event simulation, first emerging in the late 1950s and growing steadily since that time is now recognized as the most frequently used Operational Research techniques across a range of industries: manufacturing, travel, finance, and health[10]. Petri net computer tools have remained prevalent in academic environments but achieved a smaller level of adoption in industry worldwide. One reason for the greater adoption of Non-Petri net discrete event computer tools when compared to Petri Net computer tools, especially in industry is the availability of stable and well supported computer tools. For Non-Petri net Discrete Event Simulation, such tools include ARENA[11], Simul8[12], WITNESS[13], DELMIA[14] and SimEvents [15] that allow users to model large, complex systems easily.

There have been surveys of Petri Net tools completed in past years. Most notably, a Petri Net survey of tools [16] was completed that is also referenced by Petri Net World[17] provides a database of about 73 tools with several categories for those who use Petri net tools to identify what tool may be most suitable for their needs. The first category is the types of Petri nets supported by the tool, for example, Petri nets with time and stochastic Petri nets. It also provides a components category where the types of analysis are listed for each tool such as simple performance analysis, net reductions and structural analysis, and a listing of the operating systems that the tools support. Each category and sub-category is explicitly defined. Harald Storrle provided an evaluation of high-end tools for Petri nets [18]. The overall survey was conducted in three rounds. In the third round the focus was on a small number of tools that were evaluated thoroughly in terms of functionality, maintenance, openness and interfaces.

These surveys have several drawbacks. These surveys did not consider the computer software tool features and requirements needed to model real-world examples. Since these surveys were conducted the need for advanced features and requirements of computer tools has evolved, especially in industry. These surveys did not consider a solution that would require several criteria be met simultaneously for an industry application using Petri net software tools. In addition, a significant number of tools out there have websites that are no longer available, have not released an updated version of their tool in recent years or offer no maintenance or support for their tools.

This survey provides quantitative analysis across the span of Petri net computer software tools available today and illustrates and identifies trends in the Petri net computer tool software domain. It targets two groups of individuals: 1. Industry and academic professionals who are looking to use Petri net software computer tools to conduct their research and development work across a wide domain of uses including healthcare, supply chain, manufacturing, hospitality, etc. and 2. academic and industry professionals who are looking to develop or continue developing and supporting Petri net software tools. For the former group of professionals searching for viable Petri net computer tools for use for modeling, simulation and analysis this survey profiles 64 Petri net computer tools available today. The survey groups the Petri net computer tools according to whether or not they meet a set of defined criteria. The criteria were determined through a specific industry example and is defined and specified later in this survey. The results of the grouping set expectations for the actual number of tools that are suitable based on the defined criteria and provide academic and industry professionals with a head start for identifying what Petri net tools will work for them rather than taking a trial and error approach in identifying a suitable tool. For the latter group, this survey provides professionals with the primary and critical requirements and features sought out by academic and industry Petri net computer tool users. Focusing development effort on these features will ensure repeated and increased usage of these computer tools among academia and industry professionals.

This survey is organized as follows. In first section, a case study conducted by the authors is reviewed. The case study illustrates the need for Petri net software tools and was utilized to identify key criteria sought by Petri net computer tool users both in academia and industry. The next section translates the criteria into specific Petri net tool requirements. The requirements are defined into a higher level category. Within each category, the requirement is defined and a method of measurement for whether the Petri net tool meets the requirement is introduced. The following section describes the survey methodology, reviews the 68 Petri net computer tools and measures each category (within a subset of categories) using the defined measures. Data is collected, analyzed and put into a graphical format and results are interpreted and key findings identified. The survey then focuses upon key Petri net computer tools that most closely matched the criteria. Finally, the survey discusses key conclusions and recommendations.

2.1.2 Industry example

A distribution center provided the authors with data regarding the center’s process flow and loading. The process contained nine steps from part picking through shipping. The distribution center personnel was looking toward adopting sensor network in the form of radio frequency identification tags and other sensors (e.g. motion, weight, etc.) to gain efficiencies such as improved work in process distribution, station utilization and throughput in addition to bottleneck reduction. Each process step contained stations, resource pools, buffers and transportation methods with excess of 16000 units processed per 10-hour shift translating into over 500 Petri net elements (places and transitions) and 16000 tokens. During the data collection and analysis process, it was found that the distribution process was not effectively leveraging overlaps in resource skill sets and that more resource sharing across processes would enhance performance. In addition, it was found that there were interchangeable process steps for various entity types that were not being leveraged. An implementation of an efficient, real time control policy would be required to gather local and global sensing data to make real time decisions regarding resource and entity flows [1].

Given Petri nets' ability to efficiently model and analyze concurrent processes and shared resources, Petri net computer tools were the ideal tool to implement the distribution center processes and analyze control policies involving resource sharing and interchangeable process steps. In order to successfully create control policies a Petri net computer tool is required to model and analyze the system. Ideally, one would like to also use the Petri net computer tool to act as or interface with the controller for the live system implementing the control policies in real-time.

This case study was used to identify the selection criteria and requirements needed to model, analyze and implement real-time control policies in a large system containing hundreds of places and transitions and potentially thousands of tokens or more. This example has all the characteristics of problems in several domains such as manufacturing and supply chain [19] but also in other domains such as healthcare [20], retail and hospitality.

2.1.3 Petri net tool criteria, requirements, categories and measures

Suraj et al.[9] found three things are essential for modeling and analyzing by means of Petri nets - a good editor, a simulator and a powerful analysis engine. Moreover, a program should have a graphical user interface providing an opportunity to work directly with the graphical representations of Petri nets and should be able to read and write data in formats of other popular simulators of Petri nets[9] . The authors found these to be important criteria needed by a Petri net computer tool based on the case study and included them in the list. The case study pointed to several other criteria also measured as part of the survey.

The essential requirements for modeling and analyzing via Petri net computer tools are described below based on needs found in literature and the case study. The requirement is slotted into a higher level category where it is defined and a method for measuring the requirement across the Petri Net tools is specified. The category and measure for each computer tool is listed in Appendix I. How each category and requirement is measured for each of the computer tools in Appendix I is discussed below. The categories are Petri nets with Time, Performance Analysis, Multiple Attribute Support, Import and Export Capability, Graphical User Interface and Editor, Tool Support, Year of Last Tool Release, Operating System, and Commercial or Academic. Two criteria, Scalability and Live System Support, are also defined below, but were assessed on a subset of the tools based on the survey methodology described in 2.1.4

Petri nets with time:

Timed Petri nets provide a uniform environment for modeling, design and performance analysis of discrete event systems [21]. Wang states that the advantages of timed Petri nets include the ability to use the same modeling language for the specification/validation of functional/logical properties (such as absence of deadlocks) and performance properties (such as system waiting time). Timed Petri nets also enable modeling of system features such as priorities, synchronization, blocking and multiple resource holding.

A key requirement for Petri net computer tool is supporting timed Petri nets where a deterministic or stochastic time can be assigned to a place or transition within the net. Within the tools assessed in Appendix I, the tools were categorized as binary, either supporting timed Petri nets or not.

Performance analysis:

The tool shall enable simulation of system performance with time and output performance related parameters in a text file or graphically for a user to easily view Petri Net results. Performance parameters include average token time in place, transition firing delays, utilization of a place, etc. In addition, the tool shall simulate both deterministic and stochastic time intervals. The analysis engine of the Petri Net computer tool needs to simulate with accuracy and consistency. Within list of computer tools in Appendix I, tools were categorized as binary, either supporting performance analysis or not.

Multiple attribute support:

The tool shall allow user to set attributes to tokens as in colored Petri nets. Multiple attributes should be able to be assigned to a token at multiple points within the model and those attributes should be able change dynamically as the token traverses through places and transitions within the model. For example, if a token visits a place, a token attribute is assigned and may change over time based on the most recent places visited. Such behavior may be emulated via Colored Petri nets. Within Appendix I, the Petri net computer tools were categorized as binary, either supporting Colored Petri Nets or not.

Import and export capability:

The tool shall support import and export capability by supporting a simulator that reads and writes data in formats of other simulators of Petri nets. The tools were categorized in Appendix I as supporting import and export capability or not.

Graphical user interface & editor:

The tool shall have a graphical user interface providing an opportunity to construct, edit and work directly with the graphical representations of Petri nets (places, transitions and tokens). The tool shall support the ability to easily apply mathematical rules to tokens traversing places and transitions throughout the model (AND, OR, etc.). The tools were categorized in Appendix I as binary, either supporting a GUI editor and simulator or not. Ideally, the tool should also enable creation of subsystems for modularity and ease of re-use of modules throughout the model.

Tool support:

The tool shall have a support team to answer questions and provide workarounds or fixes to bugs and defects found in the tool. The tool has had a release or software update within 3 years. Petri net tools were assessed in terms of the tool support they provided. The tools were categorized as follows:

1. *No Support:* The tool does not offer any support. After the tool is downloaded, no support is available.
2. *Limited Support:* The tool offers limited support. The support team may answer questions and may consider fixing some minor bugs.
3. *Full Support:* The tool offers substantial support. The support team answers any questions about the tool, welcomes feedback and comments, and assesses and fixes major and minor tool defects and bugs or provides suitable workarounds.

Tool support Categorization was made based upon information available at the Petri net tool websites and attempts to contact the tool owner(s) via phone or email.

Year of last tool release:

The year of the last release or update of the tool can be a good indicator of the level of support one might receive, the ease of use of the tool and performance (e.g. speed). Given the evolution of operating systems and the impact of that on application design, one typically observes some level of difficulty with tools that have not evolved with updated operating systems. The survey lists the last observed year of release or update of the tool.

Operating system:

The computer tool shall be supported on recent operating systems for desktops and laptops. Within the survey, the following categorization is used:

1. For a Windows environment, Appendix I lists the most recent version available explicitly, for example, 2000, XP, Vista or Win 7+. If a less recent version is available, the survey is left blank.
2. If a Linux environment is available, “Linux” is listed; otherwise, it is left blank.
3. If neither Linux nor a recent Windows version is available, the survey lists “None”.
4. If the tool is available in Java, depending on the Java Run Time Environment Version, a Windows, MAC OS X or Linux version is provided.
5. If the computer tool is supported on MAC OS X, Appendix I lists MAC OS X.

Commercial / academic:

Whether a tool is commercial or academic is not a requirement, however, it can imply cost. Within the survey, tools marked as “Academic” are implied as available at no cost, while “Commercial” implies a cost. If marked “Both” it implies that a version is available for academia at no cost or a discount while a commercial version is also available at a cost.

Scalability:

The tool shall support the modeling of 500+ Petri net elements (places and transitions) and 1000+ tokens without becoming unstable. By effects visible to the user during instability, is that the computer tool yields incorrect results and can become unresponsive causing the tool to crash. The industry example discussed earlier provided a baseline for the level of complexity and scalability criteria a Petri net tool should support to enable Petri net applications in industry. Scalability of the tools listed in Appendix I will be discussed in 2.1.4.

Live system support:

The Petri net computer tool shall support a “live” mode, where the user is able to exit simulator mode and port real time data from devices such as sensors, databases and wireless devices into the model. The computer tool, therefore, should provide an interface to real-time enforcement of decision control policies. Live system support of the tools listed in Appendix I will be discussed in 2.1.4.

2.1.4 Petri net survey methodology and findings

A total of 80 Petri Net tools were discovered either on-line, through paper references or previous surveys. Of the 80 tools, approximately 20% of the tools are either no longer available, on websites that are no longer indexed or their corresponding academic research went in a different direction where the tool evolved and exited the Petri net domain. The remaining 64 tools surveyed are in Appendix I. The survey methodology is as follows. Individual criteria were assessed for each tool, except for scalability and live system support. To support the case study, simultaneous support of criteria in a single tool is required. After assessing each individual criteria in the tool, an assessment of what tools supported three or more criteria simultaneously was conducted that narrowed the available tools to small group. Scalability and live system support were then assessed on the smaller subset of tools. In this section, we use our collected data and observations (including what is reported in Appendix I) to analyze the current status of Petri net tools and draw conclusions when possible about the needs and potential research issues in this area.

First and last observed year of tool release:

Figure 2-1 shows a profile of the observed first and last tool release year. One of the earlier tools ARP [22] had an initial release in 1988 and is one of the earlier tools whose website is still available. The reason for this is that the early 1990s had several breakthroughs with respect to Petri net model checking [23] and ease of programming and software environments for creating more sophisticated software tools in general. There were multiple tools in the early 1990s, but there were observed only a handful that has their website still available. Of the tools found and referenced, 31% (20 of the 64 tools) have had a tool release in less than 3 years. Not having a release within three years is an indicator that the tool will not evolve with new features and functionality in the future and will most likely no longer be supported from the defect fix perspective. Given the smaller population of users utilizing these types of analysis tools in general, three years is a standard period of time for a tool to get deployed, used and get feedback or fixes in for another release. Tool updates peaked in 2006 with 9 tools releasing a software update followed by 8 tools making a release 2015.

Across all of the tools, the average life expectancy is 6.6 years. The life expectancy was calculated as the difference (in years) between their first and last release. It is assumed that these tools have reached end of life. The number of new tools has declined since 2010, with no new tools from 2011 through 2016.

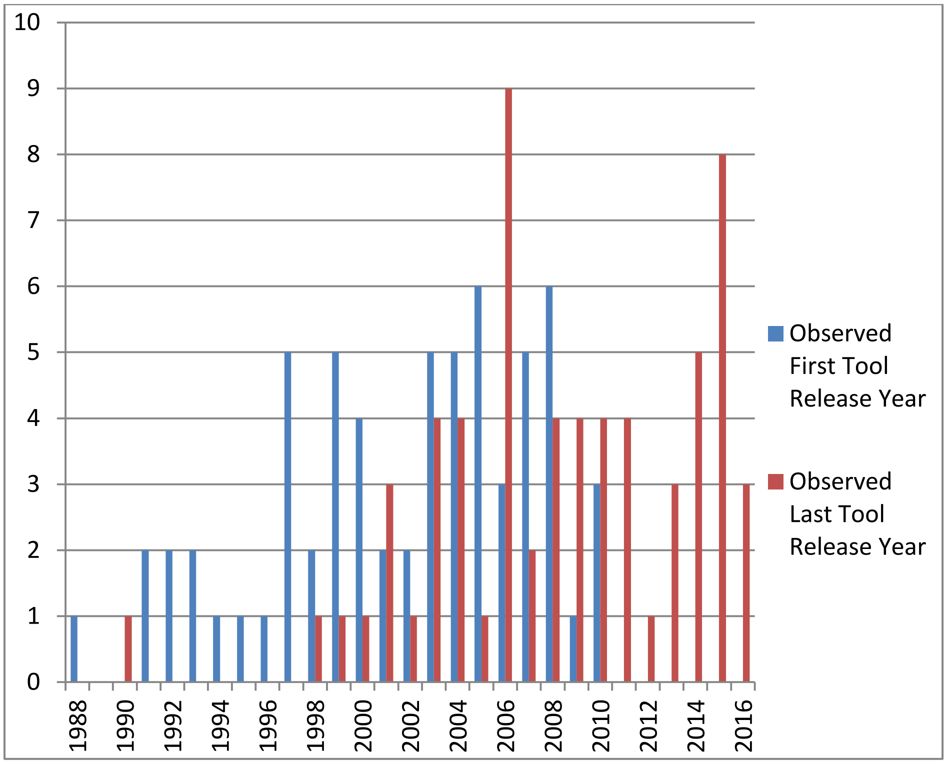


Figure 2-1. Number of tools versus observed first and last year of release

Analysis of tool support:

Figure 2-2 show for the Petri Net tools assessed how many offered full, limited or no support. Of the 64 tools, 22% offer no support, 69% offer limited support and 9% offer full support. Three of the 6 tools with full support had a last observed release after 2012.

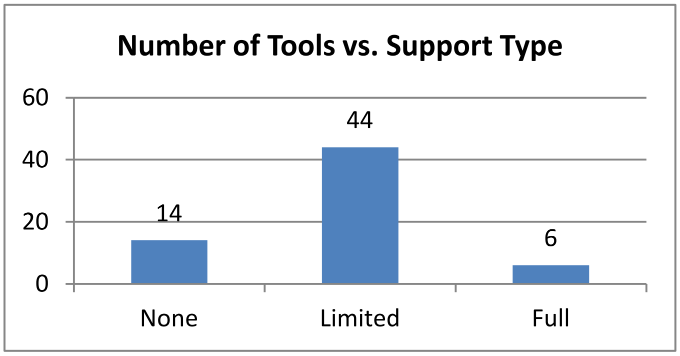


Figure 2-2. Number of tools versus support type

Operating system:

All of the tools but 1 [22] could run on a Windows OS of 2000 or later and Linux. However, for tools developed in the 1990s and early 2000s suffered in areas of performance (speed) and ease of use. There were signs of latency when working with the graphical editor and the drag and drop features were not as reliable.

Petri nets with time, performance analysis and token attribute setting:

Of the 64 tools assessed, 34 (53%) supported timed Petri nets, 26 (41%) supported performance analysis, 33 (51%) supported multiple attributes or Colored Petri Nets, 32 (50%) supported Interchange Import and Export across tools and 51 tools (80%) supported a graphical editor. 25 (39%) tools supported both timed Petri nets and performance analysis, 13 tools (20%) supported Timed Petri Nets, Performance Analysis and Colored Petri Nets. Of the 13 tools that met the Time Petri Net, Performance Analysis and Colored Petri net criteria, those that had a release in 2012 or later were narrowed to 5 tools.

Figure 2-3 shows how the number of options narrows quickly when multiple criteria are applied simultaneously to the list of tools in Appendix I. The graph shows that when 1 & 2 are applied, options decline from 64 to 25. When criteria 1 and 2 and 3 are applied options decline to 13 and when the software release after 2012 is applied, this narrows the options more to 5 tools.

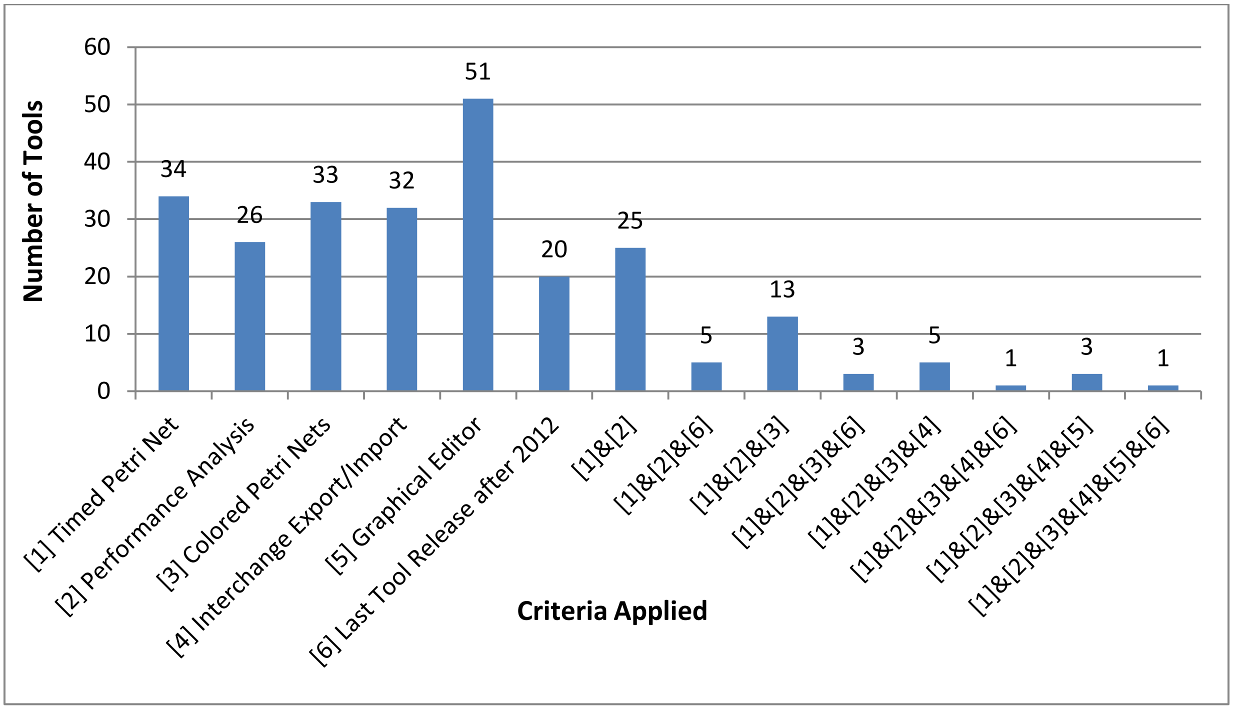


Figure 2-3. Number of Petri net tools meeting multiple criteria

Graphical user interface and editor and import and export capability:

The idea of an interchange format for Petri nets has been around for some time, especially in the last decade with the development of interchange formats based upon Extended Markup Language (XML). A tool should support such a convenient way to exchange information across different Petri net types [24][25][26][27] . 51 of the 64 (80%) tools had an editor and 52% (33 of the 64 tools) had import and export capability. For the tools that did not provide a graphical editor, it was because that tool was focused on either providing code blocks used to generate a GUI or the tool was focused upon model verification or analyzing of the Petri net and not building the Petri net. The tools provided a textual net file format for which to create the Petri net or a method to import a textual file into the tool.

Scalability:

All tools did not explicitly list scalability within the website, user guide, etc. In fact, Petri net tools were rarely explicit about how many elements (Petri net places, transitions and tokens) they could support and prior surveys have not inquired about scalability of models. Storrle's survey [16] assessed the scalability for a handful of tools where the data was available. About 7% were explicit about the number of elements supported. For example, Pnet Lab[28] supported only 25 places and transitions. Poses++[29] supports up to 500 elements without a license and claims supporting test models with a license of plants containing up to 30,000 transitions, 30,000 predicates and 100,000 arcs. Poses++ targets fast simulation of high level Petri nets focused upon colored and predicate/transition nets only, so does not meet the primary requirements sited above.

Petri net Toolbox[30] is a plug-in with Matlab[31]. The tool enables analysis of timed Petri nets, allows one to choose distribution driving stochastic Petri nets and enabled one to monitor and graph the number of tokens and utilization of places. The case study exhibited a need to model close to 500 places. However, after modeling close to 50 places, the tool would crash and there were accuracy issues, for example, transitions did not fire appropriately or transition the correct number of tokens. Tools like GreatSPN [32], Renew [33], and CPN Tools [34] were also assessed. Renew is Java based and one must program time related performance metrics as no library exists. For large nets where several performance measures are required for each place, not having a default preset of performance measures burdens the user and does not make sense for large nets in industry applications. GreatSPN and CPN Tools exhibited similar symptoms as Petri Net Toolbox and were unstable once large nets (>50 elements). CPN Tools is similar to Renew in that a programming inscription language (not as common as Java) is required to program in the user interface, but requires a steeper learning curve. Yasper [35] attempts to address the ease of use issues with tools like ExSpecT [36], Renew and CPN Tools.

Live system support:

The survey did not uncover Petri net tools supporting a live system mode where one could port real time data and execute decision and control policies in real-time. The co-habitation of modeling, simulation and real-time execution of decision and control policies was found to be important in model validation and realizing the benefits of policies targeted at reducing cycle times, improving throughput or reducing work in progress. As the case study would implement the control policies in an environment where sensing and control technology were present, linking the model to the live system would prove beneficial during validation.

2.1.5 Conclusion

The survey has found, that although a plethora of Petri net tools exist, with a number of tools providing software updates, very few support the primary requirements and features identified as part of the case study and required in domains such as manufacturing, supply chain, and healthcare. Features such as timed Petri nets, colored Petri nets, and performance analysis along with adequate tool technical support are critical for increasing Petri net tool adoption in industry. In addition, features such as scalability or the ability to model a larger scale system reduce the available tool set even further. Lastly, live system support is a desired requirement not supported in Petri net tools included in the survey. The co-habitation of modeling, simulation, and interfaces to the live system for real-time execution of the model is key to monitoring and measuring of the benefits of decision and control policies implemented. Given the lack of a tool that supported the needs of the case study, the authors turned to a non-Petri net solution [15].

2.2 State space and state space reduction

The key issues faced by large, N-Server, N+1 Queue serial lines, is that the number of possible states grows exponentially during system execution [37] due to the large number of elements or components that make up the system and the number of entities that traverse the system. In some systems, the execution of a large number of entities is required to exhibit a specific system level characteristic leading to a specific state. This large number of entities results in a large number of states, some or most of which may be unnecessary or can be significantly reduced to reach the state that is most relevant to the system's characteristic state space. These extraneous states can make it difficult to understand the characteristic state space and cloud the ability to enable control and decision policies for scheduling systems, an important aspect of the real-time control of dynamic systems [38]. In industry, where systems are well known for having large state spaces, control policies are instantiated using reachability techniques [39] [40] or simulation methods [41][42]. Once state spaces become large, it is often difficult to determine why one control policy behaves better or worse than another control policy.

State space explosion and methods to reduce state spaces have been researched extensively in areas of automatic verification of systems and model checking. Clark et al [37] uses partial order techniques while Musuvathi [43] describes three different methods. Musuvathi's first method is down-scaling which reduces the scale of the system, for example, by reducing the number of nodes. This has a large risk for eliminating critical states that may contribute to the characteristic state space of the system. The second method is abstraction of state, which standardizes distinct but equivalent states and eliminates information that is judged to be unimportant for the properties checked. With this method, there is high risk of eliminating a state that is pertinent to the characteristic state space of the system. The final method, using heuristics when checking the entire state space, is infeasible and provides for more intelligent methods for checking and reducing the state space.

The next group of state space reduction literature involves the use of Petri net formalism to reduce the state space. In the early 1960s, Petri nets were recognized as being the most sufficient method for modeling and analyzing concurrent processes and resource sharing. Sloan and Buy [44][45] discuss how reachability based methods suffer from the state explosion problem and extend several rules for the reduction of ordinary Petri nets. They also provide for the notion of equivalence among time Petri nets proving that the reduction rules yield equivalent nets such that timing and concurrency properties are preserved. Wang et al [46] proposes a set of component-level reduction rules for timed Petri nets that reduces the state space while maintaining the net's external observable timing properties. Juan et al [47] propose reduction methods using delay time Petri nets.

This research proposed in Chapters 3 differs from the above literature sited in that it derives a time-based closed form solution for determining the state of every buffer and server in an N+1 Buffer, N-Server sequential line while keeping the entire system intact. In doing so, a closed form solution for the maximum number of entities a buffer will experience is also derived.

2.3 Line balancing and buffer allocation optimization

Quantitative analysis of assembly lines, such as those in Figure 1-6, includes the line balancing problem which comes along with other decision problems such as the positioning and sizing of the buffers [48], when keeping the overall throughput in mind becomes the buffer allocation problem [49]. The decision problem of optimally partitioning (balancing) the assembly work among the stations with respect to some objective is known as the assembly line balancing problem (ALBP) [48]. The previous literature has categorized assembly line balancing into two main categories (1) Simple Assembly Line Balancing Problem (SALBP) and the (2) General Assembly Line Balancing Problem (GALBP). There are several methods that have been utilized for solving such problems including deterministic, stochastic and inexact methods (where heuristic or approximate methods are used).

The ALBP assigns tasks to the station while optimizing some criteria and not violating the constraints. The ALBP problem has the following considerations where all inputs are known with certainty, a task cannot be split among two or more stations, tasks cannot be processed in arbitrary sequences and all tasks must be processed [50]. SALBP adds the following considerations: (1) all stations are equipped and manned to process any one of the tasks (2) the task process times are independent of the station which they are performed (3) any task can be processed at any station (4) the total line is considered to be serial with no feeder or parallel sub-assembly lines and (5) the assembly system is assumed to be designed for a unique model of a single product [50]. SALBP-1 is the first version of SALBP that adds an additional constraint that the cycle time is given and fixed. The goal of SALBP-1 is to minimize total slack or the number of stations along the line [51][52]. A second version of SALBP called SALBP-2 replaces constraint of fixed cycle time with the constraint of a fixed number of stations. For this version, the goal is to minimize cycle time or production rate [53][54][55]. Methods utilized to solve SALBP-1 are linear programming [56], integer programming [57], specialized algorithms based on integer programming techniques [58] and dynamic programming [59]. Heuristic methods are utilized to solve SALBP-2 class of problems [60][61]. GALBP is a generalized form of SALBP-1 and SALBP-2 where there is no explicit concern for the fixed cost of the station or the variable cost for operating the station [50].

There are relevant properties for characterizing ALBPs due to the different conditions in manufacturing and assembly line systems [48]. For example, for paced assembly lines, every station is limited to the cycle time which can be no longer than the largest task time. When all stations operate at an individual speed, entities must wait before they can enter the next station or they become idle, unless there are buffers in between stations. For an unpaced, buffered assembly line, the line balancing problem comes along with other decision problems such as the positioning and sizing of the buffers [48].

One of the key problems in designing a production flow line is determining the number and sizes of buffers between stations keeping overall throughput in mind. This is known as the Buffer Allocation Problem [49]. Wei et al. provide an estimation of buffer size in a serial manufacturing system within a stochastic optimization problem [62]. Chaharsooghi and Nahavandi [49] present a heuristic algorithm to find the optimal allocation of buffers that maximizes throughput. Yamashita and Altiok [63] use a dynamic programming algorithm that uses decomposition for a minimum-total-buffer allocation resulting in a desired throughput in production lines with phase-type processing times.

There are several buffer allocation strategies: Equal Buffer, Chow’s Rule, L&L’s Rule, and C&N’s Rule [64]. Equal buffer strategy allocates buffers equally over the line. Chow’s rule [65] uses dynamic programming to solve the buffer allocation problem with a fixed total buffer size. Throughput and the coefficient of variation of inter-departure times are estimated by regression models. L&L’s rule is similar to Chow’s rule. It uses a different set of equations to estimate the throughput and the coefficient of variation [66]. C&N’s rule is similar to L&L’s rule except all possible allocations of buffers are tried and then the allocation with the highest estimated throughput is selected. Gershwin and Schor [67] define and analyze two problems, one called a primal problem that minimizes the total buffer space subject to a production rate constraint and another, a dual problem that maximizes production rate subject to a total buffer space constraint. Enginarlar et al [68] discuss the concept of level of buffering (LB) and provide a method for calculating the smallest LB ensuring the desired production rate in serial lines with unreliable machines and later[69] introduce Lean Level of Buffering (LLB) where a normalized buffer capacity and production line efficiency is used to develop exact formulas for two and three machine lines and approximations for lines with more than three machines.

The research proposed differs from the line balancing and buffer allocation literature sited as the proposed research identifies exact methods for determining the state of the system or an element of the system at a given time. States include the number of arrivals and departures an element sees at a given time, the maximum number of entities a buffer will experience given the placement of the stations and buffers and the time the buffer reaches capacity. If a buffer has a reduced capacity, the time the buffer transitions to a blocked state and the time the buffer becomes blocked. Such research can be used as boundary conditions or inputs to a line balancing or buffer allocation optimization problem to reduce solution time for line balancing optimization problems or to generate control policies for improved production efficiencies.

2.4 Queuing networks performance and blocking

Queuing networks were first used to model manufacturing systems in the 1950s [70]. Performance analysis is important for the design, operation and management of production systems [71]. The previous section covered the aspect of optimization related to queues and buffers, the dimensioning and placement of queues and related line balancing. There is also literature that addresses use of queuing networks for performance evaluation of queuing networks assessing metrics such as production rate or throughput, average buffer levels and probabilities of blockage and starvation.

Gershwin [72] developed an efficient decomposition method using conservation of flow for evaluation of performance measures for production systems with finite buffers. Gershwin points out the difficulty to evaluate queuing networks due to their large state spaces and presents a method of approximation for calculating production rate or throughput and the average amounts of material in the buffers. Lim et al [73] developed an aggregation method consisting of both forward and backward aggregation that converges on a production rate representing the system throughput eliminating the need for complex and costly computer simulations. Kouikoglou and Phillis use a probabilistic technique [74] that observes a limited number of events which are sufficient to determine the system performance and mean buffer levels. Earlier, the same authors developed a hybrid simulation and analytic model for the study of production networks where they utilized nonlinear difference equations to determine and obtain accurate estimates of average throughput and buffer levels. The algorithm was more efficient that traditional simulators [75][76]. Morrison [77] demonstrates that flow line models with deterministic services times can be decomposed into segments and uses recursion to calculate overall delay of entities in the system.

The research proposed in Chapter 3 differs from queuing network performance and blocking literature sited as the proposed research identifies exact methods for a class of networks, particularly reliable sequential production lines of infinite length, determining the state of the system or an element of the system at a given time without the need for simulation. States include the number of arrivals and departures and element sees at a given time, the required buffer capacity given the placement of the stations and buffers and the time the buffer reaches capacity. If a buffer has a reduced capacity, the time the buffer transitions to a blocked state and the time the buffer becomes blocked is captured. Such research can be used as boundary conditions or inputs to assessing system throughput or to generate control policies for improved production efficiencies.

2.5 Flexible manufacturing

Facilities everywhere are facing growing competition and must find ways to maximize production efficiency to remain competitive in the market place [78] . Facilities are assessing alternate production line configurations to gain production efficiencies such as throughput increases or work in progress reduction while maintaining the chronological order of operations.

Literature also discusses concepts explored with the addition of flexible manufacturing systems with varying configurations (serial, sequential, work cells) and product types [79][80]. We turn to the industries where work cell configurations are used, such as mobile device, wood, and apparel production industries [81][82][83] or any production line where a work cell configuration exists.

Considering flexible manufacturing systems and work cell literature, Ramirez-Serrano and Benhabib [84] introduce a control algorithm to analyze concurrent operation of supervisors to check for existence or absence of deadlock states within a work-cell. Outside of supervisory control, there have been several studies that investigate the utilization of work-cell and reconfigurable manufacturing systems to increase the efficiency and capacity of production lines. Ichikawa's study [85], for example, investigates a laptop production system and optimizing the supply of parts via material handlers from the receiving area to the cells. Another study [78] analyzes use of product-oriented layout, material handling and layout of work-cells to maximize production efficiency in areas such as average units produced per day, labor cost per unit and distance traveled per day to obtain parts. Logendran and Karim [86] uses a non-linear programming model comprised of binary and integer variables and a tabu search type algorithm to address the availability of alternative locations for a work-cell and the use of alternative routes to move part loads between cells when capacity of the material loader is limited. Youssef and ElMaraghy [87] introduce a configuration selection approach that minimizes reconfiguration effort but still supporting the capacity needs of production.

This paper differs from the prior literature reviewed in that it presents methods for extracting the buffer size where the buffer space is shared by several stations (via a buffer cluster) using methods derived from state space parameters with respect to time for any sequential N-Server, N+1-Buffer production line. The buffer sizing model is then utilized in an optimization framework that enables setting of the policy specifying the buffers that can be clustered ensuring no buffer overflows. The model provides an output of the required buffer cluster sizing for that policy and allows the facility to set the policy that minimizes space utilization of the production line without decreasing the number of overall production lines that fit within the facility.

3. EXACT METHODS FOR DETERMINING STATE SPACE PARAMETERS

This chapter derives exact methods for determining state space parameters for two models both using the N-Server, N+1-Buffer sequential line defined in Figure 1-6. In the first model, we impose no buffer capacity restriction and derive state space parameters. In the second model, we impose a buffer capacity restriction and in addition to the state space parameters, develop a decision tree for determining the servers and queues impacted by the buffer restriction.

3.1 Model 1: N server, N+1 sequential line model with unity capacity servers and no buffer capacity restriction

In this section, we define the parameters used for deriving the number of arrivals and departures at a given time *t* for any server or queue in Figure 1-6. We start by calculating the number of arrivals and departures at Server S_i, i = 1, 2…N by any time t. Next we derive the number of arrivals and departures from any buffer B_i, i = 1, 2,…N+1 by any given time t. We then apply the relationships generated to derive the maximum number of entities a buffer B_i will experience and the number of entities at any given time t, $B_i(t)$. $B_i(t)$ is then later extended in Chapter 4 to determine the buffer cluster size. Before deriving the aforementioned relationships, we list the notations, assumptions and definitions.

Notations:

N1) K1 = Magnitude of inventory at B_1 at time t=0, K1 = 1, 2,…N (Constant).

N2) $BA_i(t)$ = Cumulative number of arrivals to buffer B_i by time t, i=1,2,…N+1.

N3) $BD_i(t)$ = Cumulative number of departures from buffer B_i by time t, i=1,2,…N+1.

N4) $SA_i(t)$ = Cumulative number of arrivals to server S_i by time t, i=1,2,…N.

N5) $SD_i(t)$ = Cumulative number of departures from server S_i by time t, i=1,2,…N.

N6) T_i, i = 1…N is the service time for server S_i, i = 1…N (This includes both the process time of the unit and the transportation time of the unit from the buffer to the station and the station to the next buffer).

Assumptions:

A1) Each Server S_i can process at most one entity at a time (capacity = 1).

A2) Each buffer B_i, i = 1…N+1, has a capacity greater or equal to the starting inventory

A3) Service time T_i for each server S_i is deterministic

A4) The starting inventory at time $t=0^-$, K1 is located in buffer B_1

A5) At time $t = 0$, B_1 has a departure and S_1 has an arrival

A6) Buffer B_1 has only departures while Buffer B_{N+1} has only arrivals and every buffer B_i in between has both departures and arrivals; $BA_1(t) = 0$; $BD_{N+1}(t) = 0$ as shown in Figure 1-6.

A7) If there is at least one part in B_i and S_i is idle, then with no delay, an entity is moved to S_i for processing.

A8) Machines are reliable.

Definitions:

D1) $MT_i = \max[T_1,T_2…T_i]$, i=1,2,…N.

D2) $\tau_i = \sum_{j=1}^{i} T_j$, i =1,2.....N, and $\tau_0 = 0$.

D3) MB_i = Maximum number of entities that buffer B_i, i = 2,..N will experience.

D4) We use $\lfloor \ \rfloor$ as a floor function that maps a real number to the largest previous integer value.

This model supports adding inventory (that could be in the form of batches of varying sizes) anytime before the last item in inventory B_1 leaves the first server S_1 (i.e. a batch can be added anytime before t = $\tau_{i-1} + (K1-1)*MT_i$). K1 can be either the initial inventory or a summation of inventory (in the form of batches) throughout the shift.

Before proceeding to calculate the number of arrivals and departures from server S_i we derive relationships for the frequency of arrivals to server S_i. The following Lemma establishes this relationship.

3.1.1 Deriving the frequency of arrivals to Server S_i

Lemma 1: The frequency of arrivals to Server S_i is $\frac{1}{MT_i}$

We use induction to demonstrate this. We start by focusing on the first two servers, S_1 and S_2 as shown in Figure 1-6. Given buffer B_1 holds the inventory of entities for the sequential line, the frequency of arrivals to S_1 is equal to $\frac{1}{T_1} = \frac{1}{MT_1}$. We also calculate the frequency for S_2 as every other server in the line is similar to S_2 in the way that it is preceded by another server. For S_2 and its preceding server S_1, we have 3 conditions for the service times:

1. $T_1 > T_2$
2. $T_1 = T_2$
3. $T_1 < T_2$

Figure 3-1 shows a typical scenario when $T_2 > T_1$. In this case, the frequency of arrivals to S_2 (immediately after departure of the first entity) is equal to $\frac{1}{T_2}$. Figure 3-2 shows that if $T_1 > T_2$, the frequency of arrivals to S_2 is $\frac{1}{T_1}$.

Figure 3-3 shows that if $T_1 = T_2$, the frequency of arrival to S_2 is equal to $\frac{1}{T_2}$ or $\frac{1}{T_1}$.

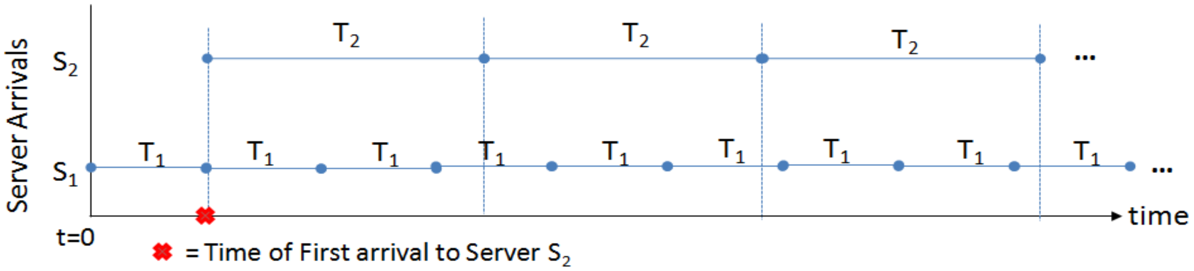


Figure 3-1. Frequency of arrivals to S_2 when $T_2 > T_1$

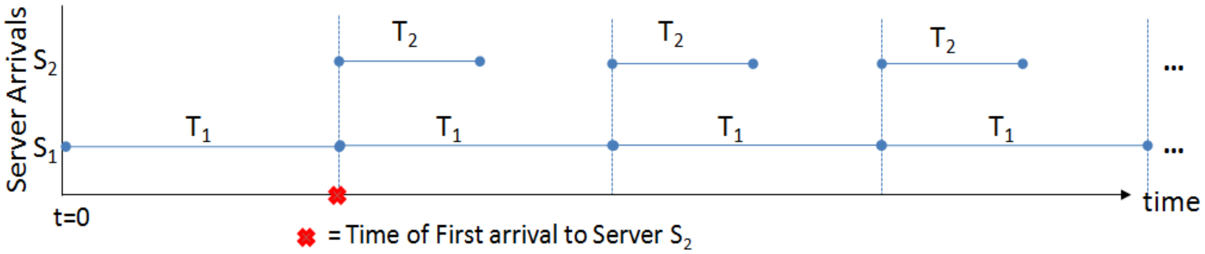


Figure 3-2. Frequency of arrivals to S_2 when $T_1 > T_2$

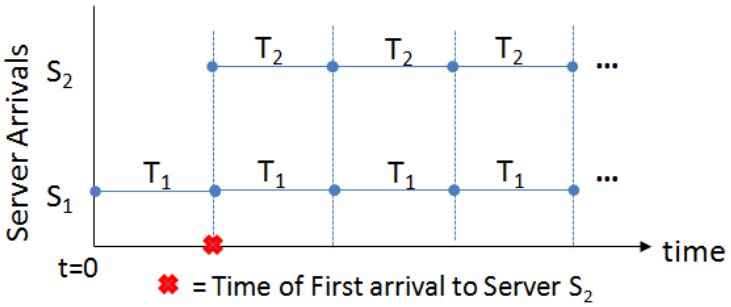


Figure 3-3. Frequency of arrivals to S_2 when $T_1 = T_2$

Therefore, from the definition of MT_2, for server S_2, the frequency of arrivals is always $\frac{1}{MT_2}$. Now we assume the first arrival occurs at τ_{i-2} for S_{i-1} ($i \leq N-1$) and after that arrivals to S_{i-1} occur in intervals of MT_{i-1}. We prove that the interval for arrivals to S_i is MT_i. In this case, we have two scenarios:

1. $T_i > MT_{i-1}$
2. $T_i \leq MT_{i-1}$

Again, we are concerned with the frequency of arrivals immediately after the departure of the first entity from S_i shown by X in Figure 3-4 and Figure 3-5. We demonstrate that the frequency of arrivals immediately after the departure of the first entity from Server S_i is the reciprocal of the maximum of services times MT_{i-1} and T_i or $\frac{1}{\text{Max}\,(MT_{i-1},T_i)} = \frac{1}{MT_i}$. For scenario 1, from Figure 3-4, the frequency of arrivals to server S_i is $\frac{1}{T_i} = \frac{1}{\text{Max}\,(MT_{i-1},T_i)} = \frac{1}{MT_i}$. For scenario 2 as shown in Figure 3-5 the frequency of arrivals to S_i is $\frac{1}{MT_{i-1}} = \frac{1}{\text{Max}\,(MT_{i-1},T_i)} = \frac{1}{MT_i}$. It is a trivial case when $MT_{i-1} = T_i$, then the arrival rate to S_i is $\frac{1}{MT_i}$. This completes the proof of Lemma 1.

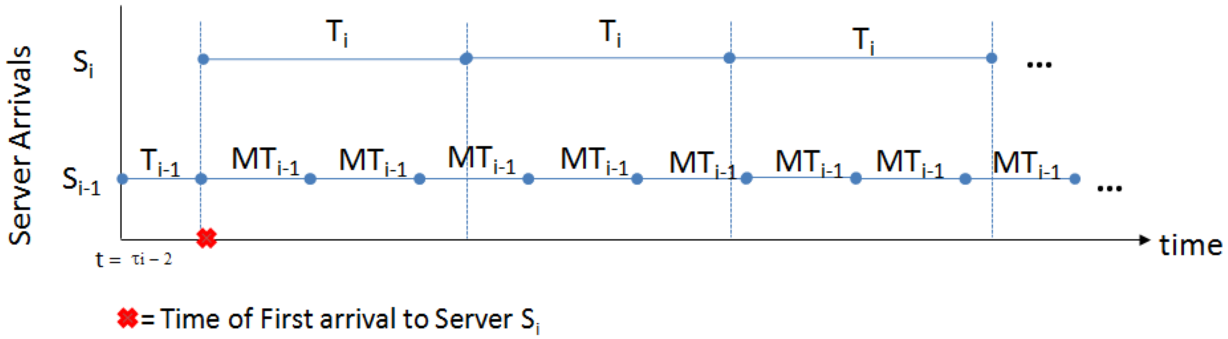


Figure 3-4. Frequency of arrivals to S_i when $T_i > MT_{i-1}$

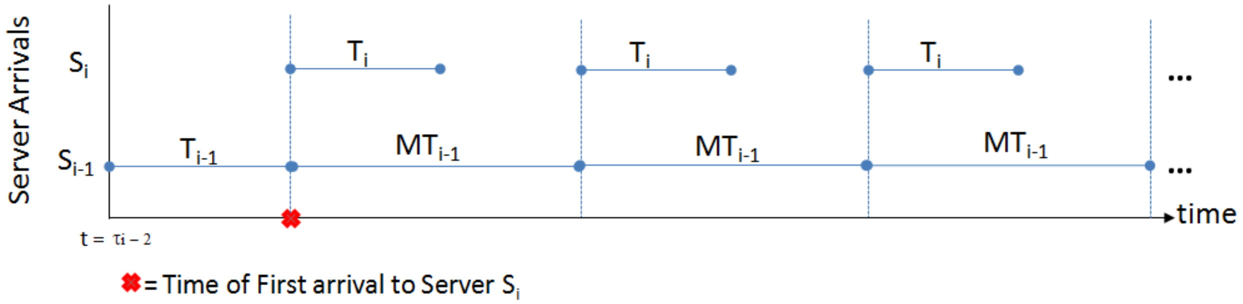


Figure 3-5. Frequency of arrivals to S_i when $MT_{i-1} > T_i$

■

3.1.2 Deriving cumulative arrivals and cumulative departures at server S_i

Theorem 1: For the sequential system the cumulative arrivals and cumulative departures at server S_i at time t is:

$$SA_i(t) = \begin{cases} \min\left\{ K1, 1 + \left\lfloor \frac{t - \tau_{i-1}}{MT_i} \right\rfloor \right\} & \text{if } t \geq \tau_{i-1} \\ 0 & \text{Otherwise} \end{cases} \qquad (3.1)$$

$$SD_i(t) = \begin{cases} \min\left\{ K1, 1 + \left\lfloor \frac{t - \tau_i}{MT_i} \right\rfloor \right\} & \text{if } t \geq \tau_i \\ 0 & \text{Otherwise} \end{cases} \qquad (3.2)$$

Proof: First, we show that (3.1) holds for i=1. At i=1, we have $\tau_1 = \sum_{j=1}^{1} T_j = T_1 = MT_1$. $\tau_0 = 0$. Based on the sequential line assumption, at time t = 0, one part is loaded to server S_1 (recall that $K1 \geq 1$). This part is processed for $T_1 = \tau_1$ units of time and if buffer B_1 still carries a part, server S_1 is loaded again. This loading operation (arrival event) happens at time $t = \tau_1$. Continuing with this pattern, one can see that server S_1 is loaded at time stamp 0, τ_1, $2\tau_1$,…., $K1\tau_1$, therefore the last loading of server S_1 happens at time $t = K1\tau_1$. After this time no loading occurs as all the parts in buffer B_1 have been depleted, and the total number of arrivals to S_1 remains K1. This means that the cumulative number of arrivals to server S_1 at time t can be shown by:

$$SA_1(t)=\begin{cases}\left\lfloor \frac{t}{\tau_1} \right\rfloor +1 & 0 \leq t < K1\tau_1 \\ K1 & t \geq K1\tau_1\end{cases}$$

and it can immediately be concluded that $SA_1(t)= \min \left\{K1,\ 1+\left\lfloor \frac{t}{\tau_1} \right\rfloor\right\}$, $t \geq 0$.

This proves that (3.1) holds for i=1. Second, we prove (3.1) holds for S_i where $1 < i \leq N$. By definition for $t < \tau_{i-1}$, $SA_i(t) = 0$ and for $t = \tau_{i-1}$, $SA_i(t) = 1$. Figure 3-6 shows the cumulative arrivals to server S_i at any time t. We notice that the interarrival times are MT_i. Now we consider the case where $1 < SA_i(t) \leq K1$.

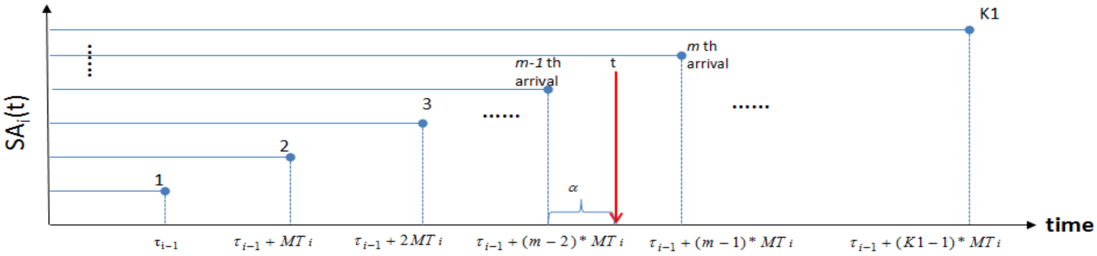


Figure 3-6. Time defined for $SA_i(t)$ from 1 through K1th arrival

Assume that $\tau_{i-1} + (m-2)*MT_i < t < \tau_{i-1} + (m-1)*MT_i$ where m – 1 is an integer number and is the number of arrivals before t. We know from the relationship that the frequency of arrivals to Server S_i is $\frac{1}{MT_i}$ and that for the m - 1th and mth arrivals, time is defined in the following interval:

$\tau_{i-1} + (m-2)*MT_i < t < \tau_{i-1} + (m-1)*MT_i$

We can write:

$t = \tau_{i-1} + (m-2)*MT_i + \alpha*MT_i$ where $0 < \alpha \leq 1$

$t - \tau_{i-1} = ((m-2) + \alpha)*MT_i$

When $\alpha = 1$, the coefficient $((m - 2) + \alpha) = m - 1 = \left\lfloor \frac{t - \tau_{i-1}}{MT_i} \right\rfloor$ and S_i experiences the mth arrival. For $0 < \alpha < 1$, the coefficient $= \lfloor (m - 2) + \alpha \rfloor = m - 2 = \left\lfloor \frac{t - \tau_{i-1}}{MT_i} \right\rfloor$ and S_i has experienced the m -1th arrival. Therefore, for $1 \leq SA_i(t) \leq K1$, $SA_i(t) = 1 + \left\lfloor \frac{t - \tau_{i-1}}{MT_i} \right\rfloor$. Based on the definition of arrival and departure of entities from server S_1 one can see that because of the relationship

$SD_i(t) = SA_i(t + T_i)$ (3.3)

that means (3.2) holds. ■

3.1.3 Deriving the cumulative arrivals and cumulative departures for buffer B_i

Corollary 1: For the sequential system described in Figure 1-6 the cumulative arrivals and cumulative departures at buffer B_i at time t are:

$$BA_i(t) = \begin{cases} \min\left\{ K1, 1 + \left\lfloor \frac{t - \tau_{i-1}}{MT_{i-1}} \right\rfloor \right\} & \text{if } t \geq \tau_{i-1} \\ 0 & \text{Otherwise} \end{cases} \qquad (3.4)$$

$$BD_i(t) = \begin{cases} \min\left\{ K1, 1 + \left\lfloor \frac{t - \tau_{i-1}}{MT_i} \right\rfloor \right\} & \text{if } t \geq \tau_{i-1} \\ 0 & \text{Otherwise} \end{cases} \qquad (3.5)$$

Proof: For any B_i where i = 2, 3…N+1, the number of arrivals at B_i is equal to the number of departures from S_{i-1} at a given time t. Therefore taking (3.2) from the perspective of S_{i-1} and applying the condition (3.6) that proves (3.4).

$BA_i(t) = SD_{i-1}(t)$ (3.6)

The number of departures from B_i is equal to the number of arrivals at server S_i. Therefore, taking (3.1) from the perspective of arrivals at S_i and applying the condition (3.7) proves (3.5).

$BD_i(t) = SA_i(t)$ (3.7) ■

3.1.4 Deriving the maximum number of entities buffer B_i will experience

Corollary 2: For the sequential system, the maximum number of entities that buffer B_i will experience given starting inventory K1 as shown in Figure 3-7 is:

$MB_i = (K1 - 1) - \lfloor Y_i*(K1-1) \rfloor$ (3.8)

Where $Y_i = \frac{MT_{i-1}}{MT_i}$ for i = 2…N.

When $T_i > MT_{i-1}$, then $MT_{i-1} < MT_i$ and $Y_i < 1$. When $T_i \leq MT_{i-1}$, then $MT_i = MT_{i-1}$ and $Y_i = 1$.

When $Y_i = 1$, $MB_i = 0$, thus a buffer size = 1 is required for transport only to the next process, we call this a transport buffer. We prove this for MB_i, i = 2,3,…N and when $Y_i < 1$. For $Y_i = 1$, as mentioned before, is a transport buffer with $MB_i = 1$.

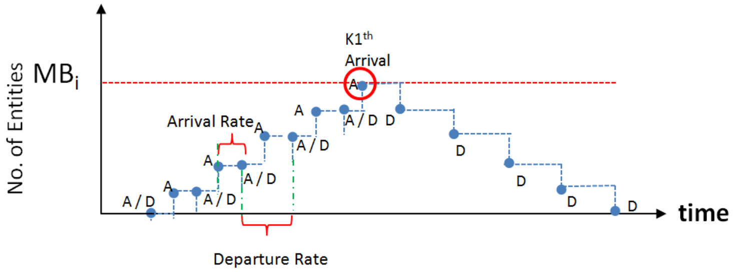


Figure 3-7. Maximum number of entities buffer B_i experiences given inventory K1

For B_i of interest, at any given time t the number of entities is equal to:

$$B_i(t) = BA_i(t) - BD_i(t),\ B_i(t) \geq 0 \quad (3.9)$$

We first find the time (say T) when $B_i(t)$ reaches its maximum level and then plug in T to (3.9), therefore calculating MB_i. By assumption, the first departure from S_{i-1} and the first arrival to S_i happen simultaneously. After this event, because $MT_i = T_i > MT_{i-1}$ and using the relationship that the frequency of arrivals to Server S_i is $\frac{1}{MT_i}$, the departure rate from S_{i-1} will be greater than the arrival rate to S_i. This causes the accumulation in B_i to increase until the last departure

(K1th departure) from *S_{i-1}* occurs. Therefore the maximum accumulation happens when the last departure from *S_{i-1}* occurs.

$T = MT_{i-1}*(K1-1) + \tau_{i-1}$ (3.10)

Plugging in T from (3.10) into (3.9) and using the results of Corollary 1 we have:

$$B_i(t) = BA_i(t) - BD_i(t) = \min\left(K1, 1+ \left\lfloor \frac{t - \tau_{i-1}}{MT_{i-1}} \right\rfloor\right) - \min\left(K1, 1 + \left\lfloor \frac{t - \tau_{i-1}}{MT_i} \right\rfloor\right)$$

$$= MB_i = \left\lfloor \frac{MT_{i-1}*(K1-1)}{MT_{i-1}} \right\rfloor - \left\lfloor \frac{MT_{i-1}*(K1-1)}{MT_i} \right\rfloor = (K1 - 1) - \lfloor Y_i*(K1-1) \rfloor \qquad \blacksquare$$

3.2 Model II: N server, N+1 sequential line with reduced capacity for one buffer

This model is synonymous with the first model in Figure 1-6. After calculating MB_i for each *B_i* in Model 1, one may discover that for a specific buffer, *B_v*, MB_v is too large for the workspace and would like to identify the impact to other buffers and servers in the serial line when the size of *B_v* is reduced to a value of L_v where $L_v < MB_v$. This model assesses the impact of this capacity reduction on each buffer and server in the serial line at the time of interest t.

In Figure 3-8, we show the typical behavior of *B_v*. In this figure "A" represents an arrival and "D" represents a departure. The arrival rate and departure rate of *B_v* are not impacted until L_v is reached (at the R1th arrival). After L_v is reached, the arrival rate follows the departure rate until K1 units arrive. This is because when *B_v* is blocked, it prevents the entity in its prior server (*S_{v-1}*) from departing. Thus, there is no space made available in *B_v* until after its subsequent server (*S_v*) has a departure. Only after *S_v* has a departure, can another entity arrive to *S_v* creating a departure from *B_v* so that *B_v* can receive an arrival. Therefore, after *B_v* is blocked, arrivals to *B_v* occur at the same time as its departures. We define this phase as *lock step*. After K1 units arrive to *B_v* (end of lock step), only departures occur.

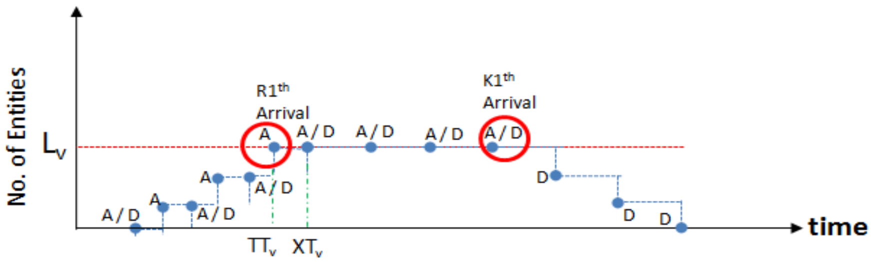


Figure 3-8. Buffer behavior when B_v capacity set to L_v

Notations:

N7) B_v is the buffer that is too large given the workspace constraints and requires capacity reduction.

N8) L_v is the reduced size of the buffer B_v.

N9) R1 is the number of arrivals that the buffer experiences when the buffer reaches a size of L_v

N10) TT_i is the transition time of B_i. It is the last time an entity arrives according to the prior arrival rate before getting blocked. We notice that TT_i has been defined in a generic way as other buffers might be blocked because of capacity reduction of B_v.

N11) XT_v is the block time of B_v. It is the first time an entity arrives at the departure rate of the blocked buffer.

Definitions:

D5) $BR_v = MB_v - L_v$, the amount that B_v is reduced, $BR_v > 0$

D6) $EC_i = B_i$ Size – MB_i, is the amount the buffer size exceeds MB_i. Given B_v is the only reduced buffer (below MB_v), $EC_i \geq 0$

D7) $\sum_i^{v-1} EC_i$= The excess capacity available from B_i of interest to B_{v-1}

Assumptions:

A9) This model applies to buffers B_i (where $Y_i < 1$) and excludes transport buffers (where $Y_i = 1$).

A10) Each B_i has its capacity set to at least MB_i and one only buffer (B_v) has its capacity set to a value (L_v) less than MB_v. (This relaxes the second assumption in the first model).

We first discuss a set of rules for which buffers and servers in the serial line are impacted or not impacted as a result of reducing the size of B_v. Then we will derive the arrival and departure formulas for each impacted buffer or server. For any B_i or S_i, Figure 3-9 below shows the rules for what equation for arrivals and departures to use based on whether i = v, i > v or i < v. For each condition, if for a specific B_i or S_i a reference is made to use an equation in Model 1, that indicates that the server or buffer is not impacted by the reduced size of B_v. For those B_i and S_i that are impacted by reduced capacity of B_v, those equation references are derived in this section.

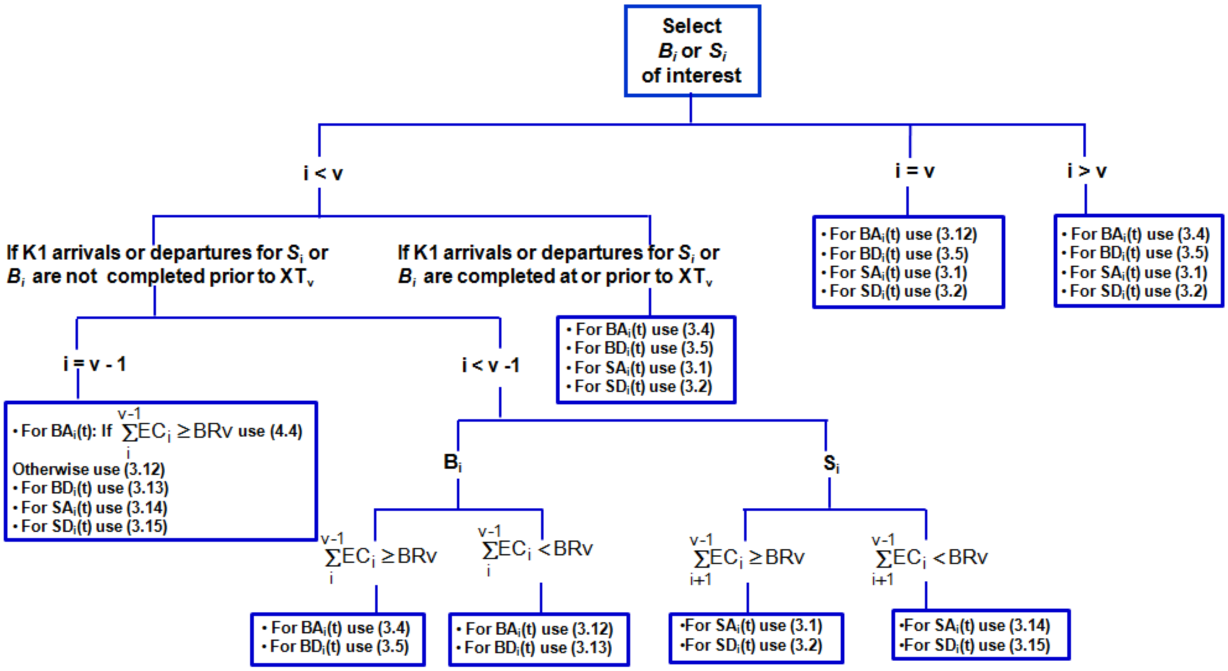

Select B_i or S_i of interest
i < v
i = v
i > v
• For $BA_i(t)$ use (3.12)
• For $BD_i(t)$ use (3.5)
• For $SA_i(t)$ use (3.1)
• For $SD_i(t)$ use (3.2)
• For $BA_i(t)$ use (3.4)
• For $BD_i(t)$ use (3.5)
• For $SA_i(t)$ use (3.1)
• For $SD_i(t)$ use (3.2)
If K1 arrivals or departures for S_i or B_i are not completed prior to XT_v
If K1 arrivals or departures for S_i or B_i are completed at or prior to XT_v
• For $BA_i(t)$ use (3.4)
• For $BD_i(t)$ use (3.5)
• For $SA_i(t)$ use (3.1)
• For $SD_i(t)$ use (3.2)
i = v - 1
i < v -1
• For $BA_i(t)$: If $\sum_{i}^{v-1} EC_i \geq BRv$ use (4.4)
Otherwise use (3.12)
• For $BD_i(t)$ use (3.13)
• For $SA_i(t)$ use (3.14)
• For $SD_i(t)$ use (3.15)
B_i
S_i
$\sum_{i}^{v-1} EC_i \geq BRv$
$\sum_{i}^{v-1} EC_i < BRv$
$\sum_{i+1}^{v-1} EC_i \geq BRv$
$\sum_{i+1}^{v-1} EC_i < BRv$
• For $BA_i(t)$ use (3.4)
• For $BD_i(t)$ use (3.5)
• For $BA_i(t)$ use (3.12)
• For $BD_i(t)$ use (3.13)
•For $SA_i(t)$ use (3.1)
•For $SD_i(t)$ use (3.2)
•For $SA_i(t)$ use (3.14)
•For $SD_i(t)$ use (3.15)

Figure 3-9. Rules for determining impacted arrivals and departures for B_i and S_i

3.2.1 Deriving arrival and departure formulas for impacted buffers and servers:

Given L_v, we solve for R1, TT_v and XT_v for B_v and ultimately $BD_v(t)$ and $BA_v(t)$. We then calculate the TT_i and XT_i for all other servers S_i and buffers B_i impacted. We then derive the number of buffer arrivals and departures and the number of server arrivals and departures with respect to time $BA_i(t)$, $BD_i(t)$, $SA_i(t)$ and $SD_i(t)$.

We use (3.8) and find R1, TT_v and XT_v for B_v. As shown in Figure 3-8, up to L_v, the buffer departures and arrivals show the exact behavior as in Model 1. The number of arrivals up to L_v is R1, a value less than K1. At $t = TT_v$, entity R1 arrives to B_v. Using (3.10) where time was defined for $BA_i(t)$ from 1 through K1th arrival, we substitute R1 for K1 for B_v.

TT_v (for R1 arrivals) $= MT_{v-1}*(R1-1) + \tau_{v-1}$ (3.11)

Plugging in T from (3.11) into (3.9) and using the results of Corollary 1 we have:

$$B_i(t) = L_v = BA_i(TT_v) - BD_i(TT_v) = 1 + \left\lfloor \frac{TT_v - \tau_{v-1}}{MT_{v-1}} \right\rfloor - 1 - \left\lfloor \frac{TT_v - \tau_{v-1}}{MT_v} \right\rfloor =$$

$$\left\lfloor \frac{MT_{v-1} * (R1-1)}{MT_{v-1}} \right\rfloor - \left\lfloor \frac{MT_{v-1} * (R1-1)}{MT_v} \right\rfloor = (R1 - 1) - \lfloor Y_v * (R1-1) \rfloor$$

We solve the above equation for R1 and then we solve for the transition time TT_v when the R1th arrival occurs using (3.11).

In Figure 3-8, when the buffer's limited capacity is reached, the arrivals are in lock step with departures for B_v. In order to calculate the XT_v, we must find the last departure that occurred by $t = TT_v$. The very next departure after TT_v will be the departure where the arrival is in lock step with the departure. That departure will occur at XT_v. We solve for $BD_v(t)$ at $t = TT_v$ via $BD_v(TT_v) = 1 + \left\lfloor \frac{TT_v - \tau_{v-1}}{MT_v} \right\rfloor$

The next arrival and departure $(BD_v(TT_v) + 1)$ will be in lock step with the $BD_v(t)$ departure profile. Because departures are not impacted by setting of L_v, we can solve for XT_v using the time defined for the first through K1th departure from $BD_v(t)$. We use (3.7) of Corollary 1 to derive $BD_i(t)$. Figure 3-10 shows the time of each departure from buffer B_i from 1 through K1.

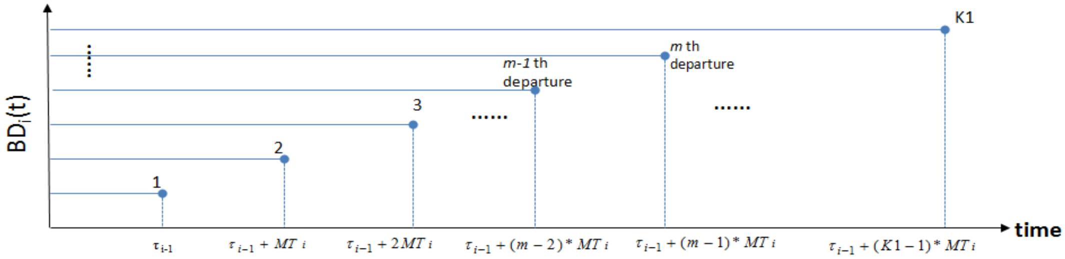


Figure 3-10. Time defined for $BD_i(t)$ from 1 through K1th departure

We substitute $BD_v(TT_v)$ +1 for m and solve for XT_v

$XT_v = MT_v*((BD_v(TT_v)+1)-1) + \tau_{v-1}$

Because of the relationships (3.6) and (3.7) in Corollary 1, the TT_v and XT_v become critical times propagated through to every buffer B_i prior to B_v. At the time a buffer is blocked, this prevents the entity in the prior server from departing, preventing an arrival in the server. Only after the server ahead of the blocked buffer has a departure, can the blocked buffer entity depart and accept another entity. Thus, the profile of each buffer B_i prior to B_v shifts to meet the departing profile of B_v and the departing profile equals the arrival profile of Server S_v as in (3.7). Therefore, the block time XT_v of B_v is the same for every B_i prior to B_v. For TT_i, the time TT_i relative to XT_v may vary for each B_i prior to B_v. That is, the time from TT_i to the next arrival where the arrival becomes in lock step with the departure varies due to the differences in service times. To calculate the TT_i for each B_i before B_v, we use the TT_v from B_v and calculate the number of arrivals using the $BA_i(t)$ in (3.4).

$$BA_i(TT_v) = 1 + \left\lfloor \frac{TT_v - \tau_{i-1}}{MT_{i-1}} \right\rfloor$$

Then we solve for TT_i at the next arrival which will be the TT_i for the B_i of interest and substituting $BA_i(TT_v)$+1 for m (see (3.10)).

$$TT_i = ((BA_i(TT_v)+1)-1)*MT_{i-1} + \tau_{i-1}$$

3.2.2 Deriving arrival and departure formulas for Impacted Buffers and Servers

Now we can derive the formula for the arrivals and departures for buffers and servers that are impacted by the reduced capacity of B_v as shown in Figure 3-9.

- B_i of interest is B_v (i = v): The arrivals of B_v are impacted and use (3.12).
- B_i of interest is B_{v-1} (where i = v-1) and if K1 arrivals and departures are not already completed before XT_v, the departure formula in (3.13) always applies to impacted buffer B_{v-1}. The arrival formula (3.12) applies when condition $\sum_i^{v-1} EC_i < BR_v$ is met.
- B_i of interest where i < v-1 and if K1 arrivals and departures are not already completed before XT_v, (3.12) and (3.13) apply when condition $\sum_i^{v-1} EC_i < BR_v$ is met.

In general, when $t < \tau_{i-1}$, no arrivals or departures have occurred to B_i. If $\tau_{i-1} \leq t \leq TT_i$, then equations (3.4) and (3.5) hold but within new intervals of interest as shown in (3.12) and (3.13). If the time of interest t for B_i is greater than TT_i and less than XT_v, (3.4) and (3.5) hold, however $t = TT_i$. As shown in Figure 3-8, there are no arrivals or departures in between the transition time and the block time. For the interval of $t \geq XT_v$, $BA_i(t)$ remains the same for entities prior to XT_v, however after XT_v, the $BA_i(t)$ and $BD_i(t)$ assumes the $BD_v(t)$ profile and thus the use of MT_v.

$$BA_i(t)= \begin{cases} 0 & \text{if } t < \tau_{i-1} \\ \min\left\{ K1, 1+\left\lfloor \frac{t-\tau_{i-1}}{MT_{i-1}} \right\rfloor \right\} & \text{if } \tau_{i-1} \leq t \leq TT_i \\ \min\left\{ K1, 1+\left\lfloor \frac{TT_i-\tau_{i-1}}{MT_{i-1}} \right\rfloor \right\} & \text{if } TT_i < t < XT_v \\ \min\left\{ K1, 1+\left\lfloor \frac{TT_i-\tau_{i-1}}{MT_{i-1}} \right\rfloor + \left\lfloor \frac{t-XT_v}{MT_v} \right\rfloor \right\} & \text{if } t \geq XT_v \end{cases} \quad (3.12)$$

$$BD_i(t)= \begin{cases} 0 & \text{if } t < \tau_{i-1} \\ \min\left\{ K1, 1+\left\lfloor \frac{t-\tau_{i-1}}{MT_i} \right\rfloor \right\} & \text{if } \tau_{i-1} \leq t \leq TT_i \\ \min\left\{ K1, 1+\left\lfloor \frac{TT_i-\tau_{i-1}}{MT_i} \right\rfloor \right\} & \text{if } TT_i < t < XT_v \\ \min\left\{ K1, 1+\left\lfloor \frac{TT_i-\tau_{i-1}}{MT_i} \right\rfloor + \left\lfloor \frac{t-XT_v}{MT_v} \right\rfloor \right\} & \text{if } t \geq XT_v \end{cases} \quad (3.13)$$

We have completed the arrival and departure profiles for buffers and proceed to the servers S_i.

- S_i of interest is S_{v-1} (where i = v-1) and if K1 arrivals and departures are not already completed before XT_v: The formulas (3.14) and (3.15) always apply
- S_i of interest where i < v-1and if K1 arrivals and departures are not already completed before X_{Tv}: (3.14) and (3.15) apply when condition $\sum_{i+1}^{v-1} EC_i < BR_v$ is met

From (3.6) and (3.7) in Corollary 1, no new calculations are needed for the servers. The TT_i for server S_i is equivalent to the TT_i calculated for the buffer ahead of server S_i or TT_{i+1}. As a result, one derives an equivalent set of server arrival and departure models shown in (3.14) and (3.15).

$$SA_i(t)= \begin{cases} 0 & \text{if } t < \tau_{i-1} \\ \min\left\{ K1, 1+\left\lfloor \frac{t-\tau_{i-1}}{MT_i} \right\rfloor \right\} & \text{if } \tau_{i-1} \le t \le TT_{i+1} \\ \min\left\{ K1, 1+\left\lfloor \frac{TT_{i+1}-\tau_{i-1}}{MT_i} \right\rfloor \right\} & \text{if } TT_{i+1} < t < XT_v \\ \min\left\{ K1, 1+\left\lfloor \frac{TT_{i+1}-\tau_{i-1}}{MT_i} \right\rfloor + \left\lfloor \frac{t-XT_v}{MT_v} \right\rfloor \right\} & \text{if } t \ge XT_v \end{cases} \quad (3.14)$$

$$SD_i(t)= \begin{cases} 0 & \text{if } t < \tau_{i} \\ \min\left\{ K1, 1+\left\lfloor \frac{t-\tau_{i}}{MT_i} \right\rfloor \right\} & \text{if } \tau_{i} \le t \le TT_{i+1} \\ \min\left\{ K1, 1+\left\lfloor \frac{TT_{i+1}-\tau_{i}}{MT_i} \right\rfloor \right\} & \text{if } TT_{i+1} < t < XT_v \\ \min\left\{ K1, 1+\left\lfloor \frac{TT_{i+1}-\tau_{i}}{MT_i} \right\rfloor + \left\lfloor \frac{t-XT_v}{MT_v} \right\rfloor \right\} & \text{if } t \ge XT_v \end{cases} \quad (3.15)$$

In Chapter 5, we will apply the results to parameters (1) through (8) identified in the Introduction.

4. OPTIMIZATION FRAMEWORK FOR BUFFER CLUSTERING POLICY

In this chapter, we derive the buffer clustering optimization framework utilizing the model from Chapter 3 to provide the buffer cluster sizing and an optimal buffer clustering policy. Before deriving the aforementioned relationships, we list the notations, assumptions and definitions. We use Figure 4-1 which shows the inventory profile in buffer B_i to illustrate some of the notations $K2_i$, $K3_i$, $\bar{t}_{1Bi}$, $\bar{t}_{2Bi}$ and MB_i -1.

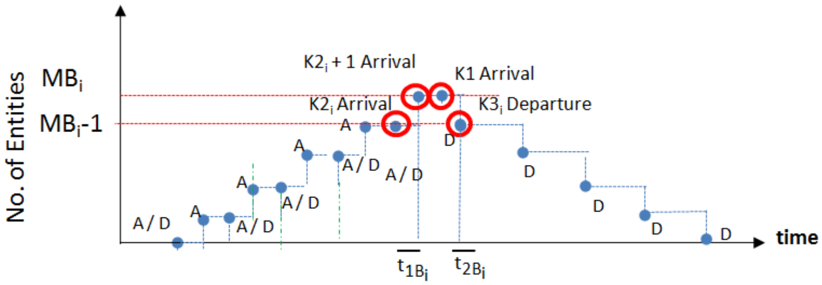


Figure 4-1. Illustration of $K2_i$, $K3_i$, $\bar{t}_{1Bi}$, $\bar{t}_{2Bi}$ and MB_i -1

Notations:

N12) W_j is a set of one or more buffers which we refer to as a buffer cluster.

N13) BB_j is the maximum number of entities a buffer cluster W_j must be able to hold to ensure that no overflows or underflows occur in the buffer cluster.

N14) X_j is a binary variable {0,1} and defines whether cluster W_j must be realized {1} or not {0} (i.e. X_j = 1 determines that cluster W_j must be selected as a part of the buffer cluster policy).

N15) $K2_i$ is the last arrival to buffer B_i that occurs when the number of entities in B_i is MB_i -1

N16) $K3_i$ is the number of entities that depart buffer B_i changing the number of entities in buffer B_i to MB_{i-1} from MB_i. This occurs right after the last arrival (K1 arrival) to buffer B_i

N17) $\bar{t}_{1Bi}$ is the time of the first arrival changing the number of entities in buffer B_i to MB_i from MB_{i-1} ($K2_i$ + 1 arrival).

N18) $\bar{t}_{2Bi}$ is the time of the first departure ($K3_i$) departing after the last arrival (K1 arrival) for buffer B_i. It is the last time the number of entities in buffer B_i equals MB_i.

N19) q is the time the last entity departs from server S_N to buffer B_{N+1}

N20) H is the size each entity occupies within a cell of a buffer in square meters

N21) G is the maximum size of a buffer cluster W_j in square meters.

*Assumptions: (*Assumptions A1 through A8 from Chapter 3 hold)

A11) Possible combinations (clusters) of buffers are given.

A12) A buffer B_i must be either a dedicated buffer or in a single cluster; thus, if {2, 3, …,*i*,…,*N*} denotes the index set of intermediate buffers B_i and {1, 2, …*j*, …, *C*} the index set of buffer clusters W_j, then

$$\bigcup_{j=1}^{C} W_j=\{2,3,\dots,N\}$$

Although a buffer cluster must maintain the sequence of operations, meaning it must facilitate an entity to move in the sequence of operations from servers S_1, S_2, S_3… *to* S_N the buffers included in a cluster should not necessarily be sequential. Therefore, a buffer cluster may include non-sequential buffers and still maintain the sequence of operations. The example in Figure 4-2 shows non-sequential buffers B_2 and B_5 clustered (W_1) and sequential buffers B_3 and B_4 clustered (W_2) while still maintaining the sequence of operations S_1 through S_5 as shown by the arrows.

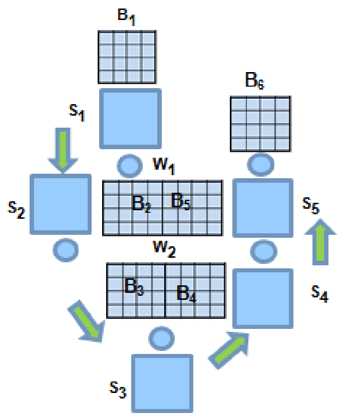


Figure 4-2. Non-sequential and sequential clusters maintaining operation sequence

Equation (3.8) in Chapter 3 provides us with the maximum number of entities that buffer B_i will experience over a given demand K1 within a production shift and is used for sizing the standalone buffers to ensure that no overflows or underflows occur. We need to understand the size required for the buffer cluster combinations at any given time. Thus, for the buffer cluster, we must assess the buffer sizes leveraging (3.9) at every given time t from the time of first arrival of an entity in buffer B_2 (τ_1) to the completion of the production shift (q) (See assumption A6).

$$B_i(t)=\min\left(K1, 1+\left\lfloor\frac{t-\tau_{i-1}}{MT_{i-1}}\right\rfloor\right)-\min\left(K1, 1-\left\lfloor\frac{t-\tau_{i-1}}{MT_i}\right\rfloor\right) \quad \forall\ t\in[\tau_{i-1},q] \qquad (4.1)$$

From (4.1) we see that the search for the buffer cluster size requires a calculation at each time step throughout the production shift for every buffer B_i. This results in a significant number of computations. We quantify and illustrate in Chapter 5 how the number of calculations and the simulation time can be substantially reduced. Before we can quantify the savings, we must first solve for BB_j defined as the maximum number of entities the buffer cluster W_j must hold such that no overflows or underflows occur in the buffer cluster.

To solve for BB_j, we first solve $\bar{t}_{1Bi}$ and $\bar{t}_{2Bi}$ (see Figure 4-1) and then prove that the maximum buffer cluster size required occurs when one of the individual buffers B_i is at a maximum during the time interval $t \in [\bar{t}_{1Bi}, \bar{t}_{2Bi}]$. To solve for $\bar{t}_{1Bi}$ we first solve for the last arrival ($K2_i$) that occurs at MB_i - 1. We modify (3.8) as shown in (4.2).

$MB_i - 1 = (K2_i - 1) - \lfloor Y_i * (K2_i - 1) \rfloor$ (4.2)

Then to get the first arrival that occurs at MB_i, we solve for the time $\bar{t}_{1Bi}$ that the $K2_i$ + 1 arrival arrives to buffer B_i

$\bar{t}_{1Bi} = \tau_{i-1} + ((K2_i + 1) - 1) * MT_{i-1}$ (4.3)

To determine $\bar{t}_{2Bi}$, we first we solve for the time of the K1th arrival to buffer B_i using (3.10) and use (3.5) to calculate the number of departures that have occurred by the K1th arrival. Then we add one to the departures (named $K3_i$) and calculate the time of this departure ($\bar{t}_{2Bi}$) using (4.4).

$\bar{t}_{2Bi} = \tau_{i-1} + (K3_i - 1) * MT_i$ (4.4)

The minimum size for a cluster W_j such that no overflows occur is shown in 4.5.

$BB_j = \max_{t \,\epsilon\, \cup_{i=1}^{N} [\bar{t}_{1Bi}, \; \bar{t}_{2Bi}]} \left\{ \sum_{B_r \,\epsilon\, W_j} B_r(t) \right\} \quad \forall\, W_j$ (4.5)

We now prove that the buffer cluster size required occurs when at least one of the individual buffers in the cluster W_j is at a maximum according to (4.5).

Lemma 2: Minimum size for cluster W_j such that no overflows occur takes place when at least one of the buffers B_i in cluster W_j has reached the maximum number of entities, MB_i.

We take buffers B_k and B_p that are in cluster W_j and output to servers S_k and S_p respectively where k < p, and the constraints $MT_{k-1} < MT_k$ and $MT_{p-1} < MT_p$ hold. We recall from Corollary 2 that when $MT_{k-1} = MT_k$ and $MT_{p-1} = MT_p$, the buffer size is 1. We observe the buffer inventory profiles of B_k and B_p at three specific time intervals of the buffer inventory covering the time from the first arrival to buffer B_k to the last departure from buffer B_p as shown in Figure 4-3. For each interval, we also observe when buffers B_k and B_p are sequential (p = k+1) and when they are not (p > k +1). Figure 4-3 shows the case when p = k+1. As p > k+1, the buffer profiles drift apart and the overlap in Time Interval 3 decreases until no overlap exists.

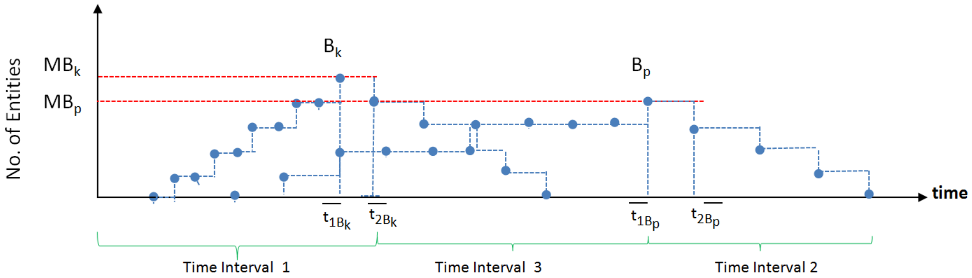


Figure 4-3. Buffer profiles of B_k and B_p and time intervals 1 through 3 for p = k + 1

<u>Time Interval 1</u>: $\tau_{k-1} < t \leq \overline{t}_{2\,Bk}$ (B_k and B_p are increasing; B_k has reached a maximum)

Buffer B_k has its first arrival at τ_{k-1} and increases until it reaches MB_k. Buffer B_p has its first arrival at τ_{p-1} and increases until it reaches its maximum, MB_p. For Buffers B_k and B_p where p = k+1, $\tau_{p-1} - \tau_{k-1} = T_k$. Thus, buffer B_p starts increasing T_k seconds after the first arrival to buffer B_k. When p > k+1, $\tau_{p-1} - \tau_{k-1}$ is greater than T_k meaning that time of the first arrival of B_p approaches and can exceed time interval $[\overline{t}_{1Bk}, \overline{t}_{2Bk}]$ when B_k is maximum. When buffer B_k is at its maximum (MB_k), buffer B_p is increasing in size, while after reaching MB_k, buffer B_k begins to decline. Therefore, a possible buffer cluster maximum between buffers B_k and B_p is at MB_k.

<u>Time Interval 2</u>: $\overline{t}_{1\,Bp} \leq t \leq \tau_{p-1} + (K1-1)*MT_p$ (B_k is decreasing and reaches zero; B_p has reached its maximum and begins to decline including its last departure)

The last arrival to buffer B_p at time $t_{ApK1} = \tau_{p-1} + (K1-1)*MT_{p-1}$

The last departure of buffer B_k occurs at time $t_{DkK1} = \tau_{k-1} + (K1-1)*MT_k$

When we subtract these times we get:

$$\tau_{p-1} + (K1-1)*MT_{p-1} - \tau_{k-1} - (K1-1)*MT_k = \tau_{p-1} - \tau_{k-1} + (K1-1)*MT_{p-1} - (K1-1)*MT_k =$$

$$\tau_{p-1} - \tau_{k-1} - (K1-1)*(MT_{p-1} - MT_k) \qquad (4.6)$$

For Buffers B_k and B_p where p = k+1 then $MT_{p-1} - MT_k = 0$ and $\tau_{p-1} - \tau_{k-1} = T_k$ then (4.6) equals T_k. Therefore, the last departure of B_k occurs T_k seconds prior to the last arrival to buffer B_p. Thus B_k is reaches zero while B_p is at a maximum. When p > k+1, then $\tau_{p-1} - \tau_{k-1}$ is greater than T_k and $MT_k \leq MT_{p-1}$, therefore $(K1-1)*(MT_{p-1} - MT_k) \geq 0$, resulting in (4.6) being greater than T_k. Therefore, a possible buffer cluster maximum between buffers B_k and B_p occurs at MB_p.

Time Interval 3: $\bar{t}_{2\,Bk} < t < \bar{t}_{1\,Bp}$ (B_k is decreasing; B_p is increasing but has not reached a maximum)

At time $\bar{t}_{2\,Bk}$, B_k has experienced its last arrival (K1) and it has reached a maximum MB_k. Therefore, after this time, only departures occur. Thus in essence, MB_k indicates the number of departures that are left for buffer B_k until it reaches zero. During this time interval, buffer B_p is increasing (it hasn't reached a maximum yet), meaning it has both arrivals and departures. When p = k+1, $MT_k = MT_{p-1}$ indicating that the number of departures remaining at buffer B_k, MB_k, is also the number of entities still to arrive at buffer B_p and they occur at the same time. However, given that B_p is increasing and hasn't reached a maximum, it is also experiencing departures at a rate of MT_p. During this time interval, the quantity of inventory of buffer B_k declines from MB_k to zero. Although Buffer B_p entities arrive at the same rate as buffer B_k departures, its inventory increases more slowly than the decline of departures from buffer B_k given buffer B_p also has entities departing at a rate of MT_p during this time interval. Therefore the sum of the inventory profiles of buffer B_k and B_p during this time interval will not exceed the maximum inventory observed in time interval (1) or (2) described above. When p > k + 1, buffer B_p has its first entity arriving even later than in the p = k+1 case. Although the decline of buffer B_k remains the same, buffer B_p starting arrival approaches the time when buffer B_k

approaches MB_k and the summation of the two inventory profiles will not exceed the maximum inventory observed in time interval (1) or (2) described above. Based on results of the analysis for each of the time intervals, we must search the union of time intervals in (4.7) for each buffer to find the maximum buffer cluster size BB_j.

$$t \in [\bar{t}_{1 B1}, \bar{t}_{2 B1}] \cup [\bar{t}_{1B2}, \bar{t}_{2 B2}] \cup \ldots [\bar{t}_{1Bi}, \bar{t}_{2 Bi}] \cup \ldots [\bar{t}_{1B_N}, \bar{t}_{2 B_N}] \quad (4.7)$$ ■

For a given production line, there can be thousands of buffer cluster combinations. Consider a collection of candidate buffer clusters W_j, $j = 1, \ldots, C$, not necessarily disjoint, for which the minimum storage requirements BB_j have been computed via (4.5). We use integer programming to find the buffer cluster combination that provides the minimum total space occupied by all clusters. From here we set the objective function as shown in (4.8) to determine the buffer cluster(s) size across the production line where X_j determines what buffer cluster combinations should be present to minimize the overall size of the buffer cluster(s) throughout the production line where $X_j \in \{0, 1\}$ is a decision variable that determines whether the buffer cluster W_j should be realized or not. As defined earlier each buffer can only participate in one and only one realized buffer cluster. The first set of constraints (4.9) show that a buffer B_i can only participate in one combination buffer cluster. The second set of constraints (4.10) is the maximum size of a buffer cluster in square meters. The third set of constraints (4.11) are the binary constraints for X_j. We define V_j to ensure constraint (4.9) applies only to buffers within cluster W_j.

$V_j = \{ i, B_i \epsilon W_j \} \quad 1 \leq j \leq C$

Objective Function:

Minimize $\sum_{j=1}^{C} BB_j X_j$ (4.8)

Subject to:

$\sum_{j:i \in V_j} X_j = 1, \; 1 \leq i \leq N$ (4.9)

$BB_j * H * X_j \leq G, \; 1 \leq j \leq C$ (4.10)

$X_j = 0 \text{ or } 1 \; \forall \, j$ (4.11)

5. APPLYING MODELS TO INDUSTRY EXAMPLES

Returning to the distribution and manufacturing examples discussed in the introduction and we illustrate how the above formulations from Chapters 3 and 4 are applied to each problem class. These methods are used and compared to results from simulation and found to be exact.

5.1 Applying Chapter 3 results to distribution center example

As discussed in the introduction, there were problem areas the distribution center team desired to investigate as part of the research. Below we illustrate how the model in Chapter 3 is applied to these problem areas.

(1) The maximum buffer size allowed such that no buffer experiences blocking during a shift.

To determine the maximum buffer size allowed such that no buffer experiences blocking during a shift where the demand K1 = 350, we use (3.8).

$MB_2 = (K1 - 1) - \lfloor Y_2 * (K1-1) \rfloor = 349 - \lfloor \frac{40}{240} * 349 \rfloor = 291$ Where $Y_2 = \frac{MT_1}{MT_2} = \frac{40}{240}$

Similarly, $MB_3 = 0$, $MB_4 = 188$, $MB_5 = 0$, $MB_6 = 0$, $MB_7 = 66$, $MB_8 = 0$, $MB_9 = 0$

(2) The shift where a buffer's capacity should change to meet the production demand changes.

Because the service times remain the same, increasing K1, only impacts those buffers where $MT_i \neq MT_{i-1}$. Buffers MB_2, MB_4 and MB_7 undergo a buffer size increase with an increase in demand. For example, when demand increases to K1=400 the capacities for buffers B_2, B_4 and B_7 must be increased to 333, 215, and 75 respectively.

(3) The number of units processed by the bottleneck station when time of blocking occurs.

Suppose that the Flash Process (S_4) and Buffer B_4 cannot support the required capacity (188) and has space available for only 160 units. The number of units that have arrived at B_4 when L_4 = 160 entities is R1.

$L_v = (R1 - 1) - \lfloor Y_v * (R1-1) \rfloor \rightarrow 160 = (R1 - 1) - \lfloor \frac{240}{520} * (R1-1) \rfloor \rightarrow R1 = 298$

Therefore, 298 units have arrived when B_4 reaches capacity of L_4 and 102 units remain to be processed to fill the K1 = 400 demand in (2).

(4) The buffer transition and block time when a buffer size is reduced ($L_v < MB_v$).

We determine the transition and block time when we reduce the capacity of Buffer B_4 to L_v =160 units. We first calculate the transition time, or the last time an entity arrives to B_4 according to the prior arrival rate before getting blocked.

TT_4 (for R1 arrivals) = $MT_3*(R1-1) + \tau_3 = 240*(298-1) + 340 = 71620$ seconds

The very next departure after TT_4 will be the departure where the arrival is in lock step with the departure. We solve for $BD_4(t)$ at $t = TT_4$

$$BD_4(TT_4) = 1 + \left\lfloor \frac{TT_4-\tau_3}{MT_4} \right\rfloor \rightarrow BD_4(TT_4) = 1 + \left\lfloor \frac{71620-340}{520} \right\rfloor = 138$$

The $BD_4(TT_4)$ departure will occur at XT_4. Therefore, we solve for XT_4 as follows:

$XT_4 = MT_4*((BD_4(TT_4)+1)-1) + \tau_3 = 520*138 + 340 = 72100$ seconds

(5) The reaction time to route a resource (operator or workstation) to a bottleneck process to prevent impact to other processes prior to the bottleneck.

The distribution center production team determined that the time to boot up a flash station in the Flash Process is t_{FB} = 310s. Therefore, to prevent the blocking at time t = 72100 seconds calculated in (4), an additional flash station would need to be booted by time $t = TT_4 – t_{FB}$ = 71620 – 310 = 71310 s so that blocking can be alleviated for the remaining 102 entities. We use time $t = TT_4$ as that is the last time an entity arrives to B_4 according to the prior arrival rate, and we ensure that the second flash station is available by that time to prevent blocking at time $t = XT_4$.

(6) The maximum demand the serial line can support given the limited shift time and buffer sizes.

We use (3.2) to find the time of the last departure from the ninth server S_9 in serial line and the fact that there are 43200 seconds in a 12-hour shift.

$SD_9 = 1 + \left\lfloor \frac{43200\text{-}2280}{640} \right\rfloor = 64$ entities depart $\rightarrow \frac{350}{64} = 5.46$ shifts required to finish 350 entities.

(7) The shift time to finish a given demand with limited buffer sizes

All 350 entities will complete at time t using (3.2): $350 = 1 + \left\lfloor \frac{t - 2280}{640} \right\rfloor$ where t = 225640 seconds to complete all 350 entities. Figure 5-1 uses (3.2) to profile for shift completion time vs. demand.

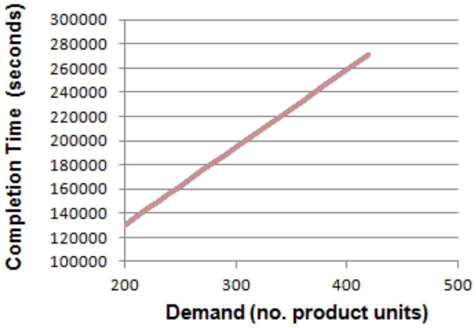


Figure 5-1. Shift completion vs. demand

(8) Determining the production line behavior with varying process times.

We can easily apply these methods to scenarios where service times vary. For a given buffer, we identified that MT_{i-1} and MT_i are the critical parameters for identifying the maximum number of entities a buffer will experience. We calculated MB_4 as 188 for the buffer prior to the Flash process. The Flash process ($T_4 = MT_i = MT_4$) is fully automated for the mobile devices. The Pre-work process ($T_2 = MT_{i-1} = MT_2$) is also fully automated. However, if the Pre-Flash process (the process in between MT_{i-1} and MT_i) is not fully automated, say it is a manual process and has variable service times depending on the operator skill, Figure 5-2 shows that the Pre-Flash service time (60 second process time) can vary significantly without impacting any other MB_i in the serial line because it is not an MT_i or MT_{i-1} value for any buffer in the serial line. It can vary from 1 to 240 seconds without impacting other buffer MB_i values. If Pre-Flash service time exceeds 240 seconds, it begins to impact MB_4 and MB_3 as shown in Figure 5-2. Therefore, from a control policy perspective, the Pre-Flash process does not require significant monitoring or immediate knowledge of events as the process can vary significantly without impacting other buffers.

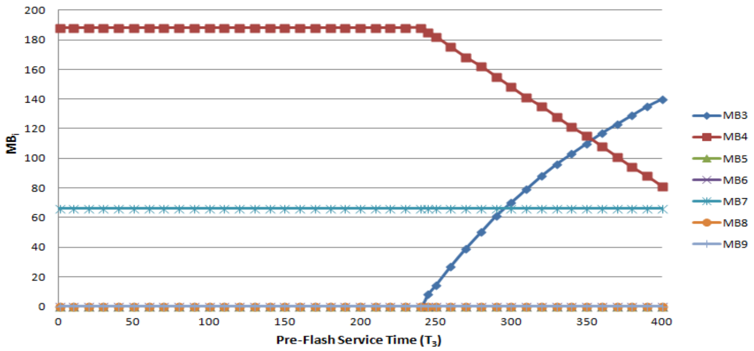
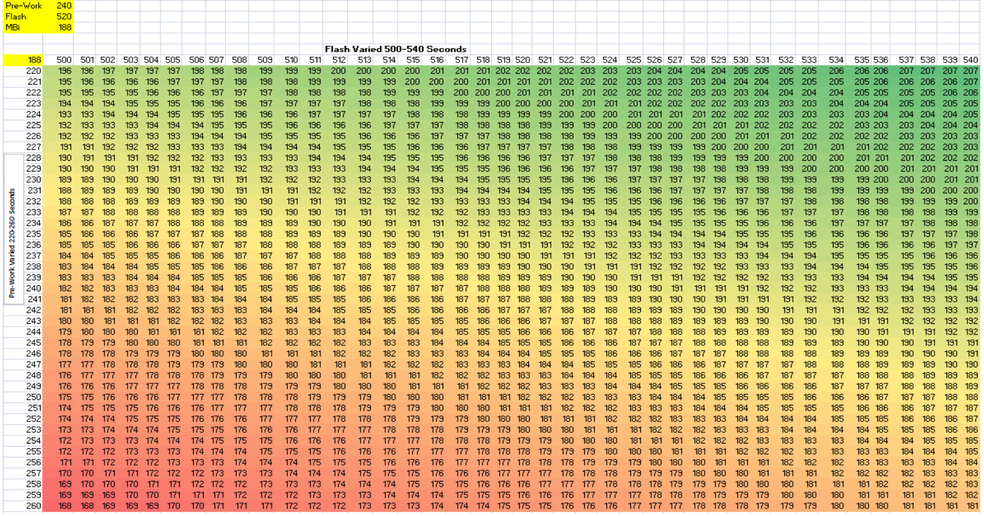


Figure 5-2. MB_i vs. pre flash service time (T_3)

We can expand this further to the critical MT_i and MT_{i-1} for MB_4. Let us say that for MB_4, where both MT_i (Flash = 520 seconds) and MT_{i-1} (Pre-Work = 240 seconds) are manual or have variable service times, one can use the results of (3.8) and create a two-way variable table (see Table 3) to find the maximum and minimum MB_4. Green indicates the maximum values and red indicates the minimum values In this case, MT_i was varied from 500 to 540 seconds and MT_{i-1} was varied from 220 to 260 seconds. Results show that deterministically MB_4 = 188. When using a two-way variable table we find MB_4 varies from a minimum of 168 to a maximum of 207. Using a simulation model, several hundred simulation runs are required to get these exact results.

Table 3. Two-way variable table for MB_4



Pre-Work 240
Flash 520
MBi 188

Flash Varied 500-540 Seconds

Pre-Work Varied 220-260 Seconds

188	500	501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523	524	525	526	527	528	529	530	531	532	533	534	535	536	537	538	539	540
220	196	196	197	197	197	197	198	198	198	199	199	199	200	200	200	200	201	201	201	202	202	202	202	203	203	203	204	204	204	204	205	205	205	205	206	206	206	207	207	207	207
221	195	196	196	196	196	197	197	197	198	198	198	199	199	199	199	200	200	200	201	201	201	201	202	202	202	203	203	203	203	204	204	204	205	205	205	205	206	206	206	206	207
222	195	195	195	195	196	196	196	197	197	197	198	198	198	198	199	199	199	200	200	200	201	201	201	201	202	202	202	202	203	203	203	204	204	204	204	205	205	205	205	206	206
223	194	194	194	195	195	195	196	196	196	197	197	197	197	198	198	198	199	199	199	200	200	200	200	201	201	201	202	202	202	202	203	203	203	203	204	204	204	205	205	205	205
224	193	193	194	194	194	195	195	195	196	196	196	197	197	197	197	198	198	198	199	199	199	199	200	200	200	201	201	201	201	202	202	202	203	203	203	203	204	204	204	204	205
225	192	193	193	193	194	194	194	195	195	195	196	196	196	196	197	197	197	198	198	198	198	199	199	199	200	200	200	200	201	201	201	202	202	202	202	203	203	203	204	204	204
226	192	192	192	193	193	193	194	194	194	195	195	195	195	196	196	196	197	197	197	198	198	198	198	199	199	199	200	200	200	200	201	201	201	202	202	202	202	203	203	203	203
227	191	191	192	192	192	193	193	193	194	194	194	194	195	195	195	196	196	196	197	197	197	197	198	198	198	199	199	199	199	200	200	200	201	201	201	201	202	202	202	203	203
228	190	191	191	191	192	192	192	193	193	193	193	194	194	194	195	195	195	196	196	196	196	197	197	197	198	198	198	199	199	199	199	200	200	200	200	201	201	201	202	202	202
229	190	190	190	191	191	191	192	192	192	192	193	193	193	194	194	194	195	195	195	196	196	196	196	197	197	197	198	198	198	198	199	199	199	200	200	200	200	201	201	201	201
230	189	189	190	190	190	191	191	191	191	192	192	192	193	193	193	194	194	194	195	195	195	195	196	196	196	197	197	197	197	198	198	198	199	199	199	199	200	200	200	201	201
231	188	189	189	189	190	190	190	190	191	191	191	192	192	192	193	193	193	194	194	194	194	195	195	195	196	196	196	197	197	197	197	198	198	198	199	199	199	199	200	200	200
232	188	188	188	189	189	189	189	190	190	190	191	191	191	192	192	192	193	193	193	193	194	194	194	195	195	195	196	196	196	196	197	197	197	198	198	198	198	199	199	199	200
233	187	187	188	188	188	188	189	189	189	190	190	190	191	191	191	192	192	192	193	193	193	193	194	194	194	195	195	195	195	196	196	196	197	197	197	198	198	198	198	199	199
234	186	186	187	187	187	188	188	188	189	189	189	190	190	190	191	191	191	192	192	192	192	193	193	193	194	194	194	195	195	195	195	196	196	196	197	197	197	197	198	198	198
235	185	186	186	186	187	187	187	188	188	188	189	189	189	190	190	190	191	191	191	191	192	192	192	193	193	193	194	194	194	194	195	195	195	196	196	196	196	197	197	197	198
236	185	185	185	186	186	186	187	187	187	188	188	188	189	189	189	190	190	190	190	191	191	191	192	192	192	193	193	193	194	194	194	194	195	195	195	196	196	196	196	197	197
237	184	184	185	185	185	186	186	186	187	187	187	188	188	188	189	189	189	190	190	190	190	191	191	191	192	192	192	193	193	193	193	194	194	194	195	195	195	195	196	196	196
238	183	184	184	184	185	185	185	186	186	186	187	187	187	188	188	188	189	189	189	189	190	190	190	191	191	191	192	192	192	192	193	193	193	194	194	194	195	195	195	195	196
239	183	183	183	184	184	184	185	185	185	186	186	186	187	187	187	188	188	188	188	189	189	189	190	190	190	191	191	191	192	192	192	192	193	193	193	194	194	194	194	195	195
240	182	182	183	183	183	184	184	184	185	185	185	186	186	186	187	187	187	187	188	188	188	189	189	189	190	190	190	191	191	191	191	192	192	192	193	193	193	194	194	194	194
241	181	182	182	182	183	183	183	184	184	184	185	185	185	186	186	186	186	187	187	187	188	188	188	189	189	189	190	190	190	191	191	191	191	192	192	192	193	193	193	193	194
242	181	181	181	182	182	182	183	183	183	184	184	184	185	185	185	186	186	186	186	187	187	187	188	188	188	189	189	189	190	190	190	190	191	191	191	192	192	192	193	193	193
243	180	180	181	181	181	182	182	182	183	183	183	184	184	184	185	185	185	185	186	186	186	187	187	187	188	188	188	189	189	189	189	190	190	190	191	191	191	192	192	192	192
244	179	180	180	180	181	181	181	182	182	182	183	183	183	184	184	184	184	185	185	185	186	186	186	187	187	187	188	188	188	189	189	189	189	190	190	190	191	191	191	192	192
245	178	179	179	180	180	180	181	181	181	182	182	182	182	183	183	183	184	184	184	185	185	185	186	186	186	187	187	187	188	188	188	188	189	189	189	190	190	190	191	191	191
246	178	178	178	179	179	179	180	180	180	181	181	181	182	182	182	183	183	183	184	184	184	185	185	185	186	186	186	187	187	187	188	188	188	188	189	189	189	190	190	190	191
247	177	177	178	178	178	179	179	179	180	180	180	181	181	181	182	182	182	183	183	183	184	184	184	185	185	185	186	186	186	187	187	187	187	188	188	188	189	189	189	190	190
248	176	177	177	177	178	178	178	179	179	179	180	180	180	181	181	181	182	182	182	183	183	183	184	184	184	185	185	185	186	186	186	187	187	187	187	188	188	188	189	189	189
249	176	176	176	177	177	177	178	178	178	179	179	179	180	180	180	181	181	181	182	182	182	183	183	183	184	184	184	185	185	185	186	186	186	186	187	187	187	188	188	188	189
250	175	175	176	176	176	177	177	177	178	178	178	179	179	179	180	180	180	181	181	181	182	182	182	183	183	183	184	184	184	185	185	185	185	186	186	186	187	187	187	188	188
251	174	175	175	175	176	176	176	177	177	177	178	178	178	179	179	179	180	180	180	181	181	181	182	182	182	183	183	183	184	184	184	185	185	185	185	186	186	186	187	187	187
252	174	174	174	175	175	175	176	176	176	177	177	177	178	178	178	179	179	179	180	180	180	181	181	181	182	182	182	183	183	183	184	184	184	184	185	185	185	186	186	186	187
253	173	173	174	174	174	175	175	175	176	176	176	177	177	177	178	178	178	179	179	179	180	180	180	181	181	181	182	182	182	183	183	183	184	184	184	184	185	185	185	186	186
254	172	173	173	173	174	174	174	175	175	175	176	176	176	177	177	177	178	178	178	179	179	179	180	180	180	181	181	181	182	182	182	183	183	183	183	184	184	184	185	185	185
255	172	172	172	173	173	173	174	174	174	175	175	175	176	176	176	177	177	177	178	178	178	179	179	179	180	180	180	181	181	181	182	182	182	183	183	183	183	184	184	184	185
256	171	171	172	172	172	173	173	173	174	174	174	175	175	175	176	176	176	177	177	177	178	178	178	179	179	179	180	180	180	181	181	181	182	182	182	183	183	183	183	184	184
257	170	170	171	171	172	172	172	173	173	173	174	174	174	175	175	175	176	176	176	177	177	177	178	178	178	179	179	179	180	180	180	181	181	181	182	182	182	182	183	183	183
258	169	170	170	170	171	171	172	172	172	173	173	173	174	174	174	175	175	175	176	176	176	177	177	177	178	178	178	179	179	179	180	180	180	181	181	181	182	182	182	182	183
259	169	169	169	170	170	171	171	171	172	172	172	173	173	173	174	174	174	175	175	175	176	176	176	177	177	177	178	178	178	179	179	179	180	180	180	181	181	181	181	182	182
260	168	168	169	169	169	170	170	171	171	171	172	172	172	173	173	173	174	174	174	175	175	175	176	176	176	177	177	177	178	178	178	179	179	179	180	180	180	181	181	181	181

5.2 Applying Chapter 4 results to the manufacturing center

In the first section we will apply the model in chapter 4 to the manufacturing center industry example. In the second section we quantify and illustrate in how the model significantly reduced number of calculations required to calculate the buffer cluster size.

5.2.1 Buffer clustering policy for manufacturing center

As discussed in the Introduction, the underlying motivation for this research was a case study where a manufacturing facility that produces mobile devices wished to change over from a serial line to a buffer cluster configuration. Table 2 in the Introduction shows the server stations and process times. The footprint for buffer cell holding an entity is $0.005 m^2$. The maximum buffer cluster size is $1.825\ m^2$.

Given B_1 and B_8 are the starting and ending inventory buffers, these are not included in the analysis. We use (3.8) and (4.5) to populate Table 4 with W_j cluster sets and size for buffers B_2 through B_7. The production line shift is 8 hours and 458 units are projected to ship by the end of the shift. There is a space constraint of 1.825 square meters for the buffer cluster size. In

Table 4, the cells in red indicate that the cluster does not meet the space constraint (i.e. the buffer cluster size $BB_j > 365$ entities).

Table 4. W_j buffer cluster sets and BB_j values for each buffer cluster set

$W_1 = \{B_2\}$	$W_2 =\{B_3\}$	$W_3 = \{B_4\}$	$W_4 = \{B_5\}$	$W_5 = \{B_6\}$
$BB_1 = 229$	$BB_2 = 229$	$BB_3 = 92$	$BB_4= 294$	$BB_5 = 1$ (Transport)
$W_6 = \{B_7\}$	$W_7 = \{B_2,B_3\}$	$W_8 = \{B_2,B_4\}$	$W_9 = \{B_2,B_5\}$	$W_{10} = \{B_2,B_6\}$
$BB_6 =121$	$BB_7 = 343$	$BB_8 = 251$	$BB_9 =294$	$BB_{10} = 230$
$W_{11}= \{B_2, B_7\}$	$W_{12}=\{B_3,B_4\}$	$W_{13}=\{B_3,B_5\}$	$W_{14} = \{B_3,B_6\}$	$W_{15} = \{B_3,B_7\}$
$BB_{11} = 237$	$BB_{12} = 274$	$BB_{13} = 346$	$BB_{14} = 230$	$BB_{15} = 246$
$W_{16} = \{B_4,B_5\}$	$W_{17}=\{B_4,B_6\}$	$W_{18} = \{B_4,B_7\}$	$W_{19}=\{B_5,B_6\}$	$W_{20}=\{B_5,B_7\}$
$BB_{16} = 326$	$BB_{17} = 93$	$BB_{18} = 126$	$BB_{19} =295$	$BB_{20}=337$
$W_{21}=\{B_6,B_7\}$	$W_{22} = \{B_2,B_3, B_4\}$	$W_{23} = \{B_2,B_3,B_5\}$	$W_{24} = \{B_2,B_3,B_6\}$	$W_{25} = \{B_2,B_3,B_7\}$
$BB_{21}=122$	$BB_{22} = 365$	$BB_{23} = 401$	$BB_{24} = 344$	$BB_{25} = 351$
$W_{26} = \{B_2,B_4,B_5\}$	$W_{27} = \{B_2,B_4,B_6\}$	$W_{28} = \{B_2,B_4,B_7\}$	$W_{29} = \{B_2,B_5,B_6\}$	$W_{30} = \{B_2,B_5,B_7\}$
$BB_{26} = 326$	$BB_{27} = 252$	$BB_{28} = 259$	$BB_{29} = 295$	$BB_{30} = 337$
$W_{31} = \{B_2,B_6,B_7\}$	$W_{32}=\{B_3,B_4,B_5\}$	$W_{33}=\{B_3,B_4,B_6\}$	$W_{34} =\{B_3,B_4,B_7\}$	$W_{35}=\{B_3, B_5,B_6\}$
$BB_{31} = 238$	$BB_{32} = 391$	$BB_{33} = 275$	$BB_{34} = 291$	$BB_{35} = 347$
$W_{36}=\{B_3, B_5,B_7\}$	$W_{37}=\{B_3, B_6,B_7\}$	$W_{38}=\{B_4,B_5,B_6\}$	$W_{39}=\{B_4,B_5,B_7\}$	$W_{40}=\{B_4, B_6,B_7\}$
$BB_{36} = 363$	$BB_{37} = 247$	$BB_{38} = 327$	$BB_{39} = 360$	$BB_{40} = 127$
$W_{41}=\{B_5, B_6,B_7\}$	$W_{42} = \{B_2,B_3,B_4,B_5\}$	$W_{43} = \{B_2,B_3,B_4,B_6\}$	$W_{44} = \{B_2,B_3,B_4,B_7\}$	$W_{45} = \{B_2,B_3,B_5,B_6\}$
$BB_{41} = 338$	$BB_{42} = 423$	$BB_{43} = 366$	$BB_{44} = 373$	$BB_{45} = 402$
$W_{46} = \{B_2,B_3,B_5,B_7\}$	$W_{47} = \{B_2,B_3,B_6,B_7\}$	$W_{48} = \{B_2,B_4,B_5,B_6\}$	$W_{49} = \{B_2,B_4,B_5,B_7\}$	$W_{50} = \{B_2,B_4,B_6,B_7\}$
$BB_{46} = 409$	$BB_{47} = 352$	$BB_{48} = 327$	$BB_{49} = 360$	$BB_{50} = 260$
$W_{51} = \{B_2,B_5,B_6,B_7\}$	$W_{52} = \{B_3,B_4,B_5,B_6\}$	$W_{53}= \{B_3,B_4,B_5,B_7\}$	$W_{54} = \{B_3,B_4,B_6,B_7\}$	$W_{55} = \{B_3,B_5,B_6,B_7\}$
$BB_{51} = 338$	$BB_{52} = 392$	$BB_{53} = 408$	$BB_{54} = 292$	$BB_{55} = 364$
$W_{56} = \{B_4,B_5,B_6,B_7\}$	$W_{57} = \{B_2,B_3,B_4,B_5,B_6\}$	$W_{58} = \{B_2,B_3,B_4,B_5,B_7\}$	$W_{59} = \{B_2,B_3,B_4,B_6,B_7\}$	$W_{60} = \{B_2,B_3,B_5,B_6,B_7\}$
$BB_{56} = 361$	$BB_{57} = 424$	$BB_{58}= 431$	$BB_{59} = 374$	$BB_{60} = 410$
$W_{61} = \{B_2,B_4,B_5,B_6,B_7\}$	$W_{62} = \{B_3,B_4,B_5,B_6,B_7\}$	$W_{63} = \{ B_2,B_3,B_4,B_5,B_6,B_7\}$		
$BB_{61} = 361$	$BB_{62} = 409$	$BB_{63} = 432$		

The objective function according to (4.8) and constraints according to (4.9), (4.10) and (4.11) are below. As discussed in Chapter 4, a buffer cannot participate in multiple clusters at the same time. Table 5 shows the buffer storage space savings for the top buffer cluster configurations considered by the manufacturing center compared to that of the cluster that does not consider the space constraint and that for dedicated buffers. The manufacturing center desired to leverage the cluster for work cells to minimize the buffer storage space. The top buffer cluster configurations considered by the manufacturing center contained buffer clusters with three or four buffers clustered together.

Objective Function:
Minimize $\sum_{i=1}^{63} BB_j * X_j$

Subject to:
Constraint for B_2:
X1+X7+X8+X9+X10+X11+X22+X23+X24+X25+X26+X27+X28+X29+X30+X31+X42+X43+X44 + X45 +X46+ X47+X48+X49+X50+X51+X57+X58+X59+X60+X61+X63=1
Constraint for B_3:
X2+X7+X12+X13+X14+X15+X22+X23+X24+X25+X32+X33+X34+X35+X36+X37+X42+X43+ X44+X45+X46+X47+X52+X53+X54+X55+X57+X58+X59+X60+X62+X63=1
Constraint for B_4:
X3+X8+X12+X16+X17+X18+X22+X26+X27+X28+X32+X33+X34+X38+X39+X40+X42+X43+ X44+ X48+X49+X50+X52+X53+X54+X56+X57+X58+X59+X61+X62+X63=1
Constraint for B_5:
X4+X9+X13+X16+X19+X20+X23+X26+X29+X30+X32+X35+X36+X38+X39+X41+X42+X45+ X46+X48+X49+X51+X52+X53+X55+X56+X57+X58+X60+X61+X62+X63=1
Constraint for B_6:
X5+X10+X14+X17+X19+X21+X24+X27+X29+X31+X33+X35+X37+X38+X40+X41+X43+ X45+X47+X48+X50+X51+X52+X54+X55+X56+X57+X59+X60+X61+X62+X63=1
Constraint for B_7:
X6+X11+X15+X18+X20+X21+X25+X28+X30+X31+X34+X36+X37+X39+X40+X41+X44+X46+ X47+X49+X50+X51+X53+X54+X55+X56+X58+X59+X60+X61+X62+X63=1

Space Constraints: $BB_j * 0.005 * X_j \leq 1.825$, $1 \leq j \leq 63$

Binary Constraint: X_j = 0 or 1

Figure 5-3 is the original production line configuration. Batches of 80 come from inventory and enter the production line (B_1). Batches of 80 are put on a pallet and shipped (B_8). The top cluster configurations considered by the facility based on buffer storage savings were then entered into a facility layout tool. The configuration shown in Figure 5-4 was selected {B_3, B_4, B_7}, {B_2, B_5, B_6} resulting in a 39.3% buffer storage savings (1.9 square meters).

Table 5. Buffer cluster sets and buffer storage savings

Cluster Set	Size (No. Entities)	Size (m^2)	Buffer Storage Space Savings (m^2)	Space Savings %
$\{B_2\},\{B_3\},\{B_4\},\{B_5\}, \{B_6\},\{B_7\}$*	966	4.83		
$\{B_2,B_3,B_4,B_5,B_6,B_7\}$**	432	2.16	2.67	55.3%
$\{B_2,B_4, B_5,B_6,B_7\}$, $\{B_3\}$	590	2.95	1.88	38.9%
$\{B_2,B_5,B_6,B_7\}$, $\{B_3 ,B_4\}$	612	3.06	1.77	36.7%
$\{B_3,B_5,B_6,B_7\}$, $\{B_2,B_4\}$	615	3.08	1.76	36.3%
$\{B_3,B_4,B_6,B_7\}$, $\{B_2,B_5\}$	586	2.93	1.90	39.3%
$\{B_2,B_4,B_5,B_6\}$, $\{B_3,B_7\}$	573	2.87	1.97	40.7%
$\{B_2,B_4,B_6,B_7\}$, $\{B_3,B_5\}$	606	3.03	1.80	37.3%
$\{B_3,B_5,B_7\}$, $\{B_2,B_4,B_6\}$	615	3.08	1.76	36.3%
$\{B_3,B_5,B_7\}$, $\{B_6\}$,$\{B_2,B_4\}$	615	3.08	1.76	36.3%
$\{B_3,B_4,B_7\}$, $\{B_2, B_5, B_6\}$	586	2.93	1.90	39.3%
$\{B_3,B_4,B_7\}$, $\{B_6\}$,$\{B_2, B_5\}$	586	2.93	1.90	39.3%
$\{B_3,B_4,B_6\}$, $\{B_2, B_5, B_7\}$	612	3.06	1.77	36.6%
$\{B_2,B_5,B_7\}$, $\{B_6\}$,$\{B_3, B_4\}$	612	3.06	1.77	36.6%
$\{B_2,B_4,B_5\}$, $\{B_6\}$,$\{B_3, B_7\}$	573	2.87	1.97	40.7%

*Dedicated Buffers

** Optimal Buffer Cluster without space constraints

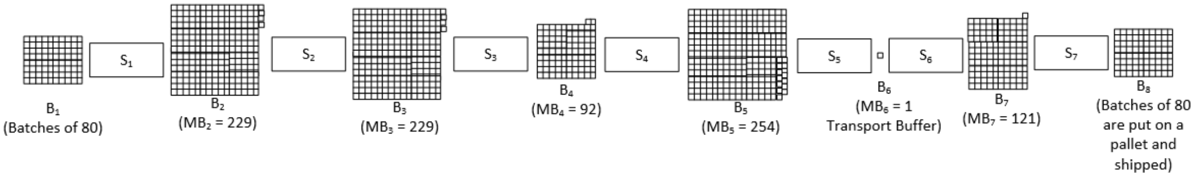


Figure 5-3. Serial production line

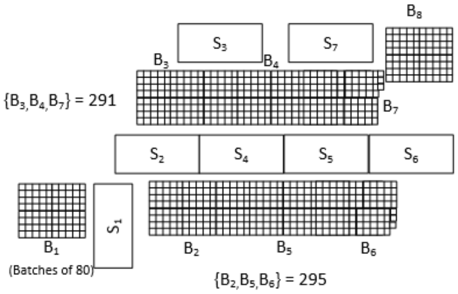


Figure 5-4. Production line with buffer clusters

As discussed in the introduction, equation (4.5) along with the objective function (4.8) and constraints in (4.9) through (4.11) can be used to conduct sensitivity analysis of the buffer cluster size by varying parameters such as server process time T_i and production demand K1. Now we leverage the framework of the model and vary the process time of one server to demonstrate how the model can be used for sensitivity analysis. In this case we vary the process time of server S_2 to three seconds, calculate the BB_j values for each cluster set and show the buffer cluster sets in Table 6. As before, the cells in red indicate that the cluster does not meet the space constraint and are the same cells that did not meet the space constraint in Table 4. If the cells are highlighted in purple, they used to meet the space constraint, but due to the change in the process times, no longer meet the constraint. The cells highlighted in blue exceeded the space constraint in Table 4, but now meet the constraint in Table 6. The BB_j values in red indicate a change in the size of the cluster from Table 4.

We take the configurations from Table 5 and identify in Table 7 that there are configurations that now, with S_2 equaling 3 seconds do not meet the space constraint (highlighted in red). We see that the configuration selected with a process time S_2 equaling 2 seconds, {$B_{3,}$ $B_{4,}$ B_7}, {B_2, B_5, B_6}, with 586 entities, achieves a total buffer size of 572 entities when the process time of S_2 is 3 seconds. This scenario is highlighted in green in Table 7. So the initial buffer cluster set shown in Figure 5-4 can remain and still satisfy the space constraints when the process time of S_2 varies from two to three seconds.

Table 6. W_j buffer cluster sets and BB_j values for each buffer cluster set ($T_2 = 3s$)

$W_1 = \{B_2\}$	$W_2 = \{B_3\}$	$W_3 = \{B_4\}$	$W_4 = \{B_5\}$	$W_5 = \{B_6\}$
$BB_1 = 305$	$BB_2 = 115$	$BB_3 = 92$	$BB_4 = 294$	$BB_5 = 1$ (Transport)
$W_6 = \{B_7\}$	$W_7 = \{B_2, B_3\}$	$W_8 = \{B_2, B_4\}$	$W_9 = \{B_2, B_5\}$	$W_{10} = \{B_2, B_6\}$
$BB_6 = 121$	$BB_7 = 343$	$BB_8 = 327$	$BB_9 = 363$	$BB_{10} = 306$
$W_{11} = \{B_2, B_7\}$	$W_{12} = \{B_3, B_4\}$	$W_{13} = \{B_3, B_5\}$	$W_{14} = \{B_3, B_6\}$	$W_{15} = \{B_3, B_7\}$
$BB_{11} = 313$	$BB_{12} = 183$	$BB_{13} = 294$	$BB_{14} = 116$	$BB_{15} = 140$
$W_{16} = \{B_4, B_5\}$	$W_{17} = \{B_4, B_6\}$	$W_{18} = \{B_4, B_7\}$	$W_{19} = \{B_5, B_6\}$	$W_{20} = \{B_5, B_7\}$
$BB_{16} = 326$	$BB_{17} = 93$	$BB_{18} = 126$	$BB_{19} = 295$	$BB_{20} = 337$
$W_{21} = \{B_6, B_7\}$	$W_{22} = \{B_2, B_3, B_4\}$	$W_{23} = \{B_2, B_3, B_5\}$	$W_{24} = \{B_2, B_3, B_6\}$	$W_{25} = \{B_2, B_3, B_7\}$
$BB_{21} = 122$	$BB_{22} = 365$	$BB_{23} = 401$	$BB_{24} = 344$	$BB_{25} = 351$
$W_{26} = \{B_2, B_4, B_5\}$	$W_{27} = \{B_2, B_4, B_6\}$	$W_{28} = \{B_2, B_4, B_7\}$	$W_{29} = \{B_2, B_5, B_6\}$	$W_{30} = \{B_2, B_5, B_7\}$
$BB_{26} = 385$	$BB_{27} = 328$	$BB_{28} = 335$	$BB_{29} = 364$	$BB_{30} = 371$
$W_{31} = \{B_2, B_6, B_7\}$	$W_{32} = \{B_3, B_4, B_5\}$	$W_{33} = \{B_3, B_4, B_6\}$	$W_{34} = \{B_3, B_4, B_7\}$	$W_{35} = \{B_3, B_5, B_6\}$
$BB_{31} = 314$	$BB_{32} = 358$	$BB_{33} = 184$	$BB_{34} = 208$	$BB_{35} = 295$
$W_{36} = \{B_3, B_5, B_7\}$	$W_{37} = \{B_3, B_6, B_7\}$	$W_{38} = \{B_4, B_5, B_6\}$	$W_{39} = \{B_4, B_5, B_7\}$	$W_{40} = \{B_4, B_6, B_7\}$
$BB_{36} = 337$	$BB_{37} = 141$	$BB_{38} = 327$	$BB_{39} = 360$	$BB_{40} = 127$
$W_{41} = \{B_5, B_6, B_7\}$	$W_{42} = \{B_2, B_3, B_4, B_5\}$	$W_{43} = \{B_2, B_3, B_4, B_6\}$	$W_{44} = \{B_2, B_3, B_4, B_7\}$	$W_{45} = \{B_2, B_3, B_5, B_6\}$
$BB_{41} = 338$	$BB_{42} = 423$	$BB_{43} = 366$	$BB_{44} = 373$	$BB_{45} = 402$
$W_{46} = \{B_2, B_3, B_5, B_7\}$	$W_{47} = \{B_2, B_3, B_6, B_7\}$	$W_{48} = \{B_2, B_4, B_5, B_6\}$	$W_{49} = \{B_2, B_4, B_5, B_7\}$	$W_{50} = \{B_2, B_4, B_6, B_7\}$
$BB_{46} = 409$	$BB_{47} = 352$	$BB_{48} = 386$	$BB_{49} = 393$	$BB_{50} = 336$
$W_{51} = \{B_2, B_5, B_6, B_7\}$	$W_{52} = \{B_3, B_4, B_5, B_6\}$	$W_{53} = \{B_3, B_4, B_5, B_7\}$	$W_{54} = \{B_3, B_4, B_6, B_7\}$	$W_{55} = \{B_3, B_5, B_6, B_7\}$
$BB_{51} = 372$	$BB_{52} = 359$	$BB_{53} = 384$	$BB_{54} = 209$	$BB_{55} = 338$
$W_{56} = \{B_4, B_5, B_6, B_7\}$	$W_{57} = \{B_2, B_3, B_4, B_5, B_6\}$	$W_{58} = \{B_2, B_3, B_4, B_5, B_7\}$	$W_{59} = \{B_2, B_3, B_4, B_6, B_7\}$	$W_{60} = \{B_2, B_3, B_5, B_6, B_7\}$
$BB_{56} = 361$	$BB_{57} = 424$	$BB_{58} = 431$	$BB_{59} = 374$	$BB_{60} = 410$
$W_{61} = \{B_2, B_4, B_5, B_6, B_7\}$	$W_{62} = \{B_3, B_4, B_5, B_6, B_7\}$	$W_{63} = \{B_2, B_3, B_4, B_5, B_6, B_7\}$		
$BB_{61} = 394$	$BB_{62} = 385$	$BB_{63} = 432$		

Table 7. Buffer cluster sets and buffer storage savings with T_2 at 2 and 3 seconds

Cluster Set	**Size (No. Entities) $T_2 = 2$ sec**	**Size (No. Entities) $T_2 = 3$ sec**	**Size (m^2) $T_2 = 2$ sec**	**Size (m^2) $T_2 = 3$ sec**	**Buffer Storage Space Savings (m^2) $T_2 = 2$ sec**	**Buffer Storage Space Savings (m^2) $T_2 = 3$ sec**	**Space Savings % $T_2 = 2$ sec**	**Space Savings % $T_2 = 3$ sec**
$\{B_2\},\{B_3\},\{B_4\},\{B_5\},\{B_6\},\{B_7\}$*	966	928	4.83	4.64				
$\{B_2,B_3,B_4,B_5,B_6,B_7\}$**	432	432	2.16	2.16	2.67	2.48	55.3%	53.5%
$\{B_2,B_4,B_5,B_6,B_7\}$, $\{B_3\}$	590	509	2.95	2.55	1.88	2.10	38.9%	45.2%
$\{B_2,B_5,B_6,B_7\}$, $\{B_3,B_4\}$	612	555	3.06	2.78	1.77	1.87	36.7%	40.2%
$\{B_3,B_5,B_6,B_7\}$, $\{B_2,B_4\}$	615	665	3.08	3.33	1.76	1.32	36.3%	28.3%
$\{B_3,B_4,B_6,B_7\}$, $\{B_2,B_5\}$	586	572	2.93	2.86	1.90	1.78	39.3%	38.4%
$\{B_2,B_4,B_5,B_6\}$, $\{B_3,B_7\}$	573	526	2.87	2.63	1.97	2.01	40.7%	43.3%
$\{B_2,B_4,B_6,B_7\}$, $\{B_3,B_5\}$	606	630	3.03	3.15	1.80	1.49	37.3%	32.1%
$\{B_3,B_5,B_7\}$, $\{B_2,B_4,B_6\}$	615	665	3.08	3.33	1.76	1.32	36.3%	28.3%
$\{B_3,B_5,B_7\}$, $\{B_6\}$,$\{B_2,B_4\}$	615	665	3.08	3.33	1.76	1.32	36.3%	28.3%
$\{B_3,B_4,B_7\}$, $\{B_2, B_5, B_6\}$	586	572	2.93	2.86	1.90	1.78	39.3%	38.4%
$\{B_3,B_4,B_7\}$, $\{B_6\}$,$\{B_2, B_5\}$	586	572	2.93	2.86	1.90	1.78	39.3%	38.4%
$\{B_3,B_4,B_6\}$, $\{B_2, B_5, B_7\}$	612	555	3.06	2.78	1.77	1.87	36.6%	40.2%
$\{B_2,B_5,B_7\}$, $\{B_6\}$,$\{B_3, B_4\}$	612	555	3.06	2.78	1.77	1.87	36.6%	40.2%
$\{B_2,B_4,B_5\}$, $\{B_6\}$,$\{B_3, B_7\}$	573	526	2.87	2.63	1.97	2.01	40.7%	43.3%

5.2.2 Computational savings for buffer cluster sizing

Table 8 shows that for this case study, we identified 25 critical time steps to measure the buffer size, resulting in 25 calculations. For Buffer B_6, because $MT_{i-1} > MT_i$, no time interval to detect the maximum buffer size is required because buffer size required is always 1 (as discussed in Corollary 2, this is a transport buffer).

Table 9 shows the average savings in time steps processed and average solution time savings benefits based on number of buffers to cluster in the sequential line. We start with 6 buffers similar to the industry example and double the production line size to 12 buffers and 24 buffers respectively. We also vary the K1 or production demand (from 100 to 300) such that it would cover a production shift interval spread of 8 to 12 hours.

Table 8. Number of time steps for required buffer size computations

Element B_i	B_2	B_3	B_4	B_5	B_6	B_7
$\bar{t}_{1\,Bi}$	458	917	1831	2292	0	6434
$\bar{t}_{2\,Bi}$	459	918	1836	2307	0	6438
$\bar{t}_{2\,Bi}$- $\bar{t}_{1\,Bi}$ Sum B_2 – B_7 is 25	1	1	5	15	0	4

Table 9. Calculation and computation time savings varying K1

(A)	**(B)**	**(C)**	**(D)**	**(E)**	**(F)**	**(G)**
No. Buffers	Average Time Steps	Average Solution Time 36000 time steps (sec)	Average Sum $\bar{t}_{2\,Bi}$- $\bar{t}_{1\,Bi}$ Time Steps	Solution Time for $\bar{t}_{2\,Bi}$- $\bar{t}_{1\,Bi}$ Calculations (sec)	Savings in time steps processed ((B)-(D))/(B) (%)	Savings in Solution Time ((C)-(E))/(C) (%)
6	36000	137	422	19	99%	86%
12	36000	227	921	25	97%	89%
24	36000	466	2355	33	93%	93%

6. CONCLUSIONS AND FUTURE WORK

The main contributions of the research are:

1. Closed Form, exact method, time-dependent models were derived in Chapter 3 for extracting state space parameters for an N-Server, N+1-Buffer sequential line with accurate results. State space parameters include the number of arrivals and departures at any buffer or server for any given time of interest and the maximum number of entities a buffer will experience when servers have capacity of unity and buffers have unlimited capacity. When a queue has its capacity reduced, the model determined the blocking time and transition times for any queue and server in the serial line. An algorithm is also derived for determining which queues and which servers are impacted by the reduced capacity queue and the number of entities in any queue or server at time of interest t. The algorithm provided the rules for determining impacted arrivals and departures for B_i and S_i and the corresponding equations to use.
2. We used the time dependent model of the sequential line described in Chapter 3 and applied them to a different model, a buffer cluster as shown in Chapter 4. We derived an optimization framework that enabled a buffer clustering policy and provides output of the required buffer sizing for that policy. The result reduces the buffers storage space and thus the production line footprint when implemented, while ensuring no bottlenecks. We also reduced the buffer and time search space significantly reducing the number of computations. We demonstrated in the case study how the models derived can be used to conduct sensitivity analysis of the buffer cluster size by varying parameters such as process time or production demand. Lastly, we showed how the buffer clustering policy can be used in a facility layout tool where a feasible layout concept is generated.

Our results suggest that time-dependent exact methods can be derived and applied with accurate results. These methods were applied to the industry examples for both a distribution center where distribution line consisted of 9 processes and manufacturing center where production line consisted of 7 processes. For the distribution center we validated and aided in key problem areas identified by the distribution center team: (1) the maximum buffer size allowed such that no buffer experiences blocking during a shift (2) the shift where a buffer's capacity should change to meet the production demand changes (3) the number of units processed by a bottleneck station when time of blocking occurs (4) the buffer transition and block time when a buffer size is reduced (5) the reaction time to route a resource (operator or workstation) to a bottleneck process to prevent impact to other processes (6) the time a failed machine must recover by as not to impact the production line or the time interval which to route a resource (operator or workstation) to a failed process (7) the maximum demand that the serial line can support given the limited shift time and buffer sizes (8) the ability to predict production line behavior with varying process times. For the manufacturing center, we developed and applied a buffer clustering model reducing buffer storage space. The models are applied to these areas in Chapter 5.

Results such as those presented in this paper enable one to extract the parametric state space of a system at a given time without any significant computational efforts and as an alternative to running a full simulation. We now discuss the benefits of these models. From the case study above, the results of this research enable us to identify sensitivity of processes (servers) to human or machine failure rates and impact based on the duration of the failure. We are able to determine by what time a workstation must recover without impacting the remaining workstations. Another benefit is that given the knowledge of timing of events across the entire system, allows one to enable control policies minimizing technology investments. For example, knowledge of the entire state space at a given time means that technologies such as sensors and wireless communications do not need to reside at every server or queue. In manufacturing and distribution centers today, workstations, operators and product entities have wireless communication devices that report sensing data, radio frequency identification or bar code information up to a central database. Deploying these devices across all production elements such as buffers and workstations in addition to sending data at a high frequency rate leads to high fixed and maintenance costs for investment in communication devices and large database storage. A parametric state space allows for a minimum set of devices and monitoring or reporting data for critical events as opposed to every event.

Our results suggest that parametric time-dependent exact methods can be derived and applied with accurate results. We derived and demonstrated usage of a time based parametric model for N+1- Queue, N-Server sequential line to assist production environments in sizing buffers, in particular, buffer clusters appropriately when alternate production line configurations are desired. We derived an optimization framework that enabled a clustering policy and provides output of the required buffer sizing for that policy. The result reduces production line footprint when implemented, thus minimizing the facility space utilized while ensuring no bottlenecks. We also reduced the buffer and time search space significantly reducing the number of computations.

Related future studies relax assumptions of the models in this paper and also expand configurations. In particular, studies that relax the capacity constraints and reliability constraints of the servers would provide added value to the sequential line and buffer clustering models. The ability to extract time dependent state space models is a rich area for Operations Research with several applications in industry.

CITED LITERATURE

[1] F. Schuler and H. Darabi, "Supervisory control and data collection policies for a distribution center modeled as a discrete event system," in *IEEE International Conference on Networking, Sensing and Control*, Chicago, 2010.

[2] C. Cassandras and S. LaFortune, Introduction to discrete event systems, New York City, New York: Springer Science and Business Media, Inc., 2007.

[3] T. Murata, "Petri nets: properties, analysis and applications," *Proceedings of the IEEE,* vol. 77, no. 4, pp. 541-580, 1989.

[4] K. VanHee, A. Serebrenik and N. Sidorova, "Token history Petri nets," *Fundamente Informaticae,* vol. 85, no. 1-4, pp. 219-234, 2008.

[5] G. Zeng, W. Wu, M. Zhou, W. Mao, H. Su and J. Chu, "Design of Petri net based deadlock prevention controllers for flexible manufacturing systems," in *IEEE International Conference on Systems, Man and Cybernetics*, San Antonio, Texas, 2009.

[6] Y. Tang and L. Pogach, "A conceptual model for a value-driven learning healthcare system," in *IEEE International Conference on Systems, Man and Cybernetics.*, San Antonio, Texas, 2009.

[7] J. Li, Y. Fan and M. Zhou, "Timing constraint workflow nets for workflow analysis," *IEEE Transactions on Systems, Man and Cybernetics, Part A. Systems and Humans,* vol. 33, no. 2, pp. 179- 193, 2003.

[8] R. Guillerm, H. Demmou and N. Sadou, "ESA Petrinet: Petri net based tool for reliability analysis," in *IEEE International Conference on Systems, Man and Cybernetics*, San Antonio, Texas, 2009.

[9] Z. Suraj, B. Fryc, Z. Matusiewicz and K. Pancerz, "A Petri net system - an overview," *Fundameta Informaticae,* vol. 71, no. 1, pp. 101-120, 2006.

[10] B. Hollocks, "Forty years of discrete event simulation - a personal reflection," *Journal of the Operational Research Society,* vol. 57, no. 12, pp. 1383-1399, 2006.

[11] Rockwell Automation, "Arena Simulation Software," Rockwell Automation, [Online]. Available: http://www.arenasimulation.com/. [Accessed 23 June 2016].

[12] Simul8, "Simul8 - Process improvement with simulation software," Simul8, [Online]. Available: http://www.simul8.com/. [Accessed 6 February 2016].

[13] Lanner Group, "Witness," Lanner, [Online]. Available: www.lanner.com. [Accessed 6 February 2016].

[14] Dassault Systems, "Delmia Digital Manufacturing and Production," Dassault Systems, [Online]. Available: http://www.3ds.com/products/delmia. [Accessed 6 February 2016].

[15] MathWorks, "SimEvents - model and simulate discrete-event systems," MathWorks, [Online]. Available: http://www.mathworks.com/products/simevents/. [Accessed 6 February 2016].

[16] University of Hamburg, Germany, "Petri nets tools database quick overview," University of Hamburg, Germany, [Online]. Available: http://www.informatik.uni-hamburg.de/TGI/PetriNets/tools/quick.html. [Accessed 6 February 2016].

[17] University of Hamburg, "Petri nets world," University of Hamburg, [Online]. Available: http://www.informatik.uni-hamburg.de/TGI/PetriNets/. [Accessed 6 February 2016].

[18] H. Storrle, "An evaluation of high-end tools for Petri-nets," Ludwig-Maximilians-Universit at Munchen Institut fur Informatik, [Online]. Available: http://www.pst.ifi.lmu.de/~stoerrle/V/Evaluierung.pdf. [Accessed 6 February 2016].

[19] M. Dong and F. Chen, "Process modeling and analysis of manufacturing supply chain networks using object oriented Petri nets," *Robotics and Computer Integrated Manufacturing,* vol. 17, no. 1-2, pp. 121-129, 2001.

[20] S. Ling and H. Schmidt, "Time Petri nets for workflow modeling and analysis," in *IEEE International Conference on Systems, Man, and Cybernetics*, Nashville, Tennessee, 2000.

[21] J. Wang, Timed Petri nets: theory and applications, Massachusetts: Kluwer Academic Publishers, 1998, pp. 4-6.

[22] C. Maziero, "The ARP tool," Federal University of Santa Catarina - Brazil, 14 September 2009. [Online]. Available: http://www.ppgia.pucpr.br/~maziero/doku.php/software:arp_tool. [Accessed 23 June 2012].

[23] O. Grumber and H. Veith, Twenty-five years of model checking history, achievements, perspectives, New York City: Springer, 2008.

[24] R. Lyngs and T. Mailund, "Textual interchange format for high-level Petri nets," in *Proceedings of the workshop on practical use of coloured Petri nets and design/CPN*, Denmark, 1998.

[25] PNML.org, "PNML.org," PNML.org, [Online]. Available: http://www.pnml.org/. [Accessed 6 February 2016].

[26] G. Frey and L. Litz, "XML based interchange for Petri nets - a control engineer's point of view," University of Kaiserslautern, Institute of Process Automation, Kaiserslautern, Germany, 2000.

[27] M. Cabasino, L. Contini, A. Giua, C. Seatuzu and A. Solinas, "A software platform for the integration of discrete event system tools," in *IEEE Conference on Automation, Science and Engineering*, Trieste, Italy, 2011.

[28] J. Coppola, "Pnet Lab," Automatic Control Group of the University of Salerno, 23 June 2012. [Online]. Available: http://www.automatica.unisa.it/PnetLab.html. [Accessed 23 June 2012].

[29] S. S. O. L. Projects, "Poses++," Gesellschaft fur prozebautomation and consulting mbh , August 2011. [Online]. Available: http://www.gpc.de/e_poses.html. [Accessed 23 June 2012].

[30] Department of Automatic Control and Applied Informatics of the Technical University of Iasi, Romania., "Petri net toolbox for Matlab," Department of Automatic Control and Applied Informatics of the Technical University of Iasi, Romania., [Online]. Available: http://www.ac.tuiasi.ro/pntool/. [Accessed 6 February 2016].

[31] G. A. T. U. o. Iasi, "Third-Party Products & Services - Petri net toolbox," Mathworks, June 2007. [Online]. Available: http://www.mathworks.com/products/connections/product_detail/product_35741.html. [Accessed 23 June 2012].

[32] U. o. Torino, "Graphical editor and analyzer for timed and stochastic Petri nets," Dipartmento di Informatica, University of Torino, [Online]. Available: http://www.di.unito.it/~greatspn/index.html. [Accessed 6 February 2016].

[33] Theoretical Foundations Group of the Department for Informatics of the University of Hamburg, "Renew - the reference net workshop," Theoretical Foundations Group of the Department for Informatics of the University of Hamburg, [Online]. Available: http://www.renew.de/. [Accessed 6 February 2016].

[34] University of Technology Eindhoven, "CPN Tools," University of Technology Eindhoven, [Online]. Available: http://cpntools.org/. [Accessed 6 February 2016].

[35] University of Technology Eindhoven and Deloitte, "Yasper - Yet another smart process editor," University of Technology Eindhoven and Deloitte, [Online]. Available: http://www.yasper.org/. [Accessed 6 February 2016].

[36] E. University of Technology, "ExSpecT - Executable Specification Tool," Stichting ASPT, [Online]. Available: http://www.exspect.com/. [Accessed 6 February 2016].

[37] E. Clark, O. Grumber, M. Minea and D. Peled, "State space reduction using partial order techniques," *International Journal on Software Tools for Technology Transfer ,* vol. 2, no. 3, pp. 279-287, 1999.

[38] S. Park, N. Raman and M. Shaw, "Adapative scheduling in dynamic flexible manufacturing system: a dynamic rule selection approach," *IEEE Transactions on Robotics and Automation,* vol. 13, no. 4, pp. 486- 502 , 1999.

[39] V. Kochikar and T. Narendran, " A framework for assessing the flexibility of manufacturing systems," *International Journal of Production Research,* vol. 30, no. 12, pp. 2873-2895, 1992.

[40] M. Jeng, C. Lin and Y. Huang, "Petri net dynamics-based scheduling of flexible manufacturing systems with assembly," *Journal of Intelligent Manufacturing ,* vol. 10, no. 6, pp. 541-555, 1999.

[41] R. Brennan, "Toward real-time distributed intelligent control: a survey of research themes and applications," *IEEE Transactions on Systems, Man and Cybernetics, Part C,* vol. 37, no. 5, pp. 744- 765 , 2007.

[42] J. Moffett and M. Sloman, "Policy hierarchies for distributed systems management," *IEEE Journal on Selected Areas in Communications,* vol. 11, no. 9, pp. 1404-1414 , 1993.

[43] M. Musuvathi, "Handling state space explosion," 10 August 2002. [Online]. Available: http://static.usenix.org/events/osdi2002/tech/full_papers/musuvathi/musuvathi_html/node11.html. [Accessed 23 June 2012].

[44] U. Buy and R. Sloan, "A Petri net based approach to real-time program analysis," in *Proceedings of the Seventh International Workshop on Software Specification and Design*, Los Alamitos, 1993.

[45] U. Buy and R. Sloan, "Reduction rules for time Petri nets," *Acta Informatica,* vol. 33, no. 7, pp. 687-706, 1996.

[46] J. Wang, Y. Deng and M. Zhou, "Compositional time Petri nets and reduction rules," *IEEE Transactions on Systems, Man, and Cybernetics, Part B.,* vol. 30, no. 4, pp. 562-572, 2000.

[47] E. Juan, J. Tsai, T. Murata and Y. Zhou, "Reduction methods for real time using delay time Petri nets," *IEEE Transactinos on Software Engineering.,* vol. 27, no. 5, pp. 422-448, 2001.

[48] C. Becker and A. Scholl, "A Survey on problems and methods in generalized assembly line balancing," *European Journal of Operations Research,* vol. 168, no. 3, pp. 694-715, 2006.

[49] S. Charharsooghi and N. Nahavandi, "Buffer allocation problem, a heuristic approach," *Scientia Iranica,* vol. 10, no. 4, pp. 401-409, 2003.

[50] I. Baybars, "A survey of exact algorithms for the simple assembly line balancing problem," *The Institute of Management Sciences,* vol. 32, no. 8, pp. 909-931, 1986.

[51] J. Jackson, "A computing procedure for a line balancing problem," *Management Science,* vol. 2, no. 1, pp. 261-271, 1956.

[52] D. Freeman, "A general line balancing model," in *1968*, Tampa, Florida, Proceedings 19th Annual Conference AIIE.

[53] A. Mastor, "An experimental investigation and compariative evalution of production line balancing techniques," *Management Sciences,* vol. 16, no. 11, pp. 728-746, 1970.

[54] T. Wee and M. Magazine, "An efficient branch and bound algorithm for assembly line balancing - Part 1: minimize number of work stations," Working Paper: #150, 1981.

[55] T. Wee and M. Magazine, "An efficient branch and bound algorithm for assembly line balancing - Part 2: maximize production rate," Department of Management Sciences, University of Waterloo, Waterloo, Ontario, 1981.

[56] M. Salveson, "The assembly line balancing problem," *Journal of Industrial Engineering,* vol. 6, no. 3, pp. 18-25, 1955.

[57] S. a. S. C. Thangavelu, "Assembly line balancing by zero-one integer programming," *AIIE Transactions,* vol. 3, no. 1, pp. 61-68, 1971.

[58] J. Patterson and J. Albracht, "Assembly line balancing: 0-1 programming with Fibonacci search," *Operations Research,* vol. 23, no. 1, pp. 166-174, 1975.

[59] M. K. R. a. S. R. Held, "Assembly line balancing - dynamic programming with precedence constraints," *Operations Research,* vol. 11, no. 1, pp. 442-459, 1963.

[60] E. Mansoor, "Assembly line balancing – an improvement on the ranked positional weight technique," *Journal of Industrial Engineering,* vol. 15, no. 1, pp. 73-77, 1964.

[61] E. Mansoor and M. Yadin, "On the problem of assembly line balancing," in *Development in Operations Research*, New York, Gordon and Breach, 1971, pp. 361-375.

[62] K. T. Q. a. O. N. Wei, "Estimation of buffer size using stochastic approximation methods," in *IEEE Proceedings of the 28th Conference on Decision and Control*, Tampa, Florida, 1989.

[63] H. Yamashita and T. Altiok, "Buffer capacity allocation for a desired throughput in production lines," *IIE Transactions,* vol. 30, no. 1, pp. 883-891, 1998.

[64] F. T. S. Chan and E. Y. H. Ng, "Comparative evaluations of buffer allocation strategies in a serial production line," *The International Journal of Advanced Manufacturing Technology.,* vol. 19, no. 11, pp. 789-800, 2002.

[65] W. M. Chow, "Buffer capacity analysis for sequential production lines with variable processing times," *International Journal of Production Research,* vol. 25, no. 8, p. 1183–1196, 1987.

[66] C. M. Liu and C. L. Lin, "Performance evaluation of unbalanced serial production lines," *International Journal of Production Research,* vol. 32, no. 12, p. 2897–2914, 1994.

[67] S. Gershwin and J. Schor, "Efficient algorithms for buffer space allocation," *Annals of Operations Research,* vol. 93, no. 1-4, pp. 117-144, 2000.

[68] E. Enginarlar, J. Li and S. Meerkov, "Buffer capacity for accommodating machine downtime in serial production lines," in *IEEE Conference on Decision and Control*, Orlando, Florida, 2001.

[69] E. Enginarlar, J. Li and S. Meerkov, "How lean can buffers be?," *IIE Transactions,* vol. 37, no. 1, pp. 333-342, 2005.

[70] M. Govil and M. Fu, "Queuing theory in manufacturing: a survey," *Journal of Manufacturing Systems,* vol. 18, no. 3, pp. 214-240, 1999.

[71] J. Li, D. Blumenfeld, N. Huang and J. Alden, "Throughput analysis of production systems: recent advances and future topics," *International Journal of Production Research,* vol. 47, no. 14, p. 3823–3851, 2009.

[72] S. Gershwin, "An efficient decomposition method for the approximate evaluation of production lines with finite storage space," in *Proceedings of 23rd Conference on Decision and Control*, Las Vegas, Nevada, 1984.

[73] J. Lim and S. Meerkov, "Homeogeneous, asymptotically reliable serial production lines: theory and case study," *IEEE Transactions on Automatic Control,* vol. 35, no. 5, pp. 524-534, 1990.

[74] V. Kouikoglou and Y. Phillis, "A serial finite unreliable queue model for production lines," in *Proceedings of the 31st Conference on Decision and Control*, Tuscon, Arizona, 1992.

[75] V. Kouikoglou and Y. Phillis, "An efficient discrete-event model for production networks of general geometry," in *Proceedings of the 29th IEEE Conference on Decision and Control*, Honolulu, Hawaii, 1990.

[76] V. Kouikoglou and Y. Phillis, "An exact efficient discrete-event model for production lines with buffers," in *Proceedings of the 28th IEEE Conference on Decision and Control*, Tampa, Florida, 1989.

[77] J. R. Morrison, "Deterministic Flow Lines with Applications," *IEEE Transactions on Automation Science and Engineering,* vol. 7, no. 2, pp. 228-239, 2010.

[78] S. Aghazadeh, S. Hafeznezami, L. Najjar and Z. Huq, "The influence of work-cells and facility layout on the manufacturing efficiency.," *Journal of Facilities Management,* vol. 9, no. 3, pp. 213-224, 2011.

[79] K. C. So, "Allocating Buffer Storages in a Flexible Manufacturing System.," *International Journal of Flexible Manufacturing Systems,* vol. 1, pp. 223-237, 1989.

[80] C. D. Senanayake and V. Subramaniam, "Analysis of a two-stage, flexible production system with unreliable machines, finite buffers and non-negligible setups.," *Flexible Services and Manufacturing Journal,* vol. 25, pp. 414-442, 2013.

[81] S. Leavengood, "Increasing the production capacity of a work cell using modeling and simulation," in *International Conference on Management of Engineering and Technology*, Portland, Oregon, 2002.

[82] E. Muth, "Analysis of closed loop conveyor systems, the discrete flow case," *AIIE Transactions,* vol. 6, no. 1, pp. 73-83, 1974.

[83] R. Tomastik, "Schedule flexible manufacturing systems for apparel production," *Proceedings of IEEE Transactions on Robotics and Automation,* vol. 12, no. 5, pp. 789-799, 1996.

[84] A. Ramirez-Serrano and B. Benhabib, "Supervisory Control of Multiworkcell Manufacturing Systems and Shared Resources," *IEEE Transactions on System, Man and Cybernetics - Part B,* vol. 30, no. 5, pp. 668-683, 2000.

[85] H. Ichikawa, "Simulating an applied model to optimize cell production and parts supply (Mizusumashi) for Laptop Assembly," in *2009 Winter Simulation Conference Proceedings*, Austin, Texas, 2009.

[86] R. Logendran and Y. Karim, "Design of manufacturing cells in the presence of alternative cell locations and material transporters," *Journal of Operational Research Society,* vol. 54, pp. 1059-1075, 2003.

[87] A. Youssef and H. A. ElMaraghy, "Optimal configuration selection for reconfigurable manufacturing systems," *International Journal of Flexible Manufacturing Systems,* vol. 19, pp. 67-106, 2007.

VITA

NAME: Francesca Schuler

EDUCATION:

Master of Science in Computer Engineering - Networking/Communications Emphasis, Illinois Institute of Technology- Chicago, Illinois – May, 2004

Master of Business Administration – Operations Management Emphasis
DePaul University - Chicago, Illinois, August, 2000

Master of Science in Mechanical Engineering – Robotics/Manufacturing Emphasis, University of Illinois, Chicago, Illinois, May, 1997

Bachelor of Science in Mechanical Engineering
University of Illinois - Urbana, Illinois - May, 1995

PUBLICATIONS:

F. Schuler and H. Darabi. “Buffer clustering policy for sequential production lines with deterministic processing times,” *Journal of Flexible Services and Manufacturing.* Tentatively Accepted.

F. Schuler and B. Connor. “Securing your critical infrastructure against cyber attacks,” in 2014, Washington D.C. *Proceedings of the Public Safety Broadband Summit and Expo*.

F. Schuler and H. Darabi, “Supervisory control and data collection policies for a distribution center modeled as a discrete event system,” in 2010, Chicago, Illinois, Proceedings of the *IEEE International Conference on Networking, Sensing and Control.* pp. 177-182.

H. Darabi, L. Baghdasyran, F. Schuler, and A. Schaller, “A modeling and optimization management tool for large-scale supply chain networks,” *International Journal of Industrial and Systems Engineering,* vol. 5, no.1, pp. 48 – 78, 2010.

H. Darabi, L. Baghdasyran, F. Schuler, and A. Schaller, “Evaluation of supply chain performance with information accuracy considerations,” in 2006, Orlando, Florida, Proceedings of the Institute Industrial Engineers Annual Conference.

F. Schuler, J. Liu, C. Zhou, and M. Toloo, “Technology insertion in a supply chain organization: experiences, challenges and lessons learned,” in 2006, Chicago, Illinois, Proceedings of the National Manufacturing Week.

M. Toloo, F. Schuler, and J. Liu, “Integrating discrete-event simulation into a supply chain organization’s transportation strategy,” in 2005, Orlando, Florida, Proceedings of the Winter Simulation Conference.

A. Schaller, L. Baghdasaryan, F. Schuler, and H. Darabi, “Experience and challenges in creating mathematical models that facilitates simultaneous product and supply chain design, in 2005, Atlanta, Georgia, Proceedings of the Annual Industrial Engineering Research Conference.

H. Darabi, L. Baghdasyran, F. Schuler, A. Schaller, ”A graph based method for handling model flexibility of supply chains,” in 2005, San Francisco, California, Proceedings of the Institute of Operations Research and Management Science Conference.

A. Schaller, F. Schuler, and C. Zhou, “Impact of digital six sigma manufacturing practices on cost of poor manufacturing quality,” in 2005, Chicago, Illinois, Proceedings of the National Manufacturing Week.

A. Anant, J. Baik, F. Ruffolo, and N. Eickelmann, “Applying simulation tools for quantitative management of software process improvement,” in 2003, Chicago, Illinois, Proceedings of Chicago Software Process Improvement Network.

A. Anant, J. Baik, N. Eickelmann, and F. Ruffolo, “An empirical study of modifying the fagan inspection process and the resulting main effects and interaction effects among defects found, effort required, rate of preparation and inspection, number of team members and product,” in 2002, Greenbelt, Maryland, Proceedings of the 27th Annual NASA Goddard Software Engineering Workshop.

PATENTS:

F. Schuler, E. Chen, K. Reitsma, J. Marocchi, L. Whitelock. 2015. Method and apparatus for receiving a data stream during an incident. US 9,226,124. Issued December 2015.

T. Miller, S. Glass, D. Klein, W. Mao, F. Schuler. 2015. Secure ad hoc communication systems and methods across heterogeneous systems. US 9,178,893. Issued November 2015.

E. Chen, A. Agulnik, F. Liang, W. Mao, T. Miller, F. Schuler. 2015. Method and apparatus for managing quality of service settings for group communications. US 9,173,134. Issued October 2015.

W. Mao, T.Miller, F.Schuler. 2015. Method and apparatus for maintaining priority and quality of service across multi-user devices. US 9,037,145. Issued May 2015.

T. Miller, E. Chen, W. Mao, F. Schuler, S. Vanswol. 2014. Method and apparatus for ensuring critical resource allocation for group calls made in a push-to-talk communication environment. US 8,886,244. Issued November 2014.

F. Schuler; I. Ahmed, K. Jonnalagadda. 2013. Method and apparatus for determining a physiological parameter using a fingerprint sensor on a portable electronic device. US 8,605,961. Issued December 2013.

I. Ahmed, J. Farshi, K. Jonnalagadda, F. Schuler. 2013. Method and apparatus for determining blood oxygenation using a mobile communication device. US 8,503,712. Issued August 2013.

J. Gyorfi, F. Schuler, E. Buhrke, J. Lopez, H. Yu. 2013. Mobile Virtual and Augmented Reality System. Korea 10-2010-7023341. Issued January 2013.

F. Schuler, K. Jonnalagadda. 2012. Portable electronic device and method of power management for same to accommodate projector operation. US 8,231,233. Issued July 2012.

F. Schuler, J. Gyorfi, S. Mok. 2011. Method and apparatus for providing a prioritized list of display devices for display of a media file. US 8,078,230. Issued December 2011.

F. Schuler, D. Hong, K. Jonnalagadda, K. Kaustubh, A. Khawand, J. Lacal, P. Ramadas. 2011. Method for autonomously monitoring and reporting sound pressure level exposure for a user of a communication device. US 7,983,426 B2. Issued July 2011.

F. Schuler, K. Jonnalagadda. 2011. Dynamic updating of product profiles for active lifestyles. US 7,924,158 B2. Issued April 2011.

J. Danvir, K. Jonnalagadda, F. Schuler. 2010. Monitoring for radio frequency enables items based on activity profiles. US 7,834,762 B2. Issued: November 2010.

F. Schuler, K. Jonnalagadda. 2010. Method and apparatus for predictive, context-aware, and networked exposure time monitoring. US 7,818,142 B2. Issued: October 2010.

L. Grajales, M. Krizik, I. Nicolaescu, J. Preston, F. Schuler, M. Toloo. 2010. Method and system for facilitating command of a group. Korea 10-09888988. Issued: October 2010.

F. Schuler, K. Jonnalagadda, X. Luo. 2010. Monitoring for radio frequency enabled items based on shared group activity profiles. US 7,688,208 B2. Issued: March 2010.

APPENDIX I: PETRI NET TOOLS ASSESSED

Tool	Website	Supports Petri Nets with Time	Supports Performance Analysis	Supports Colored Petri Nets	Interchange Import & Export	Graphical Editor	Tool Support	Initial Observed Release	Last Observed Release	Operating System	Academic/ Commercial
AlPiNA	http://alpina.unige.ch/	No	Yes	Yes	Yes	Yes	Limited	2000	2011	Win Vista, XP, 7 , Linux, Mac OS X	Academic
ARP	http://www.ppgia.pucpr.br/~maziero/doku.php/software:arp_tool	Yes	Yes	No	No	No	None	1988	1990	None	Academic
CoopnBuilder	http://coopn.wordpress.com/	No	No	No	No	Yes	Limited	2006	2006	Win Vista, XP, Linux, Mac OS X	Academic
CPN-AMI	http://move.lip6.fr/software/CPNAMI/	No	No	Yes	Yes	Yes	Full	1994	2010	Win XP, Linux, Mac OS X	Academic
CPN Tools	http://cpntools.org/	Yes	Yes	Yes	Yes	Yes	Full	2000	2015	Win7+, Linux	Academic
Disc Software Platform	http://www.disc-project.eu/software_platform.html	No	No	No	Yes	Yes	Limited	2008	2011	WinXP	
ePNK	http://www2.imm.dtu.dk/~eki/projects/ePNK/	No	No	No	No	Yes	Limited	2010	2012	Win7+	Academic
ExSpect	http://www.exspect.com/	Yes	Yes	No	Yes	Yes	None	1999	2000	Win XP	Academic
GDToolkit	http://www.dia.uniroma3.it/~gdt/gdt4/index.php	No	No	Yes	No	No	Limited	1997	2008	Win Vista, XP, Linux	Academic
GreatSPN	http://www.di.unito.it/~greatspn/index.html	Yes	Yes	Yes	No	Yes	Limited	1993	2001	Linux, Solaris, Mac OS X	Academic
Helena	http://helena.cnam.fr/	No	No	Yes	Yes	No	Limited	1991	2013	Linux	Academic
HiQPN-Tool	http://ls4-www.informatik.uni-dortmund.de/QPN/	No	No	Yes	Yes	Yes	Limited	2007	2009	Win XP, Linux, Mac OS X	Academic
HISIm	http://sourceforge.net/projects/hisim/	Yes	No	Yes	No	Yes	Limited	2007	2007	Win XP Linux	Academic
HPSim	http://www.winpesim.de	Yes	Yes	No	No	Yes	None	1999	2001	Win2000	Academic
INA	http://www2.informatik.hu-berlin.de/~starke/ina.html	Yes	Yes	Yes	Yes	No	Limited	1992	2003	Solaris,Win XP Linux	Academic
Income Suite	http://www.get-process.com	Yes	Yes	Yes	No	Yes	Full	1991	2008	Solaris, Win XP Linux	Both

Tool	Website	Supports Petri Nets with Time	Supports Performance Analysis	Supports Colored Petri Nets	Interchange Import & Export	Graphical Editor	Tool Support	Initial Observed Release	Last Observed Release	Operating System	Academic/ Commercial
JARP	http://jarp.sourceforge.net/	No	No	No	Yes	Yes	Limited	2003	2006	Win XP Linux	Academic
JFern	http://sourceforge.net/projects/jfern	Yes	Yes	No	Yes	Yes	Limited	1998	2009	Win XP Linux	Academic
JPetriNet	http://jpetrinet.sourceforge.net/	Yes	No	No	No	Yes	Limited	2003	2003	Win XP Linux	Academic
Kontinuum (Web and Flo)	http://www.webandflo.com/	Yes	Yes	Yes	No	Yes	Full	2000	2014	Win 7+	Both
LoLA	http://service-technology.org/lola/	No	No	No	No	No	Limited	2008	2014	Solaris, Win7+, Linux	Academic
Maria	http://www.tcs.hut.fi/Software/maria/	No	No	Yes	Yes	No	Limited	1999	2005	Solaris, Win XP, Linux, Mac OS X	Academic
MISS-RdP	http://www.laas.fr/~robert/missrdp.html	Yes	Yes	Yes	No	Yes	None	1992	1998	Win 2000 Linux	Commercial
mist2	https://github.com/pierreganty/mist	No	No	No	No	No	Limited	2005	2015	Linux, MAC OS X	Academic
Netlab	http://www.irt.rwth-aachen.de/index.php?id=101	No	No	No	Yes	Yes	Limited	2007	2008	Win XP	Both
ORIS	http://www.oris-tool.org/	Yes	Yes	No	No	Yes	Limited	2004	2013	Win 7+, Linux, Mac OS X	Academic
P3	http://www.sfu.ca/~dgasevic/projects/P3net/	No	No	No	Yes	Yes	Limited	1997	2004	Win XP	Academic
PACE	http://www.ibepace.com/	Yes	Yes	Yes	No	Yes	Full	2001	2008	Win XP, Vista, 7	Both
PEP	http://peptool.sourceforge.net/	Yes	No	Yes	Yes	Yes	None	1998	2004	Linux	Academic
PetitPetri	http://scg.unibe.ch/download/petitpetri/	Yes	No	No	No	Yes	None	2008	2009	Win XP Linux, Mac OS X	Academic
Petri-lld	http://apps.sourceforge.net/mediawiki/petrilld/index.php?title=Main_Page	No	No	No	No	Yes	Limited	2005	2006	Win XP Linux, Mac OS X	Academic
Petri Net Kernel	http://www2.informatik.hu-berlin.de/top/pnk/	No	No	Yes	Yes	Yes	None	1999	2002	Win XP, Linux	Academic

Tool	Website	Supports Petri Nets with Time	Supports Performance Analysis	Supports Colored Petri Nets	Interchange Import & Export	Graphical Editor	Tool Support	Initial Observed Release	Last Observed Release	Operating System	Academic/ Commercial
Petri .NET Simulator	http://www.qweas.com/download/home_education/science/petri_net_simulator.htm	Yes	Yes	No	No	Yes	None	2006	2006	Win XP	Both
Petri Net Toolbox	http://www.mathworks.com/products/connections/product_detail/product_35741.html	Yes	Yes	No	Yes	Yes	Limited	2005	2009	Win XP	Both
PetriSim	http://staff.um.edu.mt/jskl1/petrisim/	Yes	No	Yes	No	Yes	Limited	1997	2004	Win XP	Academic
Petruchio	http://petruchio.informatik.uni-oldenburg.de/	Yes	Yes	No	Yes	Yes	Limited	2010	2010	Linux, MAC OS X, Win XP, 7	Academic
Platform Independent Petri Net Editor 2	http://pipe2.sourceforge.net/	No	No	No	Yes	Yes	Limited	2002	2007	Win XP Linux	Academic
PNEditor	http://www.pneditor.org/	No	No	No	Yes	Yes	Limited	2010	2014	Win 7+, MAC OS X, Linux	Academic
PNetLab	http://www.automatica.unisa.it/PnetLab.html	Yes	Yes	Yes	Yes	Yes	Limited	2006	2011	Win XP	Academic
PNML Framework	http://pnml.lip6.fr/	No	No	Yes	Yes	No	Limited	2008	2015	Win 7+, Linux, Mac OS X	Academic
PNSim	http://www.elyros.com/PNSim/index.html	No	No	No	No	Yes	None	2003	2006	Win XP	Academic
PNtalk	http://perchta.fit.vutbr.cz:8000/projekty/12	Yes	Yes	Yes	No	Yes	Limited	1993	2006	Solaris, Win XP	Academic
Poses++	http://www.gpc.de/	No	No	Yes	No	No	Limited	1996	2011	Win Vista, Linux	Academic
PROD	http://www.tcs.hut.fi/Software/prod/	No	No	Yes	No	No	None	2001	2006	Win XP Linux	Academic
ProM framework	http://promtools.org/prom6/	Yes	Yes	Yes	Yes	No	Limited	2008	2010	Win 7+, Linux, Mac OS	Academic
QPME	http://se.informatik.uni-wuerzburg.de/tools/qpme/	No	No	Yes	Yes	Yes	Limited	2005	2015	Win 7+, Linux, Mac OS X	Academic
Renew	http://www.renew.de/	Yes	No	Yes	Yes	Yes	Limited	1999	2015	Win7+, Linux, Solaris, Mac OS X,	Academic
Romeo	http://romeo.rts-software.org/	Yes	No	No	No	Yes	Limited	2004	2016	Win7+, Linux, Mac OS X	Academic

Tool	Website	Supports Petri Nets with Time	Supports Performance Analysis	Supports Colored Petri Nets	Interchange Import & Export	Graphical Editor	Tool Support	Initial Observed Release	Last Observed Release	Operating System	Academic/ Commercial
RSOFT	http://www.rsoftdesign.com/applications.php?sub=Sort+by+Application&itm=Device	Yes	Yes	Yes	No	Yes	Limited	2003	2003	Win XP Linux	Both
SIPN-Editor	http://www.graphviz.org/	No	No	No	No	Yes	Limited	2002	2015	Win 7+, Linux, Mac OS, Solaris	Academic
SNAKES	https://www.ibisc.univ-evry.fr/~fpommereau/SNAKES/	No	No	Yes	Yes	No	Limited	2008	2013	Linux	Academic
Snoopy	http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy	Yes	No	Yes	Yes	Yes	Limited	2005	2016	Win 7+, Linux, Mac OS X	Academic
SPNP	http://people.ee.duke.edu/~kst/	No	No	Yes	No	No	None	1997	1999	Win 2000, Linux, Solaris	Both
SYROCO	http://w3.univ-tlse1.fr/irit/soc/coo/Syroco.html	Yes	Yes	No	Yes	Yes	None	1995	2001	Linux	Academic
TAPAAL	https://launchpad.net/tapaal	Yes	Yes	No	Yes	Yes	Limited	2009	2015	Win 7+, Linux, Mac OS X	Academic
TimeNET	https://www.tu-ilmenau.de/de/sse/timenet/	Yes	Yes	Yes	No	Yes	Limited	2007	2015	Win 7+, Linux	Both
Tina	http://www.laas.fr/tina/	Yes	No	No	No	Yes	Limited	2004	2016	Win 7+, Linux, Mac OS X, Solaris	Academic
VisualPetri	http://sourceforge.net/projects/visual-petri/	No	No	No	No	Yes	None	2004	2006	Win XP	Academic
Visual Object Net ++	http://www.r-drath.de/Home/Visual_Object_Net++.html	Yes	Yes	No	No	Yes	Limited	1997	2004	Win XP, Vista	Academic
WinPeSim	http://www.winpesim.de/default.html	Yes	Yes	No	No	Yes	Limited	2003	2006	Win XP	Academic
WOFLAN	http://www.win.tue.nl/woflan/doku.php	No	No	No	Yes	Yes	None	2000	2003	Win XP, MAC OS X	Academic
WoPeD	http://sourceforge.net/projects/woped/	No	No	No	Yes	Yes	Limited	2007	2014	Win 7+, Linux, Mac OS X	Academic
Yasper	http://www.yasper.org/	Yes	Yes	Yes	Yes	Yes	Limited	2005	2010	Win XP	Both
YAWL	http://yawlfoundation.org/	No	No	Yes	Yes	Yes	Full	2004	2014	Win7+, Linux, Mac OS X	Academic