Dynamic Control of Flexible Queueing Networks with Application to Shipbuilding

by

Fang Dong

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Industrial and Operations Engineering)
in The University of Michigan
2013

Doctoral Committee:

Associate Professor Mark P. Van Oyen, Chair
Assistant Professor David J. Singer, Co-Chair
Assistant Professor Matthew Collette
Professor Jack Hu
Professor Lawrence M. Seiford

ACKNOWLEDGEMENTS

Completing graduate school at the University of Michigan has been an amazing life journey. I could not have achieved any of this without the help and support from my advisors, friends, colleagues, and family.

First, I would like to thank Mark P. Van Oyen and David J. Singer, who I was very fortunate to have had as advisors. They have trusted and believed in me, and, most importantly, they gave me the freedom to explore the unknown. They did all they could to provide various resources and opportunities, such as enabling me to present at conferences, visit shipyards, and meet other influential colleagues. My advisors not only provided me with professional support, but friendship, and it has been a joy working with them. Without my advisors, I would not be the person I am today.

Additionally, I would like to thank my friends and colleagues Jonathan Eugene Helm and Jivan Deglise-Favre-Hawkinson. They have supported me while acquiring my Ph.D and will always remain my best friends. I also want to thank my dear friends Hoda Parvin and Shervin AhmadBeygi, who supported me in the early years of my Ph.D., my friend and mentor Olof H. Minto, who has always helped me with everything in and out of school, and Rod Capps, Chris Konrad, Tina Blay, and Wanda Dobberstein, the wonderful staff who met my endless requests.

I was also fortunate enough to have a number of outstanding professors who taught and inspired me throughout my research. Thanks to my committee mem-

bers, Professor Lawrence M. Seiford, Professor Jack Hu, and Professor Matthew Collette. Thank you to Professor Goker Aydin and Professor Volodymyr O. Babich, who created inspiring lectures in the classroom and were fun colleagues to be with outside of school as well. I want to thank Professor Michael H. Veatch, who is a great mathematician and collaborates with me on shipbuilding research, which really increases my confidence in our research. Thanks to Professor Richard L. Storch, I started my shipbuilding research by reading his *Ship Production* book. He is also another operations researcher who loves ships! I would like to thank Kelly Cooper from Office of Naval Research, who funded this research project and also supported my research by providing ideas from shipbuilding perspective and guiding me for the future research.

My colleagues from the Naval Architecture and Marine Engineering Department were also integral influences in my research. Thank you Morgan C. Parker, Thomas A. McKenney, Brian Cuneo, Jason D. Strickland, Joshua Knight, Nathan D. Niese, Justin W. Gillespie, and Douglas Rigterink. I had my best year in NEEC working with all of you. Thank you all for creating such a friendly and relaxing work atmosphere, and strongly support one another.

Last, but not least, I would like to thank my family. My parents, Shijun Dong and Weiping Zhang, who gave me the strength to come halfway across the world to pursue my dream. You both were always the big part of my dream that I am fighting for! I also want to thank my dear aunts Weihong (Melissa) Zhang and Weiqun (Christina) Zhang, and my uncle David Wang. You are my family in the U.S. and have always made me feel at home. I could not have achieved any of this without your unconditional love and support. Thank you!

This dissertation is the result of work partially sponsored by the Office of Naval

Research, ONR Grant N00014-08-1-0579, and National Science Foundation, NSF grant DMI-0542063. This support is gratefully acknowledged.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS . . . . . **ii**

LIST OF FIGURES . . . . . **vii**

LIST OF TABLES . . . . . **x**

CHAPTER

I. Introduction 1

1.1 Problem Overview 1
1.2 The Shipbuilding Process 3
1.3 Research Structure 5
1.3.1 Methodologies 7
1.4 Dissertation Outline 8

II. Dynamic Control of the Flexible Block Assembly Processes 9

2.1 Introduction 9
2.1.1 The Concept of Manufacturing Flexibility 10
2.1.2 Block Assembly Process with Operational Flexibility 12
2.2 The "N" Structure Queueing Network 16
2.2.1 Model Description 16
2.2.2 Literature Survey 17
2.2.3 Markov Decision Process Formulation 19
2.2.4 Numerical Examples 23
2.3 Computational Experiments 24
2.3.1 Selection of Heuristic Control Policies: The Base Set 25
2.3.2 Stability Analysis 28
2.3.3 Method of Investigation for the Analysis of the System Congestion and Mix of Job Types 29
2.3.4 Benchmarking the Base Set of Heuristic Policies with Respect to Optimality 32
2.4 System Analysis for a New Analytical Threshold Policy 34
2.4.1 Two Birth-Death Processes Useful for Estimating the Threshold . 35
2.4.2 Cost Function 36
2.4.3 The Sub-optimality Gap of the Analytical Threshold 38
2.5 A Regression Based Approximation of the Threshold Level: The Regression Threshold Policy 39
2.6 Conclusion and Future Work 41
2.7 Appendix 43
2.7.1 Definition of Event Operators 43
2.7.2 Relative Value Iteration Equations 49
2.7.3 Proof of the Stability Region 50

III. Capacity Planning and Control of Flexible Hull Construction Processes under CONWIP Discipline 52

3.1 Introduction 52
3.1.1 Methodology Behind Hull Construction Planning in Shipbuilding . 53
3.1.2 CONWIP Release Policy 54
3.1.3 The CONWIP Model of Shipbuilding 55
3.2 Control of Flexible Workshop under CONWIP Discipline: A Simulation Model 57
3.2.1 Simulation Model 57
3.2.2 Simulation Structure 60
3.2.3 Result Analysis 64
3.3 Dynamic Two-Loop CONWIP Model 71
3.3.1 Markov Decision Process Formulation 73
3.3.2 Control of Flexible Workshop 77
3.3.3 Capacity Planning of Flexibility 82
3.4 Conclusion and Future Work 87

IV. Two-Stage Queueing Network for Outfitting Planning and Control 89

4.1 Introduction 89
4.1.1 Existing Outfitting Obstacles 89
4.1.2 Outfitting Process 90
4.1.3 State of The Art: Research on Ship Outfitting Planning 91
4.2 Strategic Level Model: An Open Queueing Network Model 92
4.2.1 Model Formulation 93
4.2.2 Analytical Results 94
4.2.3 Numerical Study 96
4.3 Execution Level Model: A Closed Queueing Network Model 103
4.3.1 Literature Review 105
4.3.2 Markov Decision Process Formulation 106
4.3.3 Numerical Examples 107
4.4 A Static Model for Mean Value Analysis 111
4.4.1 Recursive Function and Iteration 112
4.4.2 Numerical Example 113
4.4.3 MVA vs MDP 115
4.5 A Regression Based Threshold Policy 117
4.5.1 Conditions for Threshold Policy Type 117
4.5.2 The Regression Model 119
4.5.3 Design of the Heuristic 124
4.6 Conclusion and Future Work 125

V. Conclusions 128

BIBLIOGRAPHY **131**

LIST OF FIGURES

Figure

1.1 Hull Blocks . . . 3

1.2 Shipyard Layout . . . 5

2.1 Flat Panel Assembly line . . . 14

2.2 Curved Panel Assembly Line . . . 15

2.3 Flexible Block Assembly Process Paradigm . . . 15

2.4 Optimal Control Policy Structure: Impact of Holding Cost . . . 24

2.5 Optimal Control Policy Structure: Impact of Processing Time . . . 25

2.6 Threshold Policy . . . 27

2.7 Analysis of System Congestion Factor when $h_1\mu_{21} > h_2\mu_{22}$ . . . 30

2.8 Analysis of System Congestion Factor when $h_1\mu_{21} < h_2\mu_{22}$ . . . 31

2.9 The birth-death process approximation of the flat block queue subsystem . . . 35

2.10 The birth-death process approximation of the curved block queue subsystem . . . 35

3.1 Hull Planning Steps . . . 53

3.2 Flexible Shipbuilding CONWIP Model . . . 56

3.3 CONWIP Shipbuilding Simulation Model . . . 57

3.4 Simulation Structure . . . 61

3.5 Event Departure of Flat Block . . . 63

3.6 Event Departure of Curved Block . . . 64

3.7 Event Departure from Final Erection Process . . . 65

3.8 Case 1: Ship Completion Time Comparison . . . 67

3.9 Case 2: Ship Completion Time Comparison . . . 68

3.10 Case 1: Drydock Utilization Comparison . . . 69
3.11 Case 2: Drydock Utilization Comparison . . . 70
3.12 Case 1: Block Buffer Size . . . 71
3.13 CONWIP MDP Model . . . 72
3.14 Optimal Control Policy of Flexible Workshop . . . 79
3.15 Optimal Control Policy for Sequence "FFCCCFFF" . . . 81
3.16 Optimal Control Policy: Impact of Change of Capacity . . . 82
3.17 Flexible System: Average Profit with Change of θ_C . . . 85
3.18 Inflexible System: Average Profit with Change of θ_C . . . 85
3.19 Percentage Improvement by Flexibility . . . 86
4.1 Hull Construction Processes . . . 92
4.2 Strategic Two-stage Outfitting Model . . . 93
4.3 Test Suite 1 . . . 98
4.4 Test Suite 2 . . . 99
4.5 Case 10: $\alpha^* = 0$ . . . 102
4.6 Two Stage Outfitting Model . . . 104
4.7 Outfitting MDP Numerical Example 1: $\gamma > min\{C_{SO}, C_{AO}\}$ . . . 108
4.8 Outfitting MDP Numerical Example 2: $\gamma > min\{C_{SO}, C_{AO}\}$ . . . 109
4.9 Outfitting MDP Numerical Example 3: $\gamma < min\{C_{SO}, C_{AO}\}$ . . . 110
4.10 Outfitting MDP Numerical Example 4: Change of C_{SO} and C_{AO} . . . 110
4.11 Mean Value Analysis Model . . . 112
4.12 MVA Numerical Example . . . 114
4.13 MVA vs MDP: K = 3 . . . 115
4.14 MVA vs MDP: K = 10 . . . 116
4.15 Threshold Policy Types . . . 118
4.16 Type 1 Threshold Policy Regression Model . . . 119
4.17 Regression Structure . . . 120

4.18 MVA, MDP, and Regression Comparison: K = 6 123

4.19 Heuristic Design 124

LIST OF TABLES

Table

2.1 MDP Numerical Test Cases . . . 23

2.2 Test-suite for the Congestion Factor . . . 29

2.3 Large Test Suite Design . . . 33

2.4 Optimality Gap(%) with $h_1\mu_{21} \leq h_2\mu_{22}$ (*FBS*, *Cμ rule*, and *Optimal Threshold* policies are identical) . . . 34

2.5 Optimality Gap(%) with $h_1\mu_{21} > h_2\mu_{22}$ . . . 34

2.6 Optimality Gap(%) with $h_1\mu 21 > h_2\mu 22$ . . . 41

3.1 CONWIP Model Simulation Test-Suite . . . 66

3.2 Notations . . . 74

3.3 CONWIP MDP model Test Case . . . 78

3.4 Flexibility Capacity Planning Test Case . . . 84

4.1 Notation for Outfitting Model . . . 94

4.2 Test-suite 1 . . . 97

4.3 Test-suite 2 . . . 99

4.4 Test-suite 3: Processing Rate . . . 100

4.5 Test-suite 3 . . . 100

4.6 Test-suite 3 Results . . . 100

4.7 Numerical Example Test Cases . . . 107

4.8 MVA Test-Suite . . . 113

4.9 Regression Test-Suite . . . 121

4.10 Regression Test-Suite . . . 122

4.11 Heuristic, Regression Threshold, and Mean Value Analysis Comparison in Percentage Error 125

CHAPTER I

Introduction

U.S. shipbuilders produce the finest warships in the world, but cost growth is eroding the purchasing power of the Navy.

–Office of the Deputy Under Secretary of Defense[38]

For the past decades, the U.S. shipbuilding industry has been recognized as not competitive as the leading international shipbuilders. The majority of contemporary research in shipbuilding has been centered on technology, such as welding and cutting, or design oriented research. Ship production, as an essential execution process of building a ship, should also draw on the world-class manufacturing and operations philosophies and techniques. This research focuses on introducing the most recent operations research techniques to shipbuilding, to design a more robust and efficient system, and to improve ship production planning and control. At the same time, shipbuilding also motivates us to identify and solve new problems in operations research. The models and methodologies developed in this research also can be applied to improve other systems in manufacturing and serves industries.

1.1 Problem Overview

Shipbuilding is a unique industry that uses a wide variety of manufactured components and requires a large number of workers possessing various skills as well as

specialized facilities. U.S. shipbuilders have been preeminent in the production of military vessels; However, in the world merchant ship market, they have not been competitive since the Second World War, with the exception of small specialized crafts. Over the last decade, the U.S. shipbuilding industry has improved significantly on productivity as a result of Navy and industry initiatives and investment. The technology gap between the U.S. industry and leading international shipbuilders is closing. However, there are still large technology gaps in some U.S. shipyards that present opportunities to make further substantial improvements, particularly in the preproduction functions which include design, production engineering, and planning [38].

To improve the efficiency of shipbuilding methods, it is essential to investigate and measure how current problems affect overall productivity. Problems in current naval ship production include construction commencing with immature designs, demand changes due to new ship specialization, material and other schedule delays, inexperienced labor, ineffective production control, lack of design for supply chain resilience, high cost due to delay, and lack of standard parts. These issues result in high variability in production workload and low facility utilization in ship production. Therefore, improvement in the system may be achieved by (1) introducing a methodology that enhances the system's robustness; (2) developing a model that can capture system variability; and (3) designing a smart control and planning policy that has quicker response to the change of production as well as delay. To introduce new technology to shipbuilding, one must first understand the basic process of shipbuilding.

1.2 The Shipbuilding Process

There are two major processes in shipbuilding: hull construction and outfitting. Hull construction includes all the activities associated with fabricating and assembling the hull, while outfitting refers to the process of fabrication and installation of nonstructural components.

Hull construction normally uses the block construction method, which is also called *Modularization* . This construction method has been used in ship production since World War II. The ship is divided geometrically into blocks. Depending on the ship size and type, a typical ship might consist of 100 blocks. Blocks are assembled in a block assembly area and then lifted to the drydock for the final erection. The main hull construction workflow contains the processes of part fabrication, sub-block assembly, block assembly, and final hull erection. Figure 1.1 illustrates a layout of blocks used for a bulkcarrier.

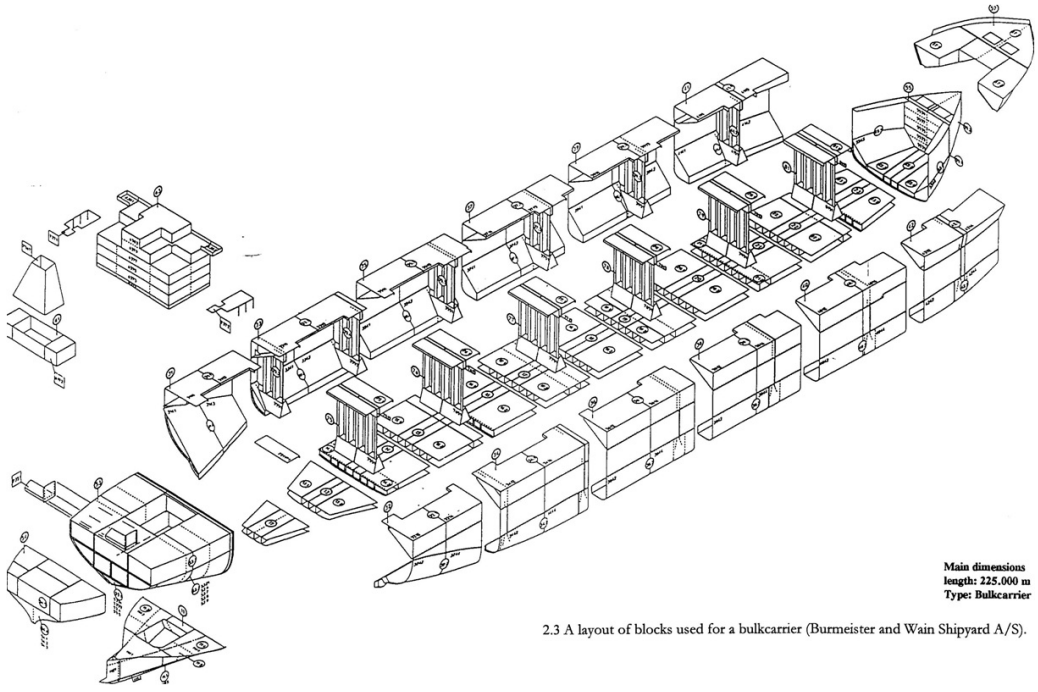


Figure 1.1: Hull Blocks

Outfitting in shipbuilding includes everything from painting to installing pumps and piping systems, the main propulsion system, electrical system, air conditioning (HVAC), and so forth. Most outfitting work can be performed during or after any of the stages in hull construction workflow. It is always easier and more cost efficient to process the outfitting at an early stage since there is easier access and more feasibility in using large machinery. The scheduling of outfitting and hull construction are usually separate processes, and scheduling of outfitting processes depends greatly on human experience. Nonetheless, it remains very challenging to integrate outfitting activities in hull construction.

Figure 1.2 depicts the Oshima shipyard layout. Oshima shipbuilding is a Japanese shipbuilding company which ranked No. 7 in Gross Tonnage among the world shipbuilders in 2012. This layout shows the basic workflow in a typical shipyard beginning with hull construction process and some of the outfitting processes, such as painting and piping. Some outfitting is additionally carried out during sub-assembly and block assembly. There are two outfitting quays in this shipyard, indicating that a large amount of outfitting activities will be performed after hull construction.

Overall, shipbuilding is a complex and massive manufacturing process with various labor skills, and unique processing and equipment. Therefore, it is no surprise that a large amount of planning, scheduling, and decision-making is necessary at almost every level of the shipbuilding process. There are many aspects and areas in shipbuilding that can be improved by operations methodology. A fundamental problem is at which level to attack problems and what type of approach to use. In the following section, the research structure will be laid out.

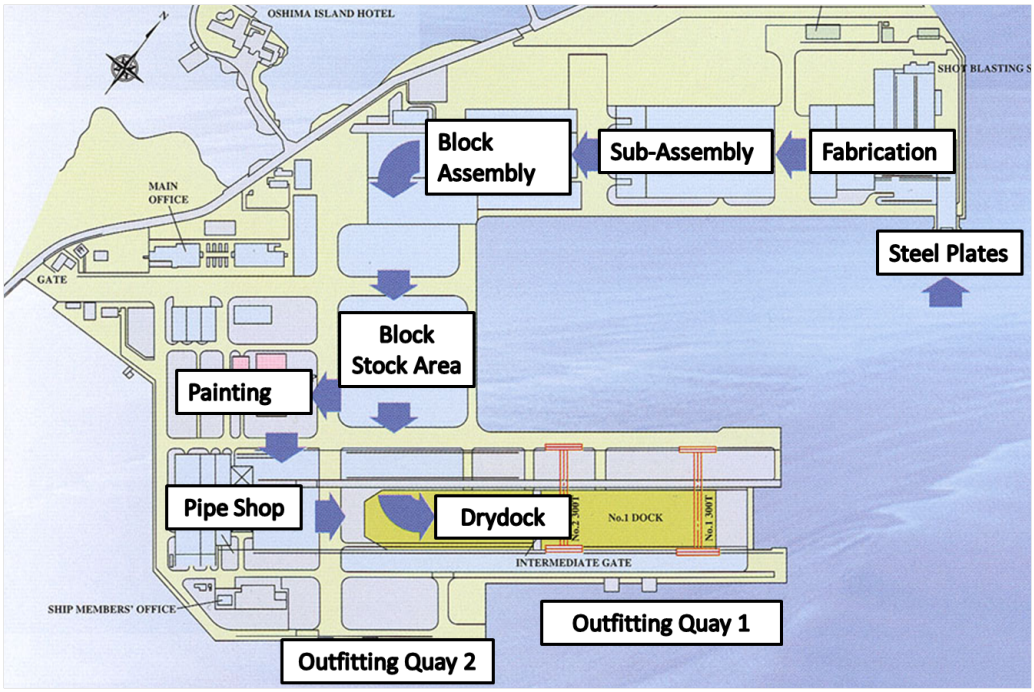


Figure 1.2: Shipyard Layout

1.3 Research Structure

There are three models detailed here that target on both the hull construction process and outfitting process:

Model 1: Flexible Block Assembly Process (Chapter II)

Model 2: Hull Construction under CONWIP Planning Policy (Chapter III)

Model 3: Two-stage Closed Queueing Network for Outfitting Planning (Chapter IV)

Model 1 and model 2 focus on hull construction, while model 3 emphasizes on the outfitting planning process.

Model 1 introduces operational flexibility to the block assembly process, allowing some block assembly workshops to process different types of blocks rather than solely one. Operational flexibility entails that the flexible workshop can be controlled to process different blocks in a dynamic manner. The operational flexibility concept is based upon current shipbuilding facilities, and it does not require investment

in any new equipment or facilities. Flexibility can bring robustness to the system to combat common problems in shipbuilding like demand change, large variability, and low facility utilization. Another critical problem is how to control the flexible resource (i.e. which job should be assigned to the flexible workshop). Flexibility undoubtedly improves the system's robustness, but it additionally makes the system more complex. The goal of this research is to develop a simple, applicable, and efficient policy to control the flexible resource.

Model 2 is focused at the planning level of hull construction. Most shipyards are currently using the Manufacturing Resource Planning (MRP) system to plan and schedule hull construction. In recent decades, a lot of new planning and scheduling methodologies have been developed, such as the very popular Lean Manufacturing. However, Lean Manufacturing is limited in its ability to schedule and plan at the strategic level and thus is difficult to apply to the American shipbuilding environment. Additionally, Lean methodologies do not provide any new planning methodology with fundamental improvement. On the contrary, Constant Work In Process (CONWIP) is a combination of push (MRP) and pull (JIT) system, that is easier to implement and has more robust control. It can also be modeled as a closed queueing network to capture the variability of the system. Therefore, the CONWIP discipline is inducted at the strategic level in order to improve the current planning and scheduling methodology of hull construction. At the execution level, flexible block assembly is maintained as part of the hull construction process. This is a very innovative model, mainly due to its flexibility under CONWIP control. There are three objectives in this model: (1) Discovering the optimal CONWIP level; (2) How to control the flexible resource under CONWIP discipline; (3) How much capacity should be invested in flexibility.

Model 3 targets the planning and control of outfitting activities. Current planning of outfitting depends mostly on previous experience. A mathematical model of outfitting planning can provide analytical information to improve the current outfitting technique. Two models are developed for different levels of the outfitting process. A static model provides the percentage of outfitting work that should be performed at each stage, and a dynamic model provides a control policy at the execution level, indicating when to process the outfitting work given the current status of the system. Both models are stochastic models, which capture the high variability of outfitting processes.

1.3.1 Methodologies

The main methodologies used in this research include Queueing Networks, Markov Decision Processes, and Discrete-Event Simulation.

Queueing Networks:

Queueing networks are used to model the aforementioned problems. Queueing network can capture the variability in shipbuilding assuming different distribution processing times. Exponential distribution of the block processing time is assumed in the model due to the great processing time variability. Model 1 uses an open queueing network with Poisson Arrivals to incorporate the variability of job release in the shipbuilding system. Model 2 and 3 use closed queueing networks to model the CONWIP system.

Markov Decision Processes:

Markov Decision Processes (MDPs) are used in each model to (1) analyze the system; and (2) obtain the insights of the optimal control policy structure. MDP can provide the optimal control policy dynamically according to the system's state. In particular, MDP is applied to control the flexible workshop to decide which job should be

assigned to the flexible workshop dynamically.

Discrete-Event Simulation:

Discrete-event simulation concerns the modeling of a system where the state variables change instantaneously at separate points in time [23]. Discrete-event simulation is used to simulate Model 1 and 2, to (1) look into the intuition of the system behave; and (2) compare the heuristics with the MDP solution.

1.4 Dissertation Outline

This research intends to discover an innovative way to improve shipbuilding productivity using operations research. New flexible queueing network models and dynamic control heuristics are also developed using Markov Decision Processes and regression models. Chapter II (model 1) introduces the operational flexibility to block assembly process and a MDP model is developed for searching the optimal control policy of the flexible workshop. Chapter III (model 2) extends the flexible block assembly model from Chapter II, using CONWIP release policy at strategic level. The problem of how to invest on the capacity of flexibility resource is also investigated in this model. Chapter IV (model 3) explores an analytical model to improve the planning and control of ship outfitting process. One static model provides the planning information of how much outfitting should be processed at each stage, and one dynamic model seeks an efficient control policy to decide when to perform the outfitting activities. Finally, Chapter V reviews the important contributions of the work from both the theoretical and the application perspectives.

CHAPTER II

Dynamic Control of the Flexible Block Assembly Processes

2.1 Introduction

In recent decades, flexibility has been developed as a general concept to enhance performance in manufacturing and service industries. Flexibility in this research specifically refers to the use of cross trained labor, enhanced information systems, advanced logistics, small batch sizes, and multi-purpose machines/tools, all of which have dramatically improved the robustness of a manufacturing system [19].

When a manufacturing firm is facing issues such as uncertain demand, congestion in workflow, and work force availability, using flexible machines or workshops will help to balance the production workload. Fundamental production process paradigms include craft production, project build and the use of build bays, job shops, batch production, assembly lines, and continuous flow lines. Shipbuilding provides a very unusual paradigm incorporating both build bays for some types of ship blocks (basic ship construction unit) and the use of more automated assembly/flow lines for other types of blocks. We examine an approach to incorporate flexibility into the block assembly process of shipbuilding.

2.1.1 The Concept of Manufacturing Flexibility

The development of flexible manufacturing systems began in the early 1970s, and brought the efficiency of mass production to batch production for multiple products. According to [30],"With the emergence of new microprocessor technologies, the concept of flexibility in manufacturing has become an important consideration in the design, operation, and management of manufacturing systems." It is also noted in [20] that the flexibility of a manufacturing system indicates its control capacity by means of an increase in variety, speed, and effectiveness of responses to uncertain future environmental developments.

Dimensions of Flexibility

Based on the terminology and classification system in [22] and [30], several dimensions of flexibility can be identified at three levels of manufacturing: (1) individual resource level, (2) shop floor level, and (3) plant level.

Individual Resource Level:

- Machine Flexibility: The various types of operations that the machine can perform without requiring a prohibitive effort in switching from one operation to another or large changes in performance outcomes.

- Labor Flexibility: The number and variety of operations or tasks a worker can execute without incurring high transition penalties or significant changes in performance outcomes.

Shop Floor Level:

- Routing Flexibility: A manufacturing systems ability to produce various products by alternate routes through the system without incurring high transition penalties or significant changes in performance outcomes.

Plant Level:

- Product Flexibility: The ability with which new parts or products can be produced without high cost or significant changes in performance outcomes.

- Production Flexibility: The number and variety of part or product types that the manufacturing system can produce without adding costly capital equipment or increasing significant changes in performance outcomes.

- Mix Flexibility: The number and variety of parts or products that can be produced without incurring high transition penalties or significant changes in performance.

Flexibility at the individual resource level serves as a basic building block for flexibilities at the shop floor level and plant level. It was also noted in [22] that flexibility at the individual resource level tends to be more tactical, and flexibilities at the shop floor level and plant level tend to be more strategic.

Operational Flexibility

Operational flexibility is the ability of a production operation in a plant department or work cell to respond quickly and cost-effectively to changing demand mix, demand volume, processing time uncertainty, etc. This type of flexibility focuses on the production planning and control in a manufacturing system. Operational flexibility results from processes that coordinate cross-trained workers, multi-functional machines, information technology, and effective controls of production lines and work cells so that the operation is more efficient and responsive in a dynamic, uncertain environment.

The benefits of operational flexibility include increasing facility utilization and availability, reducing processing time uncertainty, as well as improving responsiveness by reducing delay. To introduce the potential benefits of operational flexibility to

the shipbuilding process, a flexible block assembly process is proposed as a first step toward the overall flexible shipbuilding management system.

2.1.2 Block Assembly Process with Operational Flexibility

Shipbuilding, traditionally a labor-intensive manufacturing industry, includes several types of manufacturing processes, such as the assembly process, out-fitting process, painting process, and erection process. Flexibility can be introduced into the shipbuilding process in many ways. The research presented here focuses on the ship block assembly manufacturing system, because the block assembly processes usually require the most consistent work in shipbuilding, and the ship block assembly process planning activity is essential to other process planning activities. Operational flexibility is introduced to the block assembly process via the flexible curved block workshop, so that flat blocks can be produced either on a flat block panel production line or in a curved block workshop.

Modularization and Block Type

Blocks are the basic building units for a ship. This modular block construction strategy has been preferred in shipbuilding since World War II. Shipbuilders follow this strategy to erect a ship block-by-block until the ship is complete [8]. Each block is composed of a fixed amount of materials and components, such as steel plates and sections of various sizes and shapes. The construction of blocks involves scheduling and routing steel parts, defining operations sequence, assembling parts, outfitting components on block, and painting.

There are multiple types of blocks; however, they all have either flat or curved features. Block types are generalized as flat blocks or curved blocks in the presented model:

-Flat block: The outer surface of a flat block is a flat plate. There are no projections from panel under sides which require special jigs. Blocks are to be assembled on a panel production line.

-Curved block: The outer surface of a curved block is curved. Curved blocks are formed with curved plates which require the pin jigs to support the block from underside. Blocks of this type can only be built in a bay-build facility.

Flat blocks are characterized by less complex assembly work content and assembled with actual flow, while the curved features of curved blocks make them more difficult to build and are assembled with virtual flow. The work content and time required for curved block assembly is considerably greater than the work content and amount of time required for flat block assembly.

Block Assembly Processes

Many U.S. shipyards have adopted a process lane concept, in which flat blocks are assembled on the flat block panel production line with actual flow. Based on the extensive description in [31], we describe the basic block assembly process as follows. Firstly, the basic flow pattern for flat blocks begins from the steel plate and structural sections. The steel goes to the plate shop for initial surface preparation and coating, followed by cutting the parts. Then the parts may go directly through the panel line or be palletized for subassembly. Actual flow of the flat blocks on the panel production line includes stages for assembling the block and outfitting. Finally, after painting and completing additional outfitting on-block work, the blocks are sent directly to the building position. Figure 2.1 shows the panel production line for flat block production.

Curved blocks are usually assembled in bay-build facilities held by pin jigs. Parts are manufactured in the plate shop, but some of the parts may be sub-assembled

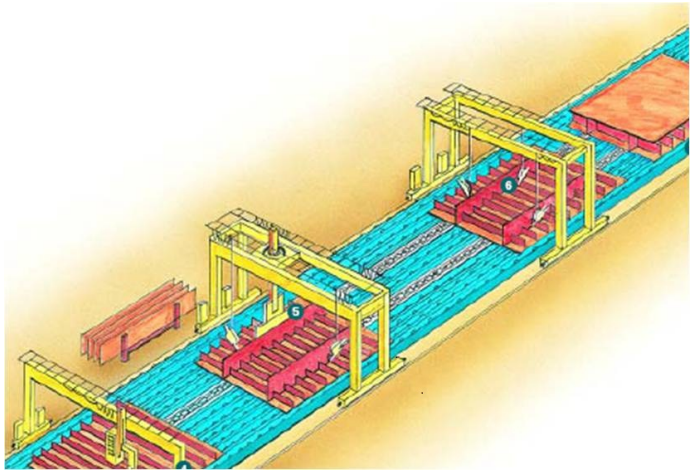


Figure 2.1: Flat Panel Assembly line

before being sent to the curved block build-bay. Figure 2.2 shows the working conditions in a curved block workshop.

To maintain efficiency and uniform work flows in the process lanes, planning and process control are essential. The time to complete each block directly affects the time to complete the whole ship. However, issues such as delays in other schedules, changes in the requirement, and defects in work will impact the assembly process and increase the block processing time. The variation of each blocks processing time will accumulate, resulting in high variation in the time need to complete the final ship. To reduce both the mean and variation of block processing time when the block assembly system faces these disruptions, operational flexibility is introduced to the block assembly process.

If the production planning and control in the ship block manufacturing systems can be improved by coordinating the flexible curved block facilities, additional benefits may be realized; for example, the curved block facilities can be used more effectively, the bottleneck effect can be reduced, and advanced planning and scheduling for other process can be improved.

Figure 2.2: Curved Panel Assembly Line

Proposed Flexible Block Assembly Paradigm

The flexible block assembly processes allow a curved block workshop to dynamically allocate its build bays to the appropriate mix of flat blocks and curved blocks. The flexible structure is shown in Figure 2.3.

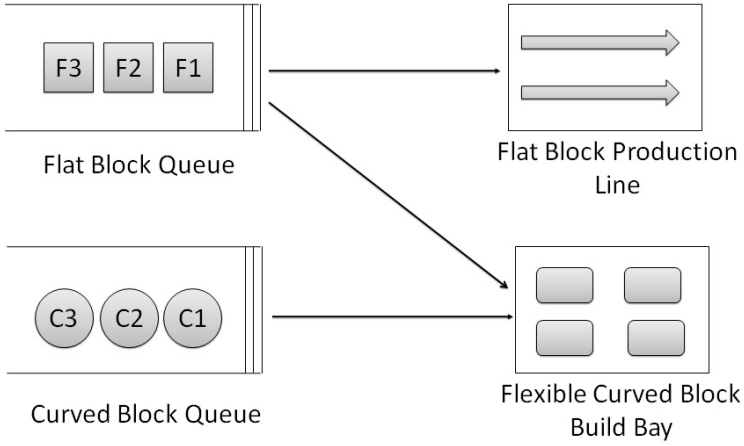


Figure 2.3: Flexible Block Assembly Process Paradigm

Flexibility is only introduced to the curved block build bays since flexibility cannot be cost effectively introduced to the flat block panel production line. One reason for this is that the specialization of the production system on the flat block panel production line requires the flat block bottom for the process of the panel production

line. Additionally, the semi-automatic welding systems on the panel production line are more efficient when joined to the flat parts rather than to the curved parts.

To implement a flexible block assembly process, three primary activities may be required when flexible curved block build bays assemble the flat blocks. First, parts and the sub-assembled structure of the flat blocks must be delivered to flexible curved block build bays. Second, it requires the adjustment of the facilities in curved block build bays, such as by adapting pin jigs to hold the flat block. Finally, cross-training workers who can operate the assemble process for both curved blocks and flat blocks are preferred. The flexible curved block build bays are not as efficient as the flat block production line when assembling flat blocks. Therefore, the problem formulation discussed later in this chapter assumes that the working time is longer for the curved block build bays to produce the flat block than for the panel production line.

Based on this flexible block assembly paradigm, this model examines how incorporating a flexible curved block facility can, with appropriate control/scheduling, increase the efficiency and timeliness of the block assembly process for both flat and curved blocks while enhancing the robustness of the shipbuilding system.

2.2 The "N" Structure Queueing Network

2.2.1 Model Description

The model is formulated as a flexible queueing system that has two streams of job arrivals to infinite capacity buffers and two parallel workshops. Workshop 1 is the panel production line. Workshop 2 is the flexible curved-block build bay. Workshop 1 can process only jobs from the flat block buffer, while workshop 2 can process jobs from either the flat block buffer or curved block buffer. The processing time for both workshops are modeled as exponentially distributed. Each queue has an infinite

capacity buffer. There is holding cost for each block waiting in the system and the objective is to minimize the long-run average holding cost per unit time (with long-run average flowtime being a special case given that the holding cost for each block is one). This type of queueing network with partial flexibility is also called an "N" structure queueing network.

The model is a stylized abstraction of a real system, and it was designed and created under multiple assumptions and limitations in order to make it most accurately represent the plausible responses of an actual system. The assumption of Poisson arrivals for block release is based on the reality that block release times can be highly variable in practice (in part due to design change orders and also timing delays in the supply chain process). Ship blocks are complex three-dimensional structures, so the variation across blocks and defects require real time rework, introducing great processing time variability. An essential assumption in our model is that there is no preemption or collaboration allowed for a single block. In the block assembly process, it will not be cost effective to preempt the unfinished block due to the size and weight of the block. More importantly, it will increase the processing time and affect the quality. Collaboration between servers is sometimes allowed in the literature (see [16]). However, in shipbuilding systems, a particular block can be only assembled either on the flat block production line or in the curved block workshop. Finally, the assumption of infinite buffers for both workshops is due to the fact that we cannot reject any block jobs.

2.2.2 Literature Survey

This "N" queueing network has been studied by [14] and [4]. However, their attention was restricted to a heavy traffic regime. [14] assumed Poisson input streams and deterministic service time. [14] analyzed this problem as a Brownian control

problem and provided a discrete-review policy using the BIGSTEP method. [4] analyzed the system using the solution of Brownian control problem and provided a threshold policy which is asymptotically optimal under the heavy traffic limit. [1] studied "N" structure model with no arrivals, but the problem is much harder with arrivals. [32] studied a "N" system with many servers, and they focused on the heavy traffic regime with the reneging of customers. They proved that a $c\mu$ type greedy policy is asymptotically optimal in many-server heavy traffic model. Our purpose is to achieve better performance in less than heavy traffic, and we are focused on near-optimal performance from policies that can still be implemented in practice.

Recently, a similar model has been studied by [36] and [7]. [36] studied the "N" network with preemption and found an optimality sufficient condition for $c\mu$ control policy to be optimal for the "N" network. [7] studied the "N" network model with upgrades and provided a structural control policy which includes $c\mu$ rule and a threshold policy with optimality conditions. [26] focuses on the "N" queueing network with reneging and limited buffers. The significant difference between their models and ours is that it is not allowed preemption or collaboration in our model due to the constraints of the shipbuilding application. This makes our problem more complicated and harder to control. Further, our focus is on finding tractable near-optimal policies that can be applied.

Much work has been done in the field of controlling flexible servers for different queueing structures. [29] studied a model of two servers and three customer classes, where servers are trained to serve a shared task (the structure is also called "W"). They proved the efficiency of the "W" structure, and sufficient optimality conditions for $c\mu$ rule, which prioritizes serving the fixed task before shared task.

Within the "N" structure queueing model there are existing problems still under-

going research, such as attempting to optimally assign partially flexible servers to jobs in a parallel queueing system to minimize the holding cost. Additionally, after much research, it appears the non-preemptive "N" model has not been studied. By disallowing preemption, the state space must be enlarged. The optimality equations become greatly complicated, and the structure of an optimal policy changes so that it becomes optimal to idle in some states for various problem instances. The structural analysis of the problem becomes extremely difficult. We do not focus on proving properties of optimal policies; rather we focus on creating an effective heuristic control policy, which is benchmarked using the numerical solution of the optimal policy via Markov Decision Process (MDP). Our primary contribution is to design effective new heuristic policies, and to benchmark the potential impact for a flexible block assembly process in shipbuilding. These heuristics have been shown to perform very well; further, they reveal insight into how effective control can be achieved.

2.2.3 Markov Decision Process Formulation

Arrivals of block i follow a Poisson process with rate λ_i. The processing rate of workshop j for block type i is exponentially distributed with rate μ_{ji}. $\mu_{12} = 0$ since "N" structure queueing network only has partial flexibility. Q_i is the queue capacity for block type i. Our model will be general in allowing for finite or infinite queue length at station 1 or 2, respectively. That is, queue i overflows if an arrival sees Q_i jobs already in queue i; however, loss is costless and $Q_i = \infty$ is allowed. h_i is the holding cost for block type i.

Let $X^\pi(t) = (X_1^\pi(t), X_2^\pi(t))$ denote the number of blocks in the system at time t under policy π, where $X_i^\pi(t) \in \{0, 1, 2, ..., Q_i\}$ is the number of block type i at time t. The objective is to find an optimal job assignment policy to control the flexible workshop in order to minimize the long run average holding cost of the

system, assuming no preemption and collaboration are allowed. Both workshop 1 and workshop 2 can stay idle even though the queues are not empty. The expected long-run average holding cost criterion is:

$$J^{\pi} = \liminf_{T\to\infty} \frac{1}{T} E\left[\int_0^T \left(h_1 X_1^{\pi}(t) + h_2 X_2^{\pi}(t)\right) dt \right]. \tag{2.1}$$

Average Cost Per Stage

The state variables of the system are (1) vector $\mathbf{X(t)} = \{X_1(t), X_2(t)\}$, the number of blocks in the system at time t; (2) vector $\mathbf{s(t)} = \{s_1(t), s_2(t)\}$, the status of the workshop at time t. $s_1(t)$ is the status of flat block workshop, where $s_1(t) \in \{0, 1\}$; $s_2(t)$ is the status of flexible workshop, where $s_2(t) \in \{0, 1, 2\}$. The workshop status is included as part of the state variable due to the model assumption that there is no preemption or collaboration allowed. The job cannot be assigned to the workshop if the workshop is not idle. Moreover, there are circumstances that the optimal control polity will idle the workshop even when there are jobs waiting in the queue due to the non-preemption assumption. Therefore, it is necessary to keep track of the status of both workshops in order to find a complete optimal control policy.

Uniformization is used to formulate the equivalent Controlled Discrete Time Markov Chain (CDTMC). Let the uniformization factor $\psi = \lambda_1 + \lambda_2 + \mu_{11} + \max\{\mu_{21}, \mu_{22}\}$, where $\psi > 0$. Let $\lambda_1' = \lambda_1/\psi$, $\lambda_2' = \lambda_2/\psi$, $\mu_{11}' = \mu_{11}/\psi$, $\mu_{21}' = \mu_{21}/\psi$, and $\mu_{22}' = \mu_{22}/\psi$ denote the discrete time parameters after uniformization corresponding to the transition probabilities in the embedded DTMC. This implies that the discrete time costs can be defined in terms of the continuous time costs as $h_1' = h_1/\psi$ and $h_2' = h_2/\psi$. The expected instantaneous cost function is $g(\mathbf{x}, \mathbf{s}) = h_1' x_1 + h_2' x_2$. The average cost per stage starting from state $\{\mathbf{x_0}, \mathbf{s_0}\}$ is

defined by

$$J^{\pi}(\mathbf{x_0},\mathbf{s_0}) = \lim_{N\to\infty}\frac{1}{N}E\left[\sum_{k=0}^{N-1} g(\mathbf{x_k},\mathbf{s_k},\mu_{\mathbf{k}}(\mathbf{x_k},\mathbf{s_k})) \mid \mathbf{x_0},\mathbf{s_0}\right]. \tag{2.2}$$

Optimality Equation

Let $J_k(\mathbf{x},\mathbf{s})$ denote the optimal k-stage cost to go function, and set the terminal cost function, $J_0(\mathbf{x},\mathbf{s}) = 0$. Event operators are defined to simplify the notation. There are five event operators: (1) T^{a_1}, the event of arrival of flat block; (2) T^{a_2}, the event of arrival of curved block; (3) $T^{d_{11}}$, the event of the departure of flat block from flat block workshop; (4) $T^{d_{21}}$, the event of the departure of flat block from flexible workshop; (5) $T^{d_{22}}$, the event of the departure of curved block from flexible workshop. The control action is to minimize the long-run average holding cost. Given any state $\{x_1, x_2, s_1, s_2\}$, the event operator will search for the optimal action to minimize the cost (see the Appendix **2.7.1** at the end of this chapter for all the detailed event operator definitions with control actions).

Based on the analysis of the system dynamics, the most natural version of the recursive value function within finite queues is the MDP:

$$\begin{aligned}
J_{k+1}(\mathbf{x},\mathbf{s}) = {} & g(\mathbf{x},\mathbf{s}) + \lambda_1' \cdot \mathbb{1}_{\{x_1<Q_1\}} T^{a_1} J_k(\mathbf{x},\mathbf{s}) + \lambda_2' \cdot \mathbb{1}_{\{x_2<Q_2\}} T^{a_2} J_k(\mathbf{x},\mathbf{s}) \\
& + \mu_{11}' \cdot \mathbb{1}_{\{x_1\geq 1, s_1=1\}} T^{d_{11}} J_k(\mathbf{x},\mathbf{s}) + \mu_{21}' \cdot \mathbb{1}_{\{x_1\geq 1, s_2=1\}} T^{d_{21}} J_k(\mathbf{x},\mathbf{s}) \\
& + \mu_{22}' \cdot \mathbb{1}_{\{x_2\geq 1, s_2=2\}} T^{d_{22}} J_k(\mathbf{x},\mathbf{s}) \\
& + (1 - \lambda_1' \mathbb{1}_{\{x_1<Q_1\}} - \lambda_2' \mathbb{1}_{\{x_2<Q_2\}} - \mu_{11}' \mathbb{1}_{\{x_1\geq 1, s_1=1\}} \\
& - \mu_{21}' \mathbb{1}_{\{x_1\geq 1, s_2=1\}} - \mu_{22}' \mathbb{1}_{\{x_2\geq 1, s_2=2\}}) J_k(\mathbf{x},\mathbf{s}).
\end{aligned} \tag{2.3}$$

To prevent J_{k+1} from going to infinity, the relative value iteration method is applied to solve the MDP. The relative value iteration (see [5]) uses the corresponding

differential cost vector h. The same constant is subtracted from the value function for all states $J_k(\mathbf{x}, \mathbf{s})$, so that the difference vector h remains bounded. The differential cost vector, $h_k(\mathbf{x}, \mathbf{s})$, is defined as followes:

$$h_k(\mathbf{x}, \mathbf{s}) = J_k(\mathbf{x}, \mathbf{s}) - J_k(\mathbf{x_0}, \mathbf{s_0}) \qquad \forall \mathbf{x}, \mathbf{s}\ , \tag{2.4}$$

$\mathbf{x_0}, \mathbf{s_0}$ are chosen to be the state $\{0, 0, 0, 0\}$ as the fixed state $(\mathbf{x_0}, \mathbf{s_0})$. The relative value function recursion is:

$$\begin{aligned}
h_{k+1}(x_1, x_2, s_1, s_2) &= g(x_1, x_2, s_1, s_2) \\
&+ \lambda_1' \cdot \mathbb{1}_{\left\{x_1 < Q_1\right\}} T^{a_1} h_k(x_1, x_2, s_1, s_2) \\
&+ \lambda_2' \cdot \mathbb{1}_{\left\{x_2 < Q_2\right\}} T^{a_2} h_k(x_1, x_2, s_1, s_2) \\
&+ \mu_{11}' \cdot \mathbb{1}_{\left\{x_1 \geq 1, s_1 = 1\right\}} T^{d_{11}} h_k(x_1, x_2, s_1, s_2) \\
&+ \mu_{21}' \cdot \mathbb{1}_{\left\{x_1 \geq 1, s_2 = 1\right\}} T^{d_{21}} h_k(x_1, x_2, s_1, s_2) \\
&+ \mu_{22}' \cdot \mathbb{1}_{\left\{x_2 \geq 1, s_2 = 2\right\}} T^{d_{22}} h_k(x_1, x_2, s_1, s_2) \\
&+ (1 - \lambda_1' \mathbb{1}_{\left\{x_1 < Q_1\right\}} - \lambda_2' \mathbb{1}_{\left\{x_2 < Q_2\right\}} - \mu_{11}' \mathbb{1}_{\left\{x_1 \geq 1, s_1 = 1\right\}} \\
&- \mu_{21}' \mathbb{1}_{\left\{x_1 \geq 1, s_2 = 1\right\}} - \mu_{22}' \mathbb{1}_{\left\{x_2 \geq 1, s_2 = 2\right\}}) h_k(x_1, x_2, s_1, s_2) \\
&- (\lambda_1' \cdot T^{a_1} h_k(0, 0, 0, 0) + \lambda_2' \cdot T^{a_2} h_k(0, 0, 0, 0)),
\end{aligned} \tag{2.5}$$

The above algorithm is mathematically equivalent to the value iteration equation in equation (2.3), but is computationally stable for solving the infinite horizon problem. It has been shown that the iterates $h_k(\mathbf{x}, \mathbf{s})$ generated by the relative value iteration method are bounded in [5]. The relative value iteration is guaranteed to converge to some vector $h(\mathbf{x}, \mathbf{s})$, with the optimal long run average holding cost

calculated as:

$$\lambda^* = (\lambda_1' \cdot T^{a_1} h_k(0,0,0,0) + \lambda_2' \cdot T^{a_2} h_k(0,0,0,0)) \ . \tag{2.6}$$

2.2.4 Numerical Examples

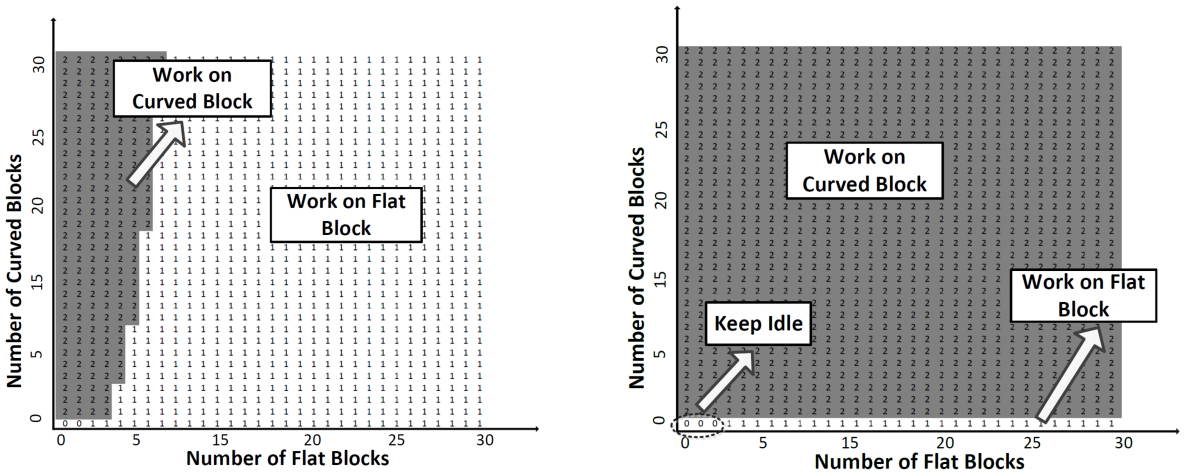
Table 2.1 presents a set of illustrative numerical examples based on the MDP model to gain some insights into the structure of the optimal control policy. We numerically solve the MDP with a convergence criterion of 10^{-5} and truncate the buffer capacity at $Q_1 = Q_2 = 50$. One reason for the finite but large buffer sizes here is to reveal the optimal policy structure without the complexity of "boundary effects" induced when the queues reach Q_1 or Q_2. Therefore, the policy presented is only for states numbered from 0 to 30.

Cases	λ_1	λ_2	μ_{11}	μ_{21}	μ_{22}	h_1	h_2	Scenario
1	1	0.6	1	1	1	2	1	$h_1 > h_2$
2	1	0.6	1	1	1	1	2	$h_1 < h_2$
3	1	0.2	1	0.5	0.25	1	1	$\mu_{11} > \mu_{21}, \mu_{21} > \mu_{22}$
4	1	0.6	1	0.5	1	1	1	$\mu_{11} > \mu_{21}, \mu_{21} < \mu_{22}$
5	1	1.5	1	2	1	1	1	$\mu_{11} < \mu_{21}, \mu_{21} > \mu_{22}$
6	1	1.5	1	2	4	1	1	$\mu_{11} < \mu_{21}, \mu_{21} < \mu_{22}$

Table 2.1: MDP Numerical Test Cases

In cases 1 and 2, the impact of changes in the holding cost on the structure of optimal control policy are analyzed. Figure 2.4 illustrates that (1) when $h_1\mu_{21} > h_2\mu_{22}$, the structure of the optimal control policy is a state-dependent threshold policy; (2) when $h_1\mu_{21} < h_2\mu_{22}$, the optimal control policy is a strict priority policy, which serves to exhaust the curved blocks first before "helping" the dedicated flat block workshop.

In cases 3, 4, 5, and 6 of Table 2.1, workshop processing time was altered to measure its impact on the optimal control policy structure. Figure 2.5 presents the



(a) Case 1: $h_1\mu_{21} > h_2\mu_{22}$

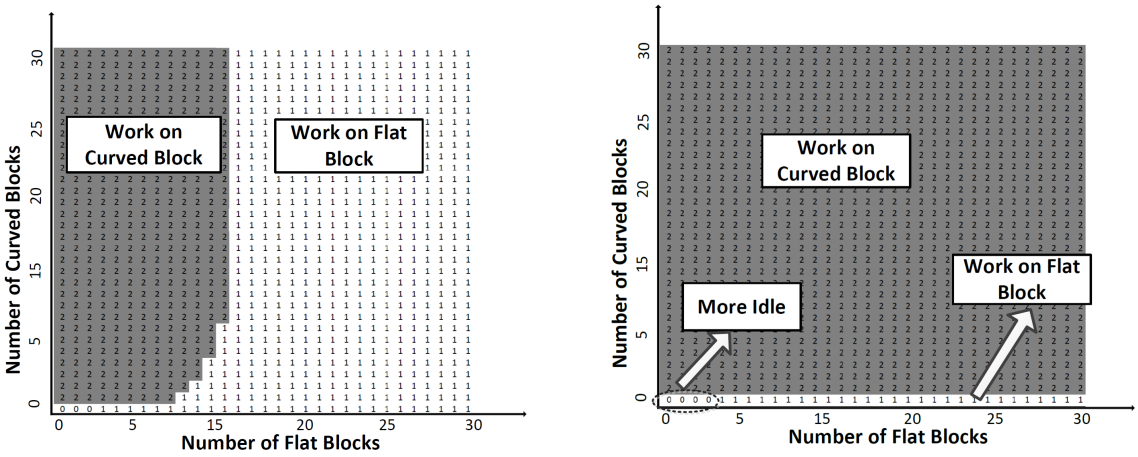
(b) Case 2: $h_1\mu_{21} < h_2\mu_{22}$

Figure 2.4: Optimal Control Policy Structure: Impact of Holding Cost

MDP numerical results of cases 3 and 4. When $h_1\mu_{21} > h_2\mu_{22}$, the structure of the optimal control policy is the same as case 1, a switching curve depending on system states. However, when the flexible workshop is not as efficient as the dedicated flat block workshop, the threshold level of number of flat blocks increased; when $h_1\mu_{21} < h_2\mu_{22}$, the optimal policy is a strict priority giving priority to curved blocks. The result graphs of cases 5 and 6 are not presented since they are consistent with the results of cases 3 and 4. The only difference between the graphs of cases 5 and 6 and those of cases 3 and 4 is that there are more idling states for the flexible workshop in cases 3 and 4, since the flexible workshop is not as efficient as the dedicated flat block workshop, and the optimal control policy tries to prevent the flexible workshop from "stealing" jobs from the dedicated flat block workshop.

2.3 Computational Experiments

In this section, we address the question of how to dynamically assign blocks to the flexible workshop over time. We develop a state-dependent threshold policy



(a) Case 3: $h_1\mu_{21} > h_2\mu_{22}$

(b)Case 4: $h_1\mu_{21} < h_2\mu_{22}$

Figure 2.5: Optimal Control Policy Structure: Impact of Processing Time

and use simulation based optimization to determine the threshold level. A large benchmarking test-suite is designed and it covers different circumstances to compare the performance of difference heuristics. The optimal solution from MDP is used as a benchmark to compare the performance of the heuristics.

2.3.1 Selection of Heuristic Control Policies: The Base Set

Six heuristic policies are tested in this simulation to control the flexible workshop. The static policies (Fixed-Before-Shared policy, $C\mu$ rule) do not use any information regarding the state of the system; some dynamic policies (First Come, First Served policy, Generalized-$C\mu$ rule) use the historical data; and queue-length-based dynamic policies (Longest Queue, Optimal Threshold) use real-time information on the number of jobs in the queue.

First Come, First Served (FCFS): *FCFS* services the requests in the order that they arrive, and may also be referred to *First in First Out (FIFO)*.

Fixed-before-shared (FBS): *FBS* is a strict priority rule in which the flexible workshop will keep working on the curved block until there are no curved blocks waiting

in queue. This is also called the *serve to exhaustion* or *clearing rule*.

$C\mu$ rule: The *$C\mu$ rule* is a strict priority rule in which the queue with larger $C\mu$ index has higher priority. c is the holding cost, which is denoted by h in this paper. μ is the processing rate. When $h_1\mu_{21} < h_2\mu_{22}$, the *$C\mu$ rule* will give curved block priority, which works the same as the *FBS* policy; otherwise, the *$C\mu$ rule* will give flat block priority.

Generalized $C\mu$ rule ($Gc\mu$): A job with a larger $wc\mu$ index has higher priority. In our model, w is the waiting time of job in the system.

Longest Queue (LQ): The LQ policy prioritizes the queue with the larger number of jobs waiting.

FBS policy has proven to be optimal under the condition $h_1\mu_{21} < h_2\mu_{22}$ for the "N" queueing network with pre-emption by [36]. This rule was also studied by [10] in a serial CONWIP system. These studies identified that an effective work-sharing policy should have the flexible worker do the task that only he/she can perform before helping other workers. [29] studied the optimality condition of *FBS* policy controlling the "W" queueing network, which is similar to the "N".

$C\mu$ rule has proven to be optimal for a model of no dedicated server and several parallel stations by [6], and a single-server multi-class queue model studied by [34]. [15] has shown the optimality conditions for $C\mu$ rule controlling the G/G/1 queue with K different customer classes. However, the $C\mu$ rule is static priority rule and not stable for some queueing networks.

The *$GC\mu$ rule* is not a Markov policy since it depends on historical information of the jobs in the queue. In a general single-server multi-class queueing system with convex delay cost, [33] proved that the *$GC\mu$ rule* is asymptotically optimal in heavy traffic.

The LQ policy is widely studied and implemented because it is simple to apply to any flexible structure and works very well for many systems. LQ was applied in [19] for controlling both closed and open flexible queueing structures.

To design a policy that is tailored to the "N" structure and works effectively, we developed a state-dependent threshold policy: the *Optimal Threshold policy*. Let θ denote the threshold level and x_1 be the number of flat blocks in the system. When $x_1 \leq \theta$, use *FBS* policy to control the flexible workshop; otherwise, use the $C\mu$ *rule* (see Figure 2.6). The logic behind this rule is that (1) when the number of flat blocks is smaller than the threshold, using *FBS* policy can prevent the flexible workshop from "stealing" flat block from the dedicated workshop and starving the dedicate workshop; (2) when the number of flat blocks exceeds threshold, using $C\mu$ *rule* can improve the cost by by selecting the job that maximize the greedy reward rate at which it can remove holding cost from the system, which is the intuition contained in [6]. It will not starve the dedicated flat block workshop if the threshold is set at an appropriately high level.

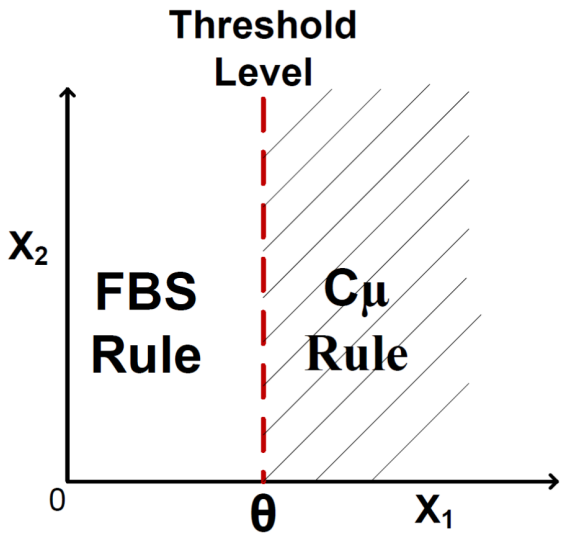


Figure 2.6: Threshold Policy

A challenge for this type of threshold control policy is to find the optimal threshold level to minimize the long-run average holding cost. In simulation, we searched the threshold levels to obtain the one optimizing the system. To avoid the need for a computationally expensive simulation based search, a method using birth-death processes is developed to approximate the threshold level in next section. Other heuristics, such as allowing the threshold to be linear with a slope of less than 90 degrees, have been investigated, but these alternatives did not works better.

For the control of workshop 1 (the non-flexible workshop), we used non-idling policy: (1) if workshop 1 will work on block type 1 as long as the queue of block type 1 is not empty; (2) when both workshop 1 and workshop 2 are idle, first available block type 1 will be assigned to workshop 1, since workshop 1 is more efficient in our model.

2.3.2 Stability Analysis

In simulation, the capacity for block job queue are infinity. Therefore, it is important to identify the system's stability region when $Q_i = \infty$, the set of model parameters for which an optimal policy can yield finite long run average queue lengths. It can provide insights and a practical design guideline for the heuristic test-suite. By solving a linear program similar to that of [2] and [29] (see Appendix **2.7.3** in the end of this chapter), we identify stability region for arrival rates λ_1 and λ_2:

$$\lambda_1 < \mu_{11} + \mu_{21}(1 - \lambda_2/\mu_{22}), \tag{2.7}$$

$$\lambda_2 < \mu_{22}. \tag{2.8}$$

In contrast, the stability region of the inflexible system is $\lambda_1 < \mu_{11}$ and $\lambda_2 < \mu_{22}$, so the new stability region is extended considerably by using the flexible workshop to allow λ_1 to be $(1 + \mu_{21}/\mu_{11}(1 - \lambda_2/\mu_{22}))$ times lager.

2.3.3 Method of Investigation for the Analysis of the System Congestion and Mix of Job Types

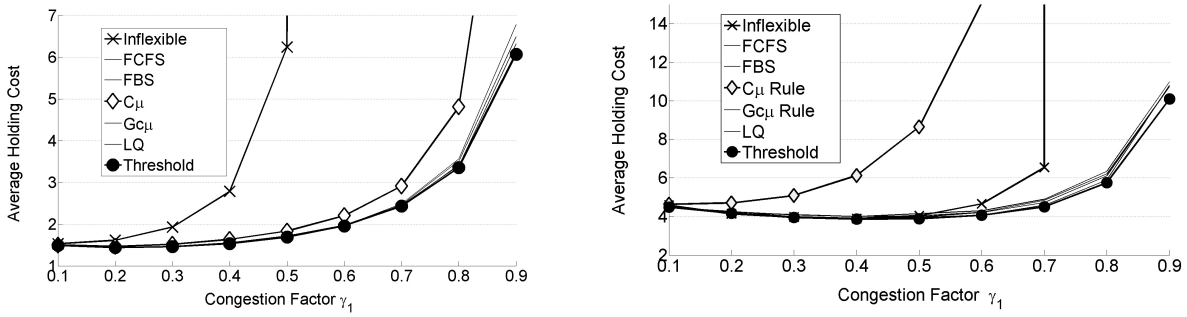
The systems were simulated using a custom discrete event simulation. Each simulation run began with an empty system, and ended after 100,000 blocks exited the line, including a warmup period of 1,000 blocks. Each run was replicated 50 times. For variance reduction purposes, the common random numbers technique was used for each policy, and each workshop had its own random number stream for processing times. At a confidence level of 95%, all standard errors were within 0.5%.

First, the sensitivity of the policies were analyzed with respect to the mix of job types and congestion factors. Let γ_1 and γ_2 denote the system congestion factors of the flat block queue and curved block queue, where $\gamma_1 = \frac{\lambda_1}{\mu_{11}+(1-\gamma_2)\mu_{21}}$, $\gamma_2 = \frac{\lambda_2}{\mu_{22}}$, and $\gamma_1, \gamma_2 \in \{0, 1\}$ to keep systems stable. Table 2.2 shows test-suites to test four different scenarios. When γ_2 is low, the flexible workshop has more free capacity to help the dedicated flat block workshop to produce flat block; when γ_2 is high, the flexible workshop will not have extra capacity to assist. We increased γ_1 from 0.1 to 0.9 in each test-suite to test how control policies perform under different congestions of flat block.

Test-suite	μ_{11}	μ_{21}	μ_{22}	h_1	h_2	γ_2	γ_1	Scenario
1	1	1.25	1	1	1	0.4	$[0.1, 0.9]$	γ_2 is low and $h_1\mu_{21} > h_2\mu_{22}$
2	1	1.25	1	1	1	0.8	$[0.1, 0.9]$	γ_2 is high and $h_1\mu_{21} > h_2\mu_{22}$
3	1	0.8	1	1	1	0.4	$[0.1, 0.9]$	γ_2 is low and $h_1\mu_{21} < h_2\mu_{22}$
4	1	0.8	1	1	1	0.8	$[0.1, 0.9]$	γ_2 is high and $h_1\mu_{21} < h_2\mu_{22}$

Table 2.2: Test-suite for the Congestion Factor

Figure 2.7 illustrates the simulation results of test-suite 1 and test-suite 2. In these two test-suites, $h_1\mu_{21} > h_2\mu_{22}$. Under these circumstances, the $C\mu$ *rule* gives priority to flat blocks. From Figure 2.7 (a), we can observe that (1) the inflexible system



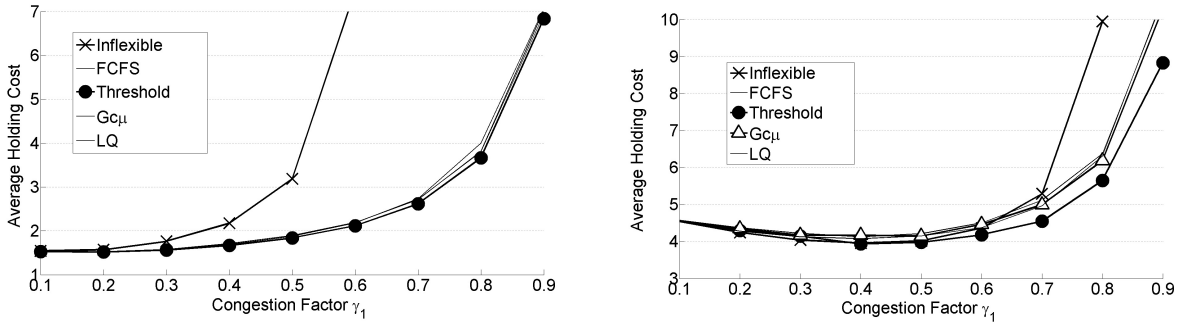
(a) Test-suite 1

(b) Test-suite 2

Figure 2.7: Analysis of System Congestion Factor when $h_1\mu_{21} > h_2\mu_{22}$

is unstable when $\gamma_1 > 0.5$, since the system's dynamics are beyond the stability region of inflexible system; (2) $C\mu$ rule has the worst performance of controlling the flexible system with increasing γ_1; and (3) *FCFS*, *FBS Generalized* $C\mu$ *rule*, *Longest Queue (LQ)*, and *Optimal Threshold* policy perform similarly when the congestion of the curved block is low. This demonstrates that, when the flexible workshop has a large amount of excess/discretionary capacity, various sensible flexible control policies will have about the same impact on the system performance. Figure 2.7 (b) shows that the system performance under control of $C\mu$ *rule* is even worse than the inflexible system. These findings indicate that a flexible system with a bad control policy makes the system worse. *Optimal Threshold* policy has the best performance in both two test-suites, as shown in Figure 2.7 (a) and (b) that *Optimal Threshold* policy has minimal cost .

In test suite 3 (Figure 2.8 (a)) and test suite 4 (Figure 2.8(b)), $h_1\mu 21 < h_2\mu 22$, and the performance of $C\mu$ *rule* and *FBS* will be the same as *Threshold policy.* Therefore, it was only necessary to test *FCFS*, *Optimal Threshold*, *Generalized* $C\mu$ *rule*, and *Longest Queue (LQ)*. Figure 2.8(a) shows that the inflexible system is unstable when $\gamma_1 > 0.6$. When $\gamma_1 = 0.7$, $\lambda_1 = 1.036$, where $\lambda_1 > \mu_{11}$, which is beyond the inflexible



(a) Test-suite 3

(b) Test-suite 4

Figure 2.8: Analysis of System Congestion Factor when $h_1\mu_{21} < h_2\mu_{22}$

system stability region. From Figure 2.8 (b), one can observe that (1) when the congestion of the flat block is low, $\gamma_1 < 0.7$, the performance of inflexible system and flexible system are similar, showing that flexibility cannot greatly benefit the system when the flexible workshop is busy and there are not many flat block jobs; and (2) *Optimal Threshold* policy distinctly has the best performance in all cases demonstrating the robustness of the *Optimal Threshold* policy.

From the results, it can be concluded that the performance under the *Optimal Threshold* policy consistently achieves the minimum cost under all types of environments in the test-suite. When $h_1\mu_{21} < h_2\mu_{22}$ and $x_1 < \theta$, *Optimal Threshold* policy gives priority to the curved block type and thus avoids "stealing" flat block jobs from the dedicated flat block workshop; when $h_1\mu_{21} < h_2\mu_{22}$ and $x_1 > \theta$, *Optimal Threshold* policy gives priority to the flat block, which can help the dedicated flat block workshop assemble the flat block. These test results also indicate under what circumstances flexibility can benefit the system, and how control policies impact the performance of flexibility.

2.3.4 Benchmarking the Base Set of Heuristic Policies with Respect to Optimality

The MDP is solved numerically in order to benchmark the performance with the state space is truncated at a level at which the boundary states are only reached with sufficiently small probability ($Q_1 = Q_2 = 100$).

Let Z^π denote the average holding cost under policy π, and λ^* denote the optimal cost from the MDP. The percentage optimality gap for policy π, G^π is defined as follows:

$$G^\pi = \frac{Z^\pi - \lambda^*}{\lambda^*} \cdot 100\% \ . \tag{2.9}$$

We developed an extensive test suite (see Table 2.3) to compare the performance of our proposed heuristic with the optimal cost from MDP. This test suite covers different combinations of holding costs, processing rates, and arrival rates (congestion factors). There are three test cases for holding cost, i.e. cases of $h_1 = h_2$, $h_1 > h_2$, and $h_1 < h_2$. Four different combinations of processing rate were designed: (1) $\mu_{11} > \mu_{21}$ and $\mu_{11} > \mu_{22}$, (2) $\mu_{11} < \mu_{21}$ and $\mu_{11} > \mu_{22}$, (3) $\mu_{11} > \mu_{21}$ and $\mu_{11} < \mu_{22}$, (4) $\mu_{11} < \mu_{21}$ and $\mu_{11} < \mu_{22}$. The congestion factor γ_1 has four cases to capture the utilization levels of greatest interest: 0.7, 0.8, 0.9, 0.95. γ_2 has four cases: 0.3, 0.5, 0.7, and 0.9. This test suite generates $3 \times 16 \times (12 + 6 + 8 + 6) = 1536$ cases.

Table 2.4 illustrates the performance of policies *FCFS*, *Optimal Threshold*, *Generalized $C\mu$ rule*, and *Longest Queue (LQ)*. When $h_1\mu_{21} \leq h_2\mu_{22}$, *FBS*, *$C\mu$ rule*, and *Optimal Threshold* have the same control action. Therefore, it is only necessary to test the *Optimal Threshold* in these cases. When the system is under low traffic, the mean optimality gap for *Optimal Threshold* is only 0.28% in low traffic and 2.61% in high traffic, by far the lowest. The standard deviation in Table 2.4 shows that (1)

Holding Cost					
		$h_1 = h_2$	$h_1 > h_2$	$h_1 < h_2$	
3 Cases	h_1	1	2	1	
	h_2	1	1	2	
Processing Rate					
$\mu_{11} > \mu_{21}$, $\mu_{11} > \mu_{22}$	μ_{11}	1			
	μ_{21}	0.8	0.9	1	
12 cases	μ_{22}	0.4	0.6	0.8	1
$\mu_{11} > \mu_{21}$, $\mu_{11} < \mu_{22}$	μ_{11}	1			
	μ_{21}	0.8	0.9	1	
6 cases	μ_{22}	1.2	1.4		
$\mu_{11} < \mu_{21}$, $\mu_{11} > \mu_{22}$	μ_{11}	1			
	μ_{21}	1.1	1.2		
8 cases	μ_{22}	0.4	0.6	0.8	1
$\mu_{11} < \mu_{21}$, $\mu_{11} < \mu_{22}$	μ_{11}	1			
	μ_{21}	1.1	1.2	1.4	
6 cases	μ_{22}	1.2	1.4		
Congestion Factor					
16 cases	γ_1	0.7	0.8	0.9	0.95
	γ_2	0.3	0.5	0.7	0.8

Table 2.3: Large Test Suite Design

Optimal Threshold has the smallest standard deviation and the performance is very stable; (2) *LQ* is more robust than *FCFS*. Although *LQ* and *FCFS* have a similar mean of optimality gap, *LQ* has smaller standard deviation.

Table 2.5 illustrates the performance of the base set of heuristic policies. The mean optimality gap for the *Optimal Threshold* policy is 0.43% in low traffic and 3.02% in high. The results show that *Optimal Threshold* policy has the best performance in all circumstances and it is more robust and stable than the other policies. *LQ* is a very robust and stable policy as well, but *Cμ rule* becomes very unstable, especially under high traffic. *Generalized Cμ rule* is more stable than *Cμ rule* since the *Generalized Cμ rule* is not a strict priority policy and it is able to adapt based on the job waiting time in the system. The standard deviation in Table 2.5 shows the robustness of *Optimal Threshold*.

Traffic	Policy	No. of Cases	Mean	Std. Dev.	Min	Max
Low: $\gamma_1 < 0.9$	Opt.Threshold	360	0.28%	0.88%	0.00%	3.35%
	FCFS	360	7.43%	3.93%	0.79%	20.35%
	Generalized-$c\mu$	360	5.51%	2.93%	0.58%	13.63%
	LQ	360	8.01%	2.74%	1.79%	15.43%
High: $\gamma_1 \geq 0.9$	Opt.Threshold	360	2.61%	2.69%	0.00%	10.13%
	FCFS	360	10.78%	5.92%	1.43%	27.11%
	Generalized-$c\mu$	360	9.46%	4.33%	1.24%	20.84%
	LQ	360	14.89%	5.63%	1.93%	25.01%

Table 2.4: Optimality Gap(%) with $h_1\mu_{21} \leq h_2\mu_{22}$ (*FBS*, *Cμ rule*, and *Optimal Threshold* policies are identical)

Traffic	Policy	No. of Cases	Mean	Std. Dev.	Min	Max
Low: $\gamma_1 < 0.9$	Opt.Threshold	408	0.43%	0.44%	0.00%	2.83%
	FCFS	408	5.37%	3.13%	0.77%	18.93%
	FBS	408	5.79%	5.61%	0.00%	28.01%
	$c\mu$	408	49.76%	36.48%	1.93%	103.87%
	Generalized-$c\mu$	408	4.96%	2.83%	0.92%	15.61%
	LQ	408	2.74%	2.56%	0.00%	9.03%
High: $\gamma_1 \geq 0.9$	Opt.Threshold	408	3.02%	2.21%	0.00%	11.31%
	FCFS	408	15.66%	9.78%	3.74%	41.35%
	FBS	408	18.79%	12.83%	0.61%	48.72%
	$c\mu$	408	85.33%	26.01%	15.74%	105.04%
	Generalized-$c\mu$	408	12.97%	6.52%	3.31%	35.49%
	LQ	408	6.63%	3.31%	0.40%	17.81%

Table 2.5: Optimality Gap(%) with $h_1\mu_{21} > h_2\mu_{22}$

2.4 System Analysis for a New Analytical Threshold Policy

The computational results show that the simple *Optimal Threshold* policy has the best performance among all heuristics and is nearly optimal. However, searching for the optimal threshold level is computationally demanding and requires simulation-based search, which is not easily applicable in the real world. In this section, we use a birth and death process as a simpler approximation of the flexible network to develop an *Analytical Threshold* policy.

2.4.1 Two Birth-Death Processes Useful for Estimating the Threshold

When the system is controlled under the *Threshold* policy with threshold level θ, the flat block queue can be estimated as a continuous time Markov chain birth-death process as shown in Figure 2.9. The state X_1 is the number of flat blocks in system. When X_1 is smaller than the threshold level θ, only flat block workshop will work on the flat block and the processing rate is μ_{11}. When x_1 is larger than the threshold level θ, both the flat block workshop and the flexible workshop can be applied to work on the flat block and the processing rate is $\mu_{11} + \mu_{21}$.

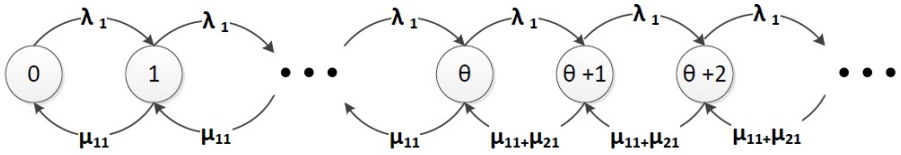


Figure 2.9: The birth-death process approximation of the flat block queue subsystem

A different birth-death process is used to approximate the curved block queue. Let $P^2(\theta)$ denote the long run average fraction of time that the flexible workshop works on curved blocks, which depends on θ and X_1. The processing rate for the curved block queue is $P^2(\theta)\mu_{22}$. Therefore, the approximation of the curved block queue is as shown in Figure 2.10.

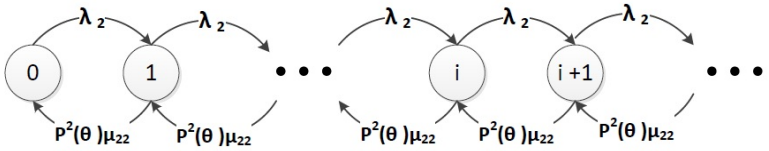


Figure 2.10: The birth-death process approximation of the curved block queue subsystem

The birth-death process is applied to estimate the average queue length for both the flat block queue and curved block queue, and can calculate the long run average holding cost. Let $X_j(t)$ denote the number of type j blocks in the queue at time t and

define the steady-state probabilities P_i^j, $i \geq 0$, as the probability that, on average, the length of queue j equals i.

Let $L^1(\theta)$ and $L^2(\theta)$ denote the average queue length of flat block and curved block, respectively, and $Z(\theta)$ denote the long run average holding cost, which is defined as follows:

$$Z(\theta) = h_1 L^1(\theta) + h_2 L^2(\theta) \ . \tag{2.10}$$

Here, the goal is to find the optimal θ^* to minimize the long run average holding cost $Z(\theta)$.

2.4.2 Cost Function

The average queue length of the flat block queue $L^1(\theta)$ can be estimated by using the steady-state probabilities. From the probability flow balance principle we can obtain them as follows:

$$P_i^1 = (\frac{\lambda_1}{\mu_{11}})^i \cdot P_0^1, \forall i \leq \theta \ , \tag{2.11}$$

$$P_i^1 = (\frac{\lambda_1}{\mu_{11}})^\theta (\frac{\lambda_1}{\mu_{11}+\mu_{21}})^{i-\theta} \cdot P_0^1, \forall i > \theta \ . \tag{2.12}$$

To determine P_0^1, we use the fact that the P_i^1 must sum to 1:

$$\sum_{i=0}^{\infty} P_i^1 = P_0^1 \{ \sum_{i=0}^{\theta} (\frac{\lambda_1}{\mu_{11}})^i + \sum_{i=\theta+1}^{\infty} (\frac{\lambda_1}{\mu_{11}})^\theta \cdot (\frac{\lambda_1}{\mu_{11}+\mu_{21}})^{i-\theta} \} = 1, \text{so}$$

$$P_0^1 = \frac{1}{\sum_{i=0}^{\theta} (\frac{\lambda_1}{\mu_{11}})^i + \sum_{i=\theta+1}^{\infty} (\frac{\lambda_1}{\mu_{11}})^\theta \cdot (\frac{\lambda_1}{\mu_{11}+\mu_{21}})^{i-\theta}} \ , \tag{2.13}$$

Let $\rho_1 = \frac{\lambda_1}{\mu_{11}}$ and $\rho_2 = \frac{\lambda_1}{\mu_{11}+\mu_{21}}$. To keep the system stable, $\lambda_1 < \mu_{11} + \mu_{21}$. Therefore $0 < \rho_2 < 1$. When $\mu_{11} > \lambda_1$, $0 < \rho_2 < \rho_1 < 1$; otherwise when $\mu_{11} < \lambda_1$, $0 < \rho_2 < 1 < \rho_1$. The steady-state probability of state 0 is as follows:

$$
\begin{aligned}
P_0^1(\theta) &= [\frac{1-{\rho_1}^{\theta+1}}{1-\rho_1} + \frac{{\rho_1}^{\theta}\rho_2}{1-\rho_2}]^{-1} \\
&= \frac{(1-\rho_1)(1-\rho_2)}{\rho_1^{\theta}(\rho_2-\rho_1)+(1-\rho_2)} \ . \qquad (2.14)
\end{aligned}
$$

It is simple to prove that $P_0^1(\theta)$ is a concave function of θ and always larger than 0 under all circumstances of ρ_1, ρ_2 and θ, given that those parameters satisfy the stability conditions (2.7). We use the steady-state probabilities to calculate the average queue length of flat block queue $L^1(\theta)$:

$$
\begin{aligned}
L^1(\theta) &= \sum_{i=0}^{\theta} i \cdot {\rho_1}^i \cdot P_0^1(\theta) + \sum_{i=\theta+1}^{\infty} i \cdot {\rho_1}^{\theta}{\rho_2}^{i-\theta} \cdot P_0^1(\theta) \\
&= P_0^1(\theta)(\frac{\theta{\rho_1}^{\theta+2} - (\theta+1)\cdot{\rho_1}^{\theta+1} + \rho_1}{(1-\rho_1)^2} + \frac{-\theta{\rho_2}^2{\rho_1}^{\theta} + (\theta+1)\rho_2{\rho_1}^{\theta}}{(1-\rho_2)^2}) \ . \qquad (2.15)
\end{aligned}
$$

The last equation used the algebraic identity:

$$
\sum_{i=1}^{N} i \cdot a^i = \frac{Na^{N+1}}{a-1} - \frac{a(a^N-1)}{(a-1)^2} \ . \qquad (2.16)
$$

To calculate the average queue length of curved block queue, $L^2(\theta)$, the probability that the flexible workshop works on curved block must be computed:

$$
\begin{aligned}
P^2(\theta) &= P(x_1 \le \theta) \\
&= \sum_{i=0}^{\theta} P_i^1 \\
&= \sum_{i=0}^{\theta} i \cdot {\rho_1}^i \cdot P_0^1(\theta) \\
&= P_0^1(\theta)\frac{1-{\rho_1}^{\theta+1}}{1-\rho_1} \ . \qquad (2.17)
\end{aligned}
$$

To keep the curved block queue stable, θ must satisfy: $P^2(\theta) \cdot \mu_{22} > \lambda_2$. Otherwise the system will not be stable. Since $P_0^1(\theta) > 0$ and $\frac{1-{\rho_1}^{\theta+1}}{1-\rho_1} > 0$, $P^2(\theta) > 0$. $P^2(\theta)$ is a concave function increasing with θ. Therefore, when θ is too small, $P^2(\theta) \cdot \mu_{22} < \lambda_2$, and the system will be unstable. This evidence explains why the system can be unstable under the $c\mu$ rule.

When $P^2(\theta) \cdot \mu_{22} > \lambda_2$, the curved block queue is similar to the M/M/1 queue. let $\rho_3 = \frac{\lambda_2}{\mu_{22}}$. The average queue length $L^2(\theta)$ as follows:

$$\begin{aligned} L^2(\theta) &= \frac{\lambda_2}{P^2(\theta) \cdot \mu_{22} - \lambda_2} \\ &= \frac{\rho_3}{P^2(\theta) - \rho_3}. \end{aligned} \tag{2.18}$$

The convexity of $L^1(\theta)$ and $L^2(\theta)$ depends on the value of θ, ρ_1, and ρ_2. Generally, there is no closed form for optimal θ^* that minimizes the cost function $Z(\theta)$. However, this function can still be used to numerically search the θ^*, which is a much faster than the simulation search.

2.4.3 The Sub-optimality Gap of the Analytical Threshold

The same test-suite from Table 2.3 with $h_1\mu_{21} > h_2\mu_{22}$ is used in order to accurately compare to the previous results. This test includes 916 cases in total. We numerically search for the value of θ^* that minimizes $Z(\theta)$ for each of these test cases, and for a value of θ^* is the threshold level for the *Analytical Threshold* policy. We test the performance of the *Analytical Threshold* policy via simulation and compare them with MDP. Table 2.6 illustrates the performance optimality gap of *Analytical Threshold* policy and *Optimal Threshold* Policy.

As expected, the performance of the *Analytical Threshold* policy is worse than the *Optimal Threshold* Policy. A detailed analysis revealed that the analytical threshold

level is smaller than the optimal threshold level in simulation. One possible reason for this is that when we use the birth-death processes to decouple the system, it is difficult to incorporate the cases when the curved block queue is empty, the flexible workshop will work on flat blocks without any consideration to the threshold level. Therefore, the average processing rate for the flat block queue approximated by the birth-death process is generally smaller than the average processing rate in simulation. With smaller average processing rate, the analytical threshold level will be lower than the optimal one, to increase the probability that the flat block queue will be improved by the flexible workshop.

2.5 A Regression Based Approximation of the Threshold Level: The Regression Threshold Policy

We are not aware of prior literature that used an empirical statistical method being used to refine a heuristic policy in the context of the control of queues, and we show that it is possible to obtain additional performance in this way. The *Analytical Threshold* policy has already yielded a reasonably good nonlinear functional approximation of the optimal policy. Given that the range of systems captured in our test suite is representative of the application context, the empirically computed *Optimal Thresholds* can be used to tweak the *Analytical Threshold* policy. The numerical results show that the analytical threshold levels approximated by birth-death process are generally smaller than the optimal threshold levels from simulation. Therefore, we use a linear regression model to achieve a more accurate approximation of the optimal threshold level.

Let $\tilde{\theta}$ denote the analytical threshold level approximated by the birth-death process, and θ be the optimal threshold level from simulation. We randomly chose half of the test cases from Table 2.3 with $h_1\mu_{21} > h_2\mu_{22}$, which includes 408 cases. $\rho_1 = \frac{\lambda_1}{\mu_{11}}$,

$\rho_2 = \frac{\lambda_1}{\mu_{11}+\mu_{21}}$, and $\rho_3 = \frac{\lambda_2}{\mu_{22}}$, which are defined in previous section.

A second-order regression model is applied in this regression, which included $\tilde{\theta}$, $\rho_1, \rho_2, \rho_3, h_1/h_2,\ \lambda_1, \lambda_2, \mu_{11}, \mu_{21}, \mu_{22}, h_1, h_2,$ and interaction and quadratic terms of $\lambda_1, \lambda_2, \mu_{11}, \mu_{21}, \mu_{22}, h_1, h_2,$ as predictors, and θ as the response. Forward Stepwise Selection is used ([24]) to select the most significant variables from these initial predictors. $\tilde{\theta}$ is the most significant predictor, which is to be expected based on the good quality of the Analytical Threshold. We also used Matrix Plot to analyze the relationship between the response and predictors. The plot of θ and $\tilde{\theta}$ indicates that there exists a quadratic term of $\tilde{\theta}$ which can improve the fitness of the regression model. Therefore, we added $\tilde{\theta}^2$ to the regression equation. The R-squared value is increased from 88.06% to 91.2% by adding this quadratic term. The forward stepwise method selected 7 significant predictors out of 34 initial predictors, and the regression equation is:

$$\begin{aligned}(2.19)\qquad \theta =& 2.343 + 1.65\tilde{\theta} - 0.0293\tilde{\theta}^2 - 0.54h_1/h_2 + 12.6\lambda_2 h_2 - 4.25\mu_{21} - 4\lambda_2 h_1 \\ &+ 5.39\mu_{22} - 5.5\lambda_1\lambda_2.\end{aligned}$$

We use the regression equation above to calculate the Regression Threshold level for each test case in the other half of the large test-suite from Table 2.3. Simulation is used to calculate the system cost under the control of both *Regression Threshold* policy and *Analytical Threshold* policy. Table 2.6 illustrates the performance optimality gap of *Analytical Threshold* policy, *Regression Threshold* policy, and *Optimal Threshold* Policy.

The results in Table 2.6 show that the cost of the system under *Regression Threshold* policy control is very close to the cost of *Optimal Threshold* policy. There are two reasons for the good performance of the *Regression Threshold* policy: (1) the

Traffic	Policy	Mean	Std. Dev.	Min	Max
Low: $\gamma_1 < 0.9$	Optimal Threshold	0.43%	0.44%	0.00%	2.83%
	Analytical Threshold	3.06%	1.88%	0.44%	9.75%
	Regression Threshold	0.50%	0.49%	0.00%	3.12%
High: $\gamma_1 \geq 0.9$	Optimal Threshold	3.02%	2.21%	0.00%	11.31%
	Analytical Threshold	6.24%	3.40%	0.71%	19.22%
	Regression Threshold	3.55%	3.04%	0.01%	12.57%

Table 2.6: Optimality Gap(%) with $h_1\mu 21 > h_2\mu 22$

regression equation includes the most significant parameters, and is therefore very robust and flexible to any changes of the system dynamics; (2) the regression threshold level is frequently larger than the optimal threshold level. We found that the system is not that sensitive to having a threshold larger than the optimal threshold level, analogous to the (S,s) policy of inventory theory.

2.6 Conclusion and Future Work

In this chapter, operational flexibility is introduced to the block assembly process in shipbuilding, which allowed the curved block workshop to build both flat blocks and curved blocks. This approach assumes that using current shipbuilding technology, flexibility can only be built into the curved block build bays since flexibility cannot be cost effectively introduced in the flat block panel production line. Even with only partial (one-way) flexibility, the innovation allows for dynamic workload balancing for improved performance. The flexible block assembly process is formulated as the "N" structure queueing network.

The problem of dynamically controlling the flexible workshop in the "N" queueing network is investigated to minimize the long run average holding cost. First we developed a Markov Decision Process (MDP) model for the non-preemptive "N" queueing network to (1) gain insight of the structure of the optimal control policy,

(2) provide the optimal cost benchmark for our heuristics. The non-preemptive feature makes the MDP model complicated. The numerical MDP results suggest that (1) when $h_1\mu_{21} > h_2\mu_{22}$, the structure of the optimal control policy is a state-dependent switching curve; (2) when $h_1\mu_{21} < h_2\mu_{22}$, the optimal control policy is a strict priority policy, which is exhausting the curved blocks first before helping the dedicated flat block workshop to assemble flat blocks.

A simple and robust, state dependent *Optimal Threshold* policy is developed which requires simulation-based search. The *Optimal Threshold* policy depends only on the number of flat blocks in the system. With θ denoting threshold level, when $x_1 \leq \theta$, we employ a strict priority for curved block policy to control the flexible workshop; and when $x_1 > \theta$, we are using *$C\mu$ rule* to control the flexible workshop. The *Optimal Threshold* policy is compared with other heuristic control policies via simulation, including *FCFS*, *FBS*, *$C\mu$ rule*, *Generalized $C\mu$ rule*, and *Longest Queue (LQ)* control policies. The extensive test suite shows that (1) this threshold policy performs the best amount all the heuristic policies we tested; (2) the average holding cost is very closed to the optimal cost calculated by the MDP model.

Although the *Optimal Threshold* policy simply depends only on the number of flat blocks in the system, finding the threshold level is essential and challenging. We use two birth-death processes to develop an *Analytical Threshold* policy, which is subsequently employed to develop a regression model to compute a *Regression Threshold* policy. The regression equation is based on a large test-suite and results shows that the approximation of the optimal threshold level is promising as a way to get very close to the optimal policy. This approach uses a great deal of pre-computation, but it is computationally easy to apply in real time.

It remains as future research to identify if the approach of integrating analytical

approximations is valuable not only in open flexible parallel open queueing networks, but also in a wide ranges of systems, including closed flexible parallel queueing networks and tandem queueing networks.

2.7 Appendix

2.7.1 Definition of Event Operators

Event operator T^{a_1}: Arrival of flat block with probability λ_1'

$$
\begin{aligned}
&T^{a_1} J_k(x_1, x_2, s_1, s_2) = \\
&\mathbb{1}_{\left\{x_1=0, x_2=0, s_1=0, s_2=0\right\}} \min\left\{J_k(x_1+1, x_2, s_1=1, s_2=0), J_k(x_1+1, x_2, s_1=0, s_2=1),\right. \\
&\left. J_k(x_1+1, x_2, s_1=0, s_2=0)\right\} \\
&+ \mathbb{1}_{\left\{x_1\geq 1, x_2=0, s_1=0, s_2=0\right\}} \min\left\{J_k(x_1+1, x_2, s_1=1, s_2=1), J_k(x_1+1, x_2, s_1=1, s_2=0),\right. \\
&\left. J_k(x_1+1, x_2, s_1=0, s_2=1), J_k(x_1+1, x_2, s_1=0, s_2=0)\right\} \\
&+ \mathbb{1}_{\left\{x_1=0, x_2\geq 1, s_1=0, s_2=0\right\}} \min\left\{J_k(x_1+1, x_2, s_1=1, s_2=2), J_k(x_1+1, x_2, s_1=1, s_2=0),\right. \\
&\left. J_k(x_1+1, x_2, s_1=0, s_2=1), J_k(x_1+1, x_2, s_1=0, s_2=2), J_k(x_1+1, x_2, s_1=0, s_2=0)\right\} \\
&+ \mathbb{1}_{\left\{x_1\geq 1, x_2\geq 1, s_1=0, s_2=0\right\}} \min\left\{J_k(x_1+1, x_2, s_1=1, s_2=1), J_k(x_1+1, x_2, s_1=1, s_2=2),\right. \\
&J_k(x_1+1, x_2, s_1=1, s_2=0), J_k(x_1+1, x_2, s_1=0, s_2=1), J_k(x_1+1, x_2, s_1=0, s_2=2), \\
&\left. J_k(x_1+1, x_2, s_1=0, s_2=0)\right\} \\
&+ \mathbb{1}_{\left\{s_1=0, s_2\neq 0\right\}} \min\left\{J_k(x_1+1, x_2, s_1=1, s_2), J_k(x_1+1, x_2, s_1=0, s_2)\right\} \\
&+ \mathbb{1}_{\left\{x_2=0, s_1\neq 0, s_2=0\right\}} \min\left\{J_k(x_1+1, x_2, s_1, s_2=1), J_k(x_1+1, x_2, s_1, s_2=0)\right\} \\
&+ \mathbb{1}_{\left\{x_2\geq 1, s_1\neq 0, s_2=0\right\}} \min\left\{J_k(x_1+1, x_2, s_1, s_2=1), J_k(x_1+1, x_2, s_1, s_2=2), J_k(x_1+1, x_2, s_1, s_2=0)\right. \\
&+ \mathbb{1}_{\left\{s_1\neq 0, s_2\neq 0\right\}} J_k(x_1+1, x_2, s_1, s_2)
\end{aligned}
$$

(2.20)

Event operator T^{a_2}: Arrival of curved block with probability λ_2'

$$
\begin{aligned}
&T^{a_2} J_k(x_1, x_2, s_1, s_2) = \\
&\mathbb{1}_{\{x_1=0, s_1=0, s_2=0\}} \min\big\{J_k(x_1, x_2+1, s_1=0, s_2=2), J_k(x_1, x_2+1, s_1=0, s_2=0)\big\} \\
&+ \mathbb{1}_{\{x_1=1, s_1=0, s_2=0\}} \min\big\{J_k(x_1, x_2+1, s_1=1, s_2=2), J_k(x_1, x_2+1, s_1=1, s_2=0), \\
&J_k(x_1, x_2+1, s_1=0, s_2=1), J_k(x_1, x_2+1, s_1=0, s_2=2), J_k(x_1, x_2+1, s_1=0, s_2=0)\big\} \\
&+ \mathbb{1}_{\{x_1\geq 2, s_1=0, s_2=0\}} \min\big\{J_k(x_1, x_2+1, s_1=1, s_2=1), J_k(x_1, x_2+1, s_1=1, s_2=2), \\
&J_k(x_1, x_2+1, s_1=1, s_2=0), J_k(x_1, x_2+1, s_1=0, s_2=1), J_k(x_1, x_2+1, s_1=0, s_2=2), \\
&J_k(x_1, x_2+1, s_1=0, s_2=0)\big\} \\
&+ \mathbb{1}_{\{x_1=1, s_1=1, s_2=0\}} \min\big\{J_k(x_1, x_2+1, s_1=1, s_2=2), J_k(x_1, x_2+1, s_1=1, s_2=0)\big\} \\
&+ \mathbb{1}_{\{x_1\geq 2, s_1=1, s_2=0\}} \min\big\{J_k(x_1, x_2+1, s_1=1, s_2=1), J_k(x_1, x_2+1, s_1=1, s_2=2), \\
&J_k(x_1, x_2+1, s_1=1, s_2=0)\big\} \\
&+ \mathbb{1}_{\{x_1=1, s_1=0, s_2=1\}} \min\big\{J_k(x_1, x_2+1, s_1=0, s_2=1)\big\} \\
&+ \mathbb{1}_{\{x_1\geq 2, s_1=0, s_2=1\}} \min\big\{J_k(x_1, x_2+1, s_1=1, s_2=1), J_k(x_1, x_2+1, s_1=0, s_2=1)\big\} \\
&+ \mathbb{1}_{\{x_1=0, s_1=0, s_2=2\}} \min\big\{J_k(x_1, x_2+1, s_1=0, s_2=2)\big\} \\
&+ \mathbb{1}_{\{x_1\geq 1, s_1=0, s_2=2\}} \min\big\{J_k(x_1, x_2+1, s_1=1, s_2=2), J_k(x_1, x_2+1, s_1=0, s_2=2)\big\} \\
&+ \mathbb{1}_{\{s_1\neq 0, s_2\neq 0\}} J_k(x_1, x_2+1, s_1, s_2)
\end{aligned}
$$

(2.21)

Event operator $T^{d_{11}}$: Departure of flat block from workshop 1 with probability μ'_{11}

$$
\begin{aligned}
&T^{d_{11}} J_k(x_1, x_2, s_1, s_2) = \\
&\mathbb{1}_{\left\{x_1=1, x_2=0, s_1=1, s_2=0\right\}} J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 0) \\
&+ \mathbb{1}_{\left\{x_1=2, x_2=0, s_1=1, s_2=0\right\}} \min\big\{J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 0), J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 1), \\
&J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1\geq 3, x_2=0, s_1=1, s_2=0\right\}} \min\big\{J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 1), J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 0), \\
&J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 1), J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1=1, x_2\geq 1, s_1=1, s_2=0\right\}} \min\big\{J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 2), J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1=2, x_2\geq 1, s_1=1, s_2=0\right\}} \min\big\{J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 2), J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 0), \\
&J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 1), J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 2), J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1\geq 3, x_2\geq 1, s_1=1, s_2=0\right\}} \min\big\{J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 1), J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 2), \\
&J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 0), J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 1), J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 2), \\
&J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1=2, s_1=1, s_2=1\right\}} J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 1) \\
&+ \mathbb{1}_{\left\{x_1\geq 3, s_1=1, s_2=1\right\}} \min\big\{J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 1), J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 1)\big\} \\
&+ \mathbb{1}_{\left\{x_1=1, x_2\geq 1, s_1=1, s_2=2\right\}} J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 2) \\
&+ \mathbb{1}_{\left\{x_1\geq 2, x_2\geq 1, s_1=1, s_2=2\right\}} \min\big\{J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 2), J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 2)\big\}
\end{aligned}
$$

(2.22)

Event operator $T^{d_{21}}$: Departure of flat block from workshop 2 with probability μ'_{21}

$$
\begin{aligned}
&T^{d_{21}} J_k(x_1, x_2, s_1, s_2) = \\
&\mathbb{1}_{\left\{x_1=1, x_2=0, s_1=0, s_2=1\right\}} J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 0) \\
&+ \mathbb{1}_{\left\{x_1=2, x_2=0, s_1=0, s_2=1\right\}} \min\big\{J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 0), J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 1), \\
&J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1\geq 3, x_2=0, s_1=0, s_2=1\right\}} \min\big\{J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 1), J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 0), \\
&J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 1), J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1=1, x_2\geq 1, s_1=0, s_2=1\right\}} \min\big\{J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 2), J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1=2, x_2\geq 1, s_1=0, s_2=1\right\}} \min\big\{J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 2), J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 0), \\
&J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 1), J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 2), J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1\geq 3, x_2\geq 1, s_1=0, s_2=1\right\}} \min\big\{J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 1), J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 2), \\
&J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 0), J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 1), J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 2), \\
&J_k(x_1 - 1, x_2, s_1 = 0, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1=2, x_2=0, s_1=1, s_2=1\right\}} J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 0) \\
&+ \mathbb{1}_{\left\{x_1\geq 3, x_2=0, s_1=1, s_2=1\right\}} \min\big\{J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 1), J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1=2, x_2\geq 1, s_1=1, s_2=1\right\}} \min\big\{J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 2), J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1\geq 3, x_2\geq 1, s_1=1, s_2=1\right\}} \min\big\{J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 1), J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 2), \\
&J_k(x_1 - 1, x_2, s_1 = 1, s_2 = 0)\big\}
\end{aligned}
$$

(2.23)

Event operator $T^{d_{22}}$: Departure of curved block from workshop 2 with probability μ'_{22}

$$
\begin{aligned}
&T^{d_{22}} J_k(x_1, x_2, s_1, s_2) = \\
&\mathbb{1}_{\left\{x_1=0, x_2=1, s_1=0, s_2=2\right\}} J_k(x_1, x_2 - 1, s_1 = 0, s_2 = 0) \\
&+ \mathbb{1}_{\left\{x_1=1, x_2=1, s_1=0, s_2=2\right\}} \min\big\{J_k(x_1, x_2 - 1, s_1 = 1, s_2 = 0), J_k(x_1, x_2 - 1, s_1 = 0, s_2 = 1), \\
&J_k(x_1, x_2 - 1, s_1 = 0, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1\geq 2, x_2=1, s_1=0, s_2=2\right\}} \min\big\{J_k(x_1, x_2 - 1, s_1 = 1, s_2 = 1), J_k(x_1, x_2 - 1, s_1 = 1, s_2 = 0), \\
&J_k(x_1, x_2 - 1, s_1 = 0, s_2 = 1), J_k(x_1, x_2 - 1, s_1 = 0, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1=0, x_2\geq 2, s_1=0, s_2=2\right\}} \min\big\{J_k(x_1, x_2 - 1, s_1 = 0, s_2 = 2), J_k(x_1, x_2 - 1, s_1 = 0, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1=1, x_2\geq 2, s_1=0, s_2=2\right\}} \min\big\{J_k(x_1, x_2 - 1, s_1 = 1, s_2 = 2), J_k(x_1, x_2 - 1, s_1 = 1, s_2 = 0), \\
&J_k(x_1, x_2 - 1, s_1 = 0, s_2 = 1), J_k(x_1, x_2 - 1, s_1 = 0, s_2 = 2), J_k(x_1, x_2 - 1, s_1 = 0, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1\geq 2, x_2\geq 2, s_1=0, s_2=2\right\}} \min\big\{J_k(x_1, x_2 - 1, s_1 = 1, s_2 = 1), J_k(x_1, x_2 - 1, s_1 = 1, s_2 = 2), \\
&J_k(x_1, x_2 - 1, s_1 = 1, s_2 = 0), J_k(x_1, x_2 - 1, s_1 = 0, s_2 = 1), J_k(x_1, x_2 - 1, s_1 = 0, s_2 = 2), \\
&J_k(x_1, x_2 - 1, s_1 = 0, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1=1, x_2=1, s_1=1, s_2=2\right\}} J_k(x_1, x_2 - 1, s_1 = 1, s_2 = 0) \\
&+ \mathbb{1}_{\left\{x_1\geq 2, x_2=1, s_1=1, s_2=2\right\}} \min\big\{J_k(x_1, x_2 - 1, s_1 = 1, s_2 = 1), J_k(x_1, x_2 - 1, s_1 = 1, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1=1, x_2\geq 2, s_1=1, s_2=2\right\}} \min\big\{J_k(x_1, x_2 - 1, s_1 = 1, s_2 = 2), J_k(x_1, x_2 - 1, s_1 = 1, s_2 = 0)\big\} \\
&+ \mathbb{1}_{\left\{x_1\geq 2, x_2\geq 2, s_1=1, s_2=2\right\}} \min\big\{J_k(x_1, x_2 - 1, s_1 = 1, s_2 = 1), J_k(x_1, x_2 - 1, s_1 = 1, s_2 = 2), \\
&J_k(x_1, x_2 - 1, s_1 = 1, s_2 = 0)\big\}
\end{aligned}
$$

2.7.2 Relative Value Iteration Equations

We define $h_k(\mathbf{x}, \mathbf{s})$, the differential cost vector:

$$h_k(\mathbf{x}, \mathbf{s}) = J_k(\mathbf{x}, \mathbf{s}) - J_k(\mathbf{x_0}, \mathbf{s_0}) \qquad \forall \mathbf{x}, \mathbf{s}., \tag{2.24}$$

where $\mathbf{x_0}, \mathbf{s_0}$ is chosen to be the state $\{0, 0, 0, 0\}$ as the fixed state $(\mathbf{x_0}, \mathbf{s_0})$. The relative value function recursion is:

$$\begin{aligned}
h_{k+1}(x_1, x_2, s_1, s_2) &= g(x_1, x_2, s_1, s_2) \\
&+ \lambda_1' \cdot \mathbb{1}_{\left\{x_1 < Q_1\right\}} T^{a_1} h_k(x_1, x_2, s_1, s_2) \\
&+ \lambda_2' \cdot \mathbb{1}_{\left\{x_2 < Q_2\right\}} T^{a_2} h_k(x_1, x_2, s_1, s_2) \\
&+ \mu_{11}' \cdot \mathbb{1}_{\left\{x_1 \geq 1, s_1 = 1\right\}} T^{d_{11}} h_k(x_1, x_2, s_1, s_2) \\
&+ \mu_{21}' \cdot \mathbb{1}_{\left\{x_1 \geq 1, s_2 = 1\right\}} T^{d_{21}} h_k(x_1, x_2, s_1, s_2) \\
&+ \mu_{22}' \cdot \mathbb{1}_{\left\{x_2 \geq 1, s_2 = 2\right\}} T^{d_{22}} h_k(x_1, x_2, s_1, s_2) \\
&+ (1 - \lambda_1' \mathbb{1}_{\left\{x_1 < Q_1\right\}} - \lambda_2' \mathbb{1}_{\left\{x_2 < Q_2\right\}} - \mu_{11}' \mathbb{1}_{\left\{x_1 \geq 1, s_1 = 1\right\}} \\
&- \mu_{21}' \mathbb{1}_{\left\{x_1 \geq 1, s_2 = 1\right\}} - \mu_{22}' \mathbb{1}_{\left\{x_2 \geq 1, s_2 = 2\right\}}) h_k(x_1, x_2, s_1, s_2) \\
&- (\lambda_1' \cdot T^{a_1} h_k(0, 0, 0, 0) + \lambda_2' \cdot T^{a_2} h_k(0, 0, 0, 0)),
\end{aligned} \tag{2.25}$$

The above algorithm is mathematically equivalent to the value iteration equation in equation (2.3), but is computationally stable for solving the infinite horizon problem. It has been shown that the iterates $h_k(\mathbf{x}, \mathbf{s})$ generated by the relative value iteration method are bounded in [5]. The relative value iteration is guaranteed to converge to some vector $h(\mathbf{x}, \mathbf{s})$, with the optimal long run average holding cost calculated as:

$$\lambda^* = (\lambda_1' \cdot T^{a_1} h_k(0, 0, 0, 0) + \lambda_2' \cdot T^{a_2} h_k(0, 0, 0, 0)). \tag{2.26}$$

2.7.3 Proof of the Stability Region

We use a linear programming model to analyze the system stability. A similar allocation LP model has been studied by [2] and [29]. Let y_{ij} denote the long run proportion of time workshop i spends working on job j. I_i is the long run proportion of time that workshop i is idle. We develop the LP model to find the allocation rate of y_{ij} to maximize the minimum idle time among all the workshops. The general LP formulation is as follows:

$$MaxMin\{I_i\} \tag{2.27}$$

$$s.t.$$

$$\lambda_j < \sum_i y_{ij}\mu_{ij} \qquad \forall j \tag{2.28}$$

$$I_i = 1 - \sum_j y_{ij} \qquad \forall i \tag{2.29}$$

$$I_i \geq 0 \qquad \forall i \tag{2.30}$$

$$y_{ij} \geq 0 \qquad \forall i, j. \tag{2.31}$$

Equation (2.29) is the system stability constraint. Equation (2.30) represents the relationship between y_{ij} and I_i. In our model, we have two workshops,$i = 1, 2$, and two job types, $j = 1, 2$. Specifically,$y_{12} = 0$ in our model. As long as there is existing the feasible solution for I_1 and I_2, the system should remain stable.

Proof. First, the stability region for the curved block queue is $\lambda_2 < \mu_{22}$, since the flexible workshop is the only workshop for curved block queue.

Second, to prove the stability region for the flat block queue, we considerate two cases:

(1) When $\mu_{11} > \mu_{21}$, the optimal solution solved by LP will make full use of workshop 1. Under this condition, when $\lambda_1 < \mu_{11}$, the system is stable. The corresponding

optimal solution for LP is $y_{11} = \frac{\lambda_1}{\mu_{11}}$, $y_{21} = 0$, and $y_{22} = \frac{\lambda_2}{\mu_{22}}$. When $\lambda_1 > \mu_{11}$, the optimal solution is $y_{11} = 1$, $y_{21} = \frac{\lambda_1 - \mu_{11}}{\mu_{21}}$, and $y_{22} = \frac{\lambda_2}{\mu_{22}}$. The system is stable if, and only if, $y_{21} + y_{22} \leq 1$. Thus we have $\frac{\lambda_1 - \mu_{11}}{\mu_{21}} + \frac{\lambda_2}{\mu_{22}} \leq 1$. The stable region for flat block queue is $\lambda_1 < \mu_{11} + \mu_{21}(1 - \lambda_2/\mu_{22})$, when $\mu_{11} > \mu_{21}$.

(2) When $\mu_{11} < \mu_{21}$, the optimal solution solved by LP is $y_{11} = \frac{\lambda_1 - y_{21}\mu_{21}}{\mu_{11}}$, $y_{21} = 1 - \frac{\lambda_2}{\mu_{22}}$, and $y_{22} = \frac{\lambda_2}{\mu_{22}}$. The system is stable if, and only if, $y_{11} \leq 1$. Thus we have $\frac{\lambda_1 - y_{21}\mu_{21}}{\mu_{11}} < 1$. The stable region for flat block queue is $\lambda_1 < \mu_{11} + \mu_{21}(1 - \lambda_2/\mu_{22})$, when $\mu_{11} < \mu_{21}$. Therefore, the stability region for the flat block queue is $\lambda_1 < \mu_{11} + \mu_{21}(1 - \lambda_2/\mu_{22})$. □

CHAPTER III

Capacity Planning and Control of Flexible Hull Construction Processes under CONWIP Discipline

3.1 Introduction

Chapter II introduced operational flexibility to the block assembly process in ship hull construction, which is modeled as an "N" structure queueing networks. This model provides an effective control policy for operating the flexible workshop to minimize the average holding cost at execution level. In this chapter, Constant Work In Process (CONWIP) release policy is introduced at the strategic level of the flexible block assembly process. The reason for introducing CONWIP release policy to shipbuilding is that CONWIP policy is an effective and robust release policy with easier implementation than current shipbuilding planning policy. This CONWIP model is also used to measure the improvement of system performance by flexibility. Two research problems are targeted in the model: (1) how to control the flexible workshop to maximize the system throughput under CONWIP release policy; and (2) how to plan the capacity of flexible resource. To introduce a new planning policy to the hull construction, current hull construction planning methodology will be reviewed first in the next section.

3.1.1 Methodology Behind Hull Construction Planning in Shipbuilding

As mentioned in Chapter 2, hull construction has been using the modular block construction strategy since World War II. The processes of hull construction shown in Figure 3.1 includes part fabrication, part assembly, sub-block assembly, semi-block assembly, block assembly, grand-block joining and final hull erection. The main work flow consists of part fabrication, sub-block assembly, block assembly, and final hull erection. However, processes such as part assembly, semi-block assembly, and grand-block joining provide useful planning alternatives.

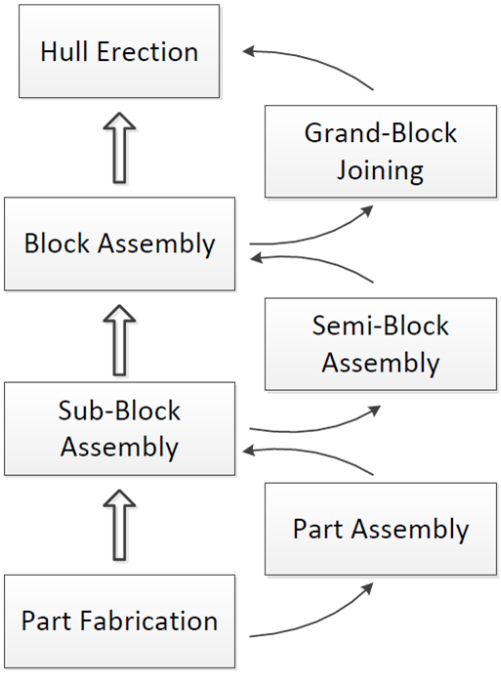


Figure 3.1: Hull Planning Steps

The hull construction planning is based on its processes. The current most commonly used scheduling approaches in shipbuilding are Manufacturing Resource Planning (MRP II or Push System) and network scheduling (critical path method). Given hull construction due date and block sequence, the schedules and due date is broken down level by level, down to part fabrication, which is the beginning of the process.

MRP system and network scheduling methods have been widely used in all types of industries and are being updated and improved by Operations Management and Industrial Engineering researchers. However, some fundamental difficulties caused by the MRP System are extremely resolve or improve. These problems include: (1) Every step in the process must be given a "planned time", which is hard to predict. Plans fall apart without simple real-time remedies; (2) The budgeted time for every step in the process must be given "slack time" to accommodate variability, which will create wasted time when things go well or just average; (3) Waste and inefficiency is built into this approach. As compared to the MRP planning methodology, CONWIP release policy has more robustness in the jobs release times and easier control of the system.

3.1.2 CONWIP Release Policy

Constant Work In Process (CONWIP) is a hybrid of push-and-pull type systems, which attempts to regulate Work In Process (WIP) in the system. In a CONWIP system, the departing jobs send production cards back to the beginning of the line to authorize release of new jobs and it results in a WIP level that is nearly constant [17]. This policy is also known as the Closed Loop Release policy [11].

From a modeling perspective, a CONWIP system can be modeled as a closed queueing network. In contrast, a MRP system can be modeled as on open queueing network. In MRP systems, job release depends on the material requirement plan without any feedback from the system status. Therefore, the number of jobs can vary over time in a MRP system.

Compared to MRP system, the benefits of using CONWIP are:

1. It is easier to implement and provides more robust control.
2. There is less congestion in production flow.

3. It has greater predictability since it keeps constant work in process.

Therefore, the benefits of using CONWIP in shipbuilding process include:

1. It makes the current planning system easier to control, especially for the limited capacity and storage space in shipyard.

2. It can be applied to almost every level of ship production scheduling. For instance, a CONWIP policy can be used to control a entire shipyard and keep a constant number of ships under construction, or it can be used to keep a workshop with a constant number of blocks under construction.

3. CONWIP can increase the efficiency of shipbuilding. It has been showed that for the same Work in Process, CONWIP has higher throughput than MRP.

The remaining question is at which level to introduce the CONWIP control policy in the shipbuilding process and how to plan the number of jobs in the system as the CONWIP level.

3.1.3 The CONWIP Model of Shipbuilding

In this section, a shipbuilding system is developed using the CONWIP planning policy. In ship hull construction, the block is the prime zone for all planning and scheduling. Therefore, the CONWIP release policy is introduced at the block assembly level, i.e., a constant number of block jobs is maintained at the execution level. The flexible block assembly system from chapter 2 is also maintained in this model in order to evaluate the benefits of flexibility. A dry dock workstation is added at the execution level of the model. Therefore, this model includes all major processes in hull construction.

Figure 3.2 illustrates the flexible shipbuilding CONWIP system model. Ship information including number of blocks, block type, and hull erection sequence are provided. In this model, a one dimensional block erection sequence is assumed for

the process in the final drydock station. A two or three dimensional block erection sequence will be studied in the future research. At the strategic level, CONWIP release policy is applied to release jobs.

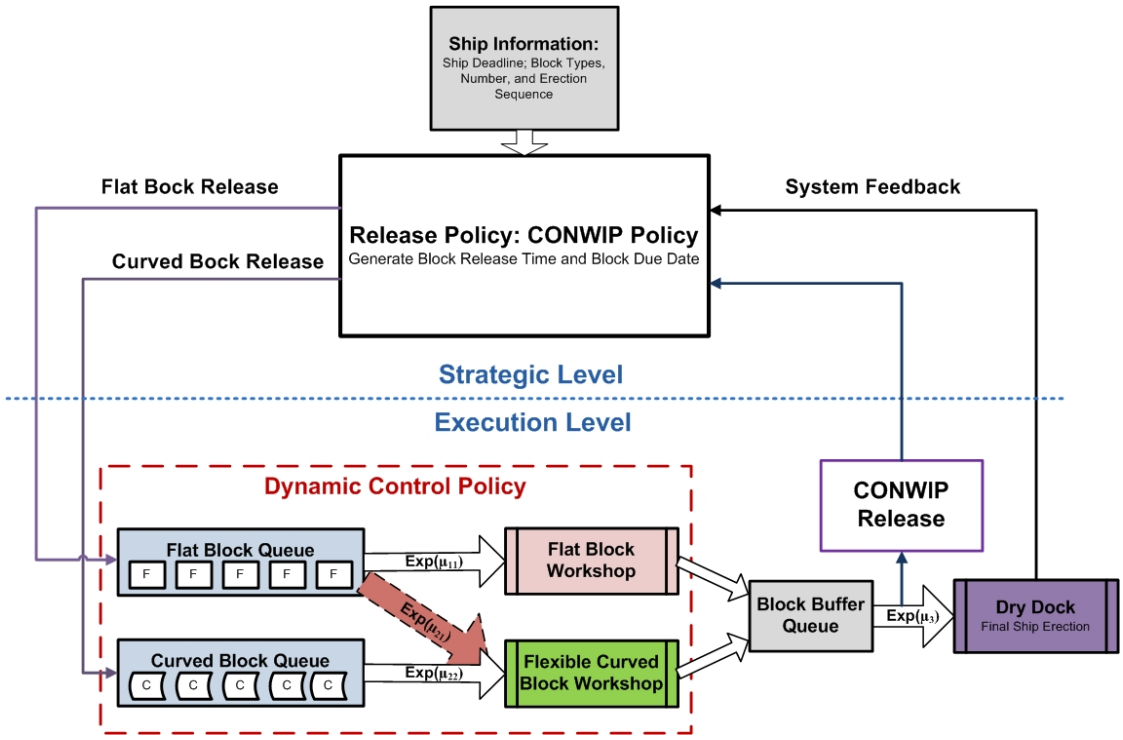


Figure 3.2: Flexible Shipbuilding CONWIP Model

In the execution level, there are two processes: (1) block assembly process, (2) hull erection process. In the block assembly process, there are two types of block workshops: a flat block workshop (a panel production line) and a flexible curved block workshop (a curved block build bay). The flexible block assembly processes allow a curved block workshop to dynamically allocate its build bay to the appropriate mix of flat blocks and curved blocks. In the hull erection process, there is a block buffer and a dry dock. The hull erection process follows the block erection sequence constraint. If the adjacent block is not available in the block buffer, the dry dock will be idle.

This CONWIP shipbuilding system is formulated as a closed flexible queueing

networks with three types of jobs: (1) flat block jobs in the block assembly process, (2) curved block jobs in the block assembly process, and (3) blocks in the final hull erection. There are three distinct workshops: (1) the flat block workshop, (2) the flexible curved block workshop, and (3) the final dry dock workshop. The processing time of these three workshops are exponentially distributed. This CONWIP model holds the same assumption as in the "N" structure model in Chapter II: (1) exponential processing time is assumed for each work station; (2) there is no preemption allowed for any block; and (3) there is no collaboration between workshops.

3.2 Control of Flexible Workshop under CONWIP Discipline: A Simulation Model

3.2.1 Simulation Model

This CONWIP model described above is a very complex model. This is because the drydock workshop needs to follow the block sequence constraint, which make it very difficult to model this system mathematically, and also hard to develop an effective policy to control the flexible workshop. Therefore, a simulation model is developed to address these problems, which is presented in Figure 3.3.

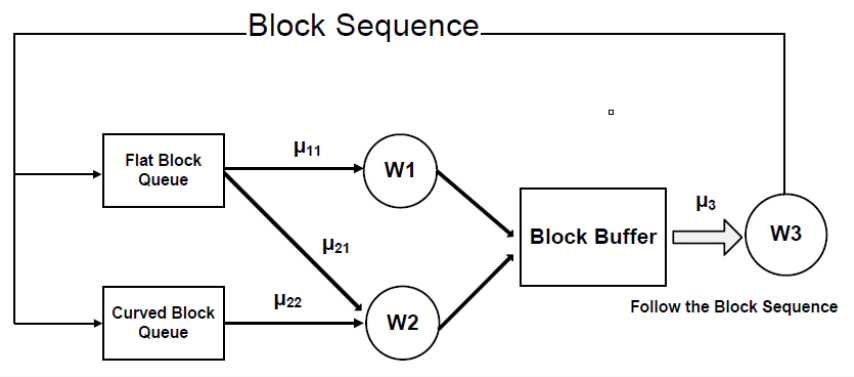


Figure 3.3: CONWIP Shipbuilding Simulation Model

The CONWIP level K is the number of blocks in the system. N is the total number of blocks for a ship. P_1 is the percentage of flat blocks in a ship, and P_2 is

the percentage of curved blocks in a ship. Let block 1 and block 2 denote flat block and curved block respectively. Workshop 1 (W1) represents the flat block workshop, workshop 2 (W2) indicates the flexible curved block workshop, and workshop 3 (W3) denotes the final dry dock workshop. μ_{jk} is the processing rate for block j in workshop k. The same processing rate at workshop 3 is assumed for both flat blocks and curved blocks, i.e., $\mu_{31} = \mu_{32} = \mu_3$. Workshop 3 must to follow the block sequence, which is given at beginning of the simulation.

A major concern investigated here is how to control the flexible workshop while achieving the three following objectives: (1) reducing ship completion time, (2) improving the utilization of drydock, and (3) reducing block buffer size. These three objectives may have positive or negative correlations. However, in this model, we only presents how CONWIP level and control of flexibility impact on these three objectives without analyzing the insight correlations between these objectives.

Each shipyard works on several ships simultaneously and has only a finite amount of resources available; therefore, it is essential that ship completion time be as brief as possible to meet customer demand. This is why reducing ship completion time, which is often also referred to as "makespan", is a main objective of this model. Improving the utilization of drydock, which is equivalent to reducing the idle time of the drydock, is also essential to maintain while seeking control. In ship production scheduling, drydocks often represent the primary choke point to delivering a ship. Planned construction schedules are crucial since shipyards generally have a backlog of ships waiting for a drydock. The time that each ship spends in drydock must be minimized as much as possible to ensure on-time deliveries and to maximize the throughput of work in shipyards. The final objective, reducing the block buffer size before the final hull erection, is also of great importance since the final hull erection

follows a master block erection sequence, and requires a pre-erection area to avoid inserting blocks and to maintain stability in the erection process [35]. Flexibility in the block assembly process may reduce the block buffer size and stabilize the hull erection process, but it is essential to simultaneously maintain these specified objectives.

In the next section, three heuristics are introduced to the simulation model to control the flexible workshop.

Heuristics

Three heuristics to control the flexible curved workshop are examined in the simulation model: (1) First Come, First Served (FCFS); (2) Longest Queue (LQ); and (3) Shortest Processing Time First rule (SPT).

First Come, First Served (FCFS) is a service policy by where the requests of customers or clients are attended to in the order that they arrived, without other biases or preferences. In this simulation model, the flexible workshop under control of FCFS will select the block which arrives the block queue first.

Longest Queue (LQ) is a typically high-performing policy. LQ will select the job at the head of the queue that has the largest number of jobs. The LQ policy has been widely used and researched since it works well for most systems and is simple to implement into any flexible structure. In this model, the flexible workshop under control of LQ policy will select the block from either the flat block queue or curved block queue which has the largest number of blocks waiting in the system.

Shortest Processing Time First rule (SPT) is a control policy that selects the job with the shortest processing time regardless of all other aspects. This type of selection reduces the total number of jobs in queue as quickly as possible. Some research has shown that SPT rule is optimal for minimizing the makespan for single

machine [28].

3.2.2 Simulation Structure

Discrete-event simulation is used to compute and compare the performance of the above heuristics. Discrete-event simulation considers the modeling of a system in which the state variables change instantaneously at discrete, aperiodic points in time. A C-based simulation toolkit, Simlib, is incorporated for efficiency [23] because it facilitates filing and removing records from lists, processing event lists, and computing discrete-time statistics on variables of interest.

Fundamental Aspects of the Simulation Structure

This simulation model consists of seven queues and four events. Seven queues include the information of the block sequence, three workshops, and three block buffer before each workshop. Since there are three types of jobs: (1) build flat blocks, (2) build curved blocks, and (3) block erection, three events will occur due to the departure of each job. Another event is to end simulation, which is predetermined. Figure 3.4 illustrates the structure of the simulation.

1. List of Queues:

(1) LIST-QUEUE-RELEASE: This queue stores the block information including the total number of blocks, block type, and block erection sequence. The fraction of flat and curved blocks are P_1 and P_2 respectively.

(2) LIST-QUEUE-FLAT: This queue stores the flat block jobs which have been released to the system.

(3) LIST-WORKSHOP-FLAT: This queue represents the flat block workshop, and has a capacity of one. The status of this queue is either one (processing flat block), or zero (idle). A non-idling policy is applied here to control the flat block workshop,

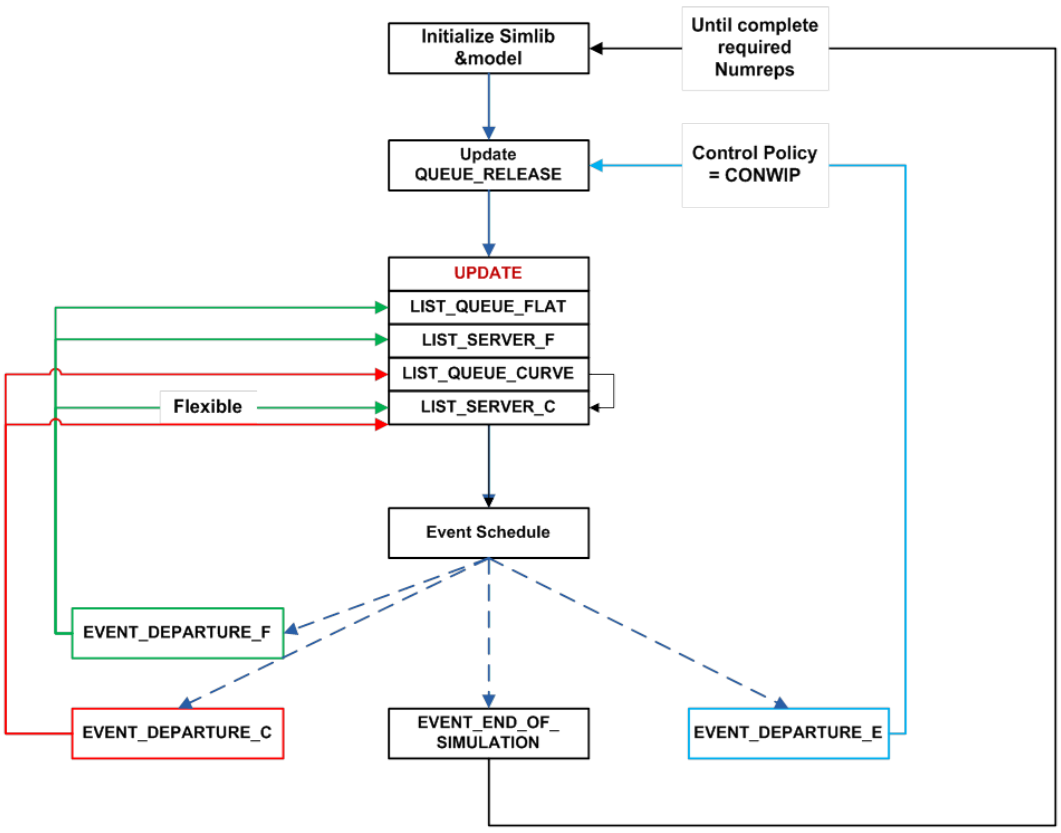


Figure 3.4: Simulation Structure

i.e., never idle flat block workshop as long as the flat block queue is not empty.

(4) LIST-QUEUE-CURVE: This queue stores the curved block jobs that have been released to the system.

(5) LIST-WORKSHOP-CURVED: This queue represents the flexible curved block workshop which has a capacity of one. The status of this queue is either one (processing either flat block or curved block), or zero (idle). The processing time differs when the flexible curved block workshop is working on different type of block. The heuristics listed in the previous section will be applied here to control the flexible workshop to select the block.

(6) LIST-QUEUE-BOCK-BUFFER: This queue stores the finished blocks awaiting final hull erection, and ranks jobs based on the erection sequence.

(7) LIST-WORKSHOP-ERECTION: This queue represents the drydock workshop which has capacity of one. The status of this queue is either one or zero. This workshop needs to follow the block sequence. If the required block is not available in the block buffer, the drydock workshop will be idle.

2. List of Events:

(1) EVENT-DEPARTURE-F: Flat block workshop finishes assembling a flat block.

(2) EVENT-DEPARTURE-C: Flexible curved block workshop finishes assembling either a curved block or flat block.

(3) EVENT-DEPARTURE-E: A block has been erected in the dry dock. This event will trigger to release next block to the system.

(4) EVENT-END-SIMULATION: The simulation is complete.

The major steps and logics in each event will be introduced in the next section.

Simulation Demonstration

The simulation starts with initializing the "Simlib" toolkit and input the ship design information. N blocks are generated using a uniform random number generator. These blocks are ordered based on sequence ID and stored in the LIST-QUEUE-RELEASE. When the system commences, blocks are released to either LIST-QUEUE-FLAT or LIST-QUEUE-CURVE, which controlled by the CONWIP level. A block is released if and only if WIP is less than the CONWIP level.

The detailed logic in each event are as follows:

(1) Block Finished by Flat Block Workshop (EVENT-DEPARTURE-F)

Figure 3.5 illustrates the departure process from the flat block workshop. When a block is finished by the flat block workshop, it is removed from the LIST-WORKSHOP-FLAT, delivered to the erection workshop directly if the erection workshop is idle and the block satisfies the block erection sequence constraint. Otherwise, it is inserted

in the LIST-QUEUE-BLOCK-BUFFER based on its sequence ID. Subsequently, the first block in the LIST-QUEUE-FLAT is sent to the flat block workshop if the LIST-QUEUE-FLAT is not empty. Otherwise the flat block workshop remains idle until a LIST-QUEUE-FLAT has an available block.

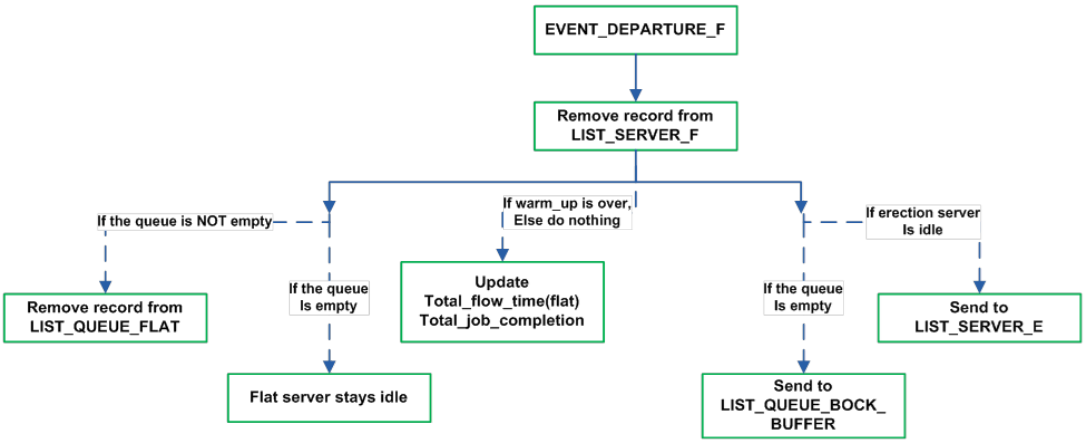


Figure 3.5: Event Departure of Flat Block

(2) Block Finished by Flexible Curved Block Workshop (ERECTION-DEPARTURE-C)

Figure 3.6 illustrates the departure process from the flexible workshop. When a block is completed by the flexible curved block workshop, the finished block is removed from the LIST-WORKSHOP-CURVE, and delivered to the erection workshop directly if the erection workshop is idle and the block satisfies the block erection sequence constraint. Otherwise, it is inserted to the appropriate position of the LIST-QUEUE-BLOCK-BUFFER based on its sequence ID. Following this, depending on the control policy, either a flat block or a curve block will be sent to the flexible curved block workshop. If both the LIST-QUEUE-FLAT and the LIST-QUEUE-CURVE are empty, the flexible curved block workshop remains idle.

(3) Block Completed by Erection Workshop (EVENT-DEPARTURE-E)

The completion of a block erection triggers the CONWIP system release next block

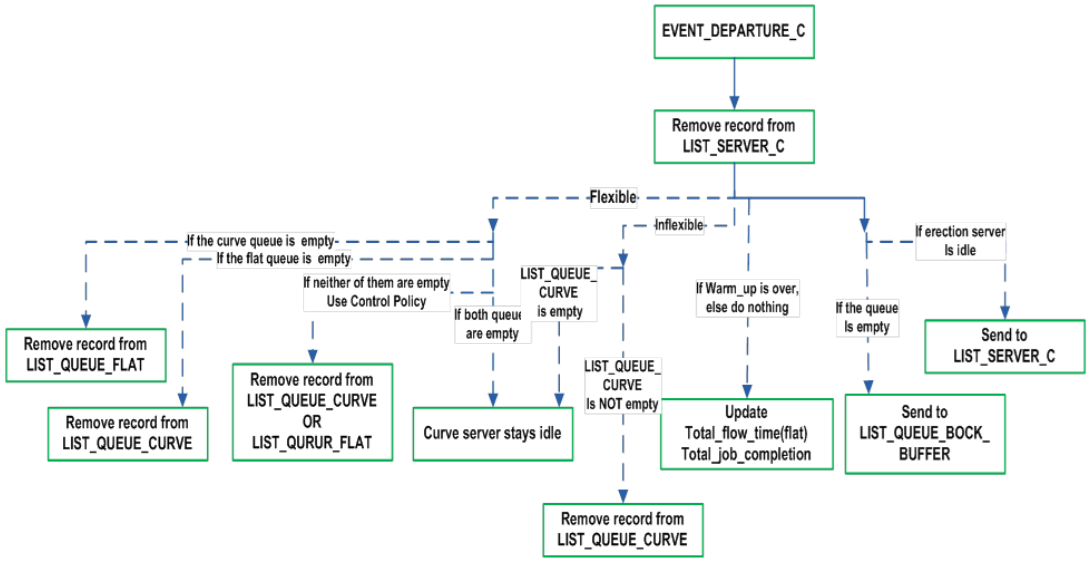


Figure 3.6: Event Departure of Curved Block

to the system. Figure 3.7 shows the process of this event. Once a block has finished the erection process in the drydock workshop, a block from LIST-QUEUE-RELEASE is released to either LIST-QUEUE-FLAT or LIST-QUEUE-CURVE to maintain the constant WIP level. If the block following the finished block in the sequence is available from LIST-QUEUE-BLOCK-BUFFER, it will be assigned to the drydock workshop. Otherwise, the drydock workshop will remain idle until the block with correct sequence ID arrives at the LIST-QUEUE-BLOCK-BUFFER. After the completion of all blocks in the LIST-QUEUE-RELEASE and no blocks remain in the system, the simulation for one ship terminates. Afterwards, either a new replication of a ship is conducted or the simulation is completed.

3.2.3 Result Analysis

This system was simulated using a computer program written in VBA. Each simulation run began with an empty system and ended after 100,000 blocks exited the line (assume 300 blocks per ship), including a warm-up period of 1,000 blocks. Each run was replicated 50 times. To reduce variability between runs, common

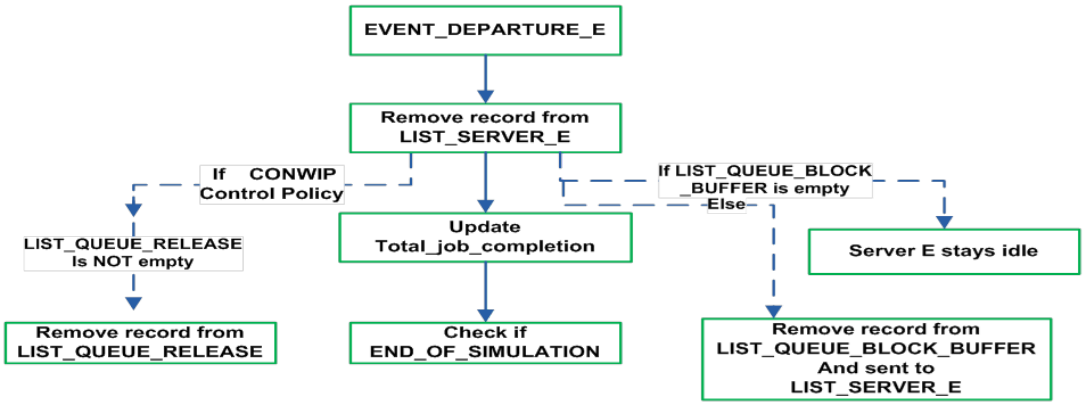


Figure 3.7: Event Departure from Final Erection Process

random numbers were used for each policy, and each workshop processing time had its own random number stream. At a confidence level of 95%, all standard errors were within 0.5%.

Simulation analyzes the performance of the inflexible systems and flexible systems in terms of (1) ship completion time, (2) drydock utilization, and (3) drydock block buffer size. Three control policies are applied to control the flexible curved block workshop: (1) FCFS, (2) LQ, and (3) SPT. The criteria are the mean of ship completion time, drydock utilization, and drydock block buffer size.

Test-Suite

P_1 is the fraction of flat blocks, and P_2 ($P_2 = 1 - P_1$) is the fraction of curved blocks. The values of P_1 and P_2 highly depend on the ship type. There are normally more flat blocks in commercial ships like cargo ships and tankers since there will be fewer curved hull structures and more flat hull structures for the efficient storage. These two parameters are very essential to test the benefits of flexibility. When $P_1 > P_2$, the flexibility performs better than the inflexible system since the curved block workshop can assist the flat block workshop to produce flat blocks. When $P_1 < P_2$, the flexible workshop will be busy with curved blocks and have no extra

capacity to assist the flat block workshop. For this reason, setting $P_1 = 0.7$ and $P_2 = 0.3$ is suitable for the heuristic test cases design.

Two test cases of system dynamics were picked to illustrate some interesting results. Table 3.1 provides the parameter value for these two cases. In case 1, $\mu_{11} < \mu_{21}$ and $\mu_{11} < \mu_{22}$, i.e., the flexible curved block workshop is more efficient than the flat block workshop. In case 2, $\mu_{11} > \mu_{21}$ and $\mu_{11} > \mu_{22}$, i.e., the flexible curved block workshop is not as efficient as the flat block workshop. These two cases cover both scenarios of an efficient and inefficient flexible workshop.

μ_3 is the processing rate of final hull erection for both flat and curved blocks. When μ_3 is smaller than the sum of μ_{11} and μ_{22}, the drydock workshop becomes the bottleneck of the production system. The problem of drydock being the bottleneck is that the advantage of flexibility will not be significant since most blocks will be kept in the erection block buffer, which will cause the idling of the flexible workshop. Therefore $\mu_3 = 3$ would balance the capacity between block assembly process and final hull erection process.

Case 1	Case 2	Description
$\mu_{11} = 1$	$\mu_{11} = 1$	Processing rate for flat block workshop producing flat blocks
$\mu_{21} = 1.2$	$\mu_{21} = 0.9$	Processing rate for flexible workshop producing flat blocks
$\mu_{22} = 1.1$	$\mu_{22} = 0.8$	Processing rate for flexible workshop producing curved blocks
$\mu_{31} = 3$	$\mu_{31} = 3$	Processing rate for erection

Table 3.1: CONWIP Model Simulation Test-Suite

Ship Completion Time

The first criterion tested was ship completion time. While examining completion times, it is most significant to note which heuristic, either FCFS, LQ, or SPT, has the best performance in controlling the flexible curved block workshop, and how much performance improved by introducing a flexible system compared with the inflexible system. Under these circumstances, an inflexible system is defined as a curved block

workshop that does not have flexibility and can only work on the curved block. Therefore, there is no control policy needed for the curved block workshop in the inflexible system.

Figure 3.8 presents the ship completion time of the system under each heuristics in case 1. In this figure, the y-axis is the ship completion time, and the x-axis is the CONWIP level, or the number of blocks in the system, which ranges from 1 to 20.

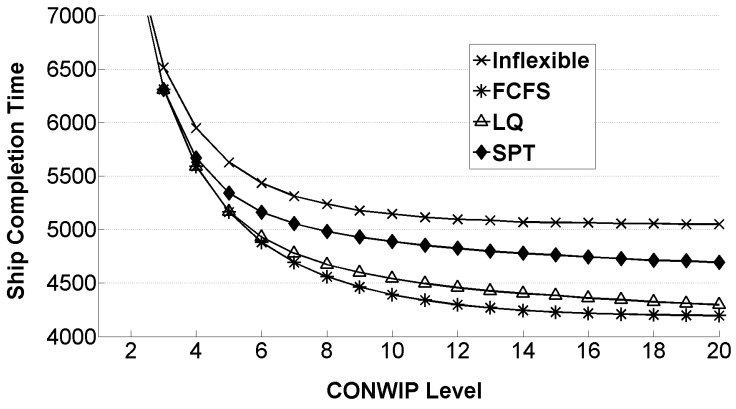


Figure 3.8: Case 1: Ship Completion Time Comparison

The results of this figure show that the ship completion time under the FCFS policy is the shortest among all the control policies, which indicates that the FCFS policy has the best performance in reducing the ship completion time. The performance of LQ is very close to the performance of FCFS, and the gap between these two policies performance decreases as the CONWIP level K increases. In this test case, $\mu_{21} > \mu_{22}$ causing SPT policy to give the flat block priority, i.e., the flexible workshop only works on the curved block until the curved block queue is empty. The performance of SPT is not as good as FCFS and LQ, but still better than the inflexible system. It shows the benefits of the flexibility since the inflexible system has the worst performance. On average, having flexibility and employing FCFS reduced the ship completion time by 17%.

The results in Figure 3.8 also provide information on how to set the CONWIP level K. It shows that when $K > 10$, the improvement by increasing K is not significant. Therefore, under this circumstance, it is more efficient for a shipyard to keep the WIP level K around ten to maintain a high system throughput and a low WIP level.

In case 2, the FCFS policy still exhibits the best performance as shown in Figure 3.9. On average, the FCFS flexible system reduced the ship completion time by 11% as compared to the inflexible system. In this case, $\mu_{11} > \mu_{21}$, which indicates that the flexible workshop is not as efficient as the flat block workshop. Therefore, the improvement by flexibility in case 2 is smaller than the improvement in case 1.

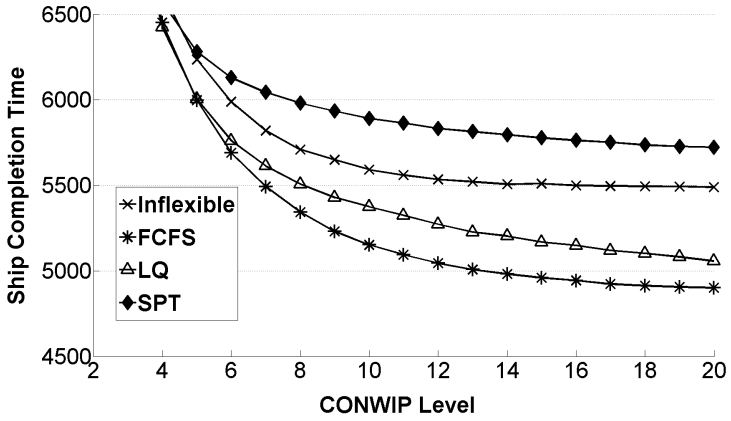


Figure 3.9: Case 2: Ship Completion Time Comparison

Figure 3.9 also shows that the SPT policy has the worst performance. In this case, the SPT policy gives flat block priority, i.e., as long as there is flat block in the queue, flexible workshop will work on the flat block instead of curved block. However, the flexible workshop is not efficient and SPT policy keeps the flexible workshop processing flat blocks, which causes a high congestion for the curved block. This leads to the problem of the lack of curved blocks for the final drydock workshop to process, which follows the block sequence to erect the blocks. Therefore, the flexible workshop under control the SPT policy slows down the entire system and increases

the ship completion time. This concludes that the flexible system with a bad control policy can make the system even worse. Developing an efficient and robust control policy for the flexible resource is therefore the most essential aim of the flexible system.

Drydock Utilization

One common goal in most shipyards is to use the drydock efficiently. Therefore, the second criterion in this simulation is the drydock utilization. Figure 3.10 illustrates results form the case 1 for the drydock utilization. The y axis in this figure is the drydock utilization which ranges from 0 to 1. The results show that the flexible system under control of FCFS policy constantly has the largest drydock utilization over all CONWIP levels. The inflexible system has the lowest drydock utilization. On average, flexibility controlled by the FCFS policy increased the drydock utilization by 19% as compared to the inflexible system.

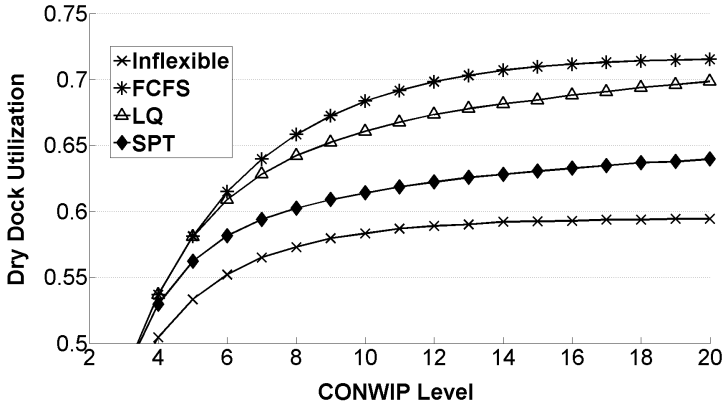


Figure 3.10: Case 1: Drydock Utilization Comparison

Figure 3.11 illustrates case 2 results which shows the flexible system under control of FCFS policy has the largest drydock utilization. The flexible system under control of SPT policy has the worst performance. Considering both results for the ship completion time and drydock utilization, these results indicate that increasing the

drydock utilization enhances the system throughput and reduce the ship completion time. This results also indicate the positive correlation between the ship completion time and the drydock utilization.

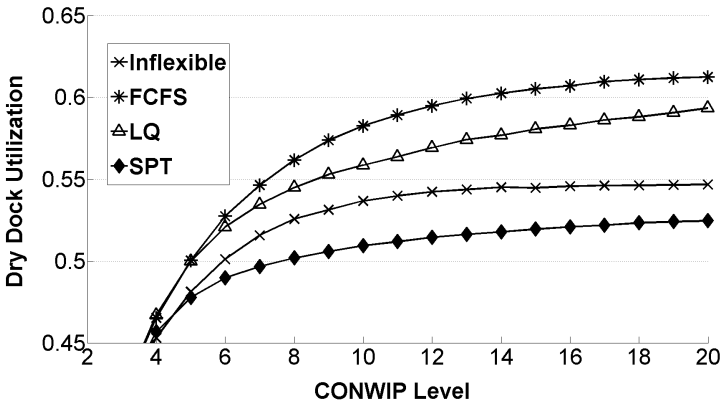


Figure 3.11: Case 2: Drydock Utilization Comparison

Average Block Buffer Size for Pre-erection

To maintain a high utilization of the drydock, the storage of finished blocks is inevitable. Using flexibility can reduce the number of finished blocks required for the final erection and allow for the same possible utilization of the drydock. Figure.3.12 illustrates the average length of the block buffer placed before the final hull erection. The flexible system operated with FCFS can significantly reduce the block buffer size as compared to the inflexible system under the same CONWIP level. Recalling that the flexible system under FCFS performs better in every respect, we note that the size of the queue at the block buffer grows linearly, whereas the flexible FCFS system has a sub-linear growth rate. This reveals that another benefit of flexibility can stabilize the number of blocks in the queue buffer before the final drydock by increasing the number of blocks in the entire system.

The simulation results above show that: (1) flexibility can significantly reduce the ship completion time, increase the drydock utilization, and reduce the block buffer

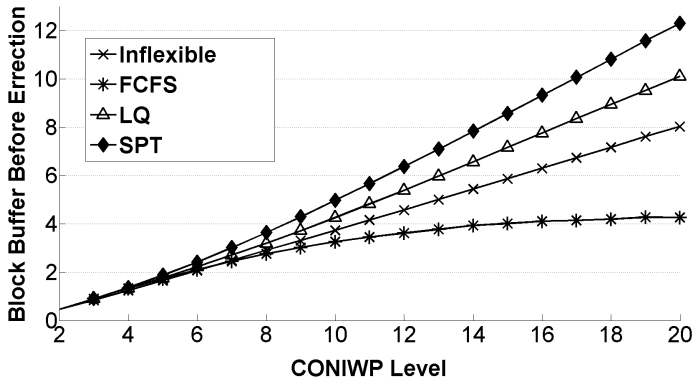


Figure 3.12: Case 1: Block Buffer Size

size for the final erection process; (2) the FCFS policy has the best performance among all the test policies; and (3) the SPT policy is unstable for controlling the flexibility under CONWIP release with a block sequence constraint.

The discrete-event simulation model provides insight of the complex CONWIP shipbuilding system and demonstrates the benefit of flexibility. However, finding an optimal control policy of the flexible workshop is quite difficult due to the block sequence constraint. In the next section, a mathematical model is developed to search for the structure of the optimal control policy with a simplified block sequence constrain.

3.3 Dynamic Two-Loop CONWIP Model

In this section, a two-loop CONWIP model formulated using Markov Decision Processes is developed to search for the structure of an optimal policy to control the flexible workshop. There are two CONWIP loops in this model; one loop aims to keep a constant number of flat blocks in the system, while the other loop aims to keep a constant number of curved blocks in the system. The reason for using two CONWIP loops is to simplify the job release process. In the previous simulation

model, blocks are released following the block assembly sequence, and the total number of flat blocks and curved blocks is a constant number. However, the separate numbers of flat blocks or curved blocks are not constant, and it is very difficult to use Markov Decision Processes to formulate the problem since it requires a very large state variables to record each possible number of flat blocks and curved block. Therefore, two CONWIP loops are used to simplify the model. Figure 3.13 illustrates the two-loop CONWIP model.

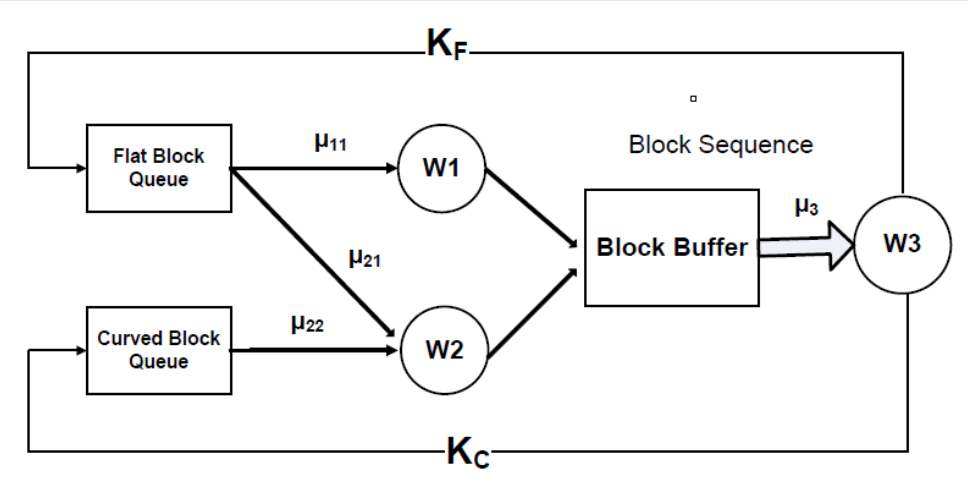


Figure 3.13: CONWIP MDP Model

Let K_F denote the number of flat blocks in the system, or the CONWIP level for flat blocks, and K_C denote the number of curved blocks in the system, or the CONWIP level for the curved blocks. When there is a flat block finished at final stage (the drydock), a flat block will be released to the system. This same process occurs for curved blocks as well.

At the execution level, there are three types of jobs: (1) flat block jobs in the block assembly process, (2) curved block jobs in the block assembly process, and (3) blocks in the final hull erection. There are three workshops: (1) the flat block workshop, (2) the flexible curved block workshop, and (3) the final dry dock workshop.

The processing time of these three workshops are exponentially distributed. μ_{11} is the processing rate of workshop 1 (flat block workshop) working on flat blocks; μ_{21} is the processing rate of workshop 2 (curved block workshop) working on flat blocks; μ_{22} is the processing rate of workshop 2 (curved block workshop) working on curved blocks; μ_3 is the processing rate of workshop 3 (drydock) working on either flat or curved blocks (i.e., we assume flat and curved blocks have the same processing time in the final hull construction, because labors can be adjusted to render these capacities equal, which we believe to be more desirable for attaining better operational performance). $\mu_{11} > \mu_{21}$ since flat block workshop is more efficient in processing flat blocks. No pre-emption or collaboration is allowed for any single block.

Let γ denote the reward for system throughput, i.e., γ is earned when each block is completed at the final hull construction. P_F and P_C are the costs per unit time per unit of capacity of workshop 1 (flat block workshop) and workshop 2 (curved block workshop). θ_F and θ_C represent the decision of capacity investment for workshops 1 and 2, respectively. The objective is to maximize the system benefit, which is the long run expected discounted throughput reward minus investment cost: $\gamma \cdot TH(K) - P_F\theta_F - P_C\theta_C$.

The notations of this two-loop CONWIP model are summarized in Table 3.2.

3.3.1 Markov Decision Process Formulation

To formulate a MDP model, it is necessary to define the system state variables first. Let x_i be the number of jobs in queue i for $i = 1, 2, 3, 4$. x_1 is the number of flat blocks in the flat block queue, which includes the block being processed at workshop 1 or workshop 2. x_2 is the number of curved blocks in the curved block queue, including the block being processed at workshop 2. x_3 is the number of flat blocks at the block buffer, including the block being processed at workshop 3. x_4 is

Notation	Description
K_F	Number of flat block in the System
K_C	Number of curved block in the System
L	Length of the block sequence period
$\tau(i), i \in 1, ..., L$	Block sequence
θ_F	Capacity of flat block workshop
θ_C	Capacity of flexible curved block workshop
μ_{11}	Processing rate of flat block workshop
μ_{21}	Processing rate of flexible workshop on flat block
μ_{22}	Processing rate of flexible workshop on curved block
μ_3	Processing rate of final hull construction rate
γ	Reward per block
P_F	Cost per block per unit of time of flat block workshop
P_C	Cost per block per unit of time of flexible workshop

Table 3.2: Notations

the number of curved blocks at the block buffer, including the block being processed at workshop 3. These variables are important in determining the total amount of flat and curved blocks in the system via the equations, $x_1 + x_3 = K_F$ and $x_2 + x_4 = K_C$.

Let s_j denote the status of workshop j for $j = 1, 2, 3$. The flexible workshop 2 can work on flat blocks, (where $s_2 = 1$), curved blocks (where $s_2 = 2$), or idle (where $s_2 = 0$). We assumed that workshop 1 has priority in processing flat blocks (which means $s_1 = 1$ either $x_1 > 1$ or $x_1 > 0$ and $s_2 \neq 1$). s_3 denotes the index of block assembly sequence for workshop 3 (not the type of block that workshop 3 processes). $s_3 = 1, 2, ..., L$, L is the length of block assemble sequence. s_3 indexes the stage of the block final assembly sequence. $\tau(s_3)$ represents the block type at stage s_3 in the sequence, where $\tau(s_3) = 1$ (a flat block), or $\tau(s_3) = 2$ (a curved block). If the necessary type of block is unavailable, workshop 3 is left idling until one arrives.

The system state is $\{x_1, x_2, s_2, s_3\}$. x_3 and x_4 are not necessary for the state since $x_1 + x_3 = K_F$ and $x_2 + x_4 = K_C$. The status of s_1 also depends on x_1 and s_2 (where $s_1 = 1$ either $x_1 > 1$ or $x_1 > 0$ and $s_2 \neq 1$). Therefore, $\{x_1, x_2, s_2, s_3\}$ are the necessary variables to describe this system.

Optimality Equations

Let $J_k(\mathbf{x}, \mathbf{s})$ denote the optimal k-stage cost to go function, and set the terminal cost function, $J_0(\mathbf{x}, \mathbf{s}) = 0$. The state space is $\{0 \leq x_1 \leq K_F$, $\{0 \leq x_2 \leq K_C$, $s_2 = 0, 1, 2$, and $s_3 = 1, 2, ..., L$.

Some indicator functions are defined as follows: $u_1 = \mathbb{1}\{x_1 > 1 \cup (x_1 > 0 \cap s_2 \neq 1\}$; $u_{21} = \mathbb{1}\{s_2 = 1 \cap x_1 \geq 1\}$; $u_{22} = \mathbb{1}\{s_2 = 2 \cap x_2 \geq 1\}$; $u_3 = \mathbb{1}\{\tau(s_3) = 1 \cap x_3 > 0\}$; $u_4 = \mathbb{1}\{\tau(s_3) = 2 \cap x_4 > 0\}$. Let Λ denote the uniformization factor and $\Lambda = \theta_F \mu_{11} + \theta_C \cdot \max\{\mu_{21}, \mu_{22}\} + \mu_3$.

This particular MDP is rather difficult. So for simplicity of exposition, we write out a problem for the case of total expected β-discounted cases. β is the discount factor. This MDP with discount factor is more intuitive and can easily be converted to the average cost case for numerical computation. Our methodology is useful for finite or infinite horizon total expected discounted cost, or could be extended to the long run average cost per unit time. Because we do not need a high-fidelity financial model to gain the high-level insights, the criterion assumed is that the infinite horizon average cost per unit time. So with an investment cost of $P_F \theta_F + P_C \theta_C$ per period and reward of $\gamma \cdot TH(K)$, the total investment cost and rewards can be compared on an annual basis. The recursive value function is:

(3.1)

$$
\begin{aligned}
J_{k+1}(x_1,x_2,s_2,s_3) = \frac{1}{\Lambda}\big[&(u_3+u_4)\mu_3\cdot\gamma - P_F\theta_F - P_C\theta_C \\
&+ \beta\big\{u_{21}\{u_1\theta_F\mu_{11}J_k(x_1-1,x_2,s_2=1,s_3) + \theta_C\mu_{21}J_k(x_1-1,x_2,s_2=0,s_3) \\
&+ (\Lambda - u_1\theta_F\mu_{11} - (u_3+u_4)\mu_3 - \theta_C\mu_{21})J_k(x_1,x_2,s_2=1,s_3)\} \\
&+ u_{22}\{u_1\theta_F\mu_{11}J_k(x_1-1,x_2,s_2=2,s_3) + \theta_C\mu_{22}J_k(x_1,x_2-1,s_2=0,s_3) \\
&+ (\Lambda - u_1\theta_F\mu_{11} - (u_3+u_4)\mu_3 - \theta_C\mu_{22})J_k(x_1,x_2,s_2=2,s_3)\} \\
&+ u_3\mu_3 J_k(x_1+1,x_2,s_2,s_3\oplus 1) + u_4\mu_3 J_k(x_1,x_2+1,s_2,s_3\oplus 1) \\
&+ \mathbb{1}_{\{x_1\geq 2,x_2=0,s_2=0\}}\max\{[\textit{SERVE 1}]u_1\theta_F\mu_{11}J_k(x_1-1,x_2,s_2=1,s_3) \\
&+ \theta_C\mu_{21}J_k(x_1-1,x_2,s_2=0,s_3) + (\Lambda - u_1\theta_F\mu_{11} - (u_3+u_4)\mu_3 - \theta_C\mu_{21})J_k(x_1,x_2,s_2=1,s_3), \\
&[\textit{IDLE}]u_1\theta_F\mu_{11}J_k(x_1-1,x_2,s_2=0,s_3) + (\Lambda - u_1\theta_F\mu_{11} - (u_3+u_4)\mu_3)J_k(x_1,x_2,s_2=0,s_3)\} \\
&+ \mathbb{1}_{\{x_1\leq 1,x_2\geq 1,s_2=0\}}\max\{[\textit{SERVE 2}]u_1\theta_F\mu_{11}J_k(x_1-1,x_2,s_2=2,s_3) \\
&+ \theta_C\mu_{22}J_k(x_1,x_2-1,s_2=0,s_3) + (\Lambda - u_1\theta_F\mu_{11} - (u_3+u_4)\mu_3 - \theta_C\mu_{22})J_k(x_1,x_2,s_2=2,s_3), \\
&[\textit{IDLE}]u_1\theta_F\mu_{11}J_k(x_1-1,x_2,s_2=0,s_3) + (\Lambda - u_1\theta_F\mu_{11} - (u_3+u_4)\mu_3)J_k(x_1,x_2,s_2=0,s_3)\} \\
&+ \mathbb{1}_{\{x_1\geq 2,x_2\geq 1,s_2=0\}}\max\{[\textit{SERVE 1}]u_1\theta_F\mu_{11}J_k(x_1-1,x_2,s_2=1,s_3) \\
&+ \theta_C\mu_{21}J_k(x_1-1,x_2,s_2=0,s_3) + (\Lambda - u_1\theta_F\mu_{11} - (u_3+u_4)\mu_3 - \theta_C\mu_{21})J_k(x_1,x_2,s_2=1,s_3), \\
&[\textit{SERVE 2}]u_1\theta_F\mu_{11}J_k(x_1-1,x_2,s_2=2,s_3) + \theta_C\mu_{22}J_k(x_1,x_2-1,s_2=0,s_3) \\
&+ (\Lambda - u_1\theta_F\mu_{11} - (u_3+u_4)\mu_3 - \theta_C\mu_{22})J_k(x_1,x_2,s_2=2,s_3), \\
&[\textit{IDLE}]u_1\theta_F\mu_{11}J_k(x_1-1,x_2,s_2=0,s_3) + (\Lambda - u_1\theta_F\mu_{11} - (u_3+u_4)\mu_3)J_k(x_1,x_2,s_2=0,s_3)\} \\
&+ \mathbb{1}_{\{x_1\leq 1,x_2=0,s_2=0\}}\{u_1\theta_F\mu_{11}J_k(x_1-1,x_2,s_2=0,s_3) \\
&+ (\Lambda - u_1\theta_F\mu_{11} - (u_3+u_4)\mu_3)J_k(x_1,x_2,s_2=0,s_3)\}\big\}\big].
\end{aligned}
$$

This recursive value function $J_k(\mathbf{x},\mathbf{s})$ is very complex, mainly due to two unique

aspects of this model: (1) non-preemption is required in this model, which adds difficulty, and (2) the MDP model allows idling of the flexible workshop. The non-preemption was also assumed in the "N" structure queueing network in Chapter II. This assumption is applied to this CONWIP model for the same reason as that for the shipbuilding constraint. Allowing idling of the flexible workshop in this MDP is due to the fact that the flexible workshop is not as efficient as the flat block workshop, and it may be optimal to idle the flexible workshop under some circumstances.

Using this MDP model, some numerical examples are developed in the next section to show how to optimally control the flexible workshop.

3.3.2 Control of Flexible Workshop

One major concern in this research is how to control the flexible workshop under the CONWIP discipline. In the previous section **3.2**, the simulation results showed that the First Come, First Served (FCFS) policy has the best performance. Although the MDP model is different from the simulation model, since the MDP model has two CONWIP loops and a simplified block sequence, it can still provide insight into controlling the flexible workshop under job sequence constraint in the CONWIP model. A test case was designed to show the structure of the optimal policy to control the flexible workshop, which is illustrated in Table 3.3.

As shown in Table 3.3, the length of the block sequence period L is 8, with 3 curved blocks and 5 flat blocks. There are normally more flat blocks than curved blocks in a ship depending on the ship type. $N_F = 10$ and $N_C = 6$, therefore, $\frac{N_F}{N_F+N_C}$ = fraction of the flat blocks in the sequence. To test the benefit of the flexibility, any situation of the drydock workshop being the bottleneck is avoid. Otherwise, a lot of blocks will stay at the block buffer before the final drydock, and the flexibility will not significantly contribute to the system. Therefore, we let $\mu_3 = \theta_F\mu_{11} + \theta_C\mu_{21}$, to

Notation	Description	Test Value
K_F	Number of flat block in the System	10
K_C	Number of curved block in the System	6
L	Length of the block sequence period	8
$\tau(i), i \in 1, ..., L$	Block sequence	CFFCFFFC
θ_F	Capacity of flat block workshop	2
θ_C	Capacity of flexible curved block workshop	2
μ_{11}	Processing rate of flat block workshop	3
μ_{21}	Processing rate of flexible workshop on flat block	2
μ_{22}	Processing rate of flexible workshop on curved block	2
μ_3	Processing rate of final hull construction rate	10
γ	Reward per block	5
P_F	Cost per block per unit of time of flat block workshop	1
P_C	Cost per block per unit of time of flexible workshop	1.5

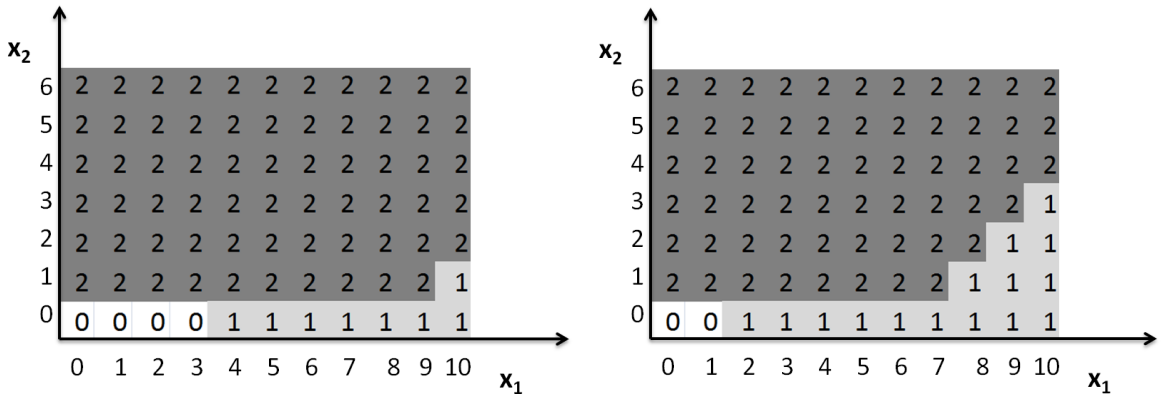
Table 3.3: CONWIP MDP model Test Case

guarantee that the drydock is not the bottleneck.

Structure of Optimal Control Policy

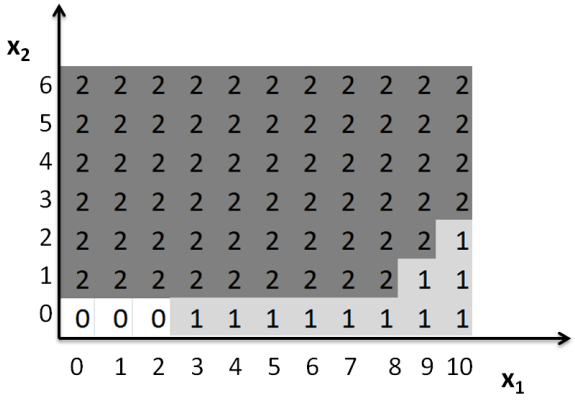
The first problem we investigated is that which block, a flat block or a curved block, should be assigned to the flexible workshop when it is available. Figure 3.14 presents the structure of the optimal control policy in the test case in Table 3.3. The x-axis is x_1, the number of flat blocks at block assembly process, where $x_1 \in 0, 1, ..., N_F$, and the y-axis is x_2, the number of curved blocks at block assembly process, where $x_2 \in 0, 1, ..., N_C$. There are three control actions for the flexible workshop: "0" which means the flexible workshop is idle; "1" which means the flexible workshop is working on a flat block; or "2" which means the flexible workshop is working on a curved block.

When $x_1 \leq 1$ and $x_2 = 0$, there are no blocks available for the flexible workshop, since only flat blocks are assigned to the flat block workshop. The control policy of the flexible workshop also depends on the stage of the sequence. There are 8 stages in the sequence and the sequence is "CFFCFFFC". Only four cases of the first 4 stages are presented here to show that the control policy greatly depends on the sequence

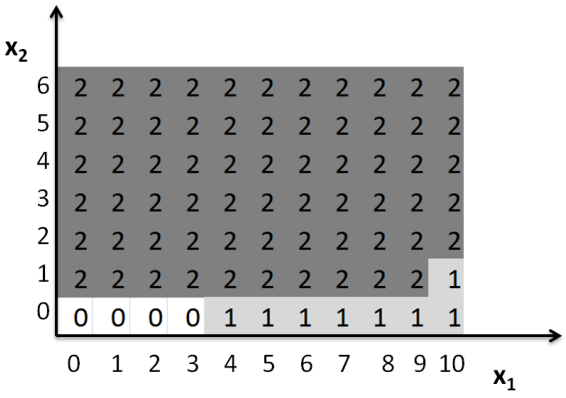


(a) $\tau(1) = 2$

(b) $\tau(2) = 1$



(c) $\tau(3) = 1$



(d) $\tau(4) = 2$

Figure 3.14: Optimal Control Policy of Flexible Workshop

stage. The other four cases of the later stages are not illustrated here since they have similar policy structure. The first 4 stages block sequence is "CFFC". Figure 3.14 (a) shows a case where the drydock is at sequence stage 1 and $\tau(1) = 2$, i.e., the drydock needs a curved block. Figure 3.14(b) shows a case where the drydock is at sequence stage 2 and $\tau(2) = 1$, i.e. the drydock needs a flat block. Figure 3.14 (c) presents a case where the drydock is at sequence stage 3 and $\tau(3) = 1$, and Figure 3.14(d) shows a case where the drydock is at sequence stage 4 and $\tau(4) = 2$.

The results in Figure 3.14 show that the structure of the optimal control policy is a threshold type policy. When x_2 is smaller than a threshold, it is more favorable to assign flat block to the flexible workshop; otherwise, it is optimal to assign curved

block to the flexible workshop. Figure 3.14 (a), (c), and (d) show that when $x_2 = 0$, it is optimal to idle the flexible workshop until $x_1 >= 3$. This is because, in this case, the flexible workshop is not as efficient as the flat block workshop, and in order to avoid starving the flat block workshop, the flexible workshop must remain idle until there are enough flat blocks for both workshops.

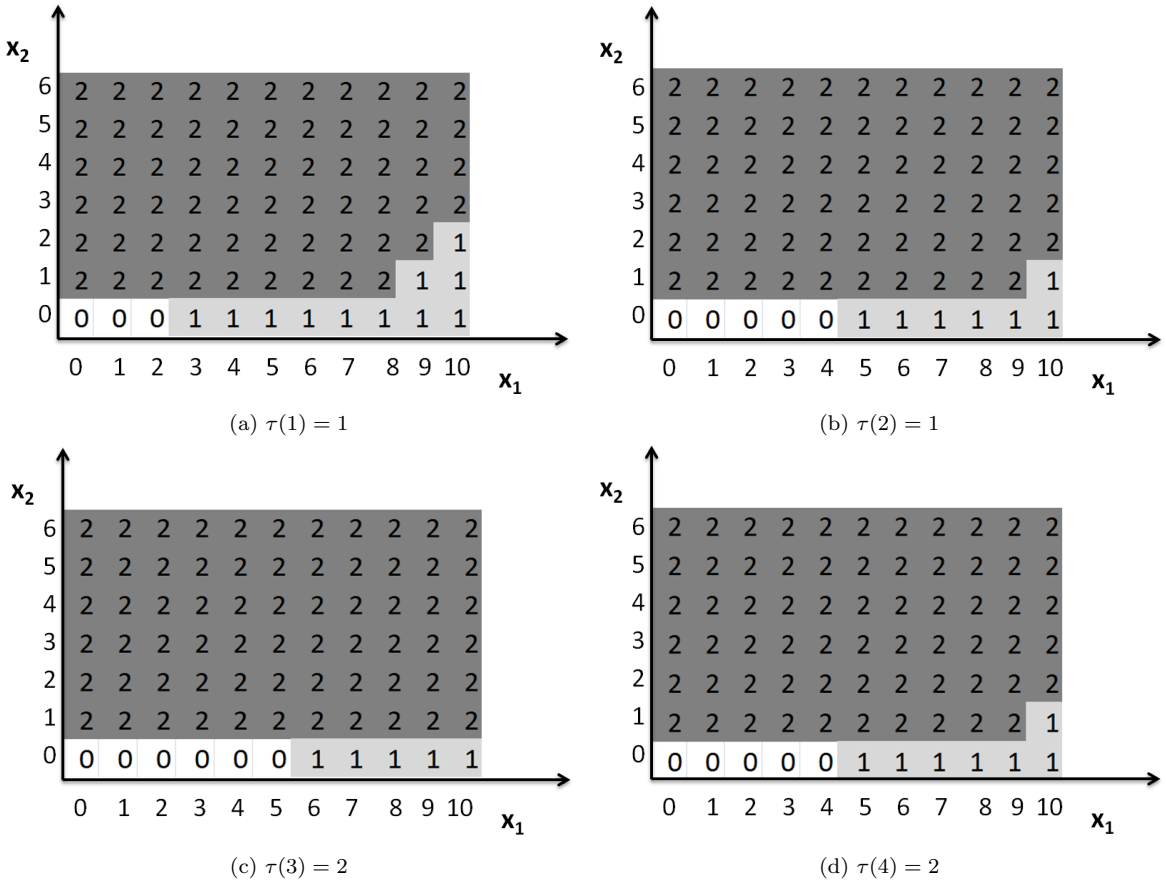
Figure 3.14 also indicates that there is idling of the flexible workshop when $\tau(i) = 2$. The reason for this is that when the final drydock currently needs a curved block, it is more cost effective to keep the flexible workshop available for curved blocks instead of processing a flat block. However, in the case of Figure 3.14 (b), it is not optimal to idle the flexible workshop and it should assist the flat block workshop to process the flat block anytime when flat blocks are waiting in the queue. This is due to the fact that $\tau(2) = 1$, the drydock needs a flat block instead of curved block. The different results for these two cases demonstrate how the sequence constraints affect the control policy.

Impact of Change of Block Sequence

To test how the change of block sequence impacts on the control policy structure, another sequence "FFCCCFFF" is tested in the MDP model. The percentage of flat blocks and curved blocks remain the same, 5 curved blocks and 3 flat blocks. The difference between two sequences are that the sequence "CFFCFFFC" has all the curved blocks separately distributed in the sequence, while the sequence "FFC-CCFFF" has all curved blocks together in the middle.

Figure 3.15 illustrates the first 4 stages of the control policy, which is "FFCC". As compared to Figure 3.14 (c) and (d), Figure 3.15 (c) and (d) show that it is optimal to idle more of the flexible workshop when $\tau(i) = 2$. This is because with the sequence "FFCCCFFF", the final drydock needs three curved blocks in a row and

the optimal control policy will then keep the flexible workshop available to process curved blocks.



(a) $\tau(1) = 1$

(b) $\tau(2) = 1$

(c) $\tau(3) = 2$

(d) $\tau(4) = 2$

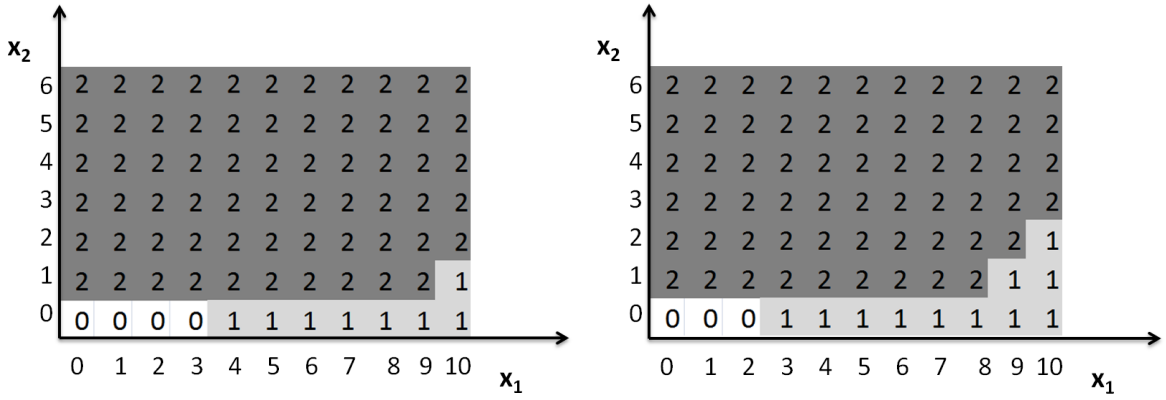
Figure 3.15: Optimal Control Policy for Sequence "FFCCCFFF"

This test case demonstrates that for the same number of curved blocks and flat blocks in a sequence period, different sequences will result in a different optimal control policy. Therefore, it is essential to design and plan an efficient block erection sequence, which may also improve the system performance.

Impact of Change of Capacity

Another interesting relationship to investigate is how the change of capacity impacts the optimal control policy. Figure 3.16 (a) shows the same test case in Table 3.3 with $\tau(1) = 2$ and $\theta_C = 2$, while Figure 3.16 (b) shows the test case where the

capacity of the flexible workshop θ_C increased to 4.



(a) Capacity of Flexible Workshop $\theta_C = 2$

(b) Capacity of Flexible Workshop $\theta_C = 4$

Figure 3.16: Optimal Control Policy: Impact of Change of Capacity

Figure 3.16 (b) shows that when the flexible workshop has greater capacity, more flat blocks should be assigned to the flexible workshop to maximize the system profit. This also allows for less idling of the flexible workshop in Figure 3.16 (b), and rationalizes why, with more capacity, the flexible workshop can process flat blocks with a sufficient speed without delaying the curved blocks. The average profit per unit time is 1.997 when $\theta_C = 2$, and it is 2.045 when $\theta_C = 4$, which is larger than the case where the capacity of flexibility $\theta_C = 2$. These results show that, with greater capacity of the flexible workshop, it can assist to produce more flat blocks and also increase the average profit in most cases. More test cases of how change of capacity impacts on the system profit will be presented in the next section.

3.3.3 Capacity Planning of Flexibility

This section focuses on how to invest and plan for the capacity of flexibility to achieve the maximum benefit from the flexibility resource. The previous research of flexibility has been focusing on controlling the flexible workshop at the execution level. It is a new approach to investigate the methodology of capacity planning of

flexibility at the strategic level. Additionally, it can also provide valuable information for shipyards and other manufacturing and service industries who are interested in investing on the flexibility.

Research related to the flexibility capacity planning at the strategic level mostly lies in the structural design of flexibility. [16] provided a comprehensive survey on the agile workforce and designed a performance matrix to evaluate the flexibility at the strategic level. They also developed a framework to guide the selection of the flexible architecture and worker coordination policy. [19] focused on a strategic-level issues of developing a method to quantify the ability of a flexible system to respond to the variability, only based on the structure of the flexibility.

The particular problem of capacity planning for flexibility in our model is how to design the capacity for the flexible resource in the "N" structure queueing network under the CONWIP release policy. Specifically, θ_F is the capacity of the flat block workshop, and θ_C is the capacity of the flexible workshop. After setting the capacity of θ_F, the optimal capacity of flexible workshop θ_C^* must then be determined.

To quantify the benefit of flexibility, it is necessary to develop the MDP model for the inflexible system as a benchmark. Therefore, a MDP model of the inflexible system is presented in the next section.

Inflexible System

The MDP formulation for the inflexible system is much simpler than the flexible system, since there is no control needed for the curved block workshop. The system state is $\{x_1, x_2, s_3\}$. $u_1 = \mathbb{1}\{x_1 >= 1\}$; $u_2 = \mathbb{1}\{x_2 >= 1\}$; $u_3 = \mathbb{1}\{\tau(s_3) = 1 \cap x_3 > 0\}$; $u_4 = \mathbb{1}\{\tau(s_3) = 2 \cap x_4 > 0\}$. Let Λ denote the uniformization factor and $\Lambda = \theta_F\mu_{11} + \theta_C\mu_{22} + \mu_3$. The recursive value function is as follows:

$$
\begin{aligned}
(3.2) \qquad J_{k+1}(x_1, x_2, s_3) = \frac{1}{\Lambda}\big[&(u_3+u_4)\mu_3 \cdot \gamma - P_F\theta_F - P_C\theta_C \\
&+ \beta\big\{u_1\theta_F\mu_{11}J_k(x_1-1, x_2, s_3) \\
&+ u_2\theta_C\mu_{22}J_k(x_1, x_2-1, s_3) \\
&+ u_3\mu_3 J_k(x_1+1, x_2, s_3 \oplus 1) + u_4\mu_3 J_k(x_1, x_2+1, s_3 \oplus 1) \\
&+ (\Lambda - u_1\theta_F\mu_{11} - u_2\theta_C\mu_{22} - (u_3+u_4)\mu_3)J_k(x_1, x_2, s_3)\big\}\big].
\end{aligned}
$$

No Budget Limit

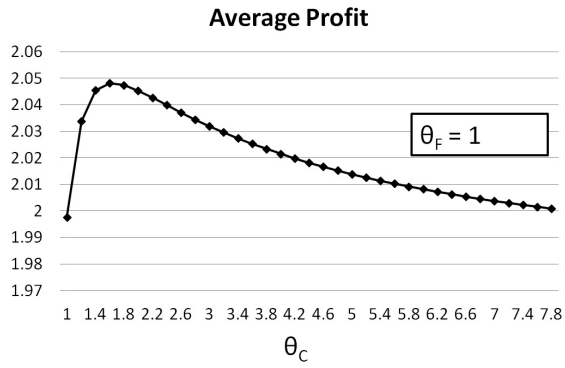
It is a challenging problem to determine the optimal capacity investment of flexibility θ_C^* given the capacity of θ_F, and quantify the benefit of flexibility analytically. Therefore, some numerical examples are presented here to reference. The same test case for the optimal policy structure is used for the capacity-planning problem in Table 3.4. Based on this test case, the MDP model is used to search for the θ_C^* from 1 to 8, given that θ_F is equals to 1 and 2.

Notation	Test Value
K_F	10
K_C	6
L	8
$\tau(i), i \in 1, ..., L$	CFFCFFFC
μ_{11}	3
μ_{21}	2
μ_{22}	2
μ_3	10
γ	5
P_F	1
P_C	1.5

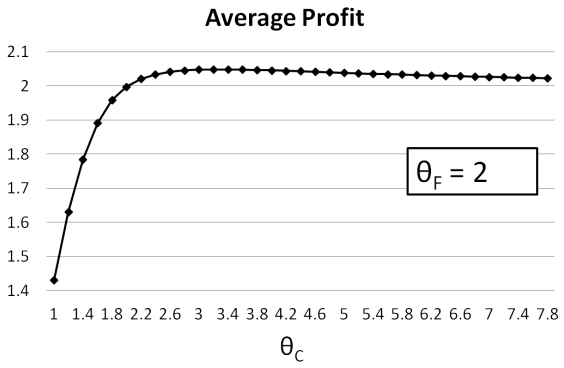
Table 3.4: Flexibility Capacity Planning Test Case

Figure 3.17 presents the average profit of the flexible system with change of capacity θ_C. The y-axis is the average profit. The x-axis is the value of θ_C, from $\{1, 1.2, 1.4, ..., 8\}$. Figure 3.17 (a) shows that when $\theta_F = 1$, the optimal capacity of

flexibility $\theta_C^* \approx 1.6$. It shows a fast profit growth from $\theta_C = 1$ to 1.6, then decreases slowly. Figure 3.17 (b) illustrates that when $\theta_F = 2$, the optimal capacity of flexibility $\theta_C^* \approx 3.2$. It shows a fast profit growth from $\theta_C = 1$ to 3.2, then maintain a high profit value.



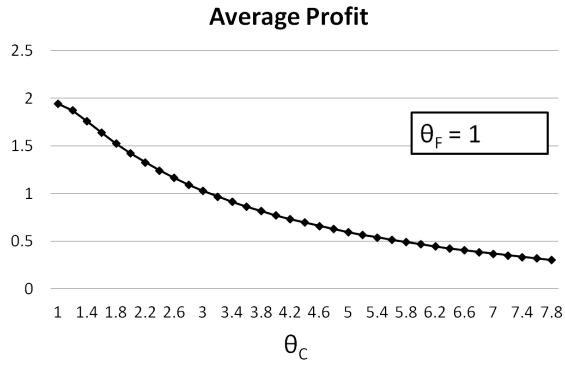
(a) Capacity of Flat Block Workshop $\theta_F = 1$



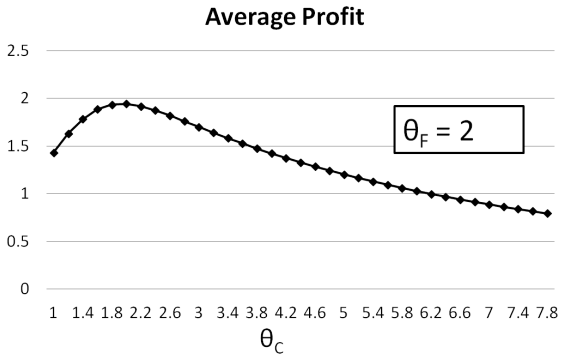
(b) Capacity of Flat Block Workshop $\theta_F = 2$

Figure 3.17: Flexible System: Average Profit with Change of θ_C

Figure 3.17 (a) and (b) demonstrate there exist the optimal capacity of flexibility θ_C^*, but the profit function may not be concave or convex in θ_C. Figure 3.17 (b) shows that when the capacity of the flat block workshop θ_F is large, a much larger capacity of the flexible resource can be useful.



(a) Capacity of Flat Block Workshop $\theta_F = 1$



(b) Capacity of Flat Block Workshop $\theta_F = 2$

Figure 3.18: Inflexible System: Average Profit with Change of θ_C

Figure 3.18 presents the average profit of the inflexible system with change of

capacity θ_C. Figure 3.18 (a) shows that when $\theta_F = 1$, $\theta_C < 1$ is optimal, and the average profit of the inflexible system decreases as the capacity of the curved block workshop increases. Figure 3.18 (b) illustrates that when $\theta_F = 2$, the optimal capacity for the curved block workshop is also 2. These results indicate that when there is no flexibility in the system, the capacity for each workshop should be balanced based on the workload for each workshop.

Next, the performance of the flexible system and inflexible system will be compared. Let p^f denote the average profit of flexible system, and p^i denote the average profit of the inflexible system. The percentage improvement by flexibility G is defined as follows:

$$G = \frac{p^f - p^i}{p^i} \cdot 100\%. \tag{3.3}$$

Figure 3.19 illustrates the the percent improvement as a result of flexibility when $\theta_F = 2$. The percent improvement increases as the flexible capacity increases, since after the point of $\theta_F = 2$, the profit of the inflexible system decreases as the flexibility capacity increases.

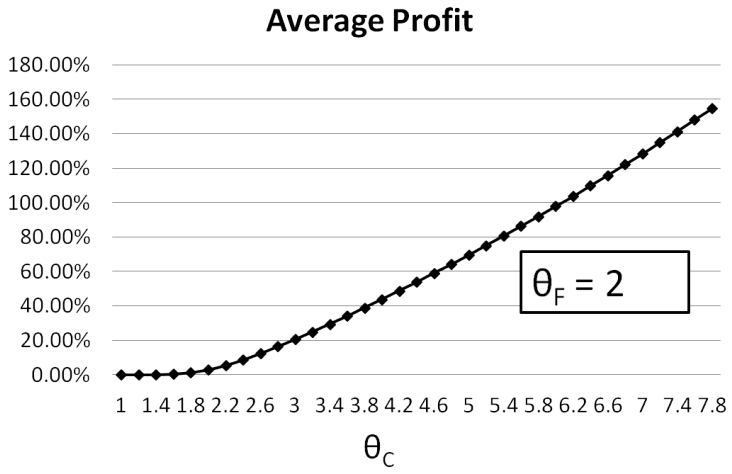


Figure 3.19: Percentage Improvement by Flexibility

The numerical study above shows that there exists the optimal flexible capacity θ_C^* given value of θ_F, and that the profit of the system remains stable with a reasonable expansion of the flexible resource with a large inflexible resource capacity θ_F.

For the future research, it is useful to seek the optimal capacity θ_F^* and θ_C^* when there is a limited budget, and study how the change of block sequence, CONWIP levels K_F and K_C impacts the capacity design.

3.4 Conclusion and Future Work

In this chapter, a Constant Work In Process (CONWIP) release policy is introduced to the strategic level of the flexible block assembly process. The "N" structure queueing networks for the block assembly process remains in this CONWIP model. The final drydock workshop is required to follow a one-dimensional block sequence, which makes the model very difficult to analyze. Two research problems are targeted in the model: (1) how to control the flexible workshop to maximize the system throughput under CONWIP release policy; (2) how to plan the capacity of flexible resource for the parallel partial flexible workshop.

The discrete-event simulation model was developed to provide insight of the complex CONWIP shipbuilding system. The simulation results show that the flexibility can significantly reduce the ship completion time and increase the drydock utilization, while reducing the block buffer size for the final erection process. For the control of the flexible workshop, the FCFS policy shows the best performance among all the test policies.

A two-loop CONWIP model formulated using Markov Decision Processes is developed to search for the structure of optimal policy in order to control the flexible workshop, and also provide the strategy for the flexibility capacity planning. Based

on the numerical study of the MDP model, the structure of the optimal control policy is a threshold type policy, which depends on the system dynamics, the CONWIP level, the block sequence, and the capacity of each workshop. The numerical study of the capacity-planning for the flexibility shows that (1) the flexibility can improve the system profit, and (2) the system remains high profit with extra capacity of the flexible resource.

There remains many unsolved problems for the future research with this CONWIP model. The numerical study of the optimal control policy shows a very complex structure of the optimal policy. We are aiming to develop a heuristic which can cooperate the block sequence constraint and also current system states. For the capacity-planning of the flexible resource, seeking the optimal capacity θ_F^* and θ_C^* for both dedicated workshop and flexible workshop is very useful for flexible production system design and investment. Another interesting research topic of the control of the flexible resource is to decompose the planning and scheduling so that local optimization control of flexibility is aligned with system performance. The simulation shows that the optimal control policy for the open "N" structure queueing network performs sub-optimal in the closed "N" structure queueing network. Developing a control policy which can achieve high performance in both execution level control (minimize the holding cost for jobs at current workshop) and the global system control (maximize the system throughput) will derive an easier implementation and simplify the execution level control.

CHAPTER IV

Two-Stage Queueing Network for Outfitting Planning and Control

4.1 Introduction

Chapter II and III focused on improving the efficiency and robustness of hull construction in shipbuilding. Outfitting, as another important processes in shipbuilding, represents as much as 50% of the cost of the ship and up to 50% of ship construction time in many instances [13]. The complexity of outfitting planning and the lack of intelligent control methodologies of outfitting processes currently in existence indicates that there are many opportunities to create new models for ship outfitting processes and, therefore, enhance overall shipbuilding performance.

4.1.1 Existing Outfitting Obstacles

Outfitting refers to the process of fabrication and installation of non-structural components. Outfitting planning is a very complex process, mainly because (1) outfitting processes occur at various stages in ship production; (2) there are dramatic time and cost differences of processing the same outfitting work at different stages; and (3) many aspects of an outfitting schedule must be integrated into the hull construction schedule. Many instances in shipyards show that the outfitting process delays the entire ship production system due to the disturbances by unexpected

delays, system variations, capacity limitations, and technological constraints. Contemporary outfitting planning relies heavily on previous experiences of shipbuilders. Our goal therefore is to develop an analytical model to support outfitting decision making and to provide a system methodology and control of outfitting planning. To develop a model that captures the main feature of the outfitting process and also can be used to analyze the system and achieve the effect planning and control methodology, we must first understand the outfitting process.

4.1.2 Outfitting Process

Ship production literature describes three stages within the ship outfitting process: on-unit, on-block, and on-board outfitting. On-unit refers to the assembly of an interim product consisting of only outfit materials, independent of hull structure. Examples include the pump room flat unit, fuel oil purifier unit, etc. On-block refers to the installation of outfit components or units on any structural subassembly or block prior to its erection on the ways. Outfitting on-block is more complex than on-unit, since it requires careful coordination between block assembly (steel work) and outfitting activities, and it may impact the duration of a block's occupation of an assembly area [13]. Therefore, outfitting can be delayed to next stage due to the tight schedule of block assembly. The third process, on-board, refers to the assembly of outfitting during hull erection and after launching. Working conditions in a hull are not ideal because of factors such as difficult access, limited space to work, and difficult work positions (e.g., overhead welding) [13]. It takes longer time and more cost to perform outfitting after hull construction.

On-unit outfitting frequently enhances safety and reduces labor, allowing more time and energy to be allocated to outfitting on-block and on-board. For the same amount of outfitting work, a task requiring one labor hour during the on-unit stage

will require approximately three hours at the on-block stage and eight hours at the on-board stage. The labor ratio of 1:3:8 may vary based on ship type and shipyard, but discrepancies in productivity remain. To minimize the task time and cost for outfitting work, it is essential that outfitting work be completed as early as possible. However, problems including system disruptions and variations, poor scheduling of hull construction and other outfitting equipment and materials delays, capacity limitations, technical infeasibility, and inefficiency in managing an effective system production rate, make it nearly impossible to develop an ideal outfitting plan in shipbuilding. Scheduling of the outfitting processes is therefore complex and a difficult topic for research.

4.1.3 State of The Art: Research on Ship Outfitting Planning

In the past, there has been little operations research in the area of ship outfitting. Some early production research on ship outfitting includes [18], [12], and [13]. [12] describes the planning and execution of pre-outfitting in structural assemblies. [13] clearly discusses and analyzes current outfitting problems, and uses a deterministic activity network model to formulate the outfitting planning problem. A mixed integer programming model of outfitting with sequence constraints was developed to address this outfitting planning problem. More recent research on the scheduling of the outfitting process has been conducted in [37], which provides a methodology that can automatically generate an outfitting sequence and planning.

All these studies only consider the deterministic processing time of the outfitting process and provide static outfitting planning. Approaches such as these will fail when dealing with common problems in shipbuilding, such as demand change during production and high variability in processing time. Therefore, we must investigate the outfitting problem with variability and develop a control policy that can

change dynamically according to the stage of the shipbuilding system. Our research on outfitting planning includes (1) an analytical approach at the strategic-planning level that provides outfitting work break-down strategy; and (2) a dynamic model approach at the tactical planning level to control the outfitting activities based on current system situation.

4.2 Strategic Level Model: An Open Queueing Network Model

The first problem we addressed is how to distribute the outfitting work to different stages. There are generally four stages of ship hull construction, as pictured below in Figure 4.1. Outfitting activities can be performed during or after any stages of the hull construction and it is more cost effective to process outfitting in the early stage.

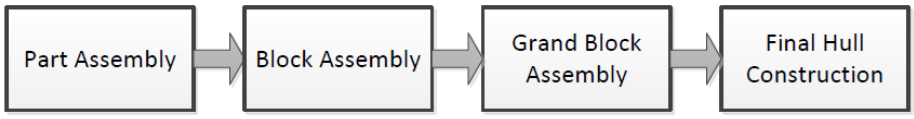


Figure 4.1: Hull Construction Processes

The number of stages in this model is simplified into two in our study. Specifically, stage 1 represents the general assembly process associated with block construction, while stage 2 represents the grand block construction. There are three tasks in this model: sub-assembly work (S), grand block assembly work (A), and outfitting work (O). Task S can only be processed at stage 1 and task A can be only processed at stage 2. Task O can either be processed at stage 1 or stage 2. Figure 4.2 illustrates the two stage outfitting model.

The decision of processing outfitting work at stage 1 or stage 2 is made by an independent and identically distributed (i.i.d.) sequence of Bernoulli (α) trials, in-

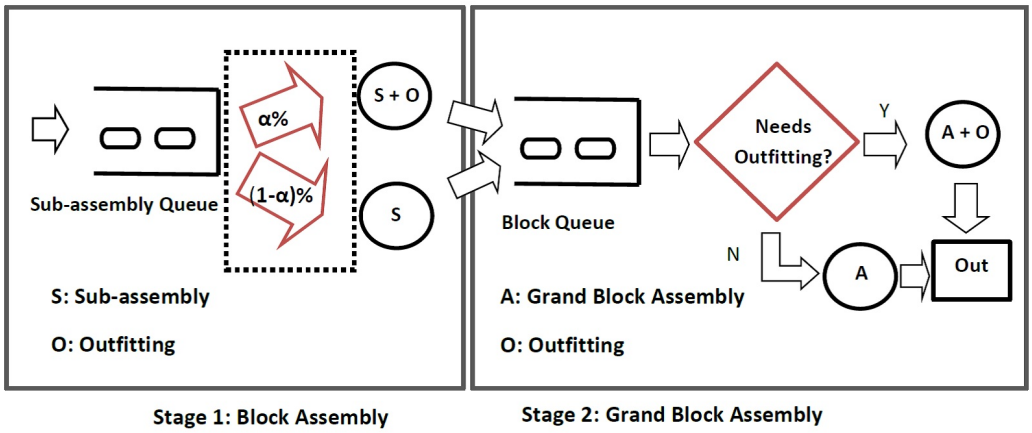


Figure 4.2: Strategic Two-stage Outfitting Model

dependent of all else, which randomly selects the task O to be performed at stage 1 or stage 2. The goal is to find the optimal value of α under any circumstances, which provides the control information needed to determine the percentage of outfitting workload distributed at each stage. Once the optimal value α is decided, α does not change over time during the process.

4.2.1 Model Formulation

The purpose of this outfitting model is to find the optimal α to minimize expected system cycle time. First, a mathematical model must be formulated in order to discover a closed-form equation of the expected system cycle time $CT(\alpha)$, the $CT(\alpha)$ equation can be used to calculate the the optimal α.

This model was devised as a queueing system to model the variability in outfitting processing time. Queueing systems are composed of an arrival process, a service/production process, and a queue (buffer). There is a single server queue in each stage. A G/G/1 queue is defined as a queueing system with a general distribution of arrival process, a general distribution of processing time, and one server. This model utilizes a general distribution for the arrival process and production processing time at stage 1 and stage 2, and assumes that the buffers before each stage

are infinite. Therefore, this model can be defined as a two-stage tandem G/G/1 queue. Processing rate and coefficient of variation (CV) are used to describe the general distributions. The CV is defined as the ratio of the standard deviation σ to the mean $1/\mu$. Table 4.1 defines each variables of the outfitting model:

α	Bernoulli (α), the percentage of outfitting done at stage 1
λ	Interarrival rate
μ_S	Processing rate of task S
μ_{O1}	Processing rate of task O at stage 1
μ_A	Processing rate of task A
μ_{O2}	Processing rate of task O at stage 2
C_{a1}	Coefficient of variation of arrival
C_S	Coefficient of variation of task S processing time
C_{O1}	Coefficient of variation of task O at stage 1 processing time
C_A	Coefficient of variation of task A processing time
C_{O2}	Coefficient of variation of task O at stage 2 processing time

Table 4.1: Notation for Outfitting Model

4.2.2 Analytical Results

Kingman's Approximation can be applied to formulate the closed-form solution of $CT(\alpha)$. For any G/G/1 queue, four values must be provided to find the average waiting time in the queue with Kingman's Approximation [21]: (1) c_a, CV of arrival process; (2) c_e, CV of processing time; (3) u, the server utilization; (4) t_e, the mean processing time. $CT_q(G/G/1)$ is the average waiting time of a G/G/1 queue and defined as follows:

$$CT_q(G/G/1) = (\frac{c_a^2 + c_e^2}{2})(\frac{u}{1-u})t_e \tag{4.1}$$

Since the model can be approximated as two tandem G/G/1 queues, Kingman's equation can be applied to estimate the system cycle time by providing values for $CT_q^1(\alpha)$ and $CT_q^2(\alpha)$. $CT(\alpha)$ is the expected total cycle time, which depends on the value of α. $CT_q^1(\alpha)$ is the expected waiting time in the queue at stage 1. $CT_q^2(\alpha)$ is the expected waiting time in the queue at stage 2. $CT_q^1(\alpha)$ and $CT_q^2(\alpha)$ are

approximated by Kingman's equation.

The expected system cycle time equals to the expected processing time plus the expected waiting time. First, the expected processing time at each stage must be calculated. $\frac{1}{\mu_1}$ is the expected processing time at stage 1, which is:

$$\frac{1}{\mu_1} = \frac{1}{\mu_S} + \frac{\alpha}{\mu_{O1}}, \tag{4.2}$$

$\frac{1}{\mu_2}$ is the expected processing time at stage 2, which is:

$$\frac{1}{\mu_2} = \frac{1}{\mu_A} + \frac{1-\alpha}{\mu_{O2}}, \tag{4.3}$$

Therefore, the expected system cycle time is as follows:

$$CT(\alpha) = CT_q^1(\alpha) + \frac{1}{\mu_1} + CT_q^2(\alpha) + \frac{1}{\mu_2}. \tag{4.4}$$

By Kingman's equation, the waiting time in the queue at stage 1 is:

$$CT_q^1(\alpha) = (\frac{c_{a1}^2 + c_{e1}^2}{2})(\frac{\lambda/\mu_1}{1-\lambda/\mu_1})\frac{1}{\mu_1}, \tag{4.5}$$

Using probability theory, the CV of the processing time in stage 1 is:

$$c_{e1}^2 = \frac{c_S^2\mu_{O1}^2 + \alpha c_{O1}^2\mu_S^2 + \alpha(1-\alpha)\mu_S^2}{(\mu_{O1} + \alpha\mu_S)^2}, \tag{4.6}$$

For stage 2, the arrival process in stage 2 is the departure process of stage 1. The CV of the arrival process in stage 2 is:

$$c_{a2}^2 = c_{d1}^2 = (\frac{\lambda}{\mu_1})^2 c_{e1}^2 + (1 - (\frac{\lambda}{\mu_1})^2)c_{a1}^2, \tag{4.7}$$

The CV of the processing time of stage 2 is:

$$c_{e2}^2 = \frac{c_A^2\mu_{O2}^2 + (1-\alpha)c_{O2}^2\mu_S^2 + \alpha(1-\alpha)\mu_A^2}{(\mu_{O2} + (1-\alpha)\mu_A)^2}, \tag{4.8}$$

By Kingman's equation, the waiting time in the queue at stage 2 is:

$$CT_q^2(\alpha) = (\frac{c_{a2}^2 + c_{e2}^2}{2})(\frac{\lambda/\mu_2}{1-\lambda/\mu_2})\frac{1}{\mu_2}. \tag{4.9}$$

Equation (4.4) is the closed-form solution for the expected system cycle time $CT(\alpha)$, which can be calculated using equation (4.2), (4.3), (4.5), (4.11), (4.10), (4.8),(4.9).

4.2.3 Numerical Study

For the numerical study, a series of test cases were designed and calculated to demonstrate how the closed-form equation can provide useful information concerning how to distribute outfitting work at each stage in shipbuilding. Let α^* denote the optimal α that minimizes the total expected system cycle time $CT(\alpha)$:

$$\alpha^* = \min_{\alpha \in [0,1]} CT(\alpha) \tag{4.10}$$

Three test-suites were developed in this section to evaluate the system. Test-suite 1 compares the deterministic system with the stochastic system, while test-suite 2 focuses on system performance at different processing times and fixed variations. Additionally, test-suite 3 analyzes how this variation impacts the system's performance.

Test-suite 1: Deterministic vs. Stochastic

Test-suite 1 consists of two cases, and was designed to test how varying processing time may impact overall system performance. The defined variables of test-suite 1 are provided in Table 4.2.

In case 1, $\mu_S = \mu_A = 5$, $\mu_{O1} > \mu_{O2}$, and all of the coefficient of variations are set equal to zero, which corresponds to no variation in the system, with processing times being deterministic. This is an ideal case where $\mu_{O1} > \mu_{O2}$, consequently allowing for the completion outfitting work at stage 1 to be more efficient. Therefore, α^* should be 1, which means processing all the outfitting at stage 1. This test case correlates

Cases	1	2
λ	1	1
μ_S	5	5
μ_{O1}	4	4
μ_A	5	5
μ_{O2}	3	3
C_{a1}	0	1
C_S	0	4
C_{O1}	0	2.5
C_A	0	4
C_{O2}	0	2.5

Table 4.2: Test-suite 1

with the "best" practice in modern shipbuilding; completing all outfitting as early as possible.

Figure 4.3 depicts two cases in Table 4.2, where $\alpha \in (0, 1)$ is on the x-axis while $CT(\alpha)$, the expected system cycle time, is on the y-axis. The results in this graph show that $\alpha^* = 1$, and the optimal expected system cycle time is $CT^*(\alpha) = 0.65$. One may also identify trends from this graph, such as how the expected system cycle time $CT(\alpha)$ decrease as α increases. In reality, the function $CT(\alpha)$ is expected to linearly decrease as α increases since the model is deterministic. However, this graph shows that a quadratic term exists in $CT(\alpha)$. The reason for this is that the coefficient of variation of processing time is not zero due to the Bernoulli trials. The CV of processing times and arrival times at stage 2 are not zero. These CV of processing times can be calculated by equation (4.11), (4.10), and (4.8) as follows:

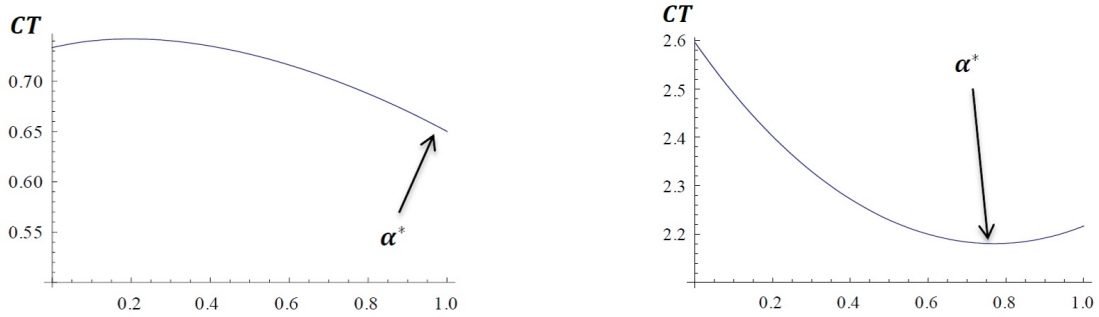
$$c_{e1}^2 = \frac{\alpha(1-\alpha)\mu_S^2}{(\mu_{O1} + \alpha\mu_S)^2}, \tag{4.11}$$

$$c_{a2}^2 = (\frac{\lambda}{\mu_1})^2 c_{e1}^2, \tag{4.12}$$

$$c_{e2}^2 = \frac{\alpha(1-\alpha)\mu_A^2}{(\mu_{O2} + (1-\alpha)\mu_A)^2}. \tag{4.13}$$

Table 4.2 shows that the coefficient of variations is not zero in case 2, correspond-

ing to a stochastic system. Fig.4.3 (b) shows the optimal percentage α^* is equal to 0.770, and the system expected cycle time is $CT^*(\alpha) = 2.18$. As compared to case 1, the value of α^* in case 2 decreased from 1 to 0.770, indicating that, although the processing time of outfitting at stage 2 is longer, it is still more favorable to process some outfitting work at stage 2, instead of processing all the outfitting work at stage 1. The system cycle time is also increased from 0.65 to 2.18 by just adding variation.



(a) Case 1: $\alpha^* = 1$

(b) Case 2: $\alpha^* = 0.770$

Figure 4.3: Test Suite 1

Test-suite 2: Change of Processing Rate

Test-suite 2 in Table 4.3 was designed to show how changing the processing rate impacts the optimal percentage value α^* and the system cycle time. For the cases where $\mu_S \geq \mu_A$ and $\mu_{O1} \geq \mu_{O2}$, the model showed that $\alpha^* \to 1$ for most cases. The cases presented here are of special interest because they diverge from this observed trend.

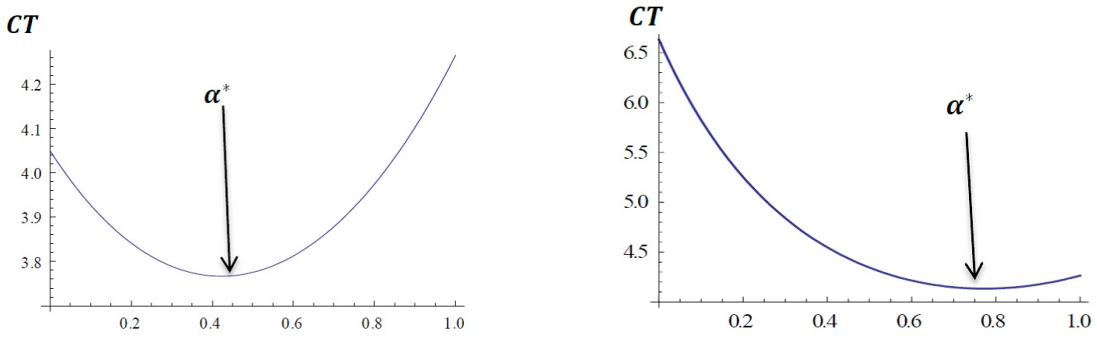
In case 3, μ_S is decreased from 5 to 3, and $\mu_S < \mu_A$, which means the sub-assembly work at stage 1 takes longer time than the block assembly at stage 2, and the outfitting work at stage 2 takes longer than at stage 1, i.e. $\mu_{O1} > \mu_{O2}$. Therefore, the value of α^* is not trivial and difficult to estimate.

Figure 4.4 case 3 shows that $\alpha^* = 0.423$ and $CT^*(\alpha) = 3.77$, indicating that

Cases	3	4
λ	1	1
μ_S	3	3
μ_{O1}	4	4
μ_A	5	5
μ_{O2}	3	2
C_{a1}	1	1
C_S	4	4
C_{O1}	2.5	2.5
C_A	4	4
C_{O2}	2.5	2.5

Table 4.3: Test-suite 2

when the sub-assembly work at stage 1 is not efficient, more outfitting work must be processed at stage 2. Case 4 is more similar to results actually found in the shipyard, where $\mu_{O2} = 2$ and $C_{O2} = 4$, which means the outfitting at stage 2 is very inefficient with high variation. Figure 4.4 case 4 shows that $\alpha^* = 0.772$ and $CT^*(\alpha) = 4.13$. It can be concluded that (1) when the processing rate of outfitting at stage 2 decreases, more outfitting at stage 1 should be processed; and (2) although the sub-assembly at stage 1 is not as efficient as grand block assembly at stage 2, more outfitting work should be processed at stage 1.



(a) Case 3: $\alpha^* = 0.423$

(b) Case 4: $\alpha^* = 0.772$

Figure 4.4: Test Suite 2

Test-suite 3: Change of Variation

Test-suite 3 in Table 4.4 and Table 4.5 was created to investigate how varying the processing time impacts the optimal percentage value α^* and the system cycle time. Table 4.4 shows the fixed system processing rates. $\mu_S = \mu_A = 5$, and $\mu_{O1} = 2\mu_{O2}$, i.e. the processing time of outfitting work at stage 1 is two times faster than the processing time of outfitting work at stage 2.

λ	1
μ_S	5
μ_{O1}	4
μ_A	5
μ_{O2}	2

Table 4.4: Test-suite 3: Processing Rate

Table 4.5 illustrates six test cases with different variation combinations:

Cases	5	6	7	8	9	10
C_{a1}	1	1	1	1	1	1
C_S	1	10	1	1	1	1
C_{O1}	1	1	1	3	1	10
C_A	1	1	10	1	1	1
C_{O2}	1	1	1	1	3	1

Table 4.5: Test-suite 3

Table4.6 summarizes the numerical results of the optimal percentage value α^* and system cycle time $CT^*(\alpha)$ for each test case.

Cases	5	6	7	8	9	10
α^*	1	0.670	1	0.589	1	0
$CT^*(\alpha)$	0.975	4.51	3.45	1.345	0.975	1.55

Table 4.6: Test-suite 3 Results

Case 5 is designed to have the equal CV for each processing time. This case also works as a benchmark for other test cases with different CV values. The numerical results show that the optimal percentage is $\alpha^* = 1$ and that the system expected cycle time is $CT^*(\alpha) = 0.975$. Because $\mu_{O1} = \mu_{O2}$, with the same CV for processing time, it is optimal to process all the outfitting work at stage 1.

In case 6, C_S is increased from 1 to 10, which means the variation of sub-assembly work at stage 1 is very large. The numerical results show that the optimal percentage is $\alpha^* = 0.67$ and the system expected cycle time is $CT^*(\alpha) = 4.51$. α^* decreases from 1 to 0.67 due to the increase of the CV of task S. It is more favorable to process some outfitting work at stage 2 when the variation of sub-assembly work at stage 1 is very large. This will provide more "slack time" for the workshop at stage 1 to work on the sub-assembly work. The system expected cycle time $CT^*(\alpha)$ increases from 0.975 to 4.51, demonstrating that, by only increasing the variation of processing time, one can increase the system cycle time significantly.

In case 7, C_A is increased from 1 to 10, therefore the variation of grand block assembly work at stage 2 is very large. Under these circumstances, the optimal percentage is $\alpha^* = 1$ and the system expected cycle time is $CT^*(\alpha) = 3.45$. α^* remains in 1 because both processing time and variation of processing time of processing outfitting work at stage 2 are larger than the processing time and variation of processing outfitting work at stage 1.

In case 8, C_{O1} is increased from 1 to 3, which means the variation of processing outfitting work at stage 1 is large. The numerical result shows that the optimal percentage here is $\alpha^* = 0.589$ and the system expected cycle time is $CT^*(\alpha) = 1.345$. α^* is sensitive with the change of C_{O1}. With larger variation of processing outfitting work at stage 1, it is optimal to process more outfitting work at stage 2.

In case 9, C_{O2} is increased from 1 to 3, which means the variation of processing outfitting work at stage 2 is very large. The numerical result shows that the optimal percentage is $\alpha^* = 1$ and the system expected cycle time is $CT^*(\alpha) = 0.975$. $\alpha^* = 1$ because the processing time and variation of outfitting at stage 2 is large. Therefore, it is optimal to never process outfitting at stage 2. $CT^*(\alpha) = 0.975$ is the same as

the $CT^*(\alpha)$ in case 5. This is because that there is no outfitting being processed at stage 2. Therefore, the value of C_{O2} is not involved in calculating $CT^*(\alpha)$.

Case 10 is an extreme case. $C_{O1} = 10$, i.e. the variation of processing outfitting work at stage 1 is very large. The optimal percentage is $\alpha^* = 0$ and the system expected cycle time is $CT^*(\alpha) = 1.55$. Figure 4.5 shows that the system expected cycle time $CT^*(\alpha)$ almost linearly increases with α. This extreme case shows that when the variation of the outfitting processing time at stage 1 is very large, the optimal control is achieved by never processing any outfitting at stage 1.

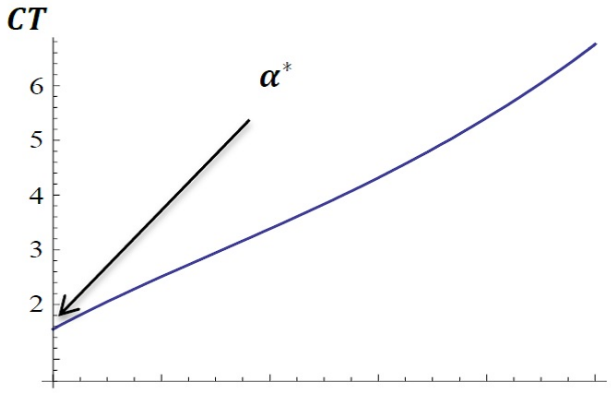


Figure 4.5: Case 10: $\alpha^* = 0$

Using the close-form equation, the optimal percentage of total outfitting work processed at each stage can be calculated under any scenario. Each test-suite allowed for definitive conclusions to be drawn. Test-suite 1 showed that the variation of processing time significantly impacts the system performance. It is specifically significant to note that, although the processing time of outfitting at stage 2 is greater than the processing time at stage 1, it is most favorable to distribute some outfitting work at stage 2, instead of processing all of the outfitting work at stage 1 with variation in the system. Test-suite 2 results indicated that it is optimal to process more outfitting at stage 2 when the sub-assembly work at stage 1 is slow. Test-suite

3 showed that when the variation of processing time at stage 1 is large, it is optimal to process more outfitting at stage 2. It also showed that $CT(\alpha)$ is more sensitive to changes of the CV values in outfitting time than the changes of the CV values in sub-assembly processing time at stage 1 and grand block assembly time at stage 2. This information can be very useful for the strategic level management, and to compare how close the current system control of outfitting is to the optimal planning results. This model also provides a quantitative understanding of the impact of variation in the ship production system.

Although this open queueing model provides the percentage of outfitting work that should be distributed at each stage, there is no control involved and it is still unknown when to do outfitting for each block at the execution level. In the next section, a dynamic closed queueing model formulated using a Markov Decision Process (MDP) will be presented. This MDP model provides more precise optimal control policies with system dynamics.

4.3 Execution Level Model: A Closed Queueing Network Model

A dynamic queueing model is developed in this section to search for an efficient control policy which can allocate the outfitting work according to the system states. Similar to the open queueing model described in the previous section, there are three tasks within the closed queueing network model: sub-assembly work (S), grand block assembly work (A), and outfitting work (O). Task S can only be processed at stage 1 and task A can be only processed at stage 2, while task O can either be processed at stage 1 or stage 2. The main difference between this dynamic model and the static model in the previous section is that there are infinite workshops at the second stage. There are two reasons it is necessary to allow for infinite workshops at the second

stage: (1) in the shipbuilding process, a structurally finished block will be moved to next stage immediately and human labor is quite flexible to complete work; and (2) from the modeling perspective, infinite workshops at the second stage prohibits block buffer queueing after the first stage. Therefore, the model does not need to record the block sequence released from stage 1. This will simplify the model significantly, especially for the MDP model (with memoryless property). Instead of Bernoulli trials, which was utilized in the static model, the action at stage 1 can be controlled in this model. Figure 4.6 illustrates

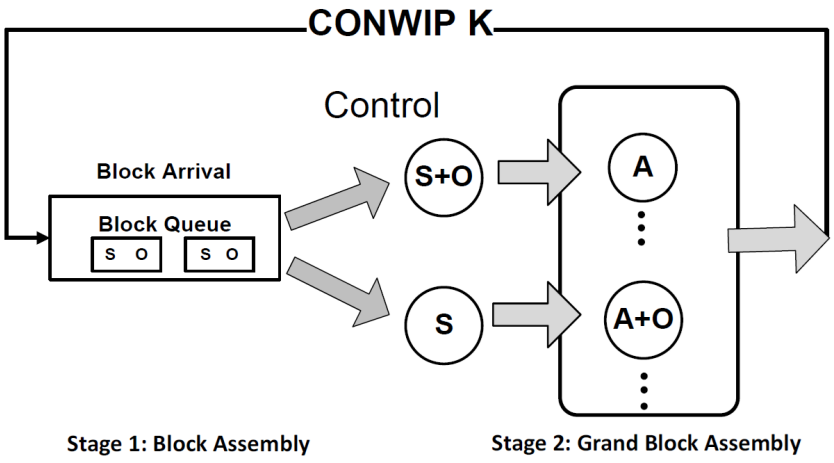


Figure 4.6: Two Stage Outfitting Model

The processing times for all the tasks are exponential distributed. The processing rates of task S and task A are μ_S and μ_A. The processing rate of sub-assembly work plus outfitting work at stage 1 is μ_{SO}, and the processing rate of grand block assembly work plus outfitting work at stage 2 is μ_{AO}. C_{SO} is the cost per unit time of the outfitting work being processed in the first stage, and C_{AO} is the cost per unit time of the outfitting work being processed in the second stage. γ is the reward rate of the finished block. The entire system is under CONWIP release policy with CONWIP level K. The objective is to effectively allocate the outfitting work to

minimize the long run average cost.

4.3.1 Literature Review

A similar model with three tasks and a two-stage queueing network under CONWIP job release policy has been studied in [9]. They consider a problem with optimal work sharing between two adjacent workers, each of whom processes a fixed task in addition to their shared tasks, and uses a MDP model that develops a threshold worksharing policy which can improve the system throughput and lower the WIP level.

Much work has been done in the field of CONWIP systems. In [25] they consider two station lines with exponential processing times and use Markov chain models to demonstrate the effectiveness of the Shortest Processing Time first (SPT) type of policies. In [10] they study a CONWIP system with two cross-trained workers and characterize an optimal worker-to-task assignment policy, which is a Fixed before shared principle prioritizing serving the fixed task. [27] introduces a new canonical model of zone-based cross-training workers and develops a methodology to employ it in U-shaped CONWIP lines. [3] consider a two-stage dynamic line balancing model under partial cross-training and define the trade-off between investing in Work-In-Process (WIP) and investing in workers.

The difference in our work is that there is infinite work capacity at second stage, and the outfitting work is agile to perform at either stage, but with different processing times and costs. The objective also considers the throughput reward. It is challenging to design an effective control policy to balance the production cost and the system throughput at same time. In the next section, a Markov Decision Process is developed to analyze the system and structure the optimal control policy.

4.3.2 Markov Decision Process Formulation

Defining the system state is the first step to formulate a MDP model. The state variables are $\{n_S, n_{SO}\}$. n_S is the number of blocks that only have steel work at the second stage. n_{SO} is the number of blocks that have both steel work and outfitting work at the second stage. Let Λ denote uniformization factor and $\Lambda = \mu_S + K\mu_A$. The objective is to minimize the long run average cost per stage. Let $J_{k+1}(n_S, n_{SO})$ denote the optimal k-stage cost to go function, and set the terminal cost function, $J_{k+1}(n_S, n_{SO}) = 0$.

(4.14)

$$
\begin{aligned}
J_{k+1}(n_S, n_{SO}) =& \frac{1}{\Lambda}[\mathbb{1}_{\{n_S \geq 1\}} n_S \mu_{AO}(-\gamma + J_k(n_S - 1, n_{SO})) \\
&+ \mathbb{1}_{\{n_{SO} \geq 1\}} n_{SO}\mu_A(-\gamma + J_k(n_S, n_{SO} - 1)) \\
&+ \mathbb{1}_{\{n_S + n_{SO} < K\}} \min\{\mu_{SO}[C_{SO} + J_k(n_S, n_{SO} + 1)] + (\mu_S - \mu_{SO})J_k(n_S, n_{SO})), \\
&\mu_S[C_{AO} + J_k(n_S + 1, n_{SO})]\} \\
&+ (\Lambda - \mathbb{1}_{\{n_S \geq 1\}} n_S \mu_{AO} - \mathbb{1}_{\{n_{SO} \geq 1\}} n_{SO}\mu_A - \mathbb{1}_{\{n_S + n_{SO} < K\}} \mu_S) J_k(n_S, n_{SO})].
\end{aligned}
$$

In the value iteration (4.14), the first term and second term denote the job throughput (or departure) from the second stage. Indicator function $\mathbb{1}\{n_S \geq 1\}$ provides the constraint that there is at least one block without outfitting at the second stage, and that the probability of finishing the block is $\frac{n_S \mu_{AO}}{\Lambda}$. The $\mathbb{1}\{n_{SO} \geq 1\}$ indicator function provides the constraint that there is at least one block with outfitting at the second stage, and that the probability of finishing the block is $\frac{n_{SO}\mu_A}{\Lambda}$. γ is the reward rate of the finished block. In this study, γ is negative since the objective is to minimize the cost.

The third term in this function is the decision term, which determines whether

to process outfitting at stage 1 or stage 2, to minimize the cost per stage. Indicator function $\mathbb{1}\{n_S + n_{SO} < K\}$ provides the constraint that there is at least one block at stage 1 ($n_S + n_{SO}$ indicate the total number of blocks at stage 2). The last term is the uniformization term.

4.3.3 Numerical Examples

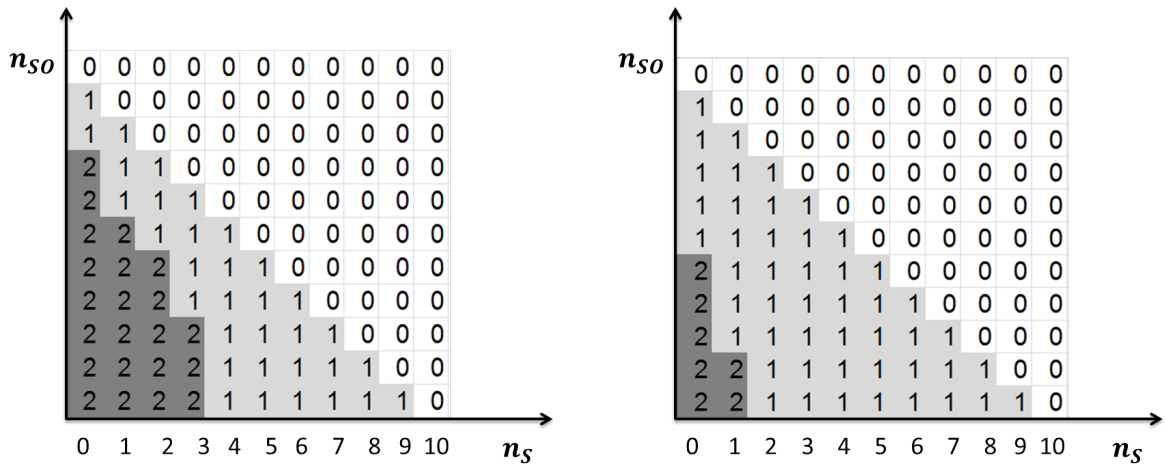
To gain insight into the structure of the optimal control policy, a set of illustrative numerical examples that is based on the MDP formulation is presented in Table 4.7. The CONWIP level K is 10 for all test cases.

Cases	1	2	3	4	5	6	7	8
K	10	10	10	10	10	10	10	10
C_{AO}	1	1	1	1	2	2	***2***	1
C_{SO}	1	1	1	1	2	2	1	***2***
γ	2	2	2	2	1	1	1	1
μ_S	5	5	5	5	5	5	5	5
μ_{SO}	2.5	***3.5***	2.5	2.5	3.75	3.75	3	3
μ_A	1	1	1	1	0.5	0.5	1	1
μ_{AO}	0.1	0.1	***0.01***	***0.75***	0.375	***0.35***	0.5	0.5

Table 4.7: Numerical Example Test Cases

Figure 4.7 illustrates the numerical results of test cases 1 and 2. The x-axis is n_S, the number of blocks without outfitting at second stage, and the y-axis is n_{SO}, the number of blocks with outfitting at the second stage. In this figure, "1" denotes that the optimal action at that particular state is processing outfitting at stage 1, while "2" denotes that the optimal action at that particular state is completing outfitting at stage 2. "0" wound denote the infeasible states since $n_S + n_{SO} < K$, and is therefore not found here.

Cases 1 through 4 are cases where $\gamma > min\{C_{SO}, C_{AO}\}$, when the system has a larger reward on throughput, and the optimal control policy aims to improve the system's efficiency. In case 1, $\mu_S = 5, \mu_{SO} = 2.5, \mu_A = 1$, and $\mu_{AO} = 0.1$, which satisfy the constraint $\frac{1}{\mu_{AO}} - \frac{1}{\mu_A} > \frac{1}{\mu_{SO}} - \frac{1}{\mu_S}$, i.e. the outfitting at the second stage



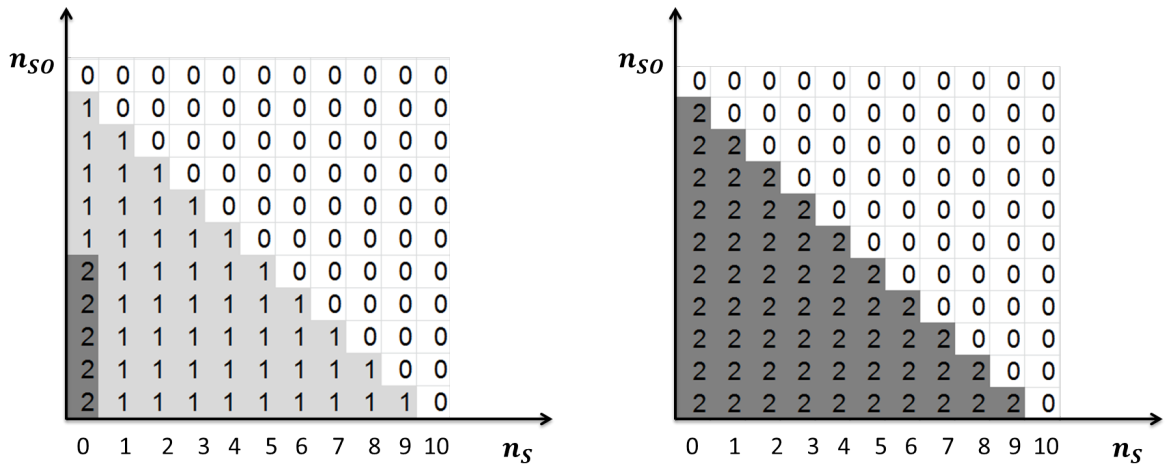
(a) Case 1

(b) Case 2

Figure 4.7: Outfitting MDP Numerical Example 1: $\gamma > min\{C_{SO}, C_{AO}\}$

takes more time. Figure 4.7 (a) shows that when n_S and n_{SO} are smaller than a certain threshold, outfitting should be completed at stage 2. In all other cases, it is more favorable to complete outfitting at stage 1. Logically, when n_S and n_{SO} are small, there are not many blocks at the second stage, and it is not efficient to idle the workshops (or workers) at second stage. Therefore, it is optimal to process outfitting at the second stage when when n_S and n_{SO} are small. When there is a sufficient amount of jobs at the second stage, it is more favorable to process outfitting at first stage since the single workshop at first stage is more efficient with outfitting work. For case 2, μ_{SO} increases from 2.5 to 3.5, which indicates the process of outfitting at first stage is more efficient. Figure 4.7 (b) shows that the optimal control will assign more outfitting work to stage 1 and less outfitting work to stage 2.

In case 3, the processing rate of outfitting at second stage, μ_{AO}, is set to be decreased from 0.1 to 0.01. Figure 4.8 (a) shows the scenario where outfitting is only processed at the second stage, when $n_S = 0$ and n_{SO} is smaller than a threshold. Most outfitting work should be processed at stage 1. In case 4, μ_{AO} increases from 0.1



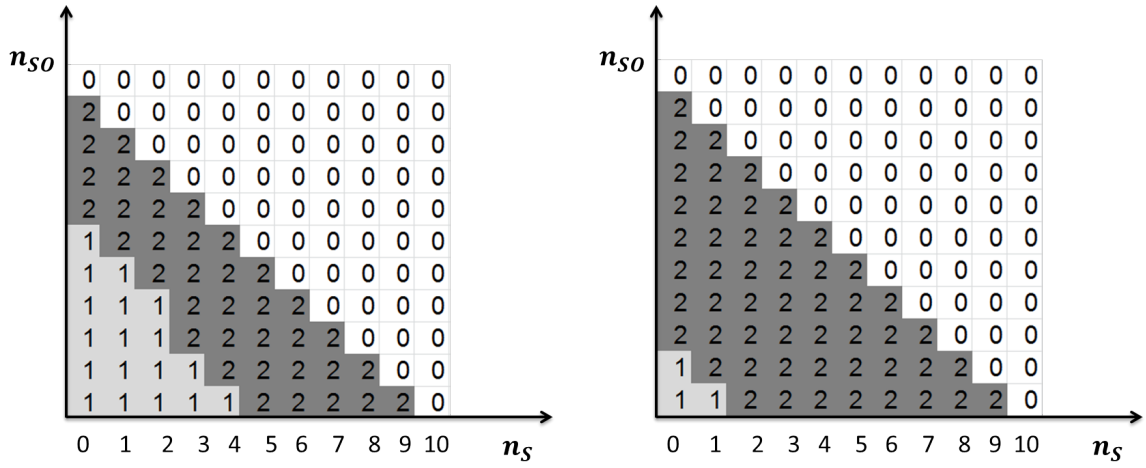
(a) Case 3

(b) Case 4

Figure 4.8: Outfitting MDP Numerical Example 2: $\gamma > min\{C_{SO}, C_{AO}\}$

to 0.75. Figure 4.8 (b) demonstrates a scenario where all outfitting processing occurs at stage 2. Although processing outfitting at the second stage is less efficient than processing outfitting at first stage, it is still optimal to complete all of the outfitting at the second stage because there are infinite workshops at the second stage and the capacity is much larger than stage 1. Therefore, to maximize the system throughput (since $\gamma > min\{C_{SO}, C_{AO}\}$), it is optimal to use all the capacity at second stage.

Case 5 and 6 are the cases where $\gamma < min\{C_{SO}, C_{AO}\}$, i.e. the throughput reward is smaller than the cost for each block. Therefore, the optimal control policy will focus on minimizing the cost instead of throughput. Case 5 is depicted in Figure 4.9 (a) and shows an "opposite" threshold control policy as compared to case 1: when n_S and n_{SO} are smaller than a threshold, outfitting should be processed at the first stage, and when n_S and n_{SO} are greater than a threshold, outfitting should be processed at the second stage. In case 6, the processing rate of outfitting at second stage, μ_{AO}, decreases from 0.375 to 0.35. Figure 4.9 (b) shows that there is more outfitting being processed at the second stage. This seems counterintuitive since processing outfitting

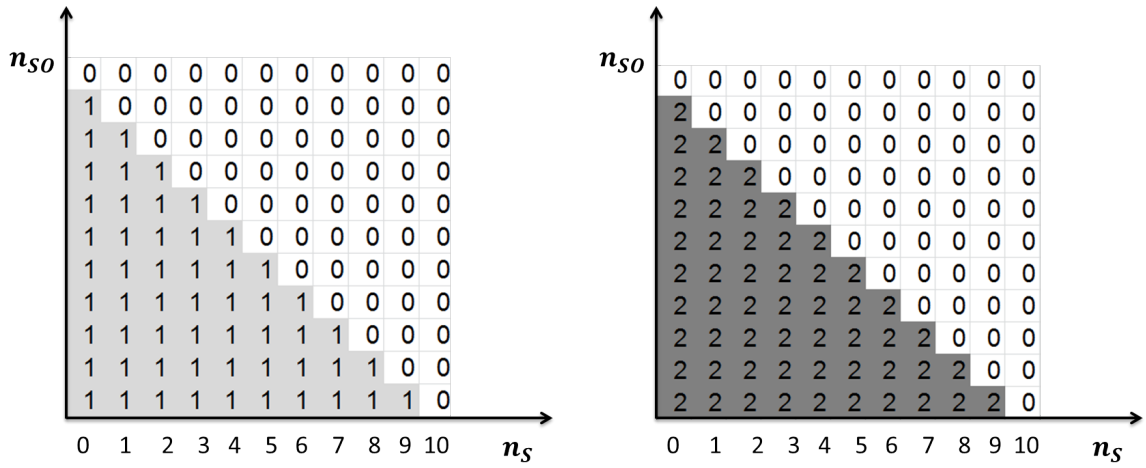


(a) Case 5

(b) Case 6

Figure 4.9: Outfitting MDP Numerical Example 3: $\gamma < min\{C_{SO}, C_{AO}\}$

at the second stage is less efficient than in case 5, there should be less outfitting work processed at the second stage. However, because $\gamma < min\{C_{SO}, C_{AO}\}$, the optimal control policy from MDP will slow down the system's throughput and minimize the cost from C_{SO} and C_{AO}.



(a) Case 7

(b) Case 8

Figure 4.10: Outfitting MDP Numerical Example 4: Change of C_{SO} and C_{AO}

Case 7 and 8 exemplify the impact of changing C_{SO} and C_{AO} on the optimal

control policy. In case 7, $C_{AO} = 2$, $C_{SO} = 1$, and $\gamma = 1$, which indicates that it costs more to process outfitting at the second stage. Figure 4.10 (a) shows that it is optimal to process all the outfitting work at stage 1. In case 8, $C_{AO} = 1$ and $C_{SO} = 2$, which indicates it costs more to process outfitting at first stage (which is in the actual shipyard, but this case is sued solely to show how cost can impact the optimal control policy). Figure 4.10 (b) shows that it is optimal to process all the outfitting work at stage 2. Case 7 and 8 show that C_{SO} and C_{AO} play a very important role in the control of allocating outfitting work to the most beneficial stages.

The numerical study shows that the optimal threshold policy is very complex. In the next section, we use the Mean Value Analysis to analyze the model and gain some insight for the optimal control policy structure.

4.4 A Static Model for Mean Value Analysis

CONWIP models can be analyzed by using a technique known as Mean Value Analysis (MVA). MVA calculates the mean queue size, mean waiting times, and throughput in a closed queueing network by utilizing a product-form solution. MVA is specifically used here in order to provide an approximate model for further understanding the system behave under different circumstances. One difference from the dynamic model is that instead of controlling the outfitting activities at the first stage, an independent and identically distributed (i.i.d.) sequence of Bernoulli(α) is used to randomly select the outfitting work to be processed at stage1 or stage 2. This is because that applying Mean Value Analysis requires a clear defined static queueing structure and it is very difficult to solve the dynamic queueing model using MVA. Figure 4.11 illustrates this static queueing model using Bernoulli(α) to select

the outfitting work.

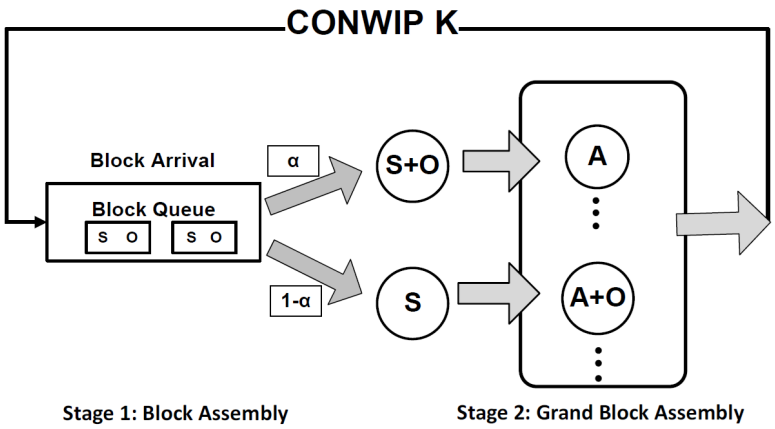


Figure 4.11: Mean Value Analysis Model

π_1 and π_2 are used to demonstrate the equilibrium distribution of stage 1 and stage 2. $\pi_1 = \pi_2 = 1$ since this is a tandem, two stage closed queue that requires all jobs going through stage 1 to also go through stage 2. μ_1 and μ_2 are the average processing rates at stage 1 and stage 2 when $K = 1$, where $\frac{1}{\mu_1} = \alpha\frac{1}{\mu_{SO}} + (1-\alpha)\frac{1}{\mu_{AO}}$ and $\frac{1}{\mu_2} = \alpha\frac{1}{\mu_A} + (1-\alpha)\frac{1}{\mu_{AO}}$. $W_i(n)$ is the average job waiting time at stage i with n jobs in the system. $L_i(n)$ is the average queue length at stage i with n jobs in the system. $\lambda_i(n)$ is the throughput rate at stage i with n jobs in the system.

4.4.1 Recursive Function and Iteration

When $K = 1$, the average waiting time at stage i is:

$$W_i(1) = \frac{1}{\mu_i}, i = 1, 2 \tag{4.15}$$

The average throughout rate at stage i is:

$$\lambda_i(1) = \frac{1}{\pi_1 W_1(n) + \pi_2 W_2(n)}, i = 1, 2 \tag{4.16}$$

Based on Little's Law, the average queue length is as follows:

$$L_i(1) = \lambda_i(1) W_i(1), i = 1, 2 \tag{4.17}$$

For $K = n$, the recursive equations are as follows:

$$W_1(n) = (L_1(n-1)+1)\frac{1}{\mu_1} \tag{4.18}$$

$$W_2(n) = \frac{1}{\mu_2} \tag{4.19}$$

$$\lambda_i(1) = \frac{n}{\pi_1 W_1(n) + \pi_2 W_2(n)}, i = 1, 2 \tag{4.20}$$

$$L_i(n) = \lambda_i(n) W_i(n), i = 1, 2 \tag{4.21}$$

The objective to minimize the long run average cost and the cost function is:

$$(\alpha C_{SO} + (1-\alpha) C_{AO} - \gamma)\lambda(\alpha, K) \tag{4.22}$$

The goal is to find the optimal α^* that minimizes the cost function. In the next section, some numerical examples will be presented to evaluate the performance of this MVA model.

4.4.2 Numerical Example

The test-suite detailed in Table 4.8 is designed to provide some comprehension behind how the change of CONWIP level K and various system dynamics may impact the optimal α^*.

K	3, 5, 10
C_{AO}	1
C_{SO}	1
γ	3
μ_S	5
μ_{SO}	3
μ_A	5
μ_{AO}	$2.5 \sim 0.1$

Table 4.8: MVA Test-Suite

Figure 4.12 represents the numerical example. The y axis in this graph is the value of α^*, the optimal fraction of outfitting work to be processed at stage 1. The

x axis is the value of μ_{AO}. μ_{AO} decreases from 2.5 to 0.1, indicating that it will take longer to process outfitting work at stage 2. When μ_{AO} is large, $\alpha^* = 0$, which shows that it is optimal to process all the outfitting work at stage 2. When μ_{AO} is small, $\alpha^* = 1$, which means it is optimal to process all the outfitting work at stage 1 when processing outfitting at stage 2 takes too long. α^* increases when μ_{AO} decreases indicating that when the process of outfitting work at stage 2 is becoming less efficient, more outfitting work should be processed at stage 1.

The difference lines show that when K increases, less outfitting work should be processed at stage 1 and more outfitting work will be processed at stage 2. It is because the capacity of stage 1 is one and the capacity of stage 2 is infinite. Therefore, when CONWIP level K increases, more outfitting work assigned to stage 2 can reduce the congestion at stage 1, while maintaining a high system throughput.

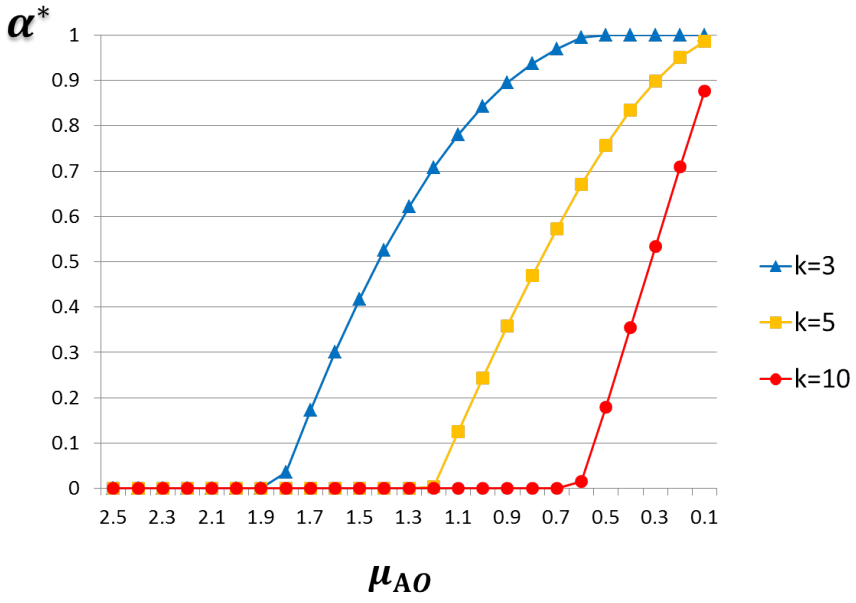


Figure 4.12: MVA Numerical Example

The numerical example shows a very reasonable behavior of the static model analyzed by MVA. The purpose of developing the MVA model is to use the information

provided by this MVA model and design an effective policy to control the dynamic model. Therefore, it is necessary to compare the performance between the MVA model and MDP model and analyze the difference system behavior.

4.4.3 MVA vs MDP

To compare the performance between the dynamic model and static model, the absolute percentage optimality gap is applied as the performance matrix. Let $Z(\alpha^*)$ denote the optimal cost of MVA, and λ^* denote the optimal cost from the MDP. The absolute percentage optimality gap $|G|$ is defined as follows:

$$|G| = |\frac{Z(\alpha^*) - \lambda^*}{\lambda^*}| \cdot 100\%. \tag{4.23}$$

We use the same test-suite as described in Table 4.8 to compare the optimal cost from both the MVA and MDP models. Figure 4.13 illustrates the optimal cost of MVA and MDP with the CONWIP level $K = 3$.

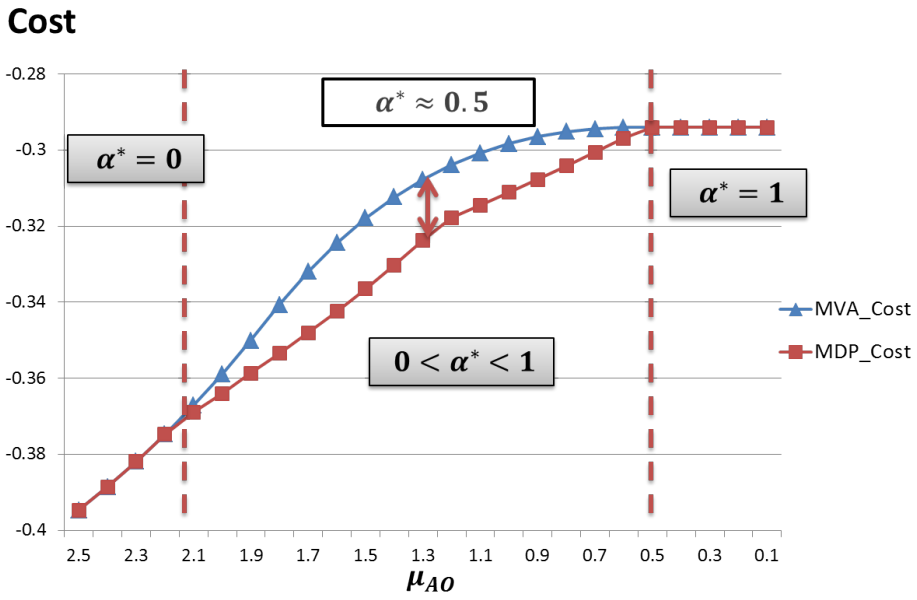


Figure 4.13: MVA vs MDP: K = 3

Figure 4.13 shows that the optimal cost from the MDP model is always smaller or equal to the optimal cost from the MVA model, which indicates the dynamic model

has better performance. The graph also illustrates that when $\alpha^* = 0$ or $\alpha^* = 1$, the absolute percentage optimality gap $|G|$ is equal or close to 0. $\alpha^* = 0$ indicates that it is optimal to process all outfitting work at stage 2, while $\alpha^* = 1$ indicates that it is optimal to process all outfitting work at stage 1. The optimal control policy from MDP model matches the output from MVA model, which shows that when $\alpha^* = 0$, the optimal policy is a strict priority policy that process all the outfitting work at stage 2, while when $\alpha^* = 1$, the optimal policy is to process all the outfitting work at stage 1. Therefore, when $\alpha^* = 0$ or $\alpha^* = 1$, the performance of MVA model (static model) and MDP model (dynamic model) are the same. When $0 < \alpha^* < 1$, MDP mode has a better performance than the MVA model since the optimal cost from MDP model is smaller. The largest absolute percentage optimality gap $|G|$ is when $\alpha^* \approx 0.5$, where $|G| = 5.53\%$. Therefore, when $0 < \alpha^* < 1$, the system is under the most complex circumstances and it is necessary to utilize the dynamic model to provide an optimal control policy based on the system dynamic.

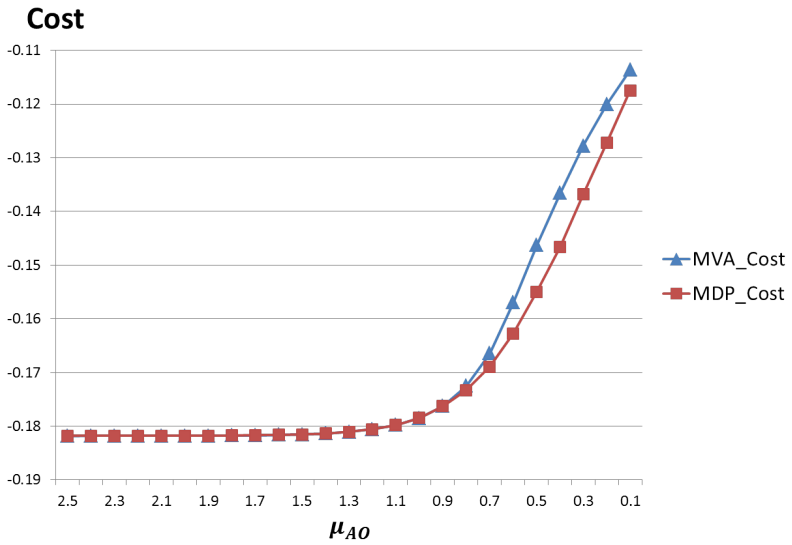


Figure 4.14: MVA vs MDP: K = 10

Figure 4.14 illustrates the cost of the MVA model and the MDP model when the

CONWIP level $K = 10$. The absolute percentage optimality gap $|G|$ decreases when the CONWIP level K increases. When there are more jobs in the system, it is less sensitive to the "mistake" made by the random decision from Bernoulli trial in the MVA model, since more jobs in the systems, less chance to idle the workshops at both stage 1 and stage 2. Therefore, the performance of the MVA model is approaching the efficiency of the MDP model when the number of jobs K increases in the system.

The numerical study above shows that the MVA model is a good approximation of the MDP model, and the information provided by the MVA model should be valuable for designing the heuristic for controlling the dynamic model. In the next section, a regression based threshold policy is designed using both information from the MVA model and the MDP model.

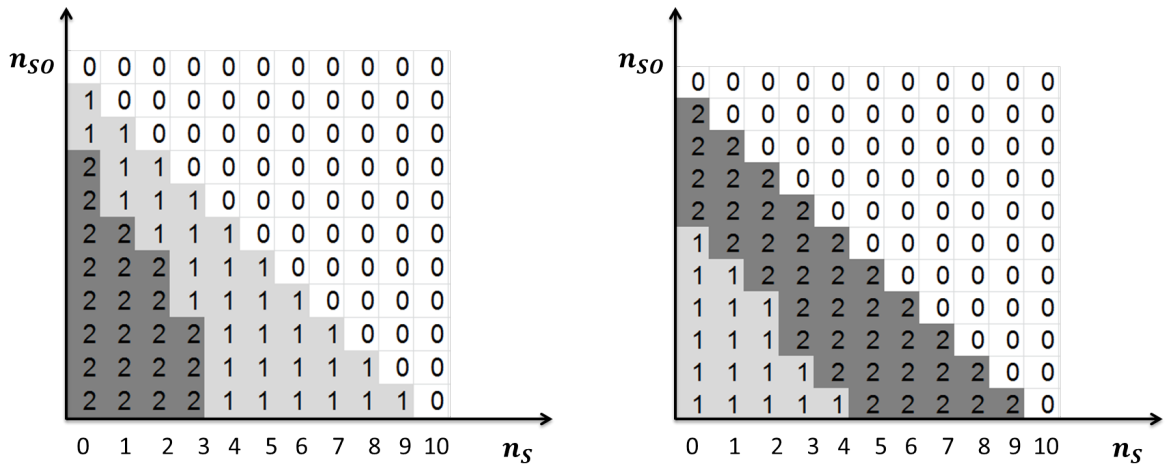
4.5 A Regression Based Threshold Policy

In this section, the goal is to develop a heuristic which is effective in controlling the outfitting activities and also simple to implement.

4.5.1 Conditions for Threshold Policy Type

Based on a large test-suite, the structure of the optimal control policy is a threshold type policy and there exist two types. Let $n_S^*(n_{SO})$ and $n_{SO}^*(n_S)$ denote the optimal threshold for n_S and n_{SO}. The type 1 threshold policy, shown in Figure 4.15 (a), illustrates that when $n_S \leq n_S^*(n_{SO})$ and $n_{SO} \leq n_{SO}^*(n_S)$, it is best to process outfitting at stage 2; otherwise, process outfitting at stage 1. The type 2 threshold policy in Figure 4.15 (b) indicates that when $n_S \leq n_S^*(n_{SO})$ and $n_{SO} \leq n_{SO}^*(n_S)$, outfitting should be completed at stage 1; otherwise, process outfitting at stage 2.

The type 1 and type 2 threshold policies shows a completely opposite policy structures, and the question remains discovering what the ideal conditions for each type



(a) Type 1 Threshold Policy

(b) Type 2 Threshold Policy

Figure 4.15: Threshold Policy Types

of policy are. Based on the results of a large test-suite which includes more than 10,000 cases, the conditions for type 1 and type 2 policies are as follows:

1. When $\gamma = min\{C_{SO}, C_{AO}\}$, the optimal cost of the MDP model $\lambda^* \to 0$ and the optimal cost of MVA model $Z(\alpha^*) = 0$ for most cases as well. The optimal control policies are strict priority policies that process outfitting at stage 1 when $C_{AO} \geq C_{SO}$, and complete outfitting at stage 2 otherwise.

2. When $\gamma > min\{C_{SO}, C_{AO}\}$, the optimal cost of the MDP model $\lambda^* < 0$. The optimal control policies are type 1 threshold policy (including the strict priority policies).

3. When $\gamma < min\{C_{SO}, C_{AO}\}$, the optimal cost of teh MDP model $\lambda^* > 0$. The optimal control policies are type 2 threshold policy (including the strict priority policies).

The condition $\gamma > min\{C_{SO}, C_{AO}\}$ indicates that the reward for each block is larger than the cost for each block. This condition is reasonable, since the production line should be profitable. Therefore, the focus is to deign a threshold policy under

the condition $\gamma > min\{C_{SO}, C_{AO}\}$, which is the condition for the type 1 threshold policy.

4.5.2 The Regression Model

Instead of having $n_S^*(n_{SO})$ and $n_{SO}^*(n_S)$ for each n_S and n_{SO}, a simplified heuristic is designed to capture the main features of the optimal threshold policy. Figure 4.16 shows the structure of this heuristic. This threshold policy has two variables n_S^* and n_{SO}^*, which are the intercepts for the threshold line. When n_S is smaller than the threshold line, outfitting is processed at stage 2; otherwise, outfitting is processed at stage 1. This is an approximation of the optimal threshold policy, and shows that the performance of this heuristic is very close to optimal. The optimal policy, based on a test-suite with approximately 2,000 cases, is only 0.03% better than this heuristic's performance. However, it is challenging to search for the n_S^* and n_{SO}^*.

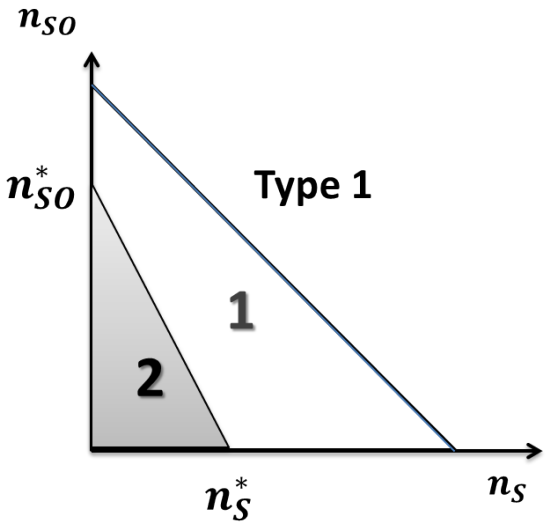


Figure 4.16: Type 1 Threshold Policy Regression Model

A regression model is developed to approximate the threshold intercepts n_S^* and n_{SO}^*. Let $\tilde{n}_S^*$ and $\tilde{n}_{SO}^*$ denote the threshold intercepts approximated by the regression model. Figure 4.17 shows that the regression model uses the information from

both the MVA model and MDP model. There are four predictors from the MVA model: (1) α^*, the optimal percentage of outfitting work processed at stage 1; (2) $\bar{n}_S$, the average number of blocks without outfitting at stage 2; (3) $\bar{n}_{SO}$, the average number of blocks with outfitting at stage 2; and (4) λ, the system throughput of the static model. $K, C_{AO}, C_{SO}, \gamma, \mu_S, \mu_{SO}, \mu_A, \mu_{AO}$ are the system parameters from the MDP model. Some second-order terms of system dynamics are also included, such as $\ln(K), \sqrt{K}, C_{AO}\mu_S, C_{SO}\mu_{SO}, \gamma\mu_A, \gamma\mu_{AO}$. In total there are 16 predictors. Forward Stepwise Selection is applied to select the significant predictors amount the 16 predictors.

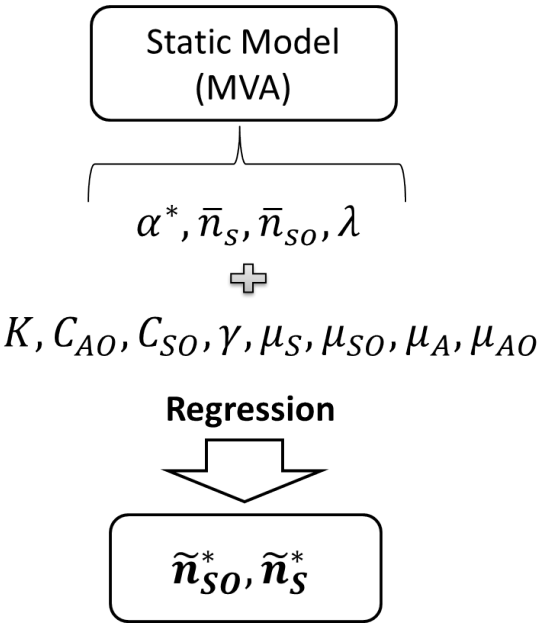


Figure 4.17: Regression Structure

Data Selection for the Regression Model

Discovering how to design a test-suite to collect data for the regression model is essential. In our test-suite, CONWIP level K is divided into small groups for each regression equations to make the regression model more accurate Table 4.10 represents the test-suite for the regression when $K \in 5, 6, 7, 8, 9, 10$. This test-suite

satisfies the constraints that $\mu_S > \mu_{SO}$ and $\mu_A > \mu_{AO}$. Only the test cases that satisfy the constraints were selected: (1) $\frac{1}{\mu_{AO}} - \frac{1}{\mu_A} > \frac{1}{\mu_{SO}} - \frac{1}{\mu_S}$, the outfitting at second stage takes longer time; (2) $\gamma > min\{C_{SO}, C_{AO}\}$, reward for each block is larger than the minimum cost. This test-suite provides 13,824 test cases.

K	5, 6, 7, 8, 9, 10
C_{SO}	5
C_{AO}	$(0.5, 1, 2, 4) \times C_{SO}$
γ	$(1, 2, 4) \times C_{SO}$
μ_S	1, 2, 5
μ_{SO}	$(0.75, 0.5, 0.1, 0.01) \times \mu_S$
μ_A	$(1.5, 1, 0.5, 0.1) \times \mu_S$
μ_{AO}	$(0.75, 0.5, 0.1, 0.01) \times \mu_A$

Table 4.9: Regression Test-Suite

Half of the test-suite is randomly selected from Table 4.10 and ran through both the MDP model and MVA model to collect the data for the regression. The other half of the test-suite is used to test the performance of the regression threshold policy. Forward Stepwise Selection selected the significant predictors and the regression equation of $\tilde{n}_S^*$ and $\tilde{n}_{SO}^*$ for $K \in 5, 6, 7, 8, 9, 10$ are as follows:

(4.24)

$$\begin{aligned}\tilde{n}_S^* =& 3.982 - 4.614\alpha^* + 0.6761K - 0.0829C_{AO} + 0.1627\bar{n_S} + 0.00308\gamma\mu_A - 0.313\mu_{SO} \\ &+ 0.1579\bar{n_{SO}} + 0.126\mu_S + 0.015\gamma - 1.69\ln(K) + 0.062\lambda \\ \tilde{n}_{SO}^* =& 2.201 - 4.246\alpha^* + 0.5103K - 0.1421C_{AO} + 0.0418\gamma + 0.01404\gamma\mu_A - 0.049\gamma\mu_{AO} \\ &- 0.309\mu_{SO} + 0.212\bar{n_{SO}} + 0.133\bar{n_S} + 0.422\mu_{AO} + 0.1\mu_A - 0.0022C_{AO}\mu_S\end{aligned}$$

Forward Stepwise Selection selected 11 predictors for the regression equation of $\tilde{n}_S^*$ with an R^2 value of 88.19%, while 12 predictors have been selected for the regression equation of $\tilde{n}_{SO}^*$ with an R^2 value of 81.05%. The most significant factor for both equations is α^*. This is an interesting result, since α^* is calculated by the static model

using Mean Value Analysis; however, it is able to capture the feature of the nonlinear queueing dynamics properly. If α^* is the only variable used as the predictor, the R^2 value is 76.98% for $\tilde{n}_S^*$ and 63.29% for $\tilde{n}_{SO}^*$. K is the second most significant factor in the regression equations.

It may be preferable in application to use some simpler regression equations with less predictors. If α^* and K are the only predictors in the regression, the R^2 value is 84.21% for $\tilde{n}_S^*$ and 71.10% for $\tilde{n}_{SO}^*$. The equations are as follows:

$$\begin{aligned} \tilde{n}_S^* &= 2.897 - 5.973\alpha^* + 0.4945K \\ \tilde{n}_{SO}^* &= 2.811 - 5.792\alpha^* + 0.5514K \end{aligned} \tag{4.25}$$

Equation (4.25) is much simpler than equations (4.24) in terms of number of predictors. The simpler regression equations provide alternatives when it is applied in the real world given different circumstances. In this thesis, equations (4.24) are applied to obtain the thresholds, since they have the most accurate approximations.

Performance of Regression Model

The heuristic is tested in the MDP model using the threshold level provided by equations (4.24). A test-suite in Table 4.10 is designed to test the performance of the regression threshold policy.

K	6
C_{AO}	5
C_{SO}	5
γ	10
μ_S	5
μ_{SO}	3
μ_A	5
μ_{AO}	$2.5 \sim 0.1$

Table 4.10: Regression Test-Suite

Figure 4.18 shows the optimal costs of the MVA model, optimal cost of the MDP model, and the cost of the MDP model under control of the regression threshold policy. The y axis in this figure is the cost and the x axis is the μ_{AO}. The performance of the MDP model under control of the regression threshold policy is always better or equal to the performance of the MVA model. When μ_{AO} is large, where $\alpha^* = 0$, the performance of the optimal MDP model, MVA model, and the dynamic model under control of regression threshold policy is the same (or very close). When μ_{AO} decreases, the cost difference between the model and MVA model increases. This is also the circumstances where the regression threshold policy performs better than the MVA model. The performance of the regression threshold policy is very close to the performance of the optimal control policy.

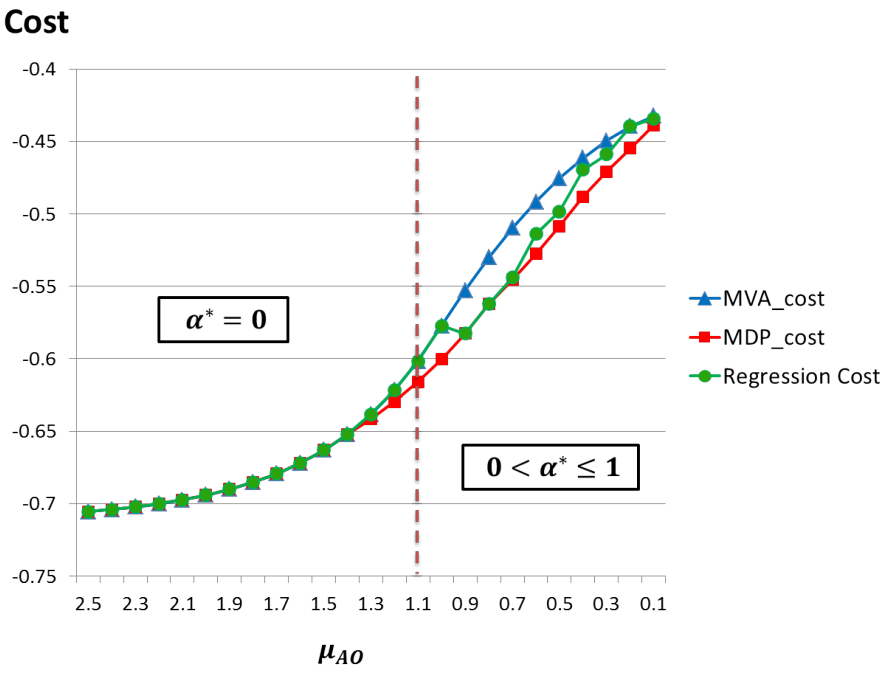


Figure 4.18: MVA, MDP, and Regression Comparison: K = 6

4.5.3 Design of the Heuristic

The previous test case shows that the performance of the MVA model matches the performance of the MDP model when $\alpha^* = 0$ or $\alpha^* = 1$ and the regression threshold policy greatly improves the performance of the MVA model when $0 < \alpha^* < 1$. Therefore, a heuristic is designed to combine both the MVA model and Regression Threshold policy as in Figure 4.19.

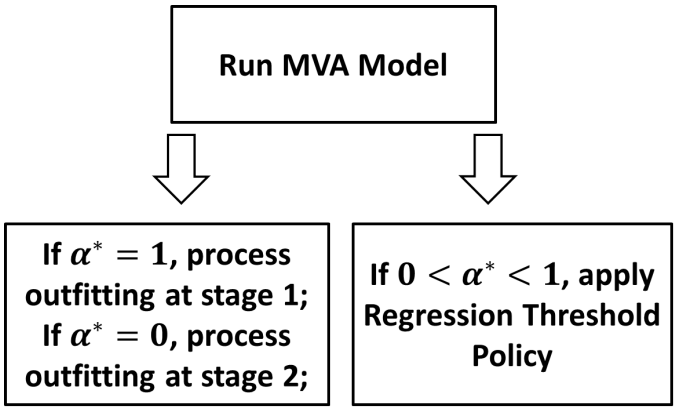


Figure 4.19: Heuristic Design

The algorithm of this heuristic is:

(1) Run the MVA model and calculate the value of α^*.

(2) If $\alpha^* = 1$, process all the outfitting at stage 1. If $\alpha^* = 0$, process all the outfitting at stage 2,

(3) If $0 < \alpha^* < 1$, apply Regression Threshold Policy.

The absolute percentage optimality gap is utilized as the performance matrix. Let Z^H denote the cost of the dynamic system under control of the heuristic, and λ^* denote the optimal cost from the MDP. The absolute percentage optimality gap $|G|$ is defined as follows:

$$|G| = |\frac{Z^H - \lambda^*}{\lambda^*}| \cdot 100\%. \tag{4.26}$$

The other half of the test-suite in Table 4.10 aims to test the performance of this heuristic. Table 4.11 presents the absolute percentage error of the performance of the Heuristic (a combination of the Regression Threshold Policy and Mean Value Analysis), the Regression Threshold Policy, and Mean Value Analysis. The numerical results show that this heuristic has the best performance with the mean of absolute percentage error 0.79%. The mean absolute percentage error is 4.6% when the system is only under control of Regression Threshold Policy. The mean of absolute percentage error is 1.2% from the Mean Value Analysis model. The Heuristic has the best performance and very close to optimal.

Policy	Mean	Std. Dev.	Max	Min
Heuristic	0.79%	2.59%	27.38%	0
Regression Threshold	4.6%	12.69%	198.9%	0
Mean Value Analysis	1.2%	3.31%	18.64%	0

Table 4.11: Heuristic, Regression Threshold, and Mean Value Analysis Comparison in Percentage Error

4.6 Conclusion and Future Work

The objective of this chapter is to improve the outfitting scheduling to make shipbuilding processes more efficient. The outfitting process is a very complex process to plan and control. Ideally, it is more cost effective and also time efficient to finish outfitting work in the early stages. However, due to the disturbances by unexpected delays and capacity limitations, outfitting work is frequently delayed. The problem is determining how much outfitting work can be delayed and how to distribute the outfitting work at each stage to keep the system productive.

At strategic level, a static open two-stage queueing model of outfitting is developed to provide information on how to optimally distribute the outfitting work at each stage in the shipbuilding process. This approach is particularly beneficial when there

is capacity misalignment which cannot be changed in the short term. A closed form equation for system cycle time is provided in this paper by applying Kingman's equation from the queueing theory. Using the closed form equation, the optimal percentage of total outfitting work processed at each stage can be calculated given any scenario.

Although the static open queueing model provides the percentage value of outfitting work that should be distributed at each stage, there is no control involved and it is still unknown when to complete outfitting for each block in the execution level. Therefore, at the execution level, a dynamic closed queueing model is formulated and analyzed using a Markov Decision Process (MDP) and Mean Value Analysis (MVA). To simplify the model, we assume the capacity at stage 2 is infinite. The control actions are to processing outfitting work at stage 1 or stage 2. There is cost associated with each decision as well. To include the throughput as part the objective as well, we introduced the reward for each throughput. The MDP formulation can provide the optimal cost and optimal control policy (given any state, the optimal action) numerically. The optimal control policy is a threshold type. However, due to the complexity of the MDP formulation, it is very hard to find any closed-form solution to calculate the threshold level dynamically.

To gain more insight into the dynamic system, Mean Value Analysis (MVA) is used to analyze the static version of the model. Instead of dynamic control, we use Bernoulli trials with parameter (α) again in this model as a approximation. This model maintains the same system dynamics and cost. The main objective is to find the optimal α^* that minimizes the average cost. Mean Value Analysis can provide the information of optimal α^*, and also other system features, including the average waiting time, the average queue length, and the throughput at each station. We use

the information provided by the static model and also the value of system parameters to search a regression based threshold policy. The Regression Threshold has better performance when $0 < \alpha^* < 1$, and the static model can provide more accurate conditions for strict priority rule (when $\alpha^* = 0$ or 1). Therefore, the heuristic is designed to combine both Regression Threshold Policy and Mean Value Analysis. The test shows that the performance of the heuristic is very close to the optimal.

CHAPTER V

Conclusions

The U.S. shipbuilding industry requires new production methodologies to reduce cost and, once again, become a leading competitor in the international shipbuilding industry. This research specifically introduces operational flexibility and Constant Work in Process (CONWIP) concepts to analyze the systems and optimize key performance measures such as cost, throughput, work in process, and cycle time. Additionally, efficient and robust dynamic control heuristics are designed to control the new flexible queueing network models in this research. These objectives are achieved through use of queueing theory, stochastic control, statistical analysis, and simulation. The models and major results in this research include:

1. An innovative flexible block assembly system that does not require expanding resource capacities. When examined under simulation, this model showed significant reduction in variation of block assembly time. In addition, an effective policy was developed to control flexible work station to minimize holding cost.

2. The development of a closed queueing network for the hull construction under control of CONWIP discipline and simulations showed 13% to 28% improvement on the system throughput by flexibility. CONWIP discipline enhanced the planning system with easier implementation and more robust control than current MRP sys-

tem. This model also provided a capacity planning strategy for the flexible resource in shipyard.

3. The design of a stochastic model for the scheduling and planning of the ship outfitting processes, which provided (1) the improved outfitting planning information at strategic level; (2) a dynamic control policy with quick response to system delays and variations.

The theoretical contributions of the work lie in designing and controlling the new flexible queueing network problems. Chapter II investigated the control of the "N" structure network with no preemption allowed. The "N" structure network is a simple model with partial flexibility, but dynamic control of "N" problem is difficult to obtain and still an unresolved area of research. The non-preemption assumption in this model causes it to be more complicated to control. In this research, we developed a threshold policy to control the non-preemptive "N" queueing network, and the numerical study shows the performance of this model is very close to optimal.

Chapter III introduces the CONWIP release policy to the "N" structure queueing network. Within this model, there is a job sequence constraint for the final workshop at the execution level which makes the control policy more complicated. However, the flexibility resource is more valuable and beneficial for the system with the job sequence constraint, since the flexibility can provide a more robust control which can allocate job dynamically according to the job sequence constraint. In Chapter III, the capacity-planning strategy of flexibility of "N" structure is also explored. An MDP model is formulated to analyze the optimal capacity plan, and the numerical results show that the flexible system has a greater average profit than the inflexible system, and the system with a larger capacity of flexibility maintains a high profit.

For future research, we plan to continue the flexibility capacity planning research.

The previous research has developed approaches that focus on local optimization of flexibility at execution level. However, it is also essential to identify ways to decompose the planning and scheduling so that local optimization is aligned with system performance for the control of flexibility at strategic level.

Another interesting research area is Global Shipbuilding Supply Chain Management (SCM). Shipbuilding is a very unique manufacturing industry and has tremendous opportunities to improve the system performance using SCM. We also can introduce flexible suppliers or outsourcing agents to improve the supply chain's stability, and to mitigate the risk of system disruptions due to the natural or various other uncontrollable disasters, in order to enhance the entire shipbuilding industry worldwide.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] H. Ahn, I. Duenyas, and R.Q. Zhang. Optimal control of a flexible server. *Advances in Applied Probability*, 36(1):139–170, 2004.

[2] S. Andradóttir, H. Ayhan, and D.G. Down. Compensating for failures with flexible servers. *Operations Research*, 55(4):753, 2007.

[3] Ronald G Askin and Jiaqiong Chen. Dynamic task assignment for throughput maximization with worksharing. *European Journal of Operational Research*, 168(3):853–869, 2006.

[4] S.L. Bell and R.J. Williams. Dynamic scheduling of a system with two parallel servers in heavy traffic with resource pooling: asymptotic optimality of a threshold policy. *The Annals of Applied Probability*, 11(3):608–649, 2001.

[5] D.P. Bertsekas. *Dynamic programming and optimal control*. Athena Scientific, 1996.

[6] C. Buyukkoc, P. Varaiya, and J. Walrand. The cμ rule revisited. *Advances in Applied Probability*, 17(1):237–238, 1985.

[7] D.G. Down and M.E. Lewis. The n-network model with upgrades. *Probability and the Engineering and Informational Sciences*, 24(2):171–200, 2010.

[8] D.A. Finke, C.B. Ligetti, M.T. Traband, and A. Roy. Shipyard space allocation and scheduling. *Journal of Ship Production*, 23(4):197–201, 2007.

[9] E.S. Gel, W.J. Hopp, and M.P. Van Oyen. Factors affecting opportunity of worksharing as a dynamic line balancing mechanism. *IIE Transactions*, 34(10):847–863, 2002.

[10] Esma S Gel, Wallace J Hopp, and Mark P Van Oyen. Hierarchical cross-training in work-in-process-constrained systems. *IIE Transactions*, 39(2):125–143, 2007.

[11] C Roger Glassey and Mauricio GC Resende. Closed-loop job release control for vlsi circuit manufacturing. *Semiconductor Manufacturing, IEEE Transactions on*, 1(1):36–46, 1988.

[12] R.A. Goldbach. Application of preoutfitting during construction of ammunition ships ae 32-35. *Marine Technology Society Journal*, 10(1), 1973.

[13] R.J. Graves and L.F. McGinnis. The outfit planning problem: Production planning in shipbuilding. *Naval Research Logistics Quarterly*, 29(2):357–384, 1982.

[14] J.M. Harrison. Heavy traffic analysis of a system with parallel servers: asymptotic optimality of discrete-review policies. *Annals of Applied Probability*, 8(3):822–848, 1998.

[15] T. Hirayama, M. Kijima, and S. Nishimura. Further results for dynamic scheduling of multiclass g/g/1 queues. *Journal of Applied Probability*, pages 595–603, 1989.

[16] Wallace J Hopp and MARK P OYEN. Agile workforce evaluation: a framework for cross-training and coordination. *Iie Transactions*, 36(10):919–940, 2004.

[17] W.J. Hopp and M.L. Spearman. *Factory physics*. McGraw-Hill Irwin Irwin, 2008.

[18] R. Hurst. Some production research activities on steelwork and outfitting. *The Society of Naval Architects and Marine Engineers*, 1968.

[19] Seyed M Iravani, Mark P Van Oyen, and Katharine T Sims. Structural flexibility: A new perspective on the design of manufacturing and service operations. *Management Science*, 51(2):151–166, 2005.

[20] W. J Kickert. The magic world of flexibility. *International Studies of Management and Organization*, 14(4):6–31, 1985.

[21] JFC Kingman. The single server queue in heavy traffic. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 57, pages 902–904. Cambridge Univ Press, 1961.

[22] L.L. Koste and M.K. Malhotra. A theoretical framework for analyzing the dimensions of manufacturing flexibility. *Journal of Operations Management*, 18(1):75–93, 1999.

[23] A.M. Law. Simulation modeling and analysis, 4th ed. 2007.

[24] J. Neter, W. Wasserman, and M.H. Kutner. *Applied linear regression models*, volume 3. Irwin Homewood, 1996.

[25] Jose Ostolaza, Joseph Thomas, and John O McClain. The use of dynamic (state-dependent) assembly-line balancing to improve throughput. *Journal of Manufacturing Operatoins Management*, 3:105–133, 1990.

[26] H. Parvin, H.S. Ahn, and M.P. Van Oyen. Flexible service scheduling under service level agreements. 2009. Working Paper, University of Michigan.

[27] Hoda Parvin, Mark P Van Oyen, Dimitrios G Pandelis, Damon P Williams, and Junghee Lee. Fixed task zone chaining: worker coordination and zone design for inexpensive cross-training in serial conwip lines. *IIE Transactions*, 44(10):894–914, 2012.

[28] M. Pinedo. Scheduling theory, algorithms, and systems,2nd ed. 2002.

[29] S. Saghafian, M.P. Van Oyen, and B. Kolfal. The w network and the dynamic control of unreliable flexible servers. 2011.

[30] A.K. Sethi and S.P. Sethi. Flexibility in manufacturing: a survey. *International journal of flexible manufacturing systems*, 2(4):289–328, 1990.

[31] R.L. Storch, C.P. Hammon, and H.M. Bunch. Ship production. 1988.

[32] T. Tezcan and JG Dai. Dynamic control of n-systems with many servers: Asymptotic optimality of a static priority policy in heavy traffic. *Oper. Res*, 2009.

[33] J. Van Meighem. Dynamic scheduling with Convex delay Costs: The Generalized cμ rule. *The Annals of Applied Probability*, 5(3):808–833, 1994.

[34] P. Varaiya, J. Walrand, and C. Buyukkoc. Extensions of the multiarmed bandit problem: the discounted case. *Automatic Control, IEEE Transactions on*, 30(5):426–439, 1985.

[35] Ranjan Varghese and Duck Young Yoon. Dynamic spatial block arrangement scheduling in shipbuilding industry using genetic algorithm. In *Industrial Informatics, 2005. INDIN'05. 2005 3rd IEEE International Conference on*, pages 444–449. IEEE, 2005.

[36] M.H. Veatch. A cμ rule for parallel servers with two-tiered cμ preferences, submitted for publication. 2008.

[37] Y. Wei, U. Nienhuis, and E. Moredo. Two approaches to scheduling outfitting processes in shipbuilding. *Journal of Ship Production and Design*, 26(1):20–28, 2010.

[38] C.J. Zimmerman and J Bissell. Global shipbuilding industrial base benchmarking study. *Office of the Deputy Under Secretary of Defense*, 2005.