The Modeling, Analysis and Control of Resilient Manufacturing Enterprises

Yao Hu
University of Kentucky, behrmanhuyao@hotmail.com

This Doctoral Dissertation is brought to you for free and open access by the Electrical and Computer Engineering at UKnowledge. It has been accepted for inclusion in Theses and Dissertations--Electrical and Computer Engineering by an authorized administrator of UKnowledge. For more information, please contact UKnowledge@lsv.uky.edu.
STUDENT AGREEMENT:

I represent that my thesis or dissertation and abstract are my original work. Proper attribution has been given to all outside sources. I understand that I am solely responsible for obtaining any needed copyright permissions. I have obtained and attached hereto needed written permission statements(s) from the owner(s) of each third-party copyrighted matter to be included in my work, allowing electronic distribution (if such use is not permitted by the fair use doctrine).

I hereby grant to The University of Kentucky and its agents the non-exclusive license to archive and make accessible my work in whole or in part in all forms of media, now or hereafter known. I agree that the document mentioned above may be made available immediately for worldwide access unless a preapproved embargo applies.

I retain all other ownership rights to the copyright of my work. I also retain the right to use in future works (such as articles or books) all or part of my work. I understand that I am free to register the copyright to my work.

REVIEW, APPROVAL AND ACCEPTANCE

The document mentioned above has been reviewed and accepted by the student’s advisor, on behalf of the advisory committee, and by the Director of Graduate Studies (DGS), on behalf of the program; we verify that this is the final, approved version of the student’s dissertation including all changes required by the advisory committee. The undersigned agree to abide by the statements above.

Yao Hu, Student

Dr. Lawrence E. Holloway, Major Professor

Dr. Zhi David Chen, Director of Graduate Studies
THE MODELING, ANALYSIS AND CONTROL OF RESILIENT MANUFACTURING ENTERPRISES

DISSERTATION

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the College of Engineering at the University of Kentucky

By
Yao Hu
Lexington, Kentucky

Director: Lawrence E. Holloway, Ph.D., Professor of Electrical and Computer Engineering
Lexington, Kentucky
Jingshan Li, Ph.D., Associate Professor of Industrial and Systems Engineering at University of Wisconsin-Madison
Madison, Wisconsin

2013
Copyright© Yao Hu 2013
ABSTRACT OF DISSERTATION

THE MODELING, ANALYSIS AND CONTROL OF RESILIENT MANUFACTURING ENTERPRISES

The resilience of manufacturing enterprises is an important research topic, since disruptions have severe effects on the normal operation of manufacturing enterprises, especially as manufacturing supply chains become global. Although many case studies have been carried out to address resilience in organizations, a systematic method to model and analyze the resilience dynamics in manufacturing enterprises is not well developed. This study is intended to conduct research on quantitative analysis and control for resilience.

After reviewing the literature addressing resilience, a modeling framework is presented to characterize the resilience of a manufacturing enterprise responding to disruptive events, which includes inventory flow between enterprise nodes, different costs, resource, demand, etc. Each node within the network is represented as a dynamic model with associated costs of production and inventory. This mathematical model is the foundation of quantitative analysis and control. With this model, an optimal control problem is formulated, by which the control can be solved to achieve minimum cost.

Several different types of systems are defined and analyzed in this work. We develop the approach of aggregation to simplify the network structures. The study is mainly focused on two categories of network systems: serial network systems and assembly tree network systems. The analysis on these two categories covers two conditions: in discrete time domain without considering capacities, and in continuous time domain with considering capacities. The methods to determining optimal operations are developed under different conditions. In the serial network systems analysis, a practical case study is introduced to show the corresponding method developed. Finally, the problems are discussed for future research.

Based on the results of these analyses, we present optimal control policies for resilience. Our method can support the analysis of the impact of disruptions, and the development of control strategies that reduce the impact of the disruption.

KEYWORDS: Resilience, production network, modeling, control

Author’s signature: Yao Hu

Date: March 6, 2013
The Modeling, Analysis and Control of Resilient Manufacturing Enterprises

By

Yao Hu

Lawrence E. Holloway, Ph.D.
Director of Dissertation

Zhi Chen, Ph.D.
Director of Graduate Studies

March 6, 2013
Date
ACKNOWLEDGEMENTS

First of all, I would like to express my sincere appreciation to my advisor, Dr. Lawrence E. Holloway for his profound knowledge, invaluable guidance and considerate supports. He has always been my mentor not only in research but also in helping me face all kinds of problems. Also, special thanks to Dr. Jingshan Li for his instructive supports and various help. He has been constantly providing all kinds of assistance without reservation. I have enjoyed working with these two great people during all these years.

I would also like to thank Dr. Bruce Walcott and Dr. Thomas Goldsby. Thanks for their instructive ideas and comments for my research.

Additionally, special thanks to Dr. YuMing Zhang for his supports and assistance during my whole process of pursuing my Ph.D. degree.

Also, I would like to thank my friends and colleagues, Dr. Yi Huang, Dr. Xiaoji Ma, Dr. Juwen Wang, and Dr. Yan Du. Thanks for their encouragements and making my life so happy in E.C.E. Department.

Last but not least, thanks to my parents for everything they give to me.
Contents

Acknowledgements iii

List of Tables ix

List of Figures x

1 Introduction 1

2 Literature Review 3

2.1 Case Studies on Resilience . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
2.2 Optimal Network for Resilience . . . . . . . . . . . . . . . . . . . . . . . . . 3
2.3 Resilient Infrastructure . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
2.4 Resilient Supply Chain . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
2.5 Modeling related to Resilience in Manufacturing and Supply Chain Systems 6
2.6 Resilience of Computer Networks . . . . . . . . . . . . . . . . . . . . . . . . 6
2.7 Resilience of Other Systems . . . . . . . . . . . . . . . . . . . . . . . . . . . 7
2.8 Conclusion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9

3 Modeling of Resilient Enterprises 10

3.1 System Description and Problem Formulation in Discrete Time Domain . . 10

3.1.1 Network System Model . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10
3.1.2 Nodes Description . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14
3.1.3 Problem Formulation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14
3.1.4 A Simple Example of Analysis in Discrete Time Domain . . . . . . . . . 18

3.2 System Description and Problem Formulation in Continuous Time Domain 22

3.2.1 Network System Model . . . . . . . . . . . . . . . . . . . . . . . . . . . . 22
<table>
<thead>
<tr>
<th>7 Optimal Control of Serial Network Systems in Continuous Time Domain with the Assumption of Downstream Buildup</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1 Serial Network Systems with Decreasing Storage</td>
</tr>
<tr>
<td>7.1.1 Operation During the Nominal-operation Period</td>
</tr>
<tr>
<td>7.1.2 Properties of Operation During the Disruption Period</td>
</tr>
<tr>
<td>7.1.3 Properties of Operation During the Pre-disruption Period</td>
</tr>
<tr>
<td>7.1.4 Optimal Operation over the Time Frame of Both Pre-disruption and Disruption Periods</td>
</tr>
<tr>
<td>7.2 Generalizing to General Serial Network Systems</td>
</tr>
<tr>
<td>7.2.1 Normalization</td>
</tr>
<tr>
<td>7.2.2 Locating the $s$-nodes and $c$-nodes</td>
</tr>
<tr>
<td>7.2.3 Aggregation</td>
</tr>
<tr>
<td>7.2.4 Main Steps to Obtain the Optimal Control for the Serial Network Systems</td>
</tr>
<tr>
<td>7.3 Case Study</td>
</tr>
<tr>
<td>7.3.1 Normalization</td>
</tr>
<tr>
<td>7.3.2 Locate $s$- and $c$- nodes</td>
</tr>
<tr>
<td>7.3.3 Aggregation</td>
</tr>
<tr>
<td>7.3.4 Quadratic Program</td>
</tr>
<tr>
<td>7.3.5 Optimal Operations</td>
</tr>
<tr>
<td>7.4 Conclusions</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>8 Optimal Control of Serial Network Systems in Continuous Time Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1 Aggregation</td>
</tr>
<tr>
<td>8.1.1 Normalization and Locating $s$-nodes and $c$-nodes</td>
</tr>
<tr>
<td>8.1.2 Two Special Nodes for Aggregation: $N_{c1}^d$ and $N_{c1}^u$</td>
</tr>
<tr>
<td>8.1.3 Aggregation</td>
</tr>
<tr>
<td>8.2 Optimization Problem</td>
</tr>
<tr>
<td>8.2.1 Operations and Cost in Nominal-operation Period and Disruption Period</td>
</tr>
<tr>
<td>8.2.2 Operations and Cost in Pre-disruption Period</td>
</tr>
<tr>
<td>8.2.3 Formulating the Optimization Problem</td>
</tr>
<tr>
<td>8.3 Case Study</td>
</tr>
</tbody>
</table>
8.3.1 Normalization
8.3.2 Locating s- and c- nodes
8.3.3 Determine \( \rho \) and \( \gamma \)
8.3.4 Generate \( S \) Matrices
8.3.5 Quadratic Program
8.3.6 Determine the Optimal Operation
8.4 Conclusion

9 Optimal Control of A Special Type of Assembly Tree Network Systems in Continuous Time Domain
9.1 The Specifications of the Assembly Tree Network Systems Consisting of One Disrupted Chain and One Branch
9.2 Operation under Assumption 13 or 14
9.3 Operation under Assumption 15
  9.3.1 Aggregation
  9.3.2 The Operations of the Aggregated System
  9.3.3 Optimization Problem
9.4 Conclusion

10 Conclusion
10.1 Conclusion of the Current Study
10.2 Future Directions

A Derivation of Solution to the Problem of Simple System in Chapter 3.1.4
A.1 Case 1
A.2 Case 4a
A.3 Case 2
A.4 Case 3a

B Proof of Associativity

C Proof of Aggregation
C.1 Proof of Theorem 5
C.2 Proof of Theorem 6

D Preliminary Work in Future Directions
List of Tables

7.1 The locations of s- and c-nodes ........................................ 83

8.1 S matrix, the types of triangles, and coefficients in $C_{j,k}$ ............ 108

9.1 S matrix, the types of triangles, and coefficients in $C_{br}^{\text{en}}_{j,a}$ .......... 136

9.2 T matrix, the types of triangles, and coefficients in $C_{c1}^{\text{cl}}_{j,a}$ .......... 146
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>A manufacturing enterprise network</td>
<td>10</td>
</tr>
<tr>
<td>3.2</td>
<td>An example of disruption</td>
<td>13</td>
</tr>
<tr>
<td>3.3</td>
<td>Time axis of observation window and disruption in discrete time domain</td>
<td>16</td>
</tr>
<tr>
<td>3.4</td>
<td>A simple system</td>
<td>18</td>
</tr>
<tr>
<td>3.5</td>
<td>$u$ of Case 1, 2, 3a and 4a</td>
<td>19</td>
</tr>
<tr>
<td>3.6</td>
<td>$u$ of Case 1, 2, 3b and 4b</td>
<td>19</td>
</tr>
<tr>
<td>3.7</td>
<td>An example of removable discontinuities that are not allowed</td>
<td>23</td>
</tr>
<tr>
<td>3.8</td>
<td>Examples of jump discontinuities that are allowed</td>
<td>23</td>
</tr>
<tr>
<td>3.9</td>
<td>Time axis of observation window and disruption in continuous time domain</td>
<td>25</td>
</tr>
<tr>
<td>4.1</td>
<td>An example of the serial structure</td>
<td>29</td>
</tr>
<tr>
<td>4.2</td>
<td>An example of the AND structure</td>
<td>29</td>
</tr>
<tr>
<td>4.3</td>
<td>An example of the OR structure</td>
<td>30</td>
</tr>
<tr>
<td>5.1</td>
<td>An example of network of nodes with fixed operations ratio</td>
<td>44</td>
</tr>
<tr>
<td>5.2</td>
<td>A disrupted chain</td>
<td>45</td>
</tr>
<tr>
<td>5.3</td>
<td>An example of network</td>
<td>48</td>
</tr>
<tr>
<td>5.4</td>
<td>System aggregation result</td>
<td>49</td>
</tr>
<tr>
<td>6.1</td>
<td>A serial network system</td>
<td>51</td>
</tr>
<tr>
<td>6.2</td>
<td>The subtree with the root node of $N_{i,p}$</td>
<td>61</td>
</tr>
<tr>
<td>7.1</td>
<td>The DSCDC network system</td>
<td>65</td>
</tr>
<tr>
<td>7.2</td>
<td>Time frame of the nominal-operation period, pre-disruption period and</td>
<td>65</td>
</tr>
<tr>
<td></td>
<td>disruption period</td>
<td></td>
</tr>
<tr>
<td>7.3</td>
<td>Inventory of the node increasing and then decreasing during a disruption.</td>
<td>68</td>
</tr>
<tr>
<td>7.4</td>
<td>$u(t)$ satisfying Stepped Production Property</td>
<td>72</td>
</tr>
<tr>
<td>7.5</td>
<td>An example of the locations of $s$-nodes and $c$-nodes</td>
<td>77</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

A variety of potential disruptive events can affect manufacturing enterprises severely. These may include natural events (such as hurricanes, tornados, ice storms, flood, or earthquakes), accidents (such as fires, power outages, or major equipment failures), transportation disruptions (e.g., bridge and road closures), or man-made events (for example, work outs, terrorism, wars, epidemics, and bankrupt suppliers). How a firm responds to such events is not only critical to the survival of these firms, but also important for the economic health of the communities in which these firms operate. It has a significant impact on the economic survival of the community whether the manufacturing enterprises are able to respond to and recover from highly disruptive events.

A manufacturing enterprise consists of interconnected suppliers, producers, and consumers working together to provide a mixture of certain goods or services to end customers. A disruptive event is such an event that will lead to partial loss of the components in the network or disable some of the connections among the components in a short period of time and may cause a significant impact in economy so that the enterprise needs to reconfigure its network. For example, a fire in Philips New Mexico plant caused $40 million loss sales of high-margin, high-tech chips, and direct damage to the plant of 39 million Euro insurance settlement ([38]). An 18 day labor strike in 1996 at a General Motors brake supplier plant led to 26 assembly plants idle, and caused an estimated reduction of $900 million in quarterly earnings ([8]). Supplier delivery failure of two critical parts resulted in an estimated loss of $2.6 billion for Boeing in 1997 ([8]). The ability of an enterprise to withstand potentially high-impact disruptive events is known as its resilience, which is characterized by the redundant or absorbing capability of the enterprise to the event to “dampen” its impact, and its recovery capability, its ability to quickly resume production or transportation by redistributing its resources. Such capabilities are critical since the impact of disruptive events is significant, and many firms may not be resilient to such events. It is reported in [41] that based on a survey to 199 financial executives and risk managers at Fortune 1000 firms, “more than 75% of respondents say a major disruption to their top earning’s driver would either cause sustained damage to their firm’s earning or threaten its continuity of operations.” Therefore, studies on resilience dynamics in manufacturing enterprises are necessary and important.

Although there are a number of studies on resilience of manufacture systems, supply chains, computer networks, etc., most of previous research is not focused on the dynamic control
issue. A systematic method to quantitatively study the resilience of a manufacturing enterprise by addressing its redundancy and recovery capabilities, response and control policies, has not been well developed to date. The tools that allow the modeling, analysis and simulation of resilience in a manufacturing enterprise are not available.

This research is intended to provide some principles and tools on a general level to help enterprises to deal with the problems of disruptions and enhance their resilience. Due to the prior lack of quantitative study, our objective is to develop the mathematical tools to analyze and enhance the resilience, and to discover in the engineering point of view the significant factors and principles related to resilience.

This research is focused on the control issue within the dynamic process, which is to figure out the solution of control during the disruption to achieve a minimum objective function value, such as cost. With such a control method, the system can respond better during the disruption.

This research includes several topics, such as the mathematical model setup, definitions of systems structures, method of simplifying general system, analysis of optimal response to the change of conditions (resource capacity), conclusion of general control policy, and so on.

The work for this dissertation initially began with discrete time dynamic models, but later work introduced continuous time dynamic models in order to effectively incorporate capacity constraints. This dissertation includes both the prior work using discrete time models, as well as later developments using the continuous time models.

The remaining part of this proposal is structured as: Chapter 2 is the literature review; Chapter 3 introduces the modeling framework of resilient manufacturing enterprises; Chapter 4 introduces several typical structures of networks and systems based on the framework; Chapter 5 discusses the normalization and aggregation to simplify complex system; Chapter 6 presents the analysis of the optimal operation of some systems in discrete time domain; Chapter 7 discuss the optimal operation for downstream buildup in serial network systems in continuous time domain; Chapter 8 extends the analysis of serial network systems to including upstream buildup in continuous time domain; Chapter 9 discuss the optimal operation of a special type of assembly tree network systems in continuous time domain; Chapter 10 makes the conclusion of the current results and presents the problems to be addressed in the future research.
Chapter 2

Literature Review

After the terrorist attack on the US in 2001, research related to resilient enterprise has received more attention. Such studies can be classified based on the areas and issues they focus on, such as resilient infrastructure, resilient supply chain, resilience of computer networks, etc. The following review is separated into several sections, each one corresponding to a particular area. Some of the reviews are introduced in [18].

2.1 Case Studies on Resilience

Many studies analyze the resilience by doing case studies. Paper [38] uses resilience to describe the ability to bounce back from disruptions and disasters by building in redundancy and flexibility. Plenty of case studies of enterprises are carried out, including Toyota, Nokia, General Motors, Southwest Airlines, UPS, Johnson and Johnson, Intel, Amazon.com, and many others. It includes the analysis of their successes, failures, preparations, and methods to reduce vulnerability and increase supply chain flexibility, which provides a rich set of lessons in preparing for and responding effectively to disruptions. Many other investigations based on case studies can also be found in [17], [36], [40], [23] and [33]. These studies are more qualitative or descriptive oriented.

2.2 Optimal Network for Resilience

Many of the quantitative studies on resilience are focused on optimal network design for better resilience. One research ([25]) proposes a methodology for the difficult estimation of traffic efficiency (TE), a measure of network resilience, and a hybrid genetic algorithm to design networks using this measure. Node failures must be taken into account while computing a network resilience measure. Nodes need to be treated differently according to the effects on network service, traffic amount, etc. TE, which is used to measure network resilience, is defined as the expected percent of the total traffic that a network can successfully deliver. A simulation approach is given, which is a very efficient implementation of sequential construction to estimate TE. A GA based algorithm is presented to design networks taking TE into account.
In the work of [44], to achieve maximum global efficiency of the network, an optimization method, memory tabu search, is used to get the optimal network structure. It is found that a network with a small quantity of hub nodes, high degree of clustering may be much more resilient to perturbations than a random network. During the optimizing process, the average shortest path length $L$ becomes short. The increase of the maximal degree of the network indicates the hub nodes’ appearance. The degree correlation coefficient $r$ decreases and is always less than zero, which indicates that nodes with high degree preferentially connect with the low degree ones. The clustering coefficient $C$ increases in the whole process and arrives to a high level, then the network shows a high degree of clustering. Besides, the network’s synchronizability is really reduced.

Cascading failure in complex networks is discussed in [3]. The cascading failures can be described as below. The network is perturbed by the breakdown of a node, which changes the balance of flows within the network, leading to the redistribution of load to other nodes, and then to other adjacent nodes probably. Some definitions are given, such as the average efficiency of a network, the load on each node, etc. The breakdown of a network is simulated by reducing the capacity of a given node. The authors have combined the recursive load redistribution algorithm and efficiency measures with a variation of the Metropolis algorithm. It is shown that many complex networks share common features, such as scale-free degree distribution, short path length, high clustering, and modularity.

### 2.3 Resilient Infrastructure

Analysis on vulnerability of critical infrastructure is presented in [9]. The vulnerability is first analyzed in the case of that the infrastructure is under an intelligent attack. To analyze such vulnerabilities, the attacker-defender models are used to model a complete infrastructure system and its value to society, including how losses of the system’s assets reduce that value, or how improvements in the system mitigate lost value. Reconstitutability is also included. This model can become a submodel in a formal model, which is defender-attacker-defender model. And this can also be simplified to defender-attacker model. Supply chain and other kinds of infrastructure are analysis with the concept of this model. Moreover, an enterprise-wide networking for manufacturing is introduces in [15].

Network-based systems enhance the speed, reliability, and flexibility of information exchange between different components, subsystems, and sectors of the manufacturing environment. The network must be capable of integrating different hardware, software, and communications systems from single or multiple vendors. It must be able to handle heterogeneous traffic. The objective is to enable efficient transmission of various types of information. When implementing a computer network, the most important evaluative criteria are reliability and robustness. Parallel or distributed cellular structures are feasible when supported by information networks and hierarchical controls. Some network technologies which can enhance the enterprise organization are introduced.

### 2.4 Resilient Supply Chain

Most of the studies on manufacturing enterprise resilience focus on supply chain networks. Paper [36] and [40] study on supply chain management to respond to terrorist attacks.
Paper [40] suggests that to protect the supply chain, a series of initiatives can be classified in three groups: physical security, information security and freight security. For each group, they identified two levels of response, basic and advanced. The basic level corresponds to more traditional initiatives that today are almost a standard practice. The advanced level instead is made of more forward-thinking actions, put in place by a limited set of companies.

In paper [37], it is presented that firms should organize to run dual procurement systems where the bulk of the material is bought from inexpensive and innovative offshore suppliers, and at the same time, a portion of the business is given to a local supplier who can pick up the slack in case an attack disrupts transportation lanes. To create a dual inventory system, logistics managers should designate a certain amount of inventory as "Strategic Emergency Stock." This stock should not be used to buffer the day-to-day fluctuations of the processes it feeds. Instead, it should be managed using an inventory discipline that can be summarized as: "Sell-One-Store-One" ("SoSo.") With this discipline the reorder quantity of the items in the strategic emergency stock is raised by the number of item required in this inventory.

In [11], it is argued that a hospital needs to implement more than one supply chain policy in order to achieve its objective of maximizing patient care while avoiding prohibitive costs. The research further proposes that a hospital should develop its supply chain for a specific product based on that product’s unit cost, demand, variability, physical size, and criticality. For hospitals implementing stockless supply chain policies, the demand needs to be analyzed and modeled on a daily basis. Intermittent demand is described as an inventory pattern where there are many periods with no demand and a few periods with either small or large demand.

Risk management to reduce supply chain vulnerability is discussed in [23], [33], [35] and [1]. An empirical study (in [8]) on supply-chain disruptions enumerates several critical issues relative to the analysis and mitigation of the detrimental impact of supply-chain disruptions in a global environment. In addition, broad areas that deserve more research attentions are presented.

Case studies in [39] show that companies’ resilience can be bolstered by either building in redundancy or in flexibility. Although such built-in resilience represents a pure cost increase due to investing in redundancy, many additional benefits for day-to-day operations can be yielded by investing in flexibility. Using Vulnerability Maps, vulnerability is highest when both the likelihood and the impact of disruption are high. Rare, low-consequence events represent the lowest levels of vulnerability and require little planning or action. To see how flexibility can be achieved, consider the essential elements of any supply chain: Material flows from supplier through a conversion process, then through distribution channels. It is controlled by various systems, all working in the context of the corporate culture.

Vulnerable options in supply chain is studied in [5] by using a single-period, multi-stage model of a two-echelon supply chain with competing risky suppliers and single manufacturer. A concept, supplier deferment option, is defined as the option of a supplier to postpone its pricing decisions. The manufacturer is facing an uncertain price for its product. The paper focused on that the benefits of supply diversification also depend on the competition among suppliers and the presence of deferment options. With the model, the effects of the competition and deferment options are analyzed under different conditions of the wholesale prices. The manufacturer always benefits from the option if the wholesale prices are fixed. When suppliers compete in price, the presence of the deferment option reduces competition.
Paper [10] discusses the importance of decoupling disruption and recurrent supply risk during mitigation strategy planning. It is shown that the growth in supply risk from increased disruption probability can be mitigated by increasing use of reliable but more expensive supplier, and the growth in supply risk from increased recurrent uncertainty could be served by increasing use of less reliable but cheaper supplier.

Paper [26] suggests the framework of a decision support system adopting case-based reasoning approach to assist managers in risk management in supply chains by integrating the previously taken actions and a record of previous risk management plans. Paper [6] presents a comprehensive review of studies on risks in supply chains. A Bayesian belief network is proposed to model the probabilities of the risk impacts. Paper [4] reviews the research on scheduling in the presence of unforeseen disruptions in manufacturing systems. It covers several approaches, including a predictive-reactive scheduling approach.

2.5 Modeling related to Resilience in Manufacturing and Supply Chain Systems

Some studies discuss the modeling of Manufacturing and Supply Chain Systems to address issues such as uncertain disruptions and restoration. Paper [16] proposes a framework for modeling production generally in continuous time domain, including material flows of inputs and outputs. It reformulates standard linear programming formulations, Manufacturing Resources Planning, and Critical Path Methods.

Paper [43] develops a mathematic model to analyze the strategies of ordering from two suppliers of which one is subjected to uncertain disruption. Different strategies such as inventory mitigation and contingent rerouting are shown to have merits under different conditions.

Paper [30] introduces a two-sector model to analyze the economic growth path and restoration process of supply chains after natural disasters. It is found that the production in the final good sector decreases as a result of the decrease in the capital and production of intermediate goods.

A hybrid inventory-production model is described in [13] to represent a production node in supply chain networks, and the optimization of inventory levels is formulated. Paper [22] develops a manufacturer - retailer dynamic model to specifically formulate different inventory policies to meet different objectives in supply chains. Maintenance of certain levels of safety stocks is found to be useful to deal with unexpected disruptions. In Paper [12], tactical planning under uncertain and disrupted environment is studied. A decision making process is proposed based on dynamic successive planning steps.

2.6 Resilience of Computer Networks

In addition, many studies are focused on fault resilience in computer and communication networks. For example, fault-resilient sensing in wireless sensor networks is introduced in paper [31], which proposes a method to determine the trajectories of mobile sinks in the wireless sensor network which would be fault-resilient. Sensors are first clustered by using
the K-means clustering algorithm. The migration route of mobile sinks is determined as an approximate solution of Traveling Salesman Problem. The main job is to keep good efficiency and to protect the sink from attack. The change of trajectories depends on the sensors that survived the attack. It can be reprogrammed according to the environment.

Paper [14] studies automatic high-performance reconstruction and recovery in computing systems. It is proposed that in order to recover from intrusions, one requires a complete and usable log of all system activity. The system supporting reliable reconstruction must gather an accurate, high-resolution image of system activities. It must also be able to generate a selective “undo” log that allows the target system to be restored as if the intrusion never happened. The activity log should be gathered in a tamper-resistant way. The system should have a small effect on the performance of the target system, and be effective post-facto analysis. Forensix system is presented in three main parts. First, it performs comprehensive monitoring of the execution of a target system at the kernel event level, giving a high-resolution, application-independent view of all activity. Second, it streams the kernel event information, in real-time, to append-only storage on a separate, hardened, logging machine, making the system resilient to a wide variety of attacks. Third, it uses database technology to support high-level querying of the archived log, greatly reducing the human cost of performing analysis and recovery.

In order to achieve optimal use of uplink bandwidth and balance distributed load for selection of serving peers, paper [42] proposes a peer-to-peer system to cooperatively stream video. In this Push-to-Peer Video-n-Demand system, video is first pushed to a population of peers. Peers seeking specific content then pull content of interest from other peers, as in a traditional peer-to-peer system. Then, two kinds of data placement policies, full-striping and code-based, are considered, which are shown to be able to achieve the upper performance bounds. Moreover, a randomized job placement algorithm is proposed to deal with the infeasibility of perfect pooling.

2.7 Resilience of Other Systems

Resilient study has also been carried out in social and eco-bio systems. A simple dynamic simulation model is applied in paper [24] to characterize the stability, resilience and sustainability of pasture-based beef grazing enterprises by using pasture envelope concept, which is a form of phase diagram to illustrate the trajectories over time of key biophysical variables. The resilience of the enterprise is analyzed through “drought” conditions by reducing the maximum growth rate of pasture species by 50%. It is found that the resilience declines as stocking rates increase.

An approach is proposed in paper [34] based on an initial optimization which uses an aggregate manufacturing representation to generate initial cost-effective and responsive alternatives. The first step is to analyze the ability of different potential plant types, so as to quantify the perception on the relative advantages of each design and to identify the best investment decision for the manufacturing of the large number of products considered. An optimization model, MILP, is proposed, which balances the trade-offs existing between capital, manufacturing and stock costs and maximizes the Net Present Value of the investment, while taking into account the constraints imposed by the manufacturing technologies. Its
main outcome is the types of plants to build, the number and size of units within the plant and the distribution of the products among the plants.

Several principles are proposed in [7]. Leadership achieves a balance between risk taking and risk containment to ensure ongoing innovation, but in the context of prudent risk minimization. A resilient culture is built on principles of organizational empowerment, purpose, trust and accountability. People who are properly selected, motivated, equipped and led will overcome almost any obstacle or disruption. The Systems of RVO is built on an infrastructure of extensive enterprise connectivity and information robustness. Workplace resilience is achieved through the distribution of the workplace into multiple, dispersed settings. Alternative workplace techniques provide the level of workplace flexibility and agility that is essential for mitigating the risk of catastrophic or disruptive incidents at an enterprise location.

In paper [46], building a career-resilient workforce in a company is introduced. A career-resilient workforce means a group of employees who not only are dedicated to the idea of continuous learning but also stand ready to reinvent themselves to keep pace with change; who take responsibility for their own career management; and, last but not least, who are committed to the company’s success. First, the traditional definition of loyalty must go; Second, the usual view of a career path must change; third, all employees must be much more aware that the purpose of the organization is to provide goods and services that customers value and that if the organization does not do that, nobody in it will have a job; fourth, a new relationship must be established between the organization and its employees. Two basics of career resilience are self-assessment and benchmarking. The company must be open with employees and help them explore opportunities, facilitate lifelong learning and job movement, and, if it comes to that, support no-fault exits. There are systems for organizations to minimize the risks involved in adopting career-resilience program. Establishing a career-management center helps a program gain credibility, and can convince the employees that career-resilience program can truly serve their interests. Finally, the career resilience program requires the support from the top.

Paper [29] introduces the resilient virtue organization, which - representing the intersection of resilience and virtual operations - is a business and organizational model that intentionally designs resilience into its business operations, security mechanisms, people selection, workplace development, communications networks, architecture, security measures, learning, collaboration, site selection, vendor evaluation and trading partner relationships. Business resilience requires risk analysis, investment and a supporting environment. Virtual business and virtual work require vigilant attention to resilience.

In [41], it is introduced that in military terminology, most enterprise risk management (ERM) programs rely on ”point solutions,” which attempt to moderate risks by ”hardening” potentially vulnerable spots against attacks, a futile exercise in a networked enterprise. ER results from a planned series of safeguards against discontinuities - encompassing everything from logistics, inventory control, and distribution channels to relations with government agencies, customers, and suppliers. It is suggested that the firm should identify its key earnings drivers and their associated risks. The firm should use modeling tools and best practices in enterprise design to produce initial snapshots of an enterprise’s ”resilience profile” for each essential aspect of a company: financial, operations, technology, personnel, and security. It should develop a new resilience program based on the analyses of the firm’s earnings-related risk mitigation needs.
Paper [2] presents that enhancing social and economic resiliency as well as meeting security and emergency response needs to develop and implement dual-use technologies that offer societal benefits even if anticipated disasters never occur.

A number of analytical studies investigate the disruptions in some specific systems other than manufacturing systems. In [32], the workforce-related economic impact of a pandemic on the commonwealth of Virginia is studied with a dynamic inoperability input-output model. In [45], a new approach of workflow modeling is proposed and applied to the incident command systems for emergent incidents. The study of [28] sets up a model for short-term air traffic flow to deal with airborne holds due to unpredicted disruptions. The fragility of Internet highlighted by the earthquake in 2006 is studied in [27], and a trust-based economic framework is proposed to improve the flexibility of the Internet routing and take economic incentives into account.

2.8 Conclusion

In spite of these efforts, the study on resilience in manufacturing enterprise is still limited. Most of these studies are focused on the basic concepts and qualitative analysis of system resilience. The approaches and policies introduced to enhance resilience are mainly about static planning issues, such as setting up redundancy. Although there are some studies using mathematical methods, such as those on computer network, they are still too specific to be applicable to general enterprise system. Some others are focused on manufacturing and supply chain systems modeling, but the issues they are solving are still too restricted and can not provide operation policies which can handle general systems to deal with disruptions. There is little research managing to discover the inner property of resilience. A systematic method to quantitatively study the resilience of a manufacturing enterprise by addressing its responding and control policies has not been well developed to date. The tools that allow the modeling, analysis and simulation of resilience in a manufacturing enterprise are not available. Real-time control issues for resilience are not well addressed.
Chapter 3

Modeling of Resilient Enterprises

In this chapter, we will introduce the mathematic models for the analysis of resilient enterprises. Initially, we developed the model in discrete time domain, and used it to solve some issues of resilient operations. A portion of the results are introduced also in [18]. However, we found the difficulty to do the analysis when considering capacity constraints in discrete time domain. Therefore, we extended the model to continuous time domain, and corresponding issues were analyzed based on it.

This chapter will introduce both the discrete and continuous time models. Later chapters will include analyses which are based on either of these two models, respectively.

3.1 System Description and Problem Formulation in Discrete Time Domain

3.1.1 Network System Model

A manufacturing enterprise can be structured as a network of nodes defining the supply chain from sources to customer. The nodes on the network represent facilities, operations, transportation links, sources, and consumers. Each node has dynamics that describe the transformation of or flow of product, the use of resources, the capacity of operations, and the cost and responsiveness to change.

![Figure 3.1: A manufacturing enterprise network](image)
Each node represents an operation to transform one or more products into one or more other products at a node. For our model, we have \( n_p \) products, each is denoted as product \( i \), where \( i = 1, 2, \ldots, n_p \). And we use the \((n_p \times 1)\) vector \( x(k) \) to represent the quantity of all \( n_p \) products in the system at the end of a time period \( k \). The element of \( x(k) \), \( x_i(k) \) stands for the quantity of product \( i \) at time \( k \). An operation of node \( i \) is represented by a \((n_p \times 1)\) vector \( b_i \). For example, consider the following:

\[
\begin{bmatrix}
-1 \\
-2 \\
1 \\
0 \\
0
\end{bmatrix}
\]

The operation \( b_1 \) represents an assembly of one of product \( 1 \) with two of product \( 2 \) to produce one of product \( 3 \).

Consider a node \( N_j \) and its vector \( b_j \) in the matrix \( B \). Denote the \( i \)-th element in \( b_j \) as \( b_j(i) \).

**Network Matrix and Operation Vector**

Let \( n_o \) indicate the number of nodes or operations. Define the \((n_p \times n_o)\) matrix \( B = [b_1, b_2, \cdots, b_{n_o}] \). Let the \((n_p \times 1)\) vector \( u(k) \) represent the production operation executed at time period \( k \), such that the value of the element \( u_i(k) \) is the number of times that operation of node \( i \) occurs over time period \( k \). We assume there is no loss or spoilage of inventory. This then gives us the inventory update equation:

\[
x(k + 1) = x(k) + Bu(k).
\]  \hspace{1cm} (3.1)

**Constraints**

There are several constraints for \( x(k) \) and \( u(k) \). We only consider four kinds of constraints. First, *No Backorder Constraint* states that inventory of each item can never be negative. It can be described as:

\[
\text{NB-Constraint: } \forall k : \quad x(k) \geq 0;
\]  \hspace{1cm} (3.2)

Our second constraint concerns the limited capacity of resources in any system. Let \( n_r \) be the number of resources, such as people, machines, facilities, trucks, etc. For each resource \( i \), let the \( 1 \times n_o \) dimension non-negative vector \( R_i \) indicate the operations that use the \( i \)-th resource. And for each node \( N_j \), let the \( n_r \times 1 \) dimension non-negative vector \( r_j \) indicate the resources used by \( N_j \). For example:

\[
\begin{bmatrix}
1 \\
2 \\
0 \\
0
\end{bmatrix}, \quad
\begin{bmatrix}
0 \\
0 \\
1 \\
0
\end{bmatrix}
\]
In this example, by \( r_1 \), operation 1 requires 1 capacity unit of resource 1, and 2 capacity units of resource 2. By \( r_2 \), we have operation 2 requires 1 capacity unit of resource 3.

Define \( R = [r_1 \ r_2 \ \cdots \ r_{n_o}] \). Thus, \( R = [R_1^T \ R_2^T \ \cdots \ R_{n_o}^T]^T \). Also define the \( n_r \times 1 \) vector \( c(k) \) as the maximum number of units of each resource that can be used at time period \( k \). Since our production operations cannot exceed this over any time period, this then gives us our Capacity Constraint:

\[
\text{Capacity Constraint: } (\forall k) : \ Ru(k) \leq c(k); \quad (3.3)
\]

Our next constraint is associated with the constraint of change in operations. With some operations, there may be limits to how quickly one can ramp up or ramp down an operation. Let the \( n_o \times 1 \) vector \( \varsigma \) indicate the limit of change of production operation over a time period. This then gives us our Setup (changeover) Constraint:

\[
\text{Setup Constraint: } (\forall k) : \ u(k) - u(k - 1) \leq \varsigma. \quad (3.4)
\]

(Note that this constraint only applies to increases in production. One could similarly define a constraint on how quickly one can decrease production.) As an example, an S-Constraint for the \( i \)-th operation might have \( \varsigma_i = 0.2 \), which would mean that production in the \( i \)-th operation cannot increase more than 0.2 units of production each time period.

One goal of production is to produce products according to a customer demand. Note from our earlier example above that the operation represented by \( b_4 \) is the delivery of a given product outside the system. Let \( n_d \), where \( n_d \leq n_o \), be the number of final operations, such as a delivery outside the system. Then let the \( n_d \times n_o \) matrix \( D \) be a binary matrix with exactly one non-zero element per row. This then extracts the final operations from the operation vector \( u(k) \), so that \( Du(k) \) is a vector of final operations done over period \( k \). Let \( d(k) \) be the desired demand for products from these final operations at time period \( k \). (If backorders are allowed, then \( d(k) \) could be defined to include some or all of the unmet demand of prior periods.) We assume that we cannot sell product that exceeds demand, so this gives us our last constraint, Demand Constraint:

\[
\text{Demand Constraint: } (\forall k) : \ Du(k) \leq d(k). \quad (3.5)
\]

As an example, consider that there are five operations, so \( u(k) \) is \( 5 \times 1 \). However, only the fourth and fifth operations are final operations (representing delivery of a product outside the system), so we have the matrix

\[
D = \begin{bmatrix}
0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 \\
\end{bmatrix}.
\]

Suppose that for each time period, there is a demand at time \( k \) of 10 of \( u_4 \) and 15 of \( u_5 \). Then \( d(k) = [10 \ 15]^T \). The D-Constraint then says that we cannot have more than the demanded delivery of these products in the \( k \) time period.

Note that although no back order assumption is introduced in this model, the unsatisfied demand at time \( k \) can be included in the demand of a later time period, \( d(k') \), where \( k' > k \). Clearly \( d(k') \) is a function of \( d(k) \) and \( u(k) \).

The operation direction is determined by the sign of elements in \( u \). At any time \( k \), if the \( i \)-th element in \( u(k) \) are less than 0, it means that \( N_i \) are using its product to produce the
raw materials, and also has negative production cost. This is the case we will not discuss in our research. So, a constraint is added to set the operation direction as from raw material to product. For any time \( k \) and any \( i \), \( u_i(k) \geq 0 \).

\[
\text{Operation Direction Constraint: } (\forall k): \ u(k) \geq 0.
\] (3.6)

**Disruption**

Generally, the disruption can cause some of the nodes to reduce their operation amount. The primary form of disruption that we consider is a loss of capacity. Such loss can be viewed as removal or reduced capacity of one or more nodes. A loss of capacity would represent a reduction in \( c(k) \) of Capacity Constraint over some time. The system can be considered as having at least two states: Nominal State and Disruption State. Nominal state is the state when capacity is not reduced by disruption. We denote nominal capacity vector as \( \bar{c} \). Disruption state is the one when the capacity of some resource(s) is reduced by disruption. In nominal state, for any resource \( i \), its capacity \( c_i(k) = \bar{c}_i \). In disruption state, the duration of disruption state is denoted as \( \kappa_{dis} \), and the start point is denoted as \( k_{dis} \). If resource \( i \) is reduced by disruption, then \( c_i(k) < \bar{c}_i \), where \( k \in [k_{dis}, k_{dis} + \kappa_{dis}] \).

**Definition 1 (Disruption).** A disruption is an event which

1. occurs at time \( k_{dis} \) when \( c_i(k_{dis} - 1) = \bar{c}_i \), and \( c_i(k_{dis}) < \bar{c}_i \), and

2. causes \( c_i(k) < \bar{c}_i \), when \( k_{dis} \leq k < k_{dis} + \kappa_{dis} \),

where \( c_i(k) \) is the capacity of disrupted resource.

![Figure 3.2: An example of disruption](image)

Figure 3.2 shows an example of disruption. The disruption state starts at time \( k_{dis} \) and has a duration of \( \kappa_{dis} \). The capacity of the disrupted resource \( c_i(k) \) is decreased in disruption state.

**Cost**

In our objective function described in Section 3.1.3, we considered four kinds of costs. The total cost is made up of production cost \( (q_u u(k)) \), inventory storage cost \( (q_x x(k)) \), changeover cost \( (q_\zeta (u_k - u_{k-1})) \), and lost demand cost \( (d(k) - Du(k)) \).
System Model

To conclude the items introduced above, a system can be defined with $B$, $R$, $D$, $q_u$, $q_x$, $q_\varsigma$, $c$, $\varsigma$, and $d$. The system can be denoted as $N = (B, R, D, q_u, q_x, q_\varsigma, c, \varsigma, d)$. $B \in \mathbb{R}^{n_p \times n_o}$, $R \in \mathbb{R}^{n_r \times n_o}$, $D \in \mathbb{R}^{n_d \times n_o}$, $q_u \in \mathbb{R}^{1 \times n_o}$, $q_x \in \mathbb{R}^{1 \times n_p}$, and $q_\varsigma \in \mathbb{R}^{1 \times n_o}$. $c$, and $d$ are functions of $k$. $c(k) \in \mathbb{R}^{n_r \times 1} \cup \infty$, and $d(k) \in \mathbb{R}^{n_d \times 1}$. $\varsigma \in \mathbb{R}^{n_o \times 1}$.

3.1.2 Nodes Description

Each node represents an operation. For a node $N_i$, its operation amount is $u_i(k)$. When $N_i$ operates, four things can be changed: inventory amount, production cost, resource used, and demand filled. From Equation (3.1), (3.3), (3.5), and (3.8), we know these four changes of the whole system are:

- change of inventory amount:
  $$Bu(k) = [b_1 \ b_2 \ ... \ b_n]u(k);$$
- resource used:
  $$Ru(k) = [r_1 \ r_2 \ ... \ r_n]u(k);$$
- demand filled:
  $$Du(k) = [d_1^D \ d_2^D \ ... \ d_n^D]u(k);$$
- production cost:
  $$q_u[u(k) = [q_u(N_1) \ q_u(N_2) \ ... \ q_u(N_n)]u(k);$$
- changeover cost:
  $$q_\varsigma[u(k) \ u(k-1)] = [q_\varsigma(N_1) \ q_\varsigma(N_2) \ ... \ q_\varsigma(N_n)]\sum[\varsigma]u(k) - u(k-1)].$$

The changes contributed by $N_i$ are: $b_iu_i(k)$, $r_iu_i(k)$, $d_i^D u_i(k)$, $q_u(N_i)u_i(k)$ and $q_\varsigma(N_i)[u_i(k) - u_i(k-1)]$. Moreover, the vector $\varsigma$ in changeover constraint defines the limit of each node respectively. $\varsigma_i$ is corresponding to $N_i$ only. Hence, the parameters of $N_i$ can be selected as $b_i$, $r_i$, $d_i^D$, $q_u(N_i)$, $q_\varsigma(N_i)$, and $\varsigma_i$. We denote this vector of parameters as:

$$P(N_i) := \begin{bmatrix}b_i \\
r_i \\
d_i^D \\
q_u(N_i) \\
q_\varsigma(N_i) \\
\varsigma_i\end{bmatrix}. \quad (3.7)$$

It is a constant vector which does not change with $k$.

3.1.3 Problem Formulation

The model defined in the previous section models the system response to production orders $u(k)$ over time, with constraints due to capacity, demand, and setup. In this section, we want to evaluate the production order function $u(k)$ compared to other possible production orders. To do this, we will define a cost function.

For this research, we consider the following types of costs.
• **Production Cost:** Define the $(1 \times n_o)$ row vector $q_u$ such that $q_u u(k)$ is the cost of the production operations at time $k$. For a product sourcing operation, this can represent the cost of raw materials purchased. For an assembly or disassembly operation, or other transformation or transportation operation, this could represent the cost of energy or the cost of using a resource, etc. For a final operation, the cost may actually be negative, representing a gain due to delivery of the final product to a customer. Thus we could view this minimization of cost as a maximization of profit.

• **Inventory Storage Cost:** This is the cost of maintaining inventory, which could include the cost of warehouse, inventory tracking, tied up capital, etc. Define the nonnegative $(1 \times n_p)$ row vector $q_x$ such that $q_x x(k)$ is the cost of inventory.

• **Operation Changeover Cost:** Define the nonnegative $(1 \times n_o)$ vector $q_\varsigma$ such that $q_\varsigma (u(k) - u(k - 1))$ is the cost of changeover to give an increase in operations over time.

• **Cost of Lost Demand:** As already defined, $d(k)$ represents the demand for final product at time $k$, and $Du(k)$ represents the demand satisfied. Thus, $d(k) - Du(k)$ is unsatisfied demand.

At time $k$, the instant total cost $C_{ins}(k)$ is the weighted sum of these four costs.

$$C_{ins}(k) := a_u q_u u(k) + a_x q_x x(k) + a_d [d(k) - Du(k)] + a_\varsigma q_\varsigma [u(k) - u(k - 1)]$$

The scalar values $a_u$, $a_x$, $a_\varsigma$, and $a_d$ allow us to weight various elements of the cost and all of them are non-negative numbers.

The objective is to minimize the cost by applying $u(k)$. $u(k)$ is the control signal which represents the operation commands given to the system. We need to find the optimal $u(k)$ to achieve the minimum cost. We evaluate the total cost within a time window defined by $\kappa_p$ and $\kappa_f$, which are the past and future coverage respectively. The evaluation over the past with actual $d(k)$ is over time interval $[k - \kappa_p, k]$, and the evaluation over the future (with predicted $d(k)$) is $[k, k + \kappa_f]$. Combining the cost components give us the following total cost $C$ over the horizon, and letting $d(k)$ for time up to $k$ be actual demand and $d(k)$ for time after $k$ be predicted demand, the objective function is the total cost over this time window, which is:

$$C := \sum_{k' = k - \kappa_p}^{k + \kappa_f} \{ a_u q_u u(k') + a_x q_x x(k') + a_d [d(k') - Du(k')] + a_\varsigma q_\varsigma [u(k') - u(k' - 1)] \}.$$  

(We note that the function $x_{k'}$ in the cost equation is a function of $u_{k'}$. Therefore, the resilience of a manufacturing enterprise is characterized through the total cost over time period $\kappa_p + \kappa_f + 1$. In addition, by considering different $\kappa_p$ and $\kappa_f$, such a function can be used to evaluate the cost of past actions, future responses, and a combination of them. Figure 3.3 shows the time axis of observation window and disruption.)
Our problem can be formulated as:

Minimize : \[ C = \sum_{k'=k-k_p}^{k+k_f} \{ a_u q_u u(k') + a_x q_x x(k') + a_d [d(k') - D u(k')] + a_\varsigma q_\varsigma [u(k') - u(k' - 1)] \}, \]

Subject to : 
- No Backorder Constraint: \( x(k) \geq 0 \),
- Capacity Constraint: \( R u(k) \leq c(k) \),
- Setup Constraint: \( u(k) - u(k - 1) \leq \varsigma \),
- Demand Constraint: \( D u(k) \leq d(k) \),
- Operation Direction Constraint: \( x(k + 1) = x(k) + B u(k), \)
- \( k > \kappa_p \),
- \( \kappa, \kappa_p, \kappa_f \in \mathbb{N} \).

In order to simplify the analysis in this proposal, we introduce some assumptions, which make the model much easier to solve.

**Assumption 1.** \( a_d \gg a_u, a_d \gg a_x, \) and \( a_d \gg a_\varsigma \), so that a necessary condition of minimum total cost is that lost demand cost is minimum.

**Assumption 2.** \( a_\varsigma \ll a_u, a_\varsigma \ll a_x, \) and \( a_\varsigma \ll a_d \), which means changeover cost is so small that could be ignored.

**Assumption 3.** Assume \( \varsigma \gg u(k) - u(k - 1) \forall k \), which means changeover constraint is so loose that at any time the increase of operations is not larger than the limit.

We also assume that we can collect advance notices of disruptions. Many disruptive events have some prior warning. The most obvious would be scheduled shutdowns of suppliers or facilities due to maintenance, or anticipated transportation disruptions due to bridge or road closures. Even supply disruptions due to political unrest, supplier bankruptcies, or labor disputes might be anticipated, based, respectively, on periods of political tensions, financial difficulties of suppliers, or periods of labor negotiations. In later chapters, we will consider the problem of how to use this warning of a disruption to change production and allocate inventory to achieve a minimum cost with minimum impact on the supply of final products. Thus, we introduce the following assumption.
Assumption 4. \( \kappa_f \geq 1 \), which means the start time of disruption can be known in advance. It is known in \( \kappa_f \) time units before disruption happens.

In discrete time domain, we don’t consider capacity limit in Nominal State. Thus, we introduce the following assumption.

**Assumption 5.** Assume \( \tau \gg R_i(k) \) when \( k \notin [k_{dis}, k_{dis} + \kappa_{dis}) \), which means all resources are always sufficient in nominal state; \( c_i(k) = 0 \) when \( k \in [k_{dis}, k_{dis} + \kappa_{dis}) \) and resource \( i \) is disrupted, which means the amount of disrupted resource is zero in disruption state. We use \( R_i \) to denote the matrix mapping \( u(k) \) to resource \( i \).

Then the simplified problem is formulated as below, with a second priority objective function that is a concern only after the first priority objective function is met.

\[
\text{Minimize : First Priority Objective:} \quad \sum_{k' = k - \kappa_p}^{k + \kappa_f} \{a_d[d(k') - D u(k')]\},
\]

\[
\text{Second Priority Objective:} \quad \sum_{k' = k - \kappa_p}^{k + \kappa_f} \{a_uq_u u(k') + a_xq_x x(k')\},
\]

**Subject to :**

- No Backorder Constraint:
  \( x(k) \geq 0 \),
- Capacity Constraint:
  \( R_i u(k) = 0, \forall k \in [k_{dis}, k_{dis} + \kappa_{dis}) \),
- Demand Constraint:
  \( D u(k) \leq d(k) \),
- Operation Direction Constraint:
  \( u(k) \geq 0 \),
  \( x(k + 1) = x(k) + Bu(k), \)
  \( \kappa_f \geq 1 \),
  \( k > \kappa_p \),
  \( \k, \k_p, \k_f \in \mathbb{N} \).

In addition, the assumptions also simplify the parameter vector of nodes. As the changeover cost and changeover constraint are ignored. The parameters of nodes can be redefined as:

\[
P(N_i) := \begin{bmatrix} b_i \\ r_i \\ d_i^p \\ q_u(N_i) \end{bmatrix}.
\]  \( (3.9) \)

Then the changes of the variables affected when \( N_i \) operates can be expressed as:

\[
P(N_i) u_i(k) = \begin{bmatrix} b_i u_i(k) \\ r_i u_i(k) \\ d_i^p u_i(k) \\ q_u(N_i) u_i(k) \end{bmatrix}.
\]
3.1.4 A Simple Example of Analysis in Discrete Time Domain

Once we set up this mathematical model and formulate the problem, we can begin our analysis of the enterprise system, and calculate the control solution to reduce the impact of disruption. Our analysis is started with a simple system, which shows the process of solving the optimal problem we formulate. Some of the results are presented in [19]. The assumptions are taken into account so that the problem is simplified. The solution is the optimal control signal, which can help to analyze the system performance and response. As the simple system is a building block of more complex system, the analysis results of the simple system are useful for the analysis of complex system.

Consider a simple system shown in the figure below:

![Figure 3.4: A simple system](image)

The network matrix can be denoted as:

\[
B = \begin{bmatrix}
  b_0 & b_1 & b_2 & b_3 \\
-1 & 1 & 0 & 0 \\
0 & -m_2 & m_2 & 0 \\
0 & 0 & -m_3 & m_3
\end{bmatrix}.
\]

Demand is \(d\), and demand matrix is \(D = \begin{bmatrix} 1 & 0 & 0 & 1 \end{bmatrix}\). Initial inventory is \(x(0) = 0\).

We want to analyze how the system will respond when disruption occurs at \(N_2\). \(R_2 = [0 \ 0 \ r_2 \ 0]\). The time point of the disruption beginning is \(k_{dis}\). The predicted length of the disruption is denoted as \(\kappa'_{dis}\). Its value equals to the smaller one between \(\kappa_f\) and the real length of the disruption.

To analyze the simple system, we consider several cases. The derivation of these cases is given in the Appendix A, where we determine the optimal control signal \(u\) (the operations of all the nodes) for the system under a disruption. We summarize these results here:

**Case 1** Nominal stage. \(k < k_{dis} - \kappa_f\).

\[
u(k) = \begin{bmatrix} d & d & d \end{bmatrix}^T.
\]

**Case 2** Preparing for disruption. \(k_{dis} - \kappa_f \leq k \leq k_{dis} - 2\).

\[
u(k) = \begin{bmatrix} d & d & d \end{bmatrix}^T.
\]

**Case 3** Right before disruption. \(k = k_{dis} - 1\).

**Case 3a** For \(q_x(N_2)m_2 > q_x(N_1)\),

\[
u(k) = \begin{bmatrix} d, & d + d \cdot \kappa'_{dis}, & d + d \cdot \kappa'_{dis}, & d + d \cdot \kappa'_{dis} \end{bmatrix}^T.
\]
Case 3b For $q_x(N_2)m_2 \leq q_x(N_1)$,

$$u(k) = [d, d, d + d \cdot \kappa'_\text{dis}, d + d \cdot \kappa'_\text{dis}]^T.$$ 

Case 4 During disruption. $k_{\text{dis}} \leq k \leq k_{\text{dis}} + \kappa'_{\text{dis}} - 1$.

Case 4a For $q_x(N_2)m_2 > q_x(N_1)$,

$$u(k) = [d \ 0 \ 0]^T.$$

Case 4b For $q_x(N_2)m_2 \leq q_x(N_1)$,

$$u(k) = [d \ d \ 0]^T.$$

The $u$ of Case 1, 2, 3a and 4a is shown in the figure below:

<Figure 3.5: $u$ of Case 1, 2, 3a and 4a.>

The $u$ of Case 1, 2, 3b and 4b is shown in the figure below:

<Figure 3.6: $u$ of Case 1, 2, 3b and 4b.>

Before disruption happens ($k < k_{\text{dis}} - \kappa_f$), we only have production cost. In order to make no lost demand, all the nodes have operation amount of $d$. In each cycle, the production cost is

$$C_{\text{nominal}} = a_uq_uu = a_uq_u \begin{bmatrix} d \\ d \\ d \\ d \\ d \\ d \end{bmatrix}.$$
During disruption \((k_{\text{dis}} \leq k \leq k_{\text{dis}} + \kappa'_{\text{dis}} - 1)\), \(N_3\) and \(N_2\) will stop working during disruption. In order to minimize the lost demand cost, \(N_0\) must try to keep working after disruption. So, the system needs to build up some inventory (during \(k_{\text{dis}} - \kappa_f \leq k \leq k_{\text{dis}} - 1\)) for use during disruption. Building up either \(N_2\) or \(N_1\) can support \(N_0\), while the former requires also \(N_1\) to work. In fact, we need to choose the one with lower storage cost (according to \(q_x(N_2)m_2\) and \(q_x(N_1)\)). The amount of the buildup can be predicted. We denote it as \(\kappa'_{\text{dis}}\). Its value equals to the smaller one between \(\kappa_f\) and the real length of the disruption.

First, let’s consider to build up \(N_1\) \((q_x(N_2)m_2 > q_x(N_1))\). Before the disruption \((k_{\text{dis}} - \kappa_f \leq k \leq k_{\text{dis}} - 1)\), the system will increase the operations of \(N_1\), \(N_2\), and \(N_3\) to build up \(N_1\). Its build-up amount needs to be \(d \cdot \kappa'_{\text{dis}}\). As we do not consider changeover cost and constraint, whether the operations are increased earlier or later can only affect the inventory storage cost. And increasing the operations later can reduce the inventory storage cost. So the system will build up \(N_1\) in the cycle right before disruption \((k = k_{\text{dis}} - 1)\).

In the cycle right before disruption \((k = k_{\text{dis}} - 1)\), the production cost will be:

\[
C^{(\text{product}_1)}_{a,k_{\text{dis}}-1} = a_u q_u \begin{bmatrix} d \\ d + d \cdot \kappa'_{\text{dis}} \\ d + d \cdot \kappa'_{\text{dis}} \\ d + d \cdot \kappa'_{\text{dis}} \end{bmatrix} = C_{\text{nominal}} + a_u q_u \begin{bmatrix} 0 \\ d \cdot \kappa'_{\text{dis}} \\ d \cdot \kappa'_{\text{dis}} \\ d \cdot \kappa'_{\text{dis}} \end{bmatrix}.
\]

Then during the next \(\kappa'_{\text{dis}}\) cycles of disruption, only \(N_0\) will work, and the operation amount is \(d\) in each cycle. After \(\kappa'_{\text{dis}}\) cycles, the system will enter steady state. The production cost during this period is:

\[
C^{(\text{product}_1)}_{a,\text{dis}} = \kappa'_{\text{dis}} \cdot a_u q_u \begin{bmatrix} d \\ 0 \\ 0 \\ 0 \end{bmatrix} = a_u q_u \begin{bmatrix} d \cdot \kappa'_{\text{dis}} \\ 0 \\ 0 \\ 0 \end{bmatrix}.
\]

Besides, the inventory storage cost is:

\[
C^{(\text{product}_1)}_{x,\text{dis}} = a_x q_x(N_1)\left[d \cdot \kappa'_{\text{dis}} + d \cdot (\kappa'_{\text{dis}} - 1) + d \cdot (\kappa'_{\text{dis}} - 2) + \ldots + d\right] = a_x q_x(N_1)d \left(\frac{\kappa'_{\text{dis}} + 1}{2}\kappa'_{\text{dis}}\right).
\]

When the disruption starts, the amount of \(\text{product}_1\) is \(d \cdot \kappa'_{\text{dis}}\). Then it will be reduced by \(d\) in each cycle.
So the total cost in the transition period \( (k \in [k_{dis} - 1, k_{dis} + \kappa'_\text{dis} - 1]) \) is:

\[
C_{\text{dis}}^{(\text{product}_1)} = C_{\text{nominal}} + a_u q_x \begin{bmatrix}
0 \\
d \cdot \kappa'_\text{dis} \\
d \cdot \kappa'_\text{dis} \\
d \cdot \kappa'_\text{dis}
\end{bmatrix} + a_u q_x \begin{bmatrix}
d \cdot \kappa'_\text{dis} \\
0 \\
0 \\
0
\end{bmatrix}
\]

\[
+ a_x q_x(N_1) d \frac{\kappa'_\text{dis} + 1) \kappa'_\text{dis}}{2}
\]

\[
= C_{\text{nominal}} + a_u q_x \begin{bmatrix}
d \cdot \kappa'_\text{dis} \\
d \cdot \kappa'_\text{dis} \\
d \cdot \kappa'_\text{dis} \\
d \cdot \kappa'_\text{dis}
\end{bmatrix} + a_x q_x(N_1) d \frac{\kappa'_\text{dis} + 1) \kappa'_\text{dis}}{2}
\]

\[
= C_{\text{nominal}} + \kappa'_\text{dis} \cdot C_{\text{nominal}} + a_x q_x(N_1) d \frac{\kappa'_\text{dis} + 1) \kappa'_\text{dis}}{2}.
\]

Although the production cost is increased before disruption \( (k_{dis} - \kappa_f \leq k \leq k_{dis} - 1) \) due to the building up, the stop of working during the disruption \( (k_{dis} - \kappa_f \leq k \leq k_{dis} + \kappa'_\text{dis} - 1) \) also reduces the production cost. Considering the overall production cost, it is equivalent to that the production cost in each cycle does not change, which is always \( C_{\text{nominal}} \). So the total production cost in transition period \( (k_{dis} - 1 \leq k \leq k_{dis} + \kappa'_\text{dis} - 1) \), which includes one cycle right before disruption and next \( \kappa'_\text{dis} \) cycles, is \( C_{\text{nominal}} + \kappa'_\text{dis} \cdot C_{\text{nominal}} \). This part remains the same even if there is no disruption. So the disruption only brings the storage cost, which is \( a_x q_x(N_1) d \frac{\kappa'_\text{dis} + 1) \kappa'_\text{dis}}{2} \).

Then we also analyze what if we build up product\(_2\) instead of product\(_1\) \( (q_x(N_2) = q_x(N_1)) \). In this case, \( N_3 \) and \( N_2 \) will increase operations before disruption \( (k_{dis} - \kappa_f \leq k \leq k_{dis} - 1) \). \( N_1 \) and \( N_0 \) will keep working during the disruption \( (k_{dis} \leq k \leq k_{dis} + \kappa'_\text{dis} - 1) \). As we discussed above, the total production cost will remain the same. The inventory storage will change.

The amount of product\(_2\) built up will be \( m_2 \cdot d \cdot \kappa'_\text{dis} \). And the inventory storage cost will be:

\[
C_{\text{dis}}^{(\text{product}_2)} = a_x q_x(N_2) m_{\text{dis}} + a_x q_x(N_2) m_2 \cdot d \cdot \frac{\kappa'_\text{dis} + 1) \kappa'_\text{dis}}{2}.
\]

The total cost is

\[
C_{\text{dis}}^{(\text{product}_2)} = C_{\text{nominal}} + \kappa'_\text{dis} \cdot C_{\text{nominal}}
\]

\[
+ a_x q_x(N_2) m_2 \cdot d \frac{\kappa'_\text{dis} + 1) \kappa'_\text{dis}}{2}.
\]

**Summary**

We can conclude the results in this way:

- To deal with the disruption, the system needs to build up product\(_2\) or product\(_3\). To choose which one depends on their inventory storage cost respectively, which are \( q_x(N_2) m_2 \) and \( q_x(N_1) \). The system will choose the cheaper one to build up.
• If building up product 1, the operations of $N_1$, $N_2$ and $N_3$ will increase to $d + d \cdot \kappa_{\text{dis}}'$ in the cycle right before disruption, and then decrease to 0 during the disruption. $N_0$ will keep the operation amount of $d$ until the $\kappa_{\text{dis}}'$-th cycle after disruption happens. The total cost increase is $a_x q_x(N_1)d\frac{(\kappa_{\text{dis}}'+1)\kappa_{\text{dis}}'}{2}$.

$$u(k) = \begin{cases} [d \ d \ d \ d]^T, & \text{when } k \leq k_{\text{dis}} - 2; \\ [d, \ d + d \cdot \kappa_{\text{dis}}', \ d + d \cdot \kappa_{\text{dis}}', \ d + d \cdot \kappa_{\text{dis}}']^T, & \text{when } k = k_{\text{dis}} - 1; \\ [d \ 0 \ 0 \ 0]^T, & \text{when } k_{\text{dis}} \leq k \leq k_{\text{dis}} + \kappa_{\text{dis}}' - 1. \end{cases}$$

• If building up product 2, the operations of $N_3$ and $N_2$ will increase to $d + d \cdot \kappa_{\text{dis}}'$ in the cycle right before disruption, and then decrease to 0 during the disruption. $N_1$ and $N_0$ will keep the operation amount of $d$ until the $\kappa_{\text{dis}}'$-th cycle after disruption happens. The total cost increase is $a_x q_x(N_2)m_2 \cdot d\frac{(\kappa_{\text{dis}}'+1)\kappa_{\text{dis}}'}{2}$.

$$u(k) = \begin{cases} [d \ d \ d \ d]^T, & \text{when } k \leq k_{\text{dis}} - 2; \\ [d \ d \ d + d \cdot \kappa_{\text{dis}}', \ d + d \cdot \kappa_{\text{dis}}', \ d + d \cdot \kappa_{\text{dis}}']^T, & \text{when } k = k_{\text{dis}} - 1; \\ [d \ d \ 0 \ 0]^T, & \text{when } k_{\text{dis}} \leq k \leq k_{\text{dis}} + \kappa_{\text{dis}}' - 1. \end{cases}$$

• If $q_x(N_2)m_2 = q_x(N_1)$, the system can build up either product 1 or product 2. The total cost will be identical for these two controls.

In order to achieve the lowest total cost, the system will choose inventory in one location to build up according to the storage cost. The amount is determined by the disruption duration. For the operations, related nodes will increase their work right before disruption to build up inventory, and during the disruption the necessary nodes will work to meet the demand.

It is very similar to analyze the situations when disruptions occur at other nodes.

3.2 System Description and Problem Formulation in Continuous Time Domain

In this section, we discuss the modeling and problem formulation in continuous time domain, which is quite similar as in discrete time domain. Many concepts are the same as in discrete time domain.

3.2.1 Network System Model

Nodes, Products and Operations

We use the same vector $b_i$ to denote the operation of node $i$. Also, Vector $x(t)$ with dimension $(n_p \times 1)$ represents the quantity of all $n_p$ products in the system at time $t$. The $i$-th element of $x(t)$, $x_i(t)$ stands for the quantity of product $i$ at time $t$.

Then the $(n_o \times 1)$ vector $u(t)$ represents the rate of operations at time $t$, such that the value of the element $u_i(t)$ is the rate of operations of node $i$. We assume there is no loss or spoilage of inventory. This then gives us the inventory update equation:
\[
\frac{dx(t)}{dt} = Bu(t), \text{ or } x(t_1) = x(t_0) + \int_{t_0}^{t_1} Bu(t)dt. \tag{3.10}
\]

In this study, we consider the operation rate \( u_i(t) \) without removable discontinuities, as shown in Figure 3.7. Jump discontinuities are allowed, but \( u_i(t) \) should satisfy that for any \( t \), \( u_i(t) \) equals to either \( \lim_{t' \to t^+} u_i(t') \) or \( \lim_{t' \to t^-} u_i(t') \), as shown in Figure 3.8.

![Figure 3.7: An example of removable discontinuities that are not allowed](image)

![Figure 3.8: Examples of jump discontinuities that are allowed](image)

Notice that we are using flow a model to represent the system, where the operation \( u(t) \) and inventory \( x(t) \) are real vectors. In this paper, the resilience problem will be formulated as an optimization problem over the real space. In practice, however, \( x(t) \) and \( u(t) \) often are restricted to be integers. Future work will consider effective ways to solve the problem in the integer case, such as adding a constraint of integer values.

**Assumption 6.** We assume that at initial time \( t = 0 \), each node has a nominal “running stock” inventory such that at time \( t = 0 \) each node is able to initiate production at the node’s full capacity. This “running stock” is not included in the inventory \( x \).

We use the same \( r_i \) and \( R \) as in discrete time domain to represent the resources usage by nodes in the system. And, we use the same \( D \) as in discrete time domain to extract the final node(s). We denote \( d(t) \) as the demand on the operation rate(s) of final node(s).

**Constraints**

Constraints are almost the same as in discrete time domain, except for that the variable of time is continuous.
No Backorder Constraint: \( (\forall t): x(t) \geq 0. \) \hspace{1cm} (3.11)

Capacity Constraint: \( (\forall t): Ru(t) \leq c(t). \) \hspace{1cm} (3.12)

Setup Constraint: \( (\forall t): \frac{du(t)}{dt} \leq \varsigma. \) \hspace{1cm} (3.13)

Demand Constraint: \( (\forall t): Du(t) \leq d(t). \) \hspace{1cm} (3.14)

Operation Direction Constraint: \( (\forall t): u(t) \geq 0. \) \hspace{1cm} (3.15)

Disruption

We still consider a disruption as a reduction on capacity. We use \( \bar{c} \) to denote the nominal capacity.

**Definition 2** (Disruption). A disruption for a resource \( i \) at disruption start time \( t_{\text{dis}} \) with duration \( \tau_{\text{dis}} \) is a reduction of capacity of that resource over the period: \( c_i(t) < \bar{c}_i \) for \( t_{\text{dis}} \leq t < t_{\text{dis}} + \tau_{\text{dis}} \), with \( c_i(t) = \bar{c}_i \) immediately prior to and immediately after the disruption period.

Cost

We still have the same four types of costs: product cost \( q_u u(t) dt \), inventory storage cost \( q_x x(t) dt \), changeover cost \( q_\varsigma du(t) \), and lost demand cost \( [d(t) - Du(t)] dt \).

System Model

The notation of the system Model is the same as in discrete time domain. Notice that the variable of \( c \) and \( d \) is continuous.

3.2.2 Nodes Description

The parameters of nodes are exactly the same as in discrete time domain. \( P(N_i) \) has the same definition as in Equation (3.7).
3.2.3 Problem Formulation

The objective function for determining our operation is formulated as a function of cost evaluated over a time window. Let $\tau_p$ and $\tau_f$ define look-back time period and a look-ahead time period, respectively. These define an observation window $[(t - \tau_p), (t + \tau_f)]$ around a time $t$ as illustrated in Figure 3.9. The following assumptions state that the duration of disruption is known upon first evidence of the future disruption within the observation window, and that disruptions are distant enough in time that only one disruption needs to be considered in the operations at any time.

![Figure 3.9: Time axis of observation window and disruption in continuous time domain](image)

**Assumption 7.** The interval between any two disruptions is larger than the look-ahead time window $\tau_f$. That is, denote $t_{\text{dis}, 1}$ and $t_{\text{dis}, 2}$ as the start time of any two disruptions, and $\tau_{\text{dis}, 1}$ as the duration of the first disruption. Then, $t_{\text{dis}, 1} + \tau_{\text{dis}, 1} + \tau_f < t_{\text{dis}, 2}$.

Then, similar as in discrete time domain, our optimization problem can be formulated as follows:

Minimize : $C = \int_{t - \tau_p}^{t + \tau_f} \left\{ a_u q_u u(t') + a_x q_x x(t') + a_d [d(t') - D u(t')] + a_\varsigma q_\varsigma \frac{u(t')}{dt'} \right\} dt'$,

Subject to : $x(t) \geq 0,$
$R u(t) \leq c(t),$
$\frac{d u(t)}{dt} \leq \varsigma,$
$D u(t) \leq d(t),$ $u(t) \geq 0,$
$\frac{d x(t)}{dt} = B u(t),$ $t > \tau_p,$ $t \in \mathbb{R},$ $\tau_p, \tau_f \in \mathbb{R}^+.$

In order to simplify the analysis in continuous time domain, we still introduce some assumptions similar as in discrete time domain. Assumptions [1] and [2] still hold.

Assumption [3] is replaced by:
**Assumption 8.** Assume \( \varsigma \gg \frac{du(t)}{dt} \) \( \forall t \), which means changeover constraint is so loose that at any time the increase of operations is not larger than the limit.

In continuous time domain, we also consider disruptive events with some prior warning. Such events could include plant shutdowns, maintenance events, weather events, and some types of supply disruptions. Assumption 4 is replaced by:

**Assumption 9.** For a given disruption, the duration of the disruption, \( \tau_{\text{dis}} \) is known at the time \( t_{\text{dis}} - \tau_f \) when the disruption is first evident within the observation window.

In continuous time domain, we consider capacity limit in Nominal State. Assumption 5 is replaced by:

**Assumption 10.** Assume \( c_i(t) = 0 \) when \( t \in [t_{\text{dis}}, t_{\text{dis}} + \tau_{\text{dis}}] \) and resource \( i \) is disrupted, which means the amount of disrupted resource is zero in disruption state. We use \( R_i \) to denote the matrix mapping \( u(t) \) to resource \( i \).

With Assumption 9, if a disruption is within the current look-ahead window or the current time is within a disruption period, then the time window extends from the initial disruption detection time \( (t_{\text{dis}} - \tau_f) \) to the end of the disruption \( (t_{\text{dis}} + \tau_{\text{dis}}) \). Otherwise, the objective function evaluation window is the observation window interval, \( [(t - \tau_p), (t + \tau_f)] \).

The optimal control problem is then formulated as follows, with a second priority objective function that is a concern only after the first priority objective function is met.

Minimize :  
1st objective: \[ \int_{T_1}^{T_2} a_d[d(t') - Du(t')]dt', \]
2nd objective: \[ \int_{T_1}^{T_2} [a_u u(t') + a_x x(t')]dt', \]

where \( T_1 = \left\{ \begin{array}{ll} t_{\text{dis}} - \tau_f, & \text{if } t_{\text{dis}} - \tau_f \leq t \leq t_{\text{dis}} + \tau_{\text{dis}} \\ t - \tau_p, & \text{otherwise} \end{array} \right. \)
and \( T_2 = \left\{ \begin{array}{ll} t_{\text{dis}} + \tau_{\text{dis}}, & \text{if } t_{\text{dis}} - \tau_f \leq t \leq t_{\text{dis}} + \tau_{\text{dis}} \\ t + \tau_f, & \text{otherwise} \end{array} \right. \)

Subject to :  
\( x(t) \geq 0, \)
\( Ru(t) \leq c(t), \)
\( Du(t) \leq d(t), \)
\( u(t) \geq 0, \)
\( \frac{dx(t)}{dt} = Bu(t), \)
\( t > \tau_p, \)
\( t \in \mathbb{R}, \)
\( \tau_p, \tau_f \in \mathbb{R}^+. \)

The 1st objective is the summation of all the lost demands, where the elements in vector \( a_d \) are the weight coefficients. If there is no specific requirement, the elements in \( a_d \) can be all set to ones to simply sum up the different lost demands. Also, \( a_u \) and \( a_x \) are set to one

26
by default. When \( t < 0 \), then \( x(t), u(t), c(t) \) and \( d(t) \) do not have physical meaning. We let them to be 0 when \( t < 0 \).

Also, Equation \( (3.9) \) still holds under the assumptions to represent the simplified node parameters in continuous time domain.

### 3.3 Conclusion

After reviewing the previous studies related to resilient enterprise, we found there is a need to set up a mathematical tool to analyze resilience. In order to study the effect of disruptive events on an enterprise, we presented a model of an enterprise that captures production, demand, capacity, changeover limits, and costs. Disruptions are represented by a sudden change of capacity over time in the model. We formulated the problem as an optimization issue, of which the objective function is the total cost over a time period. Our purpose is to figure out the optimal control for the system which can minimize the objective cost.

In this chapter, we introduced models in both discrete and continuous time domain. At the very beginning, we only set up discrete time model. The reason why we developed continuous time model as well is that it is more easy than the discrete time model to analyze the issues when considering capacity constraints. For these two models, the main differences are in time variables and Assumptions 5 and 10 which deactivate and activate the capacity constraints in \textit{Nominal State}, respectively. In the rest of this dissertation, Chapter 6 will discuss the analysis based on discrete time model; Chapters 7, 8 and 9 will discuss the analysis based on continuous time model.
Chapter 4

Definitions of Structures and Systems

In this chapter, we will define several different types of structures and systems based on the model we developed. These typical structures and systems can be considered as building blocks of more complicated network systems.

Consider a node $N_j$ and its column vector $b_j$ in the matrix $B$. Denote the $i$-th element in $b_j$ as $b_j(i)$. Recall that for a node $N_j$ of which the operation is represented by a vector $b_j$ in matrix $B$, $b_j(i) < 0$ indicates that product $i$ is consumed in the operation, and $b_j(i) > 0$ indicates that product $i$ is produced in the operation.

Define the ordering relation $\succ$ over the set of nodes such that $N_i \succ N_j$ if $N_i$ is upstream of $N_j$, i.e. product flows from $N_i$ to $N_j$, perhaps through intermediate nodes. We denote $N_i \succeq N_j$ if $N_i$ is upstream of $N_j$, or $N_i$ and $N_j$ are the same node.

4.1 Structures

A structure is a set of nodes related to each other based on their products. If a node $N_j$ produces product $i$ and $N_l$ consumes product $i$, then $N_j$ and $N_l$ are related to each other based on product $i$. In a structure, each node is related to some other node within the set. The relations all together show some pattern, which determines the type of the structure.

4.1.1 Chains

Definition 3. Given a sequence of nodes $(N_l, N_{l-1}, ..., N_1)$, for each index $j \in \{1, 2, ..., l-1\}$, if there exists an index $i$ such that

$$b_j(i) < 0 \text{ and } b_{j+1}(i) > 0,$$

then this sequence $(N_l, N_{l-1}, ..., N_1)$ forms a chain.
4.1.2 Serial Structures

**Definition 4.** For a set of nodes \( \{N_l, N_{l-1}, ..., N_1\} \), if the following statements are true:

1. \((N_l, N_{l-1}, ..., N_1)\) is a chain.
2. Any \(N_i \in \{N_l, N_{l-1}, ..., N_2\}\) produces only one type of product and feeds to only \(N_{i-1}\).
3. Any \(N_i \in \{N_{l-1}, N_{l-2}, ..., N_1\}\) consumes only one type of product which is the output of \(N_{i+1}\).
4. Within or outside the set \(\{N_l, N_{l-1}, ..., N_1\}\), there doesn’t exist a chain started at \(N_1\) and ended at \(N_i\).

Then, \(\{N_l, N_{l-1}, ..., N_1\}\) forms a serial structure.

An example of serial structure is shown by Figure 4.1.

![Figure 4.1: An example of the serial structure](image)

4.1.3 AND Structures

**Definition 5** (AND structure). For a set of nodes \(\{N_0, N_1, N_2, ..., N_l\}\), if

1. Any \(N_i \in \{N_1, N_2, ..., N_l\}\) produces only one type of product, and feeds to only \(N_0\), and
2. The product of any \(N_i \in \{N_1, N_2, ..., N_l\}\) is different from each other, and
3. \(N_0\) consumes products only from \(\{N_1, N_2, ..., N_l\}\), and
4. Within or outside the set \(\{N_0, N_1, ..., N_l\}\), there doesn’t exist a chain started at \(N_0\) and ended at \(N_i \in \{N_1, N_2, ..., N_l\}\).

Then the set of nodes \(\{N_0, N_1, N_2, ..., N_l\}\) are defined as the AND structure.

An example of the AND structure is shown by Figure 4.2.

![Figure 4.2: An example of the AND structure](image)
4.1.4 OR Structures

Definition 6 (OR structure). For a set of nodes \( \{N_1, N_2, ..., N_l\} \), if

1. each node \( N_i \) all produce the same output product \( \text{out} \) and have no input product, and
2. all of product \( \text{out} \) produced by any \( N_i \) are fed as input to the same node \( N_0 \), and
3. product \( \text{out} \) is the only input of \( N_0 \),

then the set of nodes \( \{N_1, N_2, ..., N_l\} \) are defined as the OR structure.

So the downstream node can select any of the source nodes to complete its production, although different selection can bring different costs. Figure 4.3 illustrates the OR nodes.

![Figure 4.3: An example of the OR structure](image)

4.1.5 Assembly Tree Structures

Definition 7. For a set of nodes \( \{N_0, N_1, ..., N_l\} \), the dimension of each vector \( b_i \), \( b_i \in \{b_0, b_1, ..., b_l\} \), is \( n_p \). If the following statements are true:

1. **Only one root node:**
   There exists a unique node (destination root) in \( \{N_0, N_1, ..., N_l\} \), which is denoted as \( N_0 \). \( b_0 \) satisfies that if there exists some index \( j \) such that \( b_0(j) > 0 \), then for each \( j \), \( [b_1(j), b_2(j), ..., b_m(j)] = 0 \), and there exists a node \( N_{(0,j)} \notin \{N_0, N_1, ..., N_l\} \) such that \( b_{(0,j)}(j) < 0 \).

2. **Source for each product is within the set and is unique:**
   For each \( b_n \in \{b_0, b_1, ..., b_l\} \) and each index \( i \) such that \( b_n(i) < 0 \), there exists only one node \( b_r \in \{b_1, b_2, ..., b_l\} \) such that \( b_r(i) > 0 \), and there does not exist a node \( b_p \notin \{b_1, b_2, ..., b_l\} \) such that \( b_p(i) > 0 \).
3. **Single output per node except the root**:  
Each vector of \{b_1, b_2, ..., b_l\} has only one positive element.

4. **Each product, except the root, only feeds one node in this set**:  
For each \(b_n \in \{b_1, b_2, ..., b_l\}\) and its index \(i\) such that \(b_n(i) > 0\), there exists only one node \(b_r \in \{b_0, b_1, ..., b_l\}\) such that \(b_r(i) < 0\), and there does not exist a node \(b_p \notin \{b_0, b_1, ..., b_l\}\) such that \(b_p(i) < 0\).

5. **No cycles**:  
For any set \(\{N_1, N_2, ..., N_n\} \subseteq \{N_0, N_1, ..., N_l\}\), there does not exist such a chain \((N_1, N_2, ..., N_n, N_1)\).

Then, the set \(\{N_0, N_1, ..., N_l\}\) is an assembly tree structure, with a destination root node of \(N_0\).

Recall the system shown by Figure 3.1 at the very beginning of Chapter 3. In that system, nodes \(\{N_{13}, N_{12}, N_{11}, N_7, N_6, N_3\}\) form an assembly tree structure, with \(N_3\) as its root node. \(\{N_{10}, N_9, N_8, N_5\}\) is not an assembly tree structure, since \(N_9\) and \(N_8\) produce the same product, which does not satisfy statement 2 in Definition 7.

Based on Definition 7, if \(N_0\) has any output product denoted as product \(j\), then \(N_0\) is the output node of the set of nodes, and product \(j\) is the output inventory of the set of nodes. Each node in an assembly tree structure has only one output. Each input of a node comes from a single node, although there could be multiple inputs. There are no input nodes outside this set. Except product \(j\), there are no other types of inventory going out of this set.

**Lemma 1.** For any two nodes \(N_n\) and \(N_r\) in an assembly tree structure, if there exists a chain beginning with \(N_n\) and ending with \(N_r\), then

1. all the nodes in this chain are in the assembly tree structure, and
2. this chain is unique.

**Proof.** We denote this chain as \((N_n, N_{n_1}, N_{n_2}, ..., N_r)\).

1. All the nodes in this chain are in the tree.

We prove this by contradiction. Denote \(j\) as the smallest index such that \(N_{n_j}\) is not in the tree. Then \(N_{n_{j-1}}\) is in the tree. Consider the index \(i\) such that \(b_{n_{j-1}}(i) > 0\). According to Definition 3, we have \(b_{n_j}(i) < 0\). Since \(N_{n_j}\) is not in the tree, \(N_{n_{j-1}}\) does not satisfy the statement in Definition 7. This is a contradiction.

2. This chain is unique.

We prove this by contradiction. Denote \((N_n, N_{n_1}', N_{n_2}', ..., N_r)\) as another chain starting with \(N_n\) and ending with \(N_r\). Then there exists a smallest \(j\) such that \(n_j' \neq n_j\). Therefore \(n_{j-1}' = n_{j-1}\). Consider the index \(i\) such that \(b_{n_{j-1}}(i) > 0\). According to Definition 7, \(i\) is the only index which satisfies \(b_{n_{j-1}}(i) > 0\). Then, according to Definition 3, we have \(b_{n_j}(i) < 0\), and \(b_{n_j'}(i) < 0\).
Since \( n_j \neq n_j' \), this does not satisfy Statement 4 in Definition[7] which is a contradiction. Therefore, for chain \((N_n, N_{n_1}, N_{n_2}, ..., N_r)\), all the nodes in this chain are in the tree, and this chain is unique.

\[ \square \]

**Lemma 2.** If \( \{N_0, N_1, ..., N_l\} \) is an assembly tree structure and \( N_0 \) is the root node, then for each \( N_i \in \{N_1, N_2, ..., N_l\} \), there exists a chain among the nodes within \( \{N_0, N_1, ..., N_l\} \) that begins with \( N_i \) and ends with \( N_0 \).

**Proof.** We prove this lemma by contradiction. Suppose there exists a number of nodes which are not in chains ended with \( N_0 \). We denote the set of all such nodes as \( \{N_1, N_2, ..., N_l\} \), and \( n \leq l \). Therefore, for any \( N_i \in \{N_1, N_2, ..., N_l\} \) and any \( N_j \in \{N_{n+1}, ..., N(l), N_0\} \), \( N_i \) can not be the input of \( N_j \); otherwise, \( N_i \) is in the chain of \( N_j \) ended with \( N_0 \), such as \((N_i, N_j, ..., N_0)\). Thus, there does not exist an index \( j \) such that

\[ b_{i_1}(j) > 0, \text{ and } b_{i_2}(j) < 0. \quad (4.1) \]

According to Definition[7] there exists a node \( N_{i_1,1} \) in the tree such that for some \( j \)

\[ b_{i_1}(j) > 0, \text{ and } b_{i_1,1}(j) < 0. \]

\[ N_{i_1,1} \notin \{N_{n+1}, ..., N(l), N_0\}, \]

\[ N_{i_1,1} \in \{N_1, ..., N_l\}. \]

Similarly, there exist some nodes \( N_{i_1,r} \) in the tree, where \( r \in \{2, 3, ..., p\} \), such that for some \( j_r \)

\[ b_{i_1,r-1}(j_r) > 0, \text{ and } b_{i_1,r}(j_r) < 0, \]

\[ N_{i_1,r} \in \{N_1, ..., N_l\}. \]

Hence, \((N_{i_1,1}, N_{i_1,2}, ..., N_{i_1,p})\) is a chain. Since there is no circle in the tree, any node in \((N_{i_1,1}, N_{i_1,2}, ..., N_{i_1,p})\) is different from others. Since \( \{N_{i_1,1}, N_{i_1,2}, ..., N_{i_1,p}\} \subseteq \{N_1, N_2, ..., N_{n}\} \), \( p \) is a finite number. We denote \( p_{\text{max}} \) as the maximum value of \( p \). There exists a node \( N_{i_1,p_{\text{max}}+1} \) in the tree such that for some \( j \)

\[ b_{i_1,p_{\text{max}}}(j) > 0, \text{ and } b_{i_1,p_{\text{max}}+1}(j) < 0, \]

\[ N_{i_1,p_{\text{max}}+1} \in \{N_1, ..., N(n)\}. \]

Thus, \((N_{i_1,1}, N_{i_1,2}, ..., N_{i_1,p_{\text{max}}}, N_{i_1,p_{\text{max}}+1})\) is also a chain. Therefore, the maximum value of \( p \) is not \( p_{\text{max}} \). This is a contradiction. Thus, each of \( \{N_1, N_2, ..., N_l\} \) is in a chain ended with \( N_0 \). \( \square \)

The assembly tree structure has an important property that the output of the set of nodes can only be at the root node. Except the root node, nodes in an assembly tree structure have no connections with any node outside the structure. When there is no buildup of product within the structure, we can consider the whole structure as a single node, of which the output is the output of the root node. This is useful to simplify the system, which will be discussed in detail in later chapters. A ratio of operations of nodes of an assembly tree structure can be determined when no product is built up within the structure. Such a ratio is unique, which is presented by the following theorem.
Theorem 1. For a given positive constant scalar $d$, if \( \{N_0, N_1, \ldots, N_l\} \) are nodes of an assembly tree structure, then there exists a unique vector $\alpha = [\alpha_0, \alpha_1, \ldots, \alpha_l]^T$, where $\alpha_i > 0$ for any $i$ and $\alpha_0 = d$, such that

$$b = \alpha_0 b_0 + \alpha_1 b_1 + \ldots + \alpha_l b_l,$$

$$b(i) = \begin{cases} 0, & \text{for each } i \text{ such that } b_0(i) \leq 0, \\ db_0(i), & \text{for each } i \text{ such that } b_0(i) > 0. \end{cases} \tag{4.2}$$

Proof. We denote product $\prod_i$ as the product produced by $N_i$, where $i \in \{1, 2, \ldots, l\}$. Then, $b_i(i) > 0$.

From Definition 7, $[b_1(i), b_2(i), \ldots, b_l(i)] = 0$, $\forall i \notin \{1, 2, \ldots, l\}$. Thus,

$$b(i) = [b_0(i), b_1(i), b_2(i), \ldots, b_l(i)][\alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_l]^T = b_0(i)\alpha_0 \geq 0.$$  

It follows that,

$$b(i) = [b_0(i), b_1(i), \ldots, b_l(i)][\alpha_0, \alpha_1, \ldots, \alpha_l]^T = 0, \text{ when } i \notin \{1, 2, \ldots, l\} \text{ and } b_0(i) = 0. \tag{4.3}$$

Consider each index $i$ such that $b_0(i) > 0$. According to Definition 7 we have

$$i \notin \{1, 2, \ldots, l\},$$

$$[b_1(i), b_2(i), \ldots, b_l(i)] = 0.$$

$$\Rightarrow [b_1(i), b_2(i), \ldots, b_l(i)][\alpha_1, \alpha_2, \ldots, \alpha_l]^T = 0.$$

$$\Rightarrow b(i) = [b_1(i), b_2(i), \ldots, b_l(i)][\alpha_1, \alpha_2, \ldots, \alpha_l]^T + b_0(i)\alpha_0 = b_0(i)\alpha_0.$$

Since $\alpha_0 = d$, then

$$b(i) = db_0(i), \text{ for each } i \text{ such that } b_0(i) > 0. \tag{4.4}$$

According to Lemma 2, for each node $N_{i_1} \in \{N_1, N_2, \ldots, N_l\}$, there is some chain starts with $N_{i_1}$ and ends with $N_0$. We denote it as $(N_{i_1}, \ldots, N_{i_r}, N_0)$. According to Definition 7 $\{N_{i_1}, \ldots, N_{i_r}, N_0\} \subseteq \{N_0, N_1, \ldots, N_l\}$. Thus,

$$b_{i_1}(l_r) > 0,$$

$$b_0(l_r) < 0.$$

$$b_{i_r}(l_r) = 0, \forall i \notin \{l_r, 0\}.$$

$$\Rightarrow b(l_r) = b_{i_r}(l_r)\alpha_{i_r} + b_0(l_r)\alpha_0.$$

Then we select

$$\frac{\alpha_{i_r}}{\alpha_0} = -\frac{b_0(l_r)}{b_{i_r}(l_r)}.$$

Therefore, $b(l_r) = 0$.

Similarly, we have

$$\frac{\alpha_{i_r}}{\alpha_{i_{r+1}}} = -\frac{b_{i_{r+1}}(l_i)}{b_{i_r}(l_i)} \forall i \in \{1, 2, \ldots, r - 1\}.$$
Since \( \alpha_0 = d \),
\[
\alpha_i = d \sum_{i'=1}^{r-1} \left( \frac{-b_{i,i+1}(l_{i'})}{b_{i,i}(l_{i'})} \right) \left( \frac{-b_0(l_r)}{b_i(l_r)} \right).
\]

According to Lemma 1, \( \alpha_i \) exists and is unique, and does not conflict with other chains. Therefore, by such unique \( \alpha_i \), we have
\[
\begin{align*}
\alpha_i(l_i)\alpha_i + b_{i+1}(l_i)\alpha_{i+1} &= 0, \\
\alpha_i(l_r)\alpha_i + b_0(l_r)\alpha_0 &= 0.
\end{align*}
\]

Then,
\[
[b_{i_1}(l_{i_1}), b_{i_2}(l_{i_2}), ..., b_{i_r}(l_{i_r}), b_0(l_i)] \begin{bmatrix} \alpha_{i_1} & \alpha_{i_2} & ... & \alpha_{i_r} & \alpha_0 \end{bmatrix} = 0, \quad \forall i \in \{1, 2, ..., r\}.
\]

Since \( N_i \) can be any node in \( \{N_0, N_1, ..., N_l\} \), then
\[
[b_0(i), b_1(i), b_2(i), ..., b_l(i)] \begin{bmatrix} \alpha_0 & \alpha_1 & ... & \alpha_l \end{bmatrix} = 0, \quad \forall i \in \{1, 2, ..., l\}. \tag{4.5}
\]

To sum up Equations (4.3), (4.4) and (4.5), there exists a unique \( \alpha \) such that
\[
[b_0(i), b_1(i), ..., b_l(i)] \begin{bmatrix} \alpha_0 & \alpha_1 & ... & \alpha_l \end{bmatrix} = \begin{cases} 0, & \text{when } i \notin \{1, 2, ..., l\} \text{ and } b_0(i) = 0, \\
|db_0(i)|, & \text{when } b_0(i) > 0, \\
0, & \text{when } i \in \{1, 2, ..., l\}. \end{cases}
\]

According to Definition 7 if index \( i \) satisfies \( b_0(i) > 0 \), then \( i \notin \{1, 2, ..., l\} \); if index \( j \) satisfies \( b_0(j) < 0 \), then \( j \notin \{1, 2, ..., l\} \). Therefore, the statement above can be expressed as that there exists a unique \( \alpha \) such that
\[
b(i) = [b_0(i), b_1(i), ..., b_l(i)] \begin{bmatrix} \alpha_0 & \alpha_1 & ... & \alpha_l \end{bmatrix} = \begin{cases} 0, & \text{for each } i \text{ such that } b_0(i) \leq 0, \\
|db_0(i)|, & \text{for each } i \text{ such that } b_0(i) > 0. \end{cases}
\]

**Theorem 2.** If \( \{N_0, N_1, ..., N_l\} \) is an assembly tree structure, then vectors \( \{b_1, b_2, ..., b_l\} \) are linearly independent.

**Proof.** We prove this by contradiction. If \( \{b_1, b_2, ..., b_l\} \) are linearly dependent. there exists a vector \( \beta \neq 0 \) such that:
\[
[b_1, b_2, ..., b_l] \beta = 0.
\]

Consider a set \( \{\alpha_{a_1}, \alpha_{a_2}, ..., \alpha_{a_l}\} \), where \( \alpha_i \) is the parameters in Theorem 1. Denote \( |\frac{\alpha_i}{\beta_i}| = \min\{|\frac{\alpha_1}{\beta_1}|, |\frac{\alpha_2}{\beta_2}|, ..., |\frac{\alpha_l}{\beta_l}|\} \). For any constant \( 0 < \lambda < |\frac{\alpha_i}{\beta_i}| \),
\[
\alpha_i + \lambda \beta_i = |\frac{\alpha_i}{\beta_i}||\beta_i| + \lambda \beta_i \geq |\frac{\alpha_n}{\beta_n}||\beta_i| + \lambda \beta_i > \lambda |\beta_i| + \lambda \beta_i \geq 0.
\]

Therefore, we can select another vector \( \alpha' \) such that
\[
\alpha' = \alpha + \lambda \begin{bmatrix} \beta \\ 0 \end{bmatrix}.
\]

Thus, \( \alpha' > 0 \) and
\[
[b_0, b_1, ..., b_l] \alpha = [b_0, b_1, ..., b_l] \alpha' = b.
\]

Therefore, \( \alpha \) is not unique. This does not satisfy Theorem 1 Therefore, \( \{b_1, b_2, ..., b_l\} \) are linearly independent. \( \square \)
An assembly tree structure can be considered as consisting of several subtrees, which are also assembly tree structures as per Definition [7]. For each node $N_i$ in an assembly tree structure, we can locate a subtree of which $N_i$ is the root node. This property is presented by the following theorem.

**Theorem 3.** Assume $\{N_0, N_1, ..., N_l\}$ are nodes of assembly tree structure. For any node $N_i \in \{N_0, N_1, ..., N_l\}$, if a set of nodes $\mathcal{N}_i = \{N_i,1, N_i,2, ..., N_i,l\}$ satisfies:

1. $\mathcal{N}_i \subseteq \{N_0, N_1, ..., N_l\}$, and
2. for each node $N_n \in \mathcal{N}_i$ there exists some chain among the nodes within $\{N_i\} \cup \mathcal{N}_i$ that begins with $N_n$ and ends with $N_i$, and
3. for each node $N_r \notin \mathcal{N}_i$ there does not exist any chain among the nodes within $\{N_0, N_1, ..., N_l\}$ that begins with $N_r$ and ends with $N_i$,

then $\mathcal{N}_i \cup \{N_i\}$ forms an assembly tree structure, and $N_i$ is the root node.

**Proof.** $\{b_0, b_1, ..., b_l\}$ satisfies Definition [7]. With the conditions in Theorem 3 we prove $\{N_i,1, N_i,2, ..., N_i,l\}$ satisfies Definition [7].

Since each node among $\{N_i,1, N_i,2, ..., N_i,l\}$ is in some chain ends with $N_i$, and $N_0$ is not in any chain ends with $N_i$, we know $N_0 \notin \{N_i,1, N_i,2, ..., N_i,l\}$. Therefore $\{N_i,1, N_i,2, ..., N_i,l\} \subseteq \{N_1, N_2, ..., N_l\}$.

1. We prove $N_i$ is the root of $\{N_i,1, N_i,2, ..., N_i,l\} \cup \{N_i\}$.

   If $N_i$ is $N_0$, then $b_i$ is $b_0$. Thus $b_i$ satisfies that if there exists some index $j$ such that $b_i(j) = b_0(j) > 0$, then for each $j$, $[b_1(j), b_2(j), ..., b_l(j)] = 0$, and there exists a node $N_{(0,j)} \notin \{N_0, N_1, ..., N_l\}$ such that $b_{(0,j)}(j) < 0$. Since $\{N_i,1, N_i,2, ..., N_i,l\} \subseteq \{N_1, N_2, ..., N_l\}$, for each $j$, $[b_{i,1}(j), b_{i,2}(j), ..., b_{i,l}(j)] = 0$, and $N_{(0,j)} \notin \{N_i,1, N_i,2, ..., N_i,l\}$.

   If $N_i$ is not $N_0$, then there exists only one index $j$ such that $b_i(j) > 0$. According to Definition [7] there exists only one node $b_r \in \{b_0, b_1, ..., b_l\}$ such that $b_r(j) < 0$.

   (a) We need to prove $[b_{i,1}(j), b_{i,2}(j), ..., b_{i,l}(j)] = 0$. We proof this by contradiction.

   If there exists a $b_n \in \{b_{i,1}, b_{i,2}, ..., b_{i,l}\}$ such that $b_n(j) \neq 0$, then either $b_n(j) > 0$ or $b_n(j) < 0$.

   If $b_n(j) > 0$, then for $br$ and $j$, there exists two node $b_i$ and $b_n$ such that $b_i(j) > 0$ and $b_n(j) > 0$, which does not satisfy Definition [7]. Therefore $b_n(j) < 0$, which means $N_n$ is $N_r$.

   Since $N_n$ is in a chain ends with $N_i$, which is denoted as $(N_n, N_{(n+1)}, ..., N_i)$, then $(N_i, N_n, N_{(n+1)}, ..., N_i)$ also forms a chain. $(N_i, N_n, N_{(n+1)}, ..., N_i)$ is an endless chain or a circle. Consider the chain $(N_n, N_{(n+1)}, ..., N_0)$, for $N_n$ and index $m$ such that $b_n(m) > 0$, there exists only one node such that $b_n(m) < 0$. Therefore $N_n$, $N_{(n+1)}$ and $N_{(n'+1)}$ are the same node. Thus, $N_{(n'+1)} \in (N_i, N_n, N_{(n+1)}, ..., N_i)$. Similarly, each node in $(N_n, N_{(n+1)}, ..., N_0)$ is in $(N_i, N_n, N_{(n+1)}, ..., N_i)$. Then, $N_0 \in \{N_n, N_{(n+1)}, ..., N_i\} \subseteq \{N_i,1, N_i,2, ..., N_i,l\}$, which is a contradiction. Therefore, $[b_{i,1}(j), b_{i,2}(j), ..., b_{i,l}(j)] = 0$.
(b) We need to prove \( N_r \notin \{N_{i,1}, N_{i,2}, ..., N_{i,l_i}\} \). Since \([b_{i,1}(j), b_{i,2}(j), ..., b_{i,l_i}(j)] = 0, \) and \( b_r(j) < 0 \), we know \( b_r \notin \{b_{i,1}, b_{i,2}, ..., b_{i,l_i}\} \). Therefore \( N_r \notin \{N_{i,1}, N_{i,2}, ..., N_{i,l_i}\} \).

Thus, \( N_i \) is the root of \( \{N_{i,1}, N_{i,2}, ..., N_{i,l_i}\} \cup \{N_i\} \).

2. According to Definition 7 for each \( b_n \in \{b_{i,1}, b_{i,2}, ..., b_{i,l_i}\} \) and each index \( m \) such that \( b_n(m) > 0 \), there exists only one node \( b_r \in \{b_0, b_1, ..., b_l\} \) such that \( b_r(m) > 0 \). Since \( b_n \) is in some chain ends with \( b_i \), \( b_r \) is also in some chain ends with \( b_i \). Hence, \( b_r \in \{b_{i,1}, b_{i,2}, ..., b_{i,l_i}\} \). Since \( b_r \) is the only one such that \( b_r(m) > 0 \), there does not exist a node \( b_p \notin \{b_{i,1}, b_{i,2}, ..., b_{i,l_i}\} \) such that \( b_p(m) > 0 \).

3. Since each vector of \( \{b_1, b_2, ..., b_l\} \) has and only has one positive element, each vector of \( \{b_{i,1}, b_{i,2}, ..., b_{i,l_i}\} \) has and only has one positive element.

4. According to Definition 7 for each \( b_n \in \{b_{i,1}, b_{i,2}, ..., b_{i,l_i}\} \) and each index \( m \) such that \( b_n(m) > 0 \), there exists only one node \( b_r \in \{b_0, b_1, ..., b_l\} \) such that \( b_r(m) < 0 \). Since \( b_n \) is in some chain ends with \( b_i \), which can be denoted as \( (b_0, b_{(n+1)}, ..., b_i) \), then \( b_r \) is \( b_{(n+1)} \). Thus, \( b_r \in \{b_{i,1}, b_{i,2}, ..., b_{i,l_i}\} \). Since \( b_r \) is the only one such that \( b_r(m) < 0 \), there does not exist a node \( b_p \notin \{b_{i,1}, b_{i,2}, ..., b_{i,l_i}\} \) such that \( b_p(m) < 0 \).

5. For any set \( \{N_{(1)}, N_{(2)}, ..., N_{(n)}\} \subseteq \{N_{i,1}, N_{i,2}, ..., N_{i,l_i}, N_i\} \), since \( \{N_{(1)}, N_{(2)}, ..., N_{(n)}\} \subseteq \{N_0, N_1, ..., N_l\} \), there does not exist such a chain \( (N_{(1)}, N_{(2)}, ..., N_{(n)}, N_{(1)}) \).

Therefore, \( \{N_{i,1}, N_{i,2}, ..., N_{i,l_i}\} \cup \{N_i\} \) is an assembly tree structure with destination root of \( N_i \).

\( \square \)

### 4.2 Systems

In this section, we will introduce some systems with specific structures. On the one hand, such systems are specific structures; on the other hand, the definition of any system is more detailed than that of any structure. To determine a structure, we only consider the relations of nodes based on products. However, to determine a system with specific structures, we also consider the resource and demand. Moreover, a system is a complete set includes all the sources and consumers, which means no product coming into or going out of the system, while a structure can be only a subset of nodes of a complete system.

#### 4.2.1 Serial Network Systems

In this research, we consider serial network systems because they are common in the real world and they have a relatively simple structure. In this section, we will use our framework discussed to define the serial network systems.

We can conclude the definition of the serial network systems.

**Definition 8.** The serial network systems which we study in this work are those satisfying:
S1. There are \( l + 1 \) nodes \( \{N_l, N_{l-1}, \ldots, N_2, N_1, N_0\} \) in the serial structure, with indices decreasing downstream.

S2. Every node has no more than one output product and no more than one input product. For \( i \in \{1, 2, \ldots, l\} \), the output product of \( N_i \) is product \( i \). For \( i \in \{0, 1, 2, \ldots, l - 1\} \), the input product of \( N_i \) is product \( i+1 \).

S3. Each node requires only a single resource, and that resource is not required by any other node. That is, \( R \) is a diagonal matrix.

S4. The demand is only on the operation of the final node \( N_0 \).

S5. Denote \( N_{\text{dis}} \) as the node which has the disruption.

Because we restrict ourselves to a serial network system, the operation of each node has only a single incoming product and a single outgoing product, as per statement S2. For \( i \in \{0, 1, \ldots, l - 1\} \), let \( b_{i}^{\text{in}} \) be the \((i + 1, i + 1)\) element of matrix \( B \). For \( i \in \{1, 2, \ldots, l\} \), let \( b_{i}^{\text{out}} \) be the \((i, i + 1)\) element of \( B \). Then, \( b_{i}^{\text{in}} \) is negative, \( b_{i}^{\text{out}} \) is positive, and all other elements of \( b_i \) vector are zero.

Furthermore, the operation cost vector \( q_u \) can be represented as \([q_u(N_0), q_u(N_1), \ldots, q_u(N_l)]\), in which each element is a scalar representing the unit operation cost of one node. The storage cost vector \( q_x \) can be represented as \([q_x(N_1), q_x(N_2), \ldots, q_x(N_l)]\), in which each element is a scalar representing the unit storage cost of the product of one node. The resource matrix \( R \) can be represented as a diagonal matrix with elements \( r_0, r_1, \ldots, r_l \) on the diagonal. The capacity vector in the nominal state can be represented as \([c(N_0), c(N_1), \ldots, c(N_l)]^T\).

4.2.2 Assembly Tree Network Systems

**Definition 9.** For a system of nodes \( \{N_0, N_2, \ldots, N_l\} \), if

T1. It is an assembly tree structure. The set \( \{N_0, N_1, \ldots, N_l\} \) is an assembly tree structure, with the destination root node of \( N_0 \), and

T2. There is no output from \( N_0 \). There does not exist an index \( i \) such that \( b_0(i) > 0 \), and

T3. Each node requires only a single resource, and that resource is not required by any other node. That is, \( R \) is a diagonal matrix, and

T4. The demand is only on the operation of the destination root node \( N_0 \), and

T5. Denote \( N_{\text{dis}} \) as the node which has the disruption,

then the system of nodes \( \{N_0, N_1, \ldots, N_l\} \) is an assembly tree network system, with \( N_0 \) being the destination root node.

Since any \( N_i \) except \( N_0 \) produces a single type of products, we denote product \( i \) as the product of \( N_i \). Then \( b_i(i) \) is the \((i, i + 1)\) element in \( B \). Denote \( b_i^{\text{out}} \) as \( b_i(i) \). Thus, \( b_i^{\text{out}} \) is positive.
We use the same notations to denote the parameters of assembly tree network systems as of serial network systems. The operation cost is \([q_u(N_0), q_u(N_1), \ldots, q_u(N_l)]\). The storage cost vector is \([q_x(N_1), q_x(N_2), \ldots, q_x(N_l)]\). The resource matrix \(R\) is a diagonal matrix with elements \(r_0, r_1, \ldots, r_l\) on the diagonal. The capacity vector in the nominal state is \([c(N_0), c(N_1), \ldots, c(N_l)]^T\). The demand matrix \(D\) is \(D = [1, 0, \ldots, 0]\).

### 4.3 Conclusion

We defined several basic types of structures and network systems, which commonly appear in enterprise systems and will be studied in the later research. These include chains, serial structures, AND structures, OR structures, assemble tree structures, serial network systems and assembly tree network systems.
Chapter 5

Normalization and Aggregation

The analysis in Section 3.1.4 shows that the approach for simple systems is available. We want to apply these results of simple systems in the analysis of more complex systems. We need some method to simplify the complex systems to simple ones. In this chapter, we will introduce the methods to simplify complex systems.

5.1 Normalization of Serial Network Systems and Assembly Tree Network Systems

Recall the analysis of a simple system in discrete time domain in Chapter 3. We determined two cases of optimal operations based on the comparison of storage cost of product 2 and product 1. In that analysis, we didn’t compare \( q_x(N_2) \) and \( q_x(N_1) \) directly. Instead, we compared \( q_x(N_2)m_2 \) and \( q_x(N_1) \). This is because for a unit amount of demand, the required amounts of product 2 and product 1 are different. For a unit amount of demand, the system requires one unit of product 1, or \( m_2 \) units of product 2.

Such kinds of comparisons appear in the analysis of general serial network systems and assembly tree network systems. When considering the amount and cost of storage of any product, it is more convenient to associate them with the unit amount of demand.

Besides, for a serial network system or assembly tree network system, the only demand is on the final node or root node \( N_0 \), and there exists a constant vector \( \alpha \) such that \( B\alpha = 0 \). Therefore, the operation \( u \) of these systems can also be associated with demand.

Therefore, to simplify the analysis of these systems, we convert the parameters to be associated with the demand.

Define \( \alpha := [\alpha_0, \alpha_1, ..., \alpha_l]^T \) as a vector of nominal operation rate. For a serial network system, elements \( \alpha_i \) for \( N_i \) defined by:

\[
\alpha_0 := d, \quad \text{and} \quad \alpha_i := \alpha_{i-1} - \frac{b_i^{in}}{b_i^{out}}.
\]

For an assembly tree network system, \( \alpha \) is the same vector as in Theorem 1.
Thus, for both serial network systems and assembly tree network systems, we have $\alpha_0 := d$ and $B\alpha = 0$. When $u = \alpha$, then $Du = d$ and $Bu = 0$.

Then we denote its normalized operation as:

$$\tilde{u}_i := \frac{u_i}{\alpha_i}. \quad (5.1)$$

Thus, the operation rate at any time is presented as a multiplier or fraction of the nominal operation rate. The normalized operation rate is the multiplier on $\alpha_i$.

The normalized demand $\tilde{d}$ is defined as:

$$\tilde{d} := \frac{d}{\alpha_0} = 1.$$ 

The demand matrix $D$ stays the same, i.e.:

$$\tilde{D} = D,$$

and $\tilde{d}_i^D = d_i^D$.

For $i \in \{1, 2, ..., l\}$, the normalized capacity is defined by:

$$\tilde{c}(N_i) := \frac{c(N_i)}{r_i} \frac{1}{\alpha_i},$$

which indicates the maximum number of times of nominal-state operation can be carried out at $N_i$. Consider $N_0$. The Demand Constraint states $Du \leq d$, i.e., $u_0 \leq d$. From the Capacity Constraint $Ru \leq c$, we have $r_0u_0 \leq c_0$. As we assume the capacity of the final node is large enough to satisfy the demand, then the actual limit of $u_0(t)$ is $d$. Thus, $r_0u_0 \leq r_0d$. Hence, we define:

$$\tilde{c}(N_0) := \frac{r_0d}{r_0} \frac{1}{\alpha_0} = 1.$$ 

To sum up,

$$\tilde{c}(N_i) := \begin{cases} \frac{c(N_i)}{r_i} \frac{1}{\alpha_i}, & \text{when } i \in \{1, 2, ..., l\}, \\ 1, & \text{when } i = 0. \end{cases}$$ 

Then, the resource matrix satisfies:

$$\tilde{R} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}.$$ 

The normalized inventory is defined by:

$$\tilde{x}_i := \frac{x_i}{b_i^\text{out} \alpha_i}.$$ 

Then, the normalized operation vector $\tilde{b}_i$ satisfies the following:
1. $\tilde{b}_{i}^{out} = 1$ for $i \in \{1, 2, ..., l\}$, and

2. $\tilde{b}_{i}(j) = -1$ for $i \in \{0, 1, 2, ..., l\}$ and $j$ such that $b_{i}(j) < 0$, and

3. all other elements in $\tilde{b}_{i}$ are zeros for $i \in \{0, 1, 2, ..., l\}$.

The normalized unit storage cost is defined by:

$$\tilde{q}_{x}(N_{i}) := q_{x}(N_{i})b_{i}^{out} \alpha_{i},$$

so that the total inventory storage cost keeps the same:

$$\tilde{q}_{x}(N_{i})\tilde{x}_{i} = q_{x}(N_{i})x_{i}.$$ 

The normalized unit operation cost is defined by:

$$\tilde{q}_{u}(N_{i}) := q_{u}(N_{i})\alpha_{i},$$

so that the total operation cost keeps the same:

$$\tilde{q}_{u}(N_{i})\tilde{u}_{i} = q_{u}(N_{i})u_{i}.$$ 

### 5.2 Aggregation

Aggregation is one method to simplify the general systems. Aggregation is representing the behavior of a set of nodes with a single node, such that the overall system behavior is the same. With aggregation, we hope to make the research work more operable. Moreover, we hope it can help to take the problems and solutions to a more general level. Besides, the aggregation should satisfy some basic rules so as to make it equivalent to original system and guarantee the results are always feasible. Some properties of aggregation are discussed also in [19].

In this section, first we propose the basic rules which the desired aggregation should satisfy. Then the method of aggregating nodes without disruption is introduced. Also, the disaggregation of the control law of such nodes is discussed.

#### 5.2.1 Aggregation Definition

Suppose the nodes to be aggregated are $N_{1}, N_{2}, ..., N_{n}$, and the aggregated node is $N^{a}$. The parameters of $N^{a}$ are denoted by $P(N^{a}) = [b^{a}, r^{a}, d^{3, a}, q_{u}(N^{a})]^{T}$. The aggregation is defined with a function $\text{Agg}(\cdot)$ over parameter vectors of $\{N_{1}, N_{2}, ..., N_{n}\}$ such that

$$P(N^{a}) = \text{Agg}(P(N_{1}), P(N_{2}), ..., P(N_{n})).$$
**Aggregation Property**

An aggregation function $Agg(P(N_1), P(N_2), ..., P(N_n))$ satisfies the *Aggregation Property* if for all the time and $u$, there exists a $u^a$ such that

1. The inventory change due to the production of $N^a$ is the same as the inventory change due to the production of $\{N_1, N_2, ..., N_n\}$.

$$b^a u^a = [b_1 \ b_2 \ ... \ b_n] \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix}.$$  

2. The production cost of $N^a$ is the same as production cost of $\{N_1, N_2, ..., N_n\}$.

$$q_u(N^a) u^a = [q_u(N_1) \ q_u(N_2) \ ... \ q_u(N_n)] \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix}.$$  

3. The amount of resources used by $N^a$ is the same as the amount of resources used by $\{N_1, N_2, ..., N_n\}$.

$$r^a u^a = [r_1 \ r_2 \ ... \ r_n] \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix}.$$  

4. The demand filled by $N^a$ is the same as the demand filled by $\{N_1, N_2, ..., N_n\}$.

$$d^{D,a} u^a = [d_1^{D} \ d_2^{D} \ ... \ d_n^{D}] \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix}.$$  

The *Aggregation Property* is summarized by:

$$[P(N_1) \ P(N_2) \ ... \ P(N_n)] \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} = P(N^a) u^a. \quad \text{(5.2)}$$

This equation means, the change of inventory amount, production cost, resource used and demand filled do not change before and after aggregation. Thus, the $N^a$ is equivalent to the set of $\{N_1, N_2, ..., N_n\}$.

An aggregation function $Agg(\cdot)$ satisfies the *Aggregation Property* if for all the time and $u$, there exists a $u^a$ such that Equation (5.2) is satisfied.

We say that the operations $u$ and $u^a$ correspond under aggregation function $Agg(\cdot)$ if Equation (5.2) is satisfied for all the time.
Associativity

The approaches above describe the aggregation in a single step. Sometimes, it is needed to do multiple aggregations in a sequence. A useful property is found for multiple aggregations.

**Definition 10** (Associativity). Suppose there are \( n \) nodes: \( N_1, N_2, ..., N_n \). \( n_1, n_2, ..., n_g \) are integers such that:

\[
1 \leq n_1 < n_2 < ... < n_{g-1} < n_g = n,
\]

then an aggregation function \( \text{Agg}(\cdot) \) satisfies the Associativity property if:

\[
P(N^a) = \text{Agg}(P(N_1), P(N_2), ..., P(N_n))
= \text{Agg}[\text{Agg}(P(N_1), P(N_2), ..., P(N_{n_1})),
\text{Agg}(P(N_{n_1+1}), P(N_{n_1+2}), ..., P(N_{n_2})),
\ldots,
\text{Agg}(P(N_{n_{g-1}+1}), P(N_{n_{g-1}+2}), ..., P(N_n))].
\]

**Theorem 4** (Associativity). If a method of aggregation satisfies Aggregation Property given by Equation (5.2), then this method satisfies associativity.

The proof of Theorem 4 is given in Appendix B.

The aggregation can be done recursively. It will make no difference whether all the nodes are aggregated in a single step, or in multiple steps with certain sequence. Aggregating in multiple steps means to group all the nodes and aggregate each group separately to form a number of small aggregated nodes, and then aggregate them as normal nodes to form one bigger node.

5.2.2 Aggregation Approach

**Aggregation for Networks of Nodes with Fixed Operations Ratio**

One type of networks that can be aggregated is the network of nodes with fixed operations ratio.

**Definition 11** (Network of nodes with fixed operations ratio). Consider a set of nodes \( \{N_1, N_2, ..., N_n\} \), whose operations are \( \{u_1, u_2, ..., u_n\} \), respectively. If

\[
\frac{u_1}{\alpha_1} = \frac{u_2}{\alpha_2} = ... = \frac{u_n}{\alpha_n} = \tilde{u}
\]

for all the time, and \( \alpha_i \) are constants for \( i = 1, 2, ..., n \),

which means the ratio of their operations is fixed, then the set of nodes \( \{N_1, N_2, ..., N_n\} \) are defined as the network of nodes with fixed operations ratio.

Mostly, such nodes appear when no inventory changes among them. For the set of nodes \( \{N_1, N_2, ..., N_n\} \), if all of product \( i \) produced by \( N_a \) is fed to \( N_b \) \( (a, b \in \{1, 2, ..., n\}) \), \( u_a \) and \( u_b \) can be chosen such that \( x_i \) is not changed by \( N_a \) and \( N_b \). Such networks include the
serial structures, the AND structures, and other networks made up of these two kinds of structures, etc.

Figure 4.1 shows the serial structure. If no inventory changes among the nodes, then the upstream nodes would produce the exact amount of inventory required by the downstream nodes. Their operations thus have a fixed ratio.

Figure 4.2 illustrates the AND structure. If there is no inventory change among the nodes, the amount of inventory produced by upstream nodes is also determined by the downstream node, which makes their operations in a fixed ratio.

In such subsystem shown in Figure 5.1 if one node’s operation is known, all the others’ operations can be easily calculated.

For networks of nodes with fixed operations ratio, the aggregation is to add the parameters of each nodes based on the fixed ratio.

**Theorem 5** (Aggregation for networks of nodes with fixed operations ratio). *If a set of nodes $N_1, N_2, ..., N_n$ is a network of nodes with fixed operations ratio defined by Definition 11, and if we define $Agg(\cdot)$ over these nodes as:*

$$Agg(P(N_1), P(N_2), ..., P(N_n)) := \alpha_1 P(N_1) + \alpha_2 P(N_2) + ... + \alpha_n P(N_n)$$

$$= [P(N_1) \quad P(N_2) \quad ... \quad P(N_n)] \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix}, \quad (5.4)$$

*where any of $\alpha_i$ is the parameter in Definition 11, then $Agg(\cdot)$ satisfies the Aggregation Property.*

The proof of Theorem 5 is given in Appendix C.1.

Denote $N^a$ as the aggregated node. It follows from Theorem 5 and Equation (5.4) that

$$b^a = \alpha_1 b_1 + \alpha_2 b_2 + ... + \alpha_n b_n; \quad (5.5)$$

$$q_u(N^a) = \alpha_1 q_u(N_1) + \alpha_2 q_u(N_2) + ... + \alpha_n q_u(N_n). \quad (5.6)$$
Consider the Capacity Constraint. At any time, we have

\[
\begin{bmatrix}
  r_1, r_2, ..., r_n \\
  u_1 \\
  u_2 \\
  \vdots \\
  u_n
\end{bmatrix} \leq \begin{bmatrix}
  c_1 \\
  c_2 \\
  \vdots \\
  c_n
\end{bmatrix}.
\]

Thus,

\[
r^a u^a \leq \begin{bmatrix}
  c_1 \\
  c_2 \\
  \vdots \\
  c_n
\end{bmatrix}.
\]

\[
u^a \leq \min \left\{ \frac{c_j}{r^a(j)} \mid 1 \leq j \leq n_r \right\}.
\]

(5.9)

If the operation of any of the nodes is known, we can easily obtain the operation of all other nodes by multiplying the fixed ratio. Thus, it is possible to use only one operation to control all the nodes, which are working as a whole at the same time. Each node is a part of the aggregated node. The parameter vector of the aggregated node is the sum of those of each node.

Disrupted chain is a special set of nodes which has almost the same structure of the set of the serial structure but does not satisfy the definition of the network of nodes with fixed operation ratio. So, attention needs to be paid to such disrupted chain when doing aggregation in the following chapters, as they are not fixed ratio nodes but may be easily mis-recognized as serial nodes.

**Definition 12** (Disrupted Chain). *The disrupted chain is a serial structure with a potential disrupted node at the start.*

Consider the example below:

![Figure 5.2: A disrupted chain](image)

For this example, assume \( N_2 \) is the node which will be disrupted. Although \( N_1 \) and \( N_0 \) are serial nodes, the operation ratio may not be fixed due to the disruption at \( N_2 \).

During the disruption, \( N_2 \) will stop working, and it cannot provide any input for the down-stream nodes. If there is no inventory built up before the disruption, \( N_1 \) and \( N_0 \) will also
stop due to the lack of input, so that a lost demand will be generated. Under Assumption 1, the lost demand cost weighs much more than inventory storage cost and production cost. Then, in order to keep a lower total cost, the system will build up some inventory cost before the disruption to prepare for the work afterwards.

To build up either product 2 or product 1 can make N 0 meet the demand during disruption. The system will choose to build up the inventory with lower storage cost. We assume product 1 has the lower cost. So N 1 will increase its production before the disruption while N 0 do not change its production, so that product 1 can be built up. This means the ratio of production of N 1 and N 0 is changing and does not equal to the value in steady state. In this case, N 1 and N 0 cannot be considered as the nodes with fixed ratio of production.

The disrupted chain can be defined as the chain of nodes which has disrupted node in it. In disrupted chain, the disrupted node can cause inventory built up among the downstream nodes, so that their production ratio cannot be considered as fixed directly.

In addition, if product 1 is built up in the example above, N 1 and N 0 can still have the fixed ratio of production, because no product 1 is built up.

**Aggregation for OR Nodes without Disruption in Discrete Time domain**

In discrete time domain, we assume operation won’t hit any capacity limits in nominal state, as per Assumption 5. For OR nodes, the aggregation is to select one node among them as the aggregated node, which has the lowest cost to produce the same amount of output inventory.

**Theorem 6 (Aggregation for OR nodes).** Assume a set of nodes N 1, N 2, ..., N n are OR nodes defined by Definition 6. b i is the element in b i associated with product out, which is the amount of product out produced by N i with unit operation. q u(N i) is the unit production cost of N i. If j is the index such that

\[
\frac{q_u(N_j)}{b_{i}^{out}} \leq \frac{q_u(N_i)}{b_{i}^{out}} \quad \forall i,
\]

and we define Agg(·) over these nodes as:

\[
Agg(P(N_1), P(N_2), ..., P(N_n)) := P(N_j),
\]

(5.10)

Then Agg(·) satisfies the Aggregation Property.

The proof of Theorem 6 is given in Appendix C.2.

A result of Theorem 6 is

\[
b^a = b_j, \quad q_u(N^a) = q_u(N_j), \quad r^a = r_j, \quad \text{and} \quad d^{D,a} = d^D_j.
\]

(5.11)

It is still not necessary to build up any inventory. The optional nodes can change operations identically when we do not consider changeover cost and constraint. So the only difference among them is in the production cost. We can compare the production cost of each optional nodes when they produce same amount of product out. Assume N j has the lowest cost. At any time, if any other optional node is producing, we can shift its work to N j so that the total production cost can be reduced. So, we can let the operations of all nodes be 0 except N j.
5.2.3 Disaggregate Control

By doing the aggregation, we can simplify the system. If we have the control solution for this simplified system, the problem comes to how to apply this control to the original system. The issue is some kind of disaggregation of the control. Assume we know the original system structure and have the operation command $u^a$ for the aggregated system, we need to find out the way to calculate the corresponding operation $u$ for the original system.

**Theorem 7** (Disaggregate control of nodes with fixed production ratio). A set of nodes $\{N_1, N_2, ..., N_n\}$ are defined by Definition 11. $N^a$ is the aggregated node obtained from Equation (5.4). We assume the aggregated node has the operation command of $u^a$. Correspondingly, the original subsystem has the operation command of $u = [u_1 \ u_2 \ ... \ u_n]$. Let

$$[u_1 \ u_2 \ ... \ u_n] = [u^a \alpha_1 \ u^a \alpha_2 \ ... \ u^a \alpha_n]. \quad (5.12)$$

where $\alpha_i \forall i$ is defined as in Definition 11. Then $u$ and $u^a$ correspond under aggregation function $Agg(\cdot)$.

**Proof of Theorem 7** Based on the Equation (5.12), we have:

$$[P(N_1) \ P(N_2) \ ... \ P(N_n)] \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} = [P(N_1) \ P(N_2) \ ... \ P(N_n)] \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_m \end{bmatrix} u^a.$$

According to Equation (5.4), it becomes:

$$[P(N_1) \ P(N_2) \ ... \ P(N_n)] \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} = P(N^a)u^a.$$

So, Equation (5.2) is satisfied.

**Theorem 8** (Disaggregate control of OR nodes). A set of nodes $\{N_1, N_2, ..., N_n\}$ are defined by Definition 6. $N^a$ is the aggregated node obtained from Equation (5.10). We assume the aggregated node has the operation command of $u^a(k)$. The original subsystem has the operation command of $u(k)$. Let

$$u_i(k) = \begin{cases} u^a(k), & \text{when } i = j; \\ 0, & \text{when } i \neq j. \end{cases} \quad (5.13)$$

where $j$ is defined as in Theorem 6. Then, $u(k)$ and $u^a(k)$ correspond under aggregation function $Agg(\cdot)$.

47
Proof of Theorem 8. Based on Equation (5.13), we have:

\[
\begin{bmatrix}
N_1 & N_2 & \cdots & N_n
\end{bmatrix}
\begin{bmatrix}
u_1(k) \\
u_2(k) \\
\vdots \\
u_n(k)
\end{bmatrix} = P(N_j)u^a(k).
\]

According to Equation (5.10), it becomes:

\[
\begin{bmatrix}
N_1 & N_2 & \cdots & N_n
\end{bmatrix}
\begin{bmatrix}
u_1(k) \\
u_2(k) \\
\vdots \\
u_n(k)
\end{bmatrix} = P(N^a)u^a(k).
\]

So the Equation (5.2) is satisfied.

5.2.4 An Example

Consider a network example.

Assume the network matrix is:

\[
B = \begin{bmatrix}
m_1 & 0 & 0 & 0 & -m_1 & m_1 & 0 & 0 & 0 & 0 \\
0 & m_2 & 0 & 0 & 0 & -m_2 & 0 & 0 & 0 & 0 \\
0 & 0 & m_3 & 0 & 0 & 0 & -m_3 & 0 & 0 & 0 \\
0 & 0 & 0 & m_4 & 0 & 0 & -m_4 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & m_5 & 0 & 0 & -m_5 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & m_7 & 0 & -m_7 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & m_8 & -m_8 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1
\end{bmatrix}.
\]

In steady state, the ratio of operations of any two nodes is 1. The production cost vector is:

\[
q_u = [q_u(N_1) \ q_u(N_2) \ \cdots \ q_u(N_{10})].
\]

The demand matrix is \( D = [0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1] \). The demand at any time is \( d \). \( N_5 \) is fed by OR nodes. Others can be considered as with fixed operation ratio.

We assume \( N_8 \) will be disrupted. According to Theorem 3, the remaining part of the network can be aggregated in the following steps:
1. Aggregate \( N_2 \) and \( N_6 \) to form \( N_a^2 \). According to Equation (5.5) and (5.6), \( b_1^a = b_2 + b_6 = \begin{bmatrix} m_1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \). \( q_u(N_a^1) = q_u(N_2) + q_u(N_6) \).

2. Aggregate \( N_1^a \) and \( N_7 \) to form \( N_a^2 \). As they are OR nodes, According to Equation (5.11), we compare the value of \( q_u(N_1) \) and \( q_u(N_a^1) \) to find the smaller one. We assume that \( q_u(N_1) \leq q_u(N_a^1) \). So \( b_2^a = b_1 \) and \( q_u(N_a^2) = q_u(N_1) \).

3. Aggregate \( N_2^a \) and \( N_5 \) to form \( N_a^3 \). According to Equation (5.5) and (5.6), \( b_3^a = b_2^a + b_5 = \begin{bmatrix} 0 & 0 & 0 & m_5 & 0 & 0 & 0 \end{bmatrix}^T \). \( q_u(N_a^3) = q_u(N_2^a) + q_u(N_5) = q_u(N_2) + q_u(N_5) \).

4. Aggregate \( N_3, N_4, N_7 \) and \( N_9 \) to form \( N_a^4 \). According to Equation (5.5) and (5.6), \( b_4^a = b_3 + b_4 + b_7 + b_9 = \begin{bmatrix} 0 & 0 & 0 & 0 & m_8 & -1 \end{bmatrix}^T \). \( q_u(N_a^4) = q_u(N_3) + q_u(N_4) + q_u(N_7) + q_u(N_9) \).

The aggregated system becomes:

![Diagram](image)

**Figure 5.4: System aggregation result**

Although \( N_a^4 \) and \( N_{10} \) have a fixed ratio structure, as they are in the disrupted chain, we do not aggregate them.

The aggregated network matrix becomes:

\[
B^a = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & m_5 & -m_5 & 0 \\
0 & 0 & 0 & 0 \\
0 & m_8 & -m_8 & 0 \\
0 & 0 & 1 & -1
\end{bmatrix}
\]

And the production cost vector becomes:

\[
q_u^a = [q_u(N_a^3) \ q_u(N_8) \ q_u(N_a^4) \ q_u(N_{10})]
\]

\[
= [q_u(N_3) + q_u(N_5) \ q_u(N_8) \\
q_u(N_3) + q_u(N_4) + q_u(N_7) + q_u(N_9) \ q_u(N_{10})].
\]

After such aggregation, the original system is simplified to the simple example discussed before. Based on the analysis results, we easily find the response of the original system. Moreover, we apply Theorem 7 and Theorem 8 to disaggregate the control.

To select the inventory to build up, we need to compare their cost, \( m_8 q_x(N_8) \) and \( q_x(N_{10}) \), and choose the cheaper one.
In the steady state before disruption, total production cost in each cycle is:

\[ C_{\text{nominal}} = a_u q_u u \]

\[ = a_u \begin{bmatrix} q_u(N_1) & q_u(N_5) & q_u(N_8) & q_u(N_3) + q_u(N_4) + q_u(N_7) + q_u(N_9) & q_u(N_{10}) \end{bmatrix} \begin{bmatrix} d \\ d \\ d \\ d \\ d \end{bmatrix} \]

\[ = a_u \begin{bmatrix} q_u(N_1) + q_u(N_5) + q_u(N_8) + q_u(N_3) + q_u(N_4) + q_u(N_7) + q_u(N_9) + q_u(N_{10}) \end{bmatrix} \]

Consider the period of \([k_{\text{dis}} - 1, k_{\text{dis}} + \kappa'_{\text{dis}} - 1] \), total production cost is \( C_{\text{nominal}}(\kappa'_{\text{dis}} + 1) \).

• If building up product \( 9 \), the total cost increase is \( a_x q_x(N_9) d (\kappa'_{\text{dis}} + 1)^2 \).

\[ u(k) = \begin{cases} 
\begin{bmatrix} d & 0 & d & 0 & d & d \end{bmatrix}^T, & \text{when } k < k_{\text{dis}} - 2; \\
\begin{bmatrix} d + d \cdot \kappa'_{\text{dis}} & 0 & d + d \cdot \kappa'_{\text{dis}} \\
0 & d + d \cdot \kappa'_{\text{dis}} \\
d + d \cdot \kappa'_{\text{dis}} \\
0 & d + d \cdot \kappa'_{\text{dis}} \\
d + d \cdot \kappa'_{\text{dis}} \\
0 & d \end{bmatrix}^T, & \text{when } k = k_{\text{dis}} - 1; \\
\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T, & \text{when } k_{\text{dis}} \leq k \leq k_{\text{dis}} + \kappa'_{\text{dis}} - 1. 
\end{cases} \]

• If building up product \( 8 \), the total cost increase is \( a_x q_x(N_8) m_8 \cdot d (\kappa'_{\text{dis}} + 1)^2 \).

\[ u(k) = \begin{cases} 
\begin{bmatrix} d & 0 & d & d & 0 & d & d \end{bmatrix}^T, & \text{when } k < k_{\text{dis}} - 2; \\
\begin{bmatrix} d + d \cdot \kappa'_{\text{dis}} & 0 & d & d + d \cdot \kappa'_{\text{dis}} & d \end{bmatrix}^T, & \text{when } k = k_{\text{dis}} - 1; \\
\begin{bmatrix} 0 & d & d + d \cdot \kappa'_{\text{dis}} & d \end{bmatrix}^T, & \text{when } k_{\text{dis}} \leq k \leq k_{\text{dis}} + \kappa'_{\text{dis}} - 1. 
\end{cases} \]

From this example, we find that the aggregation can simplify the system and make it much easier to analyze. The disaggregation of control can help to get the control for original system from the solution of aggregated system.

### 5.3 Conclusion

In order to analyze the complex systems, we introduced the normalization and aggregation to simplify the network structure. The approach of aggregation was presented, proved, and applied to solve for the optimal control for an example of more complex system.
Chapter 6

Analysis in Discrete Time Domain

In this chapter, we will use our model to analyze specific systems defined in Chapter 4 in discrete time domain.

6.1 Serial Network Systems

This section will discuss the analysis of serial network systems in discrete time domain. Some of the analysis and results are presented also in [20]. Consider a serial network system as per Definition 8 which is made of nodes \{N_0, N_1, ..., N_l\}, illustrated in Figure 6.1.

![Figure 6.1: A serial network system](image)

Denote products as \{product_1, product_2, ..., product_l\}. For any index \(i \in \{1, 2, ..., l\}\), product_\(i\) is the output of \(N_i\) and input of \(N_{i-1}\). The final node \(N_0\) represents the consumer of the final product product_\(1\). The demand is initiated from the final node, i.e., the demand matrix is \(D = [1, 0, 0, ..., 0]\). We assume the demand is time-invariant, which is denoted as \(d\). Thus, \(d(k) = d\). Assume the disruption happens at node \(N_{\text{dis}}\). We define that “upstream” nodes are associated with higher index, and “downstream” associated with lower index.

Let initial inventory be \(x(0) = 0\). The time point when the disruption occurs is denoted as \(k_{\text{dis}}\). The expected length of the disruption is denoted as \(\kappa'_{\text{dis}}\). Its value equals to the smaller one between \(\kappa_f\) and the real length of the disruption \(\kappa_{\text{dis}}\).

6.1.1 Control Solution of the Serial Network systems

We denote a vector \(\alpha\), whose elements are all non-negative, such that

\[
[b_1, b_2, ..., b_n] \alpha = 0, \text{ and } D \alpha = d.
\]
Recall that $N_i \succ N_j$ means that $N_i$ is upstream of $N_j$. $N_i \succsim N_j$ means that $N_i$ is upstream of $N_j$, or $N_i$ and $N_j$ are the same node.

Consider the index $s$ that minimizes $q_x(N_i)\alpha_i b_i(i)$ for $i$ such that $N_{dis} \succ N_i \succsim N_0$. If there are multiple indices satisfying this condition, it can be proved that any of them can be selected as $s$. In the following analysis, we assume that such an $s$ is unique. Then, as shown in the next section, the optimal control $u$ is:

1. when $k < k_{dis} - 1$
   \[ u(k) = \alpha; \]

2. when $k = k_{dis} - 1$
   \[ u(k) = [\alpha_0, \alpha_1, \ldots, \alpha_{s-1}, \alpha_s (1 + \kappa'_{dis}), \alpha_{s+1} (1 + \kappa'_{dis}), \ldots, \alpha_l (1 + \kappa'_{dis})]^T; \]

3. when $k_{dis} \leq k \leq k_{dis} + \kappa'_{dis} - 1$
   \[ u(k) = [\alpha_0, \alpha_1, \ldots, \alpha_{s-1}, 0, 0, \ldots, 0]^T. \]

The solution shows the policy that the control operation of the system is to build up desired inventory before the disruption, so that the demand can be satisfied as much as possible during the disruption. Specifically, the inventory will be built up in the cycle right before the disruption. Prior to buildup, the system works in a nominal state to produce exact amount of product of demand. The cheapest product in the inventory will be built up, whose amount is based on the estimation length of the disruption. During the disruption, the system will use this buildup to keep producing the final product.

A derivation of such solution and explanations of the rationales are presented next.

### 6.1.2 Derivation of Solution to the Problem of Serial Network System

We divide the solution to the problem of serial production network systems into six statements.

**A: Final node $N_0$ always must satisfy demand.**

Based on the optimization problem we formulate, the primary objective is to minimize the lost demand. Although $N_{dis}$ does not work during the disruption, nodes among \{\text{\textit{N}_i | N_{dis} \succ N_i \succsim N_0}\} can still work if there is sufficient inventory built up. Since we do not consider capacity and changeover constraints in the discrete time case, the serial network system can always build up sufficient inventory before disruption happens. Therefore, it is always possible to support $N_0$ to work to fill demand even during the disruption. In order to minimize the lost demand, the optimal control of the system will make its effort to keep $N_0$ working to satisfy the demand.

A mathematical form of this statement is given below:

- When $k \leq k_{dis} + \kappa'_{dis} - 1$, then $u_0(k) = \alpha_0$. 

52
We prove this statement by contradiction. Assume the optimal control is \( u' \) and there exists
an index \( k_1 \) such that \( u'_0(k_1) \neq \alpha_0 \). According to the Demand Constraint, \( u'_0(k_1) < \alpha_0 \).

If \( k_1 < k_{\text{dis}} \), we can select a \( u'' \) such that
\[ u''(k) := \begin{cases} 
\alpha_0, u'_1(k_1) + \alpha_1, u'_2(k_1) + \alpha_2, ..., u'_l(k_1) + \alpha_l \|^T, & \text{when } k = k_1; \\
u'(k), & \text{when } k \neq k_1, \text{ and } k \leq k_{\text{dis}} + \kappa'_{\text{dis}} - 1.
\end{cases} \]

It can be shown that \( u'' \) satisfies all the constraints and
\[ \sum_{k'=k-\kappa_p}^{k+\kappa_f} \{a_d[d(k') - D u'(k')]\} > \sum_{k'=k-\kappa_p}^{k+\kappa_f} \{a_d[d(k') - D u''(k')]\}, \]
which means \( u' \) is not optimal. This is a contradiction.

If \( k_1 \geq k_{\text{dis}} \), we can select a \( u'' \) such that
1. for \( k = k_{\text{dis}} \),
\[ u''(k) := [\alpha_0, u'_1(k_{\text{dis}}) + \alpha_1, u'_2(k_{\text{dis}}) + \alpha_2, ..., u'_l(k_{\text{dis}}) + \alpha_l] \|^T; \]
2. for \( k = k_1 \),
\[ u''(k) := [\alpha_0, u'_1(k_1), u'_2(k_1), ..., u'_l(k_1)] \|^T; \]
3. for \( k \neq k_1 \), and \( k \neq k_{\text{dis}} \), and \( k \leq k_{\text{dis}} + \kappa'_{\text{dis}} - 1 \),
\[ u''(k) := u'(k). \]

It can be shown that \( u'' \) satisfies all the constraints and
\[ \sum_{k'=k-\kappa_p}^{k+\kappa_f} \{a_d[d(k') - D u'(k')]\} > \sum_{k'=k-\kappa_p}^{k+\kappa_f} \{a_d[d(k') - D u''(k')]\}, \]
which implies that \( u' \) is not optimal. Again this leads to a contradiction.

Therefore,
\[ u_0(k) = d = \alpha_0 \text{ when } k \leq k_{\text{dis}} + \kappa'_{\text{dis}} - 1. \]  \( (6.1) \)

**B: Once a disruption is expected, the serial production network system needs to build up inventory. This is postponed until as late as possible, which is at time \( k_{\text{dis}} - 1 \).**

In order to make \( N_0 \) working to meet demand during the disruption, the network system needs to build up inventory in advance. Otherwise, downstream nodes of \( N_{\text{dis}} \) maybe starved, since \( N_{\text{dis}} \) could not work during the disruption. Hence, \( N_{\text{dis}} \) needs to work more than needed before the disruption in order to compensate its shutting down during the disruption. Therefore the inventory buildup begins before the disruption.

Since capacity and changeover constraints are not considered, the network system can build up any amount of inventory at any time before the disruption. As our objective function includes inventory storage cost, if the system can shift the building-up operation to a later
time, the same amount inventory can have a shorter storage period, which can lead to a lower storage cost. The total production cost and lost demand do not change if the system shifts the operation. Under optimal control strategy, the buildup will happen as late as possible. Therefore, it will not occur until time $k_{dis} - 1$. This implies that when $k < k_{dis} - 1$, no buildup of inventory in the network system. The mathematical form of this statement is:

- When $k < k_{dis} - 1$, then $u(k) = \alpha$.

Again we prove this statement by contradiction. Assume $u'$ be the optimal control with $u'(k) \neq u(k)$ for some values of $k$. Then, there exists one smallest time point $k_1$ such that $u'(k_1) \neq \alpha$. Denote $i$ as the smallest index such that $u'_i(k_1) \neq \alpha_i$.

Define a vector

$$\Delta u'(k_1) = [\Delta u'_0(k_1), \Delta u'_1(k_1), \ldots, \Delta u'_{i-1}(k_1)]^T,$$

where

$$\Delta u'_j(k_1) = \begin{cases} \alpha_j \frac{u'_j(k_1)}{\alpha_i} - \frac{u'_{i-1}(k_1)}{\alpha_i}, & \text{when } i \leq j \leq l; \\ 0, & \text{when } 0 \leq j \leq i - 1. \end{cases}$$

Then we can select another control signal $u''$ such that:

$$u''(k) := \begin{cases} u'(k), & \text{when } k \neq k_1 \text{ and } k \neq k_1 + 1; \\ u'(k) - \Delta u'(k_1), & \text{when } k = k_1; \\ u'(k) + \Delta u'(k_1), & \text{when } k = k_1 + 1. \end{cases}$$

It can be shown that $u''$ satisfies all the constraints and

$$\sum_{k' = k_1 - \kappa_p}^{k_1 + \kappa_f} \{a_u q_u u''(k') + a_x q_x x''(k')\} < \sum_{k' = k_1 - \kappa_p}^{k_1 + \kappa_f} \{a_u q_u u'(k') + a_x q_x x'(k')\},$$

where $x'$ corresponds to the inventory under $u'$, and $x''$ corresponds to the inventory under $u''$. This implies $u'$ is not the optimal solution, which is a contradiction.

Therefore, the optimal control in nominal stage is $u(k) = \alpha$.

**C: At any time, there will be no inventory buildup upstream of the disrupted node $N_{dis}$.**

During the disruption, $N_{dis}$ does not work. If there is any inventory buildup before $N_{dis}$, it cannot be transmitted to the nodes downstream of $N_{dis}$, and it cannot contribute to supporting the operation of $N_0$ during the disruption. Therefore there is no need to build up inventory upstream of $N_{dis}$.

From Statement B, we already know that there will be no inventory buildup until time $k_{dis} - 1$. So this statement implies that from time $k_{dis} - 1$ there will be no inventory buildup upstream of $N_{dis}$. A mathematical form of this statement is as follows:
• When $k_{\text{dis}} - 1 \leq k \leq k_{\text{dis}} + \kappa'_\text{dis} - 1$, then \( \frac{u(k)}{\alpha_l} = \frac{u_{l-1}(k)}{\alpha_{l-1}} = \ldots = \frac{u_{\text{dis}}(k)}{\alpha_{\text{dis}}} \).

We prove this by contradiction. We assume the optimal solution as \( u' \) and there exists a smallest \( k_2 \) and an index \( i \) such that \( N_i \geq N_i > N_{\text{dis}} \) and \( \frac{u'_i(k_2)}{\alpha_i} = \frac{u'_{i-1}(k_2)}{\alpha_{i-1}} \). Then we can select another control \( u'' \) such that:

1. for \( k = k_2 \): \( u''(k) := \) 
   \[
   [u'_0(k_2), u'_1(k_2), \ldots, u'_{\text{dis}}(k_2), \alpha_{\text{dis}} + 1 \frac{u'_{\text{dis}}(k_2)}{\alpha_{\text{dis}}}, \alpha_{\text{dis}} + 2 \frac{u'_{\text{dis}}(k_2)}{\alpha_{\text{dis}}}, \ldots, \alpha_i \frac{u'_{\text{dis}}(k_2)}{\alpha_{\text{dis}}}]^T;
   \]

2. for \( k = k_2 + 1 \): \( u''(k) := \) 
   \[
   \begin{bmatrix}
   u'_0(k_2 + 1) \\
   u'_1(k_2 + 1) \\
   \vdots \\
   u'_{\text{dis}}(k_2 + 1) + u'_{\text{dis}}(k_2) - \alpha_i \frac{u'_{\text{dis}}(k_2)}{\alpha_{\text{dis}}} \\
   \vdots \\
   u'_{i-1}(k_2 + 1) + u'_{i-1}(k_2) - \alpha_i \frac{u'_{\text{dis}}(k_2)}{\alpha_{\text{dis}}} \\
   u'_i(k_2 + 1) + u'_i(k_2) - \alpha_i \frac{u'_{\text{dis}}(k_2)}{\alpha_{\text{dis}}}
   \end{bmatrix};
   \]

3. for \( k \neq k_2 \) and \( k \neq k_2 + 1 \), \( u''(k) := u'(k) \).

It can be shown that \( u'' \) satisfies all the constraints and 
\[
\sum_k q_u u'(k) = \sum_k q_u u''(k),
\]
\[
\sum_k q_x x'(k) > \sum_k q_x x''(k),
\]
where \( x' \) corresponds to the inventory under \( u' \), and \( x'' \) corresponds to the inventory under \( u'' \). Therefore, \( u' \) is not optimal. This is a contradiction.

Thus, \( \frac{u_i(k)}{\alpha_i} = \frac{u_{i-1}(k)}{\alpha_{i-1}} = \ldots = \frac{u_{\text{dis}}(k)}{\alpha_{\text{dis}}} \) when \( k_{\text{dis}} - 1 \leq k \leq k_{\text{dis}} + \kappa'_\text{dis} - 1 \).

**D: The inventory buildup at time \( k_{\text{dis}} \) is determined by \( \kappa'_\text{dis} \), the predicted length of disruption.**

From Statement B, we know the inventory buildup begins at time \( k_{\text{dis}} - 1 \). At time \( k_{\text{dis}} \), the disruption happens and building-up operation stops. During the disruption, the system only makes use of the inventory buildup. Therefore, the inventory amount at time \( k_{\text{dis}} \) should be able to support the usage during the disruption. Since the length of disruption is predicted as \( \kappa'_\text{dis} \) at time \( k_{\text{dis}} \), the inventory buildup relies on \( \kappa'_\text{dis} \).

From Statement C, we know there is no inventory buildup upstream of \( N_{\text{dis}} \), that is \( x_i(k_{\text{dis}}) = 0 \) for \( i = l, l - 1, \ldots, \text{dis} + 1 \). Consider a node \( N_i \), it uses \( -b_i(i + 1) \) amount of
product\_i+1 to produce \( b_i(i) \) amount of product\_i. This suggests that for the contribution of inventory buildup, \(-b_i(i + 1)\) amount of product\_i+1 is equivalent to \( b_i(i) \) amount of product\_i. Since \( B\alpha = 0 \), \( b_{i-1}(i-1)\alpha_{i-1} \) amount of product\_{i-1} is equivalent to \( b_i(i)\alpha_i \) amount of product\_i. \( N_0 \) working to meet demand for \( \kappa'_{\text{dis}} \) requires the inventory buildup being equivalent to \( b_1\alpha_1\kappa'_{\text{dis}} \) amount of product\_1. Therefore, in a mathematical form, under optimal control \( u \), \( x(k_{\text{dis}}) \) satisfies

\[
\frac{x_1(k_{\text{dis}})}{b_1(1)\alpha_1} + \frac{x_2(k_{\text{dis}})}{b_2(2)\alpha_2} + \ldots + \frac{x_{\text{dis}}(k_{\text{dis}})}{b_{\text{dis}}(\text{dis})\alpha_{\text{dis}}} = \kappa'_{\text{dis}}.
\]

This can be proved by contradiction in a similar way as previous statements.

**E: Inventory buildup occurs to only one type of product, which is the product with the lowest storage cost downstream of the disrupted node.**

From Statement D, we know that building up any product can satisfy the inventory requirement which supports \( N_0 \) to work to meet the demand. Since the storage cost may not be exactly the same for different products, the optimal control will make the system choose to build up the product with lowest storage cost. This does not affect the total production cost and lost demand.

Let \( s \) be the index that minimizes \( q_x(N_i)\alpha_i b_i(i) \) for \( i \) such that \( N_{\text{dis}} \geq N_i \geq N_1 \). For other indices \( i \) such that \( i \neq s \) and \( N_{\text{dis}} \geq N_i \geq N_1 \), if the amount of product\_i equivalent to the amount of product\_s, i.e., \( \frac{x_i(k)}{\alpha_i b_i(s)} = \frac{x_s(k)}{\alpha_s b_s(s)} \), then the storage cost of product\_i is higher than product\_s, i.e., \( q_x(N_i)x_i(k) > q_x(N_s)x_s(k) \). Therefore, the system will build up product\_s instead of product\_i to achieve lower cost. Thus, \( x_i(k) = 0 \) for \( i \neq s \).

At time \( k_{\text{dis}} \), \( x_s(k_{\text{dis}}) = b_s(s)\alpha_s k'_{\text{dis}} \). In each cycle after time \( k_{\text{dis}} \), the amount of buildup used by the system is \( b_s(s)\alpha_s \). Thus \( x_s(k) = b_s(s)\alpha_s (k_{\text{dis}} + \kappa'_{\text{dis}} - k) \) when \( k \in \{k_{\text{dis}}, k_{\text{dis}} + 1, \ldots, k_{\text{dis}} + \kappa'_{\text{dis}}\} \). Under optimal control \( u \), \( x(k) \) satisfies \( x(k) = [0, 0, \ldots, 0, x_s(k), 0, 0, 0]^T \) when \( k \in \{k_{\text{dis}}, k_{\text{dis}} + 1, \ldots, k_{\text{dis}} + \kappa'_{\text{dis}}\} \).

This statement can also be proved by contradiction. We assume the optimal control as \( u' \) under which \( x' \) satisfies that there exists a time \( k_2 \in \{k_{\text{dis}}, k_{\text{dis}} + 1, \ldots, k_{\text{dis}} + \kappa'_{\text{dis}}\} \) such that \( x'(k_2) \neq x(k_2) \). Then we can select another control \( u'' \) such that

1. for \( k \notin [k_{\text{dis}} - 1, k_{\text{dis}} + \kappa'_{\text{dis}} - 1] \), \( u''(k) := u'(k) \).
2. for \( k = k_{\text{dis}} - 1 \), \( u''(k) := \left[\alpha_0, \alpha_1, \ldots, \alpha_{s-1}, \alpha_s(1 + k'_{\text{dis}}), \alpha_{s+1}(1 + k'_{\text{dis}}), \ldots, \alpha_1(1 + k'_{\text{dis}})\right]^T \).
3. for \( k_{\text{dis}} \leq k \leq k_{\text{dis}} + \kappa'_{\text{dis}} - 1 \), \( u''(k) := \left[\alpha_0, \alpha_1, \ldots, \alpha_{s-1}, 0, 0, \ldots, 0\right]^T \).

It can be shown that \( u'' \) satisfies all the constraints and

\[
\sum_{k=k_{\text{dis}}-1+\kappa_p}^{k_{\text{dis}}-1+\kappa_f} q_x x'(k) > \sum_{k=k_{\text{dis}}-1+\kappa_p}^{k_{\text{dis}}-1+\kappa_f} q_x x''(k),
\]
where \( x' \) corresponds to the inventory under \( u' \), and \( x'' \) corresponds to the inventory under \( u'' \). Hence, \( u' \) is not optimal, which is a contradiction.

Therefore, under optimal control \( u \), \( x(k) \) satisfies \( x(k) = [0, 0, ..., 0, b_s(s)\alpha_s(k_{\text{dis}} + \kappa'_k - k), 0, ..., 0]^T \) when \( k \in \{k_{\text{dis}}, k_{\text{dis}} + 1, ..., k_{\text{dis}} + \kappa'_k\} \).

**F:** At time \( k_{\text{dis}} - 1 \), nodes in \( \{N_l, N_{l-1}, ..., N_s\} \) work to produce more than the demand to build up inventory; during the disruption, nodes in \( \{N_l, N_{l-1}, ..., N_s\} \) do not work; nodes in \( \{N_{s-1}, N_{s-2}, ..., N_0\} \) work to meet the demand. According to Statement B, we know the inventory is 0 at time \( k_{\text{dis}} - 1 \). At time \( k_{\text{dis}} \), only \( x_s(k_{\text{dis}}) > 0 \). Thus, nodes in \( \{N_l, N_{l-1}, ..., N_s\} \) work to make more than the demand to build up inventory. Since \( x_i(k) = 0 \) for \( i \neq s \), we have

\[
\frac{u(k)}{\alpha_l} = \frac{u_{l-1}(k)}{\alpha_{l-1}} = \ldots = \frac{u_s(k)}{\alpha_s}, \text{ and }
\]

\[
\frac{u_{s-1}(k)}{\alpha_{s-1}} = \frac{u_{s-2}(k)}{\alpha_{s-2}} = \ldots = \frac{u_0(k)}{\alpha_0}.
\]

Due to \( u_{\text{dis}}(k) = 0 \) during the disruption, nodes in \( \{N_l, N_{l-1}, ..., N_s\} \) do not work. In addition, because \( u_0(k) = \alpha_0 \), nodes in \( \{N_{s-1}, N_{s-2}, ..., N_0\} \) still need to work to meet the demand.

Therefore, \( u(k_{\text{dis}} - 1) = [\alpha_0, \alpha_1, ..., \alpha_{s-1}, \alpha_s(1 + k'_{\text{dis}}), \alpha_{s+1}(1 + k'_{\text{dis}}), ..., \alpha_l(1 + k'_{\text{dis}})]^T \), and \( u(k) = [\alpha_0, \alpha_1, ..., \alpha_{s-1}, 0, 0, ..., 0]^T \) when \( k_{\text{dis}} \leq k \leq k_{\text{dis}} + \kappa'_k - 1 \).

### 6.2 Assembly Tree Network Systems

#### 6.2.1 Optimal Operation of Assembly Tree Structures without Disruption

**Theorem 9.** If \( \{N_0, N_1, ..., N_l\} \) is an assembly tree structure with \( N_0 \) as root node, in which any node is not subject to disruption, and \( \alpha \) is the vector in Theorem 7, then the optimal control of these nodes \( \{u_0(k), u_1(k), ..., u_l(k)\} \) under formulated problem satisfies

\[
\frac{u_0(k)}{\alpha_0} = \frac{u_1(k)}{\alpha_1} = \ldots = \frac{u_l(k)}{\alpha_l}.
\]

This implies that \( u(k) \) and \( \alpha \) are linearly dependent. Thus, the optimal control is a Fixed Ratio Operations Control for the nodes \( \{N_0, N_1, ..., N_l\} \).

**Proof.** We prove this by contradiction. We denote the optimal control as \( u'(k) \) with smallest \( k_{\text{min}} \) such that \( u'(k_{\text{min}}) \) and \( \alpha \) are not linearly dependent. So before time \( k \), the optimal controls are all linearly dependent to \( \alpha \). We denote \( \{i_0, i_1, ..., i_l\} \) as the the indices of the positive elements in \( \{b_0, b_1, ..., b_l\} \), respectively. According to Definition 7 and Equation 1.2, we have

\[
x'_i(k + 1) - x'_i(k) = [b_0(i), b_1(i), ..., b_l(i)]u'(k) = 0, \text{ when } k < k_{\text{min}} \text{ and } i \in \{i_1, i_2, ..., i_l\}.
\]
Since $x'(0) = 0$,

$$x_i'(k_{min}) = 0,$$

when $i \in \{i_1, i_2, \ldots, i_l\}$.

According to Lemma 2, each node $N_n$ is in a chain ended with $N_0$. We denote this chain as $(N_n, N_{n+1}, \ldots, N_{n+g}, N_0)$. Based on the Definition 7, $b_r(i_n) = 0$ for any $r$ such that $N_r \notin \{N_n, N_{n+1}\}$. Thus,

$$x_{i_n}'(k_{min} + 1) = [b_0(i_n), b_1(i_n), \ldots, b_l(i_n)]u'(k_{min}) = [b_n(i_n), b_{n+1}(i_n)] \left[ \begin{array}{c} u'_n(k_{min}) \\ u'_{n+1}(k_{min}) \end{array} \right].$$

Since $x_{i_n}'(k_{min} + 1) \geq 0$,

$$[b_n(i_n), b_{n+1}(i_n)] \left[ \begin{array}{c} u'_n(k_{min}) \\ u'_{n+1}(k_{min}) \end{array} \right] \geq 0.$$

Since $b_n(i_n) > 0$ and $b_{n+1}(i_n) < 0$, then

$$\frac{u'_n(k_{min})}{u'_{n+1}(k_{min})} \geq \frac{-b_{n+1}(i_n)}{b_n(i_n)}.$$

Since $[b_n(i_n), b_{n+1}(i_n)] \left[ \begin{array}{c} \alpha_n \\ \alpha_{n+1} \end{array} \right] = 0$, then

$$\frac{-b_{n+1}(i_n)}{b_n(i_n)} = \frac{\alpha_n}{\alpha_{n+1}}.$$

Therefore,

$$\frac{u'_n(k_{min})}{u'_{n+1}(k_{min})} \geq \frac{\alpha_n}{\alpha_{n+1}}.$$

Similarly, we have

$$\frac{u'_{n+1}(k_{min})}{\alpha_{n+1}} \geq \frac{u'_{n+2}(k_{min})}{\alpha_{n+2}},$$

$$\frac{u'_{n+2}(k_{min})}{\alpha_{n+2}} \geq \frac{u'_{n+3}(k_{min})}{\alpha_{n+3}},$$

$$\cdots$$

$$\frac{u'_{n+g}(k_{min})}{\alpha_{n+g}} \geq \frac{u'_0(k_{min})}{\alpha_0}.$$

Thus,

$$\frac{u'_n(k_{min})}{\alpha_n} \geq \frac{u'_0(k_{min})}{\alpha_0}, \quad \forall n \in \{1, 2, \ldots, l\}.$$

Moreover, as $u'(k_{min})$ and $\alpha$ are not dependent, there exists at least one $n \in \{1, 2, \ldots, l\}$ such that

$$\frac{u'_n(k_{min})}{\alpha_n} > \frac{u'_0(k_{min})}{\alpha_0}.$$
Then, we can select a new control \( u''(k_{\min}) \) such that
\[
\frac{u''(k_{\min})}{\alpha_n} = \frac{u'_0(k_{\min})}{\alpha_0}, \forall n \in \{0, 1, 2, ..., l\}.
\]

Therefore,
\[
u'(k_{\min}) > u''(k_{\min}).
\]

And we select \( u''(k_{\min} + 1) = u'(k_{\min} + 1) + u'(k_{\min}) - u''(k_{\min}), \) and \( u''(k) = u'(k) \) when \( k \neq k_{\min} \) and \( k \neq k_{\min} + 1 \). Therefore, \( u''(k) \geq 0 \), and
\[
\sum_k q_u u'(k) \geq \sum_k q_u u''(k)
\]

(6.2)

Since \( x_i''(k_{\min} + 1) = [b_0(i), b_1(i), ..., b_l(i)]u''(k_{\min}) = 0 \) when \( i \in \{i_1, i_2, ..., i_l\} \), then
\[
x_i'(k_{\min} + 1) \geq 0 = x_i''(k_{\min} + 1).
\]

And there exists at least one \( i \in \{i_1, i_2, ..., i_l\} \) such that
\[
x_i'(k_{\min} + 1) > 0 = x_i''(k_{\min} + 1).
\]

Since \( u''_m(k_{\min}) = u'_m(k_{\min}) \), then
\[
x'_m(k_{\min} + 1) = x''_m(k_{\min} + 1).
\]

Also,
\[
x_i'(k_{\min} + 2) = x_i'(k_{\min}) + [b_0(i), b_1(i), ..., b_l(i)](u'(k_{\min}) + u'(k_{\min} + 1))
= x_i''(k_{\min}) + [b_0(i), b_1(i), ..., b_l(i)](u''(k_{\min}) + u''(k_{\min} + 1))
= x_i''(k_{\min} + 2).
\]

And \( x_i'(k) = x_i''(k) \) when \( i \notin \{i_1, i_2, ..., i_l\} \).

Therefore,
\[
x'(k) = x''(k) \text{ when } k \neq k_{\min} + 1,
\]
\[
x'(k_{\min} + 1) > x''(k_{\min} + 1) \geq 0,
\]

Hence,
\[
\sum_k q_x x'(k) > \sum_k q_x x''(k).
\]

(6.3)

According to Equations 6.2 and 6.3, \( u'(k) \) is not optimal. This is a contradiction. So for the optimal control \( u(k) \), it satisfies that
\[
\frac{u_0(k)}{\alpha_0} = \frac{u_1(k)}{\alpha_1} = ... = \frac{u_l(k)}{\alpha_l}.
\]
6.2.2 Optimal Operation of Assembly Tree Network Systems

An assembly tree network system is made up of \( \{N_0, N_1, \ldots, N_l\} \) with \( N_0 \) as the root node. Denote \( N_{\text{dis}} \) as the disrupted node. Then the optimal control of this tree system can be obtained from the solution of a serial network system, to which the assembly tree network system can be transformed by aggregation.

**Disrupted chain**

According to Lemma 2, there exists a unique chain begins with \( N_{\text{dis}} \) and ends with \( N_0 \). This chain is the disrupted chain, which is denoted as \( (N_{\text{dis}}, N_{n_1}, N_{n_2}, \ldots, N_{n_g}, N_0) \).

**Subtree**

**Definition 13.** We use \( N_{i,p} \) to denote the node such that:

1. \( N_{i,p} \notin \{N_{\text{dis}}, N_{n_1}, N_{n_2}, \ldots, N_{n_g}, N_0\} \), and
2. \( N_{i,p} \) is an input node of some node \( N_i \in \{N_{\text{dis}}, N_{n_1}, N_{n_2}, \ldots, N_{n_g}, N_0\} \).

**Lemma 3.** For each node \( N_r \notin \{N_{\text{dis}}, N_{n_1}, N_{n_2}, \ldots, N_{n_g}, N_0\} \), there exists a unique node \( N_{i,p} \) such that \( N_r \) is in some chain ends with \( N_{i,p} \).

**Proof.** According to Lemma 2 for each node \( N_r \notin \{N_{\text{dis}}, N_{n_1}, N_{n_2}, \ldots, N_{n_g}, N_0\} \), there exists a unique chain begins with \( N_r \) and ends with \( N_0 \). We can denote this chain as 

\[
(N_r, N_{r_1}, N_{r_2}, \ldots, N_{r_m}, N_0).
\]

Notice that \( N_0 \) is \( N_{m+1} \), and \( N_r \) is \( N_0 \). Compare it with \( (N_{\text{dis}}, N_{n_1}, N_{n_2}, \ldots, N_{n_g}, N_0) \). We denote \( j \) as the smallest index such that \( N_{r_j} \in \{N_{\text{dis}}, N_{n_1}, N_{n_2}, \ldots, N_{n_g}, N_0\} \). Therefore \( j \) is unique for \( N_r \), and

1. \( N_{r_{j-1}} \notin \{N_{\text{dis}}, N_{n_1}, N_{n_2}, \ldots, N_{n_g}, N_0\} \), and
2. \( N_{r_{j-1}} \) is the input node of \( N_{r_j} \in \{N_{\text{dis}}, N_{n_1}, N_{n_2}, \ldots, N_{n_g}, N_0\} \).

Thus, \( N_{r_{j-1}} \) is some \( N_{i,p} \). Therefore, there exists a unique node \( N_{i,p} \) such that \( N_r \) is in some chain ends with \( N_{i,p} \).

Consider all the nodes which are in some chains ending with \( N_{i,p} \), as shown in Figure 6.2. We denote them as \( \{N_{i,p,1}, N_{i,p,2}, \ldots, N_{i,p,r}\} \). According to Definition 7

\[
\{N_{i,p,1}, N_{i,p,2}, \ldots, N_{i,p,r}\} \cap \{N_{\text{dis}}, N_{n_1}, N_{n_2}, \ldots, N_{n_g}, N_0\} = \emptyset.
\]

Consider another set \( \{N'_{i,p',1}, N'_{i,p',2}, \ldots, N'_{i,p',r'}\} \), which is the set of all the nodes that are in some chains ending with \( N'_{i,p'} \), and \( N'_{i,p'} \) satisfies Definition 13. Then according to Lemma 3 \( N_{i,p} \) is unique for any node in \( \{N_{i,p,1}, N_{i,p,2}, \ldots, N_{i,p,r}\} \). Thus, if \( N_{i,p} \) is not \( N_{i,p} \), then

\[
\{N_{i,p,1}, N_{i,p,2}, \ldots, N_{i,p,r}\} \cap \{N'_{i,p',1}, N'_{i,p',2}, \ldots, N'_{i,p',r'}\} = \emptyset.
\]

60
According to Theorem 3, \( \{N_{i,p,1}, N_{i,p,2}, ..., N_{i,p,r}\} \cup \{N_{i,p}\} \) is an assembly tree structure, with \( N_{i,p} \) as its the root node. According to Theorem 9, 
\[
\frac{u_{i,p,1}(k)}{\alpha_{i,p,1}} = \frac{u_{i,p,2}(k)}{\alpha_{i,p,2}} = \ldots = \frac{u_{i,p,r}(k)}{\alpha_{i,p,r}} = \frac{u_{i,p}(k)}{\alpha_{i,p}}.
\]

Therefore, for each node \( N_{i,p,r} \not\in \{N_{\text{dis}}, N_{n_1}, N_{n_2}, ..., N_{n_g}, N_0\} \), there exists a unique node \( N_{i,p} \) satisfying Definition 13 such that
\[
\frac{u_{i,p,r}(k)}{\alpha_{i,p,r}} = \frac{u_{i,p}(k)}{\alpha_{i,p}}. \tag{6.4}
\]

**Theorem 10.** For any node \( N_i \in \{N_{\text{dis}}, N_{n_1}, N_{n_2}, ..., N_{n_g}, N_0\} \) and its input node \( N_{i,p} \not\in \{N_{\text{dis}}, N_{n_1}, N_{n_2}, ..., N_{n_g}, N_0\} \), their operations in optimal control satisfy:
\[
\frac{u_{i,p}(k)}{\alpha_{i,p}} = \frac{u_i(k)}{\alpha_i}.
\]

**Proof.** We prove this by contradiction. We denote the optimal control as \( u' \neq u \). Thus, there exists at least one node \( N_i \in \{N_{\text{dis}}, N_{n_1}, N_{n_2}, ..., N_{n_g}, N_0\} \) and its input node \( N_{i,p} \) and a smallest \( k_1 \) such that
\[
\frac{u_{i,p}(k_1)}{\alpha_{i,p}} \neq \frac{u_i(k_1)}{\alpha_i}.
\]

We denote the product produced by \( N_{i,p} \) as product\(_{i,p}\). Since \( k_1 \) is the smallest and \( x'(0) = 0 \), we know that \( x'_{i,p}(k) = 0 \) when \( k \leq k_1 \). Since \( x'_{i,p}(k_1 + 1) \geq 0 \), we have
\[
\frac{u'_{i,p}(k_1)}{\alpha_{i,p}} > \frac{u_i'(k_1)}{\alpha_i}.
\]
And then \( x'_{i,p}(k_1 + 1) > 0 \).

Consider all the nodes which are in some chains ending with \( N_{i,p} \). We denote them as \( \{N_{i,p,1}, N_{i,p,2}, ...\} \).
unique nation 6.4, and Theorem 10, for any node $N_{i,p}$, then, according to Theorem 3, the optimal control of this tree satisfies:

$$\frac{u'_{i,p,1}(k)}{\alpha_{i,p,1}} = \frac{u'_{i,p,2}(k)}{\alpha_{i,p,2}} = \ldots = \frac{u'_{i,p,r}(k)}{\alpha_{i,p,r}} = \frac{u'_i(k)}{\alpha_i}.$$ 

Then we select another control $u''$ such that

1. $u''_j(k) = u'_j(k)$ when $j \notin \{(i, p, 1), (i, p, 2), \ldots, (i, p, r), (i, p)\}$, and
2. $u''_j(k_1) = \alpha_j \frac{u'(k_1)}{\alpha_i}$ when $j \in \{(i, p, 1), (i, p, 2), \ldots, (i, p, r), (i, p)\}$, and
3. $u''_j(k_1 + 1) = u'_j(k_1 + 1) + u'_j(k_1) - \alpha_j \frac{u'(k_1)}{\alpha_i}$ when $j \in \{(i, p, 1), (i, p, 2), \ldots, (i, p, r), (i, p)\}$, and
4. $u''_j(k) = u'_j(k)$ when $k \neq k_1, k \neq k_1 + 1$, and $j \in \{(i, p, 1), (i, p, 2), \ldots, (i, p, r), (i, p)\}$.

Thus, $u''(k) \geq 0$. $x''$ satisfies

1. $x''_j(k) = x'_j(k)$ when $j \neq (i, p)$, and
2. $x''_{i,p}(k_1 + 1) = 0$, and
3. $x''_{i,p}(k) = x'_{i,p}(k)$ when $k \neq k_1 + 1$.

Thus, $x''_{i,p}(k_1 + 1) < x'_{i,p}(k_1 + 1)$. Therefore,

$$\sum_{k' = k_1 - \kappa_p}^{k_1 + \kappa_f} q_u u'(k') = \sum_{k' = k_1 - \kappa_p}^{k_1 + \kappa_f} q_u u''(k'),$$

$$\sum_{k' = k_1 - \kappa_p}^{k_1 + \kappa_f} q_x x'(k') > \sum_{k' = k_1 - \kappa_p}^{k_1 + \kappa_f} q_x x''(k').$$

Hence, $u'$ is not optimal. This is a contradiction. Therefore, the optimal control satisfy

$$\frac{u_{i,p}(k)}{\alpha_{i,p}} = \frac{u_i(k)}{\alpha_i}.$$

Optimal control of assembly tree network systems According to Lemma 3, Equation 6.4, and Theorem 10, for any node $N_{i,p,r} \notin \{N_{dis}, N_{n_1}, N_{n_2}, \ldots, N_{n_g}, N_0\}$, there exists a unique $N_i \in \{N_{dis}, N_{n_1}, N_{n_2}, \ldots, N_{n_g}, N_0\}$ such that

$$\frac{u_{i,p,r}(k)}{\alpha_{i,p,r}} = \frac{u_i(k)}{\alpha_i}. \quad (6.5)$$
Denote \( \{N_{i,1}, N_{i,2}, \ldots, N_{i,p}\} \) as the input nodes of \( N_i \in \{N_{dis}, N_{n_1}, N_{n_2}, \ldots, N_{n_g}, N_0\} \), and

\[
\{N_{i,1}, N_{i,2}, \ldots, N_{i,p}\} \cap \{N_{dis}, N_{n_1}, N_{n_2}, \ldots, N_{n_g}, N_0\} = \emptyset.
\]

Then, \( \{N_{i,1,1}, N_{i,1,2}, \ldots, N_{i,1,r_1}\} \cup \{N_{i,1}\} \cup \{N_{i,2,1}, N_{i,2,2}, \ldots, N_{i,2,r_2}\} \cup \{N_{i,2}\} \cup \ldots \cup \{N_{i,p,1}, N_{i,p,2}, \ldots, N_{i,p,r_p}\} \cup \{N_{i,p}\} \cup \{N_i\} \) are a set of Nodes under Fixed Operations Ratio Control. By aggregation, they can form a single node \( N_i^a \).

Therefore, the whole assembly tree network system can be transformed to a serial network system \( (N_{dis}^a, N_{n_1}^a, N_{n_2}^a, \ldots, N_{n_g}^a, N_0^a) \). From the solution of serial network systems, we can obtain the optimal control of \( (N_{dis}^a, N_{n_1}^a, N_{n_2}^a, \ldots, N_{n_g}^a, N_0^a) \). Then, with disaggregation approach, the optimal control of each node in the original system can be calculated.

### 6.3 Conclusion

In this chapter, we solve the optimal operations for serial network systems and assembly tree network systems in discrete time domain. In serial network systems, storage will be built up at the node with the lowest normalized unit storage cost in the cycle right before disruption. For assembly tree network systems, it shows that they can be aggregated into serial network systems. Then, the results of serial network systems can be applied to the aggregated systems of assembly tree network systems.
Chapter 7

Optimal Control of Serial Network Systems in Continuous Time Domain with the Assumption of Downstream Buildup

In this chapter, we conduct analysis on a certain category of serial network systems in continuous time domain. We start the analysis with a special class of serial network systems in which the nodes have Decreasing Storage Costs and Decreasing Capacities with their positions. We refer to these as DSCDC network systems. Then, a method is introduced to change general serial network systems into DSCDC network systems. The results from the DSCDC network system for the optimal operation for the disruption can then be mapped to the operation of the general serial network system. The analysis and results are discussed also in [21].

7.1 Serial Network Systems with Decreasing Storage Costs and Decreasing Capacities

Before solving the problem of general serial network systems, in this section we will first analyze a special class of serial network systems that we refer to as Decreasing Storage Costs and Decreasing Capacity (DSCDC) network systems. Although such DSCDC network systems may not be common in practice, in the later sections of this paper, we will show that more complex serial network systems found in practice can be transformed into a DSCDC serial network system, and the results of DSCDC network systems still work when considering control policies that only build up inventory downstream of the node to be disrupted. Before defining DSCDC network systems formally, we must introduce some notation.

The DSCDC network system is shown in Figure 7.1. We can now formally define the DSCDC network system, which satisfies the following statements:
• **DSCDC1.** The system is a Serial Network system where both the capacity and unit storage cost of each node are decreasing with its position from the initial node to the final node, i.e. for any $i \in \{l, l-1, ..., 2\}$, $q_x(N_i) > q_x(N_{i-1})$, and $c(N_i) > c(N_{i-1})$. Also, the capacity of the final node is 1, and the unit storage cost of the product of final node is 0. That is $c(N_0) = d = 1$, and $q_x(N_0) = 0$.

• **DSCDC2.** Every node $N_i$ (excepting the initial and final nodes) has unit consuming rate of product $i+1$ and unit production rate of product $i$, i.e., $b_i = [0, ..., 0, 1, -1, 0, ..., 0]^T$, in which $b_{i_{in}} = -1$ and $b_{i_{out}} = 1$. For the initial node $N_l$, $b_l = [0, 0, ..., 0, 1]^T$, indicating it has unit production rate of product $l$ and consumes no other products. For the final node $N_0$, $b_0 = [-1, 0, ..., 0]^T$, indicating it has unit consuming rate of product $1$ and produces no other products.

• **DSCDC3.** Unit operation requires unit resource. Every node (excepting the final node) requires a single unit resource to produce each output unit product. That is, $R$ is the identity matrix of $l+1$ dimension.

• **DSCDC4.** The demand is 1. That is, $d(t) = d = 1$, for $t \geq 0$.

• **DSCDC5.** The disruption happens at the initial node $N_l$, i.e. $N_l$ is $N_{dis}$.

• **DSCDC6.** There is no initial inventory beyond running stock: $x_i(t) = 0$, $\forall i$, for $t \leq 0$.

We now analyze this system over three time periods: the nominal-operation period, the pre-disruption period, and the disruption period. The **nominal-operation period** is when $t < t_{dis} - \tau_f$. In this period, the disruption does not happen and is not yet anticipated. The system carries out nominal operation to satisfy the demand and does not take into account the future disruption. The **pre-disruption period** is the time frame from when the disruption is anticipated to when the disruption happens, i.e. $t \in [t_{dis} - \tau_f, t_{dis})$. The **disruption period** is the time frame from when the disruption happens to its end, i.e. $t \in [t_{dis}, t_{dis} + \tau_{dis}]$. When $t > t_{dis} + \tau_{dis}$, we can consider it as another nominal operation period, since by Assumption 7, the next disruption will not happen before $t_{dis} + \tau_{dis} + \tau_f$, and by the No Backorder Constraint, any lost demand from the disruption period cannot be made up in the future. These periods are shown by Figure 7.2.
7.1.1 Operation During the Nominal-operation Period

The following lemma applies to the nominal operation period.

**Lemma 4.** Consider some period \([t_1, t_2]\) of Nominal-operation, where \(t_2 < t_1 + \tau_f\). If the inventory at \(t_1\) is 0, i.e., \(x(t_1) = 0\), then the following statements are true for any optimal operation rate \(u(t)\) for any \(t \in [t_1, t_2]\):

1. The optimal operation rate of the final node \(N_0\) at \(t\) equals to the demand, i.e., \(u_0(t) = d = 1\).
2. The inventory at \(t\) equals to 0, i.e., \(x(t) = 0\).

**Proof.** In the optimization problem, the primary objective is to minimize the lost demand cost. Since \(Du(t) \leq d(t)\), then \(\int_{t_1}^{t_1 + \tau_f} [d(t') - Du(t')]dt' \geq 0\). Thus, the minimum value of the lost demand cost is 0, which occurs when delivery of final products equals demand. Since \(Du(t) = u_0(t)\), then for minimum cost we want \(u_0(t) = d(t) = 1\).

Next we consider minimizing the storage cost and operation cost. Since \(x(t) = 0\), then for any time \(t \in [t_1, t_2]\), the total cumulative production at a node \(j\) is the sum of cumulative delivered demand and the inventory accumulated at each downstream node (including node \(j\)). This gives:

\[
\int_{t_1}^{t} u_j(t')dt' = (t - t_1) + \sum_{i=1}^{j} x_i(t), \ \forall j \in \{1, 2, ..., l\}. \tag{7.1}
\]

The total cost over the period is

\[
\int_{t_1}^{t_2} [q_u u(t') + q_x x(t')]dt'.
\]

The second term is minimized when \(x(t) = 0\), which by equation (7.1) and \(u_0(t) = 1\) will make the minimum of the first term and the minimum total operation cost when \(u_j(t) = d = 1\) for any \(j \in \{1, 2, ..., l\}\).

7.1.2 Properties of Operation During the Disruption Period

At time \(t_{\text{dis}}\), we assume the inventory storage is \(x(t_{\text{dis}})\). During the disruption, the system consumes the storage to meet the demand. Since any node has lower storage cost than its upstream nodes, we can conclude a policy of the operations of all the nodes. In order to have the lowest storage cost, \(\{N_l, N_{l-1}, ..., N_1\}\) will work at their capacities to transfer the storage towards the end node \(N_0\).

This subsection mainly discusses the property of the optimal operation during the disruption period with a given \(x(t_{\text{dis}})\) as the inventory distribution at \(t_{\text{dis}}\). With this property, we formulate \(u(t)\) in the disruption period as a function of \(x(t_{\text{dis}})\). Then, we formulate the total storage cost during the disruption period as a function of \(x(t_{\text{dis}})\) based on this lemma. After that, we do the similar analysis over the time frame of pre-disruption period. We formulate the operation and total storage cost during the pre-disruption period as functions of \(x(t_{\text{dis}})\).
Thus, the optimization problem is changed into finding optimal $x(t_{\text{dis}})$ to minimize the total storage cost over both pre-disruption and disruption period.

Before we present this result, we note that at the disruption start $t_{\text{dis}}$, the inventory built up in the system will not exceed the inventory necessary to meet the demand over the disruption period, thus $\sum_{i=0}^{l} x_i(t_{\text{dis}}) \leq d_{\text{dis}}$ and $x(t_{\text{dis}} + \tau_{\text{dis}}) = 0$. Otherwise, any extra inventory at time $t_{\text{dis}}$ would result in unnecessary operation cost and storage cost.

Based on the definition of DSCDC network system, the disruption happens at $N_l$. Therefore, $u_l(t) = 0$ for $t_{\text{dis}} \leq t \leq t_{\text{dis}} + \tau_{\text{dis}}$. Since $x(t_{\text{dis}} + \tau_{\text{dis}}) = 0$, then every other node must process all inventory upstream of it. This gives us:

$$\int_{t_{\text{dis}}}^{t_{\text{dis}} + \tau_{\text{dis}}} u_j(t) \, dt = \begin{cases} 0, & \text{for } j = l, \\
\sum_{i=j+1}^{l} x_i(t_{\text{dis}}), & \text{for } j \in \{0, \ldots, l - 1\}. \end{cases}$$

(7.2)

Thus, the total operation cost $\int_{t_{\text{dis}}}^{t_{\text{dis}} + \tau_{\text{dis}}} a_q u(t') \, dt'$ is determined by $x(t_{\text{dis}})$. This leads to Lemma 5 presented below:

**Lemma 5.** Consider a DSCDC system such that a disruption occurs at time $t_{\text{dis}}$ with inventory storage $x(t_{\text{dis}})$ satisfying $\sum_{i=0}^{l} x_i(t_{\text{dis}}) \leq d_{\text{dis}}$. Define

$$\tau_{j+1} := \frac{\sum_{i=j+1}^{l} x_i(t_{\text{dis}})}{c(N_j)}$$

for $0 \leq j \leq l - 1$. (7.3)

The optimal operation rate $u(t)$ satisfies:

$$u_l(t) = 0, \text{ for } t \in [t_{\text{dis}}, t_{\text{dis}} + \tau_{\text{dis}}], \text{ and for each } j \in \{0, 1, 2, \ldots, l - 1\} :$$

$$u_j(t) = \begin{cases} c(N_j), & \text{when } t_{\text{dis}} \leq t \leq t_{\text{dis}} + \tau_{j+1}, \\
0, & \text{when } t_{\text{dis}} + \tau_{j+1} < t \leq t_{\text{dis}} + \tau_{\text{dis}}. \end{cases}$$

(7.4)

This lemma shows that (a) the optimal operation rate of the initial node $N_l$ at $t = 0$ over the period $t_{\text{dis}} \leq t \leq t_{\text{dis}} + \tau_{\text{dis}}$, and (b) for any node $N_j$ downstream of $N_l$, the optimal operation rate of $N_j$ over the interval $t_{\text{dis}} \leq t \leq t_{\text{dis}} + \tau_{j+1}$ equals to the capacity of $N_j$, and the operation rate of $N_j$ over the interval $t_{\text{dis}} + \tau_{j+1} < t \leq t_{\text{dis}} + \tau_{\text{dis}}$ equals to 0.

**Proof.** We consider $t_{\text{dis}} \leq t \leq t_{\text{dis}} + \tau_{\text{dis}}$ in this lemma. Therefore, our secondary objective function is the integral interval $[T_1, T_2] = [t_{\text{dis}} - \tau_f, t_{\text{dis}} + \tau_{\text{dis}}].$ Since the $u(t) \in [t_{\text{dis}} - \tau_f, t_{\text{dis}} + \tau_{\text{dis}}]$ does not change the integral result from $t_{\text{dis}} - \tau_f$ to $t_{\text{dis}}$, we only need to consider the objective function as from $t_{\text{dis}}$ to $t_{\text{dis}} + \tau_{\text{dis}}$. Thus, we only need to consider our secondary objective function as the total storage cost, which is $C_{x,dr-\text{dis}} = \int_{t_{\text{dis}}}^{t_{\text{dis}} + \tau_{\text{dis}}} q_x x(t') \, dt'$.

Next, we prove that for the optimal $u(t)$, $u_j(t) = c(N_j)$ whenever $x_{j+1}(t) > 0$, where $j \in \{0, 1, 2, \ldots, l - 1\}$. Since $q_x(N_{j+1}) > q_x(N_j)$, the operation of $N_j$ pushing inventory from product $j+1$ to product $j$ can reduce the inventory storage cost. The more product $j+1$ is pushed to product $j$, the lower storage cost can be achieved. Thus, $N_j$ should operate at its highest rate to minimize the storage cost, i.e. $u_j(t) = c(N_j)$.

For any $j \in \{1, 2, \ldots, l - 1\}$, since $c(N_j) > c(N_{j-1})$, then $x_j(t)$ will not decrease until $x_{j+1}(t) = 0$. Therefore, when $x_j(t) = 0$, then all the products upstream of $N_{j-1}$ have passed $N_{j-1}$. Thus, the duration from $t_{\text{dis}}$ to the time when product $j$ is consumed to 0 is $\tau_j$, as presented in Equation (7.3).
The above lemma shows that $N_j$ will operate at capacity as long as there is inventory $x_{j+1}(t) > 0$ to process, and there will be such inventory for duration $\tau_{j+1}$.

With Equation (7.2), we can formulate the total operation cost during the disruption period as follows:

$$C_{u,dr-dis} := \int_{t_{dis}}^{t_{dis}+\tau_{dis}} q_u u(t') dt' = \sum_{j=0}^{l-1} \sum_{i=j+1}^{l} x_i(t_{dis})].$$

(7.5)

Next, we are going to determine the storage cost when $t_{dis} \leq t \leq t_{dis} + \tau_{dis}$. As shown in Lemma 5 during the disruption period, each node $N_i$ in $\{N_{i-1}, N_{i-2}, ..., N_1\}$ will work at its highest rates to push inventory downstream, until the inventory amount upstream of it $x_{i+1}(t)$ becomes 0. For product $t$, $N_{t-1}$ will work at its highest rate, $c(N_{t-1})$, to transfer the inventory $x_i(t_{dis})$ to product $t_{dis} + \tau_{dis} + \tau_{dis}$ until $x_i$ becomes zero. This results in a linear decrease in $x_i(t)$ with the total time to transfer all the product $t$ as $\tau_t$. The storage cost of product $t$ is then equal to:

$$\frac{1}{2} \cdot q_x(N_i) \cdot x_i(t_{dis}) \cdot \tau_t = \frac{1}{2} \cdot q_x(N_l) \cdot [\frac{x_i(t_{dis})}{c(N_{l-1})}]^2.$$ 

For product $j$, where $j \in \{l-1, l-2, ..., 1\}$, we note that node $N_j$ is adding product at the rate of $c(N_j)$ for duration $\tau_{j+1}$. On the other hand, node $N_{j-1}$ is reducing it at the rate of $c(N_{j-1})$ for duration $\tau_j$, where $c(N_j) > c(N_{j-1})$ and $\tau_{j+1} \leq \tau_j$. As shown in Figure 7.3, inventory increases during the period $t_{dis}$ to $t_{dis} + \tau_{dis}$, and decreases from the period $t_{dis} + \tau_{dis}$ to $t_{dis} + \tau_j$.

![Figure 7.3: Inventory of the node increasing and then decreasing during a disruption.](image)

This then gives us:

$$x_j(t) = \begin{cases} x_j(t_{dis}) + [c(N_j) - c(N_{j-1})](t - t_{dis}), & \text{when } t \in [t_{dis}, t_{dis} + \tau_{j+1}], \\ x_j(t_{dis}) + c(N_j)\tau_{j+1} - c(N_{j-1})(t - t_{dis}), & \text{when } t \in (t_{dis} + \tau_{j+1}, t_{dis} + \tau_j]. \end{cases}$$

Evaluating this in the storage cost integrals for each node $N_j$ ($j \neq l$) eventually gives

$$q_x(N_j) \cdot \int_{t_{dis}}^{t_{dis}+\tau_j} x_j(t) dt = q_x(N_j) \cdot \left[ \frac{1}{2} c(N_{j-1})(\tau_j)^2 - \frac{1}{2} c(N_j)(\tau_{j+1})^2 \right]$$

$$= q_x(N_j) \cdot \frac{1}{2} \left\{ \frac{\sum_{i=j}^{l} x_i(t_{dis})^2}{c(N_{j-1})} - \frac{\sum_{i=j+1}^{l} x_i(t_{dis})^2}{c(N_j)} \right\}.$$
The total storage cost during the disruption period, given \( q_x(N_0) = 0 \), is:

\[
\sum_{j=1}^{l} \left[ q_x(N_j) \cdot \int_{t_{\text{dis}}}^{t_{\text{dis}}+\tau_j} x_j(t) \, dt \right] = \frac{1}{2} \sum_{j=1}^{l} \left[ q_x(N_j) - q_x(N_{j-1}) \right] \left\{ \frac{\left[ \sum_{i=j}^{l} x_i(t_{\text{dis}}) \right]^2}{c(N_{j-1})} \right\}. 
\]  

(7.6)

With Lemma 5, we have formulated the optimal operation and total storage cost during the disruption period as a function of \( x(t_{\text{dis}}) \). Then the following Lemma 6 will be used to determine the optimal \( x(t_{\text{dis}}) \).

**Lemma 6.** Given the DSCDC inventory cost structure, if a portion \( \Delta x \) of stored inventory at node \( N_g \) at time \( t_{\text{dis}} \) is instead stored at node \( N_h \) downstream of \( N_g \), then the total storage cost over the disruption period \( [t_{\text{dis}}, t_{\text{dis}} + \tau_{\text{dis}}] \) will be reduced.

Consider a DSCDC system such that a disruption occurs at time \( t_{\text{dis}} \) with inventory storage vector \( x(t_{\text{dis}}) \). Let the total storage cost under optimal operation (as per Lemma 5) during the disruption period be denoted as \( C_{x,dr-\text{dis}} \). Consider any two nodes \( N_g \) and \( N_h \) such that \( N_g \) is upstream of \( N_h \), and an alternative storage \( x'(t_{\text{dis}}) \) such that:

\[
x'_i(t_{\text{dis}}) = \begin{cases} 
 x_i(t_{\text{dis}}) - \Delta x, & \text{if } i = g, \\
 x_i(t_{\text{dis}}) + \Delta x, & \text{if } i = h, \\
 x_i(t_{\text{dis}}), & \text{else,}
\end{cases}
\]

where \( 0 < \Delta x \leq x_g(t_{\text{dis}}) \). Let \( C_{x',dr-\text{dis}} \) be the total storage cost during the disruption period under the optimal operation policy (as per Lemma 5) given \( x'(t_{\text{dis}}) \). Then, Lemma 6 shows \( C_{x',dr-\text{dis}} < C_{x,dr-\text{dis}} \).

**Proof.** We note that there is no cost difference for nodes downstream of \( h \) or upstream of \( g \). Using Equation (7.6), we then have

\[
C_{x,dr-\text{dis}} - C_{x',dr-\text{dis}} = \frac{1}{2} \sum_{j=h+1}^{g} \left[ q_x(N_j) - q_x(N_{j-1}) \right] \left\{ \frac{\left[ \sum_{i=j}^{l} x_i(t_{\text{dis}}) \right]^2 - \left[ \sum_{i=j}^{l} x'_i(t_{\text{dis}}) \right]^2}{c(N_{j-1})} \right\}.
\]

Since \( x'_g(t_{\text{dis}}) < x_g(t_{\text{dis}}) \), since under DSCDC, downstream nodes have lower cost,

\[
\sum_{i=j}^{l} x'_i(t_{\text{dis}}) < \sum_{i=j}^{l} x_i(t_{\text{dis}}), \text{ for } j \in \{ h + 1, h + 2, \ldots, g \}.
\]

Since \( q_x(N_j) > q_x(N_{j-1}) \) (as in DSCDC1), then \( C_{x,dr-\text{dis}} - C_{x',dr-\text{dis}} > 0 \). Therefore, Lemma 6 is proved.

The above lemma states that having inventory stored at a lower storage cost node at the stage of disruption event will result in a lower storage cost over the disruption period. In the later section, we will use it to determine the optimal \( x(t_{\text{dis}}) \).
7.1.3 Properties of Operation During the Pre-disruption Period

In this subsection, we consider the operation of the system during the interval \( t \in [t_{\text{dis}} - \tau_f; t_{\text{dis}} + \tau_{\text{dis}}] \). It is found that once the primary objective function (which is the lost demand) is minimized, then the total operation cost is a fixed value. Thus, we will only need to consider minimizing the total storage cost in the secondary objective function. We present this property as Lemma 7 below.

**Lemma 7.** Define \( X_{bx} \) as the minimum between (1) the total amount of buildup that can be made at the end of the pre-disruption period and (2) the total amount of the buildup needed at the beginning of the disruption period to avoid lost demand, i.e.,

\[
X_{bx} := \min \{ \tau_f [c(N_l) - d], \tau_{\text{dis}} d \}
\]

If \( u(t) \) is an optimal operation rate for a DSCDC network system, then: (a) the total amount of the buildup at \( t_{\text{dis}} \) is \( \sum_{j=1}^{l} x_j(t_{\text{dis}}) = X_{bx} \), and (b) the total storage cost during the period \( t \in [t_{\text{dis}} - \tau_f, t_{\text{dis}} + \tau_{\text{dis}}] \) is:

\[
\int_{t_{\text{dis}} - \tau_f}^{t_{\text{dis}} + \tau_{\text{dis}}} q_u u(t') dt' = \sum_{j=0}^{l} q_u(N_j)(X_{bx} + \tau_{fd}).
\]

**Proof.** Recall from Lemma 4 that inventory at the start of the pre-disruption period is zero, \( x(t_{\text{dis}} - \tau_f) = 0 \). The inventory at the end of the pre-disruption period is \( |x(t_{\text{dis}})| := \sum_{j=1}^{l} x_j(t_{\text{dis}}) \).

During the pre-disruption period, there is no capacity lost, and the demand can always be satisfied. Therefore, \( u_0(t) = d \) during the pre-disruption period. The inventory accumulated over the period cannot exceed the product produced by the initial node at full capacity less the demand leaving node \( N_0 \) during the pre-disruption period, so \( |x(t_{\text{dis}})| \leq \tau_f[c(N_l) - d] \). Also, as discussed in the previous section, there is no value in building more inventory than necessary to feed demand during the disruption period, so we also have \( |x(t_{\text{dis}})| \leq \tau_{\text{dis}} d \).

\( X_{bx} \) is then defined as the maximum inventory at \( t_{\text{dis}} \) that satisfies both these constraints, \( X_{bx} := \min \{ \tau_f [c(N_l) - d], \tau_{\text{dis}} d \} \).

Since at the beginning of pre-disruption (at \( t_{\text{dis}} - \tau_f \)) and at the end of the disruption period (at \( t_{\text{dis}} + \tau_{\text{dis}} \)) inventory at each node is zero, then

\[
\int_{t_{\text{dis}} - \tau_f}^{t_{\text{dis}} + \tau_{\text{dis}}} u_j(t') dt' = \int_{t_{\text{dis}} - \tau_f}^{t_{\text{dis}} + \tau_{\text{dis}}} u_{j-1}(t') dt'.
\]

Thus,

\[
X_{bx} + \tau_{fd} + 0 = \int_{t_{\text{dis}} - \tau_f}^{t_{\text{dis}} + \tau_{dis}} u_1(t') dt' = \int_{t_{\text{dis}} - \tau_f}^{t_{\text{dis}} + \tau_{dis}} u_1(t') dt' = \ldots = \int_{t_{\text{dis}} - \tau_f}^{t_{\text{dis}} + \tau_{dis}} u_0(t') dt'.
\]

Therefore, the total operation cost during the period when \( t \in [t_{\text{dis}} - \tau_f; t_{\text{dis}} + \tau_{dis}] \) is:

\[
\int_{t_{\text{dis}} - \tau_f}^{t_{\text{dis}} + \tau_{dis}} q_u u(t') dt' = \sum_{j=0}^{l} q_u(N_j)(X_{bx} + \tau_{fd}).
\]
This implies that the optimal total operation cost is not determined by \( u(t) \), as long as we have the constraint of \( \int_{t_{\text{dis}} - \tau_f}^{t_{\text{dis}} + \tau_{\text{dis}}} u_j(t') dt' = X_{b_k} + \tau_f d \). Once the primary objective function is minimized, then the total operation cost is a constant. Then we only need to consider the total storage cost in the secondary objective function. When we check whether a \( u(t) \) is optimal or not, we only need to check whether its corresponding storage cost is minimum.

Lemma 8 below states that if at any time during the pre-disruption period, the optimal operation rate for node \( N_l \) is at least as large as the capacity of \( N_j \), then all the nodes downstream of \( N_j \) will operate at their full capacity.

**Lemma 8.** For a DSCDC network system, given an optimal \( u \), consider any time \( t \) in the pre-disruption period. The following statements are true:

1. \( u_l(t) \geq u_0(t) = c(N_0) = d \),
2. Let \( j \) be the largest index such that \( u_l(t) \geq c(N_j) \). Then:
   (a) \( \forall i \in \{0, 1, 2, \ldots, j\}, u_i(t) = c(N_i) \),
   (b) \( \forall i \in \{j + 1, \ldots, l\}, u_i(t) = u_l(t) \).

Statement 1 means the optimal operation rate of the initial node \( N_l \) is greater than or equal to the demand \( d \), and the optimal operation rate of the final node \( N_0 \) equals to the demand. For the largest index \( j \) such that \( u_l(t) \geq c(N_j) \), statement 2a means the optimal operation rate of nodes downstream of \( N_{j+1} \) all equal to their capacities respectively, and statement 2b means the optimal operation rate of nodes upstream of \( N_j \) all equal to the operation rate of \( N_j \).

**Proof.** During the pre-disruption period, the capacity of each node is at its nominal capacity, which for a DSCDC is at least as much as the demand. Without any loss of capacity, the system can keep \( u_0(t) = d \) in order to meet demand to keep the primary objective function minimized, and all nodes \( N_i \) for \( i > 0 \) will have \( u_i(t) \geq u_0(t) \). This implies statement 1.

Let \( j \) be the largest index such that \( u_l(t) \geq c(N_j) \). If \( j > 0 \), since \( u_l(t) \geq c(N_j) \) and \( c(N_j) > c(N_0) = u_0(t) \) by the DSCDC, then there will be inventory buildup in the system. Each node downstream of \( N_j \) will operate at its capacity to push the inventory buildup to the lower storage cost nodes further downstream, which results in statement 2a. Any nodes downstream of \( N_j \) will have \( c(N_i) > u_l(t) \), and these nodes will also push inventory downstream at the maximum rate, \( u_l(t) \), towards lower storage costs. This gives statement 2b.

**Lemma 9.** Given an optimal \( u \) for a DSCDC network system, then \( u_l(t_1) \leq u_l(t_2) \) for any \( t_1 < t_2 \) in the pre-disruption period.

**Proof.** The proof is straightforward, so we simply outline it. During the pre-disruption period, any products entering at \( N_l \) in excess of the demanded products exiting at \( N_0 \) represent inventory accumulation in the system. This inventory will not leave the system until after \( t_{\text{dis}} \).

Building up inventory earlier in the pre-disruption period than necessary would require longer storage time and thus larger inventory storage cost. Thus, lower storage cost is
achieved by postponing inventory accumulation as long as possible, which implies $u(t)$ is non-decreasing over the pre-disruption period.

Figure 7.4: $u(t)$ satisfying Stepped Production Property

Given $x(t_{\text{dis}})$ as the distribution of inventory at $t_{\text{dis}}$, we find a similar optimal policy of operation as Lemma 5. We define the property of the operation during the pre-disruption period under this optimal policy as follows:

**Definition 14. (Stepped Production Property)** $u(t)$ satisfies the Stepped Production Property if:

- **SPP1:** $u(0) = d$, i.e., the final node's operation rate equals to the demand;
- **SPP2:** $u_i(t) \geq u_{i-1}(t)$, $\forall i \in \{1, 2, ..., l\}$, i.e., the upstream node's operation rate is greater than or equal to the downstream node;
- **SPP3:** if $u_i(t) \neq u_{i-1}(t)$, then $u_i(t)$ equals to the node's capacity $c(N_i)$;
- **SPP4:** given any $j$ such that $u_j(t) \neq u_{j-1}(t)$, then $u_i(t) \neq u_{i-1}(t)$ for all $i$ such that $q_x(N_i) < q_x(N_j)$.

Figure 7.4 shows an example $u(t)$ satisfying the Stepped Production Property. As shown, all nodes upstream of node $j$ operate at the capacity $c(N_j)$, and all other nodes downstream of $j$ operate at that node's capacity.

**Lemma 10.** For a DSCDC network system, if $u(t)$ is an optimal operation rate, then $u(t)$ satisfies the Stepped Production Property when $t \in [t_{\text{dis}} - \tau_f, t_{\text{dis}}]$.

**Proof.** Statements SPP1 and SPP2 follow from Lemma 8.

To prove SPP3, let $j$ be the largest index such that $u_l(t) = c(N_j)$. By Lemma 8, $u_l(t) = c(N_j)$ for downstream nodes $i \in \{0, \ldots, j\}$, and $u_i(t) = u_l(t)$ for all upstream nodes $i \in \{j + 1, j + 2, \ldots, l\}$. For SPP3, it only remains to be shown that $u_l(t) = c(N_j)$.

We will prove this by contradiction. Assume there exists a $u'(t)$ that is an optimal operation rate where $u'_l(t) \neq c(N_j)$. Then, there exists a time interval $[t_1, t_1 + \Delta t]$ in the pre-disruption period such that $u'_l(t) > c(N_j)$, $\forall t \in [t_1, t_1 + \Delta t]$. Since $j$ is the largest index such that $u_l'(t) = c(N_j)$, we have $u'_l(t) < c(N_{j+1})$. The buildup of product $\tau_j$ during the period when $t \in [t_1, t_1 + \Delta t]$ is $\int_{t_1}^{t_1 + \Delta t} [u'_l(t) - c(N_j)]dt$. Denote

$$\tau_s = \frac{\int_{t_1}^{t_1 + \Delta t} [u'_l(t) - c(N_j)]dt}{c(N_{j+1}) - c(N_j)}.$$
This is the amount of time the equivalent amount of product could have been built up if nodes upstream of \( j \) were operating at \( c(N_j) \) instead. It can be shown that \( \tau_s < \Delta t \).

Let \( u''_i(t) \) be an alternative operation rate such that

\[
u''_i(t) = u'_i(t), \ \forall i \leq j, \ \text{and} \ \forall t,
\]

and for all \( i > j \),

\[
u''_i(t) = \begin{cases} 
  c(N_j), & \text{when } t_1 \leq t < t_1 + \Delta t - \tau_s, \\
  c(N_{j+1}), & \text{when } t_1 + \Delta t - \tau_s \leq t < t_1 + \Delta t, \\
  u'_i(t), & \text{others.}
\end{cases}
\]

Therefore, the only difference between \( x' \) (under operation \( u' \)) and \( x'' \) (under operation \( u'' \)) is at node \( N_{j+1} \) during \( t_1 < t \leq t_1 + \Delta t \). The difference is that \( x'(t) \) builds at a constant rate of \( u'_i(t) \) over the interval of length \( \Delta t \), whereas \( x''(t) \) delays the buildup to the shorter time period \( \tau_s \). At the beginning and end of this interval \([t_1, t_1 + \Delta t]\), the inventories are equal, but elsewhere in the interval we have \( x'(t) > x''(t) \). Consequently, the storage cost under policy \( u'(t) \) is greater than under \( u''(t) \), which contradicts our statement that \( u'(t) \) was optimal. Thus, we have shown that if \( j \) is the largest index such that \( u_i(t) \geq c(N_j) \), then \( u_i(t) = c(N_j) \).

For Statement SPP4, by SPP3 we have \( u_j(t) \neq u_{j-1}(t) \), which implies that \( u_j(t) = c(N_j) \). For DSCDC network systems, \( q_x(N_i) > q_x(N_{i-1}) \) for all \( i \in \{2, 3, \ldots, l\} \) (note that there is no storage at node \( 0 \)). Thus, \( q_x(N_i) < q_x(N_j) \) for some \( i,j \) implies \( i < j \), which by lemma \ref{lemma:non-decreasing} implies that \( u_i(t) = c(N_i) \). For DSCDC network systems, \( c(N_i) < c(N_j) \) for \( i < j \), so that \( u_i(t) < u_j(t) \). This proves SPP4.

Lemma \ref{lemma:optimal_policy} gives the optimal policy of operation during the pre-disruption period. We will use it to derive the optimal \( u(t) \) during the pre-disruption period given \( x(t_{\text{dis}}) \).

**Lemma 11.** The optimal operation rate of any node in a DSCDC network system is non-decreasing during the pre-disruption period. That is, given the optimal \( u \) for a DSCDC network system, for any \( t_1 \) and \( t_2 \) such that \( t_{\text{dis}} - \tau_f \leq t_1 < t_2 < t_{\text{dis}} \) and any \( j \), then \( u_j(t_1) \leq u_j(t_2) \).

**Proof.** From Lemma \ref{lemma:non-decreasing} since \( u_i(t) \) is non-decreasing during the pre-disruption period, the largest index \( j \) such that \( u_i(t) = c(N_j) \) is also non-decreasing over time. By Lemma \ref{lemma:non-decreasing} this implies \( \forall i \in \{0, \ldots, l\}, u_i(t) \) is nondecreasing over the pre-disruption period.

With Lemmas \ref{lemma:optimal_policy} and \ref{lemma:non-decreasing} we know that for any \( j \) such that \( x_j(t_{\text{dis}}) > 0 \), then \( u_j(t) = c(N_j) \) and \( u_{j-1}(t) = c(N_{j-1}) \) during the buildup period of product \( j \). Thus, the buildup rate of product \( j \) is \( c(N_j) - c(N_{j-1}) \), and the duration of the buildup is \( \frac{x_j(t_{\text{dis}})}{c(N_j) - c(N_{j-1})} \). The operation of \( N_j \) during the pre-disruption period can be presented by the following corollary.

**Corollary 1.** Given the inventory distribution \( x(t_{\text{dis}}) \) for a DSCDC network system, define

\[
\tau_{bx,j} := \frac{x_j(t_{\text{dis}})}{c(N_j) - c(N_{j-1})}.
\]

If \( u(t) \) is the optimal operation rate, then \( u(t) \) satisfies that:

\[
u_0(t) = d, \text{ for } t \in [t_{\text{dis}} - \tau_f, t_{\text{dis}}), \text{ and for each } j \in \{1, 2, \ldots, l\}:
\]

\[
u_j(t) = c(N_j), \text{ for } t \in \left[ t_{\text{dis}} + \tau_f, \tau_{bx,j} \right).
\]
\[ u_j(t) = \begin{cases} u_{j-1}(t), \text{ when } t_{\text{dis}} - \tau_f \leq t < t_{\text{dis}} - \tau_{bx,j}, \\ c(N_j), \text{ when } t_{\text{dis}} - \tau_{bx,j} \leq t < t_{\text{dis}}. \end{cases} \]

This corollary presents the optimal operation of each node given the inventory distribution \( x(t_{\text{dis}}) \). The second equation says that for any \( N_j \) other than \( N_0 \), during the pre-disruption period, its optimal operation rate equals to the operation rate of the next downstream node until \( t_{\text{dis}} - \tau_{bx,j} \), and equals to its capacity \( c(N_j) \) afterwards.

We now determine the operation cost of this optimal policy. Since \( u_0(t) = 1 \) and \( x_j(t_{\text{dis}}) = \int_{t_{\text{dis}} - \tau_f}^{t_{\text{dis}}} [u_j(t') - u_{j-1}(t')] dt', \forall j \in \{1, 2, ..., l\} \). We have:

\[
\int_{t_{\text{dis}} - \tau_f}^{t_{\text{dis}}} u_0(t') dt' = \tau_f, \text{ and } \int_{t_{\text{dis}} - \tau_f}^{t_{\text{dis}}} u_j(t') dt' = \tau_f + \sum_{i=1}^{j} x_i(t_{\text{dis}}), \forall j \in \{1, 2, ..., l\}.
\]

Therefore, the total operation cost during the pre-disruption period can be formulated as:

\[
C_{u, pre-dis} = \int_{t_{\text{dis}} - \tau_f}^{t_{\text{dis}}} q_u u(t') dt' = \tau_f \sum_{j=0}^{l} q_u(N_j) + \sum_{j=1}^{l} [q_u(N_j) \sum_{i=1}^{j} x_i(t_{\text{dis}})]. \tag{7.7}
\]

Based on Lemma \[10\] and Corollary \[1\] we know that for product \( j \), where \( j \in \{l, l-1, ..., 1\} \), its buildup rate before disruption is \( c(N_j) - c(N_{j-1}) \). Thus, the storage cost for product \( j \) during the pre-disruption period is:

\[
q_x(N_j) \int_{t_{\text{dis}} - \tau_f}^{t_{\text{dis}}} x_j(t) dt = q_x(N_j) \int_{0}^{x_j(t_{\text{dis}})} \left[ c(N_j) - c(N_{j-1}) \right] \frac{[x_j(t_{\text{dis}})]^2}{c(N_j) - c(N_{j-1})} dt
\]

\[
= \frac{1}{2} q_x(N_j) \frac{[x_j(t_{\text{dis}})]^2}{c(N_j) - c(N_{j-1})}.
\]

The total storage cost of the pre-disruption period is:

\[
C_{x, pre-dis} = \frac{1}{2} \sum_{j=1}^{l} q_x(N_j) \frac{[x_j(t_{\text{dis}})]^2}{c(N_j) - c(N_{j-1})}. \tag{7.8}
\]

### 7.1.4 Optimal Operation over the Time Frame of Both Pre-disruption and Disruption Periods

We are now reformulating the optimization problem to determine the optimal \( x(t_{\text{dis}}) \). Equations \[7.5\]-\[7.8\] show that the total cost \( C \) is a quadratic function of \( x(t_{\text{dis}}) \).

Define \( v_{j,n} \) as a \( n \)-dimension row vector such that:

\[
v_{j,n}(i) := \begin{cases} 0, \text{ when } 1 \leq i < j; \\ 1, \text{ when } j \leq i \leq n. \end{cases}
\]

Denote \([x_1(t_{\text{dis}}), x_2(t_{\text{dis}}), ..., x_l(t_{\text{dis}})]^T\) as \( x \), and substitute this into Equation \[7.6\] and Equation \[7.8\]. We have:

\[
C_{x,dr-dis} = \frac{1}{2} x^T Q_{x,dr-dis} x, \text{ and } C_{x,pre-dis} = \frac{1}{2} x^T Q_{x,pre-dis} x,
\]

74
where matrices $Q_{x,dr-dis}$ and $Q_{x,pre-dis}$ are defined as:

$$Q_{x,dr-dis} := \sum_{j=1}^{l} [q_x(N_j) - q_x(N_{j-1})] \left( \frac{x_j v_{j,l}}{c(N_{j-1})} \right),$$

$$Q_{x,pre-dis}(i,j) := \begin{cases} \frac{q_x(N_i)}{c(N_i) - c(N_{j-1})}, & \text{when } i = j, \\ 0, & \text{when } i \neq j, \end{cases}$$

where $i, j \in \{1, 2, ..., l\}$.

Consider the operational costs based on Equation (7.5) and Equation (7.7):

$$C_u := C_{u,dr-dis} + C_{u,pre-dis} = \sum_{j=0}^{l} [q_u(N_j) \sum_{i=1}^{l} x_i(t_{dis})] + \tau f \sum_{j=0}^{l} q_u(N_j)$$

$$= q_u(v_{1,l+1})^T v_{1,l} x + \tau f v_{1,l+1}(q_u)^T.$$

Denote $Q_x := Q_{x,dr-dis} + Q_{x,pre-dis}$. Recall that $\sum_{j=1}^{l} x_j(t_{dis}) = X_{bx}$, which can be presented as:

$$v_{1,l} x = X_{bx}.$$ 

Thus, the total storage cost and operation cost over the pre-disruption and disruption period is:

$$C = C_{x,dr-dis} + C_{x,pre-dis} + C_u = \frac{1}{2} x^T Q_x x + q_u(v_{1,l+1})^T X_{bx} + \tau f v_{1,l+1}(q_u)^T.$$

The values of the buildup at the locations downstream of $N_i$ at $t_{dis}$ are $[x_i(t_{dis}), x_{i-1}(t_{dis}), ..., x_1(t_{dis})]$. The sum of these inventories will be:

$$\sum_{j=1}^{i} x_j(t_{dis}) = \int_{t_{dis}-\tau f}^{t_{dis}} [u_i(t) - d] dt.$$ 

Since $u_i(t) \leq c(N_i)$, then

$$\sum_{j=1}^{i} x_j(t_{dis}) \leq [c(N_i) - d] \tau f, \text{ for } i \in \{1, 2, ..., l\}.$$ 

Based on Lemma 5 and Corollary 1, once the optimal $x$ is found, the optimal operation $u$ derived from it can satisfy the Capacity Constraint, Demand Constraint and the Operation Direction Constraint.

To find $x$, a quadratic problem is formulated as:

Minimize: $\frac{1}{2} x^T Q_x x$

Subject to: $v_{1,l} x = X_{bx}$,

$$\langle v_{1,l} - v_{i+1,l}, x \rangle \leq [c(N_i) - d] \tau f, \text{ for } i \in \{1, 2, ..., l\},$$

$$x \geq 0.$$ 

In this quadratic program, the $x$ we solve is $x(t_{dis})$, the inventory distribution at time $t_{dis}$. Thus, we transform our original optimization problem to a quadratic problem. For this DSCDC serial network system, we can simply apply the solution of the quadratic problem to obtain the solution of optimal operation during the pre-disruption and disruption periods.
7.2 Generalizing to General Serial Network Systems

Although the DSCDC serial network system is a special system, its results can be applied for more complex serial network systems. To do this, we introduce normalization and aggregation to transform general serial network systems into the DSCDC serial network systems. Before we introduce normalization, we first define an assumption restricting solutions considered for the general serial network systems as per Definition 8.

**Assumption 11.** In this paper, we assume that products will only be built up downstream of \( N_{\text{dis}} \) in a serial network system.

With this assumption, we don’t consider any operation which builds up products upstream of \( N_{\text{dis}} \). The optimal operation under Assumption 11 is optimal only for the downstream storage. Solutions allowing buildups upstream of the disrupted node will be discussed in the next chapter.

Next, we introduce normalization and aggregation.

### 7.2.1 Normalization

Define \( \alpha := [\alpha_0, \alpha_1, ..., \alpha_l]^T \) as a vector of nominal operation rate, with elements \( \alpha_i \) for \( N_i \) defined by:

\[
\alpha_0 := d, \quad \text{and} \quad \alpha_i := \alpha_{i-1} - \frac{b_{i-1}^0}{b_{i}^0}.
\]

Thus, when \( u(t) = \alpha \), then \( Du(t) = d \) and \( \frac{dx(t)}{dt} = Bu(t) = 0 \).

Then, we use the normalization introduced in Section 5.1, we denote its normalized operation rate at time \( t \) as:

\[
\tilde{u}_i(t) := \frac{u_i(t)}{\alpha_i}.
\]

76

The normalized demand \( \tilde{d} \) is defined as:

\[
\tilde{d} := \frac{d}{\alpha_0} = 1.
\]

The normalized capacity is defined by:

\[
\tilde{c}(N_i) := \begin{cases} 
\frac{c(N_i)}{\alpha_i}, & \text{when } i \in \{1, 2, ..., l\}, \\
1, & \text{when } i = 0.
\end{cases}
\]

Then, the normalized operation vector \( \tilde{b}_i \) satisfies the following:

1. \( \tilde{b}_{in}^i = -1 \) for \( i \in \{0, 1, 2, ..., l - 1\} \), and
2. \( \tilde{b}_{out}^i = 1 \) for \( i \in \{1, 2, ..., l\} \), and
3. all other elements in \( \tilde{b}_i \) are zeros for \( i \in \{0, 1, 2, ..., l\} \).
The normalized unit storage cost is defined by:
\[ \tilde{q}_x(N_i) := q_x(N_i) b_{i}^{\text{out}} \alpha_i, \]
so that the total inventory storage cost keeps the same:
\[ \tilde{q}_x(N_i) \tilde{x}_i(t) = q_x(N_i) x_i(t). \]

### 7.2.2 Locating the s-nodes and c-nodes

A serial network system can be divided into several subsequences. In this paper, some special nodes in the serial network system are denoted with subscripts \(s\) and \(c\), indicating the nodes with lowest storage cost in the subsequences, and the lowest capacity nodes in the subsequences, respectively. Also, We use superscripts \(u\) and \(d\) to label the \(s\)-nodes and \(c\)-nodes which are upstream and downstream of \(N_{\text{dis}}\), respectively. An example is shown by Figure 7.5.

Recall that \(N_i \succ N_j\) means that \(N_i\) is upstream of \(N_j\). \(N_i \succeq N_j\) means that \(N_i\) is upstream of \(N_j\), or \(N_i\) and \(N_j\) are the same node.

Define \(N_{d}^{s_j} := N_0\). Given some \(j > 0\) with \(N_{d}^{s_j} \) defined, we can then define \(N_{u}^{s_j}\) and \(N_{d}^{c_j}\) recursively as follows:

1. Define \(N_{d}^{s_j}\) as the node closest to \(N_{\text{dis}}\) with the lowest normalized unit storage cost among \(\{N | N_{\text{dis}} \succeq N \succ N_{d}^{s_j-1}\}\).
2. Define \(N_{d}^{c_j}\) as the node closest to \(N_{\text{dis}}\) with the lowest normalized capacity among \(\{N | N_{\text{dis}} \succeq N \succ N_{d}^{c_j}\}\).

The recursion terminates when \(N_{d}^{c_j} = N_{\text{dis}}\) or \(N_{d}^{s_j} = N_{\text{dis}}\).

\(N_{u}^{s_1}\) is the node with lowest capacity among \(\{N_1, N_{l-1}, \ldots, N_{\text{dis}}\}\) which satisfies:

1. for any \(N_i\) such that \(N_i \succeq N_{u}^{s_1}\), \(\bar{c}(N_i) \geq \bar{c}(N_{u}^{s_1})\), and
2. for any \(N_i\) such that \(N_{u}^{s_1} \succeq N_i \succeq N_{\text{dis}}\), \(\bar{c}(N_i) > \bar{c}(N_{u}^{s_1})\).

In addition, \(N_{u}^{s_j}\) is the product of \(N_{u}^{s_j}\), and \(N_{d}^{s_j}\) is the product of \(N_{d}^{s_j}\).

The process of locating \(s\)- and \(c\)- nodes is presented in Figure 7.6. We locate \(N_{u}^{s_1}\) by searching the lowest capacity node upstream of \(N_{\text{dis}}\). To locate the \(s\)- and \(c\)- nodes downstream of \(N_{\text{dis}}\), we first locate \(N_{d}^{s_1}\) by searching the lowest storage cost node from \(N_0\) to \(N_{\text{dis}}\). Then we locate \(N_{u}^{s_1}\) by searching the lowest capacity node from \(N_{d}^{s_1}\) to \(N_{\text{dis}}\). Next, we locate \(N_{d}^{s_2}\) by searching the lowest storage cost node from \(N_{d}^{s_1}\) to \(N_{\text{dis}}\). The \(s\)- and \(c\)- nodes are located one after another. The searching direction is towards \(N_{\text{dis}}\). The whole process ends when a \(c\)-node reaches \(N_{\text{dis}}\).
7.2.3 Aggregation

The s-nodes and c-nodes divide the network system into small sections. Next, we will discuss a property that in each section, the optimal operation rates of nodes have a fixed ratio. This property allows us to aggregate them as one node.

Since $N_{c_1}^u$ is the node with lowest capacity upstream of $N_{dis}$. The total inventory buildup at $t_{dis}$ is determined by $\tilde{c}(N_{c_1}^u)$. Similarly as in DSCDC network system, in order to make no lost demand, then $\tilde{u}_0(t) = \tilde{d} = 1$ during the pre-disruption period. The maximum buildup can be made during the pre-disruption period is $\tau_f[\tilde{c}(N_{c_1}^u) - 1]$, and the buildup needed to satisfy the demand during the disruption period is $\tau_{dis}\tilde{d}$. Therefore, the optimal total buildup at $t_{dis}$ is:

$$\tilde{X}_{bx} := \min\{\tau_f[\tilde{c}(N_{c_1}^u) - 1], \tau_{dis}\tilde{d}\}.$$ 

In addition,

$$\int_{t_{dis}-\tau_f}^{t_{dis}} [\tilde{u}_{dis}(t') - \tilde{u}_0(t')]dt' = \tilde{X}_{bx},$$

$$\int_{t_{dis}-\tau_f}^{t_{dis}} \tilde{u}_{dis}(t')dt' = \tau_f\tilde{d} + \tilde{X}_{bx}.$$ 

During the disruption period, any node upstream of $N_{dis}$ will not work because any product upstream of $N_{dis}$ cannot pass through it to satisfy the operation downstream of it. Therefore, $\tilde{u}_j(t) = 0$ when $N_j \succ N_{dis}$ and $t \in [t_{dis}, t_{dis} + \tau_{dis}]$.

Due to $x(t_{dis} - \tau_f) = 0$ and $x(t_{dis} + \tau_{dis}) = 0$, we have:

$$\int_{t_{dis}-\tau_f}^{t_{dis}+\tau_{dis}} \tilde{u}_j(t')dt' = \int_{t_{dis}-\tau_f}^{t_{dis}+\tau_{dis}} \tilde{u}_{j-1}(t')dt', \forall j \in \{1, 2, ..., l\}.$$
Thus,
\[
\int_{t_{dis}+\tau_{dis}}^{t_{dis}+\tau_{dis}+\tau_f} \tilde{u}_j(t')dt' = \tau_f \tilde{d} + \tilde{X}_{bx}, \quad \forall j \in \{0, 1, ..., l\}.
\]

This is the constraint that \(\tilde{u}(t)\) must satisfy to be optimal. Once the primary objective function is minimized, then the total operation cost is a constant. Thus, we only need to consider the total storage cost in the secondary objective function.

Next, we consider the nodes \(\{N_i\}\) such that \(N^d_{s_j} \succ N_i \succ N^d_{c_{j-1}}\). According to the definition of \(s\)-nodes and \(c\)-nodes, we know that \(\bar{c}(N_i) \geq \bar{c}(N^d_{c_{j-1}})\) and \(\bar{q}_x(N_i) \geq \bar{q}_x(N^d_{s_j})\). Compared with products being built up among \(\{N_i\}\) at time \(t\), we can reduce the operations to let \(\tilde{u}_i(t) = \tilde{u}^d_{c_{j-1}}(t)\), which reduces the buildup among \(\{N_i\}\) and at the same time increases the buildup of the product of \(N^d_{s_j}\). Since the product of \(N^d_{s_j}\) has lower storage cost, reducing the operations of \(\{N_i\}\) will lead to lower total storage cost. Therefore, under optimal operations, the inventory among \(\{N_i\}\) should be 0. Thus, \(\tilde{u}_i(t) = \tilde{u}^d_{c_{j-1}}(t)\).

Then, we consider the nodes \(\{N_i\}\) such that \(N^d_{c_j} \succ N_i \succ N^d_{s_j}\). From the definition of \(s\)-nodes and \(c\)-nodes, we have \(\bar{c}(N_i) \geq \bar{c}(N^d_{c_j})\) and \(\bar{q}_x(N_i) \geq \bar{q}_x(N^d_{s_j})\). Compared with the situation that at some time the buildup among \(\{N_i\}\) is not 0, then \(N_i\) will be able to match up with the operation of \(N^d_{c_j}\) to push the buildup to the product of \(N^d_{s_j}\) which has lower storage cost than that of the products among \(\{N_i\}\). Thus, the optimal operations of \(\{N_i\}\) will make the inventory among \(\{N_i\}\) be 0. Therefore, \(\tilde{u}_i(t) = \tilde{u}^d_{c_j}(t)\).

We denote \(L\) as the smallest index such that \(\bar{c}(N^d_{c_L}) \geq \bar{c}(N^u_{c_1})\). Consider the nodes \(\{N_i\}\) such that \(N^u_{c_1} \succ N_i \succ N^d_{c_L}\), and \(\{N_k\}\) such that \(N^d_{dis} \succ N_k \succ N^d_{s_L}\). We have \(\bar{c}(N_i) \geq \bar{c}(N^u_{c_1})\) and \(\bar{q}_x(N_k) \geq \bar{q}_x(N^d_{s_L})\). With Assumption 11, the buildup in this subsequence can only locate among \(\{N_k\}\). Compared with the situation that some product is built up among \(\{N_k\}\) at time \(t\), the operation of \(\{N_i\}\) can match up with \(\tilde{u}^u_{c_1}(t)\) to push the buildup to the product of \(N^d_{s_L}\), which has lower storage cost than the products among \(\{N_k\}\). Therefore, the optimal operations of \(\{N_i\}\) will make the inventory among \(\{N_i\}\) be 0. Therefore, \(\tilde{u}_i(t) = \tilde{u}^u_{c_1}(t)\).

Then we consider the nodes \(\{N_i\}\) such that \(N_i \succ N_i \succ N^u_{c_1}\). We have \(\bar{c}(N_i) \geq \bar{c}(N^u_{c_1})\). Therefore, for any operation \(\tilde{u}^u_{c_1}(t)\), \(\{N_i\}\) can match up with the operation of \(N^u_{c_1}\). With Assumption 11, products cannot be built up among \(\{N_i\}\). We should let the operations of \(\{N_i\}\) make \(\tilde{u}_i(t) = \tilde{u}^u_{c_1}(t)\), so that inventory among \(\{N_i\}\) is 0. Therefore, under optimal operations, no product is built up among \(\{N_i\}\). Thus, \(\tilde{u}_i(t) = \tilde{u}^u_{c_1}(t)\).

To sum up, it is shown that the normalized operation of a node in a section is the same as the normalized operation of \(N^u_{c_1}\) in this section. If the operation of any node is obtained, the operations of all other nodes within the same section are determined. Therefore, these nodes satisfy the definition of networks of nodes with fixed operation rate. We can aggregate the nodes of a section as one node with the aggregation approach introduced in Section 5.2.2.

Besides, consider the Capacity Constraint of the aggregated node: \(r^n a \leq c\). Also, Equation (5.9) shows the capacity limit of \(u^a\). For a normalized network of \(\{N_1, N_2, ..., N_n\}\) in a serial or assembly tree network system, the Capacity Constraint of the aggregated node \(N^a\) of \(\{N_1, N_2, ..., N_n\}\) becomes:

\[
\begin{bmatrix}
1 \\
1 \\
... \\
1
\end{bmatrix}
\begin{bmatrix}
\tilde{c}(N_1) \\
\tilde{c}(N_2) \\
\vdots \\
\tilde{c}(N_n)
\end{bmatrix}
\leq
\begin{bmatrix}
\bar{c}(N_1) \\
\bar{c}(N_2) \\
\vdots \\
\bar{c}(N_n)
\end{bmatrix}.
\]
which is equivalent to
\[ u^a \leq \min \{ \tilde{c}(N_1), \tilde{c}(N_2), ..., \tilde{c}(N_n) \} \].

Therefore, we can define the aggregated capacity by:
\[ c^a := \min \{ \tilde{c}(N_1), \tilde{c}(N_2), ..., \tilde{c}(N_n) \} \],

which makes the Capacity Constraint become \( u^a \leq c^a \).

Therefore, the capacity of the aggregated node is determined by the \( c \)-nodes in this section.

The input and output of the aggregated node are the products going into and out from the section, respectively. The aggregation is presented as follows:

1. \( N^a_L \) is the aggregated node of \( \{ N_1, N_{1-1}, ..., N^d_{s_L} \} \). \( \tilde{c}(N^a_L) = \tilde{c}(N^u_{c_1}). \tilde{q}_x(N^a_L) = \tilde{q}_x(N^d_{s_L}) \).
2. \( N^a_j \) is the aggregated node of all the nodes \( N_i \) such that \( N^d_{s_{j+1}} \succ N_i \succ N^d_{s_{j}} \), where \( j \in \{ L - 1, L - 2, ..., 1 \} \). \( \tilde{c}(N^a_j) = \tilde{c}(N^d_{s_{j}}). \tilde{q}_x(N^a_j) = \tilde{q}_x(N^d_{s_{j}}). \text{product}^a_j = \text{product}^d_{s_{j}} \).
3. \( N^a_0 \) is the aggregated node of all the nodes \( N_i \) such that \( N^d_{s_1} \succ N_i \succ N_0 \).

Consider the operation rates.

\[
\tilde{u}_i(t) = \begin{cases} 
   u^a_L(t), & \text{for the } i \text{ such that } N_{c_1} \succ N_i \succ N^d_{s_L}; \\
   u^a_j(t), & \text{for the } i \text{ such that } N^d_{s_{j+1}} \succ N_i \succ N^d_{s_{j}}, \text{ and } j \in \{ L - 1, L - 2, ..., 1 \}; \\
   u^a_0(t), & \text{for the } i \text{ such that } N^d_{s_1} \succ N_i \succ N_0.
\end{cases}
\]

(7.10)

The aggregated system is shown by Figure 7.7, which satisfies:

1. For any \( i \in \{ L, L - 1, ..., 2 \} \), \( \tilde{q}_x(N^a_i) > \tilde{q}_x(N^a_{i-1}) \), and \( \tilde{c}(N^a_i) > \tilde{c}(N^a_{i-1}) \), and
2. Under the unit operation, all the nodes except \( N^a_0 \) have unit production rate of the output products, and all the nodes except \( N^a_L \) have unit consuming rate of the input products.
3. Every node (excepting the final node) requires a single unit resource to produce each output unit product.
4. The demand is 1.
5. The disruption happens at \( N^a_L \).

In this way, the serial network systems can be transformed into the DSCDC serial network systems.

![Figure 7.7: The aggregated system of a serial network system](image-url)
7.2.4 Main Steps to Obtain the Optimal Control for the Serial Network Systems

In this section, we present the procedure to obtain the optimal control for serial network systems based on the results discussed previously. First, we apply normalization and aggregation to transform the system into a DSCDC serial network system. Second, we solve the quadratic problem of the DSCDC serial network system to get its optimal operation. Finally, we calculate the optimal operation of the original system based on the quadratic problem results. A detailed procedure is introduced below:

1. Step 1: Normalization: determine $\tilde{q}_x$ and $\tilde{c}$ for the original serial network system.
2. Step 2: Locate the $s$-nodes and $c$-nodes for the original system.
3. Step 3: Find the smallest index $L$ such that $\tilde{c}(N^a_{c_L}) \geq \tilde{c}(N^a_{c_1})$ for the original system.
4. Step 4: Aggregation: determine the aggregated system consisting of $\{N^a_j\}$ with parameters $\tilde{q}_x(N^a_j)$, $\tilde{c}(N^a_j)$, and product $a_j$.
5. Step 5: For the aggregated system, solve $\tilde{x}^a(t_{dis})$ to minimize $C_{x,dr-dis} + C_{x,pre-dis}$, where $\tilde{x}^a(t_{dis})$ satisfies:
   
   \[ \tilde{x}^a(t_{dis}) \geq 0, \quad \sum_{j=1}^{L} \tilde{x}^a_j(t_{dis}) = \tilde{X}_{bx}, \quad \text{and} \quad \sum_{j=1}^{i} \tilde{x}^a_j(t_{dis}) \leq \tau_f[\tilde{c}(N^a_i) - 1], \quad \forall i \in \{1, 2, ..., L\}. \]

6. Step 6: Use $\tilde{x}^a(t_{dis})$ to solve $\tilde{u}^a(t)$ based on Lemma 5 and Corollary 1
7. Step 7: Map $u^a(t)$ of the aggregated system into $u(t)$ for the original system. First, use Equation (7.10) to calculate the normalized operation rates $\tilde{u}(t)$ in the original system. Then, use Equation (5.1) to calculate the $u(t)$ for the original system.

7.3 Case Study

In this section, we use the model and approach to analyze a production line in the customization center.

Consider a practical example, which will also be analyzed in detail in Section 7.3 of case study. We will analyze a production line in the customization center of a company which produces a type of office equipment. The customization center takes the operations to install the options to the products based on customers’ requests. There are mainly five operations: unpacking, option installation, functional testing, internal packaging, and repackaging. The entire system is shown by Figure 7.8.

Generic products are fed into the system by $N_6$. At $N_5$, each generic product is removed from the box and foam packaging. Then, it is delivered to $N_4$, where the options are installed. Extra parts, hardware and addons can be installed to the products based on received customizations. After that, each product is powered on and option functionality is checked at $N_3$. Some internal settings are also done at this node. There are multiple stations which can process more than one product at a time. However, for modeling purposes we still
consider it as a single station. The operations at $N_2$ are cleaning and internal packaging. $N_1$ carries out the operation of external repackaging. Products are repackaged with foams and boxes, and new part numbers are labeled on the boxes. Then, products will be delivered out from the system.

We consider a maintenance event as a predictable disruption in this case study. Suppose that node $N_4$ needs some maintenance which requires it to be shut down for 2 hours, i.e., $\tau_{dis} = 2$ hours. And the system is notified about this event 8 hours before it begins, i.e., $\tau_f = 8$ hours.

Denote product $i$ as the product going out from node $N_i$. After each node from $N_5$ to $N_1$ receives one product from its upstream node and finishes its operation, it outputs the processed product to the downstream node. The ratio of input to output is 1 : 1. Therefore, in its network matrix $B$, $B(i, i) = -1$, and $B(i, i + 1) = 1$, for $i \in \{1, 2, ..., 6\}$. All other elements in $B$ matrix are zeros.

The demand is on the final node $N_0$. Therefore, $D = [1, 0, 0, 0, 0, 0]$. The capacity of each node is the maximum numbers of products which can be produced per hour by each node from $N_5$ to $N_1$ independently. The nominal capacity vector $\bar{c}$ is denoted as $[c(N_0), c(N_1), c(N_2), c(N_3), c(N_4), c(N_5), c(N_6)]^T$. For each node, one operation produces one product. Then, we can simply define $R$ as a $7 \times 7$ Identity Matrix, so that the value of the operation rate of $N_i$ can not exceed the value of the capacity of $N_i$, based on the Capacity Constraint of $Ru(t) \leq c$.

Consider Assumption 11. This assumption says that we will only consider solutions with no products buildup at $N_6$ and $N_5$.

The demand is on the final node $N_0$. Therefore, $D = [1, 0, 0, 0, 0, 0]$. The value of the demand is $d = 20$ products/hour. Then, the production rate in nominal state is 20 products/hour, i.e., $\alpha_i = 20$ for $i \in \{0, 1, 2, ..., 6\}$.

The operations at $N_4$ may require additional resources such as extra parts and hardware. In this analysis, we consider the amount of these resources are always sufficient and they are always available for the operations at $N_4$. Similarly, $N_2$ and $N_1$ may require some resources such as boxes and foams. We also consider their amount can always satisfy the operations at these two nodes, respectively.

The maximum numbers of products which can be produced per hour by nodes from $N_5$ to $N_1$ are 30, 28, 36, 40 and 24, respectively. Therefore, the capacities (products/hour) at nodes from $N_5$ to $N_1$ in the Nominal State are:

$$c(N_5) = 30, \quad c(N_4) = 36, \quad c(N_3) = 36, \quad c(N_2) = 40, \quad \text{and} \quad c(N_1) = 24.$$ 

And the capacity of $N_4$ in the Disruption State is $c(N_4) = 0$.

The capacity of $N_0$ is considered as the demand based on the Demand constraint $Du(t) \leq d$. Therefore, $c(N_0) = d = 20$. $N_6$ can be considered as a warehouse which stores the original
products needing customizations. These products are pulled out from it at a rate of 20 products/hour in nominal state. In this analysis, the capacity of $N_6$ can be considered as infinite, and its unit storage cost can be considered as 0, i.e.

$$c(N_6) = +\infty, \text{ and } q_x(N_6) = 0.$$  

The storage costs at nodes from $N_5$ to $N_1$ represent the cost of space, resource, and operations for keeping products at these locations. We ignore the fixed costs since they can not be optimized. $N_5$ and $N_1$ have the same unit storage cost, which is denoted as $y$ dollars/(product·hour). Since the inner parts are exposed at $N_4$ and $N_3$, special operations are required to meet the cleanliness regulation, which makes the unit storage cost at these two nodes 20 times as much as the storage cost at $N_5$ or $N_1$. Besides, due to the lack of space of storage at $N_2$, the unit storage cost at $N_2$ is 3 times as much as the unit storage cost at $N_5$ or $N_1$. The unit storage costs (dollars/(product·hour)) are:

$q_x(N_5) = y, q_x(N_4) = 20y, q_x(N_3) = 20y, q_x(N_2) = 3y, \text{ and } q_x(N_1) = y.$

**7.3.1 Normalization**

The normalized demand is $\hat{d} = 1$. We consider 20 products as 1 load. Then, $\hat{d} = 1$ means the demand is 1 load/hour. Similarly, the normalized operation rate change from the value of how many products/hour to the value of how many loads/hour. The normalized capacity represents the maximum number of loads that can be processed in each hour. Therefore, the normalized capacities in the Nominal State are

$$\hat{c} = [\hat{c}(N_0), \hat{c}(N_1), \hat{c}(N_2), \hat{c}(N_3), \hat{c}(N_4), \hat{c}(N_5), \hat{c}(N_6)]^T,$$

i.e.:

$$\hat{c} = [\hat{c}(N_0), \hat{c}(N_1), \hat{c}(N_2), \hat{c}(N_3), \hat{c}(N_4), \hat{c}(N_5), \hat{c}(N_6)]^T = [1, 1.2, 2, 1.8, 1.8, 1.5, +\infty]^T.$$  

For the normalized unit storage cost, it represents the value of how many dollars/(load-hour). Hence, the normalized unit storage costs are

$$\tilde{q}_x = [\tilde{q}_x(N_1), \tilde{q}_x(N_2), \tilde{q}_x(N_3), \tilde{q}_x(N_4), \tilde{q}_x(N_5), \tilde{q}_x(N_6)] = [20y, 60y, 400y, 400y, 20y, 0].$$

The normalized capacities and storage costs are shown in Figure 7.9.

**7.3.2 Locate s- and c-nodes**

Based on the definitions of s- and c-nodes, the locations of them are listed in Table 7.1.

<table>
<thead>
<tr>
<th>s- and c-node</th>
<th>$N^u_1$</th>
<th>$N^d_1$</th>
<th>$N^u_3$</th>
<th>$N^d_3$</th>
<th>$N^u_4$</th>
<th>$N^d_4$</th>
<th>$N^u_2$</th>
<th>$N^d_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td>$N_5$</td>
<td>$N_0$</td>
<td>$N_1$</td>
<td>$N_4$</td>
<td>$N_1$</td>
<td>$N_2$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7.1: The locations of s- and c-nodes
7.3.3 Aggregation

Since $\tilde{c}(N_{d}^{2}) > \tilde{c}(N_{c_{1}}^{u}) > \tilde{c}(N_{c_{1}}^{d})$, there will be three aggregated nodes after aggregation. $N_{2}^{a}$ is the aggregation of $\{N_{6}, N_{5}, N_{4}, N_{3}, N_{2}\}$. $N_{1}^{a}$ is $N_{1}$. $N_{0}^{a}$ is $N_{0}$. The capacity of the aggregated system is:

$$\tilde{c}(N_{a}^{0}), \tilde{c}(N_{a}^{1}), \tilde{c}(N_{a}^{2}) = \tilde{c}(N_{0}), \tilde{c}(N_{1}), \tilde{c}(N_{5}) = [1, 1.2, 1.5].$$

The storage cost vector of the aggregated system is:

$$\tilde{q}_{x}(N_{1}^{a}), \tilde{q}_{x}(N_{2}^{a}) = \tilde{q}_{x}(N_{1}), \tilde{q}_{x}(N_{2}) = [20y, 60y].$$

7.3.4 Quadratic Program

Determine the optimal storage at $t_{dis}$ by quadratic program. The objective function is the total storage cost, in which the matrix $Q_{x}$ is:

$$Q_{x} = \begin{bmatrix} 120 & 20 \\ 20 & 253.33 \end{bmatrix}.$$

Denote $x$ as the variable vector $[\tilde{x}_{1}^{a}(t_{dis}), \tilde{x}_{2}^{a}(t_{dis})]^{T}$. For the constraints, $0 \leq x_{1} \leq \tau_{f}[^{\tilde{c}}(N_{c_{1}}^{d}) - \tilde{d}] = 1.6$. $x_{2} \geq 0$. $x_{1} + x_{2} = \tilde{X}_{bs} = \min\{\tau_{bsd}, \tau_{f}[\tilde{c}(N_{c_{1}}^{u}) - \tilde{d}]\} = 2$. The solution of this quadratic program by MATLAB is:

$$x = [1.4, 0.6]^{T}.$$

7.3.5 Optimal Operations

With the optimal $x$, we calculate the exact buildup at $t_{dis}$, which is:

$$[x_{1}(t_{dis}), x_{2}(t_{dis}), x_{3}(t_{dis}), x_{4}(t_{dis}), x_{5}(t_{dis}), x_{6}(t_{dis})] = [28, 12, 0, 0, 0, 0].$$

The total storage costs during pre-disruption period, disruption period, and the whole process are:

$$C_{x,pre-dis} = 134, C_{x,dr-dis} = 46, \text{ and } C_{x} = 180.$$
Figure 7.10: Normalized operation rates over different time periods

The buildup durations are:

$$[\tau_{bx,1}, \tau_{bx,2}] = [7, 2].$$

The consuming durations of the buildup are:

$$[\tau_1^a, \tau_2^a] = [2, 0.5].$$

The optimal normalized operation rates are shown by Figure 7.10. The first bar chart shows $\tilde{u}_i$ when $0 \leq t < t_{dis} - 7$. During this period, no inventory is built up. All the nodes keep the nominal operations to feed the demand. During the period of $t \in [t_{dis} - \tau_{bx,1}, t_{dis} - \tau_{bx,2})$, the system builds up product_1. There is a step between $\tilde{u}_1(t)$ and $\tilde{u}_0(t)$ in this period. Then when $t \in [t_{dis} - \tau_{bx,2}, t_{dis})$, the system builds up product_2, and there is a step between $\tilde{u}_2(t)$ and $\tilde{u}_1(t)$.

During the disruption period, the buildup is consumed to feed the demand. All the nodes upstream of $N_1$ have zero operation. $N_1$ operates until $t_{dis} + \tau_2^a$ when product_2 is empty. $N_0$ works till the end of the disruption $t_{dis} + \tau_{dis}$.

7.4 Conclusions

A special system, DSCDC network system, which has decreasing storage cost and capacity downstream, is studied in this chapter. During the disruption period, under the optimal operation, each node will push downstream the inventory in front of it at its full capacity. During the pre-disruption period, the system will build up inventory for the disruption.
Each node will work at its full capacity to buildup its product. With these properties of operations, we formulate the total cost as a function of the total inventory amount at the disruption starting time $t_{dis}$. The formulated optimization problem is changed into a quadratic program.

After the optimization problem of the DSCDC network system is solved, more complex serial network systems can be transformed into DSCDC network systems. Normalization is introduced to change the operation, inventory, capacity and other variables to be associated with the demand as a unit. Then the system can be divided into a number of sections. Within each section, the nodes have fixed ratio of operation, and nodes in each section can be aggregated as one node. Thus, a general serial network system can be transformed into a DSCDC network system. By applying the results of DSCDC network systems, the optimal operation of serial network systems can be obtained. Such an operation is only optimal under the assumption that the storage only locates downstream of $N_{dis}$. The total cost may be lower if we allow upstream storage.

The results of these studies can provide useful guidance for the resilient operation of manufacturing systems.
Chapter 8

Optimal Control of Serial Network Systems in Continuous Time Domain

In our previous Chapter 7, we showed the optimal operations of serial network systems under the assumption that no inventory storage is allowed upstream of the disrupted node \( N_{\text{dis}} \). In this chapter, we remove this assumption and develop the approach to determine the optimal operations of serial network systems in which the storage is allowed be built up anywhere within the systems.

8.1 Aggregation

8.1.1 Normalization and Locating \( s \)-nodes and \( c \)-nodes

Determination of the operation rates of nodes in the system is related to the storage costs of their products and their capacities. For convenience, as in the preceding chapter, the original system is normalized first in our analysis. We normalize the demand \( d \) to \( \tilde{d} := 1 \). The normalized operation rate \( \tilde{u}_i(t) \) is the quotient of the original operation rate \( u_i(t) \) divided by the nominal operation rate \( \alpha_i \). The normalized capacity \( \tilde{c}(N_i) \) is the maximum value of \( \tilde{u}_i(t) \) allowed by the capacity at \( N_i \). The normalized product amount \( \tilde{x}_i(t) \) is the quotient of the original product amount \( x_i(t) \) divided by the amount of product \( i \) needed by the operation to satisfy the demand \( d \). The normalized unit storage cost \( \tilde{q}_x(N_i) \) is multiplier to keep the storage cost unchanged before and after normalization, i.e., \( \tilde{q}_x(N_i) \tilde{x}_i(t) = q_x(N_i)x_i(t) \). In the previous paper, normalization is introduced in detail.

\( s \)-nodes and \( c \)-nodes are special nodes in a serial network system. A \( s \)-node has the lowest normalized unit storage among a section of nodes, and a \( c \)-node has the lowest normalized capacity among a section of nodes. In the previous paper, locating \( s \)-nodes and \( c \)-nodes among the nodes downstream of \( N_{\text{dis}} \) is discussed in detail.

For the nodes upstream of \( N_{\text{dis}} \), denote \( s \)-nodes and \( c \)-nodes among them as \( N^s_{\ell_1} \) and \( N^c_{\ell_1} \), respectively. \( N^s_{\ell_1} \) is the node with lowest normalized capacity among \( \{N_i, N_{i-1}, ..., N_{\text{dis}}\} \), satisfying:
1. for any $N_j$ such that $N_i \succ N_j \succ N_{c_i}^u$, $\hat{c}(N_j) \geq \hat{c}(N_{c_i}^u)$, and
2. for any $N_j$ such that $N_{c_i}^u \succ N_j \succ N_{dis}$, $\hat{c}(N_j) > \hat{c}(N_{c_i}^u)$.

Define $N_{c_i+1}^u := N_{dis}$. For $1 \leq i \leq r$, $N_{s_i}^u$ and $N_{c_i}^u$ satisfy:

1. $N_{c_i}^u \succ N_{s_i}^u \succ N_{c_i+1}^u$, and
2. $N_{s_i}^u$ is the node with lowest normalized unit storage cost among $\{N_j | N_{c_i}^u \succ N_j \succ N_{dis}\}$, i.e.,
   (a) for any $N_j$ such that $N_{c_i}^u \succ N_j \succ N_{s_i}^u$, $\bar{q}_x(N_j) \geq \bar{q}_x(N_{s_i}^u)$, and
   (b) for any $N_j$ such that $N_{s_i}^u \succ N_j \succ N_{dis}$, $\bar{q}_x(N_j) > \bar{q}_x(N_{s_i}^u)$.
3. $N_{c_i+1}^u$ is the node with lowest normalized capacity among $\{N_j | N_{s_i}^u \succ N_j \succ N_{dis}\}$, i.e.,
   (a) for any $N_j$ such that $N_{s_i}^u \succ N_j \succ N_{c_i+1}^u$, $\hat{c}(N_j) \geq \hat{c}(N_{c_i+1}^u)$, and
   (b) for any $N_j$ such that $N_{c_i+1}^u \succ N_j \succ N_{dis}$, $\hat{c}(N_j) > \hat{c}(N_{c_i+1}^u)$.

Also, the product of $N_{s_j}^d$ is denoted as product$_{s_j}^d$, and the product of $N_{s_i}^u$ is denoted as product$_{s_i}^u$.

The total operation cost is a constant if the value of primary objective function is minimized. Consider the buildup at $t_{dis}$, since $N_{dis}$ can not work during $[t_{dis}, t_{dis} + \tau_{dis}]$, no product can pass through $N_{dis}$ during this period. Thus, all the buildup should be downstream of $N_{dis}$ by $t_{dis}$. Denote the optimal total amount of buildup at $t_{dis}$ as $\tilde{X}_{bx}$. Since the lowest capacity upstream of $N_{dis}$ is at $N_{c_i}^u$, the upper bound of the amount of buildup made during the pre-disturbance period is $\tau_f[\hat{c}(N_{c_i}^u) - \tilde{d}]$. The buildup needed to satisfy the demand during the disruption period is $\tau_{dis}\tilde{d}$. Therefore,

$$\tilde{X}_{bx} := \min\{\tau_f[\hat{c}(N_{c_i}^u) - \tilde{d}], \tau_{dis}\tilde{d}\}. \tag{8.1}$$

Also,

$$\int_{t_{dis} - \tau_f}^{t_{dis}} [\tilde{u}_{dis}(t') - \tilde{u}_0(t')]dt' = \tilde{X}_{bx}. \tag{8.2}$$

$$\int_{t_{dis} - \tau_f}^{t_{dis}} \tilde{u}_{dis}(t')dt' = \tau_f \tilde{d} + \tilde{X}_{bx}. \tag{8.3}$$

Since $\tilde{u}_{dis}(t) = 0$ for $t \in [t_{dis}, t_{dis} + \tau_{dis})$, then

$$\int_{t_{dis} - \tau_f}^{t_{dis} + \tau_{dis}} \tilde{u}_{dis}(t')dt' = \tau_f \tilde{d} + \tilde{X}_{bx}. \tag{8.4}$$

Since $\tilde{x}(t_{dis} - \tau_f) = 0$ and $\tilde{x}(t_{dis} + \tau_{dis}) = 0$, then

$$\int_{t_{dis} - \tau_f}^{t_{dis} + \tau_{dis}} \tilde{u}_j(t')dt' = \int_{t_{dis} - \tau_f}^{t_{dis} + \tau_{dis}} \tilde{u}_{j-1}(t')dt', \forall j \in \{1, 2, ..., l\}. \tag{8.5}$$

$$\int_{t_{dis} - \tau_f}^{t_{dis} + \tau_{dis}} \tilde{u}_j(t')dt' = \tau_f \tilde{d} + \tilde{X}_{bx}, \forall j \in \{1, 2, ..., l\}. \tag{8.6}$$
Therefore, once the primary objective function is minimized, then the optimal total operation cost is a constant, which means we can just consider the total storage cost in the secondary objective function when determining the optimal operation.

If there are some products built up upstream of \( N_{\text{dis}} \), they should be stored at \( s \)-nodes to achieve minimum total storage cost. First we consider the nodes \( N_j \) such that \( N^u_{c_1} \succ N_j \succ N^u_{s_i} \). Since \( N^u_{c_1} \) has the lowest capacity among \( \{N_j\} \), then all the nodes among \( \{N_j\} \) have sufficient capacity to match up with the operation of \( N^u_{c_1} \) to transfer all the amount of product of \( N^u_{c_1} \) to \( N^u_{s_i} \). Since the product of \( N^u_{s_i} \) has the lowest unit storage cost among \( \{N_j\} \), the optimal operation should transfer all the amount of product of \( N^u_{c_1} \) to \( N^u_{s_i} \). Therefore, the buildup among \( \{N_j\} \) should only be at \( N^u_{s_i} \).

Then we consider the nodes \( N_j \) such that \( N^u_{s_i} \succ N_j \succ N^u_{c_1+1} \). Since \( N^u_{c_1+1} \) has the lower capacity than any node among \( \{N_j\} \), then all the nodes among \( \{N_j\} \) have sufficient capacity to match up with the operation of \( N^u_{c_1+1} \) to avoid redistributing the buildup at \( N^u_{s_i} \) to any node among \( \{N_j\} \). Since the product of \( N^u_{s_i} \) has lower unit storage cost than any node among \( \{N_j\} \), the optimal operation should keep the buildup at \( N^u_{s_i} \) from redistributed to any node among \( \{N_j\} \). Therefore, there is no buildup among \( \{N_j\} \).

Next, we consider the nodes \( N_j \) such that \( N_i \succ N_j \succ N^u_{c_1} \). Since \( N_{c_1} \) has lower capacity than any node among \( \{N_j\} \), then all the nodes among \( \{N_j\} \) have sufficient capacity to match up with the operation of \( N^u_{c_1} \) to avoid creating buildup among \( \{N_j\} \). To keep the operations of nodes among \( \{N_j\} \) matching up with that of \( N^u_{c_1} \) can ignore the unnecessary buildup among \( \{N_j\} \), so as to achieve the minimum total storage cost. Therefore, there is no buildup among \( \{N_j\} \).

### 8.1.2 Two Special Nodes for Aggregation: \( N^d_{c_{\rho}} \) and \( N^u_{c_{r+1}} \)

\( \rho \) and \( \gamma \) are defined by the following statements.

\( \rho \) is the smallest index in \( \{1, 2, \ldots, p\} \) such that there exists some index \( i \) such that \( \tilde{c}(N^d_{c_{\rho}}) \geq \tilde{c}(N^u_{c_i}) \) and \( \tilde{d}_x(N^d_{s_{\rho}}) \leq \tilde{d}_x(N^u_{s_{i}}) \). If such an index does not exist, then define \( \rho = p \).

\( \gamma \) is the smallest index in \( \{0, 1, 2, \ldots, r - 1\} \) such that \( \tilde{c}(N^d_{c_{\rho}}) \geq \tilde{c}(N^u_{c_{r+1}}) \) and \( \tilde{d}_x(N^d_{s_{\rho}}) \leq \tilde{d}_x(N^u_{s_{r+1}}) \). If such an index does not exist, then define \( \gamma = r \).

![Disruption](image)

**Figure 8.1:** \( N^d_{c_{\rho}} \) and \( N^u_{c_{r+1}} \)

### Lemma 12.

For a serial network system, during the pre-disruption period, the optimal operation should result in \( \tilde{x}^{u}_{s_{i}}(t) = \tilde{x}^{d}_{s_{j}}(t) = 0 \) for any \( i \in \{\gamma + 1, \gamma + 2, \ldots, r\} \) and any \( j \in \{\rho + 1, \rho + 2, \ldots, p\} \).

This lemma says during the pre-disruption period there is no buildup among the nodes \( \{N_i | N^u_{c_{r+1}} \succ N_i \succ N^d_{s_{\rho}}\} \)
Then, the nodes of the aggregated system become:

\[ N_{c_{\gamma+1}}^u \]

Analysis of serial network systems under the assumption of no buildup upstream of nodes downstream of rate. Thus, we can aggregate these three types of nodes, respectively. Also, consider the operation rate; and, nodes among \( \{N_1, ..., N_p\} \) has the lowest capacity. Any amount of product coming out of \( N_{c_{\gamma+1}}^u \) can be completely transferred through all these nodes immediately.

And \( N_{s_{\rho}}^d \) has the lowest storage cost among these nodes. To achieve minimum storage cost, the optimal operation will transfer all the product coming out of \( N_{c_{\gamma+1}}^u \) to product \( N_{s_{\rho}}^d \). Therefore, at any time during the pre-disruption period, there will be no buildup among these nodes, i.e., \( \tilde{x}_{s_i}^u(t) = \tilde{x}_{s_j}^u(t) = 0 \) for any \( i \in \{\gamma + 1, \gamma + 2, ..., r\} \) and any \( j \in \{\rho + 1, \rho + 2, ..., p\} \).

With Lemma 12, all the nodes among \( \{N_j|N_{c_{\gamma+1}}^u \nless N_j \nless N_{s_{\rho}}^d\} \) have the same operation rate. This operation rate can not be higher than the smallest capacity among these nodes, which is \( \tilde{c}(N_{c_{\gamma+1}}^u) \).

### 8.1.3 Aggregation

Based on the analysis above, nodes among \( \{N_j|N_{s_{\gamma}}^u \nless N_j \nless N_{s_{\rho}}^d\} \) always have the same operation rate for \( 1 \leq i \leq \gamma - 1 \); nodes among \( \{N_i|N_{s_{\gamma+1}}^u \nless N_i \nless N_{s_{\gamma}}^d\} \) always have the same operation rate; and, nodes among \( \{N_j|N_{c_{\gamma+1}}^u \nless N_j \nless N_{s_{\rho}}^d\} \) always have the same operation rate. Thus, we can aggregate these three types of nodes, respectively. Also, consider the nodes downstream of \( N_{dis} \). We can also aggregate them with same approach as in the analysis of serial network systems under the assumption of no buildup upstream of \( N_{dis} \).

The aggregation is as follows:

- \( N_0^a \) is the aggregation of \( \{N_j|N_{s_{\gamma}}^u \nless N_j \nless N_{s_{\rho}}^d\} \). \( c(N_0^a) = \tilde{c}(N_0) = \tilde{a} \).
- For \( 1 \leq j \leq \rho - 1 \), \( N_j^a \) is the aggregation of \( \{N_i|N_{s_{j+1}}^u \nless N_i \nless N_{s_{j}}^d\} \). \( c(N_j^a) = \tilde{c}(N_{c_j}) \).
- \( N_0^a \) is the aggregation of \( \{N_i|N_{s_{\gamma}}^u \nless N_i \nless N_{s_{\rho}}^d\} \). \( c(N_0^a) = \tilde{c}(N_{c_{\gamma+1}}^u) \).
- For \( \rho \leq i \leq \gamma + \rho - 1 \), \( N_i^a \) is the aggregation of \( \{N_j|N_{s_{\gamma+\rho+1}}^u \nless N_j \nless N_{c_{\gamma+\rho+1}}^u\} \). \( c(N_i^a) = \tilde{c}(N_{s_{\gamma+\rho+1+1}}^u) \).
- For \( \gamma + \rho \leq i \leq \gamma + \rho + \rho - 1 \), \( N_i^a \) is the aggregation of \( \{N_j|N_{s_{\gamma}}^u \nless N_j \nless N_{s_{\rho}}^d\} \). \( c(N_i^a) = \tilde{c}(N_{c_{\gamma+\rho+1}}^u) \).
- \( N_{\gamma+\rho}^a \) is the aggregation of \( \{N_j|N_{s_{\gamma}}^u \nless N_j \nless N_{s_{\rho}}^d\} \). \( c(N_{\gamma+\rho}^a) = \tilde{c}(N_{c_{\gamma+\rho+1}}^u) \).

Then, the nodes of the aggregated system become:

\[
N_{\gamma+\rho}^a, N_{\gamma+\rho-1}^a, ..., N_{\rho+1}^a, N_{\rho}^a, ..., N_{1}^a, N_{0}^a.
\]

\( N_{\rho}^a \) is the disrupted node. Also, we can denote \( L = \gamma + \rho \). Then the indices become:

\[
N_{L}^a, N_{L-1}^a, ..., N_{\rho+1}^a, N_{\rho}^a, ..., N_{1}^a, N_{0}^a.
\]
8.2 Optimization Problem

We assume at time $t_{\text{dis}}$ the buildup is $x^a(t_{\text{dis}})$. Based on $x^a(t_{\text{dis}})$, we formulate the operation in every period. With the operation we formulate the total cost as the objective function. Then, we solve the optimization problem to determine the optimal $x^a(t_{\text{dis}})$. Finally, we calculate the exact operation with the optimal $x^a(t_{\text{dis}})$.

There will be no buildup upstream of $N_{\text{dis}}$ at $t_{\text{dis}}$, since any buildup upstream of $N_{\text{dis}}$ can not be transferred downstream of $N_{\text{dis}}$ after $t_{\text{dis}}$. Such buildup can not provide any products for the demand during the disruption period, but introduces extra storage cost. Therefore, the upstream buildup since $t_{\text{dis}}$ should be avoid. This does not mean that buildup can not exist before $t_{\text{dis}}$. Therefore,

$$x^a_j(t_{\text{dis}}) = 0, \text{ for } \rho < j \leq L.$$  

Denote the total amount of buildup at $t_{\text{dis}}$ as $\bar{X}_{bx}$. Therefore,

$$\sum_{j=1}^\rho x^a_j(t_{\text{dis}}) = \bar{X}_{bx} := \min\{\tau_{\text{dis}} + \rho, \tau_f(c(N^a_{\rho}) - 1)\}.$$  

(8.2)

In our analysis, we denote $\tau_j$ as the duration from $t_{\text{dis}}$ to the time point when product $j$ is consumed to 0 within the disruption period. Therefore, $x^a_j(t) = 0$ for any $t$ in $[t_{\text{dis}} + \tau_j, t_{\text{dis}} + \tau_{\text{dis}})$, and there exists a $t_1 \in [t_{\text{dis}}, t_{\text{dis}} + \tau_j)$ such that $x^a_j(t) > 0$ for any $t$ in $[t_1, t_{\text{dis}} + \tau_j)$. Denote $t_j := t_{\text{dis}} + \tau_j$.

Denote $t_{bx,j}$ as the time point when the system starts building up product $j$ within the pre-disruption period. Therefore, $x^a_j(t) = 0$ for any $t$ in $[t_{\text{dis}} - \tau_j, t_{bx,j})$, and there exists a $t_1 \in (t_{bx,j}, t_{\text{dis}}]$ such that $x^a_j(t) > 0$ for any $t$ in $[t_1, t_{bx,j})$. If product $j$ is never built up in the pre-disruption period, then define $t_{bx,j} := t_{\text{dis}}$. Denote $\tau_{bx,j} := t_{\text{dis}} - t_{bx,j}$.

In the later analysis, we will show that $\{t_{bx,j} | 1 \leq j \leq \rho\}$ and $\{t_{bx,i} | \rho + 1 \leq i \leq L\}$ are interleaving. We will use $S$ matrix to represent the interleaving order of these two sets.

**Definition 15.** The $S$ matrix is defined as a $\rho \times \gamma$ matrix such that its element $S_{j,k}$ satisfies:

$$S_{j,k} = \begin{cases} 
1, & \text{if } t_{bx,k+\rho} \leq t_{bx,j}, \\
0, & \text{otherwise}.
\end{cases}$$

It will be shown later in Lemma 22 that the $S$ matrix will have an echelon form. An example of $S$ matrix is:

$$S = \begin{bmatrix} 
0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 \\
1 & 1 & 1 & 0 & 0
\end{bmatrix}.$$  

Each possible interleaving order of $\{t_{bx,j} | 1 \leq j \leq \rho\}$ and $\{t_{bx,i} | \rho + 1 \leq i \leq L\}$ is represented by a unique matrix $S^{(k)}$. Since the values of $\{t_{bx,j} | 1 \leq j \leq \rho\}$ and $\{t_{bx,i} | \rho + 1 \leq i \leq L\}$ are functions of $x^a(t_{\text{dis}})$, an order of $\{t_{bx,j} | 1 \leq j \leq \rho\}$ and $\{t_{bx,i} | \rho + 1 \leq i \leq L\}$ is a constraint on $x^a(t_{\text{dis}})$ which defines a specific domain of $x^a(t_{\text{dis}})$. For different orders, the domains defined
by them are mutually exclusive. Then our optimization process can be briefly described by Figure 8.2.

Figure 8.2: The optimization process

First, each \( S^{(k)} \) is generated to represent each possible order of \( \{t_{bx,j} | 1 \leq j \leq \rho \} \) and \( \{t_{bx,i} | \rho + 1 \leq i \leq L \} \), represented by \( S^{(k)} \). Second, for a given \( S^{(k)} \), we use optimization program to determine the optimal \( x^{a,(k)}(t_{dis}) \) and the minimum cost \( C^{(k)} \) within the domain defined by \( S^{(k)} \). Next, we compare each \( C^{(k)} \) and find the minimum one and its corresponding \( x^{a,(k)}(t_{dis}) \), which is the optimal \( x^{a}(t_{dis}) \). Last, the optimal operation \( u \) is calculated based on the optimal \( x^{a}(t_{dis}) \).

We only need to consider the inventory storage cost in the objective function. The operation cost is a fixed value depending on \( \tau_f \) and \( \tau_{dis} \), not related to \( x^{a}(t_{dis}) \). It plays no role in optimizing action.

We use figures to assist the analysis of the buildup process. In such a figure, we draw the curves associated with the operation rates of all the nodes. The derivatives of these curves stand for buildup rates and/or consuming rates of different products. For a serial network system, define \( Y_i \) as the integral curve of \( \tilde{u}_i(t) - \tilde{d} \) from \( t_{dis} - \tau_f \) to \( t \). The difference of the values of \( Y_i \) and \( Y_{i-1} \) is the instant value of the \( \tilde{x}_i(t) \).

\[
Y_i(t) := \int_{t_{dis} - \tau_f}^{t} [\tilde{u}_i(t') - \tilde{d}] dt'.
\]

\[
\tilde{x}_i(t) = Y_i(t) - Y_{i-1}(t).
\]

For example, consider a serial network system with four nodes, which is shown by Figure 8.3.

Figure 8.3: An example of simple serial network system
In this example, we assume the operation rate of \( N_0 \) is \( \tilde{u}_0(t) = \tilde{d} \) when \( t < t_{dis} \). \( t_{bx,1} \), \( t_{bx,2} \) and \( t_{bx,3} \) are starting times of buildup of product \( 1 \), product \( 2 \) and product \( 3 \), respectively. Thus, for \( i \in \{1, 2, 3\} \), we assume

\[
\tilde{u}_i(t) = \begin{cases} 
   \tilde{u}_{i-1}(t), & \text{when } t < t_{bx,i}, \\
   \tilde{u}_i, & \text{when } t_{bx,i} \leq t < t_{dis}.
\end{cases}
\]

At time \( t_{dis} \), we assume the amount of three products are \( x_0(t_{dis}) = 0 \), \( x_2(t_{dis}) \) and \( x_1(t_{dis}) \), respectively. Then, \( x_2(t_{dis}) + x_1(t_{dis}) = \tilde{X}_{bx} \). The buildup process of this example can be represented by Figure 8.4.

![Figure 8.4: An example of buildup process](image)

From \( t_{bx,1} \) to \( t_{dis} \), product \( 1 \) is built up at a constant rate of \( \tilde{u}_1 - \tilde{d} \). In the figure, we use \( Z_1 \) to represent its buildup process. The slope of \( Z_1 \) is \( \tilde{u}_1 - \tilde{d} \) from \( t_{bx,1} \) to \( t_{dis} \). Each point on \( Z_1 \) shows the instant value of \( \tilde{x}_1(t) \) at time \( t \). The intersection of \( Z_1 \) on the vertical axis is \( (t_{dis}, \tilde{x}_1(t_{dis})) \).

From \( t_{bx,2} \) to \( t_{dis} \), product \( 2 \) is built up. \( N_2 \) produces product \( 2 \) at the rate of \( \tilde{u}_2 \), while \( N_1 \) consumes product \( 2 \) at the rate of \( \tilde{u}_1 \). Therefore, we use \( Z_2 \) to represent the buildup process of product \( 2 \). The slope of \( Z_2 \) is \( \tilde{u}_2 - \tilde{d} \) from \( t_{bx,2} \) to \( t_{dis} \). The difference of the values of \( Z_2 \) and \( Z_1 \) at \( t \) is the instant value of \( \tilde{x}_2(t) \), when \( t \in [t_{bx,2}, t_{dis}] \). The intersection of \( Z_2 \) on the vertical axis is \( (t_{dis}, \tilde{X}_{bx}) \).

From \( t_{bx,3} \) to \( t_{dis} \), product \( 3 \) is built up at the rate of \( \tilde{u}_3 \). Both \( N_2 \) and \( N_1 \) work at the rate of \( \tilde{u}_3 \) when \( t \in [t_{bx,3}, t_{bx,2}] \). Then, product \( 3 \) is built up at the rate of \( \tilde{u}_3 - \tilde{u}_1 \), which is represented by \( Z_3 \) whose slope is \( \tilde{u}_3 - \tilde{d} \) when \( t \in [t_{bx,3}, t_{dis}] \). The instant value of \( \tilde{x}_3(t) \) is the difference of the values of \( Z_3 \) and \( Z_1 \) when \( t \in [t_{bx,2}, t_{dis}] \). From \( t_{bx,2} \) to \( t_{dis} \), product \( 3 \) is decreasing at the rate of \( \tilde{u}_2 - \tilde{u}_1 \). The difference of the values of \( Z_3 \) and \( Z_2 \) at \( t \) is the instant value of \( \tilde{x}_3(t) \), when \( t \in [t_{bx,2}, t_{dis}] \). At time \( t_{dis} \), product \( 3 \) is decreased to 0, i.e., \( \tilde{x}_3(t_{dis}) = 0 \). Thus, the intersection of \( Z_3 \) on the vertical axis is the same as \( Z_2 \), which is \( (t_{dis}, \tilde{X}_{bx}) \).

Therefore,

\[
Y_i = \begin{cases} 
   Y_{i-1}, & \text{when } t < t_{bx,i}, \\
   Z_i, & \text{when } t_{bx,i} \leq t < t_{dis},
\end{cases}
\]

where \( i \in \{1, 2, 3\} \) and \( Y_0 \) in this example is the horizon axis.

To calculate the storage cost of product \( i \) in a time interval, we should integrate \( \tilde{x}_i(t) \) in this interval, which is the area of the gap space between \( Y_i \) and \( Y_{i-1} \) in this interval. For
example, the integral of $\tilde{x}_1(t)$ from $t_{bx,1}$ to $t_{dis}$ is the triangle area bounded by $Z_1$, the horizon axis and the vertical axis. And the integral of $\tilde{x}_3(t)$ from $t_{bx,3}$ to $t_{dis}$ is the triangle area bounded by $Z_3$, $Z_1$ and $Z_2$. For the total storage cost, we just need to calculate the area of each product, and multiply it with its unit storage cost $\tilde{q}_x(N_i)$, and then calculate the summation over all products.

With the buildup process figure, we can easily analyze the effects of operation rates on the storage costs.

8.2.1 Operations and Cost in Nominal-operation Period and Disruption Period

In the Nominal-operation period, the system just produces to meet the demand, and no inventory is built up. The operation will be $\tilde{u}_i = 1$ for any $i$. And storage cost is 0.

In the Disruption period, given $x^a(t_{dis})$, the system pushes inventory downstream at the highest capacity. Therefore, the results in Chapter 7 of DSCDC network systems in the Disruption period can still work.

**Lemma 13.** Consider a serial network system such that a disruption occurs at time $t_{dis}$ with inventory storage $x^a(t_{dis})$ satisfying $\sum_{j=1}^{\rho} x^a_j(t_{dis}) \leq \tilde{d}_{t_{dis}}$. The optimal operation rate $u^a_j$ where $j \in \{1, 2, ..., \rho - 1\}$ satisfies the following over time period $t_{dis} \leq t \leq t_{dis} + \tau_{dis}$:

$$u^a_j = \begin{cases} c(N^a_j), & \text{when } t_{dis} \leq t \leq t_{dis} + \tau_{j+1}, \\ 0, & \text{when } t_{dis} + \tau_{j+1} < t \leq t_{dis} + \tau_{dis}, \end{cases}$$

where $\tau_j$ satisfies:

$$\tau_j = \frac{\sum_{i=j}^{\rho} x^a_i(t_{dis})}{c(N^a_{j-1})}, \forall j \in \{1, 2, ..., \rho\}.$$

The operation rates of $N^a_{\rho}$ and all upstream nodes during this period is 0.

The total storage cost over the disruption period is:

$$C^d_{x,dr-dis} = \frac{1}{2} \sum_{j=1}^{\rho} \left\{ q_x(N^a_j) - q_x(N^a_{j-1}) \right\} \left\{ \frac{\left( \sum_{i=j}^{\rho} x^a_i(t_{dis}) \right)^2}{c(N^a_{j-1})} \right\}.$$

8.2.2 Operations and Cost in Pre-disruption Period

This period is the one when the system is building up the storage $x^a(t_{dis})$. All the nodes work at their highest capacity when building their corresponding inventory.

Operations in Pre-disruption Period

For the upstream nodes, we find their operations satisfy a sort of capacity pattern. **Lemma 14.** For $1 \leq i \leq L$, the optimal operation of $N^a_i$ satisfies

$$u^a_i(t) = \begin{cases} u^a_{i-1}(t), & \text{when } t \in [t_{dis} - \tau_{j}, t_{bx,i}), \\ c(N^a_i), & \text{when } t \in [t_{bx,i}, t_{dis}). \end{cases}$$

(8.3)
Proof. This lemma is straightforward geometrically. Figure 8.5 shows the buildup process of product\textsuperscript{a}\textsubscript{i}.

At time \( t_{\text{dis}} \) the downstream buildup reaches \( \bar{X}_{\text{bx}} \), and the upstream buildup should be consumed to be 0 to avoid unnecessary storage cost. Thus, the buildup line of any product\textsuperscript{a}\textsubscript{i} should reach the fixed point \(( t_{\text{dis}}, \bar{X}_{\text{bx}} )\). By increasing the slope of the buildup line of product\textsuperscript{a}\textsubscript{i}, we can reduce the area of product\textsuperscript{a}\textsubscript{i} and increase the area of the buildup upstream of \( N_{\text{a}} \). Since the buildup upstream of \( N_{\text{a}} \) has lower storage cost than product\textsuperscript{a}\textsubscript{i}, the total storage cost can be reduced. Hence, the optimal operation of \( u_{\text{a}}^i(t) \) for \( t \in [t_{bx,i}, t_{\text{dis}}) \) should be as great as possible, which is the capacity \( c(N_{\text{a}}) \).

Corollary 2. For a serial network system, the optimal operation should satisfy:

\[ t_{bx,i} < t_{bx,i-1}, \forall i \in \{\rho + 2, \rho + 3, ..., L\}. \]

Proof. For \( i \in \{\rho+1, \rho+2, ..., L\} \), from Lemma 14, \( t_{bx,i} \) is the horizon value of the intersection of \( Y_{\text{a}}^i \) and \( Y_{\text{a}}^\rho \). The slope of \( Y_{\text{a}}^i \) during \([t_{bx,i}, t_{\text{dis}})\) is \( c(N_{\text{a}}) - \bar{d} \). Therefore, for \( i \in \{\rho + 2, \rho + 3, ..., L\} \), the slope of \( Y_{\text{a}}^i \) during \([t_{bx,i}, t_{\text{dis}})\) is smaller than the slope of \( Y_{\text{a}}^{i-1} \) during \([t_{bx,i-1}, t_{\text{dis}})\). Then, the intersection of \( Y_{\text{a}}^i \) and \( Y_{\text{a}}^\rho \) should be at the left of the intersection of \( Y_{\text{a}}^{i-1} \) and \( Y_{\text{a}}^\rho \), as shown in Figure 8.6. Therefore, \( t_{bx,i} < t_{bx,i-1} \).
Lemma 15. For a serial network system, the optimal operation should satisfy:

\[ t_{bx,j} \leq t_{bx,j+1}, \ \forall j \in \{1, 2, ..., \rho - 1\}. \]

In words, the time of start of buildup of product \( a_j \) is no later than the time of start of buildup of product \( a_{j+1} \). These nodes are shown in Figure 8.7.

\[ \text{Figure 8.7: } N_j^a, N_{j+1}^a, ..., N_1^a \]

Proof. Since product \( a_j \) has lower unit storage cost than product \( a_{j+1} \), the system will try to transfer as much product \( a_{j+1} \) as possible to product \( a_j \) to achieve minimum total storage cost. Therefore, \( N_j^a \) will try to match up its operation rate with the operation rate of \( N_{j+1}^a \). As long as the operation rate of \( N_{j+1}^a \) is less than or equal to the capacity of \( N_j^a \), there will not be any buildup of product \( a_{j+1} \).

Once the the operation rate of \( N_{j+1}^a \) is greater than the capacity of \( N_j^a \), there will be definite buildup of product \( a_{j+1} \). At this time, \( N_j^a \) will still work at its capacity to transfer as much product \( a_{j+1} \) as possible to the node \( N_j^a \), to reduce the total storage cost. Since \( N_{j-1}^a \) has lower capacity than \( N_j^a \), the production rate of product \( a_j \) is higher than its consuming rate. Thus, product \( a_j \) is built up.

To sum up, whenever product \( a_{j+1} \) starts buildup, there will exist the buildup of product \( a_j \). Therefore, the time of start of buildup of product \( a_j \) is no later than the time of start of buildup of product \( a_{j+1} \).

Lemma 16. For a serial network system, for any \( i \in \{\rho + 1, \rho + 2, ..., L\} \) and \( j \in \{1, 2, ..., \rho\} \) such that \( q_x(N_i^a) \geq q_x(N_j^a) \) and product \( a_i \) and product \( a_j \) are built up sometime during the pre-disruption period, the optimal operation should satisfy:

\[ t_{bx,j} \leq t_{bx,i}. \]

In words, for any \( i \in \{\rho + 1, \rho + 2, ..., L\} \) and \( j \in \{1, 2, ..., \rho\} \), the time of start of buildup of product \( a_j \) is no later than the time of start of buildup of product \( a_i \) if \( q_x(N_i^a) \geq q_x(N_j^a) \).

Proof. Since the unit storage cost at \( N_{j+1}^a \) is greater than or equal to the unit storage cost at \( N_j^a \), then, based on Lemma 12, the capacity of \( N_j^a \) is greater than the capacity of \( N_{j+1}^a \). Therefore, the capacity of \( N_j^a \) is the smallest among the capacities of nodes of \( \{N_i^a, N_{i-1}^a, ..., N_\rho^a, N_{j+1}^a, N_j^a\} \), and the unit storage cost of \( N_j^a \) is the smallest among the unit storage costs of nodes of \( \{N_i^a, N_{i-1}^a, ..., N_\rho^a, N_{j+1}^a, N_j^a\} \). These nodes are shown in Figure 8.8.
At the very beginning when \( t = 0 \), the operation rates of nodes from \( N^a_i \) to \( N^a_j \) all equal to the demand. There is no buildup at \( N^a_i \) as long as the operation rates of nodes from \( N^a_i \) to \( N^a_j \) equal to each other.

If the operation rate of \( N^a_i \) is not greater than the capacity of \( N^a_j \), then all the nodes from \( N^a_i \) to \( N^a_j \) has sufficient capacity to match up the operation rate of \( N^a_i \). This operation will transfer all of \( \text{product}^a_i \) to the location of \( N^a_j \). Due to the lower unit storage cost at \( N^a_j \) than any nodes from \( N^a_i \) to \( N^a_{j-1} \), transferring all of \( \text{product}^a_i \) to the location of \( N^a_j \) can achieve the minimum total storage cost. Even if the unit storage cost at \( N^a_i \) equals to that at \( N^a_j \), the system will still try to push products at \( N^a_i \) downstream to \( N^a_j \). Because the products not transferred to \( N^a_j \) will later on left among the nodes from \( N^a_{i-1} \) to \( N^a_{j+1} \), which causes higher storage cost. Transferring products from \( N^a_i \) to \( N^a_j \) as soon as possible can reduce the part of buildup left among the nodes from \( N^a_{i-1} \) to \( N^a_{j+1} \). Therefore, the operation rates of nodes from \( N^a_i \) to \( N^a_j \) equal to each other. None of \( \text{product}^a_i \) is built up.

If the operation rate of \( N^a_i \) is greater than the capacity of \( N^a_j \), then \( N^a_i \) does not have sufficient capacity to transfer all of \( \text{product}^a_i \) to \( N^a_j \). However, \( N^a_j \) will still work at its capacity to transfer as much of \( \text{product}^a_i \) as possible to \( N^a_j \) to achieve the minimum total storage cost. Since \( N^a_j \) works at its capacity, which is greater than any operation rate that \( N^a_{j-1} \) can have, then \( \text{product}^a_i \) is built up.

Therefore, \( \text{product}^a_i \) can only start buildup when the operation rate of \( N^a_i \) is greater than the capacity of \( N^a_j \). Once the operation rate of \( N^a_i \) is greater than the capacity of \( N^a_j \), then \( \text{product}^a_j \) is built up definitely. Thus, the time of start of buildup of \( \text{product}^a_i \) is no earlier than the time of start of buildup of \( \text{product}^a_j \).

\[ \square \]

For the operations of the downstream nodes, they satisfy the property same as the operations under the assumption that no inventory storage is allowed upstream of \( N_{\text{dis}} \).

**Lemma 17.** For a serial network system, the following statement is true under the optimal operation: for any \( j \in \{1, 2, ..., \rho\} \), if \( \text{product}^a_j \) is built up in the pre-disruption period then \( \text{product}^a_i \) is also built up in the pre-disruption period for any \( i < j \).

**Proof.** For \( i < j \leq \rho \), the unit storage cost of \( \text{product}^a_i \) is smaller than that of \( \text{product}^a_j \), i.e., \( q_x(N^a_i) < q_x(N^a_j) \). Therefore, whenever \( \text{product}^a_j \) is built up, the nodes downstream of \( N^a_j \) will transfer \( \text{product}^a_j \) to \( \text{product}^a_i \). These nodes among \{\( N^a_{j-1}, ..., N^a_i \)\} will work at their highest capacity to transfer \( \text{product}^a_j \) downstream to achieve minimum storage cost. Since the capacity of \( N^a_i \) is greater than that of \( N^a_{i-1} \), then \( N^a_i \) produces more than \( N^a_{i-1} \) can consumed. Therefore, \( \text{product}^a_i \) is built up. \[ \square \]
Lemma 18. If $u^a(t)$ is the optimal operation rate, then $u^a(t)$ satisfies for each $j \in \{1, 2, \ldots, \rho - 1\}$:

$$u^a_j(t) = c(N^a_j), \text{ when } t_{bx,j} \leq t < t_{bx,j+1}. \quad (8.4)$$

This lemma says that the operation rate of $N^a_j$ equals to its capacity during the interval when product $a^j$ starts buildup and product $a^{j+1}$ hasn’t started buildup.

Proof. We consider the interval when product $a^j$ starts buildup and product $a^{j+1}$ hasn’t started buildup. Based on Lemma 15, we know that no buildup has started between $N^a_\rho$ and $N^a_{j+1}$ during this interval. Therefore, all of $\{N^a_\rho, N^a_{\rho-1}, \ldots, N^a_j\}$ have the same operation rate. These nodes are shown in Figure 8.9.

Denote $i$ as the index such that $\{N^a_i, N^a_{i-1}, \ldots, N^a_{\rho+1}\}$ all have unit storage costs greater than or equal to the unit storage cost of $N^a_j$ and $\{N^a_L, N^a_{L-1}, \ldots, N^a_i\}$ all have unit storage costs less than the unit storage cost of $N^a_j$. Thus, $i \geq \rho + 1$. These nodes are shown in Figure 8.10. If such an $i$ doesn’t exist, then it means that all the nodes upstream of $N^a_\rho$ have lower storage costs than $N^a_j$. In this case, we define $i = \rho$. Based on Lemma 16, we know that $t_{bx,i} \geq t_{bx,j}$.

Proof of Lemma 18, Part I: $u^a_j(t) = c(N^a_j)$, when $\min\{t_{bx,j+1}, t_{bx,i}\} \leq t < t_{bx,j+1}$.

If $i = \rho$ or $t_{bx,j+1} \leq t_{bx,i}$, then this statement doesn’t need to be proved, since the interval $\min\{t_{bx,j+1}, t_{bx,i}\} \leq t < t_{bx,j+1}$ doesn’t exist. Thus, in this part, we only need to consider that $i \geq \rho + 1$ and $t_{bx,j+1} > t_{bx,i}$.

$t_{bx,j+1} > t_{bx,i}$ means the start of the buildup of product $a^{j+1}$ is later than the start of the buildup of product $a^i$. This statement becomes if the start of the buildup of product $a^{j+1}$ is later than product $a^i$, then the operation rate of $N^a_j$ equals to its capacity during the interval after product $a^i$ starts buildup and before product $a^{j+1}$ starts buildup.

Since $N^a_j$ has lower unit storage cost than $N^a_i$, the system will always try to store products at $N^a_j$ instead of $N^a_i$ at any time. This means that all nodes from $N^a_{i-1}$ to $N^a_j$ will always try to match up their operation rates with the operation rate of $N^a_j$, so that the products
can be transferred from \( N_i^a \) to \( N_j^a \). Therefore, the buildup at \( N_j^a \) can only start when the operation rates from \( N_i^a \) to \( N_j^a \) can not match up with the operation rate of \( N_j^a \). This happens when the operation rate of \( N_i^a \) is greater than the capacity of \( N_j^a \), which has the smallest capacity among the nodes from \( N_i^a \) to \( N_j^a \).

Since \( N_j^a \) has the lower storage cost than all the nodes from \( N_i^a \) to \( N_j^a+1 \), then to achieve minimum total storage cost, once the buildup at \( N_i^a \) starts, node \( N_j^a \) will still work at its capacity to transfer as much buildup upstream of \( N_j^a \) as possible to \( N_j^a \). From Lemma 14 we know that \( N_i^a \) will work at its capacity once the buildup at \( N_i^a \) starts. Since then, the products fed into the section from \( N_i^a \) to \( N_j^a+1 \) is more than the product coming out from this section. There will always be buildup existing among the nodes from \( N_j^a \) to \( N_j^a+1 \). Therefore, after product \( a_i \) starts buildup, the operation rate of \( N_j^a \) equals to its capacity. Thus, \( u_{a_j}^{a_j}(t) = c(N_j^a) \) during the interval after product \( a_i \) starts buildup and before product \( a_{j+1} \) starts buildup.

**Proof of Lemma 18, Part II:** \( u_{a_j}^{a_j}(t) = c(N_j^a), \) when \( t_{bx,j} \leq t < \min\{t_{bx,j+1}, t_{bx,i}\} \).

During the interval \([t_{bx,j}, \min\{t_{bx,j+1}, t_{bx,i}\})\), neither \( N_{j+1}^a \) nor \( N_i^a \) starts building up their products. From Lemma 15 and Corollary 2 we know that none of the nodes from \( N_i^a \) to \( N_j^a+1 \) starts building up. Therefore, the operation rates of all the nodes from \( N_i^a \) to \( N_j^a+1 \) equal to the operation rate of \( N_j^a \) during this interval.

At time \( \min\{t_{bx,j+1}, t_{bx,i}\} \), the amount of buildup at \( N_j^a \) will be \( x_{a_j}^a(\min\{t_{bx,j+1}, t_{bx,i}\}) \). No matter what the exact value of this amount is, we will show that the optimal policy for \( N_j^a \) is to start buildup as late as possible.

If \( i = L \), there is no storage existing upstream of \( N_i^a \) during this interval. All the nodes upstream of \( N_i^a \) have the same operation rate of \( N_i^a \). Consider the buildup of product \( a_j \). It should start as late as possible. The later product \( a_j \) starts buildup, the lower total storage cost it can achieve. In order to have the latest starting time and to satisfy the requirement of the amount of buildup of product \( a_j \) at the end of this interval, the operation rate of \( N_i^a \) should be as high as possible after the start of buildup of product \( a_j \). Therefore, \( N_j^a \) operates at its capacity during this interval.

If \( i < L \), consider the products upstream of \( N_i^a \). They all have lower unit storage cost than product \( a_j \). To achieve minimum total storage cost, the system will try to keep these storages upstream of \( N_i^a \) to stay at their locations instead of transferring them to \( N_j^a \). The later product \( a_j \) starts buildup, the fewer storages will be transferred from the nodes upstream of \( N_i^a \) to the node \( N_j^a \), and hence the less total storage cost will happen. Therefore, the optimal operation is to start the buildup of product \( a_j \) as late as possible. Thus, the operation rate of \( N_j^a \) should be as high as possible after the start of buildup of product \( a_j \). Since the buildup at \( N_j^a \) at time \( \min\{t_{bx,j+1}, t_{bx,i}\} \) reaches \( x_{a_j}^a(\min\{t_{bx,j+1}, t_{bx,i}\}) \), the nodes upstream of \( N_i^a \) will produce sufficient products for \( N_j^a \) to transfer. Thus, \( N_j^a \) can reach its capacity rate, which is the highest rate. Therefore, \( N_j^a \) operates at its capacity during this interval.

\[ u_{\rho}^{a_i}(t) = c(N_{\rho}^a), \text{ when } t \in [t_{bx,\rho}, t_{dis}). \] (8.5)
Proof. The proof is similar as the analysis above. At \( t_{dis} \), the buildup at \( N^a_\rho \) is \( x^a_\rho(t_{dis}) \). No matter what exact value of \( x^a_\rho(t_{dis}) \) is, we’ll show the best policy of \( N^a_\rho \) is to start buildup as late as possible.

All nodes upstream of \( N^a_\rho \) have lower unit storage costs than \( N^a_\rho \). To achieve minimum total storage cost, \( N^a_\rho \) should try to keep these storages upstream of \( N^a_\rho \) to stay at their locations instead of transferring them to \( N^a_\rho \). The later product \( a \) starts buildup, the fewer storages will be transferred from the nodes upstream of \( N^a_\rho \) to the node \( N^a_\rho \), and hence the less total storage cost will happen. Therefore, the optimal operation is to start the buildup of product \( a \) as later as possible. Thus, the operation rate of \( N^a_\rho \) should be as high as possible after the start of buildup of product \( a \). Since the buildup at \( N^a_\rho \) at \( t_{dis} \) reaches \( x^a_\rho(t_{dis}) \), the nodes upstream of \( N^a_\rho \) will produce sufficient products for \( N^a_\rho \) to transfer. Thus, \( N^a_\rho \) can reach its capacity rate, which is the highest rate. Therefore, \( N^a_\rho \) operates at its capacity during this interval. \( \square \)

**Lemma 20.** If \( u^a(t) \) is the optimal operation rate, then \( u^a(t) \) satisfies for each \( j \in \{1, 2, \ldots, \rho - 1\} \):

\[
u^a_j(t) = \begin{cases} u^a_{j-1}(t), & \text{when } t_{dis} - \tau_f \leq t < t_{bx,j} , \\ c(N^a_j), & \text{when } t_{bx,j+1} \leq t < t_{dis}. \end{cases}
\]

(8.6)

Proof. Before the time \( t_{bx,j} \), the buildup of product \( j \) does not start. Therefore, the operation of \( N^a_j \) should equal to its downstream nodes. After the time \( t_{bx,j+1} \), then product \( a \) starts to be built up. Therefore, \( N^a_j \) should transfer as much of product \( j+1 \) as possible to product \( j \), because the latter has lower storage cost than the former. To achieve the lowest cost, \( N^a_j \) should work at its highest rate. Since product \( j+1 \) is built up at a rate higher than the capacity of \( N^a_j \), then the highest rate that \( N^a_j \) can reach is its capacity. Therefore, the operation rate of \( N^a_j \) during \([t_{bx,j+1}, t_{dis}]\) is its capacity. \( \square \)

Lemmas 20, 19, and 18 can be combined into the following Corollary.

**Corollary 3.** Given the inventory distribution \( x^a(t_{dis}) \) for a serial network system, if \( u^a(t) \) is the optimal operation rate, then \( u^a(t) \) satisfies for each \( j \in \{1, 2, \ldots, \rho\} \):

\[
u^a_j(t) = \begin{cases} u^a_{j-1}(t), & \text{when } t_{dis} - \tau_f \leq t < t_{dis} - \tau_{bx,j} , \\ c(N^a_j), & \text{when } t_{dis} - \tau_{bx,j} \leq t < t_{dis}, \end{cases}
\]

(8.7)

where

\[
\tau_{bx,j} = \frac{x^a_j(t_{dis})}{c(N^a_j) - c(N^a_{j-1})},
\]

(8.8)

for any \( j \) such that \( 1 \leq j \leq \rho \).

The buildup process can be represented by Figure 8.11.

**Costs in Pre-disruption Period**

Downstream storage cost is:

\[
C_{x,pre-dis}^d = \frac{1}{2} \sum_{j=1}^{\rho} q_k(N^a_j) \frac{[x^a_j(t_{dis})]^2}{c(N^a_j) - c(N^a_{j-1})}.
\]
Upstream storage cost $C_{x,pre-dis}^u$ is calculated based on $S$ matrix.

Denote $Z_j^a$ as the line of $Y_j^a$ from $t_{bx,j}$ to $t_{dis}$ in the buildup process figure, where $j \in \{1, 2, ..., L\}$. Denote $Z_0^a$ as the horizontal axis. Denote Int$_j$ as the intersection of $Z_{j-1}^a$ and $Z_j^a$, where $j \in \{1, 2, ..., \rho\}$. Denote $Z_{j}^{Int}$ as the segment between Int$_j$ and the point $(t_{dis}, X_{bx})$. Denote Int$_{\rho+1}$ as $(t_{dis}, \tilde{X}_{bx})$. Notice that $Z_{\rho}^a$ is $Z_{\rho}^{Int}$. Figure 8.12 shows an example of the notations of lines and intersections in the buildup process figure.

To calculate the storage cost of each product upstream of $N_{\rho}^a$, we need to integrate $x_i^a(t)$ for each $\rho + 1 \leq i \leq L$, and multiply with its unit storage cost $q_x(N_{\rho}^a)$. The total storage cost upstream of $N_{\rho}^a$ is then the summation of the storage cost of each product upstream of $N_{\rho}^a$. The integral of $x_i^a(t)$ in the pre-disruption period is shown in the buildup process figure, as the area between $Z_i^a$ and $Z_{i-1}^a$, and above all the $Z_j^a$ where $j \in \{1, 2, ..., \rho\}$. In the example of Figure 8.12, the integral of $x_L^a(t)$ is the area between $Z_L^a$ and $Z_{L-1}^a$, and above
the horizon axis, \(Z^a_1\) and \(Z^a_2\); the integral of \(x^a_i(t)\) is the area between \(Z^a_{L-1}\) and \(Z^a_{L-2}\), and above \(Z^a_2\). To calculate the total storage cost, we need to determine the area of each \(x^a_i(t)\) in the buildup process figure.

With the lines of \(Z^a_{Int}\), the area of all the \(x^a_i(t)\) where \(\rho + 1 \leq i \leq L\) is divided into a number of small triangle pieces. In the example of Figure 8.12, the area of \(x^a_i(t)\) is divided into three small triangles, by \(Z^a_{Int}\) and \(Z^a_{Int}\). There are four combinations of the two edges between which a triangle locates. They are \((Z^a, Z^a_{Int})\), \((Z^a, Z^a_{Int})\), \((Z^a_{Int}, Z^a_{Int})\), and \((Z^a_{Int}, Z^a_{Int})\). We use such notations to denote four types of triangles. For example, a triangle of type \((Z^a, Z^a_{Int})\) means it is between two \(Z^a\) lines, and a triangle of type \((Z^a_{Int}, Z^a_{Int})\) means it is between two lines, of which the left is a \(Z^a\) line and the right is a \(Z^a_{Int}\) line. In the example of Figure 8.12, the three triangles of \(x^a_i(t)\) from left to right are of type \((Z^a, Z^a_{Int})\), \((Z^a_{Int}, Z^a_{Int})\), and \((Z^a_{Int}, Z^a_{Int})\), respectively.

For \(i \in \{\rho + 1, \rho + 2, \ldots, L\}\) and \(j \in \{1, 2, \ldots, \rho\}\), the order of the slopes of all the \(Z^a_i\) and \(Z^a_{Int}\) is important for calculating the areas of these triangles. Such an order is uniquely determined by an order of all the variables in \(\{t^a_{bx,i}|\rho + 1 \leq i \leq L\}\) and \(\{t^a_{bx,j}|1 \leq j \leq \rho\}\).

For \(j \in \{1, 2, \ldots, L\}\), since all the \(N^a_j\) work at their capacities \(c(N^a_j)\) when building up their products in the pre-disruption period, the slopes of \(Z^a_{Int}\) can be calculated by:

\[
Slope(Z^a_{Int}) = c(N^a_j) - \tilde{d}, \quad \forall j \in \{1, 2, \ldots, L\}.
\]

To calculate the slope of \(Z^a_{Int}\) for \(1 \leq j \leq \rho\), we need to first determine the coordinates of Int. The horizon axis value of Int is:

\[
t_{bx,j} = t_{dis} - \tau_{bx,j} = t_{dis} - \frac{x^a_j(t_{dis})}{c(N^a_j) - c(N^a_{j-1})}, \quad \forall j \in \{1, 2, \ldots, \rho\}.
\]

The intersection point of \(Z^a_j\) is show in Figure 8.13. The vertical axis value of this intersection is:

\[
\sum_{i=1}^{j} x^a_i(t_{dis}) - \tau_{bx,j}[c(N^a_j) - \tilde{d}].
\]
Since \( Z^*_j \) is the segment between \( \text{Int}_j \) and the point \((t_{dis}, \tilde{X}_{bx})\), its slope can be calculated by:

\[
\text{Slope}(Z^*_j) = \frac{\tilde{X}_{bx} - \left\{ \sum_{i=1}^{j} x_i^a(t_{dis}) - \tau_{bx,j} [c(N^a_j) - \bar{d}] \right\}}{t_{dis} - t_{bx,j}} \]

\[
= \sum_{i=j+1}^{p} x_i^a(t_{dis}) + \tau_{bx,j} [c(N^a_j) - \bar{d}] \]

\[
= \frac{c(N^a_j) - c(N^a_{j-1})}{x_j^a(t_{dis})} \sum_{i=j+1}^{p} x_i^a(t_{dis}) + c(N^a_{j}) - \bar{d}.
\]

If \( x_i^a(t_{dis}) = 0 \), it is easy to derive from Lemma 15 that \( x_i^a(t_{dis}) = 0 \), \( \forall i \in \{j, j + 1, \ldots, \rho\} \).

In this case, \( Z^*_j \) collapses into a point, which is at \( \text{Int}_{\rho+1} \). Then, \( \text{Int}_j \) locates at \( \text{Int}_{\rho+1} \).

Moreover, \( Z^*_j \) also collapses into a point at \( \text{Int}_{\rho+1} \). The slope of \( Z^*_j \) has no meaning.

Since \( Z^*_j \) becomes a point, there is no need to consider it when we calculate the integral of inventory, because it does not form a triangle with any other lines in the buildup figure.

For \( j \in \{1, 2, \ldots, \rho\} \) and \( i \in \{\rho + 1, \rho + 2, \ldots, L\} \), if \( t_{bx,i} \leq t_{bx,j} \), then \( \text{Slope}(Z^*_i) \leq \text{Slope}(Z^*_j) \); if \( t_{bx,i} > t_{bx,j} \), then \( \text{Slope}(Z^*_i) > \text{Slope}(Z^*_j) \). Therefore, an \( S \) matrix indicates an order of the slopes of \( \{Z^*_i\} \) and \( \{Z^*_j\} \), which is presented by the following lemma.

Lemma 21. For \( j \in \{1, 2, \ldots, \rho\} \) and \( k \in \{1, 2, \ldots, \gamma\} \),

\[
S_{j,k} = \begin{cases} 
1, & \text{when } \text{Slope}(Z^*_j) \leq \text{Slope}(Z^*_k), \\
0, & \text{when } \text{Slope}(Z^*_j) > \text{Slope}(Z^*_k).
\end{cases}
\]

With \( S \) matrix, we can determine some constraints for \( x_i^a(t_{dis}) \).

\[
(S_{j,k} - 0.5)[\text{Slope}(Z^*_k) - \text{Slope}(Z^*_j)] \leq 0,
\]

where \( j \in \{1, 2, \ldots, \rho\} \) and \( k \in \{1, 2, \ldots, \gamma\} \).

Lemma 22. \( S \) matrix satisfies the following statement:

1. if \( S_{j,k} = 0 \), then \( S_{i,l} = 0 \), \( \forall i \leq j \), and \( l \leq k \), and

2. if \( S_{j,k} = 1 \), then \( S_{i,l} = 1 \), \( \forall i \geq j \), and \( l \geq k \).

Lemma 22 says \( S \) matrix has an echelon form.

For a given \( \gamma \) and \( \rho \), the number of \( S \) matrices which satisfy Definition 15 and Lemma 22 is \( \left( \gamma + \rho - \frac{1}{\gamma} \right) \).

Assumption 12. To order the slopes of all \( \{Z^*_j\} \) and \( \{Z^*_k\} \), if \( \text{Slope}(Z^*_j) = \text{Slope}(Z^*_k) \), then we assume \( Z^*_k \) is in front of \( Z^*_j \) in the order list.

With this assumption, we assume \( Z^*_k \) and \( Z^*_j \) are not overlapping even if their slopes are equal. There is still a triangle between them. However, due to the equality, the area of the triangle between them is 0.
We can extend the $S$ matrix to the 0-th column. Define $S_{j,0} = 0$ for $j \in \{1, 2, ..., \rho - 1\}$, since $\text{Slope}(Z_{\rho}^a) > \text{Slope}(Z_{j}^{\text{Int}})$ ($t_{bx,\rho} > t_{bx,j}$). And, define $S_{\rho,0} = 1$, since $\text{Slope}(Z_{\rho}^a) = \text{Slope}(Z_{\rho}^{\text{Int}})$ ($t_{bx,\rho} = t_{bx,\rho}$). We can also extend the $S$ matrix to the 0-th row by assuming $\text{Slope}(Z_{0}^{\text{Int}}) := 0$. Then, $S(0, k) = 0$ for $k \in \{0, 1, ..., \gamma\}$.

Now, consider the four types of triangles of integral of inventory.

**Type** $(Z^a, Z^a)$ Denote the left and right edges as $(Z_i^a, Z_{i-1}^a)$, shown in Figure 8.14

![Figure 8.14: The triangle of Type $(Z^a, Z^a)$](image)

Then there is no $Z_j^{\text{Int}}$ between them. Therefore, there exists an $j$ in $\{1, 2, ..., \rho\}$ such that:

$$\text{Slope}(Z_{j-1}^{\text{Int}}) < \text{Slope}(Z_i^a) < \text{Slope}(Z_{i-1}^{\text{Int}}) \leq \text{Slope}(Z_j^{\text{Int}}).$$

Notice that $\text{Slope}(Z_0^{\text{Int}}) := 0$ and $i$ is in $\{\rho + 1, \rho + 2, ..., L\}$. This represents a sub-matrix in $S$ as:

$$S_{j-1}^{\text{Int}} S_{i}^{\text{Int}} = \begin{bmatrix} Z_i^a & Z_{i-1}^a \\ Z_{j-1}^{\text{Int}} & 0 & 0 \\ Z_j^{\text{Int}} & 1 & 1 \end{bmatrix}.$$  

That is, $S_{j-1,i-\rho} = 0$ and $S_{j,i-1-\rho} = 1$.

The bottom edge of this triangle is $Z_{j-1}^a$. Notice that $Z_0^a$ is denoted as the horizon axis. The area of this triangle equals to the difference of the area of the triangle bounded by $\{Z_i^a, Z_{j-1}^a, \text{the vertical axis}\}$ and the area of the triangle bounded by $\{Z_i^a, Z_{j-1}^a, \text{the vertical axis}\}$. The area bounded by $\{Z_i^a, Z_{j-1}^a, \text{the vertical axis}\}$ is:

$$\frac{1}{2} \tau_{bx,i} \sum_{m=j}^\rho x_m^a(t_{\text{dis}}) = \frac{1}{2} \frac{\sum_{m=j}^\rho x_m^a(t_{\text{dis}})}{\text{Slope}(Z_i^a) - \text{Slope}(Z_{j-1}^a)} \sum_{m=j}^\rho x_m^a(t_{\text{dis}}).$$

The area bounded by $\{Z_i^a, Z_{j-1}^a, \text{the vertical axis}\}$ is:

$$\frac{1}{2} \tau_{bx,i-1} \sum_{m=j}^\rho x_m^a(t_{\text{dis}}) = \frac{1}{2} \frac{\sum_{m=j}^\rho x_m^a(t_{\text{dis}})}{\text{Slope}(Z_i^a) - \text{Slope}(Z_{j-1}^a)} \sum_{m=j}^\rho x_m^a(t_{\text{dis}}).$$
Then, the area of the \((Z^a, Z^a)\) triangle is:

\[
\frac{1}{2} \frac{\left[ \sum_{m=j}^{\rho} x_m^a(t_{dis}) \right]^2}{\text{Slope}(Z_i^a) - \text{Slope}(Z_{j-1}^a)} - \frac{1}{2} \frac{\left[ \sum_{m=j}^{\rho} x_m^a(t_{dis}) \right]^2}{\text{Slope}(Z_{i-1}^a) - \text{Slope}(Z_{j-1}^a)}.
\]

The unit storage cost of the product associated with this triangle is \(q_x(N_i^a)\). Therefore, the storage cost represented by this triangle is:

\[
C_{j,i}^{(a,a)} = q_x(N_i^a) \left\{ \frac{1}{2} \frac{\left[ \sum_{m=j}^{\rho} x_m^a(t_{dis}) \right]^2}{\text{Slope}(Z_i^a) - \text{Slope}(Z_{j-1}^a)} - \frac{1}{2} \frac{\left[ \sum_{m=j}^{\rho} x_m^a(t_{dis}) \right]^2}{\text{Slope}(Z_{i-1}^a) - \text{Slope}(Z_{j-1}^a)} \right\}
\]

\[
= q_x(N_i^a) \left\{ \frac{1}{2} \frac{\left[ \sum_{m=j}^{\rho} x_m^a(t_{dis}) \right]^2}{2 c(N_i^a) - c(N_j^a)} - \frac{1}{2} \frac{\left[ \sum_{m=j}^{\rho} x_m^a(t_{dis}) \right]^2}{2 c(N_{i-1}^a) - c(N_j^a)} \right\}.
\]

If \(c(N_i^a) = c(N_{i-1}^a)\) or \(c(N_j^a) = c(N_{j-1}^a)\), then \(Z_{i-1}^a\) is overlapping with \(Z_i^a\) or \(Z_{i-1}^a\), respectively. The area of this triangle is 0. Thus, \(C_{j,i}^{(a,a)} = 0\).

**Type \((Z^a, Z^{Int})\)** Denote the left and right edges as \((Z_i^a, Z_j^{Int})\). Then there is no \(Z_j^{Int}\) or \(Z_{i-1}^{Int}\) between them. Therefore,

\[
\text{Slope}(Z_j^{Int}) < \text{Slope}(Z_i^a) \leq \text{Slope}(Z_j^{Int}) < \text{Slope}(Z_{i-1}^a).
\]

Notice that \(\text{Slope}(Z_j^{Int}) := 0\), \(j \in \{1, 2, ..., \rho - 1\}\) and \(i\) is in \(\{\rho + 1, \rho + 2, ..., L\}\). This represents a sub-matrix in \(\mathbf{S}\) as:

\[
\begin{pmatrix}
Z_j^{Int} & Z_{i-1}^a \\
Z_{i-1}^{Int} & 0 & 0 \\
Z_j^{Int} & 0 & 1
\end{pmatrix}.
\]

That is, \(S_{j-1,i-\rho} = 0\), \(S_{j,i-\rho} = 1\) and \(S_{j,i-1-\rho} = 0\). The bottom edge of this triangle is \(Z_{j-1}^a\), and unit storage cost of this product is \(q_x(N_i^a)\). Similarly, the storage cost represented by this triangle of type \((Z^a, Z^{Int})\) is formulated as:

\[
C_{j,i}^{(a,Int)} = q_x(N_i^a) \left\{ \frac{1}{2} \frac{\left[ \sum_{m=j}^{\rho} x_m^a(t_{dis}) \right]^2}{\text{Slope}(Z_i^a) - \text{Slope}(Z_{j-1}^a)} - \frac{1}{2} \frac{\left[ \sum_{m=j}^{\rho} x_m^a(t_{dis}) \right]^2}{\text{Slope}(Z_j^{Int}) - \text{Slope}(Z_{j-1}^a)} \right\}
\]

\[
= q_x(N_i^a) \left\{ \frac{1}{2} \frac{\left[ \sum_{m=j}^{\rho} x_m^a(t_{dis}) \right]^2}{2 c(N_i^a) - c(N_j^{Int})} - \frac{1}{2} \frac{\left[ \sum_{m=j}^{\rho} x_m^a(t_{dis}) \right]^2}{2 c(N_j^{Int}) - c(N_{j-1}^a)} \right\}.
\]

If \(c(N_i^a) = c(N_{j-1}^a)\), then \(Z_i^a\) and \(Z_{j-1}^a\) are overlapping. The area of this triangle is 0. Thus, \(C_{j,i}^{(a,Int)} = 0\).

**Type \((Z^{Int}, Z^a)\)** Denote the left and right edges as \((Z_i^{Int}, Z_j^a)\). Then there is no \(Z_j^{Int}\) and \(Z_{i+1}^a\) between them. Therefore,

\[
\text{Slope}(Z_{i+1}^a) \leq \text{Slope}(Z_j^{Int}) < \text{Slope}(Z_i^a) \leq \text{Slope}(Z_{j+1}^{Int}).
\]
Notice that \( i \) is in \( \{\rho, \rho + 1, \ldots, L - 1\} \) and \( j \) is in \( \{1, 2, \ldots, \rho - 1\} \). This represents a sub-matrix in \( S \) as:

\[
\begin{bmatrix}
0 & 1 \\
1 & 1 \\
\end{bmatrix}
\]

That is, \( S_{j+1-i} = 1 \), \( S_{j-i} = 0 \) and \( S_{j+1-i} = 1 \). The bottom edge is \( Z_j^a \) and the unit storage cost of this product is \( q_x(N_{i+1}^a) \). Similarly, the storage cost represented by this triangle of type \((Z_{\text{Int}}, Z^a)\) is formulated as:

\[
C_{j,i}^{(\text{Int},a)} = q_x(N_{i+1}^a) \left\{ \begin{array}{c}
\frac{1}{2} \frac{[\sum_{m=j+1}^{\rho} x_m^a(t_{\text{dis}})]^2}{\text{Slope}(Z_j^{\text{Int}}) - \text{Slope}(Z_j^a)} - \frac{1}{2} \frac{[\sum_{m=j+1}^{\rho} x_m^a(t_{\text{dis}})]^2}{\text{Slope}(Z_{j+1}^{\text{Int}}) - \text{Slope}(Z_j^a)} \\
\frac{1}{2} \frac{[\sum_{m=j+1}^{\rho} x_m^a(t_{\text{dis}})]^2}{c(N_j^a) - c(N_{j-1}^a)} - \frac{1}{2} \frac{[\sum_{m=j+1}^{\rho} x_m^a(t_{\text{dis}})]^2}{c(N_j^a) - c(N_j^a)}
\end{array} \right\}.
\]

If \( c(N_i^a) = c(N_j^a) \), then \( Z_i^a \) and \( Z_j^a \) are overlapping. The area of this triangle is 0. Thus, \( C_{j,i}^{(\text{Int},a)} = 0 \).

**Type \((Z_{\text{Int}}, Z_{\text{Int}})\)** Denote the left and right edges as \((Z_{\text{Int}}^{\text{Int}}, Z_{\text{Int}}^{\text{Int}})\). Then there is no \( Z_i^a \) between them. Therefore,

\[
\text{Slope}(Z_{i+1}^a) \leq \text{Slope}(Z_j^{\text{Int}}) < \text{Slope}(Z_{j+1}^{\text{Int}}) < \text{Slope}(Z_i^a).
\]

Notice that \( i \) is in \( \{\rho, \rho + 1, \ldots, L - 1\} \). \( j \) is in \( \{1, 2, \ldots, \rho - 2\} \). This represents a sub-matrix in \( S \) as:

\[
\begin{bmatrix}
0 & 1 \\
1 & 1 \\
\end{bmatrix}
\]

That is, \( S_{j+1-i} = 1 \) and \( S_{j+1-i} = 0 \). The bottom edge is \( Z_j^a \) and the unit storage cost of this product is \( q_x(N_{i+1}^a) \). Similarly, the storage cost represented by this triangle of type \((Z_{\text{Int}}, Z_{\text{Int}})\) is formulated as:

\[
C_{j,i}^{(\text{Int},\text{Int})} = q_x(N_{i+1}^a) \left\{ \begin{array}{c}
\frac{1}{2} \frac{[\sum_{m=j+1}^{\rho} x_m^a(t_{\text{dis}})]^2}{\text{Slope}(Z_j^{\text{Int}}) - \text{Slope}(Z_j^a)} - \frac{1}{2} \frac{[\sum_{m=j+1}^{\rho} x_m^a(t_{\text{dis}})]^2}{\text{Slope}(Z_{j+1}^{\text{Int}}) - \text{Slope}(Z_j^a)} \\
\frac{1}{2} \frac{[\sum_{m=j+1}^{\rho} x_m^a(t_{\text{dis}})]^2}{c(N_j^a) - c(N_{j-1}^a)} - \frac{1}{2} \frac{[\sum_{m=j+1}^{\rho} x_m^a(t_{\text{dis}})]^2}{c(N_{j+1}^a) - c(N_j^a)}
\end{array} \right\}.
\]

To determine all the triangles with a given \( S \) matrix, we just need to find all the sub-matrices associated with all the types of triangles. We can simply check each element in \( S \) matrix and test the neighbor elements around it. Based on the value of all these elements, we can determine whether a triangle appears. Notice that a certain triangle appears only once in total when checking different elements in \( S \) matrix. For each element \( S_{j,k} \) where \( 1 \leq j \leq \rho \) and \( 0 \leq k \leq \gamma \), we denote \( C_{j,k} \) as the storage cost represented by the triangles which appear when checking \( S_{j,k} \). Therefore, the total storage cost of the buildup upstream
of $N_{\text{dis}}$ during the pre-disruption period, which is the total storage cost represented by all the triangles, can be formulated as:

$$C^u_{x,\text{pre-dis}} = \sum_{j=1}^{\rho} \sum_{k=0}^{\gamma} C_{j,k}.$$ 

For $C_{j,k}$, we use four coefficients, $\{a_{j,k}^{(a,\text{Int})}, a_{j,k}^{(a,a)}, a_{j,k}^{(\text{Int},\text{Int})}, a_{j,k}^{(\text{Int},a)}\}$, to formulate the cost. Then $C_{j,k}$ becomes:

$$C_{j,k} = a_{j,k}^{(a,\text{Int})} C_{j,k+\rho}^{(a,\text{Int})} + a_{j,k}^{(a,a)} C_{j,k+\rho}^{(a,a)} + a_{j,k}^{(\text{Int},\text{Int})} C_{j,k+\rho}^{(\text{Int},\text{Int})} + a_{j,k}^{(\text{Int},a)} C_{j,k+\rho}^{(\text{Int},a)}.$$ 

The coefficients are either 0 or 1 to count in the storage cost of the triangle which appears when checking $S_{j,k}$. For example, if a type of $(Z^a, Z^a)$ triangle and a type of $(Z^{\text{Int}}, Z^a)$ appear when checking $S_{j,k}$, then $a_{j,k}^{(a,a)} = a_{j,k}^{(\text{Int},a)} = 1$ and $a_{j,k}^{(\text{Int},\text{Int})} = a_{j,k}^{(a,\text{Int})} = 0$.

To sum up, the relationship of $S$ matrix, the types of triangles, and coefficients in $C_{j,k}$ can be represented by Table 8.1.

Then, the coefficients in $C_{j,k}$ can be calculated by $S$ matrix with the formulas below:

$$a_{j,k}^{(a,\text{Int})} = \begin{cases} S_{j,k}(1 - S_{j-1,k})(1 - S_{j,k+1}), & \text{when } 1 \leq k \leq \gamma, \\ 0, & \text{when } k = 0. \end{cases}$$

$$a_{j,k}^{(a,a)} = \begin{cases} S_{j,k}(1 - S_{j-1,k})S_{j,k-1}, & \text{when } 1 \leq k \leq \gamma, \\ 0, & \text{when } k = 0. \end{cases}$$

$$a_{j,k}^{(\text{Int},\text{Int})} = \begin{cases} (1 - S_{j,k})S_{j,k_1}(1 - S_{j+1,k}), & \text{when } 1 \leq j \leq \rho - 1, \\ 0, & \text{when } j = \rho. \end{cases}$$

$$a_{j,k}^{(\text{Int},a)} = \begin{cases} (1 - S_{j,k})S_{j,k+1}S_{j+1,k}, & \text{when } 1 \leq j \leq \rho - 1, \\ 0, & \text{when } j = \rho. \end{cases}$$

Notice that we define $S_{j,\gamma+1} := 0$, for $i \in \{1, 2, ..., \rho - 1\}$, since it appears in these equations but has no physical meaning.

### 8.2.3 Formulating the Optimization Problem

For a given $S$ matrix, the objective function is:

$$C = C^d_{x,\text{dr-dis}} + C^d_{x,\text{pre-dis}} + C^u_{x,\text{pre-dis}}.$$ 

Constraints are:

1. $\sum_{j=1}^{\rho} x_j^a(t_{\text{dis}}) = X_{bx} := \min\{\tau_{\text{dis}} d, \tau_j [e(N_{L}^a) - 1]\}$, and
2. $x_j^a(t_{\text{dis}}) \geq 0, \forall j \in \{1, 2, ..., \rho\}$, and
3. $(S_{j,k} - 0.5)[\text{Slope}(Z_{k+\rho}^a) - \text{Slope}(Z_{j}^{\text{Int}})] \leq 0$, where $j \in \{1, 2, ..., \rho\}$ and $k \in \{1, 2, ...\gamma\}$, and
4. $t_{bx,j} \leq t_{bx,j+1}, \forall j \in \{1, 2, ..., \rho - 1\}$, and
To solve the problem, we can use Quadratic Programming.

### Table 8.1: \( S \) matrix, the types of triangles, and coefficients in \( C_{j,k} \)

<table>
<thead>
<tr>
<th>( S_{jk} )</th>
<th>( S_{j+1,k} )</th>
<th>( S_{j-1,k} )</th>
<th>( S_{j,k+1} )</th>
<th>triangle type</th>
<th>( \alpha(a,a) )</th>
<th>( \alpha(a,\text{Int}) )</th>
<th>( \alpha(\text{Int},a) )</th>
<th>( \alpha(\text{Int,Int}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>either</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>either</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>either</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>either</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>either</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>either</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

8.3 Case Study

In this section, we use the same practical system in Chapter 7 for a case study to illustrate the model and approach. Its network matrix \( B \), \( B(i,i) = -1 \), and \( B(i, i+1) = 1 \), for \( i \in \{1, 2, .., 6\} \). All other elements in the \( B \) matrix are zeros.

The demand is on the final node \( N_0 \). Therefore, \( D = [1, 0, 0, 0, 0] \). The value of the
demand is:
\[ d = 20 \text{ products/hour}. \]

Then, the production rate in nominal state is 20 products/hour, i.e., \( \alpha_i = 20 \) for \( i \in \{0, 1, 2, ..., 6\} \).

Suppose that node \( N_4 \) needs some maintenance which requires it to be shut down for 2 hours, i.e., \( \tau_{dis} = 2 \) hours. And the system is notified about this event 8 hours before it begins, i.e., \( \tau_f = 8 \) hours.

The capacities (products/hour) at nodes from \( N_6 \) to \( N_0 \) in the Nominal State are:
\[ c(N_6) = +\infty, c(N_5) = 30, c(N_4) = 36, c(N_3) = 36, c(N_2) = 40, c(N_1) = 24 \text{ and } c(N_0) = 20. \]

And the capacity of \( N_4 \) in the Disruption State is \( c(N_4) = 0 \).

\( R \) is defined as a \( 7 \times 7 \) Identity Matrix.

The unit storage costs (dollars/(product · hour)) are:
\[ q_x(N_6) = 0, q_x(N_5) = y, q_x(N_4) = 20y, q_x(N_3) = 20y, q_x(N_2) = 3y, \text{ and } q_x(N_1) = y. \]

### 8.3.1 Normalization

The normalization is the same as in Chapter 7.

The normalized demand is \( \tilde{d} = 1 \). We consider 20 products as 1 load. Then, the normalized capacities in the Nominal State are
\[ \tilde{c} = [\tilde{c}(N_0), \tilde{c}(N_1), \tilde{c}(N_2), \tilde{c}(N_3), \tilde{c}(N_4), \tilde{c}(N_5), \tilde{c}(N_6)]^T = [1, 1.2, 2, 1.8, 1.8, 1.5, +\infty]^T. \]

And, the normalized unit storage costs are:
\[ \tilde{q}_x = [\tilde{q}_x(N_1), \tilde{q}_x(N_2), \tilde{q}_x(N_3), \tilde{q}_x(N_4), \tilde{q}_x(N_5), \tilde{q}_x(N_6)] = [20y, 60y, 400y, 400y, 20y, 0]. \]

The normalized capacities and storage costs are shown in Figure 7.9.

### 8.3.2 Locating \( s \)- and \( c \)-nodes

Figure 8.15 shows the locations of all the \( s \)- and \( c \)-nodes.

### 8.3.3 Determine \( \rho \) and \( \gamma \)

The variables determining \( \rho \) and \( \gamma \) are as follows:
\[ [\tilde{c}(N^u_{c_1}), \tilde{c}(N^u_{c_2})] = [1.5, 1.8]. \]
\[ [\tilde{c}(N^d_{c_1}), \tilde{c}(N^d_{c_2}), \tilde{c}(N^d_{c_3})] = [1, 1.2, 1.8]. \]
\[ \tilde{q}_x(N^u_{s_1}) = 20y. \]
\[ [\tilde{q}_x(N^d_{s_1}), \tilde{q}_x(N^d_{s_2})] = [20y, 60y]. \]

By the definition, \( \rho = 2 \), and \( \gamma = 1 \).
8.3.4 Generate $S$ Matrices

With $\rho = 2$ and $\gamma = 1$, $S$ matrices can be generated as follows:

$$S^{(1)} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}, \quad S^{(2)} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}.$$ 

8.3.5 Quadratic Program

For each $S$ matrix, we run a quadratic program. Therefore, we get 2 solutions of $\tilde{x}_s^{d}(t_{dis})$ and total cost $C$. They are as follows:

$$\tilde{x}_s^{d,(1)}(t_{dis}) = [1.25, 0.75]^T, \quad C^{(1)} = 165y, \quad \tilde{x}_s^{d,(2)}(t_{dis}) = [0.8, 1.2]^T, \quad C^{(2)} = 192y.$$ 

By comparing the total cost, we determine that $C^{(1)}$ is the lowest and the optimal solution is $\tilde{x}_s^{d,(1)}(t_{dis})$, i.e.:

$$\tilde{x}_s^{d}(t_{dis}) = [1.25, 0.75]^T.$$ 

The optimal total storage cost is 165y. The optimal buildup at $t_{dis}$ is:

$$[x_1(t_{dis}), x_2(t_{dis}), x_3(t_{dis}), x_4(t_{dis}), x_5(t_{dis}), x_6(t_{dis})] = [25, 15, 0, 0, 0, 0].$$

8.3.6 Determine the Optimal Operation

We can calculate all the time variables as follows:

$$[\tau_1, \tau_2] = [2, 0.625].$$

$$[\tau_{bx,1}^d, \tau_{bx,2}^d] = [6.25, 1.25].$$

$$\tau_a^{u} = 2.5.$$ 

The optimal operation is shown in Figure 8.16.

At time $t_{dis} - 6.25$, $N_1$ starts buildup. At time $t_{dis} - 2.5$, products are built up at $N_5$, which is upstream of the disrupted node. Then, at time $t_{dis} - 1.25$, the buildup at $N_5$ are pushed downstream to node $N_2$. 

Figure 8.15: Locations of $s$- and $c$- nodes
Compared with the results shown by Figure 7.10 in Section 7.3, the operation allows the buildup upstream of the disrupted node. The total cost is 165\$/y, less than the optimal cost of 180\$/y in Section 7.3. In this case study, allowing upstream buildup achieves lower total storage cost.

8.4 Conclusion

In this Chapter, we solve the optimal control of serial network systems. This extends the results in Chapter 7 to more general cases, since it allows the buildup upstream of $N_{dis}$. We still formulate the optimization problem based on the given $\tilde{x}(t_{dis})$. However, instead of solving a single optimization problem, we solve multiple problems based on the interleaving orders of time variables. After that, we compare the results of multiple optimization problems and choose the real optimal one as the solution.
Chapter 9

Optimal Control of A Special Type of Assembly Tree Network Systems in Continuous Time Domain

In this chapter, we will analyze a special type of assembly tree network systems. Such a network system is made up of only one disrupted chain and one branch. We will first introduce three assumptions to identify three cases, respectively. Then for each case, the optimal operation is determined, and most of this chapter discusses the most difficult one of these three cases.

9.1 The Specifications of the Assembly Tree Network Systems Consisting of One Disrupted Chain and One Branch

The structure of assembly tree network systems consisting of one disrupted chain and one branch is shown by Figure 9.1.

$N_{\lambda}$ is also denoted as $N_{br}^0$.

In the following analysis, we will study three cases defined by the assumptions below, respectively:
Assumption 13 (Storage Costs Upstream Lower than Downstream). The storage costs upstream of $N_\lambda$ are all smaller than or equal to the storage costs downstream of $N_\lambda$. That is, for any $k$ such that $l \geq k > \lambda$ and any $j$ such that $\lambda \geq j \geq 1$, it is true that:

$$\tilde{q}_x(N_k) \leq \tilde{q}_x(N_j).$$

Assumption 14 (Capacity Upstream Lower than Branch). The capacities upstream of $N_\lambda$ are all smaller than or equal to the capacities in the branch. That is, for any $k$ such that $l \geq k > \lambda$ and any $i$ such that $n \geq i \geq 0$, it is true that:

$$\tilde{c}(N_k) \leq \tilde{c}(N_i^{br}).$$

Assumption 15 (Storage Costs Upstream greater than Downstream, Capacity Upstream greater than Branch). The storage costs upstream of $N_\lambda$ are all greater than the storage costs downstream of $N_\lambda$, and the capacities upstream of $N_\lambda$ are all greater than the capacities in the branch. That is, for any $k$ such that $l \geq k > \lambda$, any $j$ such that $\lambda \geq j \geq 1$ and any $i$ such that $n \geq i \geq 0$, it is true that:

$$\tilde{q}_x(N_k) > \tilde{q}_x(N_j), \text{ and } \tilde{c}(N_k) > \tilde{c}(N_i^{br}).$$

Normalization

In the following analysis, when mentioning operation rates, unit storage costs and capacities, we are talking about their normalized values, respectively. The normalization approach was introduced in Section 5.1.

Lost Demand and Operation Cost

To minimize the lost demand, the system needs to buildup sufficient storage downstream of $N_l$ before the disruption happens. Denote the total amount of buildup as $\tilde{X}_{bx}$. The total amount of buildup needed is $\tilde{d} \tau_{dis}$. Therefore, $\tilde{X}_{bx}$ should not be greater than $\tilde{d} \tau_{dis}$ in order to avoid unnecessary buildup costs, including operation cost and storage cost. Besides, due to the capacity of $N_l$, $\tilde{X}_{bx}$ can not be greater than $\tau_f [\tilde{c}(N_l) - \tilde{d}]$. Thus,

$$\tilde{X}_{bx} = \min\{\tilde{d} \tau_{dis}, \tau_f [\tilde{c}(N_l) - \tilde{d}]\}.$$

Once the total amount of buildup satisfies this equation, the lost demand is minimized.

With the buildup, the integral of the operation rate of $N_0$ over $[t_{dis} - \tau_f, t_{dis} + \tau_{dis})$ is:

$$\int_{t_{dis} - \tau_f}^{t_{dis} + \tau_{dis}} \tilde{u}_0(t) dt = \tilde{d} \tau_f + \tilde{X}_{bx}.$$

Since the inventory is 0 at time $t_{dis} - \tau_f$ and time $t_{dis} + \tau_{dis}$. Then, the integral of the operation rate of any node should be equal to each other. Thus,

$$\int_{t_{dis} - \tau_f}^{t_{dis} + \tau_{dis}} \tilde{u}_j(t) dt = \int_{t_{dis} - \tau_f}^{t_{dis} + \tau_{dis}} \tilde{u}_i^{br}(t) dt = \tilde{d} \tau_f + \tilde{X}_{bx}.$$
Therefore the total operation cost is:

\[ C_u = \left[ \sum_j \tilde{q}_u(N_j) + \sum_i \tilde{q}_u(N_{br}^i) \right] (\tilde{d}_f + \tilde{X}_{br}), \]

which is a constant if the lost demand is minimized.

In the following analysis, we don’t consider lost demand and operation cost in the objective functions, but consider the value of \( \tilde{X}_{br} \) as a constraint to our optimization problem. The only objective function we consider becomes the total storage cost.

### 9.2 Operation under Assumption 13 or 14

It can be proved that the system in these cases can be aggregated into a serial network system. Therefore, we can apply the results of serial network system to determine the optimal operation.

For the system under Assumption 13, we know the node with the lowest unit storage cost among \( \{N_j | N_l \geq N_j \geq N_0 \} \) is upstream of \( N_\lambda \). Let’s denote this node as \( N_{s1} \). Therefore, there is no need to build up products downstream of \( N_{s1} \), since these locations have higher storage costs than \( N_{s1} \). At any time, the nodes downstream of \( N_{s1} \) should keep the operation rate at demand to avoid transferring buildup from \( N_{s1} \) to nodes downstream of it. Therefore, \( N_\lambda \) should work at the rate of demand. And no buildup is needed in the branch. Thus, the branch nodes have the same rate of demand, and they can be aggregated with \( N_\lambda \). The system becomes a serial network system.

For the system under Assumption 14, we still denote \( N_{s1} \) as the node with the lowest unit storage cost among \( \{N_j | N_l \geq N_j \geq N_0 \} \). If \( N_{s1} \) is upstream of \( N_\lambda \), then based on the analysis of Assumption 13, we know the system can be aggregated into a serial network system. If \( N_{s1} \) is downstream of \( N_\lambda \), then we consider the node with the lowest capacity among \( \{N_j | N_l \geq N_j \geq N_{s1} \} \). Based on Assumption 14, we know it is upstream of \( N_\lambda \), and denote it as \( N_{c1} \). Since nodes between \( N_{c1} \) and \( N_{s1} \) all have sufficient capacities to match up with the operation rate of \( N_{c1} \), and \( N_{s1} \) has the lowest unit storage costs, then the nodes among \( \{N_j | N_{c1} \geq N_j \geq N_{s1} \} \) should all work at the same rate of \( N_{c1} \) to transfer the products among them to \( N_{s1} \). Therefore, \( N_\lambda \) works at the same rate of \( N_{c1} \). Since \( \{N_{br}^i | N_{n}^{br} \geq N_{br}^i \geq N_{0}^{br} \} \) all have capacities greater than or equal to the capacity of \( N_{c1} \), no matter what value of the operation rate of \( N_{c1} \) is, these branch nodes can work at the same rate to feed \( N_\lambda \). Therefore, there is no buildup needed in the branch. Thus, the system can be aggregated into a serial network system.

### 9.3 Operation under Assumption 15

#### 9.3.1 Aggregation

**Locating s- and c- nodes**

An example of the locations of s- and c- nodes is shown in Figure 9.2.
Follow the procedure to locate $s$- and $c$- nodes.

First, locate the $s$- and $c$- nodes among the node $\{N_j | N_\lambda \succ N_j \succ N_0 \}$. The procedure is similar as the one locating the $s$- and $c$- nodes downstream of $N_{dis}$ in a serial network system introduced in Subsection 7.2.2. Since some notations are different, the procedure is reworded as follows:

1. Denote $N_0$ as $N_{c_0}$. $k = 0$.

2. Search among the nodes $\{N_j | N_\lambda \succ N_j \succ N_{c_k} \}$. Find the node with the lowest unit storage cost. Denote it as $N_{s_{k+1}}$. If there is more than one node having the lowest unit storage cost, then choose the most upstream one.

3. Search among the nodes $\{N_j | N_\lambda \succ N_j \succ N_{s_{k+1}} \}$. Find the node with the lowest capacity, and denote it as $N_{c_{k+1}}$. If there is more than one node having the lowest capacity, then choose the most upstream one.

4. If $N_{c_{k+1}}$ is $N_\lambda$, then the locating process ends; otherwise, $k = k + 1$ and go back to step 2.

This procedure locates the $s$- and $c$- nodes among the node $\{N_j | N_\lambda \succ N_j \succ N_0 \}$. $N_\lambda$ is one of the $c$-nodes, $N_{c_m}$.

Then, we locate $s$- and $c$- nodes among $\{N_j | N_I \succ N_j \succ N_\lambda \}$ with the similar procedure.

1. Denote $N_\lambda$ as $N_{c_m}$, $k = m$.

2. Search among the nodes $\{N_j | N_I \succ N_j \succ N_{c_k} \}$. Find the node with the lowest unit storage cost. Denote it as $N_{s_{k+1}}$. If there is more than one node having the lowest unit storage cost, then choose the most upstream one.

3. Search among the nodes $\{N_j | N_I \succ N_j \succ N_{s_{k+1}} \}$. Find the node with the lowest capacity, and denote it as $N_{c_{k+1}}$. If there is more than one node having the lowest capacity, then choose the most upstream one.

4. If $N_{c_{k+1}}$ is $N_I$, then the locating process ends; otherwise, $k = k + 1$ and go back to step 2.
Since \( N_{l} \) is the last \( c \)-node, define index \( M \) such that \( N_{l} = N_{c_{M}} \).

With Assumption [15], we know \( \tilde{c}(N^{br}_{1}) < \tilde{c}(N^{br}_{c_{M}+1}) \). Consider the nodes among \( \{N_{j} \mid N_{c_{k}} \supseteq N_{j} \supseteq N_{s_{k}} \} \). These nodes’ capacities are all greater than or equal to that of \( N_{c_{k}} \). No matter what operation rate of \( N_{c_{k}} \) is, they can work at the same rate of \( N_{c_{k}} \) to push the products to \( N_{s_{k}} \), which has the lowest unit storage cost among these nodes. Therefore, to achieve minimum storage cost, the nodes among \( \{N_{j} \mid N_{c_{k}} \supseteq N_{j} \supseteq N_{s_{k}} \} \) should work at the same rate of \( N_{c_{k}} \) to push all the products among these nodes to \( N_{s_{k}} \).

Consider another set of nodes \( \{N_{j} \mid N_{s_{k+1}} \supseteq N_{j} \supseteq N_{c_{k}} \} \). These nodes’ capacities are all greater than or equal to that of \( N_{c_{k}} \). When the operation rate of \( N_{s_{k+1}} \) is higher than that of \( N_{c_{k}} \), there will be products built up among these nodes. Since the unit storage cost at \( N_{s_{k+1}} \) is the lowest among these nodes, the nodes \( \{N_{j} \mid N_{s_{k+1}} \supseteq N_{j} \supseteq N_{c_{k}} \} \) should work at the same rate of \( N_{c_{k}} \) to avoid transferring products from \( N_{s_{k+1}} \) to any other locations among these nodes which have higher storage costs. Therefore, to achieve minimum storage cost, the operation rate of \( \{N_{j} \mid N_{s_{k+1}} \supseteq N_{j} \supseteq N_{c_{k}} \} \) should be the same as that of \( N_{c_{k}} \), and no storage will be built up among these nodes.

The analysis above shows that there is no buildup among \( \{N_{j} \mid N_{s_{k+1}} \supseteq N_{j} \supseteq N_{s_{k}} \} \) for the optimal operation. That means the nodes among \( \{N_{j} \mid N_{s_{k+1}} \supseteq N_{j} \supseteq N_{s_{k}} \} \) have the same operation rate. Therefore, \( \{N_{j} \mid N_{s_{k+1}} \supseteq N_{j} \supseteq N_{s_{k}} \} \) can be considered as one aggregated node.

Next, we locate the \( s \)- and \( c \)-nodes in the branch. The procedure is similar as the one locating the \( s \)- and \( c \)-nodes upstream of \( N_{dis} \) in a serial network system introduced in Subsection [8.1.1]. Based on the analysis above, we use \( \tilde{q}_{x}(N_{s_{m}}) \) as the unit storage cost of \( N^{br}_{0} \).

1. Search among the nodes \( \{N^{br}_{i} \mid N^{br}_{n} \supseteq N^{br}_{i} \supseteq N^{br}_{0} \} \). Find the node with the lowest capacity. Denote it as \( N^{br}_{c_{1}} \). If there is more than one node having the lowest capacity, then choose the most downstream one. Let \( k = 1 \).

2. Search among the nodes \( \{N^{br}_{i} \mid N^{br}_{c_{k}} \supseteq N^{br}_{i} \supseteq N^{br}_{0} \} \). Use \( \tilde{q}_{x}(N_{s_{m}}) \) as the unit storage cost of \( N^{br}_{0} \). Find the node with the lowest unit storage cost. Denote it as \( N^{br}_{s_{k}} \). If there is more than one node having the lowest unit storage cost, then choose the most downstream one.

3. If \( N^{br}_{s_{k}} \) is \( N^{br}_{0} \), then the locating process ends. Otherwise, search among the nodes \( \{N^{br}_{i} \mid N^{br}_{s_{k}} \supseteq N^{br}_{i} \supseteq N^{br}_{0} \} \). Find the node with the lowest capacity. Denote it as \( N^{br}_{c_{k+1}} \). If there is more than one node having the lowest capacity, then choose the most downstream one. Let \( k = k + 1 \) and go back to step 2.

Since \( N_{\lambda} \) is the last \( s \)-node in the branch, denote \( \zeta \) as the index such that \( N_{\lambda} = N^{br}_{s_{\zeta+1}} \).

Consider the nodes among \( \{N^{br}_{i} \mid N^{br}_{n} \supseteq N^{br}_{i} \supseteq N^{br}_{c_{1}} \} \). These nodes’ capacities are all greater than or equal to that of \( N^{br}_{c_{1}} \). No matter what operation rate of \( N^{br}_{c_{1}} \) is, they can work at the same rate of \( N^{br}_{c_{1}} \). There is no need to build up products upstream of \( N^{br}_{c_{1}} \). Therefore, for the optimal operation, the operation rates of \( \{N^{br}_{i} \mid N^{br}_{n} \supseteq N^{br}_{i} \supseteq N^{br}_{c_{1}} \} \) should be the same as that of \( N^{br}_{c_{1}} \).
Consider the nodes among \( \{N_{\kappa_{br}} | N_{\kappa_{br}} \geq N_{\kappa_{br}} \geq N_{s_{k}}\} \). \( N_{\kappa_{br}} \) has the lowest capacity among these nodes. No matter what operation rate of \( N_{\kappa_{br}} \) is, any of \( \{N_{\kappa_{br}}\} \) has sufficient capacity to transfer the products from \( N_{\kappa_{br}} \) to \( N_{s_{k}} \). To achieve the minimum storage cost, no product should be left among \( \{N_{\kappa_{br}} | N_{\kappa_{br}} \geq N_{\kappa_{br}} \geq N_{s_{k}}\} \). All products should be transferred to \( N_{s_{k}} \). Therefore, the operation rates of \( \{N_{\kappa_{br}} | N_{\kappa_{br}} \geq N_{\kappa_{br}} \geq N_{s_{k}}\} \) should be the same as that of \( N_{s_{k}} \). Notice that when \( N_{s_{k}} \) is \( N_{\lambda} \), we use the unit storage cost at \( N_{s_{m}} \) as the storage cost of \( N_{\lambda} \). This is because the buildup among \( \{N_{j} | N_{\lambda} \geq N_{j} \geq N_{s_{m}}\} \) can only exist at \( N_{s_{m}} \).

Consider the nodes among \( \{N_{\kappa_{br}} | N_{s_{k}} \geq N_{\kappa_{br}} \geq N_{s_{k}} \} \). These nodes’ capacities are all greater than or equal to that of \( N_{\kappa_{br}} \). When the operation rate of \( N_{s_{k}} \) is greater than that of \( N_{s_{k}} \), then the nodes among \( \{N_{\kappa_{br}} | N_{s_{k}} \geq N_{\kappa_{br}} \geq N_{s_{k}} \} \) should work at the same rate of \( N_{s_{k}} \) to avoid transferring products from \( N_{s_{k}} \) to any other locations among these nodes which have higher unit storage cost. Therefore, to achieve minimum storage cost, these nodes should work at the same operation rate of \( N_{s_{k}} \).

With the analysis above, we show that with Assumption 15 nodes among \( \{N_{i_{br}} | N_{i_{br}} \geq N_{i_{br}} \geq N_{s_{m}}\} \) have the same operation rate under the optimal policy. With the locating \( s \) and \( c \) nodes, we can do aggregation to simplify the system.

**Locating Two Special Nodes for Aggregation: \( N_{s_{\mu}} \) and \( N_{c_{\mu+1}}^{br} \)**

Denote \( \mu \) as the smallest index in \( \{1, 2, ..., m\} \) such that there exists an index \( i \) in \( \{\zeta, \zeta - 1, ..., 1\} \) such that \( c(N_{\mu}) \geq c(N_{\kappa_{br}}) \) and \( q_{x}(N_{s_{\mu}}) \leq q_{x}(N_{s_{i}}) \). If such a \( \mu \) doesn’t exist, then let \( \mu = m \).

Denote \( \eta \) as the smallest index in \( \{\zeta - 1, \zeta - 2, ..., 0\} \) such that \( c(N_{\mu}) \geq c(N_{c_{\mu+1}}^{br}) \) and \( q_{x}(N_{s_{\eta}}) \leq q_{x}(N_{s_{\eta+1}}) \). If such an \( \eta \) doesn’t exist, then let \( \eta = \zeta \).

The locations of \( N_{\kappa_{\mu}}, N_{s_{\mu}}, N_{c_{\mu+1}}^{br} \) and \( N_{s_{\eta+1}}^{br} \) are shown in Figure 9.3.

![Figure 9.3: The locations of \( N_{\kappa_{\mu}}, N_{s_{\mu}}, N_{c_{\mu+1}}^{br} \) and \( N_{s_{\eta+1}}^{br} \).](image)

It can be proved that the storage built up among \( \{N_{i} | N_{s_{\mu}} \geq N_{i} \geq N_{s_{\eta}}\} \) can only exist at \( N_{\mu}^{a} \). For any operation rate of \( N_{c_{\mu+1}}^{br} \), the nodes among \( \{N_{i} | N_{c_{\mu+1}}^{br} \geq N_{i} \geq N_{s_{\mu}}\} \) have sufficient capacity to transfer the products from \( N_{c_{\mu+1}}^{br} \) to \( N_{s_{\mu}} \), which has the lowest storage cost among them. This can avoid unnecessary higher storage costs. Besides, compared with building up storage between \( N_{c_{\mu+1}}^{br} \) and \( N_{c_{m}} \), letting \( N_{c_{m}} \) operate at the same rate of \( N_{c_{\mu+1}}^{br} \).
makes \(N_{cm}\) ramping up operation rate earlier. This makes \(N_{cm}\) start earlier the transferring of products from \(\{N_k|N_l \gg N_k \gg N_{cm}\}\) to the nodes downstream of \(N_{cm}\), which also reduce the total storage costs. Therefore, the operations of \(\{N_i|N_{c_{m+1}}^{br} \gg N_i \gg N_{cm}\}\) will be equal. Also, we have shown that nodes among \(\{N_i|N_{s_{m}}^{br} \gg N_i \gg N_{cm}\}\) have the same operation rate. Thus, nodes among \(\{N_i|N_{s_{m}}^{br} \gg N_i \gg N_{cm}\}\) can be aggregated into one node.

### Aggregation

After we determine \(-\) and \(s\)-nodes, and indices of \(\mu\) and \(\eta\), we can aggregate the system with the following procedure:

1. Denote \(N_0^a\) as the aggregated node of \(\{N_i|N_{s_1} \gg N_i \gg N_0\}\). Then, \(c(N_0^a) = \tilde{c}(N_0) = 1\).
2. For \(1 \leq j \leq \mu - 1\), denote \(N_j^a\) as the aggregated node of \(\{N_i|N_{s_{j+1}} \gg N_i \gg N_s\}\). Then, \(q_x(N_j^a) = \tilde{q}_x(N_s)\). \(c(N_j^a) = c(N_s)\).
3. Denote both \(N_{L}^a\) and \(N_{0}^{br,a}\) as the aggregated node of \(\{N_j|N_{s_{m+1}} \gg N_j \gg N_{s_m}\}\) \(\cup\) \(\{N_{i}^{br} | N_{s_{m}}^{br} \gg N_{i}^{br} \gg N_{0}^{br}\}\). Then \(q_x(N_{L}^a) = q_x(N_{0}^{br,a}) = \tilde{q}_x(N_{s_m})\). \(c(N_{L}^a) = c(N_{0}^{br,a}) = \tilde{c}(N_{s_m})\).
4. For \(\mu + 1 \leq k \leq \mu + M - m - 1\), denote \(N_{k}^a\) as the aggregated node of \(\{N_i|N_{s_{k-\mu+m+1}} \gg N_i \gg N_{s_{k-\mu+m}}\}\). Then \(q_x(N_{k}^a) = \tilde{q}_x(N_{s_{k-\mu+m}})\). \(c(N_{k}^a) = \tilde{c}(N_{s_{k-\mu+m}})\).
5. Denote \(N_{L}^s\) \((\text{where} \ L = \mu + M - m, \text{as the aggregated node of} \ \{N_i|N_l \gg N_i \gg N_{s_m}\}\) Then, \(q_x(N_{L}^s) = q_x(N_{s_m})\). \(c(N_{L}^s) = c(N_l)\).
6. For \(1 \leq i < \eta\), denote \(N_{i}^{br,a}\) as the aggregated node of \(\{N_{j}^{br} | N_{s_{i-1}}^{br} \gg N_{j}^{br} \gg N_{s_{i-1}}^{br}\}\). Then \(q_x(N_{i}^{br,a}) = \tilde{q}_x(N_{s_{i-1}+1})\). \(c(N_{i}^{br,a}) = \tilde{c}(N_{s_{i-1}+1})\).
7. Denote \(N_{\eta}^{br,a}\) as the aggregated node of \(\{N_{j}^{br} | N_{s_{\eta}}^{br} \gg N_{j}^{br} \gg N_{s_{1}}^{br}\}\). Then \(q_x(N_{\eta}^{br,a}) = \tilde{q}_x(N_{s_{1}})\). \(c(N_{\eta}^{br,a}) = \tilde{c}(N_{s_{1}})\).

With the \(s\)- and \(-\) nodes and aggregation, we can transform the assembly tree network system consisting of one disrupted chain and one branch into the following system in Figure 9.4.

After aggregation, as shown in Figure 9.4, Chain 1 consists of nodes \(\{N_0^a, N_1^a, ..., N_\mu^a\}\). Chain 2 consists of nodes \(\{N_0^{br,a}, N_1^{br,a}, ..., N_L^a\}\). The branch consists of nodes \(\{N_0^{br,a}, N_1^{br,a}, ..., N_\eta^{br,a}\}\).

By aggregation, under Assumption 15, an assembly tree network system consisting of one disrupted chain and one branch is turned into the aggregated system, which satisfy the following properties (Properties of Aggregated Special Tree under Assumption 15 PASTUA):

**PASTUA.** \(\{N_L^a, N_{L-1}^a, ..., N_0^a, N_\eta^{br,a}, N_\eta^{br,a}, ..., N_1^{br,a}\}\) is a normalized assembly tree system, with the root node \(N_0^a\). \(\{N_0^a, N_{L-1}^a, ..., N_0^a\}\) is a disrupted chain, in which \(N_L^a\) is the disrupted node. \(\{N_\eta^{br,a}, N_\eta^{br,a}, ..., N_1^{br,a}\}\) is a branch chain, fed into node \(N_\mu^a\), which is also denoted as \(N_0^{br,a}\).
Figure 9.4: The aggregation of the system

PASTUA2. The capacities and unit storage costs are decreasing downstream along the whole disrupted chain (Chain 2 and Chain 1). That is, $q_x(N^a_k) > q_x(N^a_j)$ and $c(N^a_k) > c(N^a_j)$, for any indices $k$ and $j$ such that $L \geq k > j \geq 0$.

PASTUA3. In the branch, both the capacities and storage costs are increasing downstream. That is, $q_x(N^{br,a}_i) < q_x(N^{br,a}_j)$ and $q_x(N^{br,a}_i) < q_x(N^{br,a}_j)$, for any indices $i$ and $j$ such that $\eta \geq i > j \geq 0$.

PASTUA4. There doesn’t exist a pair of indices $i$ and $j$ such that $i \in \{1, 2, ..., \eta\}$ and $j \in \{0, 1, ..., \mu\}$ satisfying two inequalities, $c(N^a_j) \leq c(N^{br,a}_i)$ and $q_x(N^a_j) \leq q_x(N^{br,a}_i)$, at the same time.

9.3.2 The Operations of the Aggregated System

First, we define all the time variables as follows:

1. The ending times of the buildup of nodes in Chain 2:
   For $k \in \{\mu + 1, \mu + 2, ..., L\}$, denote $t_k$ as the time point when product$_k^a$ is consumed to 0 within the disruption period. That is, $x^a_k(t) = 0$ for any $t$ in $[t_k, t_{dis} + \tau_{dis})$. And there exists a $t' \in [t_{dis}, t_k)$ such that $x^a_k(t) > 0$ for any $t$ in $[t', t_k)$. If product$_k^a$ is never built up, then define $t_k := t_{dis}$. Define $t_{L+1} := t_{dis}$.

2. The starting times of the buildup of nodes in Chain 2:
   For $k \in \{\mu + 1, \mu + 2, ..., L\}$, denote $t_{bx,k}$ as the time point when the system starts building up product$_k^a$ within the pre-disruption period. That is, $x^a_k(t) = 0$ for any $t$ in $[t_{dis} - \tau_f, t_{bx,k})$, and there exists a $t' \in (t_{bx,k}, t_{dis}]$ such that $x^a_k(t) > 0$ for any $t$ in $[t_k, t_{bx,k})$. If product$_k^a$ is never built up in the pre-disruption period, then define $t_{bx,k} := t_{dis}$. Define $t_{bx,L+1} := t_{dis}$.

3. The ending times of the buildup of nodes in Chain 1:
   For $j \in \{1, 2, ..., \mu\}$, denote $t_j$ as the time point when product$_j^a$ is consumed to 0. That is, $x^a_j(t) = 0$ for any $t$ in $[t_j, t_{dis} + \tau_{dis})$. And there exists a $t' < t_j$ such that $x^a_j(t) > 0$ for any $t$ in $[t', t_j)$. If product$_j^a$ is never built up, then define $t_j := t_{bx,\mu+1}$.
4. The starting times of the buildup of nodes in Chain 1:

For \( j \in \{1, 2, ..., \mu\} \), denote \( t_{bx,j} \) as the time point when the system starts building up product \( a_j \) within the pre-disruption period. That is, \( x_j^{{a_j}}(t) = 0 \) for any \( t \in [t_{dis} - \tau_j, t_{bx,j}] \), and there exists a \( t' \in (t_{bx,j}, t_{dis}] \) such that \( x_j^{{a_j}}(\hat{t}) > 0 \) for any \( t \in [t_{bx,j}, t') \).

If product \( a_j \) is never built up in the pre-disruption period, then define \( t_{bx,j} := t_{bx,\mu + 1} \).

5. The ending times of the buildup of nodes in the branch:

For any \( i \in \{1, 2, ..., \eta\} \), denote \( t_{br,i}^b \) as the time point when product \( b_i \) is consumed to 0. That is, \( x_i^{br,a}(t) = 0 \) for any \( t \in [t_{dis} - \tau_j, t_{bx,i}] \), and there exists a \( t' < t_{br,i}^b \) such that \( x_i^{br,a}(\hat{t}) > 0 \) for any \( t \in [t', t_{br,i}^b] \). If product \( b_i \) is never built up, then define \( t_{br,i}^b := t_{bx,\mu + 1} \).

6. The starting times of the buildup of nodes in the branch:

Denote \( t_{bx,i}^a \) as the time point when the system starts building up product \( a_i \) within the pre-disruption period. That is, \( x_i^{br,a}(t) = 0 \) for any \( t \in [t_{dis} - \tau_j, t_{bx,i}] \), and there exists a \( t' \in (t_{bx,i}^a, t_{dis}] \) such that \( x_i^{br,a}(\hat{t}) > 0 \) for any \( t \in [t_{bx,i}^a, t') \). If product \( a_i \) is never built up in the pre-disruption period and disruption periods, then define \( t_{bx,i}^a := t_{bx,\mu + 1} \).

For \( j \in \{1, 2, ..., L\} \) and \( i \in \{0, 1, ..., \eta\} \), we denote

\[
Y_j^{a}(t) = \int_{t_{dis} - \tau_j}^{t} [u_j^{a}(t') - \tilde{d}]dt', \quad \text{and} \quad Y_i^{br,a}(t) = \int_{t_{dis} - \tau_j}^{t} [u_i^{br,a}(t') - \tilde{d}]dt'.
\]

Lemma 23. For an aggregated system satisfying PASTUA, for \( L \geq k \geq \mu + 1 \),

\[
t_{bx,k} \leq t_{bx,k+1}.
\] (9.1)

Notice that \( t_{bx,L+1} = t_{dis} \).

Proof. Consider the index \( k \) such that \( L - 1 \geq k \geq \mu + 1 \). The unit storage cost at \( N_k^a \) is lower than at \( N_{k+1}^a \). If products are built up at \( N_k^a \), then \( N_k^a \) will transfer them downstream at the highest rate to achieve lower storage cost.

If the operation rate of \( N_{k+1}^a \) is lower than \( N_k^a \)’s capacity, then \( N_k^a \) is able to transfer downstream all the products at \( N_{k+1}^a \), there will be no buildup at \( N_{k+1}^a \) until the operation rate of \( N_{k+1}^a \) is higher than \( N_k^a \)’s capacity.

When the operation rate of \( N_{k+1}^a \) is higher than \( N_k^a \)’s capacity, then \( N_k^a \) should work at its capacity to transfer as much as possible. Since the rate of capacity at \( N_k^a \) is higher than the rate of \( N_{k-1}^a \), then buildup appears at \( N_k^a \) at the same time when buildup starts at \( N_{k+1}^a \).

Therefore, the buildup starting time at \( N_k^a \) cannot be later than the buildup starting time at \( N_{k+1}^a \).

Also, when \( N_k^a \) works at its capacity, product \( a_k \) is built up. Then, all the nodes among \( \{N_{k-1}^a, N_{k-2}^a, ..., N_{\mu + 1}^a\} \) should all work at its capacity to transfer products downstream to achieve lower storage cost. Therefore, if product \( a_{k+1} \) is built up, any product among \( \{product_{k+1}^a, product_{k+1}^{b,a}, ..., product_{\mu+1}^{b,a}\} \) must be built up under the optimal operation.
Thus, under optimal operation, it is satisfied that

$$\{\text{If } N_{\text{at time } t} \text{ satisfies PASTUA, denote } u \text{ as high as possible to achieve the amount. Thus, the operation rate should be the capacity.}$$

$$\text{Consider the period } \tau_k \text{ when } x_{k,L} = 0 \text{ after } t_{\text{dis}}. \text{ Then, } t_{\text{dis}} > t_{\text{bx},L} \geq t_{\text{bx},L-1} \geq \ldots \geq t_{\text{bx},\mu+1}.$$  

Thus, under optimal operation, it is satisfied that $t_{\text{bx},k} \leq t_{\text{bx},k+1}$.  

From the proof of Lemma 23, we can get Corollary 4.

**Corollary 4.** For an aggregated system satisfying PASTUA, for $L - 1 \geq k \geq \mu + 1$, when the products of $N_{k+1}^a$ are built up, the operation rate of $N_k^a$ should be higher than the capacity of $N_k^a$.

**Lemma 24.** For an aggregated system satisfying PASTUA, for $L \geq k \geq \mu + 1$, the optimal operation rate of $N_k^a$ satisfies:

$$u_k^a(t) = \begin{cases} 
    u_{k-1}^a(t), & \text{when } t \in [t_{\text{dis}} - \tau_f, t_{\text{bx},k}), \\
    c(N_k^a), & \text{when } t \in [t_{\text{bx},k}, t_{k+1}), \\
    u_{k+1}^a(t), & \text{when } t \in [t_{k+1}, t_{\text{dis}} + \tau_{\text{dis}}] 
\end{cases}$$  \quad (9.2)

Notice that $u_{L+1}^a(t) := 0$, and $t_{L+1} := t_{\text{dis}}$.

**Proof.** During the period $[t_{\text{dis}} - \tau_f, t_{\text{bx},k})$ , $N_k^a$ has not started building up any products. Therefore, the operation rate of $N_k^a$ equals to its downstream node $N_{k-1}^a$. That is, $u_k^a(t) = u_{k-1}^a(t)$ when $t \in [t_{\text{dis}} - \tau_f, t_{\text{bx},k})$.

Consider the period $[t_{\text{bx},k}, t_{\text{bx},k+1})$, and notice that $t_{\text{bx},L+1} = t_{\text{dis}}$. $N_k^a$ starts buildup and all the nodes upstream of $N_k^a$ haven’t started buildup. The amount of buildup of product $k$ at $t_{\text{bx},k+1}$ is $x_{k,L}^a(t_{\text{bx},k+1})$. No matter what exact value of $x_{k,L}^a(t_{\text{bx},k+1})$ is, the optimal policy of the buildup is to start as late as possible to shorten the storage period, so that the total storage cost can be kept as low as possible. Therefore, the operation rate of $N_k^a$ should be as high as possible to achieve the amount. Thus, the operation rate should be the capacity. That is, $u_k^a(t) = c(N_k^a)$ when $t \in [t_{\text{bx},k}, t_{\text{bx},k+1})$.

During the period $[t_{\text{bx},k+1}, t_{k+1})$, products are built up at $N_{k+1}^a$. Since the unit storage cost at $N_{k+1}^a$ is higher than at $N_k^a$, node $N_k^a$ should work at highest rate to transfer the products downstream. Corollary 7 shows that $N_k^a$ can work at its capacity. Therefore, to transfer as much as possible, $N_k^a$ should work at its capacity. That is, $u_k^a(t) = c(N_k^a)$ when $t \in [t_{\text{bx},k+1}, t_{k+1})$.

During the period $[t_{k+1}, t_{\text{dis}} + \tau_{\text{dis}})$, the buildup at $N_{k+1}^a$ is 0. Therefore, the operation rates of $N_{k+1}^a$ and $N_k^a$ should be equal. That is, $u_k^a(t) = u_{k+1}^a(t)$, when $t \in [t_{k+1}, t_{\text{dis}} + \tau_{\text{dis}})$.  

**Lemma 25.** For an aggregated system satisfying PASTUA, denote $i$ as the largest index in $\{1, 2, \ldots, \eta\}$ such that any product of $\{\text{product}_{1, br, a}, \text{product}_{2, br, a}, \ldots, \text{product}_{i, br, a}\}$ has no buildup at time $t_{\text{bx},\mu+1}$, then any product of $\{\text{product}_{1, br, a}, \text{product}_{2, br, a}, \ldots, \text{product}_{i, br, a}\}$ will never be built up after $t_{\text{bx},\mu+1}$ under the optimal operation.
Proof. At time $t_{bx,µ+1}$, node $N^a_µ$ needs to work at its highest capacity to transfer product$^a_{µ+1}$ downstream. Since $i$ is the largest index such that any product of $\{product^a_i, product^a_{i-1}, ..., product^a_1\}$ has no buildup at time $t_{bx,µ+1}$. The maximum rate of $N^a_µ$ can not exceed the capacity of $N^a_i$. Therefore, for the optimal operation, nodes among $\{N^a_i, N^a_{i-1}, ..., N^a_1, N^a_µ\}$ should work at the same rate of the capacity of $N^a_i$ to transfer as much product$^a_{µ+1}$ as possible, and consume as much buildup in the branch as possible. Thus, $\{N^br,a_i, N^br,a_{i-1}, ..., N^br,a_1\}$ can not be built up as long as product$^a_{µ+1}$ is not used up.

Based on Lemma 24 product$^a_{µ+1}$ will not be used up until $t_{µ+1}$. Then $N^a_µ$ will work at 0 rate due to 0 input from $N^a_µ$, and there is no need for any buildup in the branch. Therefore, for the optimal operation, $\{N^br,a_i, N^br,a_{i-1}, ..., N^br,a_1\}$ should not be built up after $t_{µ+1}$.

Thus, the statement of Lemma 25 is proved.

From Lemma 25 we can get Corollary 5.

**Corollary 5.** For an aggregated system satisfying PASTUA, for any $i \in \{1, 2, ..., η\}$, the optimal operation satisfies:

$$t^br_{bx,i} ≤ t_{bx,µ+1}.$$  

**Lemma 26.** For an aggregated system satisfying PASTUA, for $µ - 1 ≥ j ≥ 1$, the optimal operation satisfies:

$$t_{bx,j} ≤ t_{bx,j+1} ≤ t_{bx,µ+1}. \quad (9.3)$$

**Proof.** Similar as the analysis of Lemma 23, we can easily show that product$^a_{j+1}$ can be built up only when the operation rate of $N^a_{j+1}$ is greater than the capacity of $N^a_j$. Also, if product$^a_{j+1}$ starts buildup at $t_{bx,j+1}$, then any of $\{product^a_j, product^a_{j-1}, ..., product^a_1\}$ is built up no later than $t_{bx,j+1}$.

Denote $j_1$ as the smallest index such that product$^a_{j_1}$ hasn’t started buildup before $t_{bx,µ+1}$. Therefore, any of $\{product^a_i, product^a_{i-1}, ..., product^a_1\}$ hasn’t started buildup before $t_{bx,µ+1}$. Denote $i$ as the largest index such that any product of $\{product^br,a_i, product^br,a_{i-1}, ..., product^br,a_1\}$ has no buildup at $t_{bx,µ+1}$.

Then, at time $t_{bx,µ+1}$, node $N^a_µ$ should work at its highest rate to transfer products from $N^a_µ$ + 1 downstream and consuming the buildup in the branch. Since $\{product^br,a_i, product^br,a_{i-1}, ..., product^br,a_1\}$ has no buildup at $t_{bx,µ+1}$ at $t_{bx,µ+1}$, nodes among $\{N^br,a_i, N^br,a_{i-1}, ..., N^br,a_1, N^br,a_µ\}$ should work at the same rate of the capacity of $N^br,a_i$.

If the capacity of $N^br,a_i$ is lower than or equal to the the capacity of $N^a_{j_1-1}$, then any of $\{N^a_µ, N^a_{µ-1}, ..., N^a_{j_1}\}$ will never build up products since $t_{bx,µ+1}$. Therefore, based on definition, $t_{bx,µ+1} = t_{bx,µ-1} = ... = t_{bx,j_1} ≥ t_{bx,j_1-1} ≥ ... ≥ t_{bx,1}$.

If the capacity of $N^br,a_i$ is greater than the the capacity of $N^a_{j_1-1}$, then denote $j_2$ as the smallest index such that the capacity of $N^br,a_i$ is less than or equal to the capacity of $N^a_{j_2}$. Therefore, nodes among $\{N^a_µ, N^a_{µ-1}, ..., N^a_{j_2}\}$ work at the same rate of $N^br,a_i$ to transfer products from $N^a_{µ+1}$ downstream, and nodes downstream of $N^a_{j_2}$ all work at their capacity to transfer products downstream. Thus, nodes among $\{N^a_µ, N^a_{µ-1}, ..., N^a_{j_2+1}\}$ never build up products.
Thus, the statement of Lemma 26 is proved.

From the proof of Lemma 26 we can get the Corollary 6.

**Corollary 6.** For an aggregated system satisfying PASTUA, for $\mu - 1 \geq j \geq 1$, when the products of $N_{j+1}$ are built up, the operation rate of $N_{j+1}$ should be higher than the capacity of $N_j$.

**Lemma 27.** For an aggregated system satisfying PASTUA, for any $i \in \{2, 3, \ldots, \eta\}$, the optimal operation satisfies:

$$t_{i-1}^{br} \leq t_i^{br}. \quad (9.4)$$

**Proof.** We prove this lemma by contradiction. Denote $u^*$ as the optimal operation such that there exists an $i \in \{2, 3, \ldots, \eta\}$ which satisfies $t_i^{br,*} < t_i^{br}$. That is, product $i$ is consumed to 0 later than product $i$. Then, there exists a $t' \in [t_i^{br,*}, t_i^{br})$ such that $x_i(t) = 0$ and $x_i(t') = 0$ for any $t \in [t', t_i^{br})$.

During the interval $[t', t_i^{br}]$, the operation rates of $N_i^{br,a}$ and $N_{i-1}^{br,a}$ are the same. Any products produced by $N_i^{br,a}$ is transferred immediately to $N_{i-1}^{br,a}$.

Then, we can simply lower down the operation of $N_{i-1}^{br,a}$ to let products stay at $N_{i-1}^{br,a}$ for a longer time. We denote the alternative operation as $u^{**}$. During the interval $[t', t_i^{br}]$, we first let $u_i^{br,*,*}(t) < u_i^{br,*}(t)$, so that less storage is transferred from $N_i^{br,a}$ to $N_{i-1}^{br,a}$. And after a certain period of time, we ramp up $u_i^{br,*,*}(t)$, so that the products at $N_i^{br,a}$ are still consumed to 0 by $t_i^{br,*}$. With $u^{**}$, a part of storage are kept at $N_i^{br,a}$ for a longer time, instead of being transferred immediately to $N_{i-1}^{br,a}$.

Since the unit storage cost at $N_{i-1}^{br,a}$ is higher than at $N_i^{br,a}$, the alternative operation $u^{**}$ can achieve less total storage cost than $u^*$, due to the delay of the transfer of products from $N_i^{br,a}$ to $N_{i-1}^{br,a}$. Thus, $u^*$ can not be optimal, which is a contradiction.

Therefore, the optimal operation satisfies $t_{i-1}^{br} \leq t_i^{br}$. \qed

**Lemma 28.** For an aggregated system satisfying PASTUA, denote $i$ as the largest index in $\{1, 2, \ldots, \eta\}$ such that any of $\{product_1^{br,a}, product_2^{br,a}, \ldots, product_i^{br,a}\}$ has no buildup at time $t_{bx,\mu+1}$, then the optimal operation satisfies:

$$t_{bx,\mu+1} = t_i^{br}. \quad (9.5)$$

If such an $i$ doesn’t exist, then

$$t_{bx,\mu+1} \leq t_1^{br}. \quad (9.5)$$

**Proof.** If $i$ doesn’t exist, it means product $1^{br,a}$ has buildup at time $t_{bx,\mu+1}$, based on Lemma 27. Then, product $1^{br,a}$ can be used up only after $t_{bx,\mu+1}$, i.e., $t_{bx,\mu+1} \leq t_1^{br}$. 

123
If $i$ exists, then product $x_i^{br,a}$ is consumed to 0 at time $t_i^{br}$. Based on Lemma 27, there is no buildup among $\{N_i^{br,a}, N_{i-1}^{br,a}, ..., N_1^{br,a}\}$ during the interval $[t_i^{br}, t_{bx,i+1}]$. Then, $\{N_i^{br,a}, N_{i-1}^{br,a}, ..., N_1^{br,a}, N_{\mu}^{a}\}$ has the same operation rate during this interval. Based on Lemma 25, $t_i^{br} \leq t_{bx,i+1}$, since product $x_i^{br,a}$ is never built up since $t_{bx,i+1}$. Now we need to prove that $t_i^{br}$ cannot be less than $t_{bx,i+1}$. We prove this by contradiction. Denote the optimal operation as $u^*$ under which $t_i^{br} < t_{bx,i+1}$.

Denote $j$ as the largest index in $\{1, 2, ..., \mu\}$ such that any node among $\{N_{\mu}^{a}, N_{\mu-1}^{a}, ..., N_{j+1}^{a}\}$ has no buildup right before $t_{bx,i+1}^*$. Thus, there exists a smallest $t' \in [t_i^{br}, t_{bx,i+1}]$ such that $x_j^{a,s}(t) > x_j^{a,s}(t') = ... = x_j^{a,s}(t) = 0$ for any $t \in [t', t_{bx,i+1}^*]$. Therefore, nodes among $\{N_{\mu}^{a}, N_{\mu-1}^{a}, ..., N_{j+1}^{a}\}$ have the same operation rate during $[t', t_{bx,i+1}]$.

Proof of Lemma 28: Case 1: If $q_x(N_{i+1}^{a}) > q_x(N_i^{br,a})$

During the interval $[t', t_{bx,i+1}]$, nodes among $\{N_i^{br,a}, N_{i-1}^{br,a}, ..., N_1^{br,a}\} \cup \{N_{\mu}^{a}, N_{\mu-1}^{a}, ..., N_{j+1}^{a}\}$ all have the same operation rate. They transfer the products from $N_i^{br,a}$ to $N_{j+1}^{a}$. We need to discuss two subcases, whether or not the operation rate of these nodes smaller than the capacity of $N_{j+1}^{a}$.

Case 1.1: If $u_j^{a,s}(t) < c(N_{j+1}^{a})$ for some $t \in [t', t_{bx,i+1}]$

We can select an alternative $u^{**}$ to delay the transferring from $N_i^{br,a}$ to $N_{j+1}^{a}$. The alternative operation shifts a part of the buildup lines of $\{N_{i-1}^{br,a}, N_{i-2}^{br,a}, ..., N_1^{br,a}, N_{\mu}^{a}, N_{\mu-1}^{a}, ..., N_{j+1}^{a}\}$ in the buildup figure. $u^{**}$ satisfies:

\[ Y_{j'}^{a,s}(t_{bx,i+1}^*) = Y_{j'}^{a,s}(t_i^{br} + t_{bx,i+1}^*), \ \forall j' \in \{\mu, \mu-1, ..., j\}. \]

\[ Y_{j'}^{br,a,**}(t_{bx,i+1}^*) = Y_{j'}^{br,a,**}(t_i^{br} + t_{bx,i+1}^*), \ \forall i' \in \{i-1, i-2, ..., 1\}. \]

\[ u_{j'}^{a,**}(t) = u_{j'}^{a,s}(t - t_{bx,i+1}^* + t'), \ \forall j' \in \{\mu, \mu-1, ..., j\} \text{ and } t' \text{ such that } Y_{j'}^{a,**}(t) > Y_{j-1}^{a,s}(t). \]

\[ u_{i'}^{br,a,**}(t) = u_{i'}^{br,a,s}(t - t_{bx,i+1}^* + t'), \ \forall i' \in \{i-1, i-2, ..., 1\} \text{ and } t' \text{ such that } Y_{i'}^{br,a,**}(t) > Y_{i-1}^{a,s}(t). \]

$u^{**}$ shifts the buildup operations among $\{N_{i-1}^{br,a}, N_{i-2}^{br,a}, ..., N_1^{br,a}, N_{\mu}^{a}, N_{\mu-1}^{a}, ..., N_{j+1}^{a}\}$ from $t - t_{bx,i+1}^* + t'$ to a later time $t$. Since the unit storage cost at $N_{j+1}^{a}$ is greater than $N_i^{br,a}$, $u^{**}$ can achieve less total storage cost than $u^*$, because the storage are kept at $N_i^{br,a}$ for longer time instead of transferred to $N_{j+1}^{a}$ immediately. Thus, $u^*$ can not be optimal, which is a contradiction.

Case 1.2: If $u_j^{a,s}(t) = c(N_{j+1}^{a})$ for any $t \in [t', t_{bx,i+1}]$

Case 1.2.1: If $t' > t_{bx,i}^{br}$ In this case, there exists a largest index $j_1 \in \{\mu, \mu-1, ..., j\}$ and $t'' \in [t_{bx,i}, t')$ such that $x_{j_1}^{a,s}(t) > 0$ and $u_{j_1}^{a,s}(t) < c(N_{j_1}^{a}) \ \forall t \in [t'', t')$. During the period $[t'', t')$, nodes $\{N_{i}^{br,a}, N_{i-1}^{br,a}, ..., N_1^{br,a}, N_{\mu}^{a}, N_{\mu-1}^{a}, ..., N_{j_1}^{a}\}$ all have the same operation rate, which is smaller than $c(N_{j_1}^{a})$. 

124
Then, we can select an alternative operation \( u^{**} \) to delay the transferring from \( N_{i_j}^{br,a} \) to \( N_{i_j+1}^{br,a} \). During the interval \([t'', t')\), we first lower down the operation of \( \{N_{i-1}^{br,a}, N_{i-2}^{br,a}, \ldots, N_{i-1}^{br,a}, N_{i}^{br,a}, N_{i+1}^{br,a}, \ldots, N_{j}^{br,a}\} \). We let \( u_{i}^{br,a,s}(t) > u_{i-1}^{br,a,**}(t) = \ldots = u_{j-1}^{br,a,**}(t) = u_{j}^{br,a,**}(t) = \ldots = u_{j+1}^{br,a,**}(t) \). Therefore, fewer products are transferred from \( N_{i_j}^{br,a} \) to \( N_{i_j+1}^{br,a} \). And after a certain period of time, we ramp up the operation of \( \{N_{i_j+1}^{br,a}, N_{i_j+2}^{br,a}, \ldots, N_{i-1}^{br,a}, N_{i}^{br,a}, N_{i+1}^{br,a}, \ldots, N_{j}^{br,a}\} \), so that the products at \( N_{i_j}^{br,a} \) are still consumed to 0 by \( t' \) and storage at \( N_{i_j}^{a} \) still reaches the same amount as under \( u^* \) by \( t' \).

Since the unit storage cost at \( N_{i_j}^{a} \) is greater than \( N_{i_j+1}^{br,a} \), then \( u^{**} \) can achieve less total storage cost than \( u^* \), because the storage are kept at \( N_{j+1}^{br,a} \) for longer time instead of transferred to \( N_{i_j+1}^{a} \) immediately. Thus, \( u^* \) can not be optimal, which is a contradiction.

**Case 1.2.2:** If \( t' = t_{bx,i}' \) In this case, there exists a \( t'' \in \{t_{bx,i}^l, t'\} \) such that \( x_{i}^{br,a,*}(t) > 0 \) and \( u_{i}^{br,a,s}(t) < c(N_{j}^{a}) \forall t \in \{t'', t'\} \).

Then we can select an alternative operation \( u^{**} \) to delay the transferring from \( N_{i_j}^{br,a} \) to \( N_{i_j+1}^{br,a} \). During the interval \([t'', t')\), we first lower down the operation of \( N_{i_j}^{br,a} \). We let \( u_{i}^{br,a,s}(t) > u_{i-1}^{br,a,**}(t) \). Therefore, fewer products are transferred from \( N_{i+1}^{br,a} \) to \( N_{i_j}^{a} \). And after a certain period of time, we ramp up the operation of \( N_{i_j}^{br,a} \) so that the storages at \( N_{i_j}^{br,a} \) and \( N_{i+1}^{br,a} \) still respectively reaches the original amounts under \( u^* \) by \( t' \).

Since the unit storage cost at \( N_{i_j}^{br,a} \) is greater than \( N_{i+1}^{br,a} \), then \( u^{**} \) can achieve less total storage cost than \( u^* \), because the storage are kept at \( N_{i+1}^{br,a} \) for longer time instead of transferred to \( N_{i_j}^{br,a} \) immediately. Thus, \( u^* \) can not be optimal, which is a contradiction.

**Proof of Lemma 28** Case 2: If \( q_x(N_{j}^{a}) \leq q_x(N_{j}^{br,a}) \)

Denote \( j_2 \) as the largest index in \( \{\mu, \mu - 1, \ldots, j\} \) such that \( q_x(N_{j_2}^{a}) \leq q_x(N_{j}^{br,a}) \).

The unit storage cost of \( N_{j_2}^{a} \) is the smallest among the nodes \( \{N_{i_j}^{br,a}, N_{i_j+1}^{br,a}, \ldots, N_{i}^{br,a}, N_{\mu}, \ldots, N_{j}^{a}\} \). Based on the definition of \( \mu \) and \( j \), we know the capacity of \( N_{j_2}^{a} \) is less than \( N_{i_j}^{br,a} \).

Therefore, the capacity of any of \( \{N_{i_j}^{br,a}, N_{i-1}^{br,a}, \ldots, N_{i}^{br,a}, N_{\mu}, N_{\mu - 1}, \ldots, N_{j_2}^{a}\} \) is greater than the capacity of \( N_{j_2}^{a} \).

Thus, for the optimal operation \( u^* \), the product of \( N_{i_j}^{br,a} \) can be built up when the operation rate of \( N_{i_j}^{br,a} \) is greater than the capacity of \( N_{j_2}^{a} \). At \( t_{bx,i}^l \), the operation of \( N_{j_2}^{a} \) should equal to its capacity, so that build up among \( \{N_{i_j}^{br,a}, N_{i-1}^{br,a}, \ldots, N_{i}^{br,a}, N_{\mu}, N_{\mu - 1}, \ldots, N_{j_2}^{a}\} \) can be transferred to \( N_{j_2}^{a} \) as much as possible.

Considering \( Y_{i_j}^{br,a,*}(t_{bx,i}^l) = Y_{i_j}^{a,*}(t_{bx,i}^l) \) and \( Y_{i_j}^{br,a,*}(t') = Y_{i_j}^{a,*}(t') \), since \( N_{i_j}^{br,a} \) has the lowest unit storage cost among \( \{N_{i_j}^{br,a}, N_{i-1}^{br,a}, \ldots, N_{i}^{br,a}, N_{\mu}, N_{\mu - 1}, \ldots, N_{j_2}^{a}\} \) and nodes among \( \{N_{i-1}^{br,a}, \ldots, N_{i_j}^{br,a}, N_{\mu}, N_{\mu - 1}, \ldots, N_{j_2}^{a}\} \) all have higher capacity than \( N_{j_2}^{a} \), then nodes among \( \{N_{i-1}^{br,a}, \ldots, N_{i_j}^{br,a}, N_{\mu}, \ldots, N_{j_2}^{a}\} \) should have the same operation rate of \( N_{j_2}^{a} \) during \([t_{bx,i}^l, t']\) to keep products stay at \( N_{j_2}^{br,a} \) instead of being transferred to locations among \( \{N_{i_j}^{br,a}, N_{i-1}^{br,a}, N_{\mu}, \ldots, N_{j_2}^{a}\} \).

Since the unit storage cost of \( N_{i_j}^{br,a} \) is smaller than \( N_{j_2}^{br,a} \), the optimal operation of \( N_{i_j}^{br,a} \) should be equal to the operation of \( N_{j_2}^{a} \) during \([t_{bx,i}^l, t']\), so that the products are kept stay at
During the period $t_i$ instead of being transferred to $N_i$. Therefore, $N_i$ does not build up any products during $\left[t_{br,i}^{br,*}, t'\right)$, which is a contradiction.

Therefore, since the case of $t_i < t_{br,\mu+1}$ can not be optimal, it should be satisfied that $t_{br,\mu+1} = t_i$.

\begin{lemma}
For an aggregated system satisfying PASTUA, for any $i$ in $\{1, 2, ..., \eta - 1\}$, the optimal operation rate of $N_i$ satisfies:

$$u_i^{br,a}(t) = \begin{cases} u_{i-1}(t), & \text{when } t \in [t_{dis} - \tau_f, t_{br,i}], \\ c(N_i), & \text{when } t \in [t_{br,i}, t_{br,i+1}], \\ u_{i+1}(t), & \text{when } t \in [t_{br,i+1}, t_{dis} + \tau_{dis}], \end{cases}$$

\end{lemma}

\begin{proof}
Before $t_{br,i+1}$, there is no buildup at $N_{i+1}$. Therefore, the operation rate of $N_i$ should equal to the rate of its downstream node $N_{i+1}$. That is, $u_i(t) = u_{i+1}(t)$, when $t \in [t_{dis} - \tau_f, t_{br,i}]$.

During the period $[t_{br,i}, t_{br,i+1})$, based on Lemma 27, no buildup exists among $\{N_i, N_{i-1}, ..., N_{\mu+1}\}$. Also, based on Lemma 28, $N_{\mu+1}$ starts buildup no later than $t_{br}$. Therefore, $N_{\mu+1}$ should work at its highest rate to transfer products at $N_{\mu+1}$. The highest rate of $N_{\mu+1}$ during this period is the capacity of $N_i$. Therefore, the optimal operation rate of $N_i$ should also work at its capacity. That is, $u_i(t) = c(N_i)$, where $t \in [t_{br,i}, t_{br,i+1})$.

During the period $[t_{br,i}, t_{br,i+1})$, the products of $N_{i+1}^{br,a}$ is built up and then used up. Since the unit storage cost at $N_i^{br,a}$ is greater than at $N_{i+1}^{br,a}$, then $N_i^{br,a}$ will try to avoid transferring buildup from $N_{i+1}^{br,a}$ to $N_i^{br,a}$. Starting buildup at $N_i^{br,a}$ as late as possible can achieve lower total storage cost. Therefore, to shorten the interval of building up products, $N_i^{br,a}$ should work at its capacity during the interval. That is, $u_i(t) = c(N_i)$, where $t \in [t_{br,i}, t_{br,i+1})$.

After $t_{br,i+1}$, the operation rate of $N_i^{br,a}$ should equal to the rate of $N_{i+1}^{br,a}$, since the storage at $N_{i+1}^{br,a}$ is 0. That is, $u_i(t) = u_{i+1}(t)$, when $t \in [t_{br,i+1}, t_{dis} + \tau_{dis}]$.

\end{proof}

\begin{lemma}
For an aggregated system satisfying PASTUA, the optimal operation satisfies:

$$t_{br,\eta} \leq t_{\mu+1}.$$

\end{lemma}

\begin{proof}
Based on Lemma 24, at $t_{\mu+1}$, any buildup in Chain 2 is used up. Therefore, $N_{\mu+1}^{a}$ should work at 0 rate since $t_{\mu+1}$ will be useless and unnecessary.

Based on Lemma 27, at $t_{br,\eta}$, any buildup in the branch is used up. Therefore, $t_{\eta}^{br}$ should be no later than $t_{\mu+1}$. \hfill \square

\end{proof}

\begin{lemma}
For an aggregated system satisfying PASTUA, the optimal operation of $N_{\eta}^{br,a}$ satisfies:

$$u_{\eta}^{br,a}(t) = \begin{cases} u_{\eta-1}(t), & \text{when } t \in [t_{dis} - \tau_f, t_{br,\eta}], \\ c(N_{\eta}), & \text{when } t \in [t_{br,\eta}, t_{\mu+1}], \\ 0, & \text{when } t \in [t_{\mu+1}, t_{dis} + \tau_{dis}], \end{cases}$$

\end{lemma}
Proof. Before \( t_{br,\eta}^b \), there is no buildup at \( N_{\eta}^{br,a} \). Therefore, the operation rate of \( N_{\eta}^{br,a} \) should equal to the rate of \( N_{\eta-1}^{br,a} \). That is, \( u_{\eta}^{br,a}(t) = u_{\eta-1}^{br,a}(t) \), when \( t \in [t_{dis} - \tau_f, t_{br,\eta}^b] \).

During the period \([t_{br,\eta}^b, t_{\mu+1}]\), based on Lemma 27, no buildup exists in the branch. Also, based on Lemma 28, \( N_\mu^{a} \) starts buildup no later than \( t_{br,\eta}^b \). Therefore, \( N_\mu^{a} \) should work at its highest rate to transfer products at \( N_{\mu-1}^{a} \) downstream. The highest rate of \( N_{\mu}^{a} \) during this period is the capacity of \( N_{\eta}^{br,a} \). Therefore, the optimal operation rate of \( N_{\eta}^{br,a} \) should also work at its capacity. That is, \( u_{\eta}^{br,a}(t) = c(N_{\eta}^{br,a}) \), when \( t \in [t_{br,\eta}^b, t_{\mu+1}] \).

During the period \([t_{br,x,\eta}^b, t_{br,\eta}^b] \), the products of \( N_{\eta}^{br,a} \) is built up and then used up. The amount of products excluding demand produced by \( N_{\eta}^{br,a} \) at \( t_{br,x,\eta}^b \) is \( Y_{\eta}^{br,a}(t_{br}) \). No matter what exact value of \( Y_{\eta}^{br,a}(t_{br}) \) is, the optimal policy to achieve this amount is to start buildup at \( N_{\eta}^{br,a} \) as late as possible, since it can achieve lower total storage cost. Therefore, to shorten the interval of building up products, \( N_{\eta}^{br,a} \) should work at its capacity during the interval. That is, \( u_{\eta}^{br,a}(t) = c(N_{\eta}^{br,a}) \), when \( t \in [t_{br,x,\eta}^b, t_{br,\eta}^b] \).

After \( t_{\mu+1} \), the operation rate of \( N_{\mu}^{a} \) is 0. There is no need to build up any storage in the branch. Thus, the operation rate of \( N_{\eta}^{br,a} \) should equal to the rate of \( N_{\mu}^{a} \). That is, \( u_{\eta}^{br,a}(t) = 0(t) \), when \( t \in [t_{\mu+1}, t_{dis} + \tau_{dis}] \).

Lemma 32. For an aggregated system satisfying PASTUA, for any \( i \in \{2, 3, ..., \eta\} \),

\[ t_{br,x,i}^b < t_{br,x,i-1}^b. \]

Proof. At time \( t_{br,x,i-1}^b \), the operation rate of \( N_{i-1}^{br,a} \) is its capacity, according to Lemma 29.

Since \( N_{i}^{br,a} \) has lower capacity than \( N_{i-1}^{br,a} \), there should be some buildup at \( N_{i}^{br,a} \) to let \( N_{i-1}^{br,a} \) be able to work at its capacity. Therefore, the buildup at \( N_{i}^{br,a} \) is earlier than \( N_{i-1}^{br,a} \), that is \( t_{br,x,i}^b < t_{br,x,i-1}^b \).

Lemma 33. For an aggregated system satisfying PASTUA, for any \( i \in \{1, 2, ..., \eta\} \) and \( j \in \{1, 2, ..., \mu\} \) such that \( q_{x}(N_{i}^{br,a}) \geq q_{x}(N_{j}) \) and product_{t_{br,a}}^{i} \) and product_{t_{\mu}}^{j} \) are built up sometime during the pre-disruption period, the optimal operation should satisfy:

\[ t_{br,x,j} \leq t_{br,x,i}. \]

Proof. Based on the definition of \( \eta \) and \( \mu \), since \( q_{x}(N_{i}^{br,a}) \geq q_{x}(N_{j}) \), we know that the capacities of nodes among \( \{N_{i}^{br,a}, N_{i-1}^{br,a}, ..., N_{1}^{br,a}, N_{\mu}^{a}, N_{\mu-1}^{a}, ..., N_{j+1}^{a}\} \) are higher than the capacity of \( N_{j}^{a} \). And the unit storage costs of nodes among \( \{N_{i}^{br,a}, N_{i-1}^{br,a}, ..., N_{1}^{br,a}, N_{\mu}^{a}, N_{\mu-1}^{a}, ..., N_{j+1}^{a}\} \) are also higher than the unit storage cost of \( N_{j}^{a} \).

Consider the time \( t_{br,x,i}^b \), node \( N_{i}^{br,a} \) starts building up products at its capacity. The nodes among \( \{N_{i-1}^{br,a}, ..., N_{1}^{br,a}, N_{\mu}^{a}, N_{\mu-1}^{a}, ..., N_{j+1}^{a}\} \) will try to transfer as much buildup as possible to \( N_{j}^{a} \). In this way, the buildup can be stored at the node with lowest unit storage cost among these nodes. Even if \( q_{x}(N_{i}^{br,a}) = q_{x}(N_{j}) \), starting the transfer as early as possible is still the best policy, because the buildup not transferred to \( N_{j}^{a} \) will later on be left among \( \{N_{i-1}^{br,a}, ..., N_{1}^{br,a}, N_{\mu}^{a}, N_{\mu-1}^{a}, ..., N_{j+1}^{a}\} \), which results in higher storage cost. This part of buildup can be reduced to minimum if starting the transfer to \( N_{j}^{a} \) is as early as possible.
Therefore, at time $t_{br,i}^{bx,j}$, nodes among $\{N_{i-1}^{br,a},...,N_1^{br,a},N_{i+1}^{br,a},...,N_{\mu}^{br,a}\} \setminus \{N_{j-1}^{a},N_{j+1}^{a},...,N_{\mu}^{a}\}$ will work at the capacity of $N_{j}^{a}$ to maximize the transfer rate. Since $N_{j-1}^{a}$ has a lower operation rate than the capacity of $N_{j}^{a}$, products of $N_{j}^{a}$ is built up.

Thus, the buildup at $N_{j}^{a}$ can not be later than at $N_{i}^{br,a}$. That is, $t_{bx,j} \leq t_{br,i}^{bx,j}$.

**Lemma 34.** For an aggregated system satisfying PASTUA, for $\mu - 1 \geq j \geq 1$, the optimal operation rate of $N_{j}^{a}$ satisfies:

$$u_{j}^{a}(t) = \begin{cases} 
  u_{j}^{a-1}(t), & \text{when } t \in [t_{dis} - \tau_{f}, t_{bx,j}), \\
  c(N_{j}^{a}), & \text{when } t \in [t_{bx,j}, t_{j+1}), \\
  u_{j+1}(t), & \text{when } t \in [t_{j+1}, t_{dis} + \tau_{dis}].
\end{cases} \quad (9.8)$$

**Proof.** Denote $i$ as the largest index in $\{0, 1, 2, \ldots, \eta\}$ such that $q_{x}(N_{i}^{br,a}) \geq q_{x}(N_{j}^{a})$. Based on Lemma 33, we know $t_{bx,j} \leq t_{br,i}^{bx,i}$.

**During the period** $[t_{dis} - \tau_{f}, t_{bx,j})$

The product of $N_{j}^{a}$ hasn’t started buildup. Therefore, the operation rate of $N_{j}^{a}$ equals to that of $N_{j-1}^{a}$. That is, $u_{j}^{a}(t) = u_{j-1}^{a}(t)$, when $t \in [t_{dis} - \tau_{f}, t_{bx,j})$.

**During the period** $[t_{bx,j}, \min\{t_{bx,j+1}, t_{br,i}^{bx,j}\})$

Products of $N_{j}^{a}$ start buildup, and none of products of $N_{j+1}^{a}$ and $N_{i}^{br,a}$ starts buildup. Based on Lemma 32, we know none of $\{N_{i}^{br,a}, N_{i-1}^{br,a},...,N_{1}^{br,a}\}$ starts building up its products. Also, based on lemma 26, none of $\{N_{\mu}, N_{\mu-1}^{a},...,N_{j+1}^{a}\}$ starts building up its product. Therefore, these nodes has the same operation rate of $N_{j}^{a}$.

Also, based on lemmas 26 and 23, $\{N_{L}^{a}, N_{L-1}^{a},...,N_{\mu+1}^{a}\}$ do not started buildup during this period.

Thus, nodes among $\{N_{L}^{a}, N_{L-1}^{a},...,N_{j+1}^{a}\}$ and $\{N_{i}^{br,a}, N_{i-1}^{br,a},...,N_{1}^{br,a}\}$ all have the same operation rate of $N_{j}^{a}$. And $N_{j}^{a}$ has the lowest capacity among these nodes (based on the definition of $\mu$ and $\eta$). We can consider these node as a single node whose capacity is $c(N_{j}^{a})$.

At time $\min\{t_{bx,j+1}, t_{br,i}^{bx,j}\}$, the amount of buildup at $N_{j}^{a}$ is $x_{j}^{a}(\min\{t_{bx,j+1}, t_{br,i}^{bx,j}\})$. Nodes among $\{N_{\eta}^{br,a}, N_{\eta-1}^{br,a},...,N_{i+1}^{br,a}\}$ will produce sufficient products for such amount of buildup. The operation rate of $N_{j}^{a}$ determines the transferring of products from $\{N_{\eta}^{br,a}, N_{\eta-1}^{br,a},...,N_{i+1}^{br,a}\}$ to $N_{j}^{a}$. No matter what the exact amount to be transferred is, the optimal policy for this amount is to start transfer as late as possible, since the unit storage cost at $N_{j}^{a}$ is higher than the unit storage costs at nodes among $\{N_{\eta}^{br,a}, N_{\eta-1}^{br,a},...,N_{i+1}^{br,a}\}$. This policy can keep products stay at locations with lower unit storage cost for longer time, instead of begin transferred to $N_{j}^{a}$ which has higher unit storage cost. Then, node $N_{j}^{a}$ should work at its highest possible rate during this period, which is the capacity. That is, $u_{j}^{a}(t) = c(N_{j}^{a})$ when $t \in [t_{bx,j}, \min\{t_{bx,j+1}, t_{br,i}^{bx,j}\})$. 

128
During the period $\{\min\{t_{bx,j+1}, t_{br,i}^{br}\}, t_{bx,j+1}\}$

This period exists only when $t_{bx,j+1} > t_{br,i}^{br}$. During this period, products start buildup among $\{N_i^{br,a}, N_{i-1}^{br,a}, \ldots, N_1^{br,a}\}$, but nodes $\{N_L^{a}, N_{L-1}^{a}, \ldots, N_j^{a}\}$ haven’t started buildup products. Since the unit storage costs at nodes among $\{N_i^{br,a}, N_{i-1}^{br,a}, \ldots, N_1^{br,a}\}$ are higher than at $N_j^{a}$, then $N_j^{a}$ should work at its highest rate to transfer the buildup among $\{N_i^{br,a}, N_{i-1}^{br,a}, \ldots, N_1^{br,a}\}$ to $N_j^{a}$. Based on Lemmas 28 and 27, before $t_{bx,j+1}$, the buildup among $\{N_i^{br,a}, N_{i-1}^{br,a}, \ldots, N_1^{br,a}\}$ won’t be used up. Also, they have higher capacities than $N_j^{a}$. Thus, the highest rate of $N_j^{a}$ is its capacity. That is, $u_j^{a}(t) = c(N_j^{a})$ when $t \in [\min\{t_{bx,j+1}, t_{br,i}^{br}\}, t_{bx,j+1}]$.

During the period $\{t_{bx,j+1}, t_{j+1}\}$

During this period, the buildup exists at $N_j^{a+1}$. To achieve minimum storage cost, $N_j^{a}$ must work at the highest rate to push products downstream. Therefore, $N_j^{a}$ should work at its capacity. That is, $u_j^{a}(t) = u_{j+1}(t)$, when $t \in [t_{bx,j+1}, t_{j+1}]$.

During the period $\{t_{j+1}, t_{dis} + \tau_{dis}\}$

The amount of the product of $N_j^{a+1}$ is 0. Therefore, the operation rate of $N_j^{a}$ should equal to that of $N_{j+1}^{a}$. That is, $u_j^{a}(t) = u_{j+1}(t)$, when $t \in [t_{j+1}, t_{dis} + \tau_{dis}]$.

Lemma 24 and 34 can be combined into the following Lemma 35

**Lemma 35.** For an aggregated system satisfying PASTUA, for $L \geq j \geq \mu + 1$ and $\mu - 1 \geq j \geq 1$, the optimal operation rate of $N_j^{a}$ satisfies:

$$u_j^{a}(t) = \begin{cases} 
    u_{j-1}^{a}(t), & \text{when } t \in [t_{dis} - \tau_f, t_{bx,j}], \\
    c(N_j^{a}), & \text{when } t \in [t_{bx,j}, t_{j+1}], \\
    u_{j+1}^{a}(t), & \text{when } t \in [t_{j+1}, t_{dis} + \tau_{dis}].
\end{cases} \quad (9.9)$$

Notice that $u_{L+1}^{a}(t) = 0$, and $t_{L+1} = t_{dis}$.

**Lemma 36.** For an aggregated system satisfying PASTUA, the optimal operation rate of $N_{\mu}^{a}$ satisfies:

$$u_{\mu}^{a}(t) = \begin{cases} 
    u_{\mu-1}^{a}(t), & \text{when } t \in [t_{dis} - \tau_f, t_{bx,\mu}], \\
    c(N_{\mu}^{a}), & \text{when } t \in [t_{bx,\mu}, t_{br}^{br}], \\
    u_{1}^{br,a}(t), & \text{when } t \in [t_{br}^{br}, t_{\mu+1}], \\
    0, & \text{when } t \in [t_{\mu+1}, t_{dis} + \tau_{dis}].
\end{cases} \quad (9.10)$$

**Proof.** During the period of $[t_{dis} - \tau_f, t_{bx,\mu})$, $N_{\mu}^{a}$ has not started building up products. Thus, the operation rate of $N_{\mu}^{a}$ should equal to the rate of its downstream node $N_{\mu-1}^{a}$. That is, $u_{\mu}^{a}(t) = u_{\mu-1}^{a}(t)$, when $t \in [t_{dis} - \tau_f, t_{bx,\mu})$.

During the period of $[t_{bx,\mu}, t_{bx,\mu+1})$, $N_{\mu}^{a}$ starts buildup but all the nodes upstream $N_{\mu}^{a}$ have not started buildup. Then all the nodes upstream of $N_{\mu}^{a}$ have the same operation rate of $N_{\mu}^{a}$. During this period, $N_{\mu}^{a}$ are transferring the products from branch to $N_{\mu}^{a}$. At $t_{bx,\mu+1}$, the buildup at $N_{\mu}^{a}$ is $x_{\mu}^{a}(t_{bx,\mu+1})$. Nodes in the branch will build up sufficient products by
to feed \( N^a_\mu \) for such an amount of storage at \( N^a_\mu \). Since the unit storage cost at \( N^a_\mu \) is higher than in the branch, no matter what value of \( x^a_\mu(t_{bx,\mu+1}) \) is, node \( N^a_\mu \) should start buildup as late as possible. This policy can keep the storage in the branch as long as possible, so that a lower storage cost can be achieved. Then, once the buildup starts, \( N^a_\mu \) should work at its highest rate to achieve \( x^a_\mu(t_{bx,\mu+1}) \) amount of buildup. Thus, \( N^a_\mu \) will work at its capacity. That is, \( u^a_\mu(t) = c(N^a_\mu) \), when \( t \in [t_{bx,\mu}, t_{bx,\mu+1}) \).

During the period of \([t_{bx,\mu+1}, t^br_1]\), both \( N^a_{\mu+1} \) and \( N^{br,a}_1 \) have storage. Then, \( N^a_\mu \) should work at its highest rate, which is its capacity. This can not only transfer products from \( N^a_{\mu+1} \) to \( N^a_\mu \), but also reduce the buildup in the branch, so that the total storage cost is kept to minimum. Therefore, \( u^a_\mu(t) = c(N^a_\mu) \), when \( t \in [t_{bx,\mu+1}, t^br_1) \).

From \( t^br_1 \), the buildup at \( N^{br,a}_1 \) is used up. Therefore, the operation rate of \( N^a_\mu \) should equal to the rate of \( N^{br,a}_1 \). That is, \( u^a_\mu(t) = u^{br,a}_1(t) \) when \( t \in [t^br_1, t_{dis} + \tau_{dis}) \). Since the operation rate of \( N^{br,a}_1 \) is 0 after \( t_{\mu+1} \), then \( u^a_\mu(t) = u^{br,a}_1(t) = 0 \) when \( t \in [t_{\mu+1}, t_{dis} + \tau_{dis}) \).

The optimal operation is represented by Figure 9.5.

The primary objective is to minimize the lost demand. Buildups at nodes among \( \{N^a_L, N^a_{L-1}, ..., N^a_1\} \) are used to satisfy the demand during the disruption period. For \( j \in \{L, L-1, \ldots \} \),
..., 1}, denote $x^a_j(t_{\text{dis}})$ as the the amount of the buildup of product $a$ at $t_{\text{dis}}$, and denote $X^a_{bx}$ as the sum of all the buildups of product $a$ at $t_{\text{dis}}$.

Due to the capacity of $N^a_L$, $X^a_{bx} \leq \tau_f[c(N^a_L) - \tilde{d}]$. Lost demand equals to 0 when $X^a_{bx} \geq \tilde{d}t_{\text{dis}}$. For any $X^a_{bx} > \tau_{\text{dis}}\tilde{d}$, it does not make less lost demand but introduces more operation cost and storage cost. Therefore, the optimal $X^a_{bx}$ should be less than or equal to $\tau_{\text{dis}}\tilde{d}$. Thus, the optimal $X^a_{bx}$ is:

$$X^a_{bx} = \min\{\tau_{\text{dis}}\tilde{d}, \tau_f[c(N^a_L) - \tilde{d}]\}. \quad (9.11)$$

Denote $t_{bx,L+1} = t_{\text{dis}}$. Then,

$$\sum_{i=1}^{L} [c(N^a_i) - \tilde{d}](t_{bx,i+1} - t_{bx,i}) = X^a_{bx} = \min\{\tau_{\text{dis}}\tilde{d}, \tau_f[c(N^a_L) - \tilde{d}]\}. \quad (9.11)$$

This becomes a constraint on $\{t_{bx,L}, t_{bx,L-1}, ..., t_{bx,1}\}$.

**Operation Cost**

Since inventory equals to 0 at time $t_{\text{dis}} - \tau_f$ and at time $t_{\text{dis}} + \tau_{\text{dis}}$, the integral of the operation of any node over the period $[t_{\text{dis}} - \tau_f, t_{\text{dis}} + \tau_{\text{dis}})$ equals to each other. That is,

$$\int_{t_{\text{dis}} - \tau_f}^{t_{\text{dis}} + \tau_{\text{dis}}} \tilde{u}_j(t) dt = \int_{t_{\text{dis}} - \tau_f}^{t_{\text{dis}} + \tau_{\text{dis}}} \tilde{u}^\text{br}_i(t) dt = \tau_f \tilde{d} + X^a_{bx}.$$ 

This means that the total operation cost is a fixed value once the lost demand is minimized. We do not need to consider it in the optimization problem.

**The Storage Cost in the Branch**

Define $S$ matrix to represent the interleaving of $\{t_{bx,j}\}$ and $\{t_{bx,i}^\text{br}\}$. $S$ is a $\mu \times \eta$ matrix such that each elements $S_{j,i}$ satisfies:

$$S_{j,i} = \begin{cases} 
1, & \text{if } t_{bx,j} \geq t_{bx,i}^\text{br} \\
0, & \text{otherwise}
\end{cases}$$

Consider $t_{bx,0} = -\infty$. We can extend $S$ matrix to include $S_{0,i} = 0$ for $i \in \{0, 1, ..., \eta\}$, and $S_{j,0} = 0$ for any $j \in \{0, 1, 2, ..., \mu - 1\}$. Notice that $t_{bx,\mu} = t_{bx,0}^\text{br}$. Then, we can define $S_{\mu,0} = 1$.

**Lemma 37**. $S$ matrix satisfies the following statement:

1. if $S_{j,i} = 0$, then $S_{k,l} = 0$, $\forall k \leq j$, and $l \leq i$, and
2. if $S_{j,i} = 1$, then $S_{k,l} = 1$, $\forall k \geq j$, and $l \geq i$.

When calculating the storage costs in the branch, we divide the total area in the figure of buildup process into a number of triangles. The $S$ matrix determines the lines composing each triangle.
We assume that \( t_{bx,j} \) and \( t_{bx,i}^{br} \) are not overlapping even if \( t_{bx,j} = t_{bx,i}^{br} \). In this case, we assume \( t_{bx,i}^{br} \) is on the left of \( t_{bx,j} \) geometrically. This will not change the results of calculating the total storage cost. Instead, this can help the calculation. When \( t_{bx,j} = t_{bx,i}^{br} \), we assume there still exists a triangle between them. And we will show later that the area of this triangle is 0 due to the equality.

Denote \( Z_j^a \) as the segment line of \( Y_j^a \) from \( t_{bx,j} \) to \( t_{j+1} \) for \( i \in \{1, 2, ..., \mu - 1\} \). Denote \( Z_{j}^{br} \) as the segment line of \( Y_{j}^{br} \) from \( t_{bx,j} \) to \( t_{1}^{br} \). For \( i \in \{0, 1, ..., \eta - 1\} \), denote \( Z_i^{br} \) as the segment line of \( Y_i^{br,a} \) from \( t_{bx,i}^{br} \) to \( t_{i+1}^{br} \). Denote \( Z_{\eta}^{br} \) as the segment line of \( Y_{\eta}^{br,a} \) from \( t_{bx,\eta} \) to \( t_{\mu+1} \). For each triangle of product \( t_i^{br,a} \) where \( i \in \{1, 2, ..., \eta\} \), its top corner point is \((Y_i^{br}, Y_{\mu}(t_i^{br}))\). Its left and right bottom corner points can be the intersection point of either a line in \( \{Z_j^a\} \) and a line in \( \{Z_i^{br}\} \) or two lines in \( \{Z_j^a\} \). Denote \( \text{Int}_j \) as the intersection point of \( Z_{j-1}^a \) and \( Z_{j}^a \). Notice that \( Z_0^a \) is the horizon axis. Denote \( Z_i^{br,nt} \) as the segment line from \( \text{Int}_j \) to the point \((t_i^{br}, Y_{\mu}(t_i^{br}))\) if \( t_{bx,i}^{br} \leq t_{bx,j} < t_{bx,i-1}^{br} \). Therefore, the left and right edges of each triangle are the lines in \( \{Z_i^{br,nt}\} \cup \{Z_i^{br}\} \).

There are four types of triangles, depending on their left and right edges. Denote the four types as: \((Z^{br}, Z^{br}), (Z^{br}, Z^{int}), (Z^{int}, Z^{br}), \) and \((Z^{int}, Z^{int})\).

**Type \((Z^{br}, Z^{br})\)** Check the \( S \) matrix to determine a triangle of Type \((Z^{br}, Z^{br})\). If there exists an \( j \in \{1, 2, ..., \mu\} \) and a \( i \in \{1, 2, ..., \eta\} \) such that:

\[
S_{j,i} = 1, S_{j,i-1} = 1, \text{ and } S_{j-1,i} = 0,
\]

then

\[
t_{bx,j-1} < t_{bx,i}^{br} < t_{bx,i-1}^{br} < t_{bx,j}.
\]

Notice that \( t_{bx,0} = -\infty \) There is a Type \((Z^{br}, Z^{br})\) triangle between \((Z_i^{br}, Z_i^{br})\).

The three corner points of this triangle are \((t_i^{br}, Y_{\mu}(t_i^{br})), (t_{bx,i}^{br}, Y_{j-1}(t_{bx,i}^{br})), \) and \((t_{bx,i-1}^{br}, Y_{j-1}(t_{bx,i-1}^{br}))\). Besides, the slopes of the three edges, \( Z_i^{br}, Z_{i-1}^{br}, \) and \( Z_{j-1}^{br} \), are \( c(N_i^{br,a}) - \tilde{d}, c(N_{i-1}^{br,a}) - \tilde{d} \) and \( c(N_{j-1}^{br,a}) - \tilde{d} \), respectively. Then, the area of this triangle is:

\[
\frac{1}{2} [c(N_i^{br,a}) - c(N_{j-1}^{br,a})](t_{bx,i-1}^{br} - t_{bx,i}^{br})(t_i^{br} - t_{bx,i}^{br}).
\]

This triangle represents the storage cost of product \( t_i^{br,a} \). Therefore, the storage cost represented by this triangle is:

\[
C_{j,i}^{(br,br)} = \frac{1}{2} q_2(N_i^{br,a})[c(N_i^{br,a}) - c(N_{j-1}^{br,a})](t_{bx,i-1}^{br} - t_{bx,i}^{br})(t_i^{br} - t_{bx,i}^{br}). \tag{9.12}
\]

**Type \((Z^{br}, Z^{int})\)** Check the \( S \) matrix to determine a triangle of Type \((Z^{br}, Z^{int})\). If there exists an \( j \in \{1, 2, ..., \mu - 1\} \) and a \( i \in \{1, 2, ..., \eta\} \) such that:

\[
S_{j,i} = 1, S_{j,i-1} = 0, \text{ and } S_{j-1,i} = 0,
\]

then

\[
t_{bx,j-1} < t_{bx,i}^{br} < t_{bx,j} < t_{bx,i-1}^{br}.
\]

132
Notice that $t_{bx,0} = -\infty$. There is a Type $(Z^{br}, Z^{Int})$ triangle between $(Z^{br}_i, Z^{Int}_j)$.

The three corner points of this triangle are $(t^{br}_{bx,i}, Y^a_j(t^{br}_{bx,j})), (t^{br}_{bx,j}, Y^a_j(t^{br}_{bx,j}))$ and $(t^{br}_{i+1}, Y^a_{\mu}(t^{br}_{i+1}))$. The slope of the left edge $Z^{br}_i$ is $c(N^{br,a}_i) - \tilde{d}$. The slope of the bottom edge $Z^{a}_j$ is $c(N^{a}_{j-1}) - \tilde{d}$. Then, the area of this triangle is:

$$\frac{1}{2}[c(N^{br,a}_i) - c(N^{a}_{j-1})](t^{br}_{bx,j} - t^{br}_{bx,i})(t^{br}_{i} - t^{br}_{bx,i}).$$

This triangle represents a part of the storage cost of product $i^{br,a}$. Therefore, the storage cost represented by this triangle is:

$$C^{(br,Int)}_{j,i} = \frac{1}{2} q_x(N^{br,a}_i)[c(N^{br,a}_i) - c(N^{a}_j)](t^{br}_{bx,j} - t^{br}_{bx,i})(t^{br}_i - t^{br}_{bx,i}).$$  \hspace{1cm} (9.13)

**Type $(Z^{Int}, Z^{br})$** Check the $S$ matrix to determine a triangle of Type $(Z^{Int}, Z^{br})$. If there exists an $j \in \{1, 2, ..., \mu - 1\}$ and a $i \in \{0, 1, ..., \eta - 1\}$ such that:

$$S_{j,i} = 0, S_{j+1,i} = 1, \text{ and } S_{j,i+1} = 1,$$

then

$$t^{br}_{bx,i+1} \leq t^{br}_{bx,j} < t^{br}_{bx,i} \leq t^{br}_{bx,j+1}.$$\hspace{1cm} (9.14)

There is a Type $(Z^{Int}, Z^{br})$ triangle between $(Z^{Int}_j, Z^{br}_i)$.

The three corner points of this triangle are $(t^{br}_{bx,j}, Y^a_j(t^{br}_{bx,j})), (t^{br}_{bx,i}, Y^a_i(t^{br}_{bx,i}))$ and $(t^{br}_{i+1}, Y^a_{\mu}(t^{br}_{i+1}))$. The slope of the right edge $Z^{br}_i$ is $c(N^{br,a}_i) - \tilde{d}$. The slope of the bottom edge $Z^{a}_j$ is $c(N^{a}_{j}) - \tilde{d}$. Then, the area of this triangle is:

$$\frac{1}{2}[c(N^{br,a}_i) - c(N^{a}_j)](t^{br}_{bx,i} - t^{br}_{bx,j})(t^{br}_i - t^{br}_{bx,i}).$$

This triangle represents a part of the storage cost of product $i^{br,a}+1$. Therefore, the storage cost represented by this triangle is:

$$C^{(Int,br)}_{j,i} = \frac{1}{2} q_x(N^{br,a}_{i+1})[c(N^{br,a}_i) - c(N^{a}_j)](t^{br}_{bx,i} - t^{br}_{bx,j})(t^{br}_i - t^{br}_{bx,i}).$$  \hspace{1cm} (9.15)

**Type $(Z^{Int}, Z^{Int})$** Check the $S$ matrix to determine a triangle of Type $(Z^{Int}, Z^{Int})$. If there exists an $j \in \{1, 2, ..., \mu - 2\}$ and a $i \in \{0, 1, ..., \eta - 1\}$ such that:

$$S_{j,i} = 0, S_{j+1,i} = 0, \text{ and } S_{j,i+1} = 1,$$

then

$$t^{br}_{bx,i+1} \leq t^{br}_{bx,j} < t^{br}_{bx,i+1} < t^{br}_{bx,i}.$$ \hspace{1cm} (9.16)

There is a Type $(Z^{Int}, Z^{Int})$ triangle between $(Z^{Int}_j, I^{Int}_{j+1})$. 

133
The three corner points of this triangle are \((t_{bx,j}, Y_j^a(t_{bx,j})), (t_{bx,j+1}, Y_j^a(t_{bx,j+1}))\) and \((t_{i+1}^{br}, Y_{\mu}^a(t_{i+1}^{br}))\). The slope of the bottom edge \(Z_j^b\) is \(c(N_j^a) - \tilde{d}\). Then, the area of this triangle is:

\[
\frac{1}{2} \left\{ \sum_{k=j}^{i-1} (t_{bx,k+1} - t_{bx,k})[c(N_k^a) - \tilde{d}] + (t_{i+1}^{br} - t_{bx,\mu})[c(N_\mu^a) - \tilde{d}] \\
+ \sum_{k=1}^{i}(t_{k+1}^{br} - t_k^{br})[c(N_k^{br,a}) - \tilde{d}] - (t_{i+1}^{br} - t_{bx,j})[c(N_j^a) - \tilde{d}] \right\} (t_{bx,j+1} - t_{bx,j})
\]

\[
= \frac{1}{2} \left\{ \sum_{k=j}^{i-1} (t_{bx,k+1} - t_{bx,k})c(N_k^a) + (t_{i+1}^{br} - t_{bx,\mu})c(N_\mu^a) + \sum_{k=1}^{i}(t_{k+1}^{br} - t_k^{br})c(N_k^{br,a}) \\
- (t_{i+1}^{br} - t_{bx,j})c(N_j^a) \right\} (t_{bx,j+1} - t_{bx,j})
\]

This triangle represents a part of the storage cost of product \(^{br,a}_{i+1}\). Therefore, the storage cost represented by this triangle is:

\[
C_{j,i}^{(Int,Int)} = \frac{1}{2} q_x(N_{i+1}^{br,a}) \left\{ \sum_{k=j}^{i-1} (t_{bx,k+1} - t_{bx,k})c(N_k^a) + (t_{i+1}^{br} - t_{bx,\mu})c(N_\mu^a) \\
+ \sum_{k=1}^{i}(t_{k+1}^{br} - t_k^{br})c(N_k^{br,a}) - (t_{i+1}^{br} - t_{bx,j})c(N_j^a) \right\} (t_{bx,j+1} - t_{bx,j})
\]

(9.15)

Next, we need to determine all the triangles with a given \(S\) matrix and sum up the storage costs represented by these triangles. Since each triangle can be determined by the elements in \(S\) matrix, we simply check each element in \(S\) matrix and test the neighbor elements around it. If the values of the tested elements satisfy the values associated with any type of triangle, we determine that a corresponding triangle appears in the buildup figure of this given \(S\). Notice that any triangle appears only once in total when checking all the elements in \(S\) matrix. This is because each triangle is associated with three specific elements in \(S\) matrix. For example, a triangle of type \((Z^{Int}, Z^{Int})\) is associated with \(S_{j,i}, S_{j+1,i}\) and \(S_{j,i+1}\). It can only appear when checking \(S_{j,i}\) with its neighbors \((S_{j-1,i}, S_{j,i-1}, S_{j+1,i}\) and \(S_{j,i+1}\)). And it cannot appear when checking any other element in \(S\) such as \(S_{j+1,i}\), since we check \(S_{j+1,i}\) with its neighbors \((S_{j,i}, S_{j+1,i-1}, S_{j+2,i}\) and \(S_{j+1,i+1}\)), which do not include \(S_{j,i}\). For each \(S_{j,i}\) where \(j \in \{1, 2, ..., \mu\}\) and \(i \in \{0, 1, 2, ..., \eta\}\), we denote \(C_{j,i}^{br}\) as the storage cost represented by the triangles which appear when checking \(S_{j,i}\). Therefore, the total storage cost of the buildup in the branch, which is the sum of the storage cost represented by all the triangles, can be formulated as:

\[
C_{bx}^{br} = \sum_{j=1}^{\mu} \sum_{i=0}^{\eta} C_{j,i}^{br}.
\]

(9.16)

For \(C_{j,i}^{br}\), we use four coefficients, \(\{a_{j,i}^{(br,Int)}, a_{j,i}^{(br,br)}, a_{j,i}^{(Int,Int)}, a_{j,i}^{(Int,br)}\}\), to formulate the cost. Then \(C_{j,i}^{br}\) becomes:

\[
C_{j,i}^{br} = a_{j,i}^{(br,Int)} C_{j,i}^{(br,Int)} + a_{j,i}^{(br,br)} C_{j,i}^{(br,br)} + a_{j,i}^{(Int,Int)} C_{j,i}^{(Int,Int)} + a_{j,i}^{(Int,br)} C_{j,i}^{(Int,br)}.
\]

(9.17)
The value of any coefficient in \( \{ a_{j,i}^{(br,Int)} , a_{j,i}^{(br,br)} , a_{j,i}^{(Int,Int)} , a_{j,i}^{(Int,br)} \} \) can only be 1 or 0. A coefficient equals to 1 when the corresponding triangle appears; a coefficient equals to 0 when the corresponding triangle doesn’t appear. For example if a Type \((Z^{br},Z^{br})\) triangle and a Type \((Z^{Int},Z^{br})\) triangle appear when checking \(S_{j,i}\), then \( a_{j,i}^{(br,br)} = a_{j,i}^{(Int,br)} = 1 \) and \( a_{j,i}^{(Int,Int)} = a_{j,i}^{(br,Int)} = 0 \).

The relationship of \(S\) matrix, the types of triangles and coefficients in \(C_{j,i}^{br}\) can be represented by Table 9.1.

Then, the coefficients in \(C_{j,i}^{br}\) can be calculated by \(S\) matrix with the formulas below:

\[
a_{j,i}^{(br,Int)} = \begin{cases} S_{j,i}(1 - S_{j-1,i})(1 - S_{j,i-1}), & \text{when } 1 \leq i \leq \eta, \\ 0, & \text{when } i = 0. \end{cases}
\]  

(9.18)

\[
a_{j,i}^{(br,br)} = \begin{cases} S_{j,i}(1 - S_{j-1,i})S_{j,i-1}, & \text{when } 1 \leq i \leq \eta, \\ 0, & \text{when } i = 0. \end{cases}
\]  

(9.19)

\[
a_{j,i}^{(Int,Int)} = \begin{cases} (1 - S_{j,i})S_{j,i+1}(1 - S_{j+1,i}), & \text{when } 1 \leq j \leq \mu - 1 \text{ and } 0 \leq i < \eta, \\ 0, & \text{when } j = \mu \text{ or } i = \eta. \end{cases}
\]  

(9.20)

\[
a_{j,i}^{(Int,br)} = \begin{cases} (1 - S_{j,i})S_{j,i+1}S_{j+1,i}, & \text{when } 1 \leq j \leq \mu - 1 \text{ and } 0 \leq i < \eta, \\ 0, & \text{when } j = \mu \text{ or } i = \eta. \end{cases}
\]  

(9.21)

\(t_{bx,i}^{br}\) can be expressed as a function of \(\{t_{bx,j}\}\) and \(\{\tilde{t}^{br}\}\), where \(i \in \{1, 2, ..., \eta\}\) and \(j \in \{1, 2, ..., \mu\}\). Consider the case when \(t_{bx,j-1} < t_{bx,i}^{br} \leq t_{bx,j}\). Then,

\[
\frac{Y_{j-1}^{a}(t_{bx,i}^{br}) - Y_{j-1}^{a}(t_{bx,i}^{br})}{\tilde{t}^{br}_{bx,i} - t_{bx,i}^{br}} = c(N_{i}^{br,a}) - \tilde{d},
\]

where

\[
Y_{\mu}^{a}(t_{i}^{br}) = \sum_{j=1}^{\mu-1}[c(N_{j}^{a}) - \tilde{d}](t_{bx,j+1} - t_{bx,j}) + [c(N_{\mu}^{a}) - \tilde{d}](t_{1}^{br} - t_{bx,\mu})
\]

\[
+ \sum_{k=1}^{i-1}[c(N_{k}^{br,a}) - \tilde{d}](t_{k+1}^{br} - t_{k}^{br}),
\]

and

\[
Y_{j-1}^{a}(t_{bx,i}^{br}) = \sum_{k=1}^{j-2}[c(N_{k}^{a}) - \tilde{d}](t_{bx,k+1} - t_{bx,k}) + [c(N_{j-1}^{a}) - \tilde{d}](t_{k+1}^{br} - t_{bx,j-1}).
\]

Notice that \(Y_{j-1}^{a}(t_{bx,i}^{br}) = 0\) when \(j = 1\).
Then, when \( t_{bx,j-1} < t_{bx,i}^\text{br} \leq t_{bx,j} \),

\[
t_{bx,i}^\text{br} = Y_{\mu}^a(t_i^\text{br}) - \sum_{k=1}^{j-2} [c(N_k^a) - \bar{d}] (t_{bx,k+1} - t_{bx,k}) + [c(N_{j-1}^a) - \bar{d}] t_{bx,j-1} - [c(N_i^\text{br,a}) - \bar{d}] t_i^\text{br}.
\]

In this case, \( S_{j,i} = 1 \) and \( S_{j-1,i} = 0 \). In general, for \( i \in \{1, 2, ..., \eta\} \),

\[
t_{bx,i}^\text{br} = \sum_{j=1}^{\mu} \frac{(1 - S_{j-1,i}) S_{j,i}}{c(N_j^a) - c(N_i^\text{br,a})} \left[ Y_{\mu}^a(t_i^\text{br}) - \sum_{k=1}^{j-2} [c(N_k^a) - \bar{d}] (t_{bx,k+1} - t_{bx,k})
\right.
\]
\[
+ [c(N_j^a) - \bar{d}] t_{bx,j-1} - [c(N_i^\text{br,a}) - \bar{d}] t_i^\text{br} \right].
\]

### Table 9.1: \( S \) matrix, the types of triangles, and coefficients in \( C_{j,i}^\text{br} \)

<table>
<thead>
<tr>
<th>( S_{j,i} )</th>
<th>( S_{j-1,i} )</th>
<th>triangle type</th>
<th>( a(\text{br, Int}) )</th>
<th>( a(\text{br, br}) )</th>
<th>( a(\text{Int, Int}) )</th>
<th>( a(\text{Int, br}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>either</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>either</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>either</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>either</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>either</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

136
The Storage Cost at \{N_a^L, N_a^{L-1}, \ldots, N_a^{\mu+2}\}

Consider any \(j \in \{L, L-1, \ldots, \mu+2\}\). We know that the operation rate of \(N_a^j\) is its capacity over the interval \([t_{bx,j}, t_{j+1})\). Also, the operation rate of \(N_a^{j-1}\) is the capacity of \(N_a^j\) over the interval \([t_{bx,j}, t_j)\). The triangle of product\(^a\)_\(j\) can be represented by Figure 9.6. When

\[
Y_{a}^{\mu+1}(t_{bx,\mu+1}) = Y_{a}^{\mu}(t_{bx,\mu+1}) = \sum_{j=1}^{\mu} [c(N_a^j) - \tilde{d}](t_{bx,j+1} - t_{bx,j}).
\]

\[
Y_{a}^{\mu+1}(t_{\mu+2}) = Y_{a}^{\mu+1}(t_{bx,\mu+1}) + [c(N_a^{\mu+1}) - \tilde{d}](t_{\mu+2} - t_{bx,\mu+1}).
\]

\[
Y_{a}^{\mu+1}(t_{\mu+1}) = Y_{a}^{\mu+1}(t_{bx,\mu+1}) + [c(N_a^{\mu+1}) - \tilde{d}](t_{\mu+2} - t_{bx,\mu+1}) - \tilde{d}(t_{\mu+1} - t_{\mu+2}).
\]
For $i \in \{1, 2, ..., \eta\}$,

$$Y_{\mu}^a(t_{br}^i) = Y_{\mu}^a(t_{bx, \mu+1}) + [c(N_{\mu}^a) - \tilde{d}](t_{br}^i - t_{bx, \mu+1}) + \sum_{k=1}^{i-1} [c(N_{\mu}^{br,a}) - \tilde{d}](t_{br}^k - t_{br}^{k+1}).$$

Since $Y_{\mu}^a(t_{\mu+1}) = Y_{\mu+1}^a(t_{\mu+1})$, then,

$$t_{\mu+1} = \frac{c(N_{\mu+1}^a)(t_{\mu+2} - t_{bx, \mu+1}) - c(N_{\mu}^a)(t_{br}^1 - t_{bx, \mu+1}) - \sum_{k=1}^{\eta-1}[c(N_{\mu}^{br,a})(t_{br}^k - t_{br}^{k+1})]}{c(N_{\eta}^{br,a})}$$

$$+ t_{br}^\eta.$$  \hspace{1cm} (9.25)

The storage cost of product$^a_{\mu+1}$ is:

$$C_x^{(\mu+1)} = \frac{q_x(N_{\mu+1}^a)}{2} \left\{ [Y_{\mu}^a(t_{bx, \mu+1}) + Y_{\mu+1}^a(t_{\mu+2})](t_{\mu+2} - t_{bx, \mu+1}) + [Y_{\mu}^a(t_{\mu+1}) + Y_{\mu+1}^a(t_{\mu+2})](t_{\mu+1} - t_{\mu+2}) - [Y_{\mu}^a(t_{bx, \mu+1}) + Y_{\mu}^a(t_{br}^1)](t_{br}^1 - t_{bx, \mu+1}) - \sum_{i=1}^{\eta-1}[Y_{\mu}^a(t_{br}^i) + Y_{\mu}^a(t_{br}^{i+1})](t_{br}^{i+1} - t_{br}^i) - [Y_{\mu}^a(t_{\mu+1}) + Y_{\mu}^a(t_{\eta}^1)](t_{\mu+1} - t_{\eta}^1) \right\}. \hspace{1cm} (9.26)$$

**The Storage Cost at \{N_{\mu}^a, N_{\mu-1}^a, ..., N_1^a\}**

In general, the buildup process of these nodes can be represented by the figure below: \{t_\mu, t_{\mu-1}, ..., t_2, t_1\} is interleaving with \{t_{br}^1, t_{br}^2, ..., t_{br}^{\eta}, t_{\mu+1}\}. To calculate the storage cost in Chain 1, we use the same method used when calculating the storage cost in the branch.

For $i \in \{1, 2, ..., \eta\}$ and $j \in \{1, 2, ..., \mu\}$

$$T_{j,i} = \begin{cases} 
1, & \text{if } t_{br}^i \leq t_j, \\
0, & \text{if } t_{br}^i > t_j.
\end{cases}$$
Besides, for $j \in \{1, 2, ..., \mu\}$,

$$T_{j, \eta + 1} = \begin{cases} 1, & \text{if } t_{\mu + 1} \leq t_j, \\ 0, & \text{if } t_{\mu + 1} > t_j. \end{cases}$$

Define $T_{j, \eta + 2} := 0$ and $T_{\mu + 1, i} := 0$ for any $j \in \{1, 2, ..., \mu + 1\}$ and $i \in \{1, 2, ..., \eta + 2\}$.

$T$ has many similar properties as $S$, which was introduced in Lemma 37.

**Lemma 38.** $T$ matrix satisfies the following statement:

1. if $T_{j,i} = 1$, then $T_{k,l} = 1$, $\forall k \leq j$, and $l \leq i$, and
2. if $T_{j,i} = 0$, then $T_{k,l} = 0$, $\forall k \geq j$, and $l \geq i$.

Also, for $j \in \{0, 1, ..., \mu - 1\}$ and $i \in \{0, 1, ..., \eta - 1\}$,

- if $t_{bx,j} < t_{br,i} \leq t_{bx,j+1}$, then $t_{j+1} > t_{br,i+1}$;
- if $t_{bx,j} < t_{br,\eta} \leq t_{bx,j+1}$, then $t_{j+1} > t_{\mu+1}$.

For $j \in \{2, 3, ..., \mu\}$ and $i \in \{1, 2, ..., \eta - 1\}$,

- if $t_{br,i} \leq t_j < t_{br,i+1}$, then $t_{bx,j-1} > t_{br,bx,i}$;
- if $t_{br,\eta} \leq t_j < t_{\mu+1}$, then $t_{bx,j} > t_{br,bx,\eta}$.

Therefore, we can get Lemma 39.

**Lemma 39.** For $j \in \{0, 1, ..., \mu - 1\}$ and $i \in \{0, 1, ..., \eta\}$, if $S_{j,i} = 0$, then $T_{j+1,i+1} = 1$. For $j \in \{2, 3, ..., \mu\}$ and $i \in \{1, 2, ..., \eta\}$ if $T_{j,i+1} = 0$, then $S_{j-1,i} = 1$
Again, we assume that $t_i^{br}$ is on the left of $t_j$ geometrically if $t_i^{br} = t_j$, and $t_{\mu + 1}$ is on the left of $t_j$ geometrically if $t_{\mu + 1} = t_j$.

We use the same method as in the branch to calculate the storage cost in Chain 1. Denote $Ibr_i$ as the intersection of $Z_{i-1}^{br}$ and $Z_i^{br}$ for $i \in 1, 2, ..., \eta$. Denote $Ibr_{\eta + 1}$ as the intersection of $Z_{\eta}^{br}$ and $Y_{\mu + 1}^{a}$ at $t_{\mu + 1}$. For $j \in \{1, 2, ..., \mu\}$ and $i \in \{1, 2, ..., \eta\}$, denote $Z_i^{br}$ as the segment line from $(t_{bx,i}, Y_{j}^{a}(t_{bx,i}))$ to $Ibr_i$, if $t_{j + 1}^{br} \leq t_j$, or if $t_{j + 1}^{br} < t_{\mu + 1} \leq t_j$ and $i = \eta + 1$, or if $t_{i}^{br} \leq t_{\mu}$ and $j = \mu$, or if $t_{\mu + 1} \leq t_{\mu}$ and $j = \mu$ and $i = \eta + 1$.

The storage cost can be represented by the triangles in the buildup process figure. Four types of triangles are defined by their left and right edge: $(Z^a, Z^a)$, $(Z^{ibr}, Z^a)$, $(Z^a, Z^{ibr})$ and $(Z^{ibr}, Z^{ibr})$.

**Type $(Z^a, Z^a)$** A triangle of Type $(Z^a, Z^a)$ exists when:

- $t_i^{br} \leq t_{j + 1} < t_j < t_{i + 1}^{br}$, where $i \in \{1, 2, ..., \eta - 1\}$ and $j \in \{2, 3, ..., \mu - 1\}$, or
- $t_i^{br} \leq t_{j + 1} < t_j < t_{\mu + 1}$, where $j \in \{2, 3, ..., \mu - 1\}$, or
- $t_{\mu + 1} \leq t_{j + 1} < t_j$, where $j \in \{1, 2, ..., \mu - 1\}$.

That is $T_{j + 1,i} = 1$ and $T_{j,i + 1} = 0$, where $j \in \{1, 2, ..., \mu - 1\}$ and $i \in \{1, 2, ..., \eta + 1\}$.

The left edge is $Z_{j}^{a}$. The right edge is $Z_{j-1}^{a}$. The top edge is $Z_{j}^{br}$ if $i \leq \eta$, or $Y_{\mu + 1}^{a}$ if $i = \eta + 1$.

The area of this triangle is

$$\frac{1}{2} [c(N_{j}^{a}) - c(N_{j-1}^{a})](t_{j + 1} - t_{bx,i})(t_j - t_{bx,i}).$$

The storage cost represented by this triangle is

$$C^a_{j,i} = \frac{q_x(N_{j}^{a})}{2} [c(N_{j}^{a}) - c(N_{j-1}^{a})](t_{j + 1} - t_{bx,i})(t_j - t_{bx,i}). \quad (9.27)$$

**Type $(Z^{ibr}, Z^a)$** A triangle of Type $(Z^{ibr}, Z^a)$ exists when:

- $t_{j + 1} < t_i^{br} \leq t_j < t_{i + 1}^{br}$, where $i \in \{2, 3, ..., \eta - 1\}$ and $j \in \{2, 3, ..., \mu - 1\}$, or
- $t_i^{br} \leq t_{\mu} < t_{i + 1}^{br}$, where $i \in \{2, 3, ..., \eta - 1\}$, or
- $t_{j + 1} < t_i^{br} \leq t_j < t_{\mu + 1}$, where $j \in \{2, 3, ..., \mu - 1\}$, or
- $t_i^{br} \leq t_{\mu} < t_{\mu + 1}$, or
- $t_{j + 1} < t_{\mu + 1} \leq t_j$, where $j \in \{1, 2, ..., \mu - 1\}$, or
- $t_{\mu + 1} \leq t_{\mu}$.  

That is $T_{j+1,i} = 0$, $T_{j,i} = 1$ and $T_{j,i+1} = 0$ where $i \in \{1, 2, \ldots, \eta + 1\}$ and $j \in \{1, 2, \ldots, \mu\}$.
The left and right edges are $Z_i^{br}$ and $Z_{j-1}^a$. The top edge is $Z_i^{br}$ if $i \leq \eta$, or $Y_{\mu+1}^a$ if $i = \eta + 1$.

If $i \leq \eta$, the area is
\[
\frac{1}{2} \left[ c(N_{j-1}^a) - c(N_i^{br,a}) \right] (t_j - t_i^{br})(t_j - t_{bx,j}).
\]
If $i = \eta + 1$, the area is
\[
\frac{1}{2} c(N_{j-1}^a)(t_j - t_{\mu+1})(t_j - t_{bx,j}).
\]

The storage cost represented by this triangle is
\[
C_{j,i}^{(br,a)} = \frac{q_x(N_j^a)}{2} \left\{ (1 - T_{j,\eta+1})[c(N_j^a) - c(N_i^{br,a})](t_j - t_i^{br}) + T_{j,\eta+1}c(N_{j-1}^a)(t_j - t_{\mu+1}) \right\} (t_j - t_{bx,j}). \tag{9.28}
\]

**Type ($Z^a, Z^{br}$)** A triangle of Type ($Z^a, Z^{br}$) exists when:

- $t_i^{br} \leq t_j < t_i^{br} \leq t_{j-1}$, where $i \in \{2, 3, \ldots, \eta\}$ and $j \in \{2, 3, \ldots, \mu\}$, or
- $t_i^{br} \leq t_j < t_{\mu+1} \leq t_{j-1}$, where $j \in \{2, 3, \ldots, \mu\}$.

That is $T_{j,i-1} = 1$, $T_{j,i} = 0$ and $T_{j-1,i} = 1$ for $i \in \{2, 3, \ldots, \eta + 1\}$ and $j \in \{2, 3, \ldots, \mu\}$.
The three edges are $Z_{j-1}^a$, $Z_i^{br}$, and $Z_i^{br}$.

If $i \leq \eta$, the area is
\[
\frac{1}{2} \left[ c(N_{j-1}^a) - c(N_i^{br,a}) \right] (t_j - t_{bx,j-1})(t_i^{br} - t_j).
\]
If $i = \eta + 1$, the area is
\[
\frac{1}{2} \left[ c(N_{j-1}^a) - c(N_i^{br,a}) \right] (t_j - t_{bx,j-1})(t_{\mu+1} - t_j).
\]

The storage cost represented by this triangle is
\[
C_{j,i}^{(a,br)} = \frac{q_x(N_{j-1}^a)}{2} \left[ c(N_j^a) - c(N_i^{br,a}) \right] (t_j - t_{bx,j-1}) \left[ (1 - T_{j,\eta})(t_i^{br} - t_j) + T_{j,\eta}(t_{\mu+1} - t_j) \right]. \tag{9.29}
\]

**Type ($Z^{br}, Z^{br}$)** A triangle of Type ($Z^{br}, Z^{br}$) exists when:

- $t_j < t_i^{br} \leq t_i^{br} \leq t_{j-1}$, where $i \in \{3, 4, \ldots, \eta\}$ and $j \in \{2, 3, \ldots, \mu\}$, or
- $t_i^{br} \leq t_i^{br} \leq t_{\mu}$, where $i \in \{2, 3, \ldots, \eta\}$, or
- $t_j < t_i^{br} \leq t_{\mu+1} \leq t_{j-1}$, where $j \in \{2, 3, \ldots, \mu\}$, or
- $t_i^{br} \leq t_{\mu+1} \leq t_{\mu}$.
That is \(T_{j,i-1} = 0\) and \(T_{j-1,i} = 1\) for \(i \in \{2, 3, ..., \eta + 1\}\) and \(j \in \{2, 3, ..., \mu + 1\}\).

The three edges are \(Z_{i-1}^{br}, Z_{i}^{br}\), and \(Z_{i}^{br}\). The bottom point is \((t_{bx,j-1}, Z_{j-1}^{a}(t_{bx,j-1}))\).

If \(i \leq \eta\), the area is

\[
\frac{1}{2} \left[ \sum_{k=j-1}^{i-2} c(N_{k}^{a}(t_{bx,k+1} - t_{bx,k}) + c(N_{k}^{a}(t_{1}^{br} - t_{bx,\mu})
+ \sum_{k=1}^{i-2} c(N_{k}^{br,a}(t_{k+1}^{br} - t_{k}^{br}) - c(N_{i-1}^{br,a}(t_{i-1}^{br} - t_{bx,j-1}) \right] (t_{i}^{br} - t_{i-1}^{br}).
\]

If \(i = \eta + 1\), the area is

\[
\frac{1}{2} \left[ \sum_{k=j-1}^{\mu-1} c(N_{k}^{a}(t_{bx,k+1} - t_{bx,k}) + c(N_{\mu}^{a}(t_{1}^{br} - t_{bx,\mu})
+ \sum_{k=1}^{i-2} c(N_{k}^{br,a}(t_{k+1}^{br} - t_{k}^{br}) - c(N_{i-1}^{br,a}(t_{i-1}^{br} - t_{bx,j-1}) \right] (t_{\mu+1}^{br} - t_{i-1}^{br}).
\]

The storage cost represented by this triangle is

\[
C_{j,i}^{(br,br)} = \begin{cases} 
\frac{q_{e}(N_{k}^{a})}{2} \sum_{k=j-1}^{i-2} c(N_{k}^{a}(t_{bx,k+1} - t_{bx,k}) + c(N_{k}^{a}(t_{1}^{br} - t_{bx,\mu})
+ \sum_{k=1}^{i-2} c(N_{k}^{br,a}(t_{k+1}^{br} - t_{k}^{br}) - c(N_{i-1}^{br,a}(t_{i-1}^{br} - t_{bx,j-1}) \right] (t_{i}^{br} - t_{i-1}^{br}), \\
\frac{q_{e}(N_{k}^{a})}{2} \sum_{k=j-1}^{\mu-1} c(N_{k}^{a}(t_{bx,k+1} - t_{bx,k}) + c(N_{\mu}^{a}(t_{1}^{br} - t_{bx,\mu})
+ \sum_{k=1}^{i-2} c(N_{k}^{br,a}(t_{k+1}^{br} - t_{k}^{br}) - c(N_{i-1}^{br,a}(t_{i-1}^{br} - t_{bx,j-1}) \right] (t_{\mu+1}^{br} - t_{i-1}^{br}),
\end{cases}
\]

for \(i \leq \eta\).

\[\tag{9.30}\]

We can check each element and its neighbor elements in \(T\) to determine all the triangles of the storage cost in Chain 1. Denote \(C_{j,i}^{cl}\) as the storage cost of the triangles appearing when checking \(T_{j,i}\). The total storage cost in Chain 1 is

\[
C_{x}^{\mu+1} = \sum_{j=1}^{\mu+1} \sum_{i=1}^{\eta+1} C_{j,i}^{cl}, \tag{9.31}
\]

We use four coefficients: \(a_{j,i}^{(a,a)}\), \(a_{j,i}^{(br,a)}\), \(a_{j,i}^{(a,br)}\) and \(a_{j,i}^{(br,br)}\) to switch on/off the storage costs of different types of triangles in \(C_{j,i}^{cl}\). These coefficients can be either 1 or 0, depending on whether or not the corresponding triangle appears when checking \(T_{j,i}\).

\[
C_{j,i}^{cl} = a_{j,i}^{(a,a)} C_{j,i}^{(a,a)} + a_{j,i}^{(br,a)} C_{j,i}^{(br,a)} + a_{j,i}^{(a,br)} C_{j,i}^{(a,br)} + a_{j,i}^{(br,br)} C_{j,i}^{(br,br)}, \tag{9.32}
\]

The relationship of \(T\) matrix, the types of triangles and coefficients in \(C_{j,i}^{cl}\) can be represented by Table [9.2].

142
Then, for \( i \in \{1, 2, ..., \eta + 1\} \) and \( j \in \{1, 2, ..., \mu + 1\} \), the coefficients in \( C^i_j \) can be formulated as:

\[
a_{j,i}^{(a,br)} = \begin{cases} (1 - T_{j,i})T_{j,i-1}T_{j-1,i}, & \text{if } 1 < i \leq \eta + 1 \text{ and } 1 < j \leq \mu + 1; \\
0, & \text{if } i = 1 \text{ or } j = 1. 
\end{cases} \quad (9.33)
\]

\[
a_{j,i}^{(br,br)} = \begin{cases} (1 - T_{j,i})(1 - T_{j,i-1})T_{j-1,i}, & \text{if } 1 < i \leq \eta + 1 \text{ and } 1 < j \leq \mu + 1; \\
0, & \text{if } i = 1 \text{ or } j = 1. 
\end{cases} \quad (9.34)
\]

\[
a_{j,i}^{(a,a)} = \begin{cases} T_{j,i}T_{j+1,i}(1 - T_{j,i+1}), & \text{if } 1 \leq j < \mu + 1; \\
0, & \text{if } j = \mu + 1. 
\end{cases} \quad (9.35)
\]

\[
a_{j,i}^{(br,a)} = \begin{cases} T_{j,i}(1 - T_{j,i+1})(1 - T_{j,i+1}), & \text{if } 1 \leq j < \mu + 1; \\
0, & \text{if } j = \mu + 1. 
\end{cases} \quad (9.36)
\]

For \( j \in \{1, 2, ..., \mu\} \), \( t_j \) can be represented as a function of \( \{t_{bx,1}, t_{bx,2}, ..., t_{bx,L}\} \cup \{t_{i,br}^1, t_{2,br}^1, ..., t_{\eta,br}^1\} \).

For \( j \in \{2, 3, ..., \mu\} \), if \( t_{i,br}^1 \leq t_j < t_{i+1,br}^1 \) where \( i \in \{1, 2, ..., \eta - 1\} \), or if \( t_{\eta,br}^1 \leq t_j < t_{\mu+1,br}^1 \), then

\[
(t_j - t_{bx,j-1})c(N^a_{j-1}) = \sum_{k=j-1}^{\mu-1} (t_{bx,k+1} - t_{bx,k})c(N^a_k) + (t_{1,br}^1 - t_{bx,\mu})c(N^a_{\mu})
\]

\[
+ \sum_{k=1}^{i-1} (t_{br,k+1}^1 - t_{br,k}^1)c(N^{br,a}_k) + (t_j - t_i^1)c(N^{br,a}_i).
\]

\[
t_j = \frac{1}{c(N^a_{j-1}) - c(N^{br,a}_i)} \left[ t_{bx,j-1}c(N^a_{j-1}) + \sum_{k=j-1}^{\mu-1} (t_{bx,k+1} - t_{bx,k})c(N^a_k)
\right.
\]

\[
+ (t_{1,br}^1 - t_{bx,\mu})c(N^a_{\mu}) + \sum_{k=1}^{i-1} (t_{br,k+1}^1 - t_{br,k}^1)c(N^{br,a}_k) - t_i^1c(N^{br,a}_i) \right].
\]

If \( t_{\mu+1} \leq t_j \), where \( j \in \{1, 2, ..., \mu\} \), then

\[
(t_j - t_{bx,j-1})c(N^a_{j-1}) = \sum_{k=j-1}^{\mu-1} (t_{bx,k+1} - t_{bx,k})c(N^a_k) + (t_{1,br}^1 - t_{bx,\mu})c(N^a_{\mu})
\]

\[
+ \sum_{k=1}^{\eta-1} (t_{br,k+1}^1 - t_{br,k}^1)c(N^{br,a}_k) + (t_{\mu+1} - t_\eta^1)c(N^{br,a}_\eta).
\]
Consider the value of \( T_{j,i}, T_{j,i+1} \) and \( T_{j,\eta+1} \). In general, for \( j \in \{1, 2, \ldots, \mu\}, \)

\[
    t_j = \frac{1}{c(N^a_{j-1})} \left[ t_{bx,j-1} c(N^a_{j-1}) + \sum_{k=j-1}^{\mu-1} (t_{bx,k+1} - t_{bx,k}) c(N^a_k) \right. \\
            \left. + (t^br_1 - t_{bx,\mu}) c(N^a_\mu) + \sum_{k=1}^{\eta-1} (t^br_{k+1} - t^br_k) c(N^br,a) - (t_{\mu+1} - t^br_\eta) c(N^br,a) \right]. 
\]

(9.37)

The Quadratic Program

After formulating the costs, we notice that the optimization problem is formulated as a quadratic program of variables \( \{t_{bx,1}, t_{bx,2}, \ldots, t_{bx,L}, t^br_1, t^br_2, \ldots, t^br_\eta\} \). Any of \( \{t^br_1, t^br_2, ..., t^br_\eta, t_{1}, t_{2}, ..., t_{L}\} \) can be expressed as a function of \( \{t_{bx,1}, t_{bx,2}, \ldots, t_{bx,L}, t^br_1, t^br_2, \ldots, t^br_\eta\} \) by Equations (9.22), (9.23), (9.25) and (9.37). With a given pair of \( S \) and \( T \) matrices which satisfies Lemmas 37, 38 and 39, the objective function is:

\[
    C_x = C_x^br + C_x^{L-\mu+2} + C_x^{\mu+1} + C_x^{\mu-1}. 
\]

Constraints are:

1. \( t_{dis} - \tau_f \leq t_{bx,j} \leq t_{bx,j+1}, \forall j \in \{1, 2, \ldots, L\} \). Notice that \( t_{bx,L+1} = t_{dis} \).
2. \( t_{bx,\mu+1} \leq t^br_i \leq t^br_{i+1} \leq t_{\mu+1}, \forall i \in \{1, 2, \ldots, \eta - 1\} \).
3. \( t_{dis} - \tau_f \leq t^br_{bx,1} \).
4. \( \sum_{i=1}^{L}[c(N^a_i) - \delta](t_{bx,i+1} - t_{bx,i}) = \min\{\tau_{dis} \delta, \tau_f[c(N^a_L) - \delta]\} \).
5. \( S_{j,i}(t^br_{bx,i} - t_{bx,j}) + (1 - S_{j,i})(t_{bx,j} - t^br_{bx,i}) \leq 0, \forall j \in \{1, 2, \ldots, \mu\}, \) and \( i \in \{1, 2, \ldots, \eta\} \).
6. \( T_{j,i}(t_j - t^br_i) + (1 - T_{j,i})(t^br_i - t_j) \leq 0, \forall j \in \{1, 2, \ldots, \mu\}, \) and \( i \in \{1, 2, \ldots, \eta\} \).
7. \( T_{j,\eta+1}(t_j - t_{\mu+1}) + (1 - T_{j,\eta+1})(t_{\mu+1} - t_j), \forall j \in \{1, 2, \ldots, \mu\} \).
9.4 Conclusion

We analyze a special type of assembly tree network system in this chapter. We discuss different cases based on the allocations of the capacities and unit storage costs of nodes. In our analysis, two cases can be solved by applying the results of serial network systems. Another one is analyzed in detail. The optimal operation is derived based on the given time variables. With these time variables, the optimization problem is formulated to quadratic programs. Considering the interleaving order of the time variables, we still solve multiple quadratic programs, and determine the real optimal solution based on comparison among the results of quadratic programs.

Notice that there are still many cases of other allocations of the capacities and unit storage costs of nodes which is not covered. They can be a topic of future study. Some discussion is given in Subsection D.1.1.
Table 9.2: $T$ matrix, the types of triangles, and coefficients in $C_{j,i}$

<table>
<thead>
<tr>
<th>$T_{j,i}$</th>
<th>$T_{j,i-1}$</th>
<th>$T_{j+1,i}$</th>
<th>$T_{j-1,i}$</th>
<th>$T_{j,i+1}$</th>
<th>triangle type</th>
<th>$a_{j,i}^{(a,Ibr)}$</th>
<th>$a_{j,i}^{(a,a)}$</th>
<th>$a_{j,i}^{(Ibr,Ibr)}$</th>
<th>$a_{j,i}^{(Ibr,a)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>either</td>
<td>1</td>
<td>either</td>
<td>$(Z^a, Z^{Ibr})$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>either</td>
<td>1</td>
<td>either</td>
<td>$(Z^{Ibr}, Z^{Ibr})$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>either</td>
<td>either</td>
<td>0</td>
<td>either</td>
<td>none</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>either</td>
<td>either</td>
<td>1</td>
<td>none</td>
<td>$(Z^a, Z^a)$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>either</td>
<td>1</td>
<td>either</td>
<td>0</td>
<td>$(Z^{Ibr}, Z^a)$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>either</td>
<td>0</td>
<td>either</td>
<td>0</td>
<td>$(Z^{Ibr}, Z^a)$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Chapter 10

Conclusion

10.1 Conclusion of the Current Study

The research of resilient enterprises is receiving growing attention nowadays. Resiliency is a critical factor for the manufacturing and supply chain systems to deal with disruptions. Resilient enterprises have good abilities to prepare for and recover from the disruptions. Such abilities can eliminate the negative effects of disruptions to the minimum level, so that economic benefits can be achieved.

There are many studies on resilience on different areas, such as supply chains, computer networks and infrastructures. Most of these studies are based on case studies or focused on conceptual and organizational methods for resilience. Some other studies are focused on specific problems and the methods are not easily applied to more general systems. There is a lack of quantitative research on resilient enterprises by addressing the redundancy, recovery, response and control policies of general systems.

This study intends to develop the systematic methods to solve the quantitative, general and dynamic issues of resilience. We first built mathematical models for general manufacturing enterprises. We modeled the network structures of systems, operations, resources, capacities, disruptions, costs, demands and constraints. Models were set up for discrete time domain and continuous time domain separately. For different time domains, some assumptions are different, such as for the disruption prediction and capacity constraint. With the models, solving the resilient operation was formulated into an optimization problem, of which the objective functions are lost demand and total costs.

Then, we defined several structures and systems based on the specific network structures. They are chains, assembly tree structures, serial structures, AND structures, OR Structures, serial network systems and assembly tree network systems. Some properties were discussed about these structures and systems. These structures and systems are some basic blocks building the general networks and systems. The analysis of them leads to the properties and optimal operations of more general networks and systems.

After definitions of these structures and systems, we introduced the approaches of normalization and aggregation. These two approaches simplify the analysis of general systems. The normalization makes all the parameters of nodes associated with demands. Such parameters include capacities, unit storage costs, operation rates, inventory amounts, operation vectors,
and so on. The aggregation converts the complicated systems into simple ones, of which
the optimal operations can be mapped back to the original systems by disaggregation.

With these approaches, we executed the studies in discrete time domain. These studies
cover the issues of serial network systems and assembly tree network systems. In discrete
time domain, we assumed the capacity constraint is loose. It is found that storage will
be built up in a serial network system in the cycle right before disruption. The buildup
appears at only one node, which has the lowest normalized unit storage cost downstream of
the disrupted node. The amount of buildup is determined by the predicted duration of the
disruption. For the assembly tree network system, it is found that it can be aggregated into
a serial network system, based on the disrupted chain. Therefore, the optimal operation of
a assembly tree network system can be obtained from that of the aggregated serial network
system.

After the analysis in discrete time domain, we focused on the issues in continuous time do-
main. We started the analysis with a special system, Decreasing-Storage-Cost-Decreasing-
Capacity (DSCDC) network system. We derived the optimal operations of DSCDC network
systems based on \( x(t_{dis}) \), the inventory distribution at the time when the disruption hap-
pens. The optimal operations and total costs were formulated as functions of \( x(t_{dis}) \). And
the optimization problem of DSCDC becomes a quadratic program, to determine the optimal
\( x(t_{dis}) \) which achieves the minimum total costs. Then we introduced the approach to
aggregate general serial network systems into DSCDC network systems, with an assumption
that no buildup is allowed upstream of the disrupted node. It is found that the storage
can be built up in some certain nodes, which are denoted as \( s \)-nodes. The operation of
each node is limited by the capacity of certain ”bottleneck” nodes, which are denoted as
\( c \)-nodes. Therefore, we developed the methods to solve the optimal operations of general
serial network systems. Finally, a case study was given to illustrate the method.

Next, we extended the study to cover the case that buildup is allowed upstream of the
disrupted node. Many results still work, such as determining the \( s \)- and \( c \)- nodes. We
changed the variables from \( x(t_{dis}) \) to \( \{t_{bx}\} \), which are the starting times of the buildup of
each products. The operations were formulated based on \( \{t_{bx}\} \). Among \( \{t_{bx}\} \), the times
of buildups of upstream and downstream of the disrupted node are interleaving. With
different interleaving order, the objective function of total costs is different. We defined \( S \)
matrix to represent each interleaving order and formulated the total costs based on \( \{t_{bx}\} \).
With a given \( S \) matrix, the optimization problem was formulated into a quadratic program.
Then, instead of solving a single quadratic program, we solved a quadratic program for each
different \( S \), and then compared the results to determine the real optimal solution. A case
study was also given, in which the practical system is the same as the one in the previous
chapter.

After the analysis of serial network systems, we studied a special type of assembly tree
network systems consisting of a disrupted chain and a branch. Since the issue of assembly
tree network systems is quite complicated, we only studied several cases of capacities and
unit storage costs allocation. The approach used is similar to that of serial network systems.
We derived the operation based on time variables. And the optimization problems were
formulated as quadratic problems, with a given interleaving order of time variables. We
compared the results of quadratic problems of all the possible interleavings to determine
the final optimal solution.

This study developed a mathematical framework capable to model general manufacturing
systems and address the dynamic operation of resilience of them. A number of specific network systems were analyzed with this framework, and their optimal operations to deal with disruptions were obtained. These results can provide useful information to improve the resilience in practical systems.

10.2 Future Directions

Most of the previous studies in the literature are focused on the basic concepts and qualitative analysis of system resilience. The approaches and policies introduced to enhance resilience are mainly about static planning issues, such as setting up redundancy. Therefore, we focused our research on quantitative analysis and real-time control issue, which are significant for finding the inner property of resilience. In previous sections, we introduced the framework of our model, and obtained some results of different types of systems. There are still many problems to be addressed.

1. Studies on Specific Alternative Systems and Networks

As we have studied the optimal control of serial and assembly tree network systems, there is still work to be done for more kinds of systems, such as OR nodes network systems.

2. Studies on General Approach to the Optimal Operation

- **Node conversion**
  We analyzed some simple networks, such as nodes with fixed operation ratio. For networks which are more complicated, some nodes may be replaced by a sub-network which consists of only the nodes of basic networks we analyzed. This is like a reverse action of aggregation. The problem is to convert a number of different types of nodes to the combination of the basic nodes networks.

- **Removing Certain Assumptions**
  We introduced several assumptions to reduce the complexity of the system and analysis process, such as no consideration of changeover cost and changeover constraints, highest priority to reduce loss demand cost, etc. We need to extend the current analysis to cover the cases when certain assumptions are removed. We need to figure out the approach to the optimal operation if certain assumptions are removed.

- **Adding Constraint of Limit of Inventory Storage**
  Practical systems may also have a constraint on inventory storage. There could be a limit on the maximum amount of storage allowed. We can add such a constraint to the optimization problem. For the numerical studies, it may be quite easy to solve with the new added constraint. However, for the analytical studies, we already know that this constraint can change the operation policy a lot. Deeper investigation is needed.

- **Potential Disrupted Nodes**
  In this thesis we assumed that the disruptions can be predicted. For future research, considering potential disruptions, the problem becomes a stochastic issue.
We need to include the probabilities of disruptions in our problem formulation. Also, the operation policy needs to consider such probabilities.

3. Sensitivity Analysis of System Parameters Related to Resilience

For future research, it is valuable to analyze the effects of system parameters on the response to the disruptions. Sensitivity analysis can provide much information on how the parameters, structures, and characteristics of the system can affect resilience. Such research can include analysis of closed-loop system characteristics, which means with some kind of given controller (such as linear program) we analyze the system performance under disruption.

4. Studies on General Properties of Control

It is important to conclude our results of control law to some general control policies. Such policies can offer the intuitive guidance to practical systems to achieve better resilience. They are intended to make it possible that even without mathematical calculation, the research results can still be applied to make improvement on resilience.

- Decentralized Decision Policies based on Local Information

Based on our current results, the optimal operation of any node can be represented by three modes, where the operation rate is either determined by capacity, the upstream node or the downstream node. Thus, in each mode, the operation of any node only relates to its self or its neighbor nodes. Therefore, a distributed policy based on local information may be developed.

In Appendix D we examine some of these in more depth.
Appendix A

Derivation of Solution to the Problem of Simple System in Chapter 3.1.4

A.1 Case 1

According to the Demand Constraint $D_u(k) \leq d(k)$,

\[ \Rightarrow d(k) - D_u(k) \geq 0, \forall k. \]

\[ \Rightarrow \sum_{k' = k - \kappa_p}^{k + \kappa_f} \{ a_d [d(k') - D_u(k')] \} \geq 0. \]

In order to make the above cost minimum, it is required that

\[ D_u(k) = d(k), \forall k. \]

\[ \Rightarrow u_0(k) = d. \]  \hspace{1cm} (A.1)

We consider the period when $k + \kappa_f < k_{dis}$. In this period, $c(k)$ is not changed by disruption. The problem can be reformulated as:

Minimize : $\sum_{k' = k - \kappa_p}^{k + \kappa_f} \{ a_u q_u u(k') + a_x q_x x(k') \}$,

Subject to : $x(k) \geq 0$,

$u(k) \geq 0$,

$u_0(k) = d$,

$x(k + 1) = x(k) + Bu(k)$,

$x(0) = 0$,

$\kappa_f \geq 1$,

$k > \kappa_p$,

$k + \kappa_f < k_{dis}$,

$k, \kappa_p, \kappa_f \in \mathbb{N}$. 
Assume $i$ where $u$ Also, because $u^k < u^k$ 

Given this, and given that $x^k = x^k(0) = 0$; if $u^k = [d d d d]^T$ for any $k < k_1$, then $x^k(1) = x^k(0) + B \sum_{k=0}^{k_1-1} u^k(k) = 0$. Thus, $x^k(1) = 0$.

$$x^k(1) = x^k(1) + Bu^k(1) = 0 + \begin{bmatrix} -1 & 1 & 0 & 0 \\ 0 & -m_2 & m_2 & 0 \\ 0 & 0 & -m_3 & m_3 \end{bmatrix} \begin{bmatrix} u^0(1) \\ u^1(1) \\ u^2(1) \\ u^3(1) \end{bmatrix}$$

$$= \begin{bmatrix} u^1(1) - u^0(1) \\ m_2[u^2(1) - u^1(1)] \\ m_3[u^3(1) - u^2(1)] \end{bmatrix}.$$

Given this, and given that $x^k(1) \geq 0$ from the constraints, then we have

$$\begin{bmatrix} u^1(1) - u^0(1) \\ m_2[u^2(1) - u^1(1)] \\ m_3[u^3(1) - u^2(1)] \end{bmatrix} \geq 0.$$

$$\Rightarrow u^3(1) \geq u^2(1) \geq u^1(1) \geq u^0(1) = d. \quad \text{(A.2)}$$

Since $u^k \neq [d d d d]^T$, then

$$u^1(1) - u^0(1) > 0 \text{ or } u^2(1) - u^1(1) > 0 \text{ or } u^3(1) - u^2(1) > 0.$$

Assume $i$ is the index such that $u^i(k) - u^i_{i-1}(k) > 0$. Define a vector

$$\Delta u^k(1) = \begin{bmatrix} \Delta u^0(1) \\ \Delta u^1(1) \\ \Delta u^2(1) \\ \Delta u^3(1) \end{bmatrix}.$$

where

$$\Delta u^k(1) = \begin{cases} u^i(1) - u^i_{i-1}(1), & \text{when } i \leq j \leq 3; \\ 0, & \text{when } 0 \leq j \leq i - 1. \end{cases}$$

$$\Delta u^1(1) \geq 0, \Delta u^2(1) \geq \Delta u^3(1) \geq \Delta u^4(1) \geq \Delta u^0(1).$$

Also, because $u^i(k) \geq u^i_{i-1}(k)$ when $j \geq i$, then

$$u^j(1) - \Delta u^j(1) = \begin{cases} u^j(1) - u^j_{i-1}(1) \geq u^j_{i-1}(1) \geq 0, & \text{when } i \leq j \leq 3; \\ u^j(1) - 0 \geq 0, & \text{when } 0 \leq j \leq i - 1. \end{cases}$$

$$\Delta u^j(1) \geq 0, \Delta u^3(1) \geq \Delta u^2(1) \geq \Delta u^1(1) \geq \Delta u^0(1).$$

Then we can select another control signal $u''$ such that:

$$u''(1) = \begin{cases} u^i(1), & \text{when } k \neq k_1 \text{ and } k \neq k_1 + 1; \\ u^i(1) - \Delta u^i(1), & \text{when } k = k_1; \\ u^i(1) + \Delta u^i(1), & \text{when } k = k_1 + 1. \end{cases}$$
Then,
\[ x''(k) = \begin{cases} 
  x'(k), & \text{when } k \neq k_1 + 1; \\
  x'(k) - B\Delta u'(k_1), & \text{when } k = k_1 + 1.
\end{cases} \]

Since
\[ x''(k_1 + 1) = x''(k_1) + Bu''(k_1) = x''(k_1) + B[u'(k_1) - \Delta u'(k_1)] \geq 0, \]
\[ u'' \text{ and } x'' \text{ still meet the constraints.} \]
then, comparing the objective functions, we can find:
\[
\sum_{k'=k_1-\kappa_p}^{k_1+\kappa_f} \{a_u q_u u''(k') + a_x q_x x''(k')\} < \sum_{k'=k_1-\kappa_p}^{k_1+\kappa_f} \{a_u q_u u'(k') + a_x q_x x'(k')\}.
\]

which means \( u' \) is not the optimal solution. This is a contradiction. So, the optimal control in nominal stage is \( u(k) = [d \ d \ d \ d]^T \).

### A.2 Case 4a

Now we consider the effect of a disruption at node 2, responded by the capacity of \( c_2(\cdot) \) of resource 2 going to zero. (By our definition of \( R_2 \), node 2 relies on resource 2.)

The length of the disruption that can be predicted is denoted as \( \kappa'_{\text{dis}} \). Its value equals to the smaller one between \( \kappa_f \) and the real length of the disruption. As node 2 can not work during the disruption, node 0 can still work only when there is enough inventory built up for node 0. As we try to keep node 0 working during the disruption to make no loss demand, it is needed to build up enough inventory for node 0 before disruption \( k < k_{\text{dis}} \). As the \( \kappa'_{\text{dis}} \) is the length of disruption we can predict before disruption, we are always able to prepare and build up required inventory before disruption to support node 0 to work for at least the period of \( \kappa'_{\text{dis}} \). So, \( u_0(k) \) is able to be \( d \) when \( k \leq k_{\text{dis}} + \kappa'_{\text{dis}} - 1 \). In this case, considering to minimize the lost demand, Equation (A.1) still holds when \( k \leq k_{\text{dis}} + \kappa'_{\text{dis}} - 1 \). The problem can be reformulated as:
Minimize: \[ \sum_{k'=k-\kappa_p}^{k+\kappa_f} \{a_u u(k') + a_x x(k')\}, \]

Subject to: \[ x(k) \geq 0, \]
\[ u(k) \geq 0, \]
\[ u_0(k) = d, \]
\[ R_2 u(k) = 0, \]
\[ x(k + 1) = x(k) + B u(k), \]
\[ \kappa_f \geq 1, \]
\[ k > \kappa_p, \]
\[ k_{dis} \leq k \leq k_{dis} + \kappa'_{dis} - 1, \]
\[ k, \kappa_p, \kappa_f \in \mathbb{N}. \]

First, \( N_3 \) and \( N_2 \) will stop working during disruption (when \( k_{dis} \leq k \leq k_{dis} + \kappa'_{dis} - 1 \)). According to \( R_2 u(k) = 0 \), we know \( u_2(k) = 0 \).

In the optimal solution, \( u_3(k) = 0 \) when \( k_{dis} \leq k \leq k_{dis} + \kappa'_{dis} - 1 \). To show this by contradiction, we use \( u' \) to denote the optimal control with at least one time point \( k_2 \in [k_{dis}, k_{dis} + \kappa'_{dis} - 1] \) such that \( u'_3(k_2) \neq 0 \). So, \( u'_3(k_2) > 0 \). Then, we can choose another control signal \( u'' \) such that:
\[
 u''(k) = \begin{cases} 
 u'(k), & \text{when } k \neq k_2 \text{ and } k \neq k_2 + 1; \\
 u'(k) - u'_3(k_2)[0 \ 0 \ 0 \ 1]^T, & \text{when } k = k_2; \\
 u'(k) + u'_3(k_2)[0 \ 0 \ 0 \ 1]^T, & \text{when } k = k_2 + 1.
\end{cases}
\]

This in effect makes the first elements of \( u'_3(k_2) \) equal to zero.
\[
 u''(k_2) = u'(k_2) - u'_3(k_2)[0 \ 0 \ 0 \ 1]^T \geq 0,
\]
\[
 \Rightarrow u''(k) \geq 0,
\]

which is necessary for \( u'' \) to be an allowable control. From Equation \[[3.1]\], it follows that
\[
 x''(k) = \begin{cases} 
 x'(k), & \text{when } k \neq k_2 + 1; \\
 x'(k) - B u'_3(k_2)[0 \ 0 \ 0 \ 1]^T, & \text{when } k = k_2 + 1.
\end{cases}
\]

Again, from Equation \[[3.1]\], and the fact that \( u'_3(k_2) = u'_2(k_2) = 0 \), we have
\[
 x''(k_2 + 1) = x''(k_2) + B u''(k_2) = \begin{bmatrix} x'_1(k_2 + 1) \\
 x'_2(k_2 + 1) \\
 x'_3(k_2 + 1) \\
 x''_3(k_2) \end{bmatrix} \geq 0.
\]

This is also a necessary condition for \( u'' \) to be an allowable control, since
\[
 x''(k) \geq 0.
\]

Recall from the above, we have
\[
 x''(k_2 + 1) = x'(k_2 + 1) - B u'_3(k_2)[0 \ 0 \ 0 \ 1]^T < x'(k_2 + 1).
\]
Comparing the objective functions, we can find:

\[
\sum_{k' = k_2 - \kappa_p}^{k_2 + \kappa_f} \{ a_u q_u u''(k') + a_x q_x x''(k') \} < \sum_{k' = k_2 - \kappa_p}^{k_3 + \kappa_f} \{ a_u q_u u'(k') + a_x q_x x'(k') \},
\]

which means \( u' \) is not the optimal solution. This is a contradiction. So, \( u_3(k) = 0 \) when \( k_{dis} \leq k \leq k_{dis} + \kappa'_{dis} - 1 \).

We have determined \( u \) at nodes \( N_3, N_2, \) and \( N_0 \). Now we want to consider \( u \) at node \( N_1 \). Considering our definition of \( B \) for this example, we need to consider two cases: \( q_x(N_2)m_2 > q_x(N_1) \) or \( q_x(N_2)m_2 \leq q_x(N_1) \). When \( q_x(N_2)m_2 > q_x(N_1) \), we can prove \( u_1(k) = 0 \), when \( k_{dis} \leq k \leq k_{dis} + \kappa'_{dis} - 1 \). Again, we show it by contradiction. We denote the optimal control as \( u' \), in which there exists a time point \( k_3 \) such that \( u'_1(k_3) > 0 \). Then we can select another control signal \( u'' \) such that:

\[
u''(k) = \begin{cases} u'(k), & \text{when } k \neq k_3 - 1 \text{ and } k \neq k_3; \\ u'(k) + u'_1(k_3)[0 \ 1 \ 0 \ 0]^T, & \text{when } k = k_3 - 1; \\ u'(k) - u'_1(k_3)[0 \ 1 \ 0 \ 0]^T, & \text{when } k = k_3. \end{cases}
\]

Then,

\[
x''(k) = \begin{cases} x'(k), & \text{when } k \neq k_3; \\ x'(k) + B u'_1(k_3)[0 \ 1 \ 0 \ 0]^T = x'(k) + u'_1(k_3)[1 \ - m_2 \ 0 ]^T, & \text{when } k = k_3. \end{cases}
\]

\( u'' \) and \( x'' \) still meet the constraints. \( q_x x''(k_3 + 1) < q_x x'(k_3 + 1) \). Comparing the objective functions, we can find:

\[
\sum_{k' = k_3 - \kappa_p}^{k_3 + \kappa_f} \{ a_u q_u u''(k') + a_x q_x x''(k') \} < \sum_{k' = k_3 - \kappa_p}^{k_3 + \kappa_f} \{ a_u q_u u'(k') + a_x q_x x'(k') \},
\]

which means \( u' \) is not the optimal solution. This is a contradiction. So, \( u_1(k) = 0 \), when \( k_{dis} \leq k \leq k_{dis} + \kappa'_{dis} - 1 \). So, the optimal control \( u = [d \ 0 \ 0 \ 0]^T \) when \( k_{dis} \leq k \leq k_{dis} + \kappa'_{dis} - 1 \).

### A.3 Case 2

Then, we can find that during the period \( k_{dis} \leq k \leq k_{dis} + \kappa'_{dis} - 1 \), the total amount of product 1 used by \( N_0 \) is \( d \kappa_{dis} \). So, \( x_1(k_{dis}) \geq d \kappa_{dis} \). The problem when \( k_{dis} - \kappa_f + 1 \leq k \leq \)
$k_{\text{dis}} - 1$ can be formulated as:

\[
\text{Minimize : } \sum_{k'=k-\kappa_p}^{k+\kappa_f} \{ a_u q_u u(k') + a_x q_x x(k') \},
\]

Subject to:

\[
\begin{align*}
& x(k) \geq 0, \\
& u(k) \geq 0, \\
& u_0(k) = d, \\
& u(k') = [d \ 0 \ 0]^T, \text{ when } k' \geq k_{\text{dis}}, \\
& x(k + 1) = x(k) + Bu(k), \\
& x(0) = 0, \\
& x_1(K) \geq d\kappa_{\text{dis}}, \\
& \kappa_f \geq 1, \\
& k > \kappa_p, \\
& k_{\text{dis}} - \kappa_f + 1 \leq k \leq k_{\text{dis}} - 1, \\
& k, \kappa_p, \kappa_f \in \mathbb{N}. 
\end{align*}
\]

Then we can prove with optimal control $u(k) = [d \ d \ d \ d]^T$ when $k_{\text{dis}} - \kappa_f + 1 \leq k \leq k_{\text{dis}} - 2$. Again, we show that by contradiction. We denote the optimal control as $u'$ such that $u'(k) \neq u(k)$ for some values of $k$. There exist smallest $k_4 \in [k_{\text{dis}} - \kappa_f + 1, k_{\text{dis}} - 2]$ such that $u'(k_4) \neq [d \ d \ d \ d]^T$. Since $k_4$ is the smallest, then $x(k_4) = 0$.

\[
\begin{align*}
x'(k_4 + 1) &= x'(k_4) + Bu'(k_4) = \begin{bmatrix}
u'_1(k_4) - u'_0(k_4) \\
m_2[u'_2(k_4) - u'_1(k_4)] \\
m_3[u'_3(k_4) - u'_2(k_4)]
\end{bmatrix}.
\end{align*}
\]

\[
x'(k_4 + 1) \geq 0 \\
\Rightarrow \begin{bmatrix}
u'_1(k_4) - u'_0(k_4) \\
m_2[u'_2(k_4) - u'_1(k_4)] \\
m_3[u'_3(k_4) - u'_2(k_4)]
\end{bmatrix} \geq 0.
\]

\[
\Rightarrow u'_3(k_4) \geq u'_2(k_4) \geq u'_1(k_4) \geq u'_0(k_4) = d. \quad (A.3)
\]

Since

\[
u'(k_4) \neq [d \ d \ d \ d]^T,
\]

\[
\Rightarrow u'_1(k_4) - u'_0(k_4) > 0 \text{ or } u'_2(k_4) - u'_1(k_4) > 0 \text{ or } u'_3(k_4) - u'_2(k_4) > 0.
\]

Assume $i$ is the index such that $u'_i(k_4) - u'_{i-1}(k_4) > 0$. Define a vector

\[
\Delta u'(k_4) = \begin{bmatrix}
\Delta u'_0(k_4) \\
\Delta u'_1(k_4) \\
\Delta u'_2(k_4) \\
\Delta u'_3(k_4)
\end{bmatrix}.
\]

where

\[
\begin{align*}
\Delta u'_j(k_4) &= \begin{cases}
u'_j(k_4) - u'_{j-1}(k_4), & \text{when } i \leq j \leq 3; \\
0, & \text{when } 0 \leq j \leq i - 1.
\end{cases}
\end{align*}
\]
\[ \Delta u'(k_4) \geq 0, \ \Delta u_2'(k_4) \geq \Delta u_2'(k_4) \geq \Delta u_1'(k_4) \geq \Delta u_0'(k_4). \]

Also,

\[ u'_j(k_4) - \Delta u'_j(k_4) = \begin{cases} u'_j(k_4) - u'_j(k_4) + u'_{i-1}(k_4) \geq u'_{i-1}(k_4) \geq 0, & \text{when } i \leq j \leq 3; \\ u'_j(k_4) - 0 \geq 0, & \text{when } 0 \leq j \leq i - 1. \end{cases} \]

\[ u'(k_4) - \Delta u'(k_4) \geq 0, \ u'_3(k_4) - \Delta u'_3(k_4) \geq u'_2(k_4) - \Delta u'_2(k_4) \geq u'_1(k_4) - \Delta u'_1(k_4) \geq 0. \]

Then we can select another control signal \( u'' \) such that:

\[ u''(k) = \begin{cases} u'(k), & \text{when } k \neq k_4 \text{ and } k \neq k_4 + 1; \\ u'(k) - \Delta u'(k_4), & \text{when } k = k_4; \\ u'(k) + \Delta u'(k_4), & \text{when } k = k_4 + 1. \end{cases} \]

Then,

\[ x''(k) = \begin{cases} x'(k), & \text{when } k \neq k_4 + 1; \\ x'(k) - B\Delta u'(k_4), & \text{when } k = k_4 + 1. \end{cases} \]

Since

\[ x''(k_4 + 1) = x''(k_4) + B u''(k_4) = x''(k_4) + B [u'(k_4) - \Delta u'(k_4)] \geq 0, \]

\( u'' \) and \( x'' \) still meet the constraints. Since

\[ x''(k_4 + 1) = x'(k_4 + 1) - B\Delta u'(k_4) < x'(k_4 + 1), \]

comparing the objective functions, we can find:

\[
\sum_{k' = k_4 - \kappa_f}^{k_4 + \kappa_f} \{a_u q_u u''(k') + a_x q_x x''(k')\} < \sum_{k' = k_4 - \kappa_f}^{k_4 + \kappa_f} \{a_u q_u u'(k') + a_x q_x x'(k')\}.
\]

which means \( u' \) is not the optimal solution. This is a contradiction. So, in the optimal control \( u(k) = [d \ d \ d \ d]^T \) when \( k_{dis} - \kappa_f + 1 \leq k \leq k_{dis} - 2. \)

**A.4 Case 3a**

\[ x(k_{dis} - 1) = 0. \]

And we have a constraint that \( x_1(k_{dis}) \geq d\kappa_{dis} \). So the smallest \( u(k_{dis} - 1) = [d \ d + d\kappa'_{dis} \ d + d\kappa'_{dis} \ d + d\kappa'_{dis}]. \)
Appendix B

Proof of Associativity

Proof of Theorem \[4\]. It can be simply proved that if \( P(N^a) = Agg(P(N_1), P(N_2),..., P(N_n)) \), then \( \alpha P(N^a) = Agg(P(N_1), P(N_2),..., P(N_n)) \), where \( \alpha \) is a positive scalar constant.

We denote:

\[
P(N^a) = Agg(P(N_1), P(N_2),..., P(N_n)),
P(N_1^a) = Agg(P(N_1), P(N_2),..., P(N_{n_1})),
P(N_2^a) = Agg(P(N_{n_1}+1), P(N_{n_1}+2),..., P(N_{n_2})),
\ldots
\]

\[
P(N_g^a) = Agg(P(N_{n_g-1}+1), P(N_{n_g-1}+2),..., P(N_n)),
P(N^{a,*}) = Agg(P(N_1^a), P(N_2^a),..., P(N_g^a))
\]

Based on Aggregation Property, we have:

\[
P(N^a)u^a = [P(N_1) \ P(N_2) ... \ P(N_n)] \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix},
\]

\[
P(N_1^a)u_1^a = [P(N_1) \ P(N_2) ... \ P(N_{n_1})] \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{n_1} \end{bmatrix},
\]

\[
P(N_2^a)u_2^a = [P(N_{n_1}+1) \ P(N_{n_1}+2) ... \ P(N_{n_2})] \begin{bmatrix} u_{n_1+1} \\ u_{n_1+2} \\ \vdots \\ u_{n_2} \end{bmatrix},
\]

\ldots
\[ P(N_a) u_a^a = [P(N_{n_1}) P(N_{n_2}) \ldots P(N_n)] \begin{bmatrix} u_{n_1} \\ u_{n_2} \\ \vdots \\ u_n \end{bmatrix}, \]

\[ P(N^{a,*}) u^{a,*} = [P(N_1^a) P(N_2^a) \ldots P(N_g^a)] \begin{bmatrix} u_1^a \\ u_2^a \\ \vdots \\ u_g^a \end{bmatrix}. \]

Therefore,

\[
P(N_a)^a = \sum_{i=1}^{n_1} [P(N_i)u_1] + \sum_{i=n_1+1}^{n_2} [P(N_i)u_i] + \ldots + \sum_{i=n_{n_g-1}+1}^{n} [P(N_i)u_i]
\]

\[ = P(N_1^a)u_1^a(k) + P(N_2^a)u_2^a + \ldots + P(N_g^a)u_g^a = P(N^{a,*}) u^{a,*}. \]

Thus, at any time,

\[ P(N_a)^a - P(N^{a,*}) u^{a,*} = 0. \]

Since \( P(N_a) \) and \( P(N^{a,*}) \) are time-invariant vectors and \( u^a \) and \( u^{a,*} \) are scalars at any time, then \( u^a \) is proportional to \( u^{a,*} \), and \( P(N_a) \) is proportional to \( P(N^{a,*}) \). Denote \( \alpha = \frac{u^a}{u^{a,*}} \), which is time-invariant.

\[ P(N_a) \alpha = P(N^{a,*}) = Agg(P(N_1), P(N_2), \ldots, P(N_n)). \]

\[ Agg(P(N_1), P(N_2), \ldots, P(N_n)) = Agg(P(N_1^a), P(N_2^a), \ldots, P(N_g^a)). \]
Appendix C

Proof of Aggregation

C.1 Proof of Theorem 5

Proof of Theorem 5. According to Equation (5.3),
\[
[u_1 \ u_2 \ ... \ u_n]^T = [\alpha_1 \ \alpha_2 \ ... \ \alpha_n]^T \tilde{u}.
\]
Substitute in Equation (5.4),
\[
P(N^a) = \begin{bmatrix}
P(N_1) & P(N_2) & ... & P(N_n)
\end{bmatrix}
\begin{bmatrix}
u_1 \\ u_2 \\ \vdots \\ u_n
\end{bmatrix}
\frac{1}{\tilde{u}}.
\]
Then, simply assign \( u^a = \tilde{u} \). Therefore,
\[
P(N^a)u^a = \begin{bmatrix}
P(N_1) & P(N_2) & ... & P(N_n)
\end{bmatrix}
\begin{bmatrix}
u_1 \\ u_2 \\ \vdots \\ u_n
\end{bmatrix}.
\]
So, it satisfies Aggregation Property. \( \blacksquare \)

C.2 Proof of Theorem 6

Proof of Theorem 6. Consider the set of OR nodes \( \{N_1, N_2, ..., N_n\} \), which form a subsystem. Then we consider the optimization problem for this subsystem. They produce the same output inventory product \( x_{out} \). Given the optimal optimal operation \( u \), denote \( x_{out}(k) \) as the amount of product \( x_{out} \) at \( k \) under \( u \). Then, \( x_{out}(k) \) is always greater than or equal to 0. The amount of of product \( x_{out} \) produced by \( \{N_1, N_2, ..., N_n\} \) at \( k \) under \( u \) is denoted as \( x_{out}^+(k) \), and the amount of of product \( x_{out} \) consumed at \( k \) under \( u \) is denoted as \( x_{out}^-(k) \). Then, \( x_{out}^+(k) \geq 0 \), and \( x_{out}(k + 1) = x_{out}(k) + x_{out}^+(k) - x_{out}^-(k) \).
The operation of OR nodes is \( u(k) = [u_1(k) \ u_2(k) \ ... \ u_j(k) \ ... \ u_n(k)]^T \). The network matrix of these nodes is denoted as \( B = [b_1^{\text{out}} \ b_2^{\text{out}} \ ... \ b_n^{\text{out}}] \), where \( b_i^{\text{out}} (i = 1, 2, ..., n, b_i^{\text{out}} > 0) \) is a scaler, since product \( \text{out} \) is the only inventory processed. Since we consider the aggregation without disruption occurring at these nodes, the Capacity Constraint is not in effect for the subsystem by Assumption 5. The inventory update equation is \( x_{\text{out}}(k+1) = x_{\text{out}}(k) + Bu(k) - x_{\text{out}}^-(k) \). Since we consider \( u \) as optimal operation, which minimizes the lost demand, then the problem for the subsystem is formulated as:

\[
\begin{align*}
\text{Minimize} : & \quad \sum_{k'=k-k_p}^{k+k_f} \{a_u q_u u(k') + a_x q_x x_{\text{out}}(k')\}, \\
\text{Subject to} : & \quad x_{\text{out}}(k) \geq 0, \\
& \quad u(k) \geq 0, \\
& \quad x_{\text{out}}(k+1) = x_{\text{out}}(k) + Bu(k) - x_{\text{out}}^-(k), \\
& \quad \kappa_f \geq 1, \\
& \quad k > \kappa_p, \\
& \quad k, \kappa_p, \kappa_f \in \mathbb{N}.
\end{align*}
\]

We want to prove that the optimal \( u \) should satisfy \( u_i(k) = 0 \), for any \( i \) such that there exists a \( j \) satisfying \( \frac{q_u(N_j)}{b_j^{\text{out}}} < \frac{q_u(N_i)}{b_i^{\text{out}}} \). We prove this by contradiction. We denote the optimal control as \( u' \) such that there exist a time \( k_1 \) and \( i_1 \), such that \( u'_{i_1}(k_1) > 0 \), where \( i_1 \) satisfying \( \frac{q_u(N_i)}{b_i^{\text{out}}} < \frac{q_u(N_{i_1})}{b_{i_1}^{\text{out}}} \).

Then we can select a new solution of \( u'' \), which is:

\[
u''(k) = \begin{cases} 
u_i'(k), & \text{when } k \neq k_1, \text{ or } i \neq i_1 \text{ and } i \neq j; \\ 
u_i'(k_1) + u'_{i_1}(k_1) \frac{b_i^{\text{out}}}{b_{i_1}^{\text{out}}}, & \text{when } k = k_2, \text{ and } i = j; \\ 0, & \text{when } k = k_1, \text{ and } i = i_1. \end{cases}
\]

Then, \( u''(k) \geq 0 \). \( x''_{\text{out}}(k+1) = x''_{\text{out}}(k) + Bu''(k) - x_{\text{out}}^-(k) = x''_{\text{out}}(k+1) \) for any \( k \). Since \( \frac{q_u(N_{i_1})}{b_{i_1}^{\text{out}}} < \frac{q_u(N_{i_1})}{b_{i_1}^{\text{out}}} \), then \( q_u u''(k_1) < q_u u'(k_1) \). Compare the objective functions, we find:

\[
\sum_{k'=k_1-k_p}^{k_2+k_f} \{a_u q_u u''(k') + a_x q_x x_{\text{out}}''(k')\} < \sum_{k'=k_2-k_p}^{k_2+k_f} \{a_u q_u u'(k') + a_x q_x x_{\text{out}}'(k')\}.
\]

which means \( u' \) is not the optimal solution. This is a contradiction. So, if optimal solution is \( u \), then \( u \) should satisfy \( u_i(k) = 0 \), for any \( i \) such that there exists a \( j \) satisfying \( \frac{q_u(N_j)}{b_j^{\text{out}}} < \frac{q_u(N_i)}{b_i^{\text{out}}} \).

Consider the index \( j \) such that \( \frac{q_u(N_j)}{b_j^{\text{out}}} \leq \frac{q_u(N_i)}{b_i^{\text{out}}} \) for any \( i \). If there are multiple indices satisfying this condition, it doesn’t matter we choose which one to be \( j \). Because they give the same total cost as long as \( N_j \) produces the optimal \( x_{\text{out}}^+(k) \) for any \( k \). We can let
the operations of all nodes be 0 except $N_j$, which is an optimal operation. The operation becomes:

$$[u_1(k) \ u_2(k) \ ... \ u_j(k) \ ... \ u_n(k)]^T$$

$$= [0 \ 0 \ ... \ u_j(k) \ ... \ 0]^T.$$  

Also, we define  

$$P(N^a) = P(N_j).$$  

So, the left side of Equation (5.2) becomes:

$$[P(N_1) \ P(N_2) \ ... \ P(N_n)] \begin{bmatrix} u_1(k) \\ u_2(k) \\ \vdots \\ u_n(k) \end{bmatrix} = P(N_j) \cdot u_j(k)$$

$$= P(N^a)u_j(k).$$

Then, simply assign $u^a(k) = u_j(k)$, then

$$[P(N_1) \ P(N_2) \ ... \ P(N_n)] \begin{bmatrix} u_1(k) \\ u_2(k) \\ \vdots \\ u_n(k) \end{bmatrix} = P(N^a)u^a(k).$$

Thus, the *Agg* satisfies Aggregation Property.  

□
Appendix D

Preliminary Work in Future Directions

D.1 Studies on Additional Specific Systems and Networks

D.1.1 More Complicated Assembly Tree Network Systems

For the analysis in continuous time domain, we solved the issue of a special assembly tree network systems consisting of one disrupted chain and one branch under Assumption 13, 14 and 15. But there are still cases not covered by these three assumptions. The absolute complement of Assumption 13 and 14 is:

Assumption 16. There exists a $k \in \{l, l-1, \ldots, \lambda + 1\}$ and a $j \in \{\lambda, \lambda - 1, \ldots, 1\}$ such that $\tilde{q}_x(N_k) > \tilde{q}_x(N_j)$, or there exists a $k \in \{l, l-1, \ldots, \lambda + 1\}$ and a $i \in \{n, n-1, \ldots, 0\}$ such that $\tilde{c}(N_k) > \tilde{c}(N^{br}_i)$.

Assumption 15 is only a subset of Assumption 16 of which the case is much more complicated.

Also, there are more types of assembly tree network systems, such as those with multiple branches. Following is the analysis of a special assembly tree network system with a chain upstream of $N_{dis}$. And the systems are already normalized.

An example is given in Figure D.1. In this example, we assume the capacities and storage costs satisfy:

$$c(N^a_{\lambda}) \geq c(N_{\gamma-1}) > c(N^{br}_n) > c(N_0),$$

Figure D.1: An example
\[ c(N_{\text{dis}}) > c(N_{\gamma-1}) > c(N_{\gamma}), \]

\[ q_x(N_{\gamma}) < q_x(N_{\gamma}^a) \leq q_x(N_{\gamma-1}) < q_x(N_{\text{dis}}), \]

\[ q_x(N_{\lambda}^{br}) < q_x(N_{\lambda}^a). \]

Then, the buildup process can have two cases.

Case 1: If \( q_x(N_{\lambda}^a) < q_x(N_{\lambda}^{br}) + q_x(N_{\gamma}) \), then the buildup process should be as in Figure D.2.

In this case, \( N_{\lambda}^a \) works at the rate equal to the capacity of \( N_{\gamma} \) during \([t_{bx,\lambda}^{a}, t_{br}^{a}].\)

![Figure D.2: The buildup process if \( q_x(N_{\lambda}^a) < q_x(N_{\lambda}^{br}) + q_x(N_{\gamma}) \)](image)

Case 2: If \( q_x(N_{\lambda}^a) \geq q_x(N_{\lambda}^{br}) + q_x(N_{\gamma}) \), then the buildup process should be as in Figure D.3.

In this case, \( N_{\lambda}^a \) starts ramping up its operation at a later than in case 1. The operation rate of \( N_{\lambda}^a \) equals to the capacity of \( N_{\gamma-1} \) during \([t_{bx,\lambda}^{a}, t^{br}].\)

If adding two more nodes to the example above, the new system is shown in Figure D.4.

The new nodes satisfy:

\[ c(N_0) < c(N_{\lambda}^{br}) < c(N_{\lambda}^a) < c(N_{\gamma-1}) \leq c(N_{\lambda}^a), \]

\[ c(N_{\gamma}) < c(N_{\gamma-1}) < c(N_{\text{dis}}), \]

164
Then, Case 1: if \( q_x(N^\alpha_\lambda) \geq q_x(N^\text{br}_\xi) + q_x(N_{\gamma}) \), then the buildup process should be as in Figure D.5: In this case, \( N^\alpha_\lambda \) works at the rate equal to the capacity of \( N_{\gamma-1} \) during \( [t^a_{bx,\lambda}, t^\text{br}] \).

Case 2: if \( q_x(N^\alpha_\lambda) < q_x(N^\text{br}_n) + q_x(N_{\gamma}) \), then the buildup process should be as in Figure D.6: In this case, \( N^\alpha_\lambda \) works at the rate equal to the capacity of \( N_{\gamma} \) during \( [t^a_{bx,\lambda}, t^\text{br}] \).

Case 3: if \( q_x(N^\text{br}_n) + q_x(N_{\gamma}) \leq q_x(N^\alpha_\lambda) < q_x(N^\text{br}_\xi) + q_x(N_{\gamma}) \), More subcases need to be discussed.
Case 3.1: if \( q_x(N_{br}^n) + q_x(N_i) \leq q_x(N_\lambda^a) < \min\{q_x(N_{br}^n) + q_x(N_i), q_x(N_{br}^n) + q_x(N_\gamma)\} \), then the buildup process should be as in Figure D.7. Notice that the operation rate of \( N_\lambda^a \) within 

![Figure D.7: The buildup process of Case 3.1](image)

\([t_{bx,\lambda}^a, t_{bx,\gamma}]\) should be \( \min\{c(N_{br}^n) - d, c(N_\gamma) - d\} \).

Case 3.2: if \( \max\{q_x(N_{br}^n) + q_x(N_i), q_x(N_{br}^n) + q_x(N_\gamma)\} \leq q_x(N_\lambda^a) < q_x(N_{br}^n) + q_x(N_\gamma) \), then the buildup process should be as in Figure D.8. In this case, the operation rate of \( N_\lambda^a \) within 

![Figure D.8: The buildup process of Case 3.2](image)

\([t_{bx,\lambda}^a, t_{bx,\gamma}]\) equals to the capacity of \( N_\gamma \) and the rate of \( N_\lambda^a \) within \([t_{bx,\gamma}, t_{bx,\gamma-1}]\) equals to the capacity of \( N_{\zeta}^{br} \).

Case 3.3: if \( q_x(N_{br}^n) + q_x(N_i) \leq q_x(N_\lambda^a) < q_x(N_{br}^n) + q_x(N_\gamma) \), then the buildup process should be as in Figure D.9. In this case, the operation rate of \( N_\lambda^a \) within \([t_{bx,\lambda}^a, t_{bx,\gamma}]\) equals to the capacity of \( N_\gamma \).

Case 3.4: if \( q_x(N_{br}^n) + q_x(N_i) \leq q_x(N_\lambda^a) < q_x(N_{br}^n) + q_x(N_\gamma) \), then the buildup process should be as in Figure D.10. In this case, the operation rate of \( N_\lambda^a \) within \([t_{bx,\lambda}^a, t_{bx,\gamma-1}]\) equals to the capacity of \( N_{\zeta}^{br} \).

Based on the analysis above, for more complicated assembly tree network systems, to determine the operation needs comparison between storage cost combinations. If the structures are more complicated than the examples above, then more cases need to be discussed based on more combinations of unit storage costs.
D.1.2 Extended OR Nodes Network

Definition 16. Consider a set of nodes $\{N_1, N_2, ..., N_n, N_{n+1}\}$. If their operation vectors satisfy:

$$b_{n+1} = \alpha_1 b_1 + \alpha_2 b_2 + ... + \alpha_n b_n,$$

where $\alpha_1, \alpha_2, ..., \alpha_n$ are non-negative, then $\{N_1, N_2, ..., N_n, N_{n+1}\}$ is an extended OR nodes network.

Theorem 11. If $\{N_1, N_2, ..., N_n, N_{n+1}\}$ are extended OR nodes network satisfying Definition 16, then at time $k$ in nominal state, there exists an optimal control $\{u_1(k), u_2(k), ..., u_n(k), u_{n+1}(k)\}$ such that at least one of $\{u_1(k), u_2(k), ..., u_n(k), u_{n+1}(k)\}$ is 0.

Proof. We want to prove that there exists one optimal solution $u(k) = [u_1(k), u_2(k), ..., u_n(k), u_{n+1}(k)]^T$, such that at least one of $\{u_1(k), u_2(k), ..., u_n(k), u_{n+1}(k)\}$ will be 0.

We show this by contradiction. We denote that $u'(k) = [u'_1(k), u'_2(k), ..., u'_n(k), u'_{n+1}(k)]^T$ represents any of the optimal solutions, and $u'(k)$ does not satisfy Theorem 11. Then, $\{u'_1(k), u'_2(k), ..., u'_n(k), u'_{n+1}(k)\}$ are non-zero.

Then we compare the values of $q_u(n+1)$ and $\alpha_1 q_u(1) + \alpha_2 q_u(2) + ... + \alpha_n q_u(n)$. 

Figure D.9: The buildup process of Case 3.3

Figure D.10: The buildup process of Case 3.4
1. If \( q_u(n+1) = \alpha_1q_u(1) + \alpha_2q_u(2) + \ldots + \alpha_nq_u(n) \), then we can select a new operation vector \( u''(k) \), where

\[
    u''(k) = [u'_1(k) + \alpha_1u'_{n+1}(k), u'_2(k) + \alpha_2u'_{n+1}(k), \ldots, u'_n(k) + \alpha_nu'_{n+1}(k), 0]^T.
\]

Then,

\[
    [b_1, b_2, \ldots, b_n, b_{n+1}]u'(k) = [b_1, b_2, \ldots, b_n, b_{n+1}]u''(k),
\]

\[x'' = x' \geq 0,\]

\[u''(k) \geq 0,\]

\[
[q_u(1), q_u(2), \ldots, q_u(n), q_u(n+1)]u''(k) = [q_u(1), q_u(2), \ldots, q_u(n), q_u(n+1)]u''(k).
\]

Thus, \( u''(k) \) is also optimal. However, \( u''(k) \) has at least one 0 element. Therefore, \( u'(k) \) does not represent any of the optimal solutions, which is a contradiction.

2. If \( q_u(n+1) > \alpha_1q_u(1) + \alpha_2q_u(2) + \ldots + \alpha_nq_u(n) \), then we can select a new operation vector \( u''(k) \), where

\[
    u''(k) = [u'_1(k) + \alpha_1u'_{n+1}(k), u'_2(k) + \alpha_2u'_{n+1}(k), \ldots, u'_n(k) + \alpha_nu'_{n+1}(k), 0]^T.
\]

Then,

\[
    [b_1, b_2, \ldots, b_n, b_{n+1}]u'(k) = [b_1, b_2, \ldots, b_n, b_{n+1}]u''(k),
\]

\[x'' = x' \geq 0,\]

\[u''(k) \geq 0,\]

\[
[q_u(1), q_u(2), \ldots, q_u(n), q_u(n+1)]u'(k) > [q_u(1), q_u(2), \ldots, q_u(n), q_u(n+1)]u''(k).
\]

Thus, \( u'(k) \) is not optimal, which is a contradiction.

3. If \( q_u(n+1) < \alpha_1q_u(1) + \alpha_2q_u(2) + \ldots + \alpha_nq_u(n) \), then we consider the set of \( \{\frac{u'_1(k)}{\alpha_1}, \frac{u'_2(k)}{\alpha_2}, \ldots, \frac{u'_n(k)}{\alpha_n}\} \). Denote \( \beta \) as the value of the smallest one in this set. Then,

\[\beta > 0,\]

\[
[u'_1(k), u'_2(k), \ldots, u'_n(k)] - \beta[\alpha_1, \alpha_2, \ldots, \alpha_n] \geq 0,
\]

We can select a new solution \( u''(k) \), where

\[
    u''(k) = [u'_1(k) - \beta\alpha_1, u'_2(k) - \beta\alpha_2, \ldots, u'_n(k) - \beta\alpha_n, u'_{n+1}(k) + \beta]^T.
\]

Then,

\[
    [b_1, b_2, \ldots, b_n, b_{n+1}]u'(k) = [b_1, b_2, \ldots, b_n, b_{n+1}]u''(k),
\]

\[x'' = x' \geq 0,\]

\[u''(k) \geq 0,\]

\[
[q_u(1), q_u(2), \ldots, q_u(n), q_u(n+1)]u'(k) > [q_u(1), q_u(2), \ldots, q_u(n), q_u(n+1)]u''(k).
\]

Therefore, \( u'(k) \) is not optimal, which is a contradiction.
To sum up, $u'(k)$ can not represent any of the optimal solutions. This is a contradiction. So, there exists one optimal solution $u(k) = [u_1(k), u_2(k), ..., u_n(k), u_{n+1}(k)]^T$, such that at least one of $\{u_1(k), u_2(k), ..., u_n(k), u_{n+1}(k)\}$ will be 0.

For the extended OR nodes network, we have proved at least one node has 0 operation in nominal state. But at different time, such non-working nodes can be different. An example is as follows.

![Figure D.11: An example of extended OR nodes network](image)

Assume the matrix of the network is:

$$
B = \begin{bmatrix}
x_1 & N_1 & 0 & 0 & -m_{1,4} & m_{1,5} \\
x_2 & m_{2,1} & m_{2,2} & -m_{2,3} & 0 & 0 \\
x_3 & 0 & 0 & m_{3,3} & 0 & 0 \\
x_4 & 0 & 0 & 0 & m_{4,4} & 0
\end{bmatrix}
$$

Then,

$$b_1 = \frac{m_{2,1}}{m_{2,2}} b_2 + \frac{m_{1,1}}{m_{1,5}} b_5.$$

Thus, $\{N_1, N_2, N_5\}$ is an extended OR nodes network. Now we consider which node of them has 0 operation during the nominal state.

1. If $q_u(N_1) > \frac{m_{2,1}}{m_{2,2}} q_u(N_2) + \frac{m_{1,1}}{m_{1,5}} q_u(N_5)$, then $N_1$ has 0 operation.

2. If $q_u(N_1) < \frac{m_{2,1}}{m_{2,2}} q_u(N_2) + \frac{m_{1,1}}{m_{1,5}} q_u(N_5)$, then either $N_2$ or $N_5$ has 0 operation, based on the demands of $x_2$ and $x_1$. We denote these demands as $d(x_2)$ and $d(x_1)$.

   (a) If $\frac{d(x_1)}{m_{1,1}} > \frac{d(x_2)}{m_{2,1}}$, then $N_2$ has 0 operation.

   (b) If $\frac{d(x_1)}{m_{1,1}} < \frac{d(x_2)}{m_{2,1}}$, then $N_5$ has 0 operation.

   (c) If $\frac{d(x_1)}{m_{1,1}} = \frac{d(x_2)}{m_{2,1}}$, then both $N_2$ and $N_5$ has 0 operation.

This shows that the operation is not only based on network structure but also related to the demand. Besides, to determine the operations under disruption, we need further analysis.
D.1.3 Other Networks

In this section, we list several examples of networks which needs further studies. An example is shown by Figure D.12.

![Figure D.12: An example of networks of OR nodes with different output](image)

Its network matrix can be represented by:

\[
B = \begin{bmatrix}
N_1 & N_2 & N_3 & N_4 \\
x_1 & 1 & 0 & 0 & -1 \\
x_2 & 1 & 1 & -1 & 0 \\
x_3 & 0 & 0 & 1 & 0 \\
x_4 & 0 & 0 & 0 & 1 \\
\end{bmatrix}.
\]

If \(d(x_4) > d(x_3)\), the system is not stable due to product\(_2\) building up, while meeting the demand for product\(_4\).

If \(d(x_4) < d(x_3)\), let’s assume \(d(x_4) = 1\) and \(d(x_3) = 2\). So, \(u_4 = 1\) and \(u_3 = 2\). If we select \(u_1 = 1\) and \(u_2 = 1\), the system will be stable. However, if the production cost of \(N_2\) \((q_u(N_2))\) is much larger, based on the optimization, we will get \(u_1 = 2\) and \(u_2 = 0\). Then the system becomes unstable.

Another example is shown by Figure D.13.

![Figure D.13: An example of parallel networks](image)

Its network matrix can be represented by:

\[
B = \begin{bmatrix}
N_1 & N_2 & N_3 & N_4 \\
x_1 & m_{1,1} & -1 & 0 & 0 \\
x_2 & 1 & 0 & -1 & 0 \\
x_3 & 0 & 1 & 0 & -1 \\
x_4 & 0 & 0 & 1 & -1 \\
x_5 & 0 & 0 & 0 & 1 \\
\end{bmatrix}.
\]

When \(m_{1,1} = 1\), \(\{N_1, N_2, N_3, N_4\}\) are a network of nodes with fixed ratio.
When $m_{1,1} \neq 1$, $\{N_1, N_2, N_3, N_4\}$ might have a time-varying operation ratio in dynamic process. Also they might still have a fixed operation ratio in steady state, but such ratio is determined by inventory storage cost.

Another example is shown by Figure D.14.

![Figure D.14: An example of multi-output nodes network](image)

Its network matrix is:

$$B = \begin{bmatrix} N_1 & N_2 \\ m_{1,1} & -m_{1,2} \\ m_{2,1} & 0 \\ 0 & m_{3,2} \end{bmatrix}.$$ 

When

$$\frac{d(x_3)}{d(x_2)} = \frac{m_{1,1} m_{3,2}}{m_{1,2} m_{2,1}},$$

it is found that

$$\frac{u_1}{u_2} = \frac{m_{1,2}}{m_{1,1}},$$

and there is no inventory build up. $N_1$ and $N_2$ can be considered as a network of nodes with fixed ratio.

If $\frac{d(x_3)}{d(x_2)}$ changes, they cannot be directly considered as a network of nodes with fixed ratio.

To sum up, we do not have sufficient results of the properties and operations of all the systems listed above. In the future, we can do deeper investigation to study them.

### D.2 Studies on General Approach to the Optimal Operation

#### D.2.1 Node Conversion

There are some nodes which are not nodes with fixed operation ratio or OR nodes. But they can be replaced by a sub-network which consists of only the basic two kinds of nodes. The problem is to convert a number of different types of nodes to the combination of the basic two kinds of nodes.

For a node $N_i$ which does not satisfy Definition 11 and 6, the problem is to convert $N_i$ such that the new node(s) can satisfy Definition 11 and 6. First we need to list some types of such nodes which can appear in the common system. Then we can develop methods of conversion for each type of nodes. One possible way is to find a set of virtual nodes $\{N_1^v, N_2^v, ..., N_n^v\}$ such that $N_1^v, N_2^v, ..., N_n^v$ all satisfy Definition 11 and 6, and

$$P(N_i)u_i = [P(N_1^v) \ P(N_2^v) \ ... \ P(N_n^v)]\begin{bmatrix} u_1^v \\ u_2^v \\ \vdots \\ u_n^v \end{bmatrix}.$$
D.2.2 Removing Certain Assumptions

The current study is based on several assumptions, such as no consideration of changeover cost and changeover constraints, loose resource capacity constraints, highest priority to reduce loss demand cost, etc. We use them to reduce the complexity of the system and analysis process. If some of them are removed, the Aggregation Property and the approaches may be changed. For example, more elements will be added to node parameter vector $P\{N_i\}$; some sets of nodes will no longer keep fixed ratio; and some OR nodes may have some limits on operation. All these changes can reduce the availability of current aggregation approaches. We should think about whether the approaches can be extended so that the Aggregation Property can still be satisfied even when there are fewer assumptions.

The aggregation when certain assumptions are removed can be studied by following these steps:

1. Analyze the current aggregation. Consider one assumption which can affect the current aggregation least if removed. It seems that loose resource capacity constraints and highest priority to reduce loss demand cost do not affect the current aggregation too much.

2. Remove the selected assumption. Reformulate the optimization problem. Figure out what is changed in the problem and is related to the proof of current aggregation.

3. Based on the changes, adjust the aggregation approach and proof of it so that it can be extended to satisfy the new formulated problem.

4. Go to step 1 to consider removing another assumption.

One removed assumption is the changeover constraint and cost. We’ve modeled the changeover cost in linear or quadratic form, although our analysis assumes the cost is zero. However, both of these forms may not represent the changeover cost in the real world. For example, negative change of operation may introduce a little cost, which can happen when reducing the operation rate requires extra labor force to uninstall equipments. Or, changeover cost can be close to a constant no matter how much increase occurs on a certain operation. For one node $N_i$, a possible way to model its changeover cost in the discrete time domain could be:

$$C_{\varsigma,i}(k) = \begin{cases} 
q_\varsigma(i)[u_i(k) - u_i(k-1)], & \text{when } u_i(k) > u_i(k-1) \\
0, & \text{when } u_i(k) \leq u_i(k-1)
\end{cases}.$$

$C_{\varsigma,i}(k)$ is the changeover cost of $N_i$ at time $k$. $q_\varsigma(i)$ is the changeover cost coefficient of $N_i$. Then the total changeover cost will be:

$$C_\varsigma(k) = a_\varsigma \sum_i C_{\varsigma,i}(k).$$

$a_\varsigma$ is the weight scaler of changeover cost.

With this new approach, the system model is changed and the problem is formulated not as a linear program issue. Then we need to study on how to solve this new problem.
D.2.3 Adding Constraint of Limit of Inventory Storage

In our model, we do not include any constraint of limit of inventory storage. We assume that we can store infinite amount of any product. However, in practical situation, there may be a limit for the maximum amount allowed to be stored. For example, there is a finite space for storage at a factory for its products.

We can simply add such a constraint to formulate the limit of inventory storage.

\[ 0 \leq x \leq \chi, \]

where vector \( \chi \) set the upper bound of \( x \). With this constraint added, the optimization problem can still be run to give solutions for the numerical study. However, for the analytical study, this constraint can change the current results significantly.

For example, consider a DSCDC network system with 3 nodes: \( N_2, N_1 \) and \( N_0 \). Without inventory limit, the optimal buildup process can be shown by Figure D.15:

If we increase \( x_1(t_{dis}) \) by \( \Delta x \) and decrease \( x_2(t_{dis}) \) by \( \Delta x \), then the buildup process is shown by the dash line in Figure D.16:

We can prove that the greater \( \Delta x \) is, the more storage cost is increased.
Similarly, if we decrease \( x_1(t_{\text{dis}}) \) by \( \Delta x \) and increase \( x_2(t_{\text{dis}}) \) by \( \Delta x \), then the buildup process is shown by the dash line in Figure D.17.

![Figure D.17: Decrease \( x_1(t_{\text{dis}}) \)](image)

We can prove that the greater \( \Delta x \) is, the more storage cost is increased.

This property can be extended to any serial network system. When increasing \( \bar{x}^d_{s_i}(t_{\text{dis}}) \) by \( \Delta \bar{x} \) and decreasing \( \bar{x}^d_{s_{i+1}}(t_{\text{dis}}) \) by \( \Delta \bar{x} \), the total storage cost increases monotonously with \( \Delta \bar{x} \); When decreasing \( \bar{x}^d_{s_i}(t_{\text{dis}}) \) by \( \Delta \bar{x} \) and increasing \( \bar{x}^d_{s_{i+1}}(t_{\text{dis}}) \) by \( \Delta \bar{x} \), the total storage cost also increases monotonously with \( \Delta \bar{x} \).

Now, we consider the DSCDC network system with 4 nodes: \( N_3, N_2, N_1 \) and \( N_0 \). The maximum values of buildups of product 3, product 2 and product 1 happen at \( t_{\text{dis}}, t_3 \) and \( t_2 \), respectively.

If the limit of buildup of product 2 is smaller than \( x_2(t_3) \), then we need to adjust the current operation. There can be several options, shown by Figure D.18 and D.19. It’s not straightforward to determine the new optimal operation under the inventory limit.

To sum up, we need more research on how the constraint of limit of inventory storage can affect the current results of analytical study.
D.2.4 Potential Disrupted Nodes

For the set of nodes \( \{N_1, N_2, \ldots, N_n\} \), if there is at least one node which can be potentially disrupted, what is the aggregation approach to find \( N^a \) such that the Aggregation Property can still hold. If there is no way to satisfy the Aggregation Property, is it possible to make some modifications to current Aggregation Property such that the aggregation has fewer requirements. For example, we may have a loose aggregation property in discrete time domain as:

\[
\sum_{k'=k-k_f}^{k+k_f} [P(N_1) P(N_2) \ldots P(N_n)] \begin{bmatrix} u_1(k') \\ u_2(k') \\ \vdots \\ u_n(k') \end{bmatrix} = \sum_{k'=k-k_f}^{k+k_f} P(N^a) u^a(k') \\
\forall k.
\]

An example of the analysis of potential disruption is illustrated as follows. Consider the system shown by Figure [D.20]

![Figure D.20: A system example](image)

Specifications of these system is listed as follows:

1. The network matrix is \( B = \begin{bmatrix} -1 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 1 \\ 1 & 1 & -1 & 0 & 0 \end{bmatrix} \).

2. Time window parameters are \( \kappa_p = 2 \) and \( \kappa_f \).
3. $a_u q_u u(k)$ is the production cost. $q_u = [ q_u(1) \quad q_u(2) \quad -4 \quad 2 \quad 2 ]$, $a_u = 1$.

4. $a_x q_x x(k)$ is the inventory cost. $q_x = [ 1 \quad 1 \quad 1 ]$, $a_x = \text{very large number (such as 10)}$.

5. $a_\varsigma (u(k) - u(k-1))^T q_\varsigma (u(k) - u(k-1))$ is the changeover cost, $a_\varsigma = 0.2$, $q_\varsigma = 
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}.

6. $a_d (d - D u(k))$ is the lost demand cost, $d = 4$, $D = [ 0 \quad 0 \quad 1 \quad 0 \quad 0 ]$, $a_d = \text{very large number (such as 10)}$.

7. Capacity: $R u(k) \leq c(k)$.

\[
R = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0
\end{bmatrix}.
\]

$c(k)$ is a time-varying variable. In normal condition, $c(k) = \begin{bmatrix} 10 \\ 6 \end{bmatrix}$. When disruption happens at node 1, we consider its result is to reduce the second resource to zero. Thus, $c(k) = \begin{bmatrix} 0 \\ 6 \end{bmatrix}$. The time duration of the disruption is $\kappa_{dis} = 11$, started at $k_{dis}$.

8. S-Constraint: $u(k) - u(k-1) \leq \varsigma$.

9. The probability that the disruption will happen at $N_1$ is $p_1$; the probability that the disruption will happen at $N_2$ is $p_2$.

Assumptions are as follows:

1. Lost demand cost is very large. So in any condition, the system will try to make no lost demand. Hence, the lost demand cost should always be kept to zero.

2. Inventory storage cost is very large. So in any condition, the system will try to make no inventory storage. Hence, the inventory storage cost should be kept to zero.

We can conclude these two assumptions as $u_1(k) + u_2(k) = d = u_3(k)$, $u_4 = u_1$, $u_5 = u_2$.

3. The changeover constraint is very loose. With the two assumptions above, the operation changes can always satisfy the changeover constraint. The increase of operations is always smaller than the limit.

4. $q_u(2) = a q_u(1)$, $q_u(4) = q_u(5)$.

5. $a_\varsigma$, $u_1$ and $u_2$ can not be too small.

When node 1 and node 2 have the same production cost, then the total cost increase in the transition cycles is:

\[
p_1 \frac{4u_1(k)^2}{\kappa_f + 1} + p_2 \frac{4u_2(k)^2}{\kappa_f + 1}.
\]
we can choose:

\[ u_1(k) = \frac{p_2}{p_1 + p_2} d, \]
\[ u_2(k) = \frac{p_1}{p_1 + p_2} d. \]

In this way, the cost increase due to disruption can be minimized.

When node 1 and node 2 have different production cost and disruption happens at node 1, assuming in each of the cycles of \( \{k_{\text{dis}} - \kappa_f, k_{\text{dis}} - \kappa_f + 1, \ldots, k_{\text{dis}}\} \), the change of production of any node is \( \Delta u \). Then, the total cost increase in the transition cycles is:

\[
\sum_{i=1}^{\kappa_f+1} \left\{ a_u q_u(1)(a - 1)(\kappa_f + 2 - i) \Delta u + 4a_\varsigma [\Delta u]^2 \right\}
\geq \frac{4a_\varsigma}{\kappa_f + 1} u_1(k)^2 + \frac{a_u q_u(1)(a - 1)(\kappa_f + 2)}{2} u_1(k) - \frac{[a_u q_u(1)(a - 1)]^2 (\kappa_f + 1)(\kappa_f + 2) \kappa_f}{192a_\varsigma}.
\]

Consider the disruption happens at node 1 and node 2 with probabilities of \( p_1 \) and \( p_2 \), the minimum cost increase is:

\[
\frac{4a_\varsigma(p_1 + p_2)}{\kappa_f + 1} u_1(k)^2 + \frac{a_u q_u(1)(a - 1)(\kappa_f + 2)(p_1 + p_2)}{2} u_1(k) - \frac{8a_\varsigma dp_2}{\kappa_f + 1} u_1(k) + \frac{4a_\varsigma d^2 p_2}{\kappa_f + 1}
- \frac{a_u q_u(1)(a - 1)(\kappa_f + 2)p_2 d}{2} - \frac{[a_u q_u(1)(a - 1)]^2 (\kappa_f + 1)(\kappa_f + 2) \kappa_f(p_1 + p_2)}{192a_\varsigma}.
\]

we have the operation choices as:

\[ u_1(k) = \frac{p_2}{p_1 + p_2} d - \frac{a_u q_u(1)(a - 1)(\kappa_f + 2)(\kappa_f + 1)}{16a_\varsigma}, \]
\[ u_2(k) = \frac{p_1}{p_1 + p_2} d + \frac{a_u q_u(1)(a - 1)(\kappa_f + 2)(\kappa_f + 1)}{16a_\varsigma}. \]

Parameters affecting operation choices can be concluded as:

- \( p_1 \) and \( p_2 \)
  From the equations above, we can see that \( u_1 < u_2 \) if \( p_1 > p_2 \). If node 1 has larger probability to get disruption, it’ll be better to let it take less work, vice versa.

- \( a_u q_u(1)(a - 1) \)
  This item indicates the difference between the costs of two nodes. \( a_u q_u(1)(a - 1) = a_u[q_u(2) - q_u(1)] \). When this cost difference increases, \( u_1 \) will decrease. It can be explained like this. If the cost difference becomes larger, it will cost more to shift from node 1 to node 2. In this case, it’s better to let node 1 take less work.

- \( \kappa_f \)
  Assume \( a > 1 \). According to the equations, \( \kappa_f \) increasing will cause \( u_1 \) decrease. We know the changeover cost will decrease as \( \kappa_f \) increases.
However, the $\kappa_f$ can also cause much bigger increase in production cost change. In each cycle, we have a increase in production cost change. For example, considering node 1 get disruption, the production cost change in each cycle will be between 0 and $a_u[q_u(2) - q_u(1)]u_1(k)$.

If $k_f = 0$, the shift will take only one cycle, and the total change of production will be

$$a_u[q_u(2) - q_u(1)]u_1(k).$$

If $k_f = 1$, the shift will take 2 cycles. Assume the operation changes at node 1 in each cycle is $\Delta u$. Then the total production cost change is:

$$a_u[q_u(2) - q_u(1)]\Delta u + a_u[q_u(2) - q_u(1)](\Delta u + \Delta u)$$

$$= a_u[q_u(2) - q_u(1)]\Delta u + a_u[q_u(2) - q_u(1)]u_1(k)$$

$$\geq a_u[q_u(2) - q_u(1)]u_1(k).$$

We can see when $\kappa_f$ increases then the production cost change will become greater. As a result, we hope to reduce the change if $\kappa_f$ is larger. So $u_1(k)$ should be smaller.

- $a_\varsigma$

Assuming $a > 1$, the increase of $a_\varsigma$ can cause the increase of $u_1(k)$. The increase of $a_\varsigma$ can reduce the weight of production cost. So the results will more approach to the situation when node 1 and node 2 have the same production costs.

We notice $a_\varsigma$ has a lower boundary given that $u_i \geq 0$.

In the analysis above, we only considered the cost increase in the transition cycles. To calculate the total cost in transition cycles, we need to add some basic production cost. Before disruption, the production cost is:

$$a_u[q_u(1)u_1(k) + q_u(2)u_2(k) + q_u(3)d + q_u(4)u_1(k) + q_u(5)u_2(k)]$$

$$= a_u \{[q_u(1) + q_u(4)]u_1(k) + [q_u(2) + q_u(5)]d - u_1(k) + q_u(3)d\}$$

$$= a_u \{[q_u(1) - q_u(2)]u_1(k) + [q_u(2) + q_u(3) + q_u(5)]d\}.$$

There are $\kappa_f + 1$ transition cycles, so the total basic production cost is:

$$a_u(\kappa_f + 1) \{[q_u(1) - q_u(2)]u_1(k) + [q_u(2) + q_u(3) + q_u(5)]d\}.$$

No matter whether the disruption happens at node 1 or node 2, the basic production cost has the same expression as above. What we need to add to get the total cost in transition cycles is:

$$a_u(p_1 + p_2)(\kappa_f + 1) \{[q_u(1) - q_u(2)]u_1(k) + [q_u(2) + q_u(3) + q_u(5)]d\}.$$

And the total cost in transition cycles is:

$$\frac{4a_\varsigma(p_1 + p_2)}{\kappa_f + 1}u_1(k)^2 + \left[\frac{a_uq_u(1)(a - 1)(\kappa_f + 2)(p_1 + p_2)}{2} - \frac{8a_\varsigma dp_2}{\kappa_f + 1}\right]u_1(k) + \frac{4a_\varsigma d^2p_2}{\kappa_f + 1}$$

$$- \frac{a_uq_u(1)(a - 1)(\kappa_f + 2)p_2d}{2} - \frac{[a_uq_u(1)(a - 1)]^2(\kappa_f + 1)(\kappa_f + 2)\kappa_f(p_1 + p_2)}{192a_\varsigma}$$

$$+ a_u(p_1 + p_2)(\kappa_f + 1) \{[q_u(1) - q_u(2)]u_1(k) + [q_u(2) + q_u(3) + q_u(5)]d\}.$$
Then the operation choices for $u_1(k)$ and $u_2(k)$ are:

$$u_1(k) = \frac{p_2}{p_1 + p_2} - \frac{a_u q_u(1)(a - 1)(\kappa_f + 2)(\kappa_f + 1)}{16a_\varsigma} + \frac{a_u (\kappa_f + 1)^2 q_u(1)(a - 1)}{8a_\varsigma}$$

$$u_2(k) = \frac{p_1}{p_1 + p_2} - \frac{a_u q_u(1)(a - 1)[(\kappa_f + 2)(\kappa_f + 1) - 2(\kappa_f + 1)(\kappa_f + 1)]}{16a_\varsigma}.$$

The results have similar forms compared with ones when only considering the cost increase. However, some properties and effect of parameters are entirely different, because the different sign of the second items in the equations. These properties are more close to common sense.

The effects of parameters are listed below:

- $p_1$ and $p_2$
  From the equations above, we can see that $u_1 < u_2$ if $p_1 > p_2$. If node 1 has larger probability to get disruption, it’ll be better to let it take less work, vice versa.

- $a_u q_u(1)(a - 1)$
  This item indicates the difference between the costs of two nodes. $a_u q_u(1)(a - 1) = a_u[q_u(2) - q_u(1)]$. This cost difference is larger, node 1 is cheaper to run. So node 1 will take more work.

- $\kappa_f$
  Assume $a > 1$. According to the equations, $\kappa_f$ increasing will cause $u_1$ increase. $\kappa_f$ is larger means the transition period is longer. Hence, the production cost in the period is increased. To compensate this increase, the system will prefer less shift of operation. So, the operation choices display the increase of $u_1$ and decrease of $u_2$.

- $a_\varsigma$
  Assuming $a > 1$, the increase of $a_\varsigma$ can cause the increase of $u_1(k)$. The increase of $a_\varsigma$ can reduce the weight of production cost. So the results will more approach to the situation when node 1 and node 2 have the same production costs.

Now we consider a more complicated case. We add one more optional node. Assume we still have the assumptions. We have

$$u_1(k) + u_2(k) + u_3(k) = d.$$

Assume the three nodes have production costs $q_u(1)$, $q_u(2)$ and $q_u(3)$, respectively. The cost increase in the transition cycles is:

$$\sum_{i=1}^{\kappa_f + 1} \left\{ a_u(\kappa_f + 2 - i)[q_u(1)\Delta u_1 + q_u(2)\Delta u_2 + q_u(3)\Delta u_3] + 2a_\varsigma \left[ \Delta u_1^2 + \Delta u_2^2 + \Delta u_3^2 \right] \right\}.$$
subject to:

\[ \Delta u_1 + \Delta u_2 + \Delta u_3 = 0, \]
\[ \sum_{i=1}^{\kappa_f+1} \Delta u_1 = -u_1(k) \text{ (when disruption happens at node 1).} \]

To solve the problem, at least we need to know the steady state of node 2 during the disruption.

\[ \sum_{i=1}^{\kappa_f+1} \Delta u_2 = u_2(k_{dis}) - u_2(k). \]

To sum up, when the system becomes more complicated, the solution of operations under potential disruptions is not straightforward. Therefore, to determine the optimal operations under potential disruptions, we need more investigation.

D.3 Sensitivity Analysis of System Parameters Related to Resilience

Problems in this type do not include the controller design process. Such problems are intended to relate resilience to variable system parameters. It can not only serve as guidance for system design or improvement, but also provide information for control policy design. Control signals can be either given by potential \( u(k) \) or calculated with certain optimal algorithms such as linear program. The former represents open-loop system analysis and the latter stands for closed-loop system analysis. The analysis will mostly be in the form of comparison between the systems before and after variation.

For example, in the discrete time domain, we can do the comparisons as follows:

**Product Flow Comparison** Given time period \( k_f, k_p \), and initial inventory \( x(0) \) with capacity time function \( c(k) \), the problem is to consider the optimal \( C \) for the nominal matrix \( B \) and an alternative operation matrix \( B' \).

**Resource Comparison** Given time period \( k_f, k_p \) and capacity time function \( c(k) \), the problem is to consider the optimal \( C \) for the nominal resource matrix \( R \) and an alternative operation matrix \( R' \).

**Flexibility Speed Comparison** Given time period \( k_f, k_p \) and capacity time function \( c(k) \), the problem is to consider the optimal \( C \) for the nominal changeover rate vector \( \varsigma \) an alternative changeover rate vector \( \varsigma' \).

For open-loop analysis, we will use the given \( u \) as the control signal. So, \( R \) and \( s \) will not affect the total cost \( C \). First, we need to derive a equation to express \( C \) as a function of \( B \), like \( C = f_B(B) \). Second, we can analyze the property of \( f_B \) by either differential analysis such as calculating gradient, or statistic experiments. Last, with the property
of \( f_B \) such as monotonicity, we can choose alternative \( B' \) and compare the corresponding system performance so as to find out the relation to resilience.

For closed-loop analysis, we can still use the method described above to analyze the property of \( f_B, f_R, \) and \( f_s \), which are the function of \( B, R, \) and \( s \) to express \( C, \) respectively. However, before we derive \( f_B, f_R, \) and \( f_s, \) we need to calculate the control signal \( u, \) which is also related to \( B, R, \) and \( s. \) This comes to the problem of Optimal control signal derivation. After this calculation, we can go on the steps of open-loop analysis.

### D.4 Studies on General Properties of Control

The problem is to conclude our results of mathematical control signal to some general control policies, which can be easily understood and carried out even without any solving process with the mathematical model. Once we are able to calculate the optimal control signal, we can then focus on how to conclude our solutions. First, we need to explain the physical meaning of the solution, its cause, and its condition. Then, based on the physical meaning, we can conclude some significant ideas and properties of the control. Finally, we can develop the control policy with these ideas and properties.

#### D.4.1 Decentralized Decision Polices based on Local Information

We notice that the optimal operation rate of each node can only be: (1) the operation rate of its upstream node, or (2) its own capacity, or (3) the operation rate of its downstream node. Therefore, we define modes for these three cases, respectively.

Consider a node \( N^a_j \) downstream of the disrupted node in an aggregated serial network system, where \( 1 \leq j \leq \rho - 1. \) \( N^a_{j-1} \) and \( N^a_{j+1} \) are the nodes downstream and upstream of \( N^a_j, \) respectively.

We already know the operation of \( N^a_j \) satisfies the following:

1. The buildup at \( N^a_{j-1} \) is started earlier than at \( N^a_j. \) Before \( N^a_j \) starts buildup, the operation rate of \( N^a_j \) equals to the operation rate of \( N^a_{j-1}. \) This first mode is a pull mode.

2. From the time when \( N^a_j \) starts buildup, \( N^a_j \) operates at its capacity, until the storage upstream of \( N^a_j \) is used up. During this period, the mode of \( N^a_j \) is a capacity constrained push.

3. When the storage upstream of \( N^a_j \) is used up, the operation rate of \( N^a_j \) should still be as high as possible to push products downstream, but can not be higher than its upstream node’s operation rate. Therefore, the operation rate of \( N^a_j \) equals to that of \( N^a_{j+1}. \) This mode is a inventory constrained push.

That is, for \( 1 \leq j \leq \rho - 1, \)

\[
u^a_j(t) = \begin{cases} u^a_{j-1}(t), & \text{when } t \in [t_{dis} - \tau_f, t_{bx,j}), \\ c(N^a_j), & \text{when } t \in [t_{bx,j}, t_{j+1}), \\ u^a_{j+1}(t), & \text{when } t \in [t_{j+1}, t_{dis} + \tau_{dis}). \end{cases}
\]
Consider a node $N_i^a$ upstream of the disrupted node in a serial network system, where $\rho + 1 \leq i \leq L - 1$. $N_{i-1}^a$ and $N_{i+1}^a$ are the nodes downstream and upstream of $N_i^a$, respectively.

We already know that the operation of $N_i^a$ satisfies the following:

1. The upstream nodes start buildup earlier than the downstream nodes. Although this is different from the situation downstream of $N_{\text{dis}}^a$, $N_i^a$ operates at the same rate as $N_{i-1}^a$ before the buildup starts at $N_i^a$. This is still a pull mode during this period.

2. When the buildup starts at $N_i^a$, $N_i^a$ works at its capacity, which is a capacity-constrained push mode.

3. After the storage upstream of $N_i^a$ is used up, the operation rate of $N_i^a$ equals to that of $N_{i+1}^a$, which is 0. This happens at time $t_{\text{dis}}$, the operation rate of $N_{\text{dis}}$ becomes 0. This can be considered as a pull mode.

That is, for $\rho + 1 \leq i \leq L - 1$,

$$u_i^a(t) = \begin{cases} 
   u_{i-1}^a(t), & \text{when } t \in [t_{\text{dis}} - \tau_f, t_{bx,i}), \text{ (pull)} \\
   c(N_i^a), & \text{when } t \in [t_{bx,i}, t_{\text{dis}}), \text{ (capacity constrained push)} \\
   u_{i-1}^a(t) = u_{i+1}^a(t) = 0, & \text{when } t \in [t_{\text{dis}}, t_{\text{dis}} + \tau_{\text{dis}}), \text{ (pull)}
\end{cases}$$

Consider a node $N_j^a$ in the disrupted chain of the assembly tree network system we studied in Chapter 9 under Assumption 15 and $j \in \{L - 1, L - 2, ..., \mu + 1\} \cup \{\mu - 1, \mu - 2, ..., 1\}$. The analysis is almost the same as the analysis of $N_j^a$ in serial network systems. Then, the modes of operations can be represented by:

$$u_j^a(t) = \begin{cases} 
   u_{j-1}^a(t), & \text{when } t \in [t_{\text{dis}} - \tau_f, t_{bx,j}), \text{ (pull)} \\
   c(N_j^a), & \text{when } t \in [t_{bx,j}, t_{j+1}), \text{ (capacity constrained push)} \\
   u_{j+1}^a(t), & \text{when } t \in [t_{j+1}, t_{\text{dis}} + \tau_{\text{dis}}), \text{ (inventory constrained push)}
\end{cases}$$

Consider a node $N_i^{br,a}$ in the branch of the assembly tree network system we studied in Chapter 9 under Assumption 15 and $1 \leq i \leq \eta - 1$. The analysis is similar to the analysis of $N_j^a$ above. Then, the modes of operations can be represented by:

$$u_i^{br,a}(t) = \begin{cases} 
   u_{i-1}^{br,a}(t), & \text{when } t \in [t_{\text{dis}} - \tau_f, t_{bx,i}^{br}), \text{ (pull)} \\
   c(N_i^{br,a}), & \text{when } t \in [t_{bx,i}^{br}, t_{i+1}^{br}), \text{ (capacity constrained push)} \\
   u_{i+1}^{br,a}(t), & \text{when } t \in [t_{i+1}^{br}, t_{\text{dis}} + \tau_{\text{dis}}), \text{ (inventory constrained push)}
\end{cases}$$

Consider the node $N_i^{br,a}$ in the assembly tree network system we studied in Chapter 9 under Assumption 15. The analysis is similar to the analysis of $N_i^a$ in the serial network systems. The modes can be represented by:

$$u_i^{br,a}(t) = \begin{cases} 
   u_{i-1}^{br,a}(t), & \text{when } t \in [t_{\text{dis}} - \tau_f, t_{bx,\eta}^{br}), \text{ (pull)} \\
   c(N_i^{br,a}), & \text{when } t \in [t_{bx,\eta}^{br}, t_{\mu+1}), \text{ (capacity constrained push)} \\
   u_{\mu}^a(t) = 0, & \text{when } t \in [t_{\mu+1}, t_{\text{dis}} + \tau_{\text{dis}}), \text{ (pull)}
\end{cases}$$
Consider the node $N^a_\mu$ in the assembly tree network system we studied in Chapter 9 under Assumption 15. When building up product $a_\mu$, there are two stages. In the first stage when the storage in the branch is not used up, $N^a_\mu$ works at its capacity, which is a capacity constrained push mode; in the second stage when the storage in the branch is used up, $N^a_\mu$ works at the rate of nodes in the branch, which is a inventory constrained push mode. Then, the modes of $N^a_\mu$ can be represented by:

$$u^a_\mu(t) = \begin{cases} 
  u^a_{\mu-1}(t), & \text{when } t \in [t_{\text{dis}} - \tau_f, t_{bx,\mu}), \\
  c(N^a_\mu), & \text{when } t \in [t_{bx,\mu}, t_{1b}^-), \\
  u^r_{bx}(t), & \text{when } t \in [t_{1b}^+, t_{\mu+1}), \\
  u^a_{\mu+1}(t) = 0, & \text{when } t \in [t_{\mu+1}, t_{\text{dis}} + \tau_{\text{dis}}). 
\end{cases}$$

From the analysis above, we find that the optimal operations of any systems which we have studied can be represented by three types of modes. This may be a general rule for the optimal operations. We need more study to examine whether it works for any other systems.

Based on our analysis of modes of optimal operations, it seems that the optimal operation of any node only depends on itself and its neighbors. The optimal operation may be determined by local information. It will be very convenient for practical use if a policy of operation can be developed based on local information and decisions instead of centralized decisions.

We start the analysis with the case that we only consider the operations upstream of $N_{\text{dis}}$ in a normalized serial network system. The system can be represented by Figure D.21:

![Figure D.21: Upstream serial network system](image)

$N_1$ is the only place downstream of disruption where storage can be built up. The capacities and unit storage costs of $\{N_1, N_2, ..., N_l\}$ are random values. Based on the disruption, we figure out the value $X_{bx}$ of the total amount of storage needed. This amount of storage should be pushed through $N_1$ by the disruption time $t_{\text{dis}}$.

Based on $X_{bx}$, each node can determine the latest time when it should ramp up its operation. Denote $t_{bx,i}$ as the ramping up time of $N_i$. Then,

$$t_{bx,i} \leq t_{\text{dis}} - \frac{X_{bx}}{c(N_i) - d}, \quad \forall i \in \{1, 2, ..., l\}.$$

Besides, the upstream nodes can not ramp up its operation later than the downstream nodes, i.e.,

$$t_{bx,i} \leq t_{bx,i-1}, \quad \forall i \in \{2, 3, ..., l\}.$$

Therefore, in this step, we can temporarily determine that:

$$t_{bx,1} = t_{\text{dis}} - \frac{X_{bx}}{c(N_1) - d},$$

$$t_{1b} = t_{\text{dis}} - \frac{X_{bx}}{c(N_1) - d},$$

$$t_{1b}^- = t_{bx,1} - \frac{t_{1b} - t_{bx,1}}{t_{1b} - t_{bx,1}} \tau_f,$$

$$t_{1b}^+ = t_{bx,2} - \frac{t_{1b}^- - t_{bx,2}}{t_{1b}^- - t_{bx,2}} \tau_f.$$
\[ t_{bx,i} = \min\{t_{bx,i-1}, t_{dis} - \frac{X_{bx}}{c(N_i) - d}\}, \ \forall i \in \{2, 3, ..., l\}. \]

If \( t_{bx,i} = t_{dis} - \frac{X_{bx}}{c(N_i) - d} \), then \( N_i \) is a \( c \)-node.

For \( i \in \{1, 2, ..., l\} \), if \( N_i \) is a \( c \)-node, then we select the Step 1 operation as:

\[
\begin{align*}
  u_i(t) &= \begin{cases} 
  u_{i-1}(t), & \text{when } t < t_{bx,i}, \\
  c(N_i), & \text{when } t_{bx,i} \leq t < t_{dis}.
  \end{cases}
\end{align*}
\]

If \( N_i \) is not a \( c \)-node, then

\[
  u_i(t) = u_{i-1}(t), \ \text{when } t < t_{dis}.
\]

In a word, by comparing \( t_{bx,i-1} \) and \( t_{dis} - \frac{X_{bx}}{c(N_i) - d} \), \( N_i \) can determine its Step 1 operation.

For example, in the network system represented by Figure D.22, the operation in Step 1 is shown in Figure D.23. The slopes of the lines from left to right in this figure are:

\[
\begin{align*}
  c(N_7) - d, & \quad c(N_6) - d, & \quad c(N_3) - d, & \quad c(N_1) - d.
\end{align*}
\]

Therefore, in Step 1, we get a feasible solution of buildup without considering the storage costs. Now we consider the storage costs to get an optimal solution. In Step 2, the storages
are pushed to the nodes with lowest storage costs. This will adjust the operations in Step 1 to the final optimal operations.

We start from the most upstream $c$-node, $N_{c1}$. Denote $N_{s_i}$ as the node with the lowest storage cost downstream of $N_{c_i}$. Then $N_{s_i}$ is an $s$-node. The operations of the nodes between $N_{c_i}$ and $N_{s_i}$ should be the same as the operation of $N_{c_i}$, so that the storage can be push to the location with the lowest storage cost.

Next, consider the next $c$-node $N_{c_k}$ downstream of $N_{s_i}$. Repeat the same process to locate the next $s$-node $N_{s_k}$ which has the lowest storage cost downstream of $N_{c_k}$. Then, adjust the operations of the nodes between $N_{c_k}$ and $N_{s_k}$ to the operation of $N_{c_k}$.

The adjustment is carried out recursively until the operation of $N_{dis}$ is adjusted as an $s$-node. In the example above, the storage costs are listed in Figure D.24.

![Figure D.24: Storage costs of the example of upstream serial network system](image)

Then, the operations are adjusted into Figure D.25.

![Figure D.25: The buildup process in Step 2](image)

The analysis above is about the serial network systems which do not have any node between $N_{dis}$ and $N_0$. Next we consider the serial network systems do not have any node upstream of $N_{dis}$.

Consider the simple case of DSCDC network system as shown in Figure D.26.
The capacities and storage costs satisfy:

\[ c(N_2) > c(N_1) > c(N_0) = 1, \]
\[ q_x(N_2) > q_x(N_1). \]

We know the optimal operations can be represented by Figure D.27.

The exact value of \( t_{bx,1} \) and \( t_{bx,2} \) are determined by optimization program. It seems not obvious that they can determined by local decisions and communications.

From the analysis above, it seems quite difficult to determine the optimal operation based on only local information. We are not quite clear about the approach for other systems. More investigation is needed.
Bibliography


Vita

Yao Hu was in Nanchang, Jiangxi, China.

Education

M.S. in Electrical Engineering, Shanghai Institute of Technical Physics, Chinese Academy of Sciences, Shanghai, China, July, 2007.
B.S. in Electrical Engineering, Fudan University, Shanghai, China, July, 2004.

Publications


