Summer 8-2022

Adaptive Risk Network Dependency Analysis of Complex Hierarchical Systems

Katherine L. Smith
Old Dominion University

ADAPTIVE RISK NETWORK DEPENDENCY ANALYSIS OF COMPLEX HIERARCHICAL SYSTEMS

by

Katherine L. Smith
B.S. December 2008, Old Dominion University
M.S. December 2009, Old Dominion University

A Dissertation Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

MODELING AND SIMULATION ENGINEERING

OLD DOMINION UNIVERSITY
August 2022

Approved by:

Yuzhong Shen (Director)

Rafael Diaz (Member)

Jiang Li (Member)

Hong Yang (Member)

ABSTRACT

ADAPTIVE RISK NETWORK DEPENDENCY ANALYSIS OF COMPLEX HIERARCHICAL SYSTEMS

Katherine L. Smith
Old Dominion University, 2022
Director: Dr. Yuzhong Shen

Recently the number, variety, and complexity of interconnected systems have been increasing while the resources available to increase resilience of those systems have been decreasing. Therefore, it has become increasingly important to quantify the effects of risks and the resulting disruptions over time as they ripple through networks of systems. This dissertation presents a novel modeling and simulation methodology which quantifies resilience, as impact on performance over time, and risk, as the impact of probabilistic disruptions. This work includes four major contributions over the state-of-the-art which are: (1) cyclic dependencies are captured by separation of performance variables into layers which can have different topologies, (2) temporal dependence is modeled using Bayesian networks to allow for incorporation of evidence-based data over time and produce a dynamic model incorporating risk and resilience behavior over time, (3) a combined approach maps from discrete random variables in the risk network to continuous variables in the system network allowing for the propagation of risk throughout the system, and (4) a decomposable architecture allows various components to be represented at different level of detail and overall system reconfiguration to be explored. Applications are provided in supply chain analysis and port logistics to demonstrate the performance and effectiveness of the methodology.

Dedicated to my family with gratitude for their unconditional love and support.

ACKNOWLEDGMENTS

I would like to thank all my colleagues and friends who have helped and supported me along this journey for their guidance and support.

I would especially like to thank my advisor, Dr. Yuzhong Shen, for his support and guidance over the past eight years. In addition, I would like to express my gratitude to Dr. Rafael Diaz for his mentoring and collaboration. Finally, thank you to Dr. Jiang Li and Dr. Hong Yang for serving on my committee and providing feedback and advice.

I would not be where I am today without the unwavering support of my wife, Ana, whom I cannot thank enough, my daughter, Emma, who has been my perpetual champion, and my daughter, Evie, who brings joy to our lives. Lastly, I would like to thank my mom and my siblings for always cheering me on.

NOMENCLATURE

ARNDA	Adaptive Risk Network Dependency Analysis
CDF	Cumulative Distribution Function
DES	Discrete Event Simulation
LSTM	Long-Short Term Memory
MAE	Mean Absolute Error
MSE	Mean Squared Error
PDF	Probability Density Function
RNN	Recurrent Neural Network
SCOR	Supply Chain Operations Reference
SODA	System Operational Dependency Analysis

TABLE OF CONTENTS

Chapter Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

CHAPTER 1

INTRODUCTION

This introductory chapter establishes the need and relevance of the current work and provides an overview of the problem, contributions, and overall organization of the dissertation. Section 1.1 briefly introduces recent events that highlight the importance of this work. Section 1.2 states the overall problem and synthesizes pertinent research questions. Section 1.3 summarizes the purpose of the work and emphasizes the contributions of the dissertation. Section 1.4 provides an outline for the structure of the remainder of the dissertation.

1.1 Complex Systems and Recent Events

Recently, the number, variety, and complexity of interconnected systems have increased in a variety of fields including supply chains due to the advent of the Internet of things, social media, and cyber-physical systems. As system designers and managers struggle to manage the extra risk introduced by this complexity, their efforts are hampered by requirements for practices and implementations that increase productivity while decreasing cost, resulting in fewer opportunities to increase system resilience [1]. Therefore, methods that analyze systems to expose system vulnerabilities and support efforts to enhance resilience are vital to maintain system operation and optimize return on investment with respect to resilience measures.

Recent events have heightened the overall impacts of both short- and long-term disruptions on supply chains worldwide. In particular, misestimations of consumer demand and raw material sourcing constraints have been further complicated by the COVID-19 pandemic and are expected to have lasting effects for years to come [2], [3]. For example, during the early stages of the Covid-19 pandemic, suppliers expected that consumers would scale back spending and cut back on their production of display drivers. When these projections were incorrect, the industry could not respond quickly enough due to a bottleneck in capacity. The ubiquitous nature of computer displays in modern day products has driven this problem to permeate across a number of diverse industries. These and similar effects have drawn attention and garnered opinions from not only executives of large manufacturing companies but also high ranking public officials including the president of the United States [2], [4]. As similar disruptions occur in industries worldwide, an emergent shift in culture and operations is becoming evident which focuses on enhancing overall system resilience by better quantifying and understanding system risk at all levels. Evidence of these changes in practice and policy is emerging from a wide variety of sectors including but not limited to automotive manufacturers, maritime transport, and ports [5], [6]. Overall, these events have highlighted the importance of top level manufacturers having an increased level of understanding of risk and resilience throughout the various supply chains with which they interact including not only the high tier suppliers they interact with most directly, but also all suppliers down to those that supply initial raw materials.

Another consequence of the digital revolution is the introduction of asynchronous, real-time streams of data coming from sensors throughout system networks. This results

in a massive opportunity to leverage data in support of modeling typical and atypical, or disrupted, performance of system components and subsystems. However, this asynchronous convergence of data is often difficult to manage and analyze due to lack of understanding of systems that are lower in the network hierarchy or owned by external entities that lack operational transparency. Therefore, to maximize the ease of implementation of a model, it is important to allow for models of subsystems and components to include parameters describing the level of uncertainty regarding the system behavior.

Understanding the importance of modeling complex networks of systems, this work seeks to address gaps in methodologies from diverse fields of application including supply chain management, logistics in systems such as maritime ports, and systems engineering to provide a methodology capable of not only revealing vulnerabilities throughout these large, complex systems but also simulating how those vulnerabilities change over time when exposed to one or more dynamic disruptions. In addition, most existing methodologies do not allow for cycles in the graphical representation of constituent networks. The current approach includes the ability to model cyclic dependencies, allowing it to model and analyze ripple effects traversing the network in both upstream and downstream directions. A common example of upstream and downstream disruptions in supply chains involves shocks in both supply and demand.

The previously mentioned characteristics help to define the salient qualities relevant to problems in supply chain management, cyber-physical systems, and other complex systems (Fig. 1). The critical areas these characteristics fall into are considerations for

dependencies, data processing, and historic trends. Common requirements for dependencies in complex systems are features and responses that vary with time, edges that form cyclic dependencies, and risks that occur at the same time and have common, potentially unknown, root causes. Data processing concerns are similar to those faced by researchers when dealing with big data. These concerns include high:

- volume: large quantities of data;
- velocity: rapid rates of production or collection, especially in real-time scenarios;
- variety: many different types and storage formats [7].

Finally, historic trends have shown that organizations are trying to design and operate increasingly complex systems at higher levels of efficiency leaving less margin for error. Throughout this work, these characteristics (Fig. 1) will be used to inform model dependencies, requirements for computational complexity based on data set characteristics, and considerations for balancing model complexity with required accuracy.

1.2 Problem Statement

This dissertation presents a novel methodology to study and analyze complex systems of systems represented as layers of networks by combining a dynamic Bayesian network approach and data-enabled network dependency algorithms on directed graphs. Applications will be provided specifically for supply chains and related systems such as maritime ports. This approach addresses the following research questions:

1. How can cyclic features be incorporated into analysis methodologies for system risk

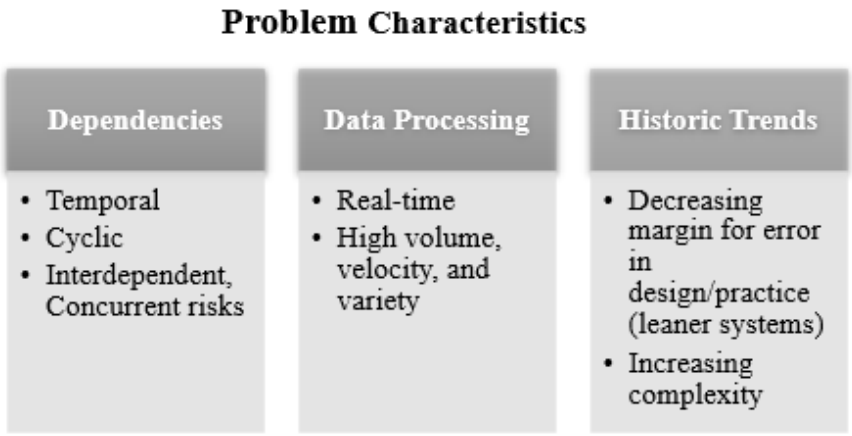


Fig. 1. Characteristics of problems under study.

and resilience?

2. How can the dynamic response of systems be analyzed over time?

3. Can continuous system performance variables be incorporated to better understand the dynamic response?

4. What improvements can be made to enhance the applicability and interpretability of the results from these analyses?

1.3 Purpose and Contributions

In response to the previously presented research questions, this dissertation presents a novel methodology for analysis of risk and resilience in complex systems and networks of systems. While there are existing algorithms from fields such as supply chain management and cybersecurity, this is the first methodology that addresses hierarchical considerations by representing different system features as layers and incorporating temporal dependence

while simultaneously accounting for risk events as features of these causal graphical models. Interactions between risk events are integrated as probabilistic dependencies using a Bayesian network approach to model various types of risk. Additionally, Adaptive Risk Network Dependency Analysis (ARNDA) is a data-driven methodology where behavior is learned implicitly from data rather than operating on rules derived from system behavior as understood by the person implementing the model. While the behavior of the model is learned directly from data, the structure and topology of the underlying networks are fully maintained. This is accomplished by modeling the underlying network as a layered system of interconnected Recurrent Neural Networks (RNNs). The following are specific contributions of the current work:

1. The methodology allows for cyclic dependencies to be captured by (a) separation of output (or performance) variables into layers [8] and (b) inclusion of temporal dependency allowing for modeling of simultaneous forward and backward dependencies, i.e., cycles, inclusion of additional informative variables, and monitoring and assessment of dynamic system response.

2. Temporal dependence is modeled using Bayesian networks to allow for incorporation of evidence-based data over time and produce a dynamic model incorporating risk and resilience behavior over time [9].

3. Discrete random variables are used in a probabilistic graphical model to characterize system risks and subsequently mapped to continuous variables which describe system performance. Effects of these continuous variables are propagated through the system as a whole using network dependency analysis allowing the system to maintain

dependent relationships.

4. Standardized performance variables and resilience metrics are incorporated to support comparisons between systems and system configurations. Further, the decomposable architecture of the layered system of RNNs allows the overall structure to be modified more easily than using a single model for the entire network. This allows dynamic assessment of performance results over time and across various risk scenarios. Results can also be aggregated using stochastic analysis to characterize the response of systems and their components [10].

1.4 Dissertation Organization

The remainder of this dissertation is organized as follows. Chapter 2 provides a review of previous work, an in depth explanation of the approach used to implement the risk model, and motivational analysis for the approach used in this work. Chapter 3 provides details on the development of the RNN-enabled hierarchical network model used for the systems model including the process of fitting this multi-layered model to data and an analysis of the sensitivity of the model to hyperparameters. Chapter 4 presents the full mathematical model, methodology, implementation, and simulation results. Chapter 5 is a case study applying the methodology to operational data similar to data from the Port of Virginia and provides novel findings, analysis, and discussion. This case study specifically addresses considerations for larger, more complex networks including how to quantify and assess system vulnerability and provides details on how results can be visualized and applied. Chapter 6 concludes the dissertation and provides directions for future work.

CHAPTER 2

BACKGROUND AND MOTIVATION

This chapter reviews relevant literature and preliminary analysis to provide background and motivation for the current work. First, existing methodologies for simulating and understanding how disruptions affect supply chains and related systems with respect to risk and resilience are reviewed. Then, an overview of previous work in network dependency analysis as it applies to the design and evaluation of complex systems is provided. Next, theory on probabilistic graphical modeling needed to develop the Bayesian risk model later in the dissertation is provided. The discussion is supported by previous literature, and the chapter concludes with a motivating initial example study conducted by the author in order to inform the design and development of the current work.

2.1 Background and Previous Work

In this section, existing methodologies for simulating and understanding how disruptions affect supply chains and related systems with respect to risk and resilience are reviewed. Then, an overview of previous work in network dependency analysis applied to the design and evaluation of complex systems is provided.

2.1.1 Disruptions, Risk, and Resilience in Supply Chains

Similar to numerous other areas of application, issues and deficiencies in supply chain modeling and analysis have, unfortunately, often been revealed through disasters and

disruptions that have resulted in unanticipated consequences that have affected industries for years following the disturbances. For example in 2011, Japan experienced a magnitude 9.0 earthquake leading to multiple large tsunami which in turn caused a nuclear plant accident [11]. The causal impact of the earthquake on the subsequent disasters was not captured and therefore predictions for these events and their corresponding supply chain disruptions were not correct [12]. Capturing these causal features is critically important as the occurrence of a risk event may increase the likelihood of occurrence of other events. In the event multiple events occur, their effects on complex systems often combine in nonlinear ways. More recently, gross misestimations indicating that a decrease in demand for display chips was likely to occur due to the COVID-19 pandemic has resulted in critical shortages. Many companies anticipated consumers would reduce their spending as stay-at-home orders increased, but this assumption proved false as people started to increase their technology purchases to equip their home offices and entertainment spaces [3]. With more products having integrated displays, this disruption has extended outside of the computing and mobile device industries and affected automotive and appliance industries as well [13]. Companies are attempting to work around the issues by limiting the number of integrated displays in their products, using different types of chips, or delaying the launch of new products, but experts still estimate that it could take a year and a half to see a full recovery [13]. As another example, recent shortages of lithium for lithium ion batteries have compelled auto manufacturers to consider resilience supporting actions all the way down to raw material sourcing in their supply chains [2]. These tactics represent a significant change in supply chain management strategy when compared to just-in-time inventory strategies and other

lean manufacturing approaches.

These supply chain disruptions highlight the need to better understand, model, and analyze the following considerations as they pertain to supply chains and potential for disruption:

1. Causal interactions between disruptive events and potential future disruptions that might be triggered by an initial event.

2. Quantification of multiple scenarios when estimating the predicted impact of events in order to prepare for a range of disrupted responses including nonlinear combinations of disruptions to performance.

3. Understanding of component supply chain entities and their behaviors not only in the tiers closest to manufacturers but also down to raw material suppliers.

There is a significant body of previous work focused on modeling supply chains and their disruptions. In addition, comprehensive literature reviews have been performed with the goal of determining gaps in the the field of research, e.g. [14]. Before discussing specific approaches that address the previous considerations, some general definitions will be provided. Risk in supply chains is understood as the likelihood of a disruption to impact performance [15]. Resilience is commonly defined as the ability of a supply chain, or one of its entities, to tolerate some disruption and the speed with which it is able to return to normalcy [16]. This definition of resilience can be broken down into three different resilience capacities a system can demonstrate, i.e., absorptive, adaptive, and restorative [17]. Per Hosseini *et al.*, these three capacities describe the ability to absorb a disruption without

taking any extraordinary action, to adapt during the occurrence of disruptive events in an attempt to prevent further disruption, and to restore normal operation after performance is lost due to disruption, respectively.

As supply chain research has recognized the importance of considering how disruption moves through supply chains, the study of the ripple, or domino, effect has become more prevalent. The ripple effect is defined as the propagation of a disruption to other entities of the supply chain [18]. When modeling the ripple effect, it is important to capture many supply chain entities while simultaneously modeling causal interactions between risks. Bayesian networks have successfully been applied to model the ripple of disruptions through complex systems such as supply chains [19] and produced quantified measures of resilience [20]. One benefit of these models is that they provide a causal representation for networks. Another is network characteristics can be learned from limited data, or subject matter expertise, incorporated in the form of priors with the opportunity for additional data to be incorporated as it becomes available increasing the accuracy of the model [21]. One problem with these approaches is that both exact and approximate algorithms for forward post-learning inference are NP-hard which directly limits efforts to evaluate complex systems in real-time [22]. Algorithmic solutions to NP-hard problems are believed to require more than polynomial time [23]. This deficiency is especially restrictive as it has become increasingly important to model systems with higher levels of complexity. Examples of increasing complexity include the ability to model a greater number of entities from raw material suppliers to consumers, real time computation using live data, and concurrent evaluation of multiple, performance-increasing interventions. An important effect of this

limitation has been to restrict researchers to studying small or less complex supply chains while acknowledging that additional work is needed [19], [24]. Additionally, most applications of graph theory, including Bayesian networks [19], and other approaches [1], neglect the temporal component of disruption propagation. Approaches to overcome this limitation include dynamic Bayesian networks and Markov chains [24]. As a final note, it has been acknowledged that there are limited methods that provide compelling visualizations to the simulated outputs of models addressing the ripple effect [14].

2.1.2 Vulnerability in Complex Systems

Vulnerability has varied definitions across different fields. Sometimes, these definitions can be perceived as conflicting. However, when carefully considered, they can be combined to build a general, flexible definition for vulnerability and factors that influence it.

First, consider a selection of definitions for vulnerability from a variety of domains. Definitions from various technical and non-technical fields are shown in Table 1. The two main components of each of these definitions are as follows: (1) Vulnerability is described as a quality, feeling, condition, or weakness, all of which are states, and (2) vulnerability is experienced when susceptible to attack, harm, uncertainty, risk, emotional exposure, hazards (and their impacts), or threat. Parts of the definitions omitted by the previous statement omit (1) particular areas or groups that could be affected by the vulnerability and (2) defining parameters or characteristics of those systems which increase or decrease vulnerability. These two items are domain specific and therefore will be excluded from the general definition developed and utilized in this work.

TABLE 1. Vulnerability definitions for various domains.

Domain	Author	Definition	Source
General	Oxford English Dictionary	"The quality or state of being exposed to the possibility of being attacked or harmed, either physically or emotionally."	[25]
Human Behavior	Brené Brown	"The feeling we get during times of uncertainty, risk, or emotional exposure"	[26]
Disaster Recovery	United Nations Office for Risk Reduction	"The conditions determined by physical, social, economic, and environmental factors or processes which increase the susceptibility of an individual, a community, assets or systems to the impacts of hazards."	[27]
Computer and Cybersecurity	National Institute of Standards and Technology	"Weakness in a system, system security procedures, internal controls, or implementation that could be exploited or triggered by a threat."	[28]

In this work, a vulnerability will be not only defined but also quantified in Chapter 5 as a level of exposure to harm or threat. Therefore, *vulnerability* is a quantity that describes the degree to which a system or entity is exposed to the risk of disruption and has the potential to be resilient to change. Higher levels of vulnerability imply that a system is exposed to more risks and is more likely to experience greater effects from those risks should they occur. The implication of this definition is that vulnerability can be evaluated with respect to risk, resilience, and uncertainty. Though only risk and resilience are stated in the definition, uncertainty will be included in the formulation because uncertainty at any stage of the vulnerability assessment can cause errors in the estimates of change that will occur

in the system (Eq. 1).

$$Vulnerability = f\left(risk, resilience, uncertainty\right) \tag{1}$$

Though the definition of vulnerability above may seem over simplified, the process of assessing and quantifying risk, resilience, and uncertainty for even a single entity can be incredibly challenging. There have been many academic studies related to determining drivers of factors and their impacts on complex systems in a variety of fields. Assessing the vulnerability of communities that are exposed to risks from natural disasters or natural hazards, such as climate change, is usually performed using vulnerability indices [29]–[31]. These indices score individuals or groups to provide values for individual variables, called indicators, that are aggregated to provide a single numerical result. Due to variations in populations and no consensus on optimum processes to create indices, overall creation of these metrics is a subjective process that can be highly dependent on decisions made by developers [30]. In the face of climate change, increasing human population, and other drivers of decreasing biodiversity, identifying vulnerabilities in the transmission of diseases from animals to humans has been under study in recent years [32], [33]. This highlights a need to compare studies and related interventions using standardized metrics for risk, resilience, and vulnerability in order to aggregate and compare results across researchers worldwide. The COVID-19 pandemic has only highlighted the importance of accurately assessing the risks associated with this transmission and exploring ways to increase human resilience against these diseases [34].

Quantifying vulnerability across domains is a difficult and important task implying that understanding and assessing gaps in vulnerability should be a research priority. Often

existing research focuses on small networks limiting system complexity, such as two-node networks in supply chain research [35]. Additionally, most work fails to account for simultaneous disruptions including those that propagate through the system in different directions [36], [37]. These facts further justify the use of methods, such as Bayesian networks, that can model causal features in the modeling of risk and resilience [19], [20].

2.1.3 Network Dependency Analysis

To balance the burden of computational complexity encountered while using Bayesian network models, a less computationally expensive model is sought. A method that has been used to model dependencies in complex systems while requiring a significantly lower computational expense is network dependency analysis methodology. These models trace back to Leontief Input-Output models [38]. It is noteworthy that input-output models have been applied to assess effects of supply chain disruption [39]. Methodologies have been extended from the traditional input-output models to conduct analysis of risk for infrastructures [40]. Then, these methodologies were extended by researchers in systems engineering to model risk in complex systems, specifically for capability portfolio risk management [41]. These methods have been further extended and applied as System Operational Dependency Analysis (SODA) which enabled a more robust characterization of the operational performance of systems, including applications to cybersecurity and aerospace systems [42], [43]. In this methodology, the level at which a given node functions, namely its operability, can be calculated as a piecewise, weighted, linear combination of the operability values for the dependent nodes and a parameter representing its own internal health. A benefit of this approach is that calculating these piecewise linear relationships imposes a lower computational

burden than inference calculations using Bayesian networks. Though simple, these weights and parameters have intuitive meaning based on the relationships between the operability values of dependent nodes. Deficiencies of the SODA methodology when applied to supply chain research include that they are defined to be acyclic and are not temporally dependent. Though [43] does mention time dependence, it is with respect to evaluation of behavior for reduced internal health in nodes at some time in the future rather than an analysis of the system's dynamic response to disruption over time. These are the issues that the current work focuses on in order to develop a more comprehensive model to simulate supply chain behavior and its disruptions.

2.2 Probabilistic Graphical Modeling

In this section, the events that pose a risk to theoretical systems under study are modeled as a directed graph. Recall that risk is defined as the likelihood that the system or one of its subsystems or component entities will be disrupted. Specifically, this dissertation is concerned with disruptions that have negative effects. However, the framework is kept general enough that events causing positive effects can be modeled as well.

2.2.1 Directed graphical model

To represent a system as a hierarchical network, relevant components and sub-components must be represented as nodes, and these nodes must be related to each other by dependencies which will be implied by connecting the nodes with arrows. The direction of the arrow indicates the nature of the dependency relationship. In this work, hierarchical networks will be represented as directed, connected graphs. The mathematical definition of

a graph is presented next. It is similar to the definition from any textbook presenting an introduction to graphs and their theory such as [44]. A directed graph, commonly referred to as a digraph, $G = (N, E)$, is defined as a set of nodes, N, connected by a set of edges, E. Individual nodes will be referred to using a subscript, e.g., n_i and n_j are nodes i and j, respectively. Edges will be referred to by two ordered subscripts indicating the nodes where the edge begins and ends, e.g., $e_{i,j}$ is the directed edge that begins at node i and ends at node j. In probabilistic graphical models, the direction of the edges implies dependence. The fact that nodes i and j are connected by edge, $e_{i,j}$ implies that the behavior of n_j is dependent on n_i, so we can say that nodes i and j belong to graph, G, and there is a directed edge between them as in equation (2) or shown graphically in Fig. 2(a). Node i is the source of edge $e_{i,j}$ and node j is its target. Note that it is not common to label edges (see Fig. 2(b)) as their labels can be inferred from the labels of the nodes connected by the edge and their order can be inferred from the direction of the connecting arrows. In Fig. 2(b), the behavior of node j is dependent on the behavior of both node i and node k since the three are connected with edges $e_{i,j}$ and $e_{k,j}$.

$$\begin{gathered} G = (N, E) \\ n_i, n_j \in N \; and \; e_{i,j} \in E \end{gathered} \tag{2}$$

(a) (b)

Fig. 2. Two networks: (a) one with labeled edges and (b) the other without.

2.2.2 Risk space as a discrete probability space

Let a risk space be defined as a probability space where the events under consideration present some risk to a system. For clarity, recall risk is defined as the likelihood that the occurrence of a given event will disrupt the behavior of all or part of the system under study. The definition of a probability space can be found in nearly any textbook on probability theory, such as [45]. This work will use the following definition and notation throughout: a *probability space* is a triple (Ω, E, P) where Ω is the sample space of all possible outcomes, E is a σ-algebra over Ω that specifies how the sample space can be divided into a set of events, and P is a measure on (Ω, E) which maps events in E to the set of real numbers.

For additional clarity, consider the simple example of tossing a coin. The coin can either land on heads or tails which will be denoted H or T, respectively. An example outcome would be that the coin landed on heads which would be denoted as H. Enumerating all such possibilities gives the sample space of all possible outcomes:

$$\Omega = \{H, T\}. \tag{3}$$

Then, the largest σ-algebra, $\mathcal{F}$, is the set of all combinations of outcomes from Ω, [46]:

$$\mathcal{F} = \{\emptyset, \{H\}, \{T\}, \{T, H\}\}. \tag{4}$$

Finally, the measure P can be defined on (Ω, E) to indicate the likelihood of each set of events from $\mathcal{F}$. The $P(A)$ where $A \in \mathcal{F}$ is the probability that one of the outcomes in A will occur when the coin is flipped. Therefore, for this simple example, the probability for each event can be used to define the measure, P as follows:

$$\begin{aligned} P(\emptyset) &= 0. \\ P(\{H\}) &= \tfrac{1}{2}. \\ P(\{T\}) &= \tfrac{1}{2}. \\ P(\{H, T\}) &= 1. \end{aligned} \tag{5}$$

Once a probability space is defined for a problem of interest, it is common to define one or more random variables to study further. From [45], a random variable, X, is a mapping from one probability space to another. Another equivalent interpretation is to consider that a random variable, X, is mapping from a set of events to a set of numeric values between 0 and 1 with each value indicating the likelihood of the corresponding event. Since all random variables are measurable functions that will map from sets in the σ-algebra to values on $[0, 1]$, any two random variables defined for the same σ-algebra can be related through the sets in the σ-algebra using the transitive property. This implies the definition of a random variable as a mapping from one probability space to another. A mapping for the coin toss

example would be $Y =$ number of heads. For Y, the following is true:

$$\begin{aligned} P(Y = 0) &= \tfrac{1}{2}. \\ P(Y = 1) &= \tfrac{1}{2}. \\ P(Y \geq 2) &= 0. \end{aligned} \tag{6}$$

The simplest mapping is an indicator which outputs a value of one when a given event has occurred and zero otherwise. When analyzing these random variables and how they interact, it becomes important to determine whether they are independent or dependent events.

Now an example related to risk events will be explored to elucidate the previous definitions. Consider a public health event and a merger, acquisition, or reorganization as two risk events that can affect the ability of a company to function normally. The risk space can be defined as all possible combinations of states for these two risks that the company may face. The σ-algebra will include all possible combinations of outcomes which are occurrences of a public health event, merger/acquisition/reorganization event, both, and neither. The probability measure, P, will map each event to a real number representing its likelihood. This example can be visualized as shown in Fig. 3(a) where the σ-algebra is used to show all the ways that the risk space can be divided into events of interest, E. There are studies in the literature, e.g., Kooli and Lock [47], that show that public health events do affect merger/acquisition/reorganization events. Therefore, it is assumed that the "merger/acquisition/reorganization event" is dependent on the "public health event". An example scenario where this might occur would be a serious outbreak of disease resulting in weakening a company through reduced labor capacity. In order to overcome this hardship and avoid bankruptcy, the company might merge with another company that was not as

severely affected.

Given this risk space, there are many random variables that can be defined. For example, an indicator function could be defined for the event "public health event" having a value of one when an outbreak of disease occurs and zero otherwise. Since an indicator function is used, it would be necessary to define a threshold for defining what it means to have an outbreak, as in, for example, more than a certain number of individuals infected or hospitalized. Similarly, a second random variable could be defined as an indicator function for "merger/acquisition/reorganization event". From Fig. 3(a), these events are not independent, and the random variables defined in the preceding way will not be either. Fig. 3(b), shows one possible graphical network representing the dependence of these random variables.

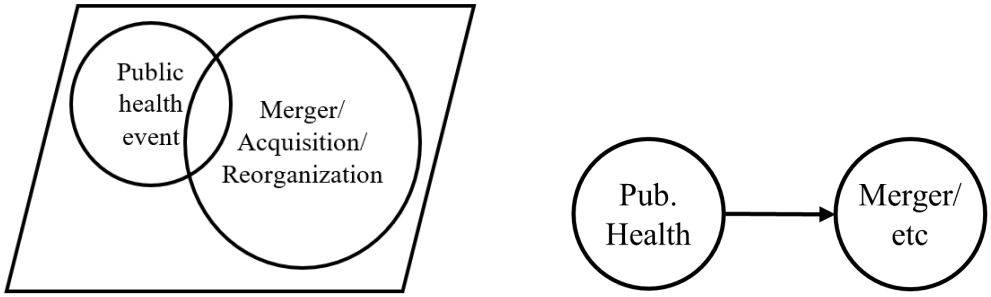


Fig. 3. Simple risk space example showing (a) division into events and (b) random variable dependence.

2.2.3 Discrete Bayesian networks in risk spaces

The next step is to define a process for developing a risk space and learn its behavior

for a system of interest. To this end, a general risk space is defined where three risks, i.e., Risk a, Risk b, and Risk c, may or may not occur at some time in the future (Fig. 4). The resulting set of outcomes, Ω, produces a set of events. The fully enumerated set of events is

$$\Omega = \{\emptyset, (a), (b), (c), (a,b), (a,c), (b,c), (a,b,c)\}, \tag{7}$$

where each tuple, that is list of elements in parentheses, indicates that the events corresponding to the risks listed are occurring. For example, (a, c) indicates that Risks a and c occurred while the event corresponding to Risk b did not. The σ-algebra would include all possible subsets of Ω which will not be listed since there are $2^8 = 256$ subsets of a set containing 8 elements. Even for a small number of risks, in this case 3, it can be seen that the resulting possibilities for sets of outcomes quickly grows very large. Particular members of the σ-algebra that may be of interest are all outcomes where a particular risk occurs. For Risk c, this would be $\{(c), (a,c), (b,c), (a,b,c)\}$.

In order to represent this risk space using a Bayesian network, random variables must be defined. For simplicity, let each random variable, X_i, be defined as an indicator as follows:

$$X_i = \begin{cases} 1 & Risk\ i\ occurs. \\ 0 & otherwise. \end{cases} \tag{8}$$

Then, one possible Bayesian network representation for the three resulting random variables is shown in Fig. 5. This configuration implies that random variable X_c is directly affected by X_a and X_b is directly affected by both X_a and X_c based on the direction of the arrows on the connecting edges in the graph. In a causal Bayesian network, the direction of these

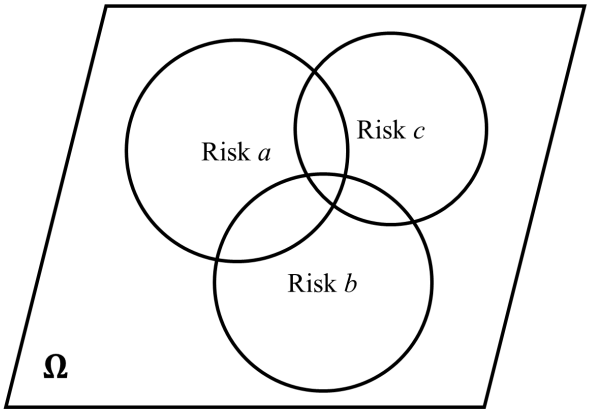


Fig. 4. Risk space with three risks.

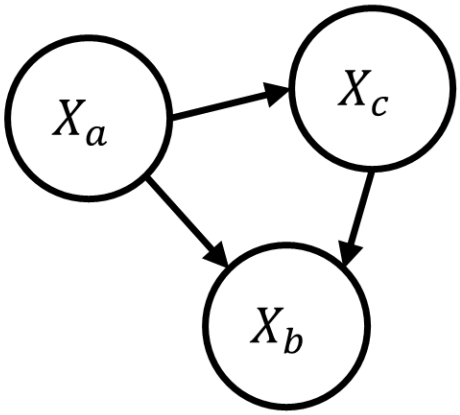


Fig. 5. Sample Bayesian network representation.

arrows implies causality. Evidence or knowledge of the relationships is required in order to determine if they are in fact causal and not just correlations. Using a Bayesian network representation allows a model to incorporate knowledge regarding the dependence of random variables and learned knowledge from data into a single abstraction of the system. There are many well-known algorithms for defining and learning network structures and features such as the one presented in [48]. It is outside the scope of this work to provide the details of these algorithms. However, it is prudent to provide an example of how the resulting Bayesian network can be interpreted.

To simplify notation and provide less cluttered equations, from this point forward, the X in the notation indicating random variable X_i will be dropped so that the variable can be referred to simply as i. One of the key advantages to using Bayesian networks is their ability to propagate evidence in both the forward and backward directions. Using the simple example in Fig. 5, the joint probability distribution can be expanded and simplified using the chain rule of probability as follows:

$$p\left(a,b,c\right) = p\left(b|c,a\right)p\left(c,a\right) \quad = p\left(b|c,a\right)p\left(c|a\right)p\left(a\right) = p\left(a\right)p\left(b|c,a\right)p\left(c|a\right). \tag{9}$$

Based on the expansion of the joint probability distribution above, the following conditional probability tables (defined in Tables 2, 3, 4) provide enough information to calculate the general probabilities for each set of events in Equation 7.

TABLE 2. Probability distribution for Risk a.

	$p(a)$
T	0.80
F	0.20

TABLE 3. Conditional probability distribution for Risk b.

	Risk a	Occur		Not Occur	
	Risk c	Occur	Not Occur	Occur	Not Occur
Risk b	Occur	0.80	0.60	0.30	0.10
	Not Occur	0.20	0.40	0.70	0.90

TABLE 4. Conditional probability distribution for Risk c.

	Risk a	Occur	Not Occur
Risk c	Occur	0.90	0.30
	Not Occur	0.10	0.70

Given that nothing is known about the occurrence of Risks b and c, the probability of Risk a occurring is directly equal to 0.8 from its probability table. For Risks b and c, it is necessary to sum over all values for the risks they are dependent on as follows: (Note: O = Occur, NO = Not Occur)

$$\begin{aligned} p\left(b=O\right) &= \sum_{a,c} p\left(b=O|a,c\right) p\left(a\right) p\left(c|a\right) \\ &= p\left(b=O|a=O,c=O\right) p\left(a=O\right) p\left(c=O|a=O\right) \\ &\quad + p\left(b=O|a=O,c=NO\right) p\left(a=O\right) p\left(c=NO|a=O\right) \\ &\quad + p\left(b=O|a=NO,c=O\right) p\left(a=NO\right) p\left(c=O|a=NO\right) \\ &\quad + p\left(b=O|a=NO,c=NO\right) p\left(a=NO\right) p\left(c=NO|a=NO\right) \\ &= 0.80 \cdot 0.80 \cdot 0.90 + 0.60 \cdot 0.80 \cdot 0.10 + 0.30 \cdot 0.20 \cdot 0.30 + 0.10 \cdot 0.20 \cdot 0.70 \\ &= 0.656. \end{aligned} \tag{10}$$

$$p\left(b=NO\right) = 1 - p\left(b=O\right) = 1 - 0.656 = 0.344. \tag{11}$$

$$\begin{aligned} p(c=O) &= \sum_{a} p\left(c=O|a\right) p\left(a\right) \\ &= p\left(c=O|a=O\right) p\left(a=O\right) + p\left(c=O|a=NO\right) p\left(a=NO\right) \\ &= 0.9 \cdot 0.8 + 0.3 \cdot 0.2 = 0.78. \end{aligned} \tag{12}$$

$$p(c=NO) = 1 - p(c=O) = 1 - 0.78 = 0.22. \tag{13}$$

From the above analysis, the probability that Risks b and c will occur are 0.656 and 0.78, respectively. For one last example, assume that evidence is available showing that Risk a

occurred. This implies $P(a = O) = 1$ and $P(a = NO) = 0$. Based on the evidence that Risk a has occurred, the probability of Risk b increases from 0.656 to 0.78 based on the following calculation:

$$\begin{aligned} p(b = O|a = O) &= \sum_c p(b = O|a = O, c)\, P(a = O)\, p(c|a = O) \\ &= p(b = O|a = O, c = O)\, p(a = O)\, p(c = O|a = O) \\ &\quad + p(b = O|a = O, c = NO)\, p(a = O)\, p(c = NO|a = O)\, s \\ &= 0.80 \cdot 1 \cdot 0.90 + 0.60 \cdot 1 \cdot 0.10 \\ &= 0.78. \end{aligned} \tag{14}$$

The probability of Risk c increases from 0.78 to 0.90 which can be deduced directly from Table 4. These increases of approximately 19% and 15%, respectively, would be expected to lead to significant changes in the expected behavior of an overall model that was dependent on these risks. This simple example illustrates not only the methodology for propagating risk through a Bayesian network but also the criticality of characterizing dependencies between risks.

2.3 Preliminary Analysis

The following subsections provide preliminary analysis to further support and inform the design and development of the current work. A theoretical supply chain leading up to the process of installing a watertight door on a ship will be developed. Discrete event simulation will be used to model and simulate the behavior of the supply chain. Finally, network dependency analysis models based on the SODA methodology at two levels of complexity will be fit to the results. The subsequent findings will motivate the improvements made in

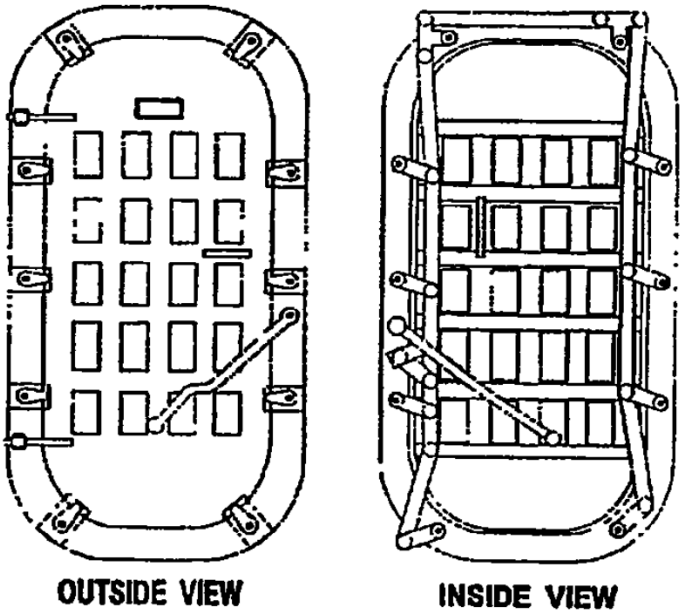


Fig. 6. Drawing of a quick-acting watertight door. Originally published as Figure 600-2.1 in [51].

the development of the methodology in Chapters 3 and 4.

2.3.1 Initial Example and Motivation

A theoretical supply chain for a watertight door on a Naval vessel will be used as an example throughout the remainder of this chapter. Watertight doors are special maritime closure devices that have specialty latches and gaskets to prevent the flow of water from one shipboard compartment to another [49]. These doors have been the subject of multiple safety alerts from the United States Coast Guard [50]. Examples of drawings and images of these maritime closures are shown in Figs. 6 and 7, respectively. The simplified supply chain that will be used for these examples is shown in Fig. 8.

Fig. 7. Picture of a quick-acting watertight door. Originally published by Pier Side Supply in [52].

Assumptions regarding watertight door installation for this simplified supply chain are consistent with the *Naval Ships' Technical Manual on Structural Closures* [51]. In Figure 8, the flow of products is mapped from raw material suppliers of steel and brass to the final installation process on board a vessel under construction. Prior to installing the door, the installer must cut through the wall of the ship, also known as the bulkhead. Then, two items are installed. The first are steel chocks which are blocks of metal welded in place to strengthen the cut bulkhead into which the door is installed. The second is the door assembly itself. The door assembly is composed of a frame made from steel and the door made from steel and brass.

In this example, a naval vessel supplier has recognized that installation of watertight doors is on the critical path for the overall ship construction process and therefore desires

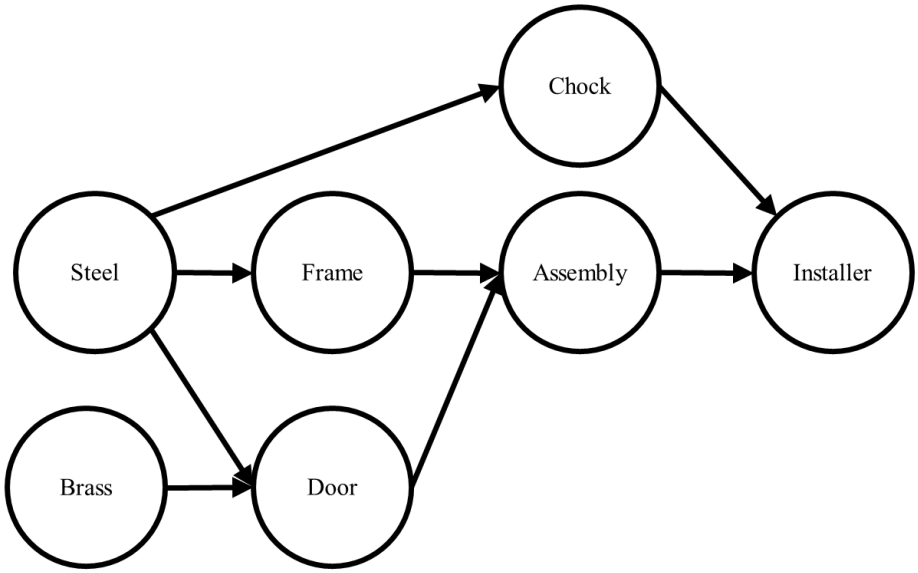


Fig. 8. Simplified network representing the supply chain for a watertight door on a ship.

to better understand vulnerabilities in the relevant supply chain entities. In reality, the listed components can be broken down further, and additional raw materials are required. However, this simplified model contains enough complexity to explain all aspects of the current methodology without unnecessary complexities.

2.3.2 Evaluation of System Operational Dependency Analysis (SODA)

Network dependency analysis provides an analytical approximation of the overall complex behavior of systems with dependencies. As a competing methodology, discrete event simulation can capture additional complexities by modeling them at a user-specified level of detail. However, these higher levels of detail can drastically increase simulation run times as well as the time and level of expertise required to build a specific model. With

this in mind, a study has been performed to assess whether or not sufficient detail can be captured by a network dependency analysis methodology, specifically SODA [43], by comparing to a more detailed discrete event simulation model.

In order to assess the ability of SODA to capture the characteristics and behaviors of a supply chain model, a three node submodel, with the Chock, Assembly, and Installer nodes, and the full seven node model were implemented as discrete event simulations in Arena [53]. The three node sub-model and its Discrete Event Simulation (DES) implementation are shown in Figs. 10 and 11, respectively. The seven node model and its DES implementation are shown in Figs. 8, 12, and 13. The implementation shown follows the Supply Chain Operations Reference (SCOR) model [54] similar to the approach by Carvalho, et al. [55]. The SCOR methodology breaks supply chain operations for each firm into phases, i.e. Plan, Source, Make, and Deliver (Fig. 9). The blocks showing the implementation for the seven node network had to be split into two separate figures in order to show sufficient detail. In Figs. 12 and 13, the colors of the blocks correspond to the distance of each production node as measured from the target node where distance is defined as the number of edges that must be transversed to move from the target node to a given production node. The target node representing the watertight door installation process is blue. Moving back through the network, the first tier suppliers, i.e., chock production and assembly production, are yellow. The second and third tier suppliers are shown in orange and purple, respectively.

All suppliers were assumed to implement an (s,S) inventory policy. Organizations following an (s, S) inventory policy reorder stock when their inventory is less than s units and continue to reorder until their inventory reaches S units [56]. If current inventory

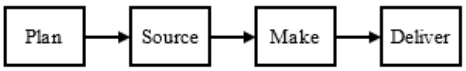


Fig. 9. Overall supply chain operations research (SCOR) model.

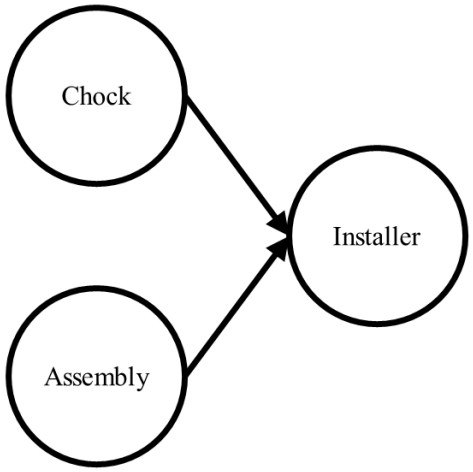
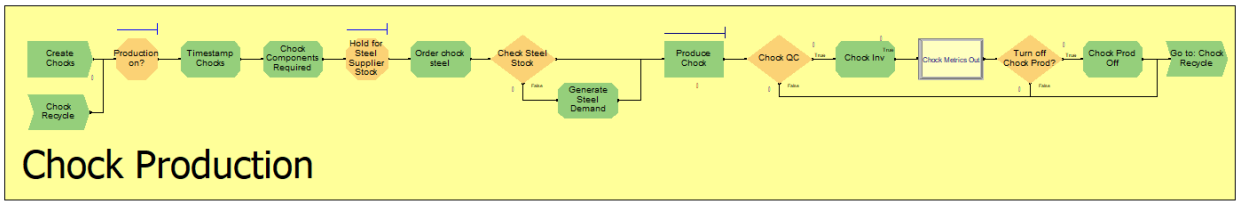


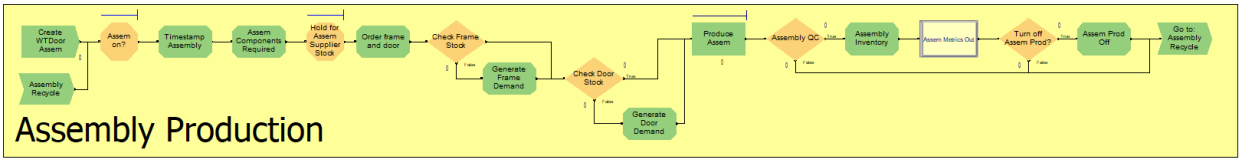
Fig. 10. Watertight door network represented as only three nodes.

was insufficient to fulfill a downstream order, the customer entity would wait for sufficient inventory to be produced. In order to implement different levels of performance, quality control nodes inspected each component produced and discarded those that were defective based on a two-way by chance decision with uniform probability. These probabilities were randomly generated and then held constant for each simulation run. This allowed SODA parameter values to be calculated for known values of internal node health.

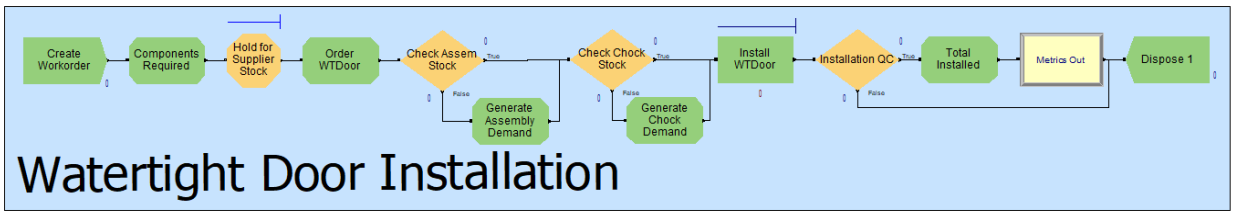
It only requires a visual assessment to develop an initial understanding of the difference in complexity between the SODA model and the discrete event simulation model.



(a)

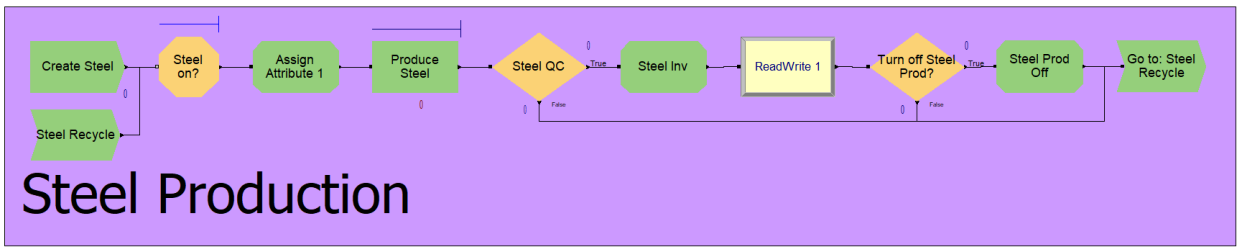


(b)

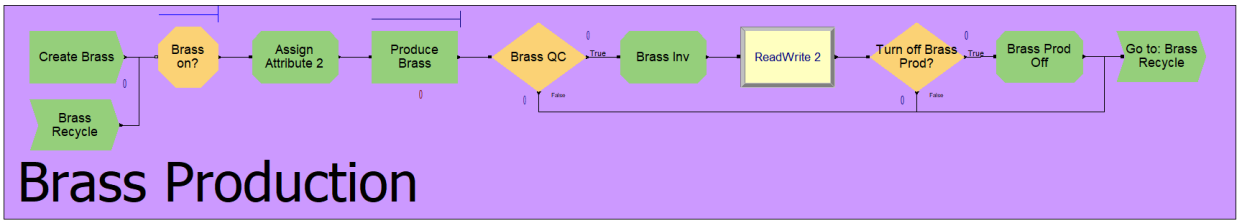


(c)

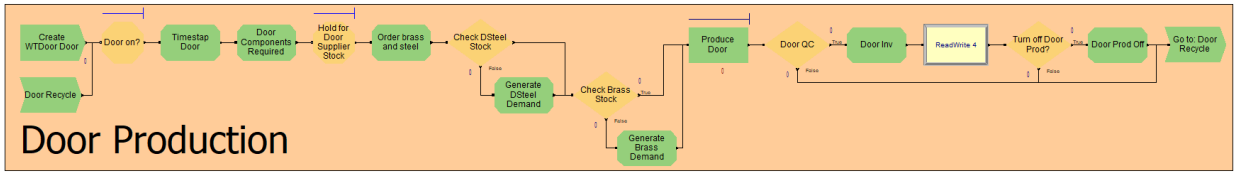
Fig. 11. Arena implementation for discrete event simulation of three-node watertight door example (Fig. 10) including blocks for (a) chock production node, (b) assembly production node, and (c) the overall installation node.



(a)



(b)



(c)

Fig. 12. Arena implementation for discrete event simulation of the first three nodes for the seven-node watertight door example (Fig. 8) including blocks for (a) steel production node, (b) brass production node, and (c) door production node, (d) frame production node, (e) chock production node, (f) assembly production node, and (g) the overall installation node.

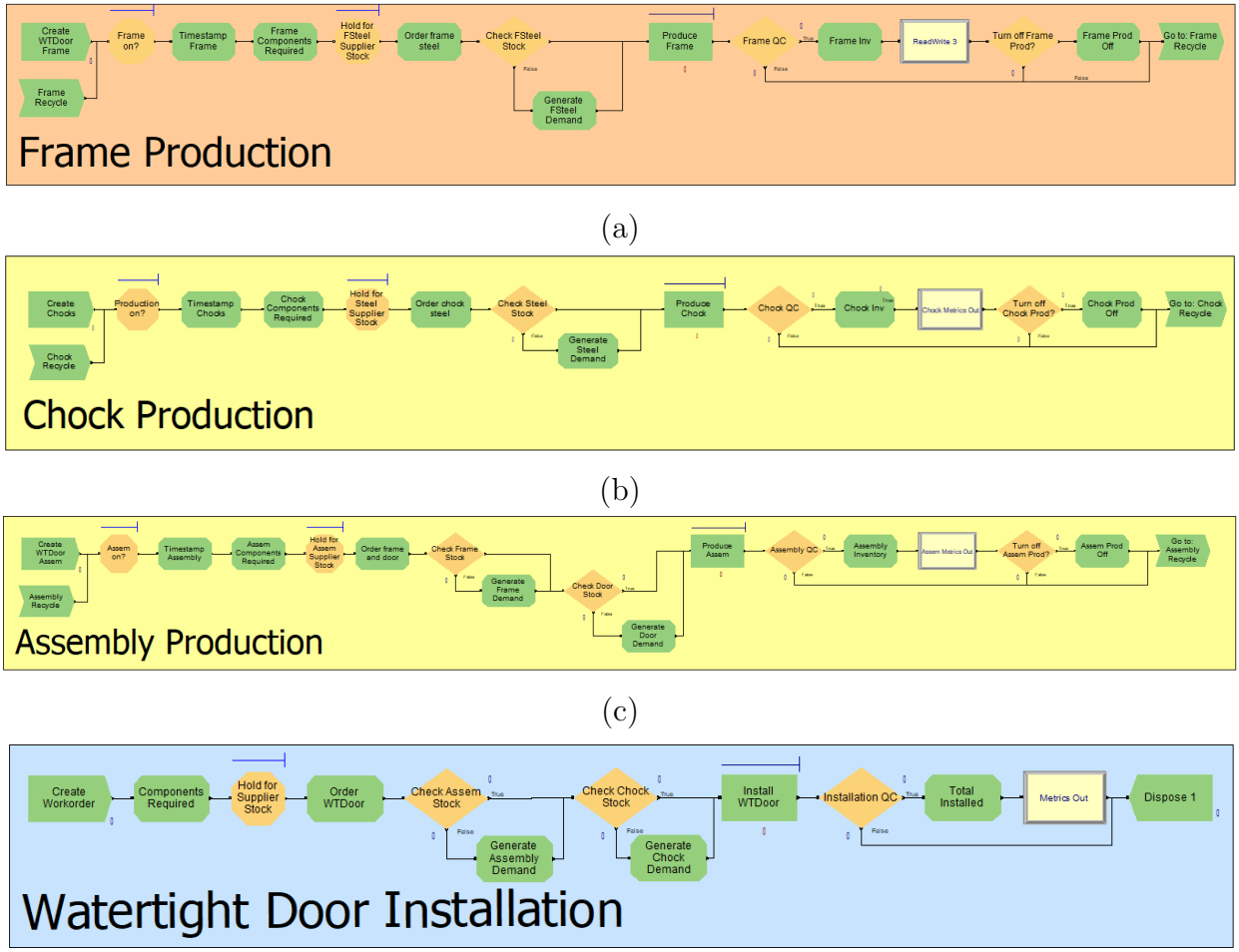


Fig. 13. Arena implementation for discrete event simulation of the remaining four nodes for the seven-node watertight door example (Fig. 8) including blocks for (a) frame production node, (b) chock production node, (c) assembly production node, and (d) the overall installation node.

The 7 (Fig. 8) and 3 node (Fig. 10) SODA models are implemented as 91 and 43 module, or node, discrete event simulation models, respectively. Certainly, increased complexity should not have an exclusively negative connotation as implementing additional detail can enable the model to capture behavior that is more representative of the real system. However in this case, the goal is to capture the overall behavior of the model while keeping the computational complexity manageable.

SODA defines specific parameters and update functions as follows. The output parameter of each node, j, O_j, is defined to be operability, $O_j \in [0, 100]$. Operability is a measure of how well a node is performing. The internal state parameter of each node, E_j, is called self-effectiveness, $E_i \in [0, 100]$.

For root nodes:

$$O_j = E_j. \tag{15}$$

For all other nodes, n_j with input, or feeder nodes $n_k, k \in K$,

$$\begin{aligned}
W_k &= \frac{1}{n-1} \sum_{i \in K, i \neq k} O_i \\
O_{kj}^C &= \frac{100}{\gamma_{kj}} O_k + W_k^{\lambda} \left(100 - \beta_{kj}\right) \\
O_j^C &= \min_{k \in K} \left(O_{kj}^C\right) \\
O_{kj}^S &= \alpha_{kj} O_k + \left(1 - \alpha_{kj}\right) E_j \\
O_j^S &= \frac{1}{n} \sum_{k \in K} O_{kj}^S \\
O_j &= min\left(O_j^S, O_j^C\right).
\end{aligned} \tag{16}$$

Each of the remaining parameters that were not described previously are weights and have a descriptive connotation that can be related to the behavior of the dependencies. In

particular, α_{ij} is referred to as the strength of dependency and acts as the weight used to calculate the target node operability as a weighted average of its own self-effectiveness and the operability of the source node when the source node is performing relatively well. When the source node is performing relatively poorly, the performance of the target node depends on β_{ij}, the criticality of dependency, and γ_{ij}, the impact of dependency. The criticality of dependency is defined as the amount the target node operability decreases from 100 when the source node operability is 0 and can be interpreted as 100 minus the intercept for the vertical axis if the operability curve is graphed. The impact of dependency allows the operability of the target to increase more quickly than the operability of the feeder node in order to decrease the range of the zone where the self-effectiveness of the target node does not contribute to its own operability. Further details on the full SODA methodology can be found in Guariniello and DeLaurentis [43].

Before fitting the SODA model to the data generated by the discrete event simulation, it is interesting to look at heatmap plots of the correlation coefficients for the number of components produced by each node in the model. The Pearson correlation coefficient is one way to quantify how two variables are related. For two variables, x and y, each with the same number of observations, the Pearson correlation coefficient is defined as

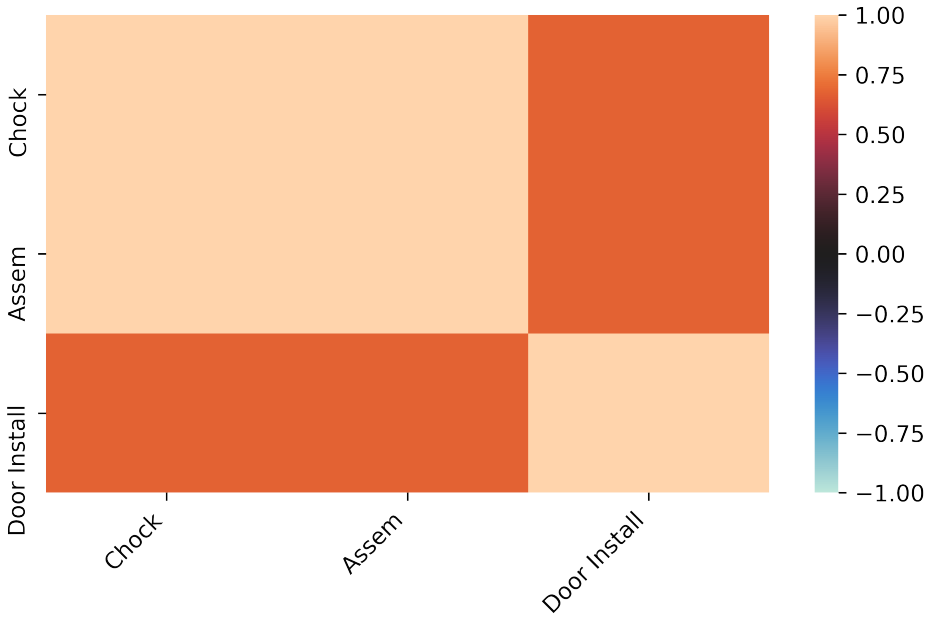
$$r = \frac{\Sigma\left(x_i - \overline{x}\right)\left(y_i - \overline{y}\right)}{\sqrt{\Sigma\left(x_i - \overline{x}\right)^2 \Sigma\left(y_i - \overline{y}\right)^2}}, \tag{17}$$

where $\overline{x}$ and $\overline{y}$ are the means of all samples for x and y, respectively [57]. The value for the Pearson correlation coefficient, r, takes a value on the interval from -1 to 1, inclusive. A value of -1 indicates a perfect, negative linear relationship. A value of 1 indicates a perfect,

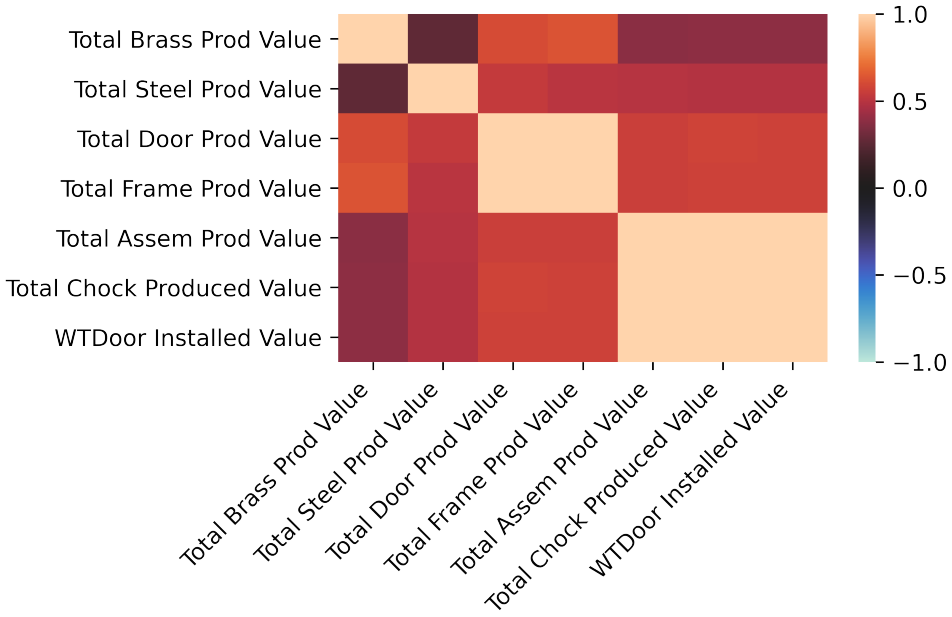
positive linear relationship. Finally, the value 0 shows there is no correlation between the variables. Fig. 14 shows the Pearson correlation coefficients for the correlation matrix for the (a) 3 node model and (b) 7 node model. It makes sense that all correlations are positive since the supply chain model forms a connected graph and the number of items a given node can produce should be roughly linearly related to the performance of the nodes it is dependent on. It can also be seen that nodes that are in the same supply chain tier, as measured as distance from the installer node at the ship manufacturer, are highly correlated except those in the lowest tier which is expected since these represent independent inputs to the system. Additionally, looking back through the supply chain, starting with the target installation node, it can be seen that nodes further upstream are less correlated with the target node.

Next, attempts were made to fit the SODA model to the simulated data generated by the discrete event model. For the 3 node model, number of chocks and assemblies produced were used as input variables while the number of watertight doors installed was the corresponding output. The key characteristic of the methodology to keep in mind for the purpose of this example is that the performance, or output, of a given node is defined as a piecewise non-linear combination of the outputs of the nodes it is dependent on and its own internal health value.

Nonlinear regression with a user-defined loss function was used to approximate the coefficients for the SODA model. The loss function minimized involved the ℓ^2-norm, the definition for which can be found in any mathematical analysis textbook such as [58]. The



(a)



(b)

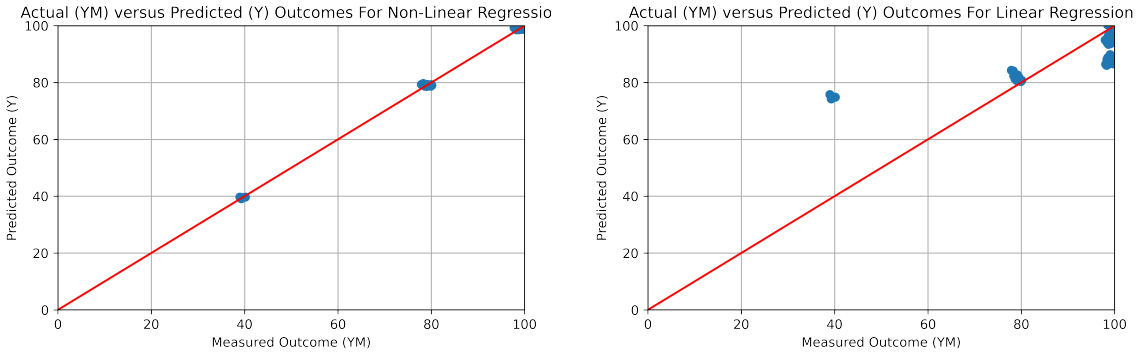
Fig. 14. Heatmaps showing Pearson correlation coefficients for units produced for (a) 3 node and (b) 7 node simulation study.

total loss function was defined as:

$$L = \frac{\| \overrightarrow{y} - \overrightarrow{y_m} \|_2^2}{\| \overrightarrow{y_m} \|_2^2} = \frac{(\overrightarrow{y} - \overrightarrow{y_m})^2}{\overrightarrow{y_m}^2}, \tag{18}$$

where $\overrightarrow{y}$ and $\overrightarrow{y_m}$ are the estimated values, in units produced, by the SODA model and the observed values from the Arena simulation, respectively. After fitting to the non-linear SODA model and a linear model, for comparison, the predicted outcome, y was plotted against the measured outcome from the Arena simulation, y_m. If the model produces a perfect fit, these values should be equal for every sample and, when plotted, fall along the $y = y_m$ line shown in red in Fig. 15. These plots show the SODA model (a) is a superior model for the data when compared with a linear model (b). Looking at the correlation coefficients, the SODA model has a value of 0.998 while the linear model has a value of 0.471 indicating that the non-linear SODA model provides a superior fit since its correlation coefficient is much closer to the maximum value of 1. It is also worth mentioning that the points appear to form three clusters corresponding to the SODA model which is a three parameter piecewise linear model. Since adding additional nodes introduces more complexity with respect to the network topology and the number of parameters, these clusters will not be visible in later analyses.

Finally, the SODA model was fit to the entire supply chain network. Coefficients were estimated at each tier of the supply chain (Fig. 16). Fig. 17 shows the results of this analysis as the predicted target node productivity plotted against the values from the simulation (a, c, e) and the error plotted against the productivity of the input nodes (b, d, f). The SODA model provides a good fit between the first-tier suppliers and watertight door installer when inspected visually in (e) and also by evaluating the correlation coefficient



(a) (b)

Fig. 15. Results for fitting to three node discrete event simulation using (a) SODA and (b) a linear model.

which is 0.999. However, the quality of the fit degrades when evaluating lower tier suppliers (a, c). This can be seen since plotting the predicted versus the measured outcomes do not follow the $y = y_m$ line shown in red. It is common to further investigate causes of poor fit for a model by examining residual plots, i.e., the plot of the overall error versus individual variables [59]. In Fig. 17b, d, and f, if the model fit the data perfectly, all points would fall along the horizontal red line at zero. In this case, the residual plots of each node that the target node is dependent on are considered. For example, in Fig. 16a the door node is dependent on steel and brass, so Fig. 17b shows the residual plots for brass and steel. Nonlinear trends in residual plots can be indicative of missing variables in the analysis [59]. In plots 17b and 17d, the magnitude of the error increases as the magnitude of the independent variable increases. This is called heteroscedasticity which is often an indicator that an explanatory variable is missing from the model. In this case, since the model works well for nodes that are closer to the overall target node, the missing variable is likely to

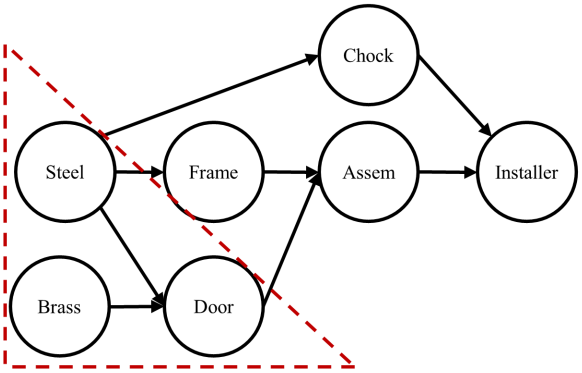
be related to the demand signal that acts as feedback and travels upstream through the supply chain. The traditional SODA methodology cannot represent this signal as it only accounts for a single direction of dependence between any two nodes. A multi-layered model in Chapters 3 and 4 will be used to capture these effects.

2.4 Summary

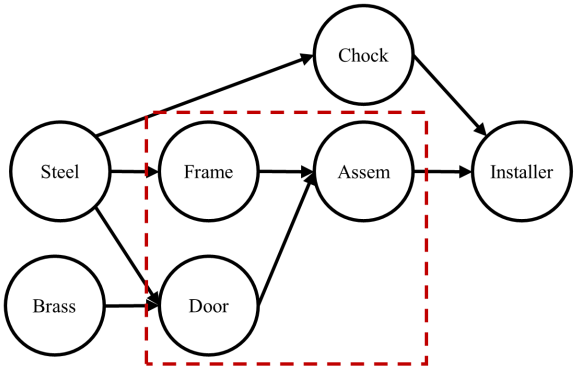
This chapter began with a summary of background and previous work in order to situate this dissertation and its contributions in the literature. Recent motivational examples were elucidated to provide impetus for further study of risk and resilience especially with respect to the causal interactions, quantification of stochastic response to disruptions, and modeling of entities that are far upstream from the target node. Then, definitions of disruption, risk, and resilience were provided followed by a discussion of previous work involving these concepts. Next, a general definition for vulnerability in complex systems was developed using definitions from various fields. A summary of work in network dependency analysis concluded the first half of the chapter.

The second half of the chapter presented an initial example along with fundamental analysis that will be used to motivate many of the decisions made in the development of the methodology in the following chapters. This analysis showed that existing methodologies for network dependency analysis, e.g., SODA, were insufficient to model supply chain systems with more than a single tier of suppliers. In order to preserve the sought-after positive attributes of SODA while combating its inability to handle cycles, effectively capture supply chain behavior at lower tiers, and propagate effects through a network over time, several modifications will be made to the SODA methodology before it is applied as part of the

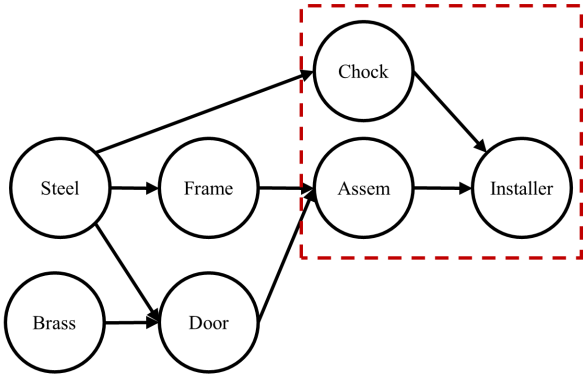
current framework as presented in Chapter 4. These modifications will include incorporating additional variables and their dependencies using multiple network layers (Fig. 18) and incorporating temporal dependence using an internal health variable to model how a node moves through states of disruption (Fig. 19). In particular, the dependence of the internal health of a node on a series of states organized into a Markov chain allows for the incorporation of dynamic risk events that span a finite period of time. Further details on the states themselves and how they interact with other components of the model will be given in the next chapter.



(a)



(b)



(c)

Fig. 16. Network diagrams showing nodes (a) door, (b) assembly, and (c) installation which were studied further by examining error plots with respect to their outputs and inputs as shown in Fig. 17.

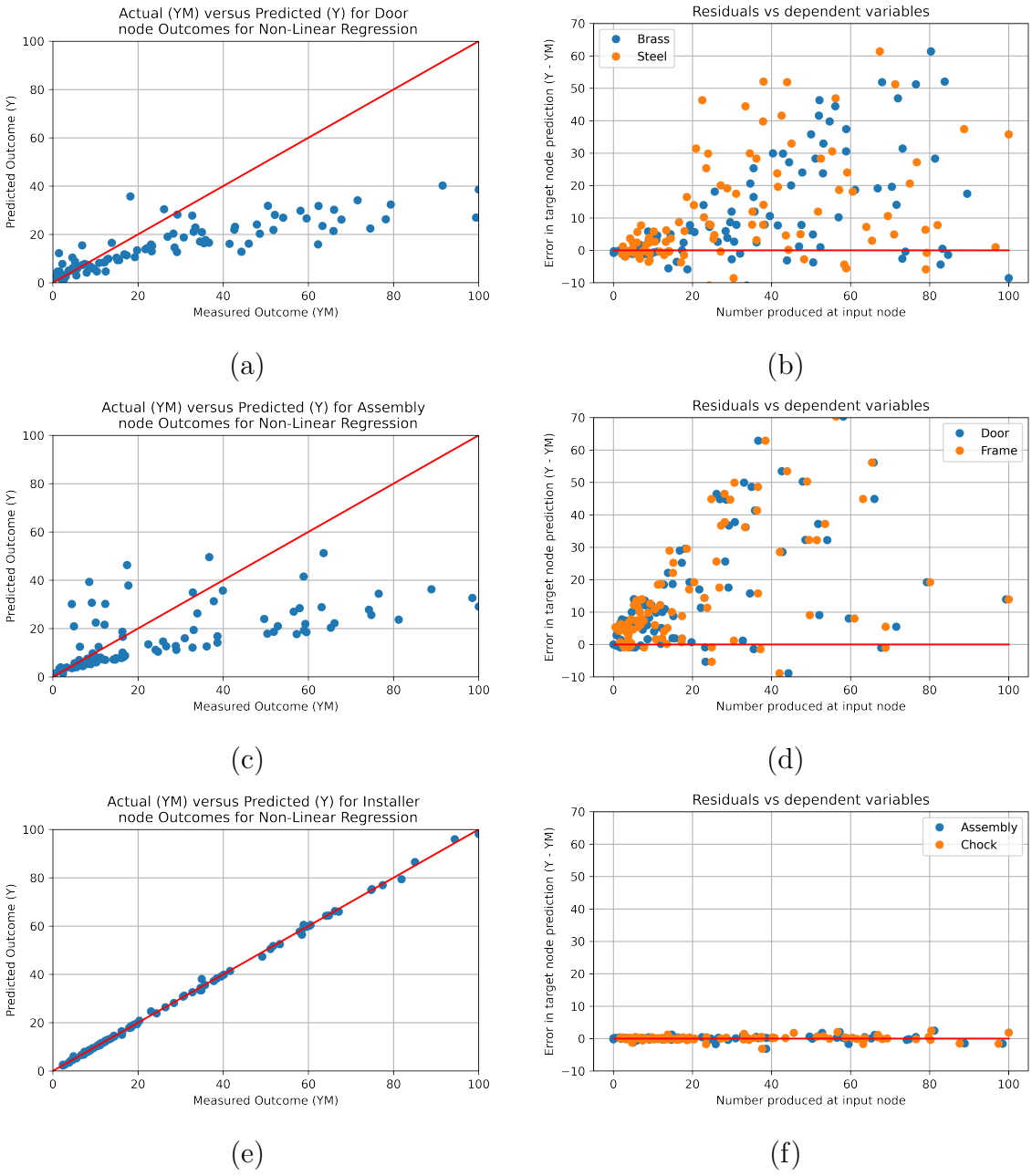


Fig. 17. Results from fitting SODA model to DES data. Left plots show estimated node output for (a) door, (c) assembly, and (e) installation versus measured values from DES. Right plots show residuals for independent variables which nodes from left are dependent on, i.e., (b) brass and steel, (d) door and frame, and (f) assembly and chock.

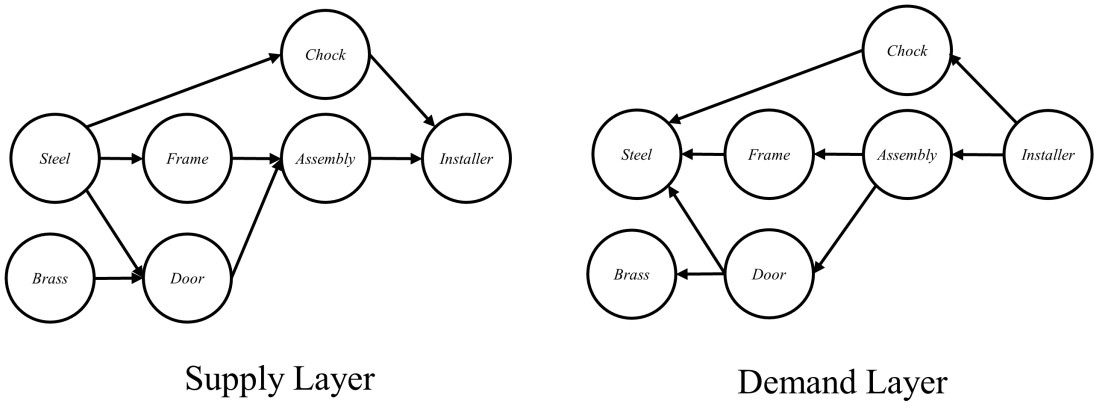


Fig. 18. Multiple network layers representing different characteristics of the watertight door supply chain including supply layer and demand layer.

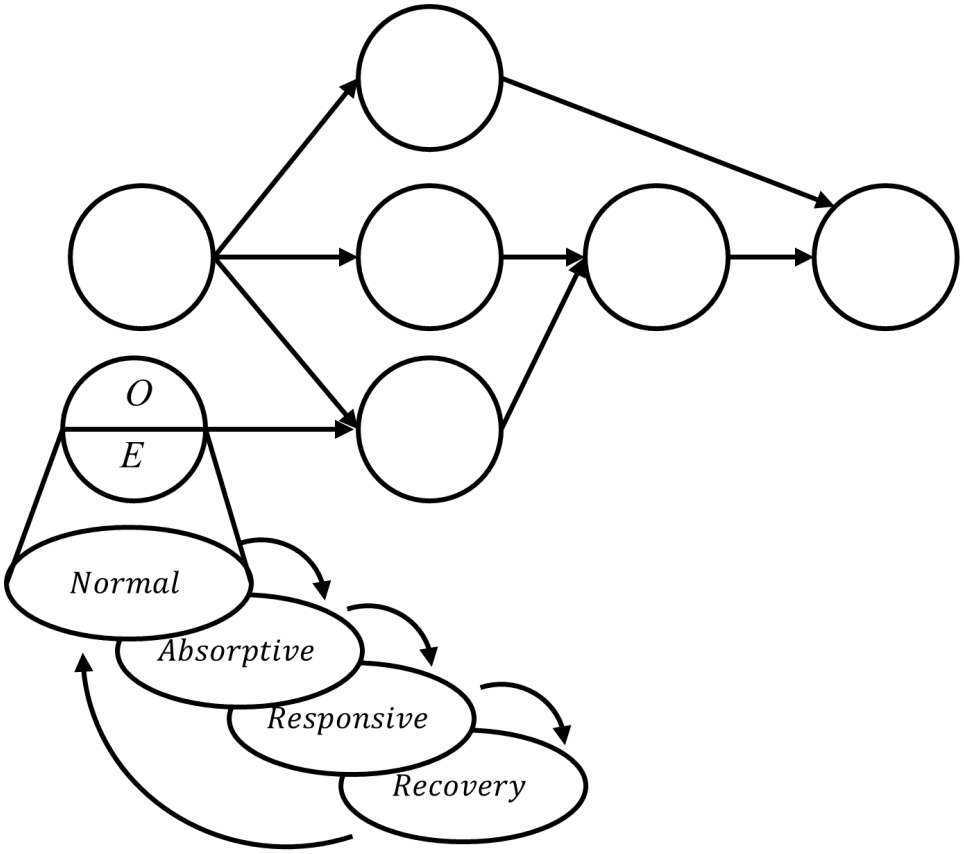


Fig. 19. Watertight door supply layer example with the dependence of the internal health, E, of one node on a Markov chain of disrupted states shown.

CHAPTER 3

RNN-ENABLED NETWORK DEPENDENCY ANALYSIS

In this chapter, details describing the development of the methodology and implementation, including training and validation, for network dependency analysis where Recurrent Neural Networks (RNNs) are used as transfer functions are presented. RNNs are a machine learning tool with many applications, including time series analysis and prediction. Machine learning is a field of study where, given data, computational algorithms learn to perform tasks for which they were not explicitly programmed. RNNs are neural network models with added memory buffers and gates designed to perform supervised learning on ordered sets of data observations by using optimization techniques to capture the underlying behavior of the system under study. First, motivated by the analysis in 2.3.1, theory for multi-layered network dependency analysis will be presented. Then, a theoretical discussion of fitting models using existing approaches which utilize non-linear, piecewise transfer functions to complex, cyclic, multi-layer networks will elucidate the necessity to replace the transfer function with a model that can be more readily fit to data. Next, the motivation for replacing the previous transfer functions with RNNs will be explained by providing a thorough background of this methodology as well as their applications in previous related work. Various architectures for RNN models will be compared to show that combining this machine learning method with network dependency analysis is superior to using deep learning on its own. Subsequently, a hyperparameter tuning study will be completed using a 2^k factorial experimental design. Finally, the chapter concludes with an experimental analysis showing that the final tuned

RNN-enable network dependency analysis model maintains at least the same accuracy as a traditional multi-output model with the added benefit of having a decomposable topology to allow the study of variations on network structure. For continuity, the same example network used in Chapter 2 will be used here and the methodology will be validated using data from a discrete event simulation model.

3.1 Multi-Layered Network Dependency Analysis

In this section the system will be modeled as a network with dependencies. By applying network dependency analysis, the operability of each node in the network will be quantified over time and allow for assessment of overall behavior as well as vulnerabilities. The topology of the network will determine how disruptions propagate through the system.

3.1.1 Multi-layered Networks

In general, the network structure for the dependency analysis model can change based on which outputs, or performance characteristics are selected for consideration. Coupled with the sparsity of the connectivity matrices representing the overall network structure, this means that implementing the process of calculating updated outputs is more natural to consider on a layered set of networks with each layer independently calculating updates to one or more output parameters whose dependencies correspond to the given network structure. Then, communication between layers can be handled as communication internal to each individual node. To simplify the explanations, it will be assumed that each output has a separate corresponding network layer.

Consider again, the supply chain example for the installation of a watertight door. It

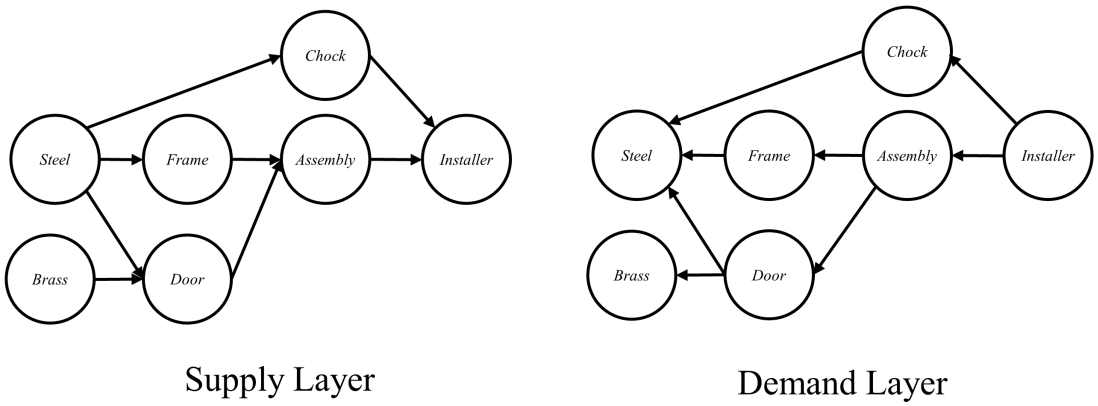


Fig. 20. Multiple network layers representing different characteristics of the watertight door supply chain.

was shown in Chapter 2 that considering the supply chain network from only one perspective, for example available supply, inhibited the ability of the network dependency analysis model to capture the behavior of the lower tier suppliers. Now, that same supply chain network will be represented by two separate layers, i.e., supply and demand, each with their own network structure (Fig. 20). From an application perspective, this improvement allows for the consideration of two completely different perspectives, in this case a capacity-based and an operational information-based perspective corresponding to supply and demand, respectively.

The final consideration is how to handle aggregation of results into a final output that can be used to calculate the behavior of resilience over time. This methodology will assume that the specific choice of this aggregation function is application dependent. For example, in a supply chain operating under a "produce-to-order" inventory control policy, the number of items produced at a given node would be, in the best case, equal to the

demand signal of the node and, in the worst case, equal to the supply capacity. In the case that there is no gap between supply and demand, the best case and the worst case would be the same since the supply and demand signals would be the equal. Specifically, if the demand signal is s_d and the supply capacity is s_c, the output, s_o, can be calculated as:

$$s_o = \begin{cases} s_d & s_c \geq s_d \\ & = min\left(s_c, s_d\right). \\ s_c & s_c < s_d \end{cases} \tag{19}$$

One practical implication of the aggregation method chosen relates to risk management in supply chain networks. In supply chain management, the order decoupling point is described as the place where a specific product meets a particular order [60]. In other words, decoupling point describes the point at which supply meets demand. Different decoupling points imply different inventory management approaches which in turn utilize different aggregation schemes for the Adaptive Risk Network Dependency Analysis (ARNDA) methodology.

Once the aggregation method is selected, a multi-layer model, like the one shown in Fig. 20, can be used to produce numerical results based on random disruptions. Traces of internal node health and output operability for each layer as well as various aggregates for both layers combined can be shown and analyzed. Later in this chapter, benefits will be shown for having layers communicating at every time step rather than aggregating layer results at the end of the analysis. Additionally, the data-driven behavior of RNN models, will allow the networks to learn the aggregation methods from data rather than having them explicitly defined.

3.1.2 Issues with State-of-the-Art Algorithms

The general formulation for forward propagation in a network dependency analysis, like those found in [41], [43], requires the nodes to have parameters for operability, internal health, and a transfer function for updating these quantities. Historically in these methodologies a vector of parameters, $\overrightarrow{\theta}_{ij}$, is defined for each edge. Additionally, a vector of output parameters, $\overrightarrow{O}_i$, and a vector of internal state parameters, $\overrightarrow{E}_i$, will be defined for each node. For a given node, n_j, with dependencies on m nodes, n_k such that $k \in \{k_1, k_2, \ldots, k_m\} = K$, the current state vector $\overrightarrow{O}_j$ is calculated as

$$\overrightarrow{O_j} = f\left(\overrightarrow{E}_j, \overrightarrow{O}_{k_1}, \ldots, \overrightarrow{O}_{k_m}; \overrightarrow{\theta}_{k_1 j}, \ldots, \overrightarrow{\theta}_{k_m j}\right), \tag{20}$$

where the function f is not necessarily differentiable but should be continuous. This relationship is also shown in Fig. 21a. In the notation above, variables that the function, f, is dependent on are defined before the semicolon and separated by commas. These variables are independent variables of the system and model usually called feature variables in the context of machine learning models and data sets. Parameters that the function, f, is dependent on are defined after the semicolon and separated by commas. These are model parameters that will be fit to the data during the optimization or training phase when the loss function is minimized. Depending on the type of model and field of application, the parameters are sometimes referred to as weights. Comparing again to Fig. 21a, the variables are independent variables of the system nodes while the parameters are weights along the edges of the graph.

In general to fit coefficients for the network, the error will need to be back-propagated through the network. To accomplish this, a methodology similar to that of applying gradient

descent in back-propagation from [61] will be used. The steps are as follows:

1. Conduct forward propagation by calculating all value for the network using the current parameters (or weights). Forward propagation starts at the root nodes and progresses to the target node.

2. Calculate the loss, $\mathcal{L}$, between the values calculated in Step 1 and the values from the data set.

3. Conduct backward propagation through the network by calculating the derivative of the loss with respect to each parameter (or weight).

4. Update each parameter (or weight) using gradient descent with a learning rate, η as shown in (21).

$$weight_i = w_i - \eta \cdot \frac{\partial \mathcal{L}}{\partial w_i}. \tag{21}$$

To perform the parameter (or weight) updates, the derivative of the loss function with respect to each parameter (or weight) must be calculated. First, define the loss function as the $\mathcal{L}^2$-norm of the difference between the calculated value and the value from the data as follows:

$$\mathcal{L}\left(O, \hat{O}\right) = \left|O - \hat{O}\right| = \sqrt{\sum \left(O - \hat{O}\right)^2}. \tag{22}$$

Finding the derivative of this loss function requires repeated use of the chain rule. Since data is available for all layers, the derivative of the loss is calculated for each node. As an example, the following will use the definition of the equations for the original System Operational Dependency Analysis (SODA) methodology shown in (15) and (16).

For root nodes:

$$O_j = E_j.$$

For all other nodes, n_j with input or feeder nodes $n_k, k \in K$,

$$W_k = \frac{1}{n-1} \sum_{i \in K, i \neq k} O_i$$

$$O_{kj}^C = \frac{100}{\gamma_{kj}} O_k + W_k^\lambda \left(100 - \beta_{kj}\right)$$

$$O_j^C = \min_{k \in K} \left(O_{kj}^C\right)$$

$$O_{kj}^S = \alpha_{kj} O_k + \left(1 - \alpha_{kj}\right) E_j$$

$$O_j^S = \frac{1}{n} \sum_{k \in K} O_{kj}^S$$

$$O_j = min\left(O_j^S, O_j^C\right).$$

There is an issue introduced by the minimum function in the original definition of SODA. Because the minimum function does not have a continuous derivative, it is not a good candidate for gradient descent. Therefore, a continuous estimation of the minimum function will be used as shown in Equation (23).

$$\min_i \left(x_i\right) \approx -\frac{1}{\rho} \log \sum_i e^{-\rho \cdot x_i} \tag{23}$$

Now, the derivative of the loss function for weight i at node j can be calculated as

$$\frac{\partial \mathcal{L}}{\partial w_i} = \frac{\partial \mathcal{L}}{\partial O_j} \cdot \frac{\partial O_j}{\partial w_i}. \tag{24}$$

While this definition does not present an issue for executing the model, i.e., forward propagation, it does present an issue for fitting the model using backward propagation.

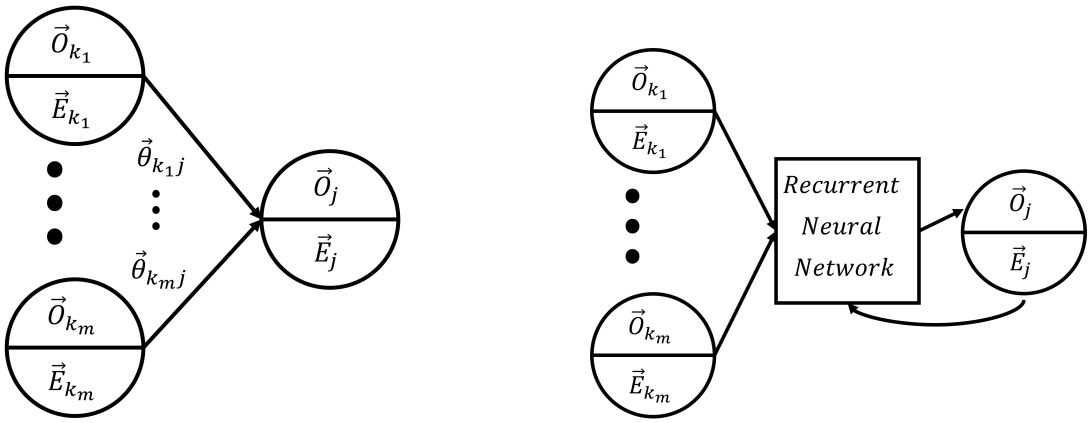


Fig. 21. Diagram to show calculation of operability for node j for (a) generalized calculation and (b) the RNN implementation.

During the application of this methodology, it quickly becomes apparent that finding an appropriate set of coefficients requires the solutions to a set of nonlinear optimization problems that are only piece-wise continuous. In addition, the number of these problems to be solved increases as the complexity of the network grows since there will be one set of coefficients to be solved for each non-root node of the network. This is troubling since one of the goals of this work is to look at increasingly complex hierarchical networks. In order to solve this issue, this work replaces the piece-wise, non-linear transfer functions with RNNs.

3.2 Exploring RNN Architectures

This section will present the background of Recurrent Neural Network (RNN) models followed by two ways in which they can be used to enable for network dependency analysis.

3.2.1 Background on Recurrent Neural Networks

Recurrent Neural Network (RNN) models have long been used for time series prediction in a variety of fields as they allow for the temporal correlation of observations [62]. The RNN unit is a variation over the time-invariant perceptron used in regular neural networks. A perceptron is a node that takes a set of inputs, combines them linearly, applies an activation function, and produces an output [63]. When perceptrons are combined in layers, the result is a neural network.

In a simple, sometimes called vanilla, RNN unit, the output of the unit depends not only on the input at the current time step but also on the unit's output from the previous time step. This can be written as

$$s_t = \phi\left(\overrightarrow{w}x_t + \overrightarrow{u}\,s_{t-1} + \overrightarrow{b}\right), \tag{25}$$

where s_t is the output of the unit at time t, x_t is the input to the unit at time t, $\overrightarrow{w}$ and $\overrightarrow{u}$ are vectors of coefficients, $\overrightarrow{b}$ is a bias vector, and ϕ is an activation function. The two main issues with the original RNN formulation are that gradients, and consequently the learning rate, may go to zero so learning stops and there is no control over the memory length [64].

The Long-Short Term Memory (LSTM) unit presents a solution to these problems by selectively storing, and forgetting, information about past states. This is done by adding three mathematical gates, i.e., input, forget, and output, controlled by weight matrices. The mathematical formulation for the input gate, forget gate, output gate, current memory

state, and current output for an individual LSTM unit are below:

$$
\begin{aligned}
i_t &= \sigma\left(W_i x_t + U_i c_{t-1} + b_i\right), \\
f_t &= \sigma\left(W_f x_t + U_f c_{t-1} + b_f\right), \\
o_t &= \sigma\left(W_o x_t + U_o c_{t-1} + b_o\right), \\
c_t &= f_t \cdot c_{t-1} + i_t \cdot tanh\left(W_c x_t + b_c\right), \\
h_t &= o_t \cdot c_t,
\end{aligned}
\tag{26}
$$

where x_t is the input at time t, all W and U are weight matrices, all b are bias vectors, and h_t is the output at time t. The activation function, shown here as σ, can be any activation function, but in this case is the sigmoid function. The dot operator implies element-wise multiplication and all other multiplication is assumed to be regular matrix multiplication. This formulation is similar to formulation that can be found in numerous references on LSTM networks including [64], [65].

Fig. 22 shows how these equations function inside the LSTM unit. The input and forget gates control the influence of the input value and previous state value on the current memory state. The output gate controls the influence of the current memory state on the output value. RNNs with LSTM and bidirectional LSTM units have been used in supply chain demand forecasting applications [66], [67]. The bidirectional implementation of an LSTM unit is very similar to that of the unidirectional implementation except that it considers both future and past time steps.

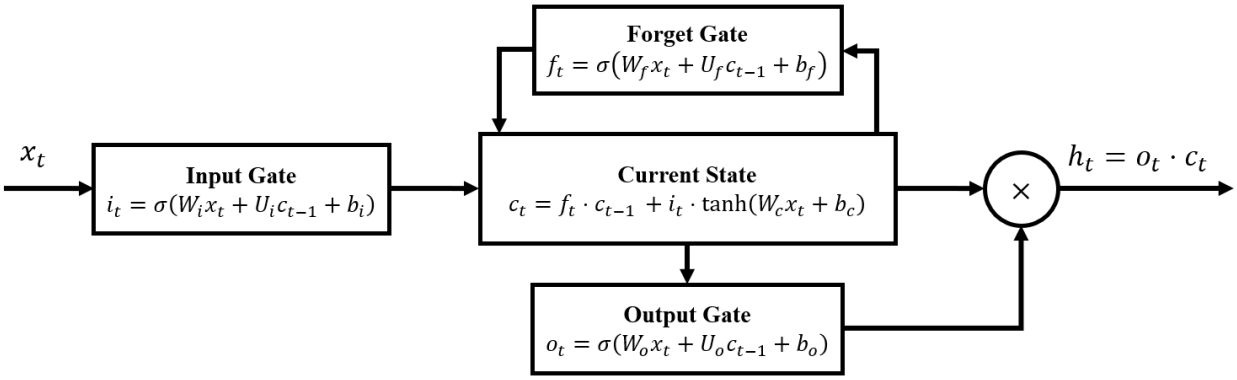


Fig. 22. Diagram showing the interaction of the functions inside a LSTM unit.

3.2.2 Single RNN for System Network

The most straightforward way to implement an RNN to model the network under study would be to use a single network with all inputs being time lagged representations of the outputs. This would allow the machine learning methodology to learn the relationships between the variables under study, limit the number of trainable parameters, and increase the overall computational efficiency by decreasing the number of networks that need training to one. However, recall that one of the overarching goals of this work is to provide a methodology that will allow practitioners and analysts to reconfigure the complex networks under study to ascertain if different network topologies lead to increases or decreases in overall system resilience. Therefore, it is better to proceed with the network dependency analysis approach, accepting the increased computational complexity, so that in future analysis individual network connections can be modified while maintaining information gained in areas of the network where the topology was not manipulated. However, it is important to ensure that the network of RNNs, with one output each, is able to provide the same accuracy as a

single RNN, with multiple outputs. Therefore, a full comparison will be done between the RNN-enabled network dependency analysis and the more traditional single RNN approach.

3.2.3 Multiple RNNs as Network Transfer Functions

Fig. 21b provides a visual representation of how the coefficients representing the edge weights from previous implementations of network dependency analysis were absorbed as parameters of the RNN while maintaining the operability of the feeder nodes and internal health of the target node as transfer function inputs.

In time series prediction, it is common to use a function of outputs at previous time steps as inputs to the transfer function. In this application, this is accounted for in the internal health parameter. Network dependency analysis implies that the output of each node is a function of its internal health. By simply assuming this function is reversible, then the internal health is also a function of the output of the target node. For the RNN model, it will be stated that the internal health signal is a function of the output of the target node at previous time steps. This function can be dependent on a variety of factors including external and internal disruptions which allows the model to be connected to the risk network in Chapter 4.

In the general case, i.e., Fig. 21(b), one RNN will be trained for each target node in each network layer in the system network under study. Inputs will be the operabilities of the feeder nodes at the current time step, $\overrightarrow{O_{k_1}}, ..., \overrightarrow{O_{k_m}}$, and internal health of the target node at the current time step, $\overrightarrow{E_j}$. The internal health of the target node at the current time step will be assumed to be a function of the operability of the target node at previous

time steps as follows:

$$\overrightarrow{E_j} = f\left(\overrightarrow{O_j}^{t-1}, ..., \overrightarrow{O_j}^{t-n}\right), \tag{27}$$

where n is the look back, number of previous time steps, the internal health of the node is dependent on. Changing the value of the look back parameter changes the behavior of the model as it affects how much historical data is used by the internal health parameter. Considerations for tuning n and other parameters are discussed in Section 3.4.

Unrolling a single layered model in time (Fig. 23), the interaction between the inputs and outputs of the RNN become clearer. In order for each RNN to calculate the operability of its target node, the RNN takes inputs which are the operabilities of the feeder nodes at the current time step and the internal health of the target node which is a function of the operability of the target node at the previous time step.

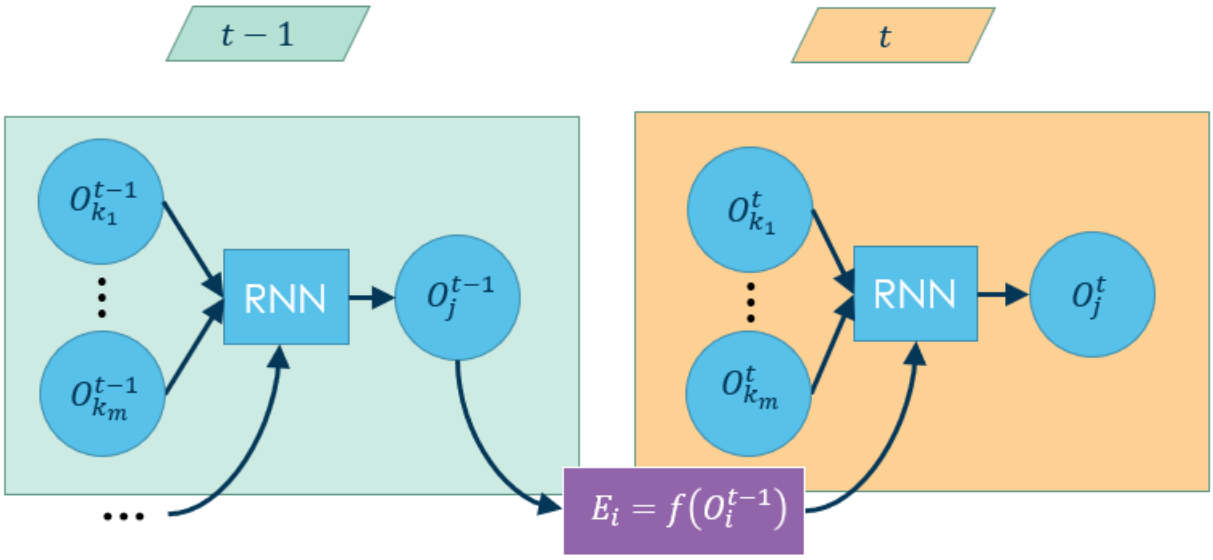


Fig. 23. Diagram showing a single layered RNN-enabled network dependency analysis model unrolled in time.

The inputs to each node will be consumed by the RNN model to compute, or predict, the operability, of the target node. Because the network itself is partitioned into separate layers for separate outputs (Fig. 20), a practical implication of the definition for internal health is that it provides a methodology for communication between layers if it is assumed that the components of the vectors are allowed to interact. For generalizability, the operability and internal health values have been presented as vectors here. If these values are scalars, this simplifies the computations, but since RNN models are capable of being trained to predict multi-objective outputs, there is no issue with having vector-valued operabilities. Additionally, reducing the function f in Eq. (27) to the identity function or a summation function allows the internal health to incorporate lagged variables which is common in time series analysis [68].

Combining all of the above will lead to the definition for f used in this work which takes the operability of each feeder node as an input, the operability of the current node in the current layer, and the operability of the current node in the other layer. A generalization of these connections unrolled in time is shown in Fig. 24. Layer A has the same topology as in Fig. 23 while the direction of the edges in Layer B is reversed, requiring additional RNNs. The only difference here is that there is an additional input to the RNN which is the internal health of the target node from the other layer(s) which allows a connection between layers. Note that in this case the internal health would be the sum of the functions of operabilities of the current node in each of the two layers. As an example, consider the door node in the supply layer in Fig. 20. The inputs to the RNN that would be used to calculate the operability of the door node in the supply layer at the current time step would

be the steel, brass, and door operabilities in the supply layer and the door operability in the demand layer all from the previous time step. That is, the operabilities of the feeder nodes in the supply layer and the operability of the door node in both layers.

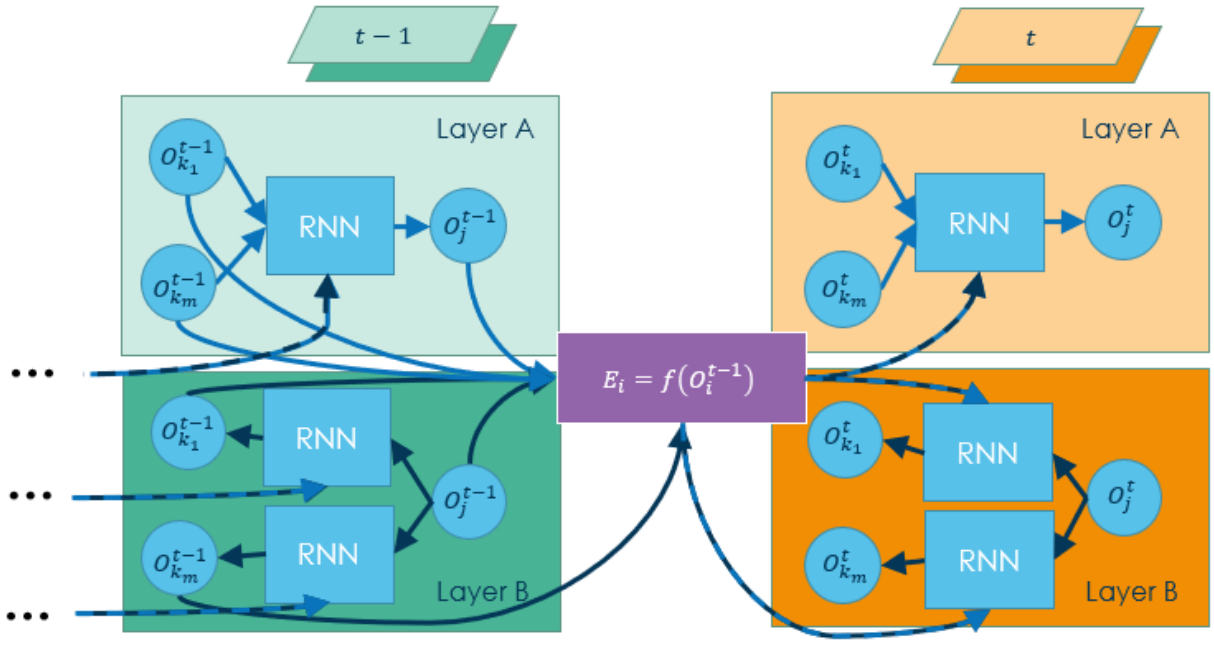


Fig. 24. Diagram showing a multi-layered RNN-enabled network dependency analysis model unrolled in time.

Finally, recall that in Chapter 2 analysis was conducted that showed single layer network dependency analysis was not able to capture feedback in networks because it was defined as an acyclic methodology. This problem has been overcome by introducing a multi-layered structure to the networks forming the basis of the network dependency analysis [8]. In the case of a supply chain, one layer may represent supply and the other demand, e.g., Fig. 20. The operability of the target node in other layers at previous time steps is used

as an additional input to the RNN model in order to connect the layers, capture this cyclic behavior, and show temporal characteristics. As with all computational methods, the value of the time step affects the behavior and convergence of the model. The effect of this, and other parameters, will be fully explored in Section 3.4.

3.3 RNN Implementation and Variations

This section provides an explanation of the supply chain example used throughout this chapter. Though the overall network is the same as the one from the previous chapter, several enhancements have been made to increase the fidelity of the model. Additionally, information regarding data preprocessing and libraries for RNN implementation is provided.

3.3.1 Example Supply Chain Network

The overall model for the network dependency layer will be a directed graphical model as was defined in Section 2.2.1. Throughout this section a theoretical supply chain for a shipboard watertight door will be presented (Fig. 25). This is the same example that was used in Section 2.3.1 for background and comparative analysis. Fig. 26 and Table 5 provide a diagram with node numbers instead of names to ease notation in equations and a lookup table linking numbers to names. The overall supply chain and assembly process for maritime vessels is very complex. This chapter will present the same simplified network that was previously used to motivate this work. However, the current model used to produce the training data used to fit the novel methodology has been implemented using the Supply Chain Operations Reference (SCOR) methodology which is an approach that would potentially be used in a real world case study [54]. The modeling approach used is

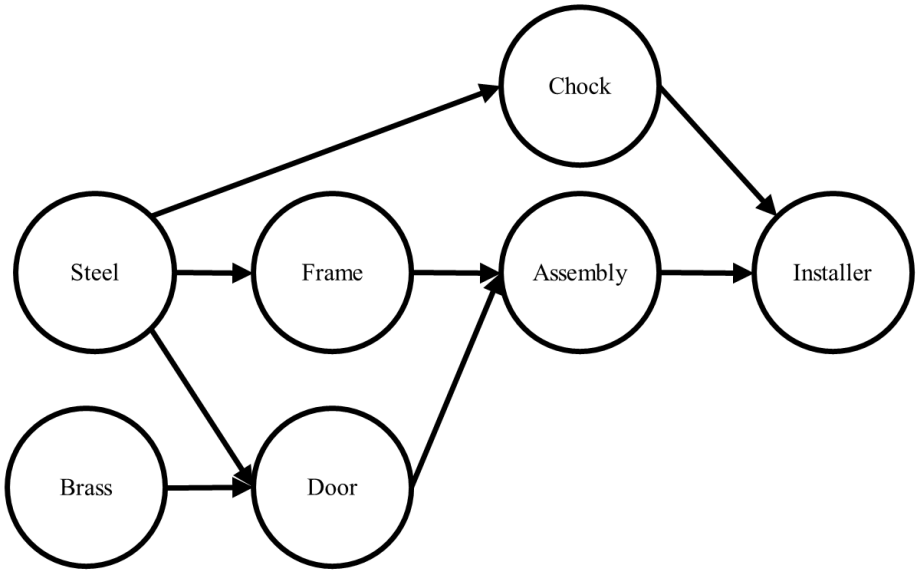


Fig. 25. Simplified network representing the supply chain for a watertight door on a ship.

like the one used by Carvalho, et al. in [55] where triangular distributions are used to model the uncertainty in planning, sourcing, and production times. Since watertight doors are often custom ordered and not delivered in quantities greater than one, the deliver stage of the SCOR methodology was absorbed into the source stage. A (s, S)-model as defined in [69] (and many others) was chosen to determine inventory reordering logic. Under this logic, each entity in the supply chain operates under the assumption that no orders are placed until inventory falls below the level defined by s. Once the inventory falls below s, orders are placed to return the inventory quantity to S. Then, the cycle continues with the system waiting for the inventory to fall below s and repeating the same behavior.

Even throughout this simplified example, assumptions will be consistent with watertight door installation on board a Naval vessel based on the procedures in the *Naval Ships'*

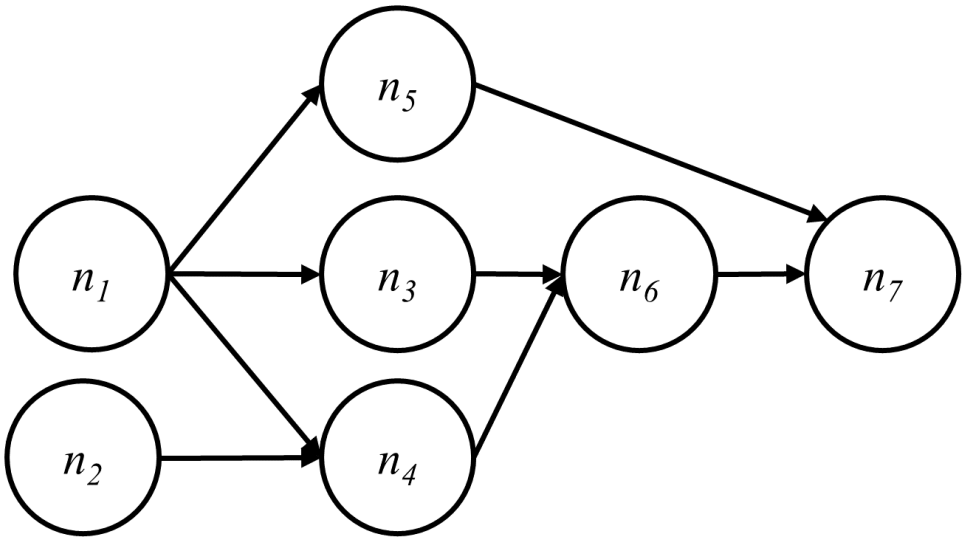


Fig. 26. Watertight door supply chain network with numeric labels.

Technical Manual on Structural Closures [51]. In order to fully understand the model, it is easiest to work backwards from the node corresponding to the installation process. Please note that many of the nodes include aggregations of various suppliers and component types, and other nodes are omitted in order to promote clarity since this network is used as a first example for the application of the Adaptive Risk Network Dependency Analysis (ARNDA) methodology. In Fig. 25, the flow of products is mapped from raw material suppliers of steel and brass to the final installation step on board a vessel.

3.3.1.1 SCOR Model Implementation

The SCOR methodology assumes that the supply chain operates in four phases, i.e., Plan, Source, Make, and Deliver [54]. As previously mentioned, since each unit was assumed to be shipped individually, the implementation for this work combined the Make and Deliver

TABLE 5. Node names for Fig. 26.

Node Number	Node Name
1	Steel raw material suppliers
2	Brass raw material suppliers
3	Frame manufacturers
4	Door assembly manufacturers
5	Steel chock manufacturers
6	Watertight door assembly supplier
7	Watertight door on board installation

and the overall combined length of both phases was estimated using a single probability distribution. Table 6 shows an example of one set of SCOR parameters that was used. All other parameter sets were similar in that they also used triangular distributions to model uncertainty in process times and (s, S) inventory rules. For example, the installed door work planning time is represented by a triangular distribution with parameters $(0.5, 1, 4$, i.e., the minimum is 0.5 hours per unit, the mode is 1 hour per unit, and the maximum is 4 hours per unit.

3.3.2 Data Preprocessing

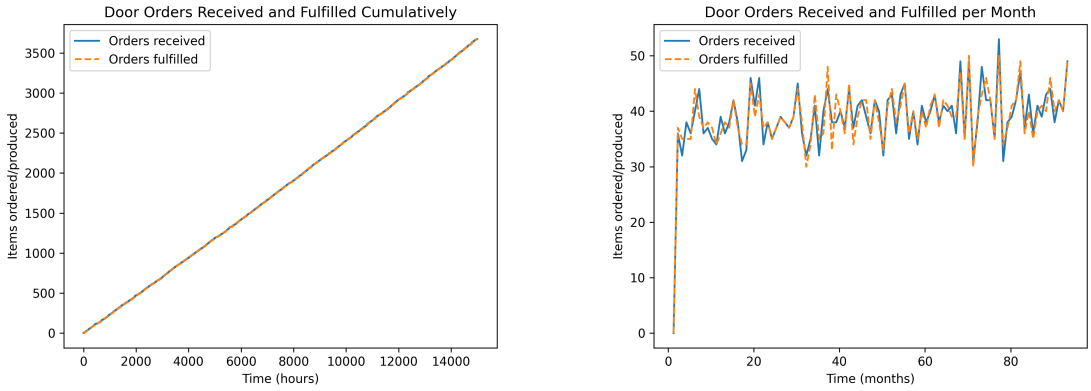
In this section, the data collected from the discrete event simulation will be described including any pre-processing steps that were taken. Data from the discrete event simulation was recorded at one hour intervals. It is known that this is likely a higher sampling rate than would be available for real world data sets. However, collecting hourly data allowed for the use of various accumulation windows during pre-processing between the Discrete Event

Simulation (DES) and fitting the RNN models to experiment with different data collection frequencies. Therefore, data was processed using 20, 40, 80, and 160 hour (0.5 week, 1 week, 2 week, and 1 month) windows. An approach which considers data accumulated over a window represented by a fixed time period is utilized, similar to the work of Carvalho, et al. in [55] and Park, et al. in [70].

TABLE 6. SCOR parameters for supply chain model example.

Process	Plan	Source		Make
Component *Sub-components*	Work planning Tri. dist. (h/unit)	Inventory (max, min)	Inspect/Move Tri. dist. (h/unit)	Production Tri. dist. (h/unit)
Installed door	(0.5, 1, 4)			(4, 6, 10)
Chock		(15, 10)	(1.9, 2, 2.2)	
Assembly		(5, 2)	(3, 4, 6)	
Chock	(0.1, 0.2, 0.4)			(0.5, 1, 1.25)
Steel		(20, 10)	(0.3, 0.5, 0.7)	
Assembly	(0.5, 1, 2)			(2, 5, 7)
Door		(3, 1)	(3, 4, 6)	
Frame		(3, 1)	(3, 4, 6)	
Door	(0.25, 0.75, 1.25)			(3, 6, 12)
Steel		(20, 10)	(0.3, 0.5, 0.7)	
Brass		(20, 10)	(0.3, 0.5, 0.7)	
Frame	(0.15, 0.5, 0.8)			(4, 7, 11)
Steel		(20, 10)	(0.3, 0.5, 0.7)	
Steel	(0.1, 0.15, 0.2)			(0.5, 1, 2)
Brass	(0.1, 0.15, 0.2)			(0.25, 0.5, 1)

The cumulative variables for units ordered and produced at each node were subtracted at set intervals to calculate the units ordered and produced during each of those



(a) (b)

Fig. 27. Diagram to show data processing calculation from (a) cumulative orders recorded hourly to (b) monthly orders in terms of order received and orders fulfilled.

intervals, respectively. Given that there are 8 hours per work day and 5 work days per week, a data series representing items produced monthly can be produced by selecting every 160th data element and then subtracting adjacent elements. An example of the original data compared with the transformed data is shown in Fig. 27. When compared with subfigure (a), (b) shows that even though the door node is able to accommodate demand overall, there are several periods with shortfalls. These shortfalls could lead to a disruption causing a ripple effect through the supply chain under study and as such are an important dynamic to study.

3.3.3 Code Implementation

Keras [71] and Tensorflow [72] were used for the implementation of RNNs. Each RNN was implemented as a sequential model with one or more hidden layers each having equal numbers of LSTM units, e.g., Fig. 28. Both a single RNN with multiple outputs and a network of RNNs as discussed in 3.2 were implemented for comparison, and results will be shown in section 3.5. For the results presented in this work, the single LSTM and RNN-enabled network depencdency analysis LSTM were implemented with n hidden layers with m units each. Specific values for n and m will be experimented with and recommendations for selection will be provided in Section 3.4. The code snippet below shows an example implementation with two hidden layers and "numUnits" units in each of the three layers.

Listing 3.1. LSTM example

```
from keras.models import Sequential
from keras.layers import Dense, LSTM


model = Sequential()
model.add(LSTM(numUnits,
            dropout=drop,
            return_sequences=True,
            input_shape=(look_back, total_variables)))
model.add(LSTM(numUnits, dropout=drop))
model.add(Dense(len(ind_out)))
model.compile(optimizer='adam', loss='mse')
```

For networks with two hidden layers and 80 units in each hidden layers, the number of parameters in the single RNN with multiple outputs is shown in Table 7. For the output

shape, the first parameter is the batch size and is listed as None as this can be variable depending on how the model is trained. The LSTM nodes can generate outputs for multiple previous time steps, i.e. a look back value greater than 1. Recall how the look back parameter was related to the node's internal health in Section 3.2.3. When there are three shape parameters, the second parameter of the output shape is the look back parameter. The remaining parameters are the shape of each individual output observation passed to the next layer. Since each of the LSTM layers has 80 nodes, the output shape of the previous layer should be 80. For the single RNN model, the dense layer outputs all 11 variables while for the multiple RNN model each dense layer only outputs a single variable as there are 11 total models. For networks with multiple RNNs, the number of parameters for one of the eleven networked RNNs is shown in Table 8. Each of the eleven individual RNNs has between 78,481 and 79,121 parameters.

TABLE 7. RNN parameters for single multi-output model.

Layer type	Output shape	Number of Parameters
LSTM	(None, 1, 80)	30,400
LSTM	(None, 80)	51,520
Dense	(None, 11)	891
Total parameters :	82,811	

3.4 RNN Hyperparameter Tuning

There are a variety of parameters that, when adjusted, affect the RNN model's ability to learn the data provided to it. This section will investigate how sensitive RNN

TABLE 8. RNN parameters for one of, i.e., Assembly Supply, the networked single-output models.

Layer type	Output shape	Number of Parameters
LSTM	(None, 1, 80)	27,200
LSTM	(None, 80)	51,520
Dense	(None, 1)	81
Total parameters:	78,801	

models are to several types of parameters including those that define the structure of the network, characteristics of the input data, and behavior of the model. While there are many established methodologies that can be used for tuning hyperparameters for individual neural networks, this methodology is using a combination of interdependent RNNs. Therefore, a 2^k experimental design approach similar to [73] was used to determine which hyperparameters had statistically significant effects and interactions on the model.

3.4.1 Factors

Each of the following factors was varied at two different levels (shown as + and −, corresponding to +1 and −1 in Table 9) in order to produce a set of responses in using the 2^k design.

TABLE 9. Factor coding chart for 2^8 factorial design for model hyperparameters.

Factor	−	+
Training Ratio	70%	90%
Data accumulation window	20 hours	80 hours
Number of layers	2 layers	4 layers
Unit dropout rate	0%	20%
Units per layer	20 units	80 units
Loss metric	Mean Absolute Error (MAE)	Mean Squared Error (MSE)
Look back steps	1 step	4 steps
Model type	Simple RNN	LSTM

3.4.1.1 Training Ratio

When training machine learning models, the data under study must be split into two sets, namely train and test [61]. Recent articles on LSTM used train-to-test ratios of 50-50, 60-40, 70-30, and 80-20 [66], [74], [75]. The two levels used as shown in Table 9 are 70% training, 30% testing and 90% training, 10% testing.

3.4.1.2 Data Accumulation Window

The data accumulation window was varied as discussed in section 3.3.2. Time windows were selected to be $\frac{1}{2}$-week, 1-week, 2-weeks, and 4-weeks (approximately one month). This was reasonably consistent with the time scale used by Carbonneau, Laframboise, and Vahidov in [66].

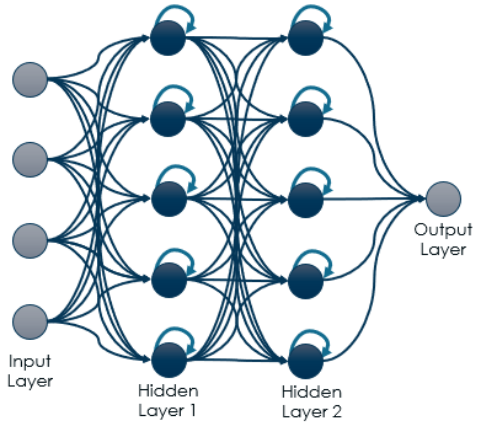


Fig. 28. RNN with four inputs, two hidden layers with five units each, and one output.

3.4.1.3 Number of RNN Layers

Each RNN used in this work will have an input layer, n hidden layers (all of which will be LSTM layers), and an output layer. For the purposes of model tuning, the number of hidden layers will be under consideration. Therefore, the model architecture will be referred to by the number of hidden layers. For example, the model shown in Fig. 28 has two hidden layers, so this would be referred to as a two layer model.

3.4.1.4 Unit Dropout Ratio

In order to avoid overfitting, many machine learning techniques have employed dropout. Dropout has been successfully used to regularize RNNs as well [76]. When applying dropout, a certain percentage of the LSTM units in the hidden layers are randomly masked, or turned off, during different training steps.

3.4.1.5 Units per Layer

The number of units used per hidden layer varies greatly across different applications. For example, Duan, Lv, and Wang [77] successfully used between 1 and 5 to predict travel time while Pacella and Papadia [67] used 200 for time series forecasting in supply chain management. In this study, a variety of values were selected for experimentation, but in keeping with the results from Pacella and Papadia greater numbers of hidden units tended to work better and were therefore selected for the factor study.

3.4.1.6 Loss Metric

Consistent with recent literature, loss metrics for this work will be presented as Mean Absolute Error (MAE) and Mean Squared Error (MSE). The MAE and MSE are defined in [78] as

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_t - \hat{y}_t\right|, \tag{28}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_t - \hat{y}_t\right)^2, \tag{29}$$

where y_t and $\hat{y}_t$ are the observed and predicted variables, respectively, and n is the number of observations.

3.4.1.7 Look Back Steps

As with any time series problem, the model can be sensitive to the look back, or time delay, from which the inputs are removed from the output [79]. In Section 3.2.3, the relationship between the look back and the internal health of the node was discussed. In

RNNs, the units in the hidden layer have built-in memory buffers that allow memory to be stored in addition to the explicit incorporation of previous time steps. In this study, look back values between 1 and 4 time steps were used which was consistent with work by Duan, Yisheng, and Wang in [77].

3.4.1.8 RNN Model Type

Even though it was noted in section 3.2.1 that there are advantages of using LSTM networks over simple, or vanilla, RNNs, the two models were included in the hyperparameter study for completeness.

3.4.2 2k Factorial Analysis and Results

For this analysis, the effect of the factors on the mean squared error (MSE) will be considered since this is one measure of how well the model fits the data. Recall that the 8 factors are shown in Table 9. In addition to these 8 factors, it is important to consider interactions for factors that may be dependent on each other. For example, the training ratio may not appear to affect the response variable, in this case MSE, when it is manipulated on its own, but when the training ratio and nodes per layer are changed at the same time, there could be a significant effect on the response variable. When there are 8 factors being considered, two-, three-, four-, five-, six-, seven-, and eight- factor interactions can be calculated to determine if they are significant. This equates to taking combinations of n, here 8, objects r at a time.

Therefore, there are 8 one-factor effects, 28 two-factor interactions, 56 three-factor interactions, 70 four-factor interactions, 56 five-factor interactions, 28 six-factor interactions,

8 seven-factor interactions, and 1 eight-factor interaction. To consider all factor effects and multi-factor interactions, 255 responses would have to be calculated and assessed for statistical significance (256 minus the 1 corresponding to the empty set). In order to keep the complexity of the analysis reasonable, this study will examine the single factor effects and two-factor interactions for a total of 36 responses.

TABLE 10. Partial design matrix and single run simulation results for 2^8 factorial design for model hyperparameters.

Design Point	1	2	3	4	5	...	254	255	256
Training ratio	−	−	−	−	−	...	+	+	+
Data accumulation window	−	−	−	−	−	...	+	+	+
Number of layers	−	−	−	−	−	...	+	+	+
Unit dropout rate	−	−	−	−	−	...	+	+	+
Units per layer	−	−	−	−	−	...	+	+	+
Loss metric	−	−	−	−	+	...	+	+	+
Lag steps	−	−	+	+	−	...	−	+	+
Model type	−	+	−	+	−	...	+	−	+
Response	1.459	1.308	1.859	1.337	1.650	...	0.052	0.066	0.051

Now that the factors and interactions have been identified, factor levels can be coded to define design points. The coding for the factors is shown at the top of Table 9 as − and + values for each factor. Given that there were 8 factors selected for model tuning and hyperparameter experimentation, there were 2^8 or 256 design points. Notice that this is the same as the number of effects and interactions since it is all possible combinations of the two factor levels. For each design point, a set of 11 RNN-models was trained and

tested to get a response. A sample of the design matrix for these 256 points is shown in Table 10. Once the response was calculated as the MSE, the effect for each factor and interactions between factors were calculated as well. In order to calculate the factor effects and 2-factor interactions, a technique leveraging the design matrix was used (see [73] for a complete explanation). The technique involves finding the change in response when a particular factor has a value of -1 compared to when that factor has a value of $+1$. To find this effect for any factor, take the signs in the row corresponding to that factor in the design matrix and multiply by the response row. Then, sum and divide by 2^k. For example, using Table 10 for the factor model type,

$$\begin{aligned} \text{effect}_{model} &= \frac{-1.459 + 1.308 - 1.859 + 1.337 - 1.650 + ... + 0.052 - 0.066 + 0.051}{2^8} \\ \text{effect}_{model} &= -0.078. \end{aligned} \tag{30}$$

Since the signs are different for each factor row, the effect will be different for each calculation. Conveniently, interactions can be calculated similarly by multiplying the rows corresponding to the interacting factors component-wise and then multiplying the result by the response row. The end of the procedure is the same as for the effect calculation where the products are summed and then the results are divided by 2^k. For example, the training

ratio and model interaction would be:

$$\begin{aligned}
\text{interaction}_{ttratio\times model} &= \frac{N}{2^8} \\
\text{where } N =&(-1)(-1)\cdot 1.459 + (-1)(+1)\cdot 1.308+ \\
&(-1)(-1)\cdot 1.859 + (-1)(+1)\cdot 1.337+ \\
&(-1)(-1)\cdot 1.650 + ...+ \\
&(+1)(+1)0.052 + (+1)(-1)\cdot 0.066+ \\
&(+1)(+1)0.051 \\
\text{interaction}_{ttratio\times model} &= -0.003.
\end{aligned} \tag{31}$$

So far, the calculations have only been done for a single run of the model. However, the response of the model is itself a random variable that needs to be estimated using a confidence interval rather than a point estimate. Therefore, the 11 RNN-models were tested on 25 different runs of 3000 hours (or about 1.5 years) of data each. For each design point and replication, the response was calculated. Then, the mean, standard deviation, and a 95% confidence interval for each factor effect and interactions between factors was calculated to see which tuning and hyperparameters had the greatest effect on the model. An effect or interaction is deemed to be statistically significant if the confidence interval does not include zero. These results for the effect on the mean squared error are given in Table 11 and shown graphically in Fig. 29 and Fig. 30. The mean value is an estimate of the overall magnitude of the effect resulting from changing the value of the factor from the -1 to $+1$ level. The standard deviation and confidence interval provide information on how much confidence can be placed in the accuracy of these results based on the variation in the 25 individual experimental runs that were performed for each design point.

TABLE 11. Single factor effect results for mean squared error.

Factor	sample mean	sample standard deviation	95% confidence interval
Training ratio	0.004829	0.000253	(0.004309, 0.005349)
Data accumulation window	-0.802244	0.038603	(-0.881749, -0.72274)
Number of layers	0.090199	0.003277	(0.08345, 0.096948)
Unit dropout rate	0.039300	0.001498	(0.036215, 0.042385)
Units per layer	-0.031324	0.001765	(-0.03496, -0.027688)
Loss metric	-0.021632	0.000532	(-0.022727, -0.020537)
Look back steps	-0.001749	0.011504	(-0.025441, 0.021944)
Model type	-0.07843	0.004226	(-0.087134, -0.069726)

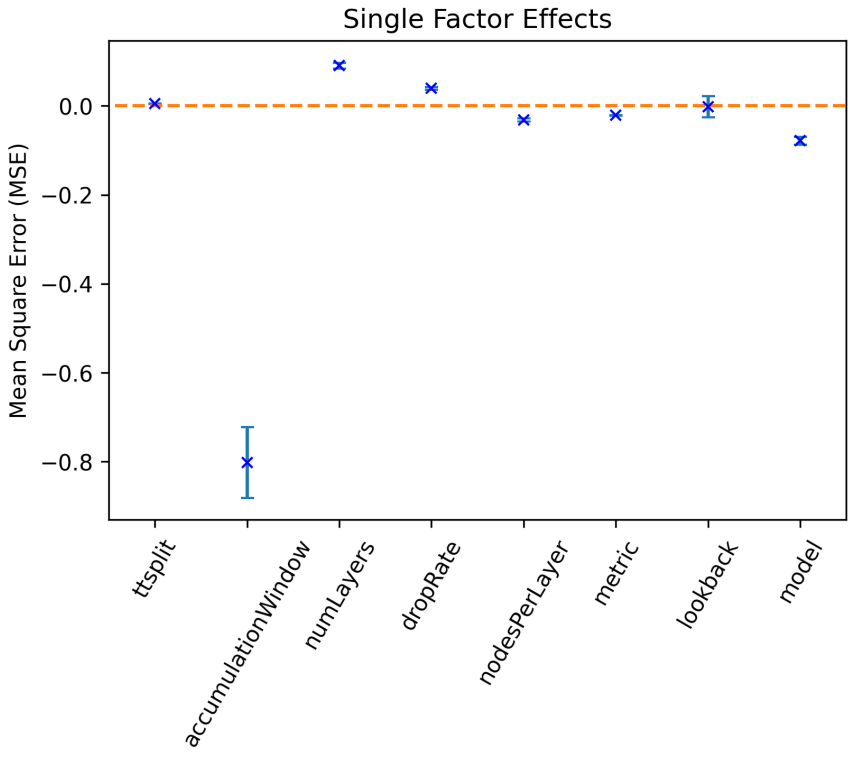


Fig. 29. Results showing 95% confidence intervals for all 8 factors evaluated for RNN hyperparameter tuning.

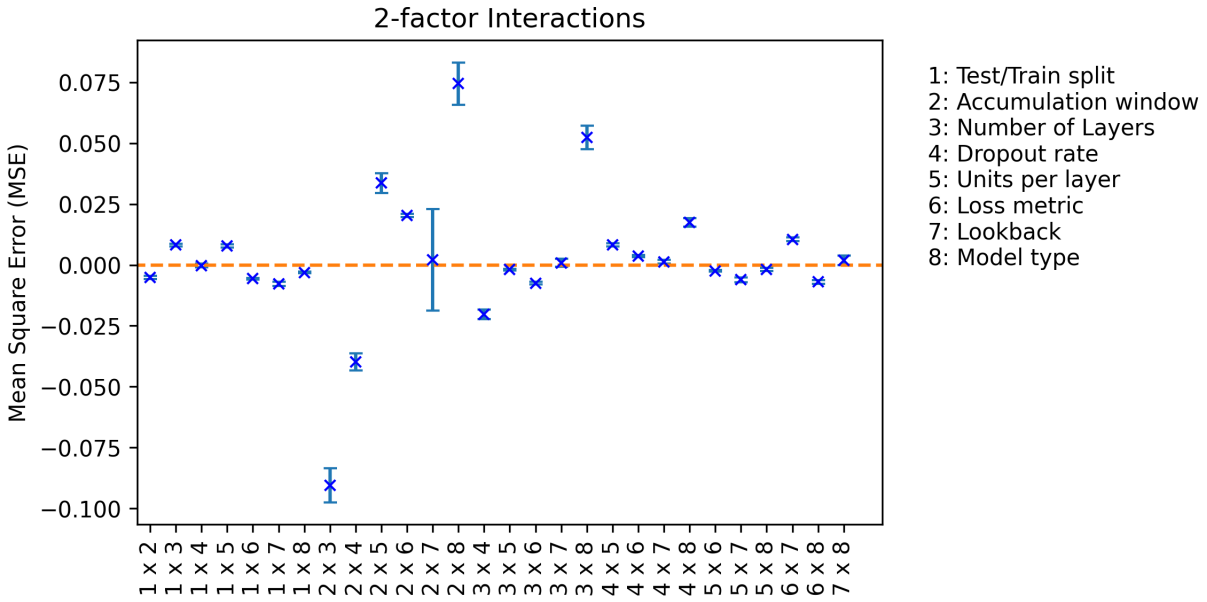


Fig. 30. Plots showing results for factor interactions.

The goal is to determine which factors and interactions between factors have the largest effect on the error of the model. Because the mean value for the data accumulation window has the largest magnitude, it has the most significant effect on the mean squared error. One of the effects of enlarging the data accumulation window is to decrease the amount of detail or granularity that exists in the data which has an effect similar to smoothing. It stands to reason that by increasing the window indefinitely, virtually all of the dynamic fluctuations could be lost from the data and the behavior for the desired model to capture would all but disappear. In addition, the frequency with which the data is captured is likely controlled by individuals working in the industry under study rather than the modeling and simulation engineers and data scientists applying this methodology. Therefore, accumulation window is an important parameter but not necessarily one that can be tuned.

The only factor with an effect that is not significant, i.e. the confidence interval of

the effect contains zero, is the look back. This can be seen in Fig. 29. Even though the training ratio effect is significant, it is very small and does not appear in any of the higher level interactions that will be considered so that will be neglected as well. Table 12 shows the top 10 factors and interactions that will be investigated further. The table identifies whether each result is an effect or an interaction, the contributing factor(s), and the sample mean in decreasing order of sample mean magnitude. The data accumulation window and any interactions it was involved in have been removed for clarity.

TABLE 12. Table showing top single factor effects and two factor interactions for further analysis.

Type	Factor(s)	sample mean
Effect	Number of layers	0.090199
Effect	Model type	-0.078430
Interaction	Number of layers x Model type	0.052344
Effect	Unit dropout rate	0.039300
Effect	Units per layer	-0.031324
Effect	Loss metric	-0.021632
Interaction	Number of layers x Unit dropout rate	-0.020209
Interaction	Unit dropout rate x Model type	0.017397
Interaction	Loss metric x Look back steps	0.01053
Interaction	Unit dropout rate x Units per layer	0.008267

Considering the next effect, it is interesting to note that the analysis is showing that as the number of hidden layers in the model is increasing, the error is also increasing when usually increasing the depth of a machine learning model decreases the error. This can be explained by a common finding with RNNs which is that due to their recurrent structure,

they are already "deep" and adding additional layers does not tend to decrease the error of the model [80], [81].

Recall that model type was included for completeness as there is a large body of evidence in the published literature that shows that LSTM RNN models are superior to simple RNN models. As such, it was expected that there would be a significant decrease in the MSE induced by switching the model from a simple RNN to an LSTM which is reflected in the results. The other effects indicate that a larger number of hidden units will lead to a better fit as will using the mean squared error as opposed to the mean absolute error.

Looking further down Table 12, it becomes interesting to investigate the interactions. If it were possible to estimate the degree of interaction between the hyperparameters as linear or quadratic, it might make sense to employ response surface methodology here which has recently been used for regular artificial neural networks with success by Bozkurt and Buruk in [82]. While this is an interesting direction for future work, it will not be pursued here as the nature of the interaction may not be linear, many of the variables are discrete rather than continuous, and the magnitude of the interactions is relatively small. For the remaining parameters in Table 12 that do have significant interaction, it is not too computationally intensive to do a full study varying both parameters simultaneously. For example, Fig. 31 shows the results of this process for number of layers and the dropout rate. It can be seen that the combination of these two parameters that minimizes the mean square error is a single hidden layer and no dropout.

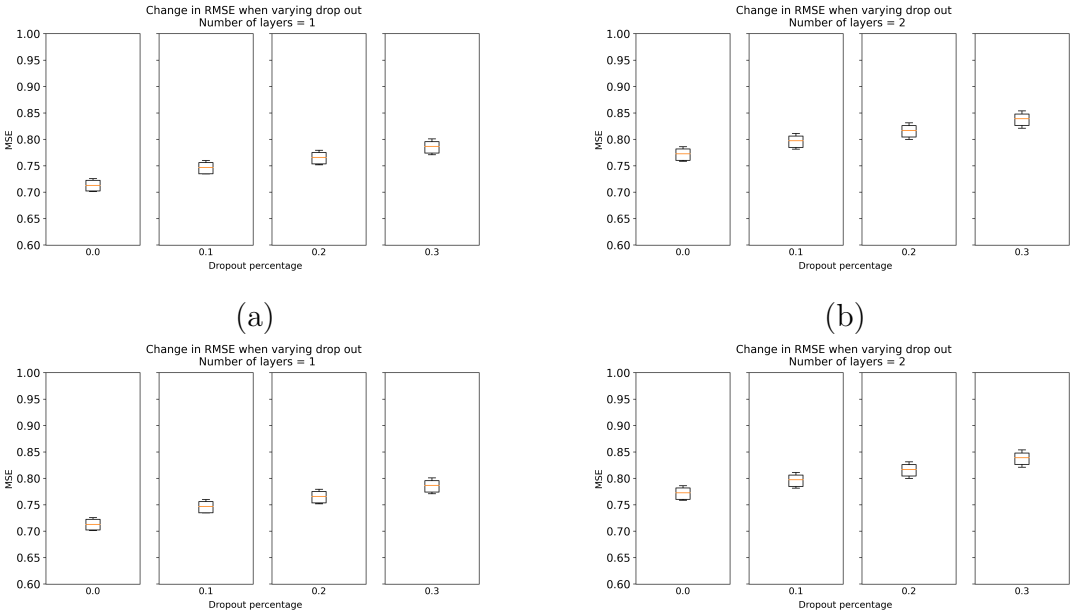


Fig. 31. Results showing interaction for drop out for (a) 1, (b) 2, (c) 3, and (d) 4 hidden layers.

3.5 RNN Enabled Cyclic Network Dependency Analysis

In order to validate the results, three methods were used. The first is the novel methodology, RNN-enabled network dependency analysis (rNDA). The second was a traditional multi-output RNN (Single RNN). The final method, for completion, was a naive method for time series models where the current output value is assumed to be equal to the output at the previous time step (labelled as naive on all figures). Each of the RNN based methods used mean squared error as a loss metric and will be used throughout the results in order to compare the methods. Based on the results from Section 3.4, the single RNN and rNDA networks had a single hidden layer with 80 LSTM units. No drop out was implemented, and a 90% training testing split was used.

The networks were trained and validated on 15,000 hours of data averaged over 40 hour (1 week) periods. They were then tested on 25 independent samples of 3,000 hours each. Table 13 shows the average of the mean squared error results for all three models for each of the eleven output nodes. In addition, the average MSE across all 11 output nodes is provided as well as a percent difference comparing the single LSTM and naive models to the rNDA model. Recall that in the supply chain network, raw material supply, i.e., steel and brass supply, and target node demand, i.e., watertight door demand, are overall inputs to the network. In practical applications (and in the discrete event simulation), these are represented by appropriate statistical distributions. In Table 13 the minimum mean square error for each output has been bolded. Comparing the results for rNDA and the single RNN shows that they are very close, but the rNDA method does outperform the single RNN method in all but one case. A visualization of the distributions of the mean square error is provided in Fig. 32 showing each distribution as a box plot. Again, it is clear that both rNDA and single RNN outperform the naive method and have similar performance to one another.

Since the results for the rNDA and single RNN methods were consistently close for all outputs, a second analysis methodology was applied to ascertain if they were indeed the same. In this case, the 95% confidence intervals were computed for the set of mean squared errors for each output (Fig. 33). These confidence intervals are significantly overlapped and, in fact, nearly identical for all nodes. Therefore, it is reasonable to conclude that the methods achieve at least similar, if not better, results with respect to error.

TABLE 13. Mean squared error results for 25 test runs across three models for comparison.

Node	Model Side	rNDA	Single LSTM	Naive
Assembly	Supply	**0.015939**	0.016025	0.031452
Chock	Supply	**0.020100**	0.022003	0.044749
Door	Supply	**0.008536**	0.009412	0.021430
Frame	Supply	**0.015092**	0.017408	0.041121
WTDoor	Supply	**0.016152**	0.018672	0.033434
Assembly	Demand	**0.010464**	0.011223	0.021361
Brass	Demand	0.0467477	**0.044216**	0.123428
Chock	Demand	**0.026805**	0.027397	0.067470
Door	Demand	**0.008936**	0.010018	0.022815
Frame	Demand	**0.012497**	0.013841	0.034894
Steel	Demand	**0.022217**	0.023443	0.045131
Average		0.018499	0.019424	0.04429
Percent Difference			26.73%	139.42%

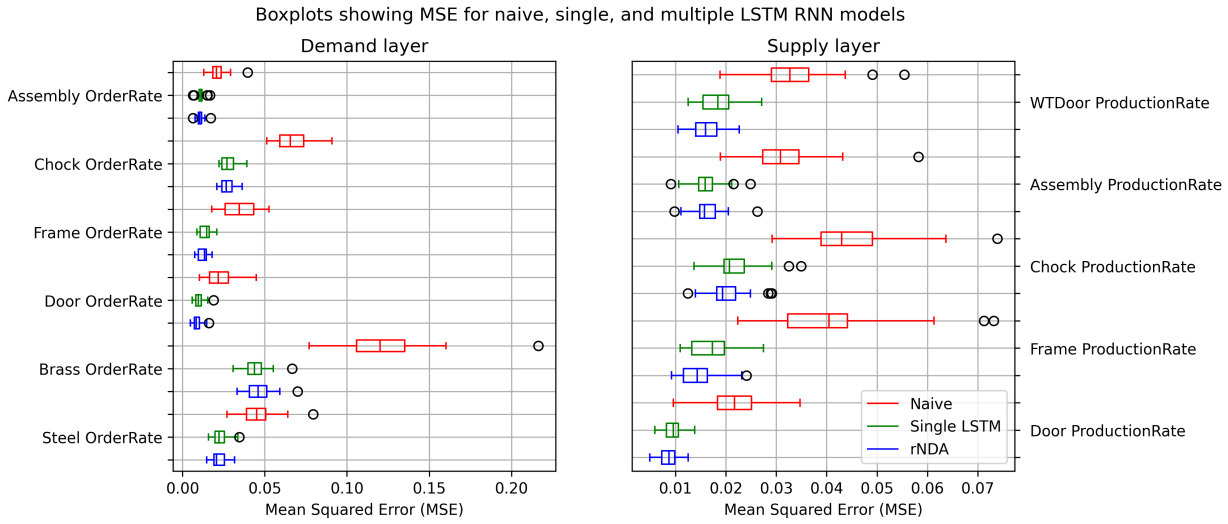


Fig. 32. Boxplots comparing mean squared error for rNDA, single LSTM and naive models.

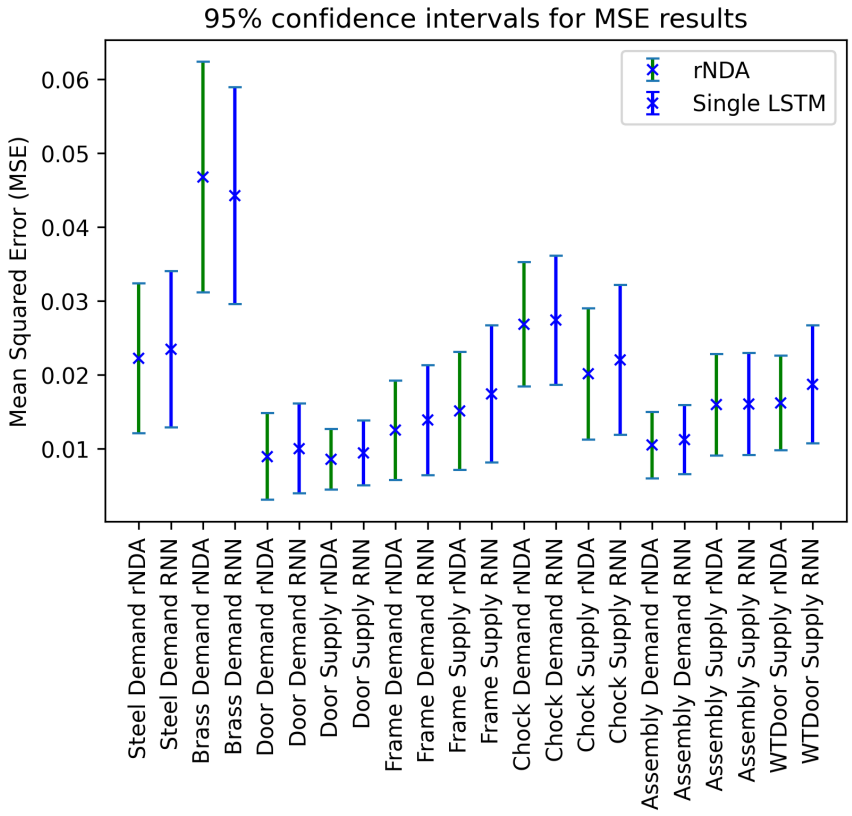


Fig. 33. 95% confidence intervals for mean squared error for rNDA and single LSTM models.

3.6 Summary

This chapter provided a detailed description of the development of the methodology and implementation, including training and validation, for network dependency analysis where RNNs are substituted for the traditional nonlinear piecewise transfer functions. Grounded in the literature and motivated by the analysis in 2.3.1, the overall theory for the methodology was presented with supporting equations and implementation details. Model fitting and hyperparameter tuning details were elucidated as the overall methodology was applied to a supply chain example. Finally, the methodology was shown to have better or equivalent accuracy to a traditional multi-output RNN but with the added flexibility of having a modifiable architecture to enable studies of complex networks where the topology itself is a variable. In the next chapter, this methodology will be employed to model the system network for Adaptive Risk Network Dependency Analysis (ARNDA) and a Bayesian risk network will be used to apply disruptions to study complex network resilience under disruptions.

CHAPTER 4

ADAPTIVE RISK NETWORK DEPENDENCY ANALYSIS

The overall process for performing Adaptive Risk Network Dependency Analysis (ARNDA) involves mapping a system to a hierarchical network divided into layers, defining a risk space with probabilistic constructs to allow for simulation of risk space behavior, and finally situating the hierarchical network model in the risk space by defining dependencies between the network nodes and risks. In this chapter, the methodology for implementing, simulating, and analyzing an ARNDA model will be presented. The theoretical groundwork for the risk space and dependencies was introduced in Chapter 2. The methodology for defining the network as a group of network layers, assigning properties to nodes, and equations for propagating behaviors through the network was developed in Chapter 3. In the current chapter, the hierarchical network and the overall risk environment will be brought together by defining the dependent relationships between entities in the risk space, i.e., risks, and nodes in the hierarchical network, i.e., system components. Throughout the discussion, the same model of a theoretical supply chain introduced in Chapters 2 and 3 will be used to elucidate concepts and implementation details.

4.1 Hierarchical System Network

In this section, the system under study will be modeled as a network with dependencies on other nodes within the system as well as nodes in the risk network. The connections to the nodes within the risk network will be made by allowing these risk nodes to directly

affect the internal health variable of system nodes which are dependent on them.

4.1.1 Network Node Characteristics

Nodes in the previously defined dependency network analysis are described by two values. The first value relates to the node's internal ability to perform work without depending on external factors from other nodes in the network. This valuation should consider the state of the risk space the overall network resides in because an internal feature, such as workforce, can be heavily affected by a risk-based event, such as a pandemic. The second value involves the actual work performed by the node considering the work performed by upstream nodes on which it is dependent. Both of these values are sensitive to disruptions, albeit, these disruptions are modelled differently in the overall network. As such, they both have capacity to affect resilience which will be decomposed into three different categories as in [20]. The three categories for resilience capacity are absorptive, adaptive, and restorative. Each of these will be defined and some examples will be given with respect to internal state (i.e., internal health) and external output (i.e., operability). It may seem strange at first to differentiate between internal and external capacities since the network dependency methodology defines the behavior of the external performance as directly dependent on the internal node health. However, the strength of this dependency varies with the extent to which the performance is dependent on internal health, i.e., the parameters calculated in the dependency analysis formulation. The overall definitions will be consistent with the work of [19], [83] and the examples will be related to the watertight door supply chain defined in 3.3.1.

Absorptive and adaptive capacities are both defined in terms of a system's ability to

absorb the effects of a disruption. The difference between the two is that absorptive capacity is a measure of capacity that the system innately has without exerting noticeable effort while adaptive capacity implies the exertion of some effort to resist the disruption. For the watertight door supply chain, the installation node could increase absorptive capacity with respect to its internal health by keeping extra inventory on hand in the event that a supplier fails. It is important to note that actions that increase resilience from one perspective can actually increase risk in other areas. For example, Chopra and Sodhi note that while increasing inventory provides a small decrease in disruption risk and a larger decrease in delay risk, it can also increase inventory risk as inventory value depreciates over time and inventory of a particular type may become obselete [84]. The installation node could increase absorptive capacity with respect to its operability by sourcing components from multiple suppliers so that it is not as dependent on internal inventory as it would be if a sole-source supplier failed. Restorative capacity is the ability of the system to heal or recover after a disaster or disruption has occurred. From an internal health perspective, an example of investment in restorative capacity would be purchasing sufficient insurance to rebuild after a covered disaster. From an external health perspective, the company could investigate newer, more efficient designs to allow them to recover the same levels of productivity without requiring the same level of internal resources.

The ability of each node in the system network to perform its required functions is dependent on the random variables in the risk network (Fig. 34). The risk network nodes are shown as squares while the system network node is shown as a circle. The relationship between the two node types will define how a node will react in the event that a disruption

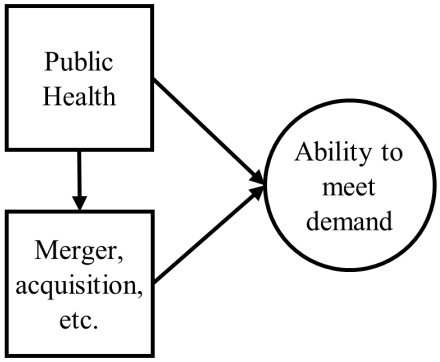


Fig. 34. Ability of a company to meet demand shown as a system node dependent on two different risks that are dependent on one another.

occurs. In general, occurrence of a disruption will increase the likelihood that a node is disrupted, and the magnitude of this occurrence will increase the severity of the node's reaction. The specifics of this relationship will be enumerated in the next section.

4.2 Adaptive Risk Network Dependency Analysis

In sections 2.2 and 3.2.3, two networks were defined along with two different analysis methods. For simplicity, the probabilistic graphical model from section 2.2 will be referred to as the risk network and the network dependency analysis model from section 3.2.3 will be referred to as the system network.

4.2.1 Overall Methodology

The overall methodology for building and simulating a complex network using ARNDA is as follows:

1. Identify the system network nodes and dependencies.

2. Collect data from the system under study.

3. Fit models as transfer functions between nodes in the system network.

4. Set risk network parameters and dependencies to be consistent with realistic risk environment.

5. Simulate the two network system to investigate vulnerabilities and overall behavior.

4.2.2 ARNDA Theory

In order to analyze and simulate the two networks, the connection between them must be defined. This connection will relate the output from the risk network to the vector of internal state parameters for nodes in the system network. In particular, the internal state parameter behavior will be dependent on the absorptive, adaptive, and restorative states resulting from the occurrence of a disruptive event in the risk network (Fig. 35). Following the lead of other researchers, the overall performance response curve will be a combination of linear functions as a node moves through the disrupted states. Immediately after the disruption, the performance will decrease at a constant rate, then stabilize, and finally increase at a constant rate prior to stabilizing at a new normal level of performance.

To keep the computational complexity reasonable, random variables modeled in the risk network will be considered to be exclusively discrete. In order to link these discrete variables to the continuous variables defined in the system network, a transformation must be defined. This transformation will leverage the fact that a common general dynamic response model for a disrupted node in a supply chain is assumed to be piecewise linear [17]. To allow for mapping of this piecewise linear function to behavioral states, the assumption is made

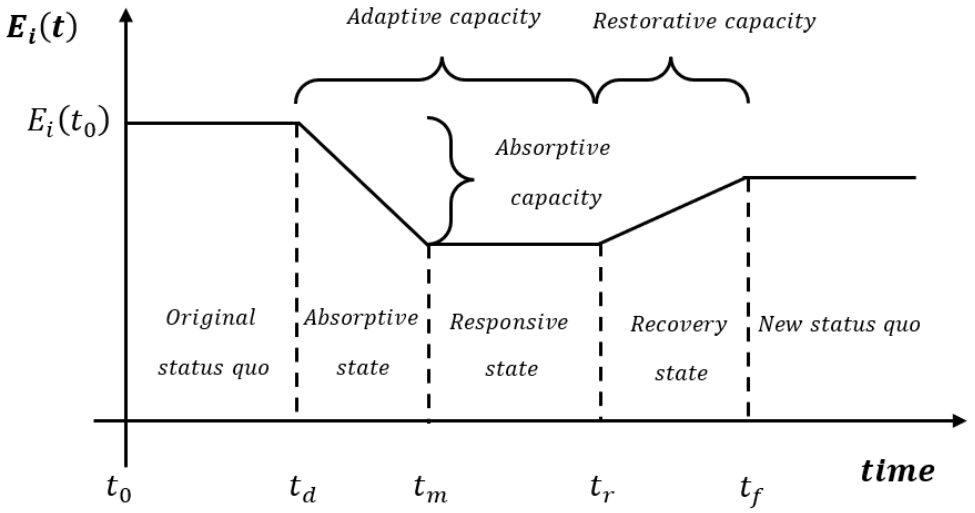


Fig. 35. Resilience capacities mapped to resilience states (adapted from [17]).

that the derivative of the internal health of the node with respect to time is constant for each state. These states are referred to as original status quo, absorptive, reactive, responsive, and new status quo. In the absorptive and recovery states, the internal health of the node has a non-zero rate of change with respect to time while all other states assume a constant internal health (zero rate of change with respect to time).

The probability that a node will be disrupted is conditioned on the probability that events will occur in the risk network. Since different systems can have very different responses to the same disruption, disruption parameters will be allowed to vary and a range of effects will be characterized through stochastic analysis.

While the occurrence of a disturbance is assumed to be dependent on nodes in the Bayesian risk network, once a disruption has occurred, the node moves from the normal status quo state to the absorptive state and, from this point forward, its state follows the Markov property. That is, its state in the current time step is dependent only on its state

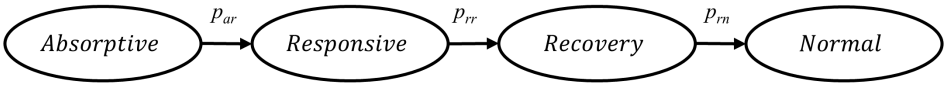


Fig. 36. Markov chain for resilience states.

in the previous time step. An additional assumption is that the node only moves forward in states and cannot transition backward. A final assumption is that the system remains in each state, on average, for a set amount of time.

The result of these assumptions is the Markov chain shown in Fig. 36. Each state operates with a given probability, p, to transition to the next state and a self-transition probability, 1–p. Since it has been assumed that a disrupted node will stay in each state for a given amount of time, the expression for these transition probabilities can be derived. The expectation for the time a node spends in each state can be calculated as the sum of the probabilities that it is in that state for each time step multiplied by the length of the time step. This can be expressed as:

$$\overline{T}_N = \sum_{i=0}^{N} (1-p)^i \cdot \Delta t, \tag{32}$$

where T_N is the time spent in the given state during N timesteps, p is the probability of transitioning to the next state, and Δt is the length of the timestep. The sum of this

geometric series can be calculated as:

$$\begin{aligned}\overline{T}_N &= \sum_{i=0}^{N} (1-p)^i \cdot \Delta t \\ &= \frac{\Delta t - \Delta t \cdot (1-p)^{N+1}}{1-(1-p)} \\ &= \frac{\Delta t - \Delta t \cdot (1-p)^{N+1}}{p} \\ &= \frac{\Delta t}{p} - \frac{\Delta t}{p} \cdot (1-p)^{N+1}.\end{aligned} \tag{33}$$

To obtain the expected time spent in a state for all possible sequences of state transitions, take the limit as the number of time steps goes to infinity, as follows

$$\overline{T}_N = \lim_{N\to\infty} \left[\frac{\Delta t}{p} - \frac{\Delta t}{p} (1-p)^{N+1} \right]. \tag{34}$$

Since p is a probability, its value is constrained to $[0, 1]$. Further, it is assumed that the value of p cannot be identically equal to either 0 or 1 since that would imply that the node either remained in the current state indefinitely or for zero time, respectively. Therefore, $p \in (0, 1)$ and, consequently, $(1-p) \in (0, 1)$. As a result, the limit evaluates as

$$\overline{T}_N = \lim_{N\to\infty} \left[\frac{\Delta t}{p} - \frac{\Delta t}{p} (1-p)^{N+1} \right] = \frac{\Delta t}{p}. \tag{35}$$

Consequently, the probability of transition out of a given state can be calculated given the expected time in the state and the length of the time step. All nodes in the system network that have non-zero probability of being disrupted have their internal health variable modeled as moving through states in a Markov chain. See Fig. 37 for a diagram showing this concept for one node in a system network. Since the behavior of each node in each state is defined as a linear function of time, the derivative of the internal state variable is to be constant with

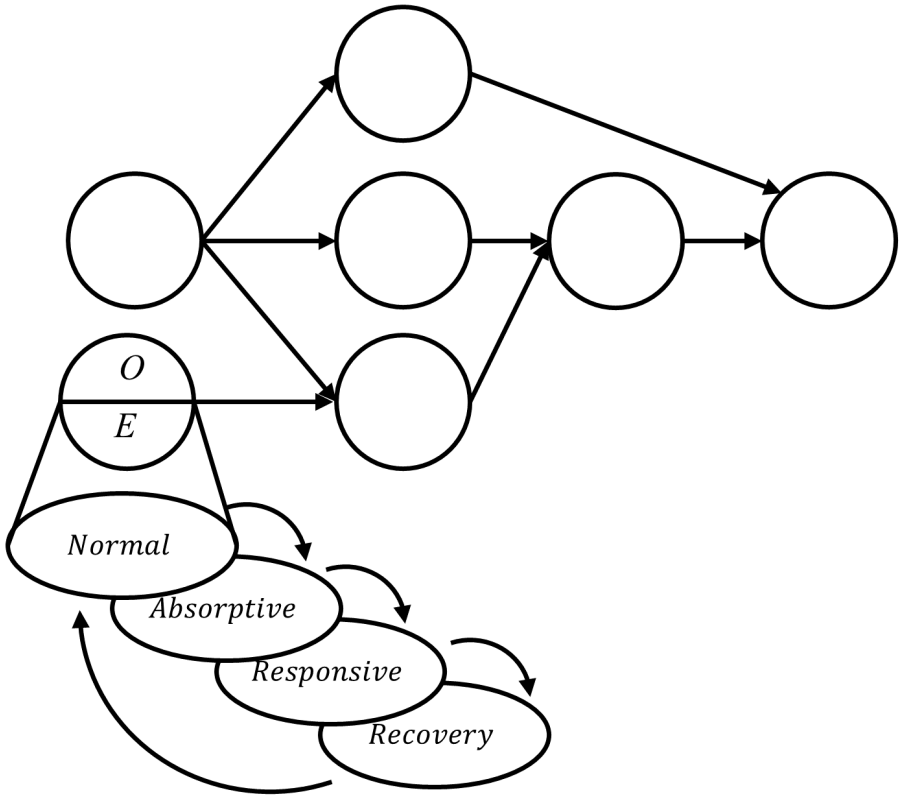


Fig. 37. System network with the dependence of the internal health of one node on a Markov chain of disrupted states shown.

respect to time. Therefore, the internal health state value, e_t, can be updated as follows:

$$e_t = e_{t-1} + \Delta t \cdot \frac{\partial e}{\partial t}. \tag{36}$$

In general, since the state the node is currently in will define its behavior, the above equation will be used to update relevant parameters of the statistical distribution to simulate the behavior of the nodes.

Once all internal state variables are updated for a given time step, the generalized

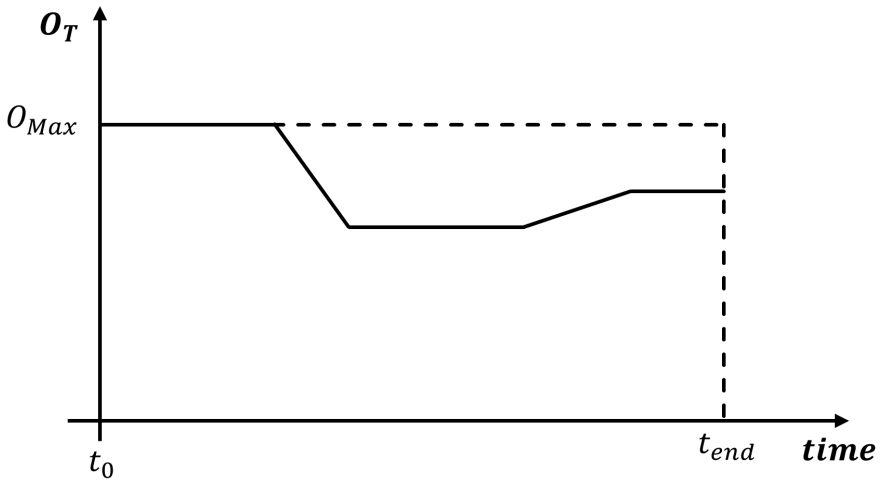


Fig. 38. Example operability, or performance, curve for a single node based on a single simulation run.

forward update method from the previous section can be applied to calculate all output variables. Due to randomness in ARNDA and the real systems under study, it is advantageous to run simulations many times. Visually assessing multiple individual traces where disruptions of differing magnitude are occurring at different points in time is not possible. Therefore, a methodology to evaluate the performance of each individual run so that it can be presented and compared with other runs is required. There are many measures of performance that may be applicable depending on the application. As an initial approach, the concept of average functionality from [85] will be used.

Consider the performance of a single node over time. An example is shown in Fig. 38. For a general performance, or operability, curve $O(t)$, the average functionality is defined

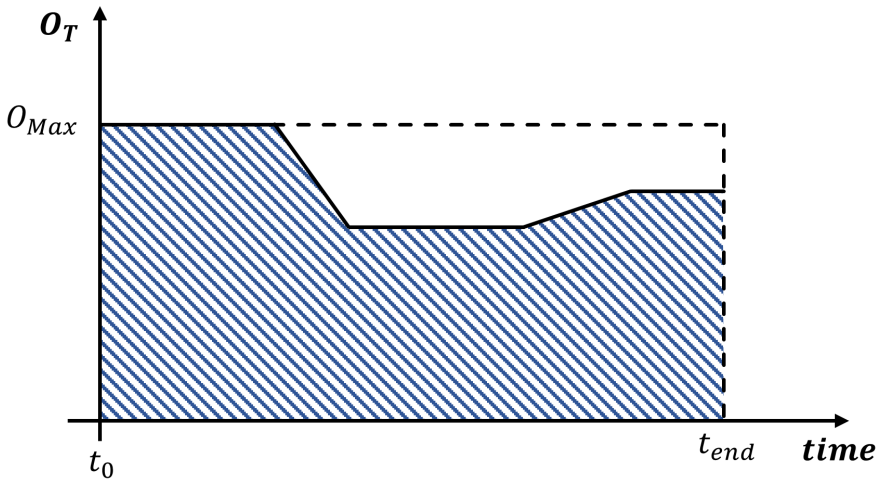


Fig. 39. Example operability curve showing the area under the curve as an example evaluation of the integral from equation 37.

as:

$$\overline{f} = \frac{1}{t_{end} - t_0} \int_{t_0}^{t_{end}} O(t)\, dt, \tag{37}$$

where t_0 and t_{end} are the start and end times of the simulation run, respectively. For the example simulation run in Fig. 38, the integral in (37) evaluates to the area under the operability curve as shaded in Fig. 39. In order to express the average functionality relative to the maximum functionality, equation 37 will be modified by dividing by the maximum possible value for operability as follows:

$$\overline{f}_{rel} = \frac{1}{O_{Max}\,(t_{end} - t_0)} \int_{t_0}^{t_{end}} O(t)\, dt. \tag{38}$$

This is the equivalent of dividing the area under the curve in Fig. 39 by the area of the rectangle formed by the dashed horizontal line at O_{Max}, the dashed vertical line at t_{end}, and

the axes.

4.3 Example Application: Disrupted Supply Chain

Now that all theoretical considerations have been expounded, the watertight door supply chain will be used to produce some numerical results. The risk and system networks are shown in Fig. 40 with the two layers, supply and demand, having the topologies shown in Fig. 20.

Before moving on, it is prudent to define one new notational convention used in Fig. 40 which will be utilized throughout the remainder of the dissertation to avoid overwhelming figures with arrows. When nodes are dependent on a node in the risk network, shading will be used to indicate the relationship. For example, the nodes associated with X_b, n_1, n_2, n_3, n_4, and n_7 are all shaded grey indicating that nodes 1, 2, 3, 4, and 7 are dependent on the random variable X_b. In static visualizations, this requires showing a different figure for each risk. However, in interactive visualizations, this will allow users to see which system nodes are dependent on risk nodes by selecting a risk node.

While the system network has been introduced at length in Sections 2.3.1 and 4.3, additional details regarding Risks a, b, and c are required. Risk a relates to competition for brass and steel from other companies. As larger companies strive to invest in resilience some are buying material years in advance to lock in supply. In these instances, other companies may not be able to get the share of the market they need. Risk b is a risk related to hurricanes that affect the United States East Coast. The organizations represented by nodes 1, 2, 3, 4, and 7 are all assumed to be located along the United States Eastern Seaboard. Finally, Risk c is related to a particular cybersecurity company facing a cyberattack. In this case, this

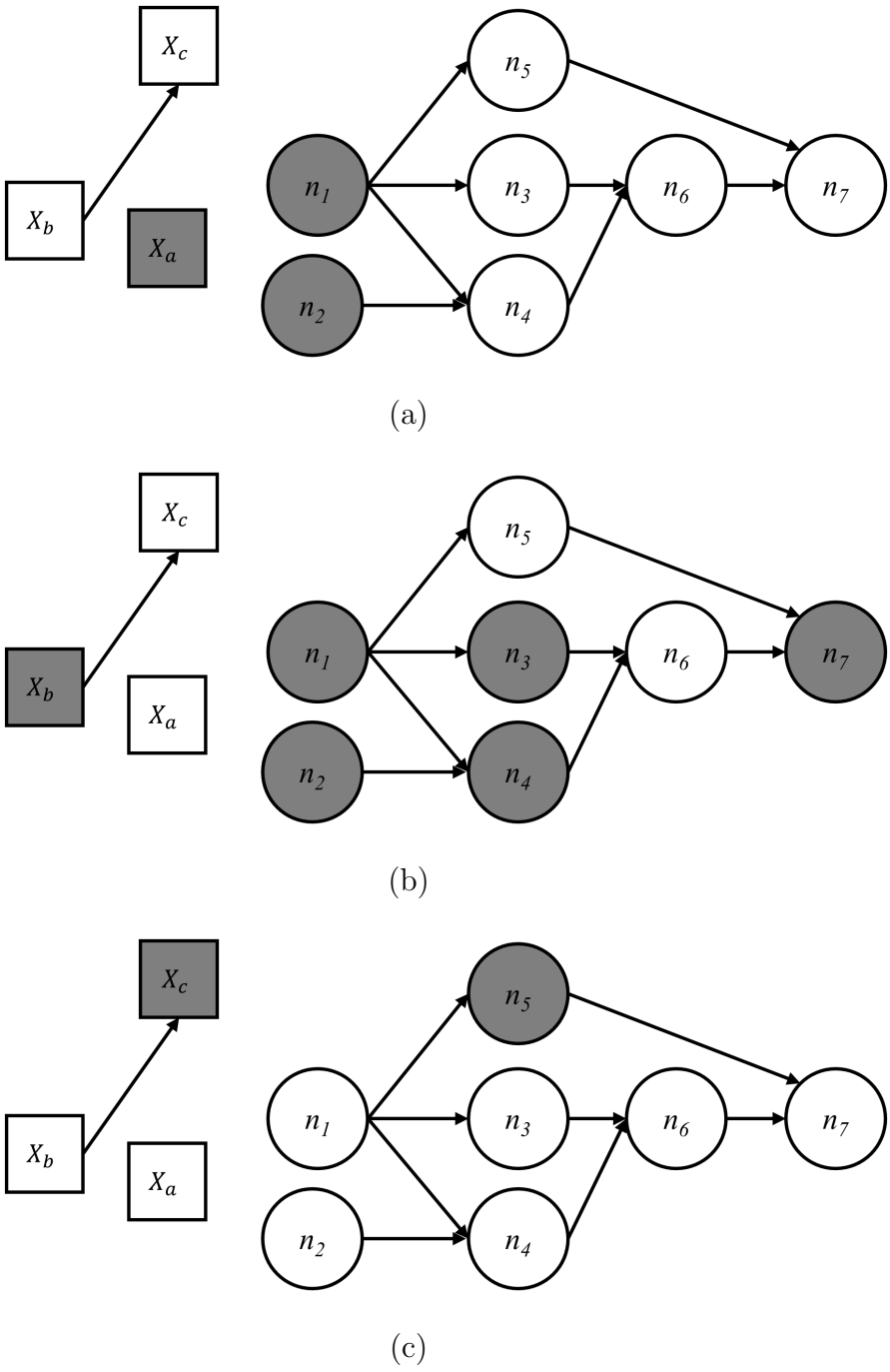


Fig. 40. Complete ARNDA model applied to watertight door supply chain example with shading indicating which nodes are dependent on (a) Risk *a*, (b) Risk *b*, and (c) Risk *c*.

company is headquartered on the Eastern Seaboard and provides technology related services for node 5. The dependency on Risk b indicates that attackers may be more likely to plan an attack when the company is facing a weather related disruption in order to achieve a compound disruptive impact.

4.3.1 Disrupted Single Run Behavior

In this subsection, a single run disrupted behavior for the watertight door supply chain will be exhibited in order to show overall behavior and performance.

4.3.1.1 Fitting Recurrent Neural Network (RNN)-enabled Network Dependency Analysis

While variability in demand and fluctuation in availability of resources were implemented in the models trained in Chapter 3, this ARNDA requires that the RNN-enabled network can handle disruptions. When training data-driven models, the data used should be reflective of the data that the operational model will be used on [61]. Therefore, the Discrete Event Simulation (DES) model was updated to reflect disruptions using the generalized procedure for implementing resource failures in [86]. The length of failure was 75% normal operation to 25% disrupted operation which is roughly consistent with the proportion of time for failure in [24]. An exponential distribution was used to simulate the length of the normal operation and the length of the disruption. The length of normal operation and disruption were varied in lengths including approximately 30 normal days followed by 10 disrupted days, 63 normal days followed by 21 disrupted, and 21 normal followed by 7 disrupted. These ranges were selected after reviewing the literature and finding that other work varied from zero to twenty disrupted days [87]–[90]. The disrupted nodes were selected

to be the sets of nodes in Fig. 40.

The training procedure was the same as in Chapter 3, and the time step used was one day so that the disrupted behavior could be seen clearly. Test results from a single case where Risk b was active causing the steel, brass, door, frame, and watertight door nodes to be disrupted are shown in Figs. 41 and 42. Boxplots representing the error for 55 test cases are shown in Fig. 43. All values for the Mean Squared Error (MSE) are below 0.05 except the brass demand node which has a mean of 0.107. The plots showing the RNN-enabled Network Dependency Analysis model versus the data from the DES model show that the overall behavior is captured reasonably well, and the goodness of fit is confirmed by the small values for MSE across all test cases. It should be noted that the values for MSE are noticeably increased for the disrupted case when compared to the non-disrupted case from Chapter 3. In fact, the increase ranges from 3% to 367% and is 117% on average. This can be attributed to the fact that the disrupted model exhibits more complex behavior and in fact is operating in two modes, i.e., disrupted and non-disrupted. One way to address this and improve the overall model accuracy that will be considered for future work is to classify the behavior of the model by, for example, using anomaly classification and training separate models for the disrupted and non-disrupted behaviors exhibited by the model.

4.3.1.2 Single Run Performance

Now that the RNN models have been trained, the model can be run with individual disruptions. As an example, the door and frame supply nodes were selected to be semi- and fully disrupted. The definitions of semi- and fully disrupted were 50% and 0% operability, respectively, which is consistent with [24]. These states were compared to normal operation

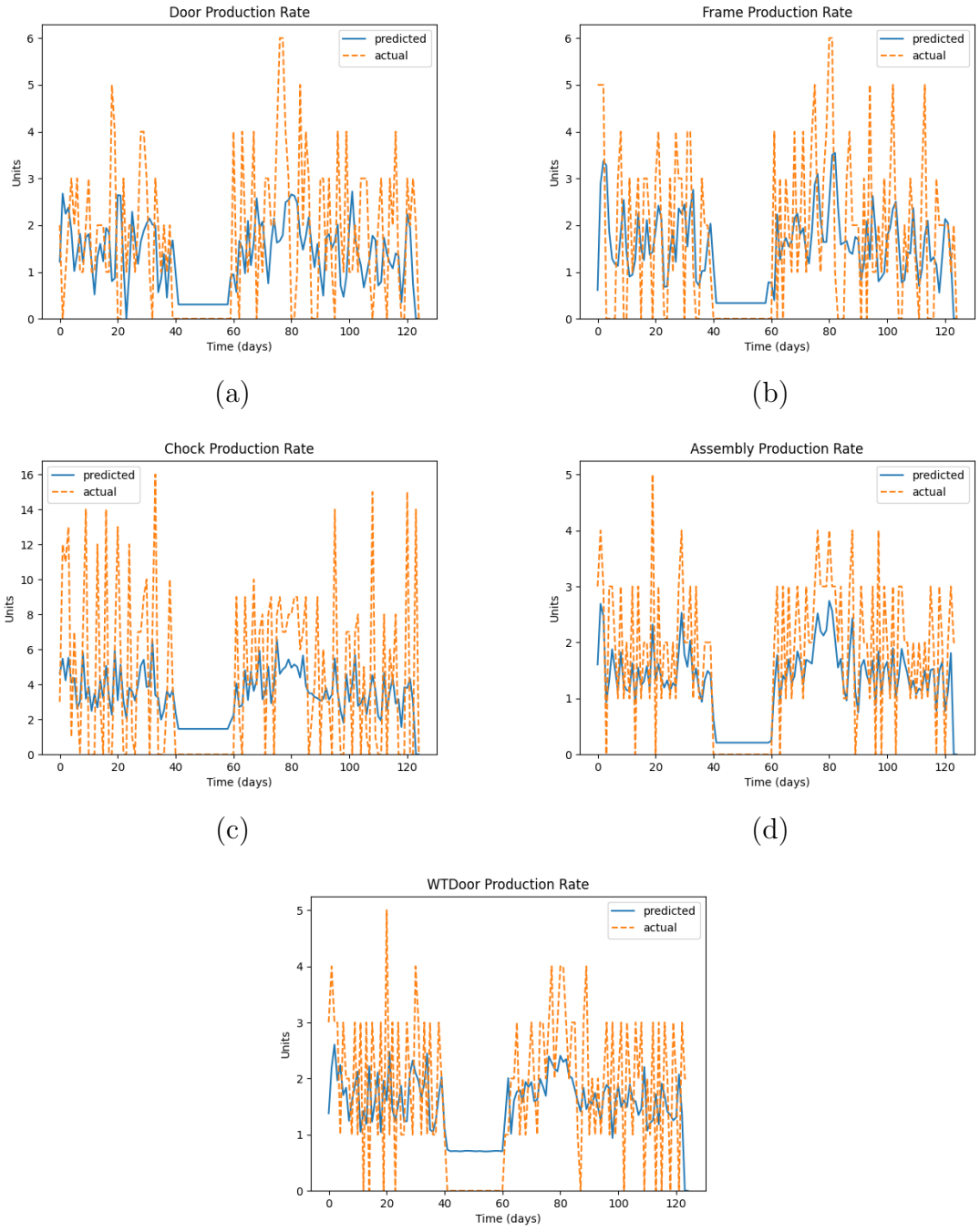


Fig. 41. Single run test example disrupted by inclement weather for supply layer results for (a) door, (b) frame, (c) chock, (d) assembly, and (e) watertight door installation nodes.

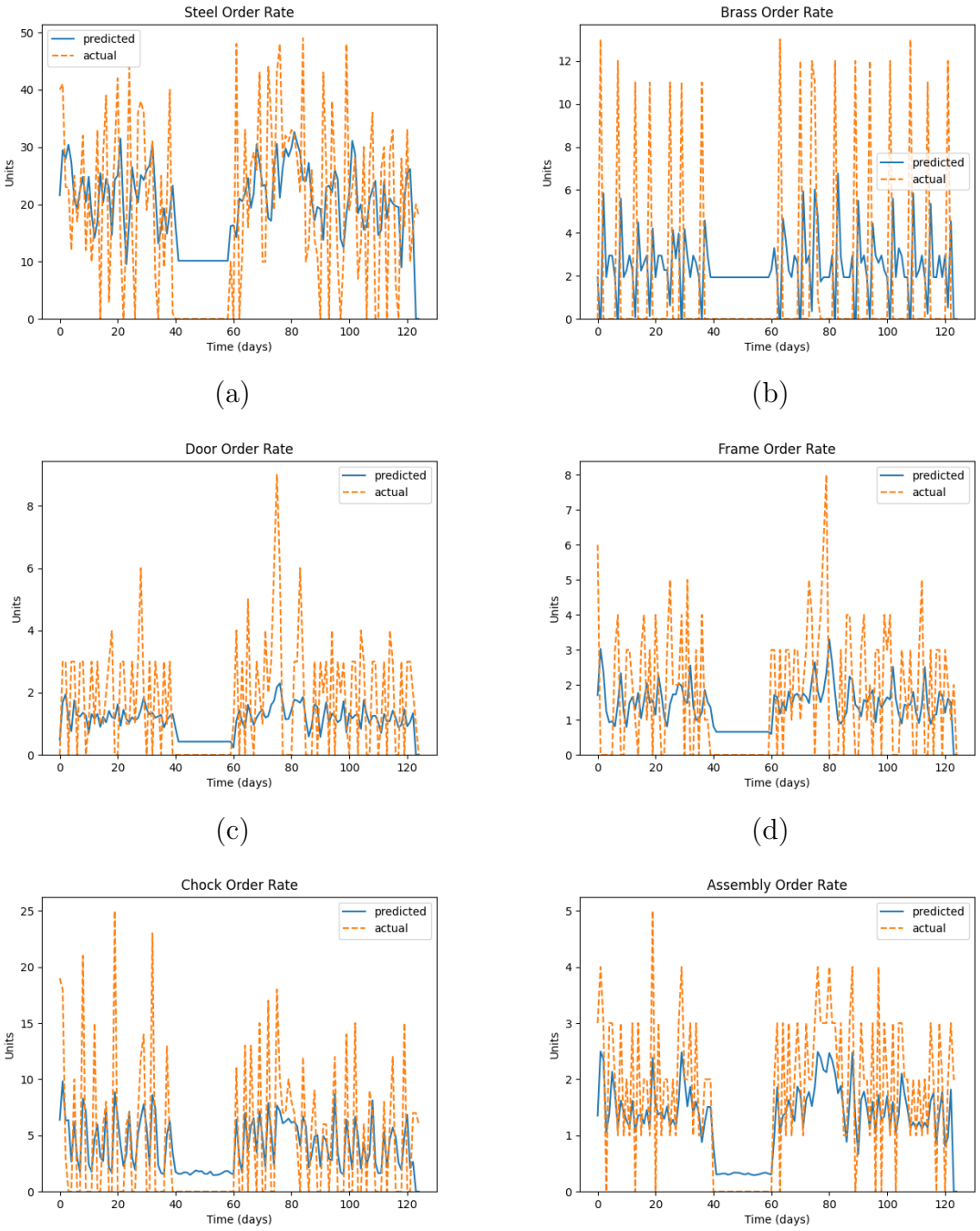


Fig. 42. Single run test example disrupted by inclement weather for supply layer results for (a) steel, (b) brass, (c) door, (d) frame, (e) chock, and (f) assembly nodes.

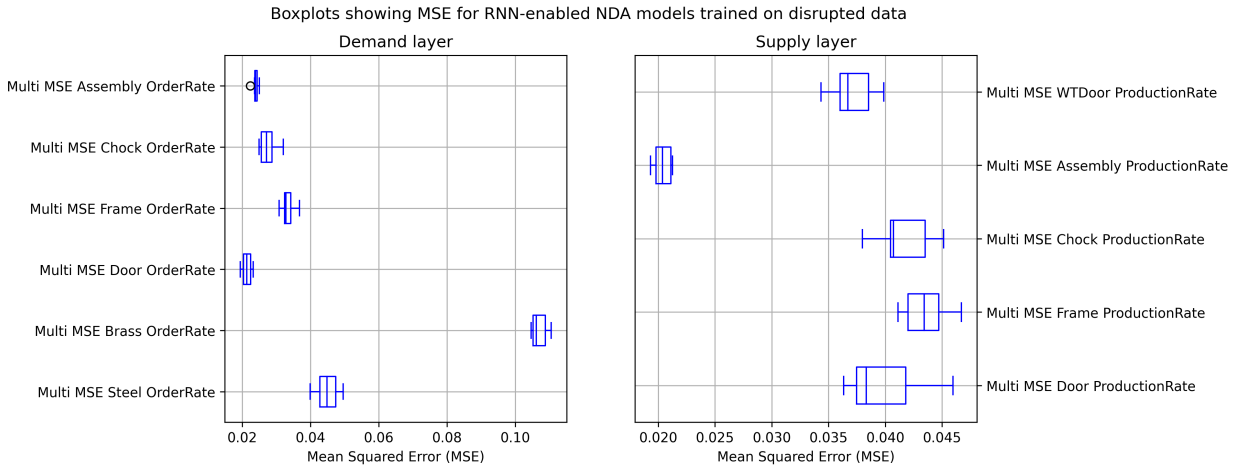


Fig. 43. Boxplots showing test error by node for model trained on disrupted data.

which was considered 100%. As with any model, boundary conditions are required for the input nodes of each layer.

Poisson distributions were used to estimate the demand for the watertight door node and supply for steel and brass nodes with means of 4.3, 16, and 1.5 units per day. These values were consistent with those used for the DES model which was assumed to represent the real system under study in this case. Modeling supply and demand as a rate of items demanded or produced in a given unit of time and drawn from a Poisson distribution is consistent with [91]. Though it has been shown that this assumption may not match reality, it is the most common approach in the literature [92]. Therefore, it will be used in this introductory example but will not be used in actual case studies unless the data indicate it is appropriate. The definition of a Poisson process can be found in most calculus-based probability and statistics texts, for example Evans and Rosenthal [93], and is provided below

as follows:

$$p_Y(y) = P(Y = y) = \frac{\lambda^y}{y!} \cdot e^{-\lambda}, \tag{39}$$

where y is the random variable described as the rate of items per unit time and λ is the mean overall rate per unit time.

For both the semi- and fully disrupted cases, the length of disruption was 20 days and started on day 30. The performance was found using the methodology described in section 4.2.2 which was to find the area under the curve for the measure of performance of interest, in this case either units produced or units ordered/demanded. The time window for averaging was selected to be a five-day work week, but this could be adjusted based on the interests of the study. Then, for the disrupted cases, this value was divided by the same value for the non-disrupted case. Plots showing the results of this analysis for the entire time period are shown as Figs. 44 and 45.

For the simulation runs shown in Figs. 44 and 45, several interesting observations can be made. The first is that the model is capturing the forward ripple effect in the supply layer (Fig. 44) from the door (Fig. 44(a)) and frame (Fig. 44(b)) nodes through the assembly node (Fig. 44(d)) and finally to the watertight door installation node (Fig. 44(e)). Also, both the assembly node and the watertight door installation node take longer to recover after the end of the disruption period than the door and frame nodes.

For the demand results (Fig. 45), the disruption is transferred to the demand layer for the affected door, frame, and assembly nodes (Fig. 45(c), (d), and (f)). In addition, there is an upstream ripple effect that affects the steel order rate seen in Fig. 45(a). There is little to no effect on the brass and chock nodes for this disruptive event. Considering the brass

node, there is a disruption, but it is small. This indicates that there is a weak dependence of the network on the brass supplier which makes sense given that only the door supplier was dependent on the brass supplier while the door, frame, and chock suppliers were all dependent on the steel supplier.

As for the chock node, it is topologically separated from the door and frame nodes and forms a path directly from the steel node to the watertight door node. Because the steel node is a root node for the supply layer, its supply is an input to the system. The watertight door is a root node for the demand layer, so its demand is also an input to the system. Therefore, the chock node is isolated from the disrupted. This is an artifact of the small size and overall topology of the network and should be something that is considered and checked for when employing and applying results from the methodology.

4.3.2 Stochastic Analysis

In this subsection, the Monte Carlo method will be used to simulate a large number of randomized simulation results and report on statistically generalized results.

4.3.2.1 Performance Distribution

In this example, supply and demand are positive valued real numbers. The random variables in the watertight door supply chain example are the supply and demand operability at each node in the network. Under the RNN-enabled functional dependency analysis described previously in Chapter 3, the operabilities, or performance values, for each node are non-linear combinations of the operabilities of their predecessor nodes and their own operabilities at previous time steps since the activation functions of the nodes in the RNN

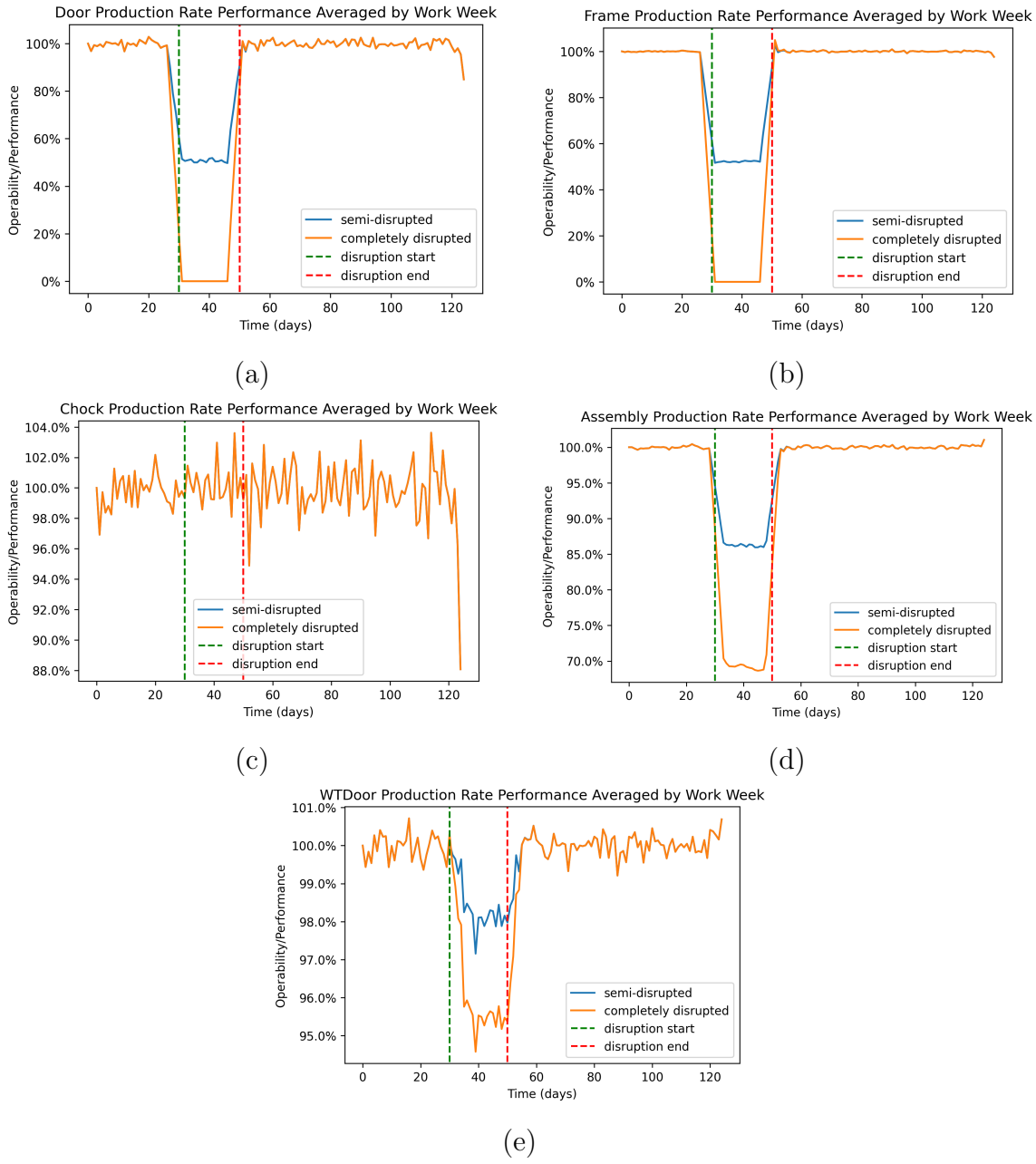


Fig. 44. Supply layer single run results from full-ARNDA methodology with disruptions at door and frame nodes for (a) door node, (b) frame node, (c) chock node, (d) assembly node, and (e) watertight door installation node.

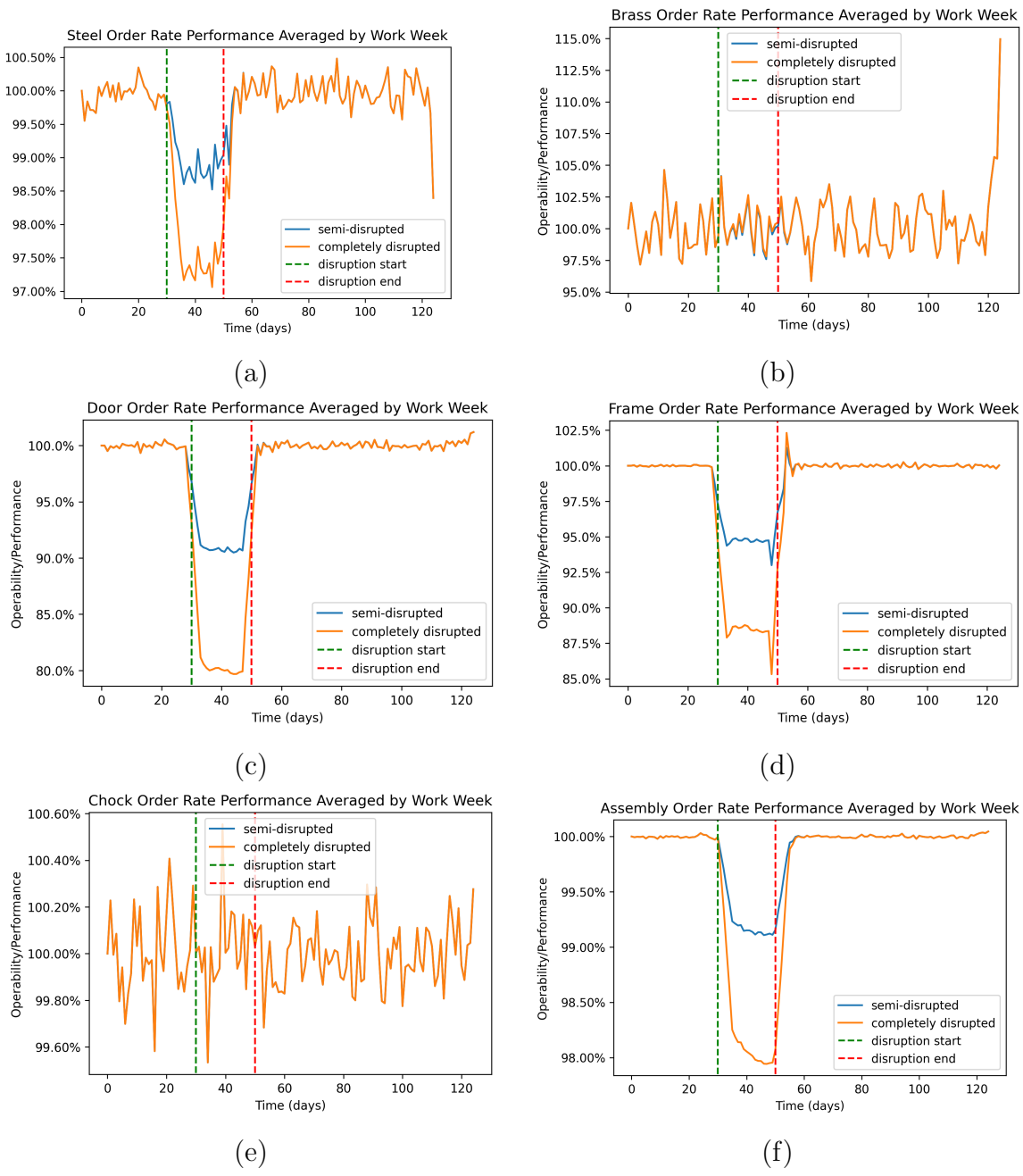


Fig. 45. Demand layer single run results from full-ARNDA methodology with disruptions at door and frame nodes for (a) steel node, (b) brass node, (c) door node, (d) frame node, (e) chock node, and (f) assembly node.

layers are non-linear. If the transformations were linear and all internal health values were independent, the probability density function for the operability of each node could be calculated as the convolution of probability density functions for the nodes it depends on [93]. Since these transformations are non-linear and it is clear from Fig. 40 specifically as well as, in general, based on the assumptions of the ARNDA methodology, that the behavior of these nodes is not independent, the probability density function cannot be calculated as a function of the inputs. Therefore, while the single run analysis gives insight into the overall behavior of the system, it is important to simulate the system stochastically.

4.3.2.2 Disruption Probability

Now, the probability of disruption needs to be estimated. Based on a survey of work in supply chain disruption research, local disruptions were usually assumed to be between 0.5% and 20% while global disruptions were estimated between 0.5% and 1% [94]–[97]. In this work, the global disruptions will be considered as prior probabilities and the local will be posterior probabilities based on additional evidence provided in each node. Conditional probability tables will be defined to reflect this assumption. Probability tables for the three nodes in the risk network are shown in Tables 14, 15, and 16. For the nodes in the system network, a 1% chance of disruption will be assumed under normal circumstances, increased to 80% when one relevant risk is occurring and 90% for two relevant risks occurring. As a result, an experimental matrix with the probability of each state of risk occurrences can be produced.

TABLE 14. Probability distribution for Risk a - Raw Material Competition.

	$p(a)$
Occur	0.05
Not Occur	0.95

TABLE 15. Probability distribution for Risk b - Inclement Weather, Hurricane.

	$p(b)$
Occur	0.01
Not Occur	0.99

TABLE 16. Conditional probability distribution for Risk c - Cyberattack.

	Risk c	Occur	Not Occur
Risk b	Occur	0.15	0.85
	Not Occur	0.05	0.95

4.3.2.3 Results

A set of runs for each possible set of risk occurrences has been produced (Table 17). Then, descriptive statistics showing characteristic values for each network node have been calculated and plotted. In an applied case study, subject matter experts could select sets of risks they were most interested in for analysts to study in greater detail. Based on Table 17, it can be seen that the most likely state is the state where no disruptions occur, i.e., N, N, N. The next two most likely states are the ones where either Risk A occurs, or Risk C occurs. This section will provide preliminary thoughts related mainly to the methodology in preparation for application to more complex applications in later chapters. In particular, previous research has found that supply chain management often does a good job preparing for and being resilient to recurrent, low-impact events but is not as competent at addressing less frequent, high-impact events [84]. By combining the analyses for the risk and the system models, this dissertation allows for the investigation and quantification of both scenarios.

TABLE 17. Set of possible test cases based on risks with calculated probability of occurrence. *Note:* O: Occurred; N: Not Occurred.

Simulation case	Risk A	Risk B	Risk C	Probability
1	O	O	O	0.0075%
2	O	O	N	0.0425%
3	O	N	O	0.2475%
4	O	N	N	4.7025%
5	N	O	O	0.1425%
6	N	O	N	0.8075%
7	N	N	O	4.7025%
8	N	N	N	89.3475%

Recall that supply and demand are modeled as positive valued measures of performance. Since the estimates can be made for the lower and upper limiting value and the expected value, or mean, the triangular distribution will be used to model the data. For this case, the mode of the triangular distribution will be 50% with upper and lower limiting values of ±10%. This is selected as opposed to the normal distribution since these measures of performance can only take on positive values and the normal distribution can take values on $(-\infty, \infty)$ The Probability Density Function (PDF) and Cumulative Distribution Function (CDF) of this distribution are as follows:

$$f(x) = \begin{cases} 0 & x < a \\ \frac{2(x-a)}{(b-a)(c-a)} & a \leq x < c \\ \frac{2(b-x)}{(b-a)(b-c)} & c \leq x \leq b \\ 0 & x > b. \end{cases}, \tag{40}$$

$$F(x) = \begin{cases} 0 & xa \\ \frac{(x-a)^2}{(b-a)(c-a)} & a < x \leq c \\ 1 - \frac{(b-x)^2}{(b-a)(b-c)} & c < x < b \\ 0 & x \geq b. \end{cases}, \tag{41}$$

where a, b, and c are the minimum, maximum and mode of the distribution, respectively.

Nodes were assumed to be semi-disrupted. This was modeled using a triangular distribution for the internal health with a mode of 50% operability and extents of ±10%. Disruptions had the probabilities of 1% in general increasing to 80% and 90% when one or two of the risks the node was dependent on occurred. These are roughly consistent with recent results from Hosseini and Ivanov in [98] where the probability of a single disruption was, at worst, near 77%. Finding the probability of two independent disruptions would give approximately 94.7%, and these values rounded to the nearest 10% are 80% and 90%, respectively. When a disruption occurs, the mode of the node behaves in a manner consistent with Fig. 35. An example of triangular probability distribution showing a historgram with 10,000 samples, the PDF, and CDF is shown in Fig. 46.

The time period for each run was 150 days, and each run was repeated 300 times. The mean start time for the disruptions was day 30 with absorptive, responsive, and recovery periods having mean durations of 5, 10, and 5 days, respectively. For each simulation, disrupted and normal operation values were calculated and then compared using the ratio of the area under the curve over a five day period. The result is that the values can exceed 100% when they exceed the value of operation under normal conditions, i.e., assuming no

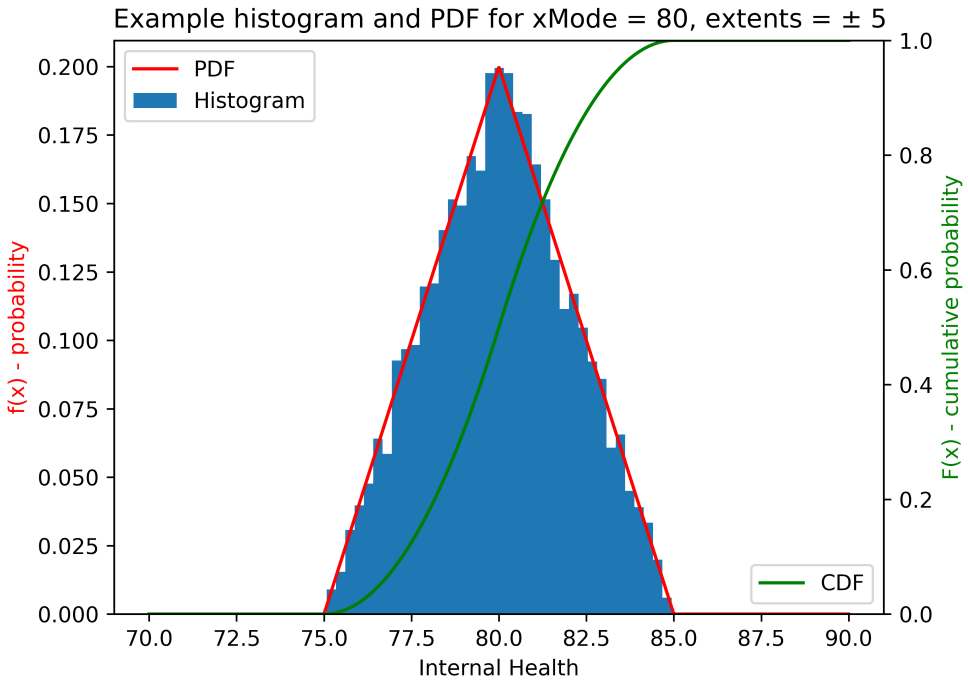


Fig. 46. Example histogram, PDF, and CDF for a triangular distribution with mode = 80, minimum = 75, and maximum = 85.

disruptions. In general, there will be an upper limit on the capacity of any node within the system that must be respected within the model. In this case, the results never exceed 115% of normal operation. This would imply that as long as normal operation did not exceed 87% of the node's capacity that the upper limit on capacity would be respected. However, when applying results to a real rather than a simulated system, the overall upper limit on capacity should be thoroughly investigated and applied.

The results are shown aggregated as violin plots in Figs. 47, 48, 49, and 50. Each violin plot is a visualization of the approximate probability density function distribution of the underlying values of operability for all 300 runs. Originated by Hintze and Ray, violin plots usually show the distributions vertically rather than horizontally and mirror them over their vertical center-line [99].

Throughout this analysis, Cases 1 through 7 will be compared to Case 8 since this is the case where no risks have occurred to create an increased risk of disruption. Risk b affects the greatest number of nodes and therefore will act most like a global disruption. The images showing the results when Risk b occurs are Figs. 48 and 50. All eight of these figures show that the results for performance are more constant across all nodes than in other cases. In addition, the traces between the minimum and maximum extents (horizontal end caps) tend to be more spread out when compared to the corresponding scenarios where Risk b does not occur, which are Figs. 47 and 49. For the supply layer, most of the nodes see more movement in the lower extent when Risk b is active. The results for the demand layer are similar except for the frame node which sees a marked increase in its upper extent. As expected, the brass and steel suppliers are highly affected when Risk a is occurring which

corresponds to competition from other material suppliers. This ripple effect can also be seen moving downstream through the system most clearly when this effect is applied in isolation and comparing Case 8 (Fig. 47(a-b)) to Case 4 (Fig. 49(a-b)).

Another interesting finding comes when comparing Case 7 (only Risk c occurring) to Case 8 (no risks occurring). Risk c only directly affects the chock manufacturer. Comparing Case 8 (Fig. 47(a-b)) and Case 7 (Fig. 47(c-d)) on the supply layer, the lower extents of the frame and chock nodes and the upper extent of the watertight door installer are increased. At first it may seem strange that the performance increases when a disruption is applied. However, inspecting the disrupted demand plot (Fig. 47(d)) shows that there is an increase in demand for steel which is a reasonable explanation for the resulting increases in production. Amplifying disruptions in demand magnitude in supply chains are so common they are referred to as the bullwhip effect, and recent literature confirms that ripple effects from disruptions do interact with these bullwhip effects [100]. The ability to study these types of counterintuitive results is a desired, highly valuable feature of this modeling approach.

4.4 Summary

This chapter detailed the implementation, simulation, and analysis of the ARNDA methodology. The ARNDA methodology combines Bayesian networks for risk modeling with a generalization of network dependency analysis enabled by RNNs to model disruption propagation. In particular, this provides answers to research questions 1, 2, and 3 from Chapter 1. The first research question asks how cyclic features can be incorporated into an analysis methodology based on Bayesian networks and RNN-enabled network dependency analysis, both of which are acyclic, by definition. The solution to this problem comes from

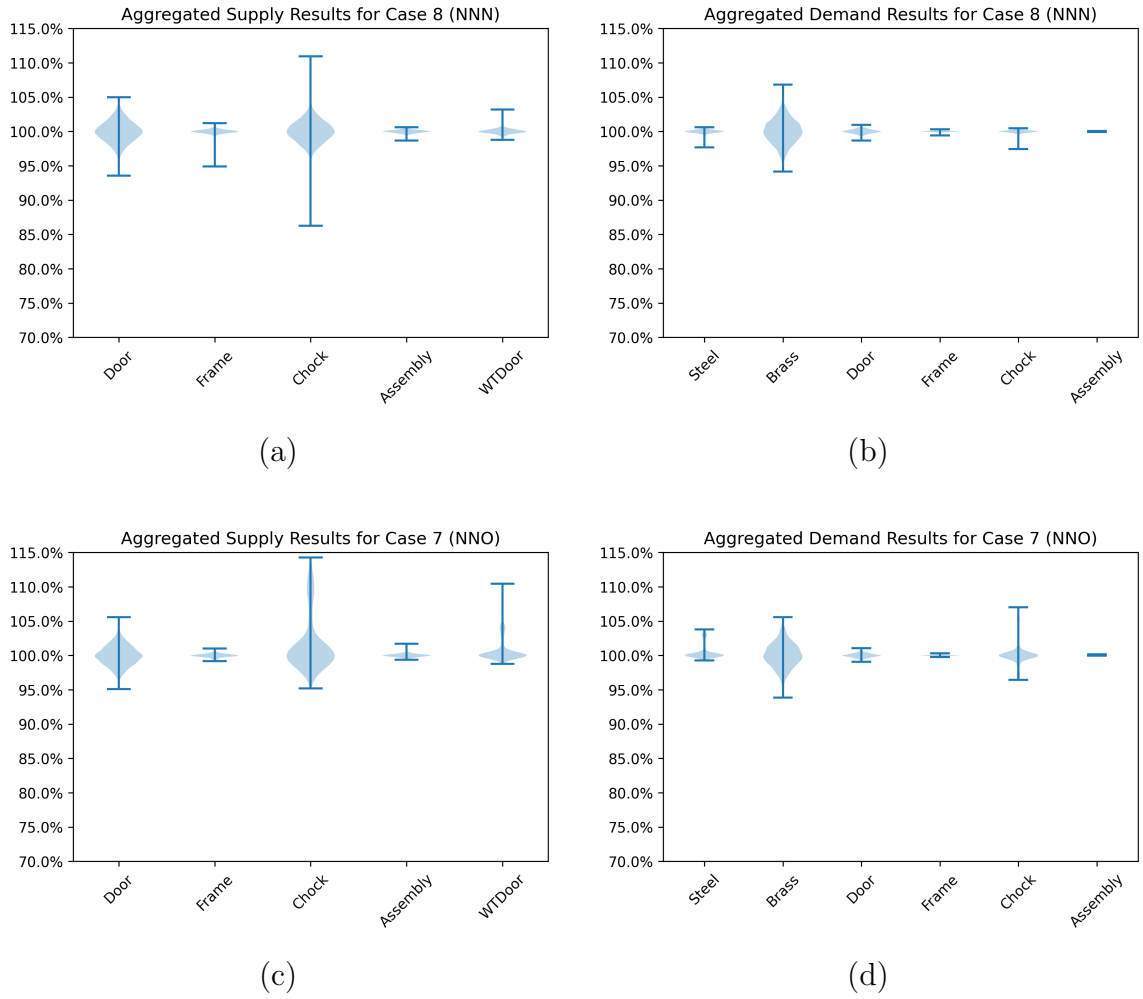


Fig. 47. Plots showing aggregated results for no risks occurring (case 8, (a) and (b)) and for Risk c occurring but not Risk a or b (case 7, (c) and (d)) for supply (a) and (c) and demand (b) and (d) layers.

separating the output (or performance) variables into layers and then aggregating the results. These layers can have different topologies which allows for forward and backward propagation between nodes. The second research question deals with the dynamic, time-dependent response of systems over time. In ARNDA, time is incorporated by using an RNN-enabled network dependency analysis model trained on time-dependent data. The

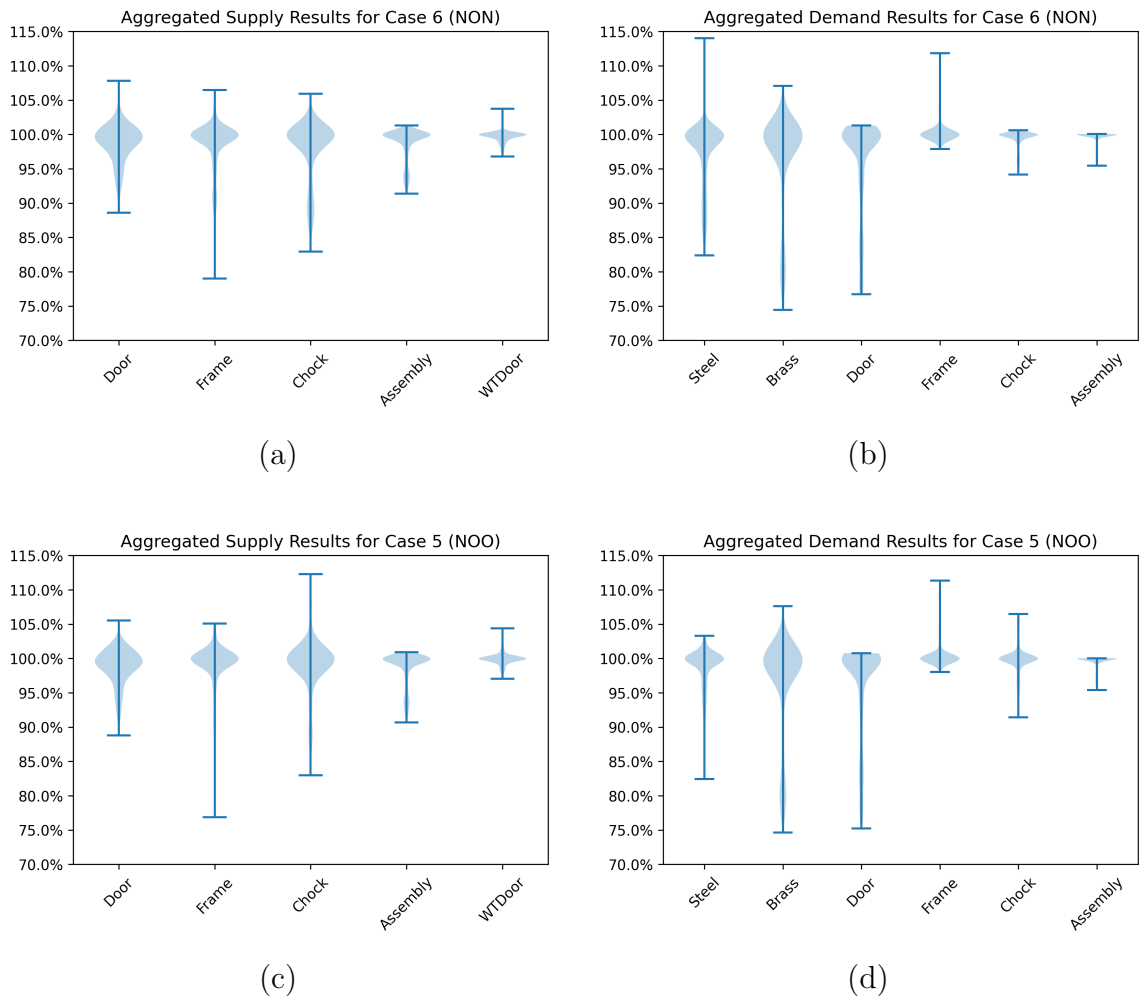


Fig. 48. Plots showing aggregated results for Risk b occurring but not Risk a or c (case 6, (a) and (b)) and for Risks b and c occurring but not Risk a (case 5, (c) and (d)) for supply (a) and (c) and demand (b) and (d) layers.

disrupted states are modeled using a Markov chain which allows the static Bayesian model to be connected to the temporally dependent system network modeled using RNN-enabled network dependency analysis. The third research question addresses whether or not continuous system variables can be incorporated. Most of the recent literature studying the ripple effect of risks and disruptions through complex systems uses Bayesian networks with

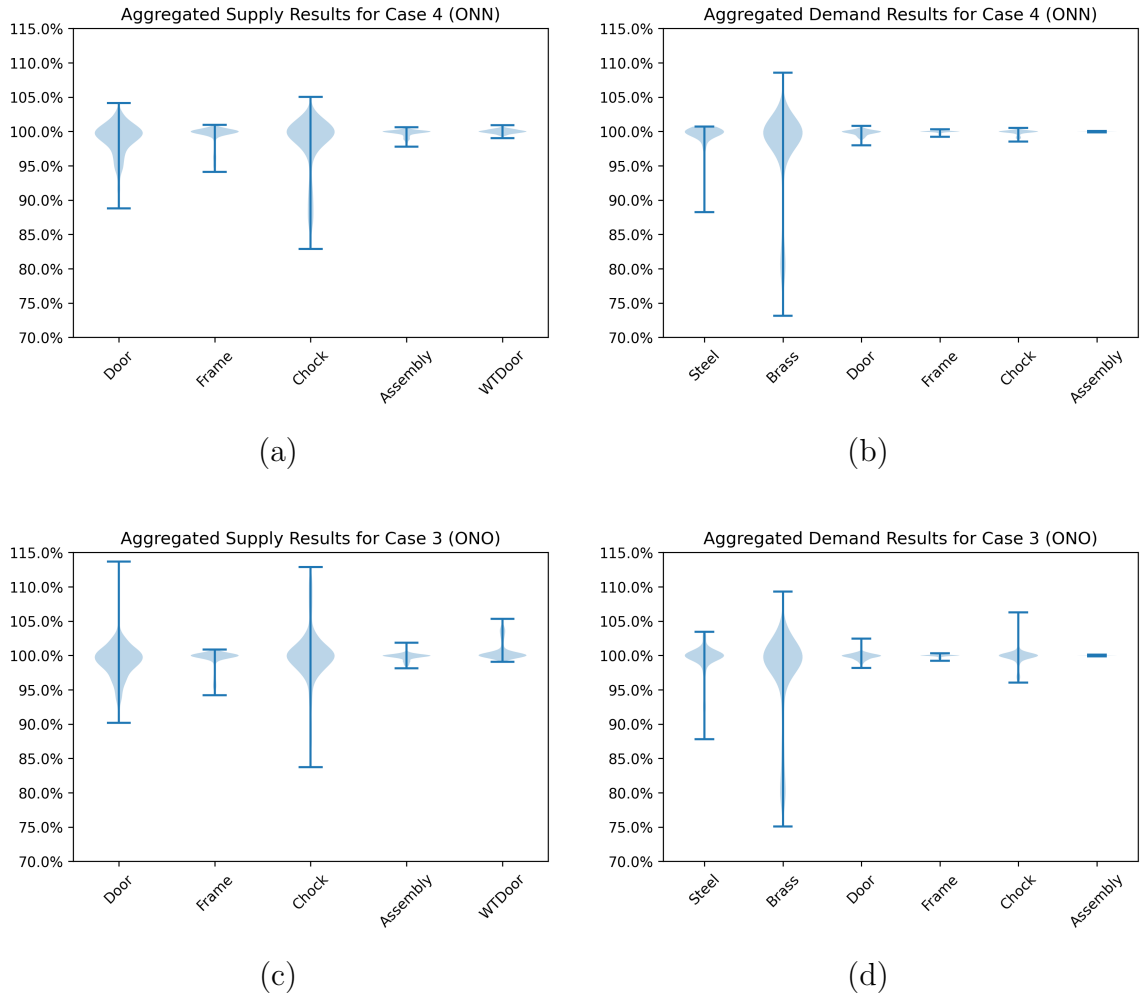


Fig. 49. Plots showing aggregated results for Risks a occurring but not Risk b or c (case 4, (a) and (b)) and Risks a and c occurring but not Risk b (case 3, (c) and (d)) for supply (a) and (c) and demand (b) and (d) layers.

discrete random variables in order to avoid prohibitively long simulation and analysis times. In order to address this, ARNDA uses Bayesian networks with discrete random variables to model system risks while using the continuous RNN-enabled network dependency analysis to define and analyze the system behavior.

In addition to addressing the first three research questions posed in this dissertation,

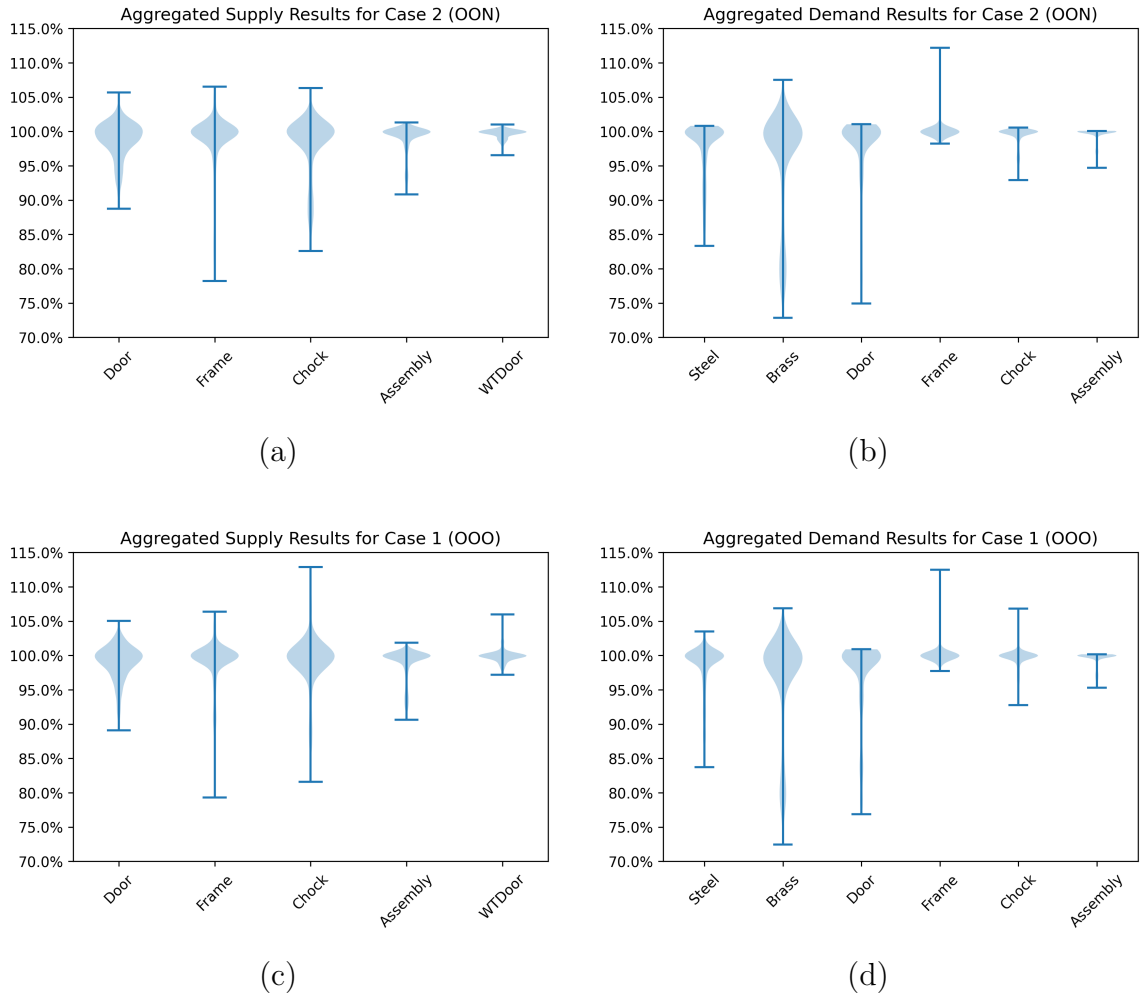


Fig. 50. Plots showing aggregated results for Risks a and b occurring but not Risk c (case 2, (a) and (b)) and all risks occurring (case 1, (c) and (d)) for supply (a) and (c) and demand (b) and (d) layers.

this chapter also provided details on the methodology and implementation that will support additional applications in the upcoming case study. The probabilistic risk spaces and RNN-enabled network dependency analysis were combined to define the overall theory of ARNDA through the connection between the risk network, and the system network was defined in terms of a disruption probability and Markov chain behavior for the nodes' internal health.

This chapter closed by providing an example implementation of the ARNDA methodology to a watertight door installation supply chain. In order to more fully understand the system's behavior, single run results were studied. Then, a generalized performance metric for each run was introduced to allow comparisons through stochastic analysis. This leads into the next chapter where a larger case study on port logistics and logistics will be analyzed to show and understand additional features of the ARNDA methodology.

CHAPTER 5

CASE STUDY: PORT LOGISTICS AND RESILIENCE

Long term planning documents for maritime ports have identified investment in advanced equipment capability as a goal that will decrease their environmental footprint and operational costs while simultaneously increasing their container handling capacity [101]. These investments involve those in electrification, intermodal transportation and cyber physical systems. Because of this digitalization, port operations can increasingly be divided into at least three networked layers including traditional operations, information technology, and operational technology. These advancements position port systems as a perfect case study to the Adaptive Risk Network Dependency Analysis (ARNDA) methodology presented in Chapter 4. The initial conceptualization of the application of ARNDA to a maritime port in order to assess the internal health and vulnerability of nodes is developed by Smith *et al.* in [9]. This chapter will extend that conceptual work by incorporating metrics for vulnerability, identifying thresholds for hypervulnerability, and implementing the methodology in order to present and analyze results.

5.1 Problem Statement

In this work, the ARNDA methodology will be applied to a maritime port that serves freight with multi-modal transport including ships, rail, and trucks. Since many ports, including the one under study (Fig. 51), have separate stacks for rail, the study of truck operations can be separated from rail operations. The current work will only study

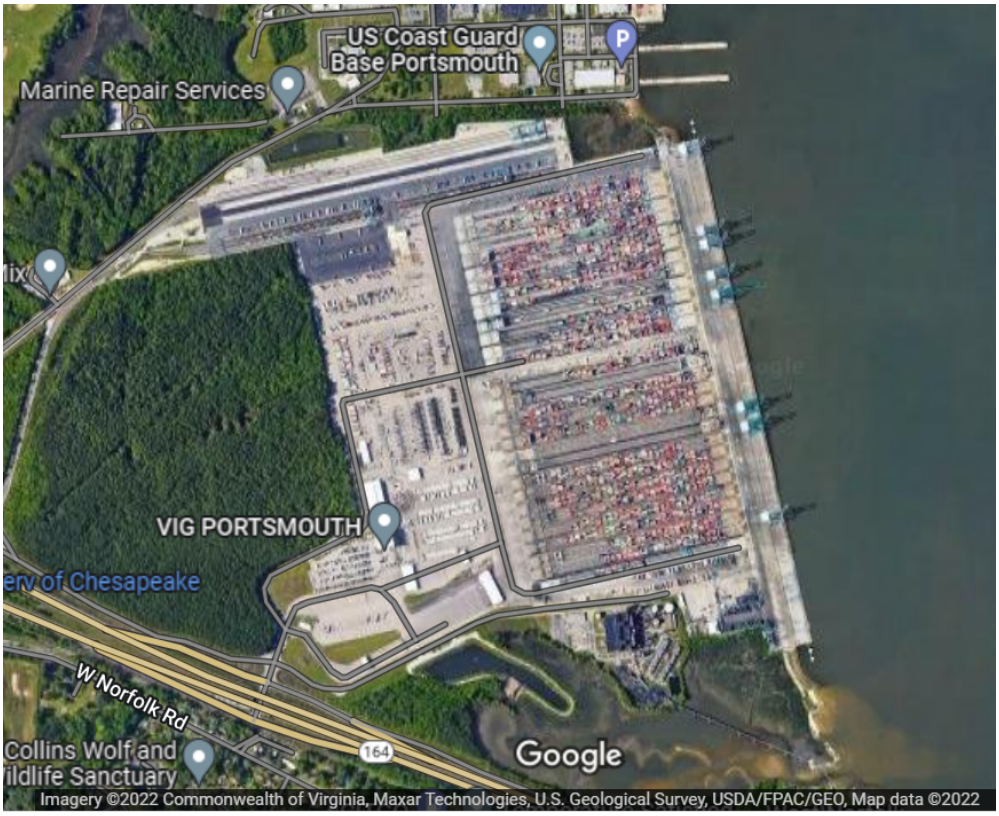


Fig. 51. Satellite image of the Port of Virginia's Virginia International Gateway Terminal. Map Data: Google, Commonwealth of Virginia, Maxar Technologies, U.S. Geological Survey, USDA/FPAC/GEO [102].

operations involving ships, trucks and the container stacks dedicated to serving containers traveling by those modes of transport.

5.1.1 System Network

In order to apply ARNDA, the operability, or performance, of each node must be quantified. It is important to monitor performance of ports using key performance indicators compared to target, or goal, values to close the feedback loop and generate increases in

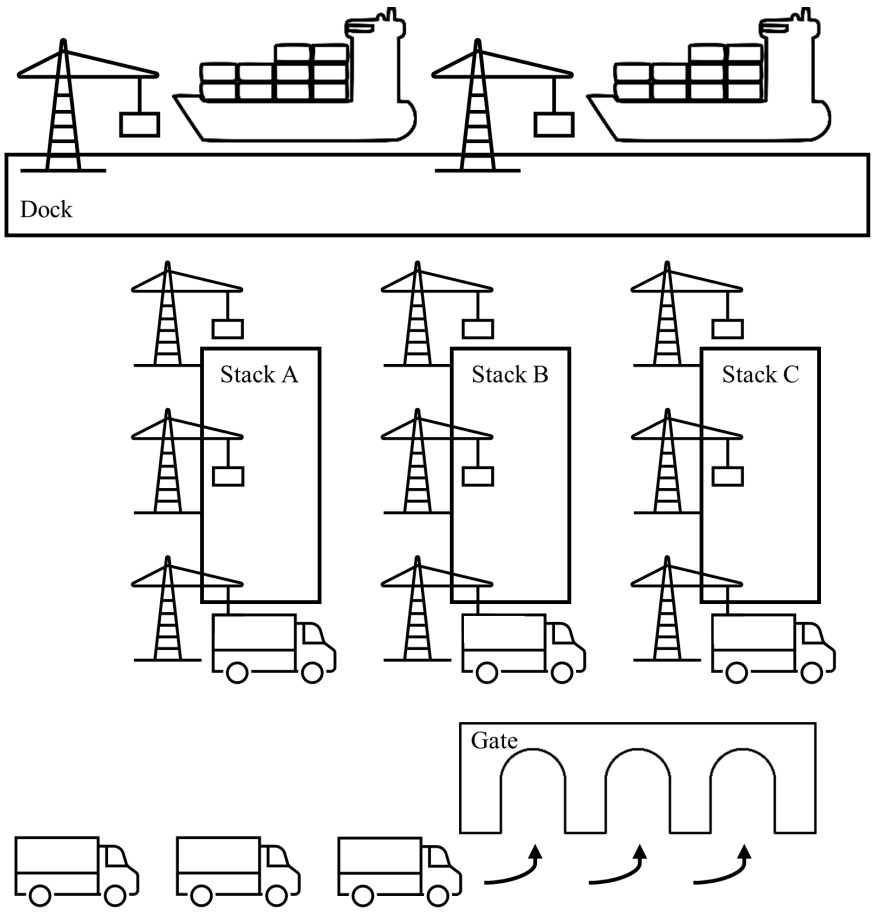


Fig. 52. Port schematic showing ship and truck modes of freight container transportation.

port performance [103]. Performance indicators have evolved significantly since 1976 when they were divided into only financial and operational considerations [103]. More recent publications include additional categories for performance measures such as safety, connectivity, and environmental measures of performance [104]. As climate change, technological advancements, and other factors drive changes in operations across a variety of industries, maritime ports will be required to adapt how they conduct and assess their operations. However, access to data sets that support quantification of these performance indicators is confounded by issues including proprietary or secure nature of data [105]. As such, this work will focus on a compromise to use the most relevant measures which can be calculated from the data sets that are available.

In this work, two types of measures will be used based on the data obtained. Each measure will be represented as its own layer in the system network. These measures are divided into two categories as follows:

1. Time-based: Length of time an event takes from start to finish or length of time between two consecutive events.

2. Count-based: Number of occurrences of an event in a given period of time.

The overall model with all nodes grouped into their respective areas of port operation is shown in Fig. 53. Each node will appear in each of the two layers and have an operability value for (a) the count of trucks served and (b) service time (inclusive of wait time). The overall flow of traffic through the port can be traced from bottom to top in Fig. 53.

There are two ways to consider the topology of the layers. First, it could simply be assumed that both layers are connected with the edges directed parallel to traffic flow

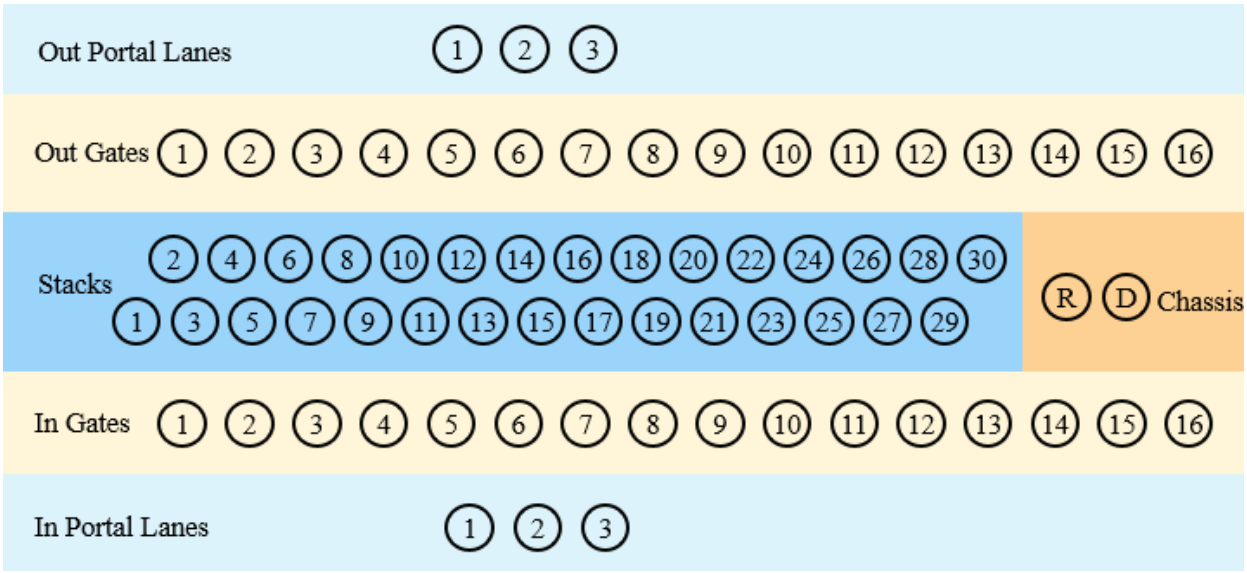


Fig. 53. Schematic showing all seventy port model nodes.

through the port. However, this would not account for the information control systems implemented at gates and portals in modern ports [106], [107]. Therefore, the mean service time layer will be directed in the opposite direction of truck traffic flow to represent operational information provided to upstream nodes. The capacity-based layer, which is count of trucks served, will be directed in the same direction as the flow of traffic through the port. In the operational context of the port, this allows the industrial control systems to time the release trucks as downstream nodes finish processing trucks. Trucks entering through the in portal are released to the in gates where they are held in queues. The in gates receive notification from the container stacks or chassis area when there is sufficient availability to release the next truck to be served. After being served at the container stacks or the chassis area, the trucks proceed through the out gates and exit the port through the out portal lanes.

Fig. 54 shows the two layers discussed in the previous paragraph along with nodes aggregated by area of operation. Fig. 55 shows the fully connected seventy node model, but the edges are less clear due to the large number of connections. Since the edges are less clear in Fig. 55, only the layer for the count of trucks is shown, and the figure is not repeated for the service time layer. The application of ARNDA was trained and evaluated on both the aggregated and full configurations. Trucks enter the port through one of the three in portal lanes and then proceed to one of the sixteen in gates. After proceeding through an in gate, the truck travels to either the assigned stack (one of thirty) or the chassis area. Trucks can visit multiple stacks and the chassis area during a single visit to the port if they are performing multiple tasks while on site. Once the truck has finished processing it proceeds to one of the sixteen out gates. Finally, the truck exits the port through one of the three out portal lanes.

5.1.2 Risk Network

Finally, the risk network for the port is shown in Fig. 56. The same conventions from Chapter 4 are used to differentiate between risk and system nodes using square and circular nodes, respectively. For clarity, Fig. 56 only shows dependencies within the risk network and system network but not the dependencies between them. In the remainder of this section, each set of risk nodes will be described, and a separate figure will be presented showing the appropriate dependencies.

Supply chain delays and disruptions during the Covid-19 pandemic have caused considerable disruption to port cargo operations worldwide in recent years. These include

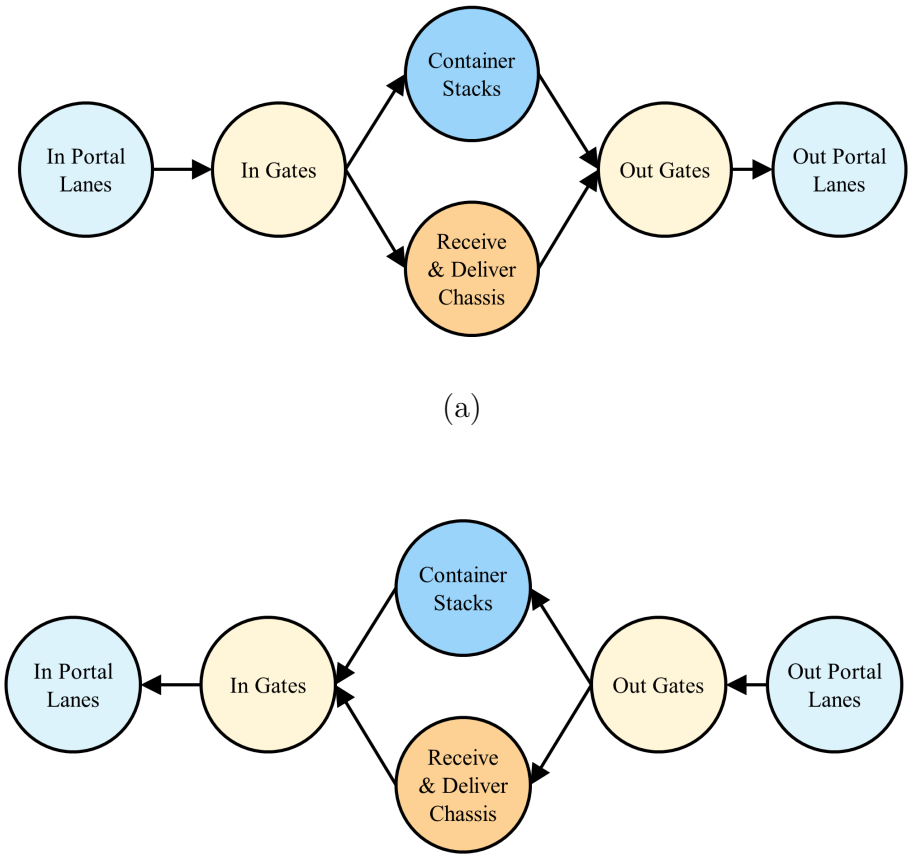


Fig. 54. Schematic showing port model with nodes aggregated by port area of operations for (a) count of trucks served layer and (b) service time for each truck (inclusive of wait time) layer.

backlogs of cargo, congestion, and bottlenecks [104], [108]. Further compounding this problem is a shortage of truck drivers with an estimated deficit of 80,000 nationwide [109]. Since truck drivers are compensated by the amount of cargo they successfully deliver and their driving hours are limited by law, time spent in port congestion severely limits their ability

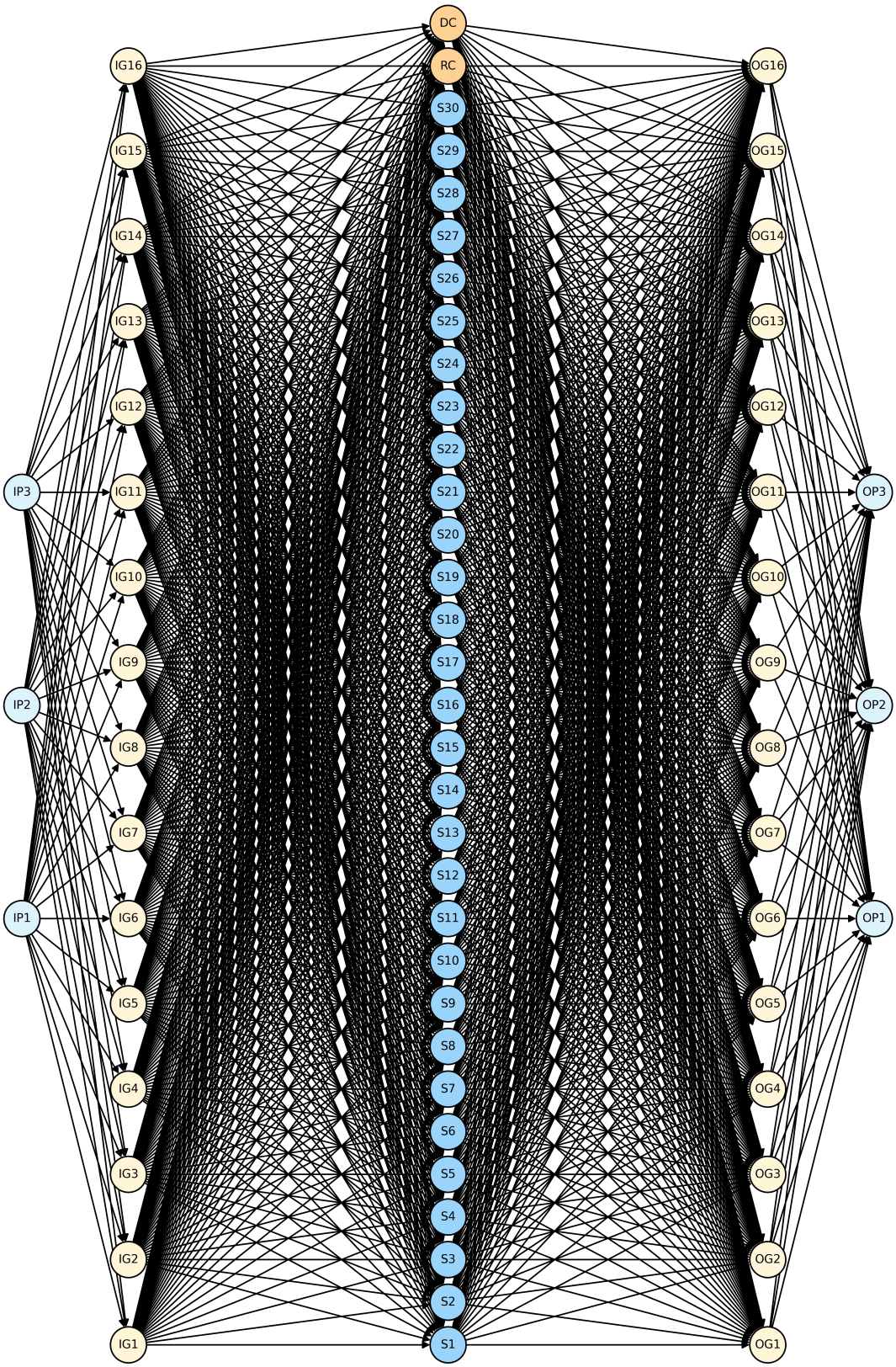


Fig. 55. Schematic showing full seventy node port model with edges.

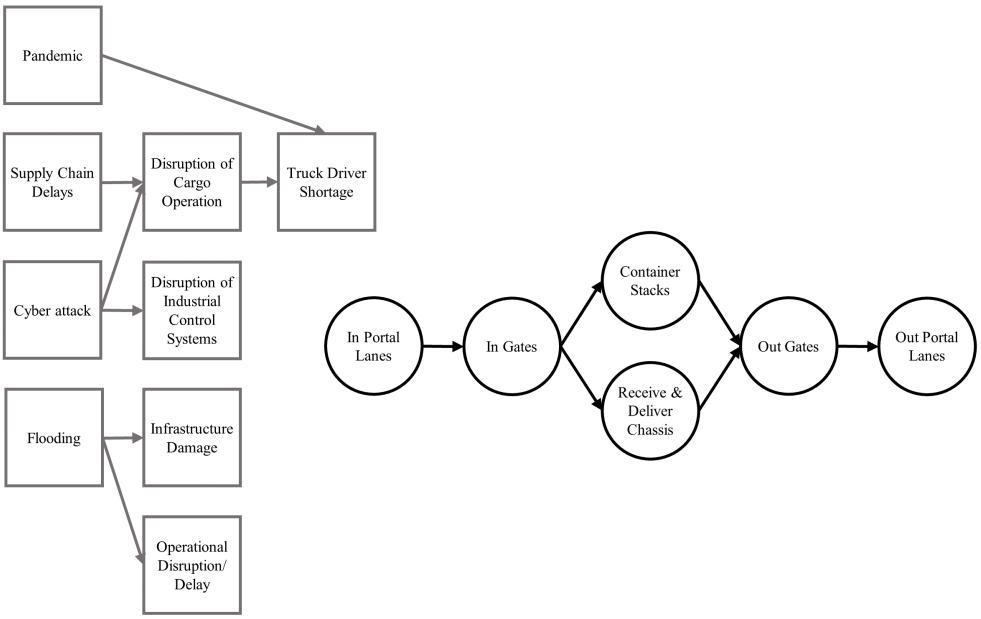


Fig. 56. An illustrative combined risk (grey square nodes) and system (black circular nodes) networks for the port example.

and will to work [110]. Therefore, along with other contributing factors, supply chain delays have been a driver of trucker shortages. Fig. 57 shows the resulting dependencies where the port's in portal lanes, in gates, and container stacks are all dependent on the disruption of cargo operation from the risk network. In addition, the container stacks node is dependent on the truck driver shortage risk node which is in turn dependent on the disruption of cargo operation.

Another prominent risk in port operations is the threat of cyber attack. In February 2022, European ports had experienced disruptions due to a cyber attack, and in September 2021 a major United States port was targeted as well, so it is clear that these are not uncommon events [111], [112]. According to a note released by the United States Department

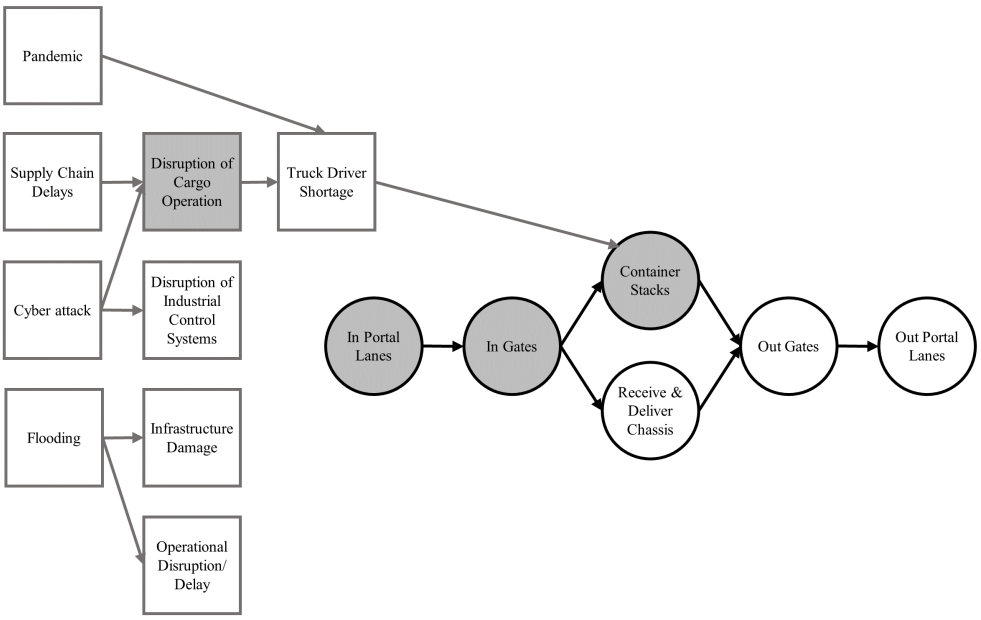


Fig. 57. Combined risk (grey square nodes) and system (black circular nodes) networks for supply chain delays in the port example. Shading is used to show the dependency between nodes in the risk and system networks.

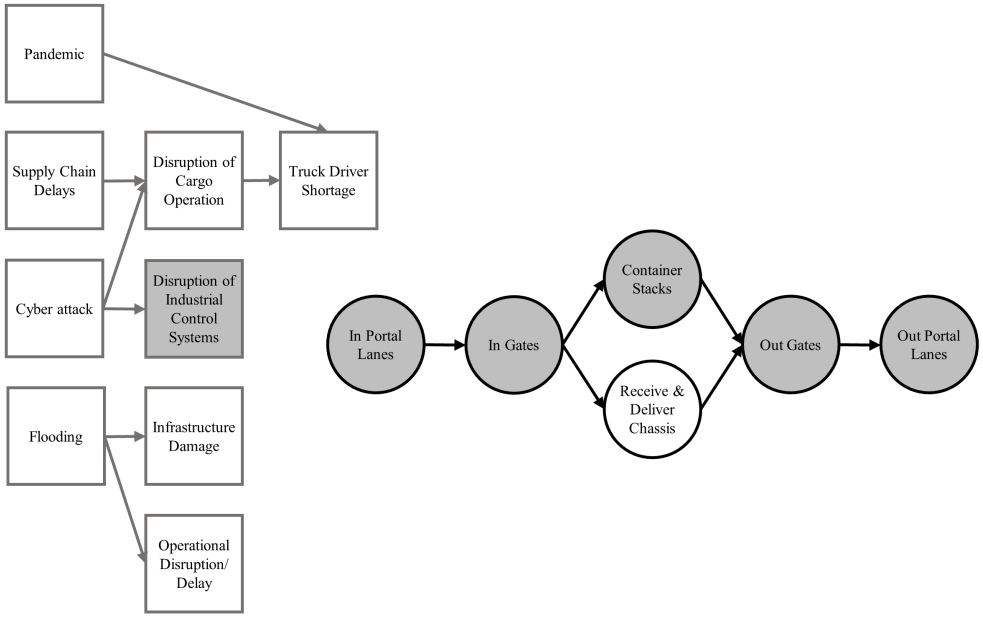


Fig. 58. Combined risk (grey square nodes) and system (black circular nodes) networks for industrial control system disruptions in the port example. Shading is used to show the dependency between nodes in the risk and system networks.

of Homeland Security, cyber attacks on ports include targeted disruptions of cargo operation and industrial control systems [107]. The port system nodes dependent on the risk node for disruption of cargo operations were shown previously in Fig. 57. Fig. 58 shows that all nodes except the receive and deliver chassis nodes are dependent on the risk node for the disruption of industrial control systems. This is because all other nodes in the port under study are at least partially automated with computerized gates and/or cranes. The area where the chassis are received and delivered is staffed by human mechanics.

As ports are necessarily located near the coast, they are highly susceptible to risks from flooding due to climate change. Flooding can result in infrastructure damage as well

as operational disruption or delay throughout the port [113]. Flooding is a global risk for the port, and both risks that are dependent on flooding affect all port system nodes.

While in Chapter 4 it was possible to perform an exhaustive stochastic analysis for all combinations of risk scenarios due to the smaller number of risk scenarios, here only four highly likely risk scenarios will be considered. Each of these scenarios has been selected because it is relevant as evidenced by current events and stands to have considerable impact on port operations. The four selected scenarios are as follows:

1. Normal operation: Operation under normal conditions with a 1% uniform probability of disruption for all nodes.

2. Cyber attack: A cyber attack where attackers have targeted nodes in the in portal and in gate operational areas of the port (Fig. 58).

3. Truck driver shortage: Truck driver shortage in the midst of congestion partially caused by supply chain delays. In portal lanes and in gates will experience a single risk, while container stacks will experience two risks (Fig. 57).

4. Coastal flooding: Flooding is causing delays for groups of stacks that is worse near high tide and improves during the rest of the day. While all nodes are susceptible to flooding, nodes at lower elevations will flood more regularly.

5.2 ARNDA Implementation

The ARNDA methodology will now be applied to data from a maritime port on the east coast of the United States. The author has access to data from a limited time period

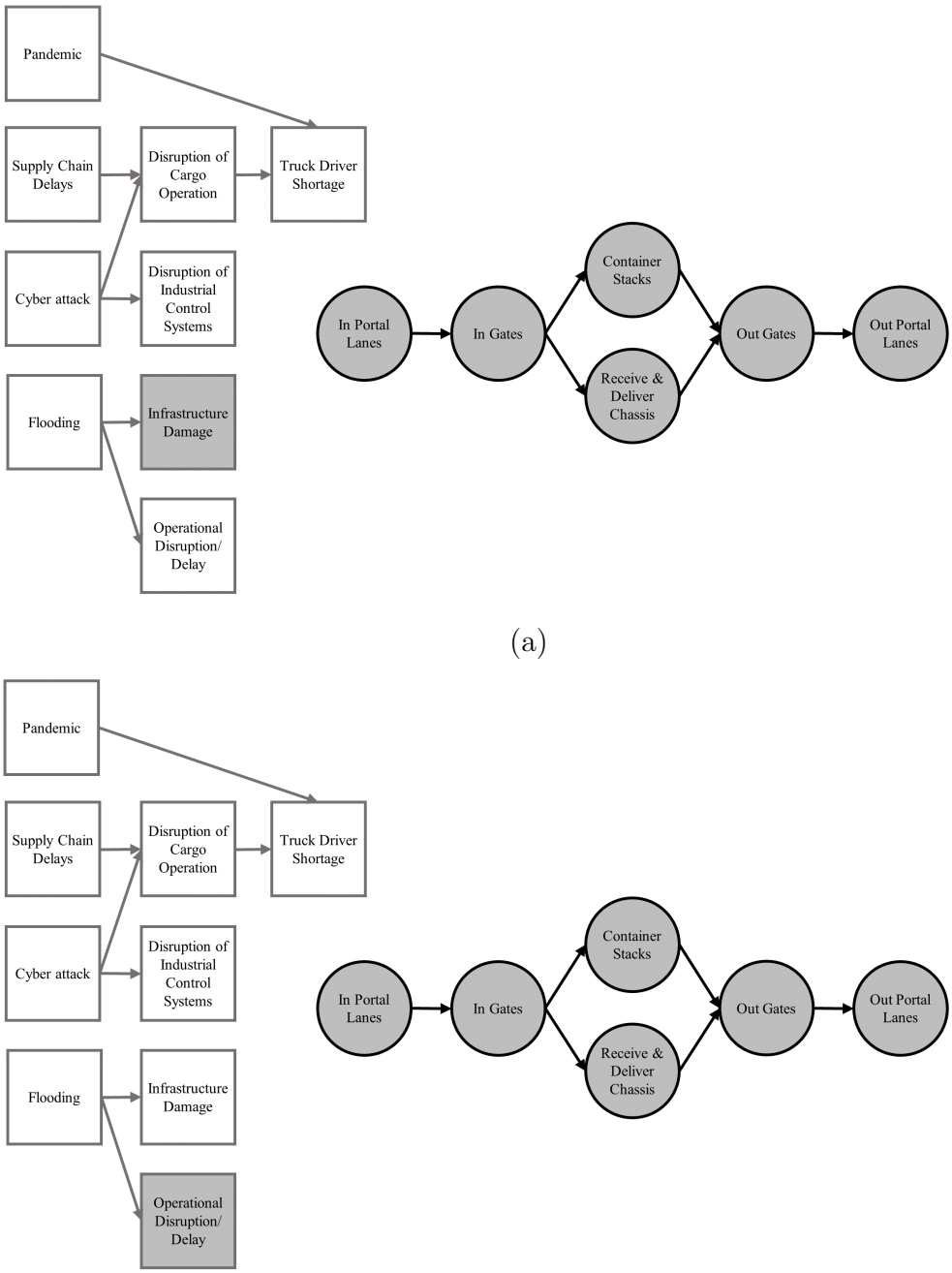


Fig. 59. Combined risk (grey square nodes) and system (black circular nodes) networks for coastal flooding disruptions in the port example for (a) infrastructure damage and (b) operational disruption/delay. Shading is used to show the dependency between nodes in the risk and system networks.

for truck visits to the port and container movements. Additionally, aggregated historical metrics have been provided. Some sets are provided for the past 7 working days while others are provided for the past 30 working days. Due to data privacy restrictions, all data and results presented will use simulated data that is designed to be similar to the actual data without revealing specifics.

5.2.1 Initial Data Analysis

The overall data set has thousands of entries for more than fifty variables. All analysis used Python 3 [114]. The first step in analysis was to eliminate variables that contain different representations of the same information. Examples of variables that were combined include driver identification number and driver name. Next, all time strings needed to be converted to date time objects. Finally, the time stamps indicating the start and finish of each process were subtracted in order to find the time spent at each node. Any observations with missing data were omitted from the analysis, and this left approximately 80% of the original samples for analysis.

5.2.2 Port Distributions

The next step is to learn distributions from the data sets so that they can be used in the model. For the variables dealing with time which are in-gate time, in-gate queue time, container transfer time, out-gate queue time, and out-gate time, various statistical distributions were fit to the data. The distributions chosen were exponential and gamma. The Probability Density Function (PDF) for each of these distributions can be found in many machine learning textbooks such as [61]. For the exponential distribution the PDF is

$$f(x) = \lambda e^{-\lambda x}, \tag{42}$$

where λ is the inverse of the mean of the distribution. λ is also known as the rate of the distribution. For the gamma distribution the PDF is

$$f(x) = \frac{1}{\Gamma(a)} b^a x^{a-1} e^{-bx}, \tag{43}$$

where a and b are the shape and rate parameters. The mean of the distribution is $\frac{a}{b}$.

For the data from each individual node, exponential and gamma functions were fit and a Kolmogorov-Smirnov test [115] was performed in accordance with the NIST Engineering Statistics Handbook [116] to determine whether or not it was reasonable to conclude that the data samples came from populations with underlying gamma or exponential distributions. In general, it was necessary to separate the nodes and test them individually to determine if the p-values were statistically significant. An aggregate summary of the fit and Kolmogorov-Smirnov test results from the real port data showing which p-values were significant is shown in Table 18. The results showed that most nodes could be assumed to come from a gamma distribution with the remaining two nodes, i.e., out portal and out gate nodes, coming from exponential distributions.

Since the actual port data sets cannot be published, the same types of underlying distributions with different parameters were used to generate a data set that could be used for this work and published. Gamma distributions were used for all nodes except the wait time distributions for the out gates and out portals which used exponential distributions. The parameters for all distributions are shown in Table 19. Histograms with 1,000 sampled

TABLE 18. Summarized overview of original data samples for Kolmogorov-Smirnov test results. When samples have p-values greater than the threshold value they are assumed to come from the stated distribution.

	Variable	Distribution	Node Count	Count of Nodes with p >0.1	Count of Nodes with p >0.01
Out Portal Lanes	Count	gamma	3	3 (100.00%)	3 (100.00%)
Out Gates	Count	gamma	11	10 (90.91%)	10 (90.91%)
Stacks	Count	gamma	19	16 (84.21%)	19 (100.00%)
Chassis	Count	gamma	2	2 (100.00%)	2 (100.00%)
In Gates	Count	gamma	16	14 (87.50%)	15 (93.75%)
In Portal Lanes	Count	gamma	3	3 (100.00%)	3 (100.00%)
Out Portal Lanes	WaitTime	exponential	3	3 (100.00%)	3 (100.00%)
Out Gates	WaitTime	exponential	11	10 (90.91%)	11 (100.00%)
Stacks	WaitTime	gamma	19	19 (100.00%)	19 (100.00%)
Chassis	WaitTime	gamma	2	2 (100.00%)	2 (100.00%)
In Gates	WaitTime	gamma	16	16 (100.00%)	16 (100.00%)
In Portal Lanes	WaitTime	gamma	3	3 (100.00%)	3 (100.00%)

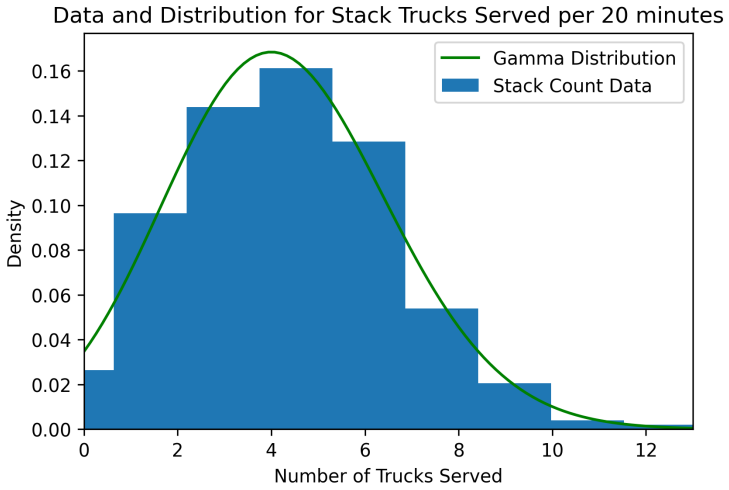


Fig. 60. Stack truck count histogram and fit distributions.

values and the probability density functions for the gamma and exponential distributions are shown in Figs. 60 and 61 for the stack count and out portal wait time variables. Based on the parameters some of the values generated by sampling from these distributions would be negative. To avoid this, all distributions were truncated for values less than zero.

5.2.3 System Network Training

Using the procedure from Chapter 3, three variations for the system network were fit to the two layer port system network. The first, single Recurrent Neural Network (RNN), was a single RNN. The second, agg-rNDA, was an aggregated implementation of the rNDA methodology with nodes aggregated by functional areas of the port resulting in 1 input node and 5 trained RNNs per layer resulting in 10 RNNs total (Fig. 54). The final variation, rNDA, was the full RNN-enabled Network Dependency Analysis (rNDA) model with 3 input nodes and 67 RNNs trained per layer resulting in 134 RNNs total (Fig. 55). Approximately

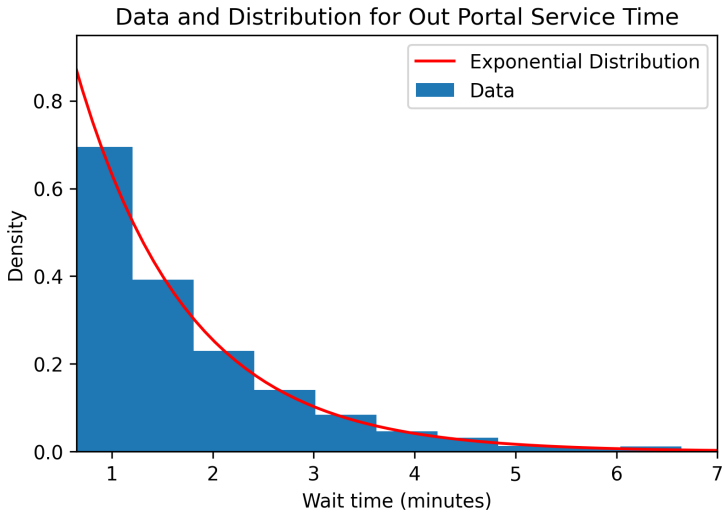


Fig. 61. Out portal wait and processing time histogram and fit distributions.

TABLE 19. Parameters for distributions.

Node type	Distribution	variable	loc	$\frac{1}{\lambda}$	a	$\frac{1}{b}$
Out Portal	Gamma	Count	7		7.9	3.3
Out Gate	Gamma	Count	0		13	0.7
Stack	Gamma	Count	-24		141	0.2
Receive Chassis	Gamma	Count	-90		90	1.2
Deliver Chassis	Gamma	Count	-85		144	0.7
In Gate	Gamma	Count	-10		21	10
In Portal	Gamma	Count	-40		19	3.9
Out Portal	Expon	WaitTime	0.6	1.1		
Out Gate	Expon	WaitTime	0.3	1.8		
Stack	Gamma	WaitTime	-48		30	2.5
Receive Chassis	Gamma	WaitTime	-90		280	0.43
Deliver Chassis	Gamma	WaitTime	-20		200	0.2
In Gate	Gamma	WaitTime	0.5		3	0.8
In Portal	Gamma	WaitTime	0		2	0.6

one month of training data was used to train the models. It was split into 90% for training and 10% for testing. The model architecture had a single hidden layer with 80 hidden units since this was previously found to produce good results. The loss metric used was mean squared error. Drop out was set to be zero and the look back was set equal to 1 or 2 time steps. Whether looking at the aggregated or non-aggregated models, performance results were split between look back values of 1 and 2, so results for both will be presented. For comparison, the overall mean squared error for the single RNN with look back of 1 was 0.026884 and 0.025366 with look back of 2. Plots for the mean square error for the aggregated and full RNN-enable network dependency analysis methodology are shown in Figs. 62 and 63. The mean square error for all non-aggregated nodes has been averaged across each functional area of the port so that the models can be compared point-wise as shown in Table 20. Training for the single RNN model took 45 seconds. Training for the model aggregated by functional area took 7 minutes. Finally, training for the full rNDA model took 104 minutes.

In most cases, the error is improved for the non-aggregated models. However, the error is close overall and the aggregated models are more computationally efficient. This indicates that the RNN-enabled network dependency analysis methodology has been shown to be a system of systems approach where trade-offs can be considered between model complexity and accuracy under normal operation. Later in this chapter (Section 5.4), both models will be used to detect disruption though they will produce different results under disruptions and will require further validation using data representative of disrupted scenarios to predict disruption magnitude which can be further tuned as additional data is

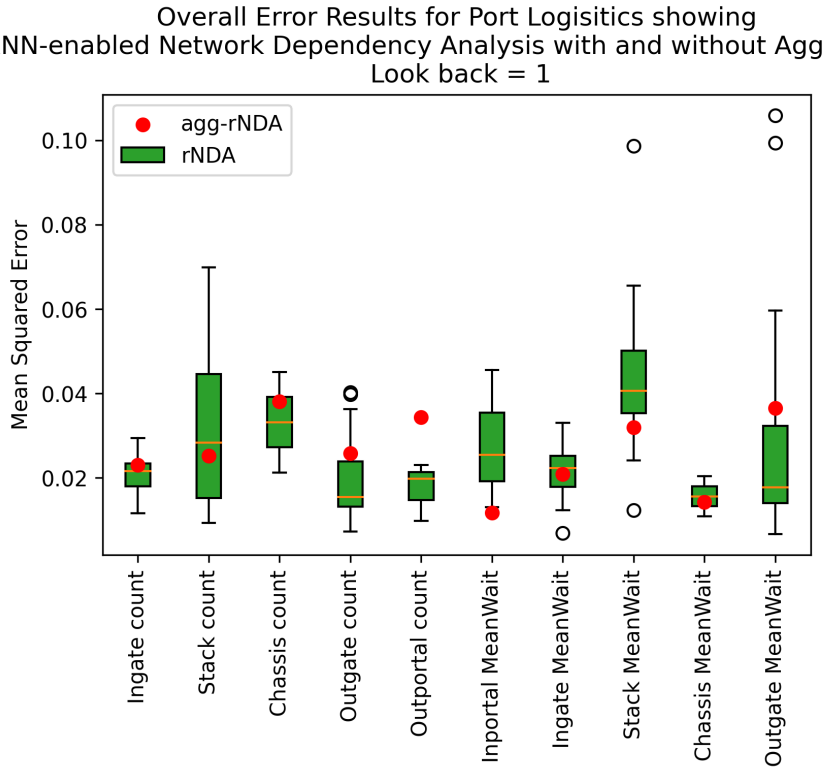


Fig. 62. Plots for aggregated and non-aggregated mean squared error for RNN-enabled network dependency analysis using a look back of one time step. For comparison, the single RNN model MSE was 0.026884.

collected.

5.3 Hypervulnerability Detection

Increases in system complexity and focus on leaner system design and operation have been seen across an array of fields. This has resulted in smaller margins of error and a push to make investments in systems where they stand to have a large, positive impact on system operation. With this in mind, system designers are not just looking to shore up resilience for any vulnerable system entity but also to invest funds in increasing resilience for the

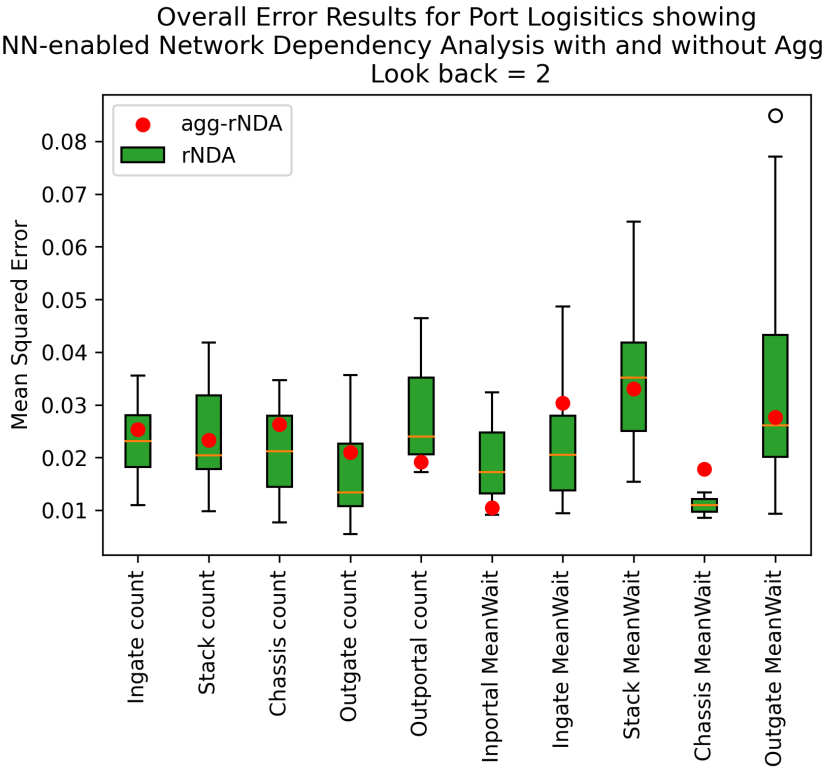


Fig. 63. Plots for aggregated and non-aggregated mean squared error for RNN-enabled network dependency analysis using a look back of two time steps. For comparison, the single RNN model MSE was 0.025366.

TABLE 20. Mean squared error for RNN-enabled network dependency analysis.

Look back		1	2	1	2
Node type	Variable	Aggregated		Non-Aggregated	
Out Portal	Count	0.034402	**0.019202**	**0.017507**	0.029184
Out Gate	Count	0.025807	**0.021013**	0.019726	**0.017020**
Stack	Count	0.025239	**0.023330**	0.031809	**0.024138**
Chassis	Count	0.038103	**0.026323**	0.033174	**0.021185**
In Gate	Count	**0.023058**	0.025286	**0.020815**	0.022724
Out Gate	WaitTime	0.036530	**0.027612**	**0.030991**	0.033984
Stack	WaitTime	**0.031988**	0.033049	0.043470	**0.035636**
Chassis	WaitTime	**0.014235**	0.017806	0.015605	**0.010939**
In Gate	WaitTime	**0.020904**	0.030343	**0.021357**	0.022986
In Portal	WaitTime	0.011738	**0.010499**	0.027988	**0.019585**
Average	Count	0.029322	**0.023031**	0.024606	**0.02285**
Average	WaitTime	**0.023079**	0.023862	0.027882	**0.024626**

most vulnerable entities. It is prudent to apply some level of definition and scale so that a determination can be made on whether one entity is more vulnerable than another. In this work, we propose a definition for an exceedingly vulnerable system entity and refer to this vulnerability as a hypervulnerability.

5.3.1 Hypervulnerability Initial Theory

The definition of hypervulnerability is developed based on the literature on characteristics of an entity that increase its vulnerability. We define a system hypervulnerability as a vulnerability that demonstrates X or more of the following criteria:

1. Prevents the system from meeting at least one of its critical operational requirements.

2. Adversely affects more that Y% (or Z) system entities.

3. Effects are felt at more than A% of the initial (or maximum) impact of the disturbance for at least B months (years, day, etc.)

4. Occurs with a probability (or expectation) of at least once per month (year, day, etc.)

5. The expected impact is more than some threshold (e.g., 10% of annual gross revenue for the company).

6. Is a combination of two individual vulnerabilities that are likely to occur together and whose effects may not combine linearly.

It is best to exercise careful consideration and consult subject matter experts when setting limits for the above criteria. However, there are many cases in which the only option is to start with a best guess for these parameters. Section 5.4 will show results of choosing one such set of thresholds, and it is recommended that analysts work with domain experts on choosing appropriate parameters.

It is reasonable to assume that vulnerability has an implicit scale of measurement as it is common to say one entity or feature of a system is more vulnerable than another. With this basic assumption, seeking to identify hypervulnerabilities in a system is a task in ascertaining where the vulnerability attains a local maximum. Defining the vulnerability of node i as v_i, then the index of the node where a maximum vulnerability occurs can be written:

$$\underset{i}{argmax}\,(v_i)\,. \tag{44}$$

However, v_i is a function of characteristics of the potentially affected system nodes, the probability of occurrence of the risk (i.e., $P(R_i)$, and the magnitude of the effect at each effective node j if the risk occurs (i.e., $A_j(R_i)$) (an extension of theory from [1]). Let's assume we are discussing a risk at node i and the group of nodes affected by the occurrence of the risk event is a cluster around node i that we can define as V^i. Let us further define each individual node in V^i as v_j^i with a feature vector of characteristics assigned to these nodes as C_j^i. With these definitions in mind, we can write the following:

$$v_i = f\left(C_j^i, P(R_i), A_j(R_i)\right). \tag{45}$$

5.3.2 Hypervulnerability Thresholds

While Section 5.3.1 suggests the use of the *argmax* function to identify the *most* vulnerable node, in reality, it is better to identify a set of hypervulnerable nodes. This is because the most vulnerable node may be resistant to efforts aimed at decreasing its vulnerability. In general, since the resources allocated to decrease vulnerability (and therefore increase resilience) are limited, it is prudent to apply these resources to a set of nodes in a way that optimizes the decrease in vulnerability for the entire system rather than specifically targeting the most vulnerable node.

Fig. 64 shows a conceptual example of hypervulnerability thresholds. For clarity, the risk network and system network are shown separately with the dependency of nodes in the system network on particular risk nodes shown by superimposing the appropriate risk network node symbol on the system node. For the example in Fig. 64, the risk network shows an internal and external risk where the probability of occurrence of the internal risk

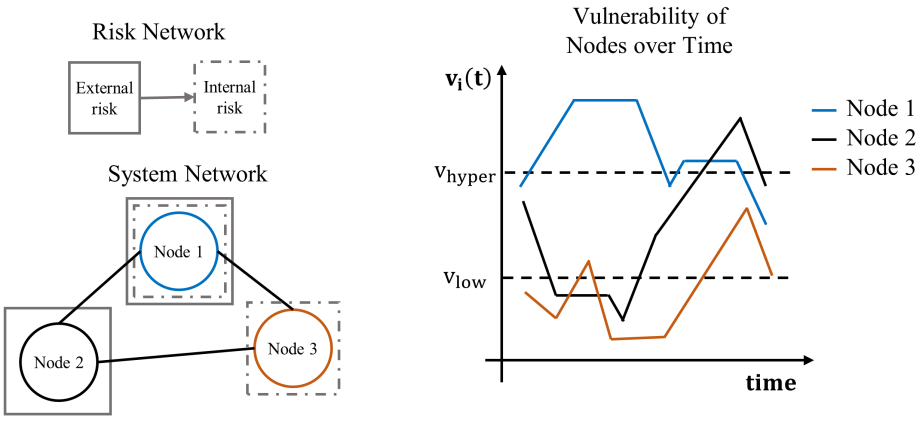


Fig. 64. Hypothetical risk and system networks with vulnerability of system nodes compared to threshold values.

is dependent on the external risk. The system network has three nodes. The direction of dependencies is not shown since it is implied that the overall system network model is representative of a set of layers each with its own dependencies. The network shows that Nodes 1 and 2 are dependent on the external risk and Nodes 1 and 3 are dependent on the internal risk.

Finally, the plot shows the hypothetical vulnerability of each node over time. By quantifying the vulnerability of each node, these values can be compared to threshold values that allow system analysts to be alerted when a node is either hypervulnerable or approaching hypervulnerability. In the plot shown, the value for Node 1 is above the threshold for hypervulnerability most of the time. This indicates that actions that increase the overall resilience of Node 1 may be best. Node 2 is above the hypervulnerability threshold for only a small window of time. It may be prudent to check and see if there were particular events occurring that time that impacted the vulnerability of Node 2 and explore actions

that decrease the impact of those events. Finally, Node 3 is below the low risk threshold for the majority of the time, though it has a small increase in vulnerability that coincides with the peak resulting in hypervulnerability for Node 2. This may indicate that Nodes 2 and 3 have a common vulnerability factor increasing their vulnerability or that the increase in vulnerability of either Node 2 or 3 causes an increase in the other. Investigating these possibilities can lead analysts to develop better plans to increase resilience and decrease risk.

5.4 Results and Discussion

In order to assess the overall vulnerability of the nodes at the port, the disrupted cases were investigated for both the aggregated and the non-aggregated RNN-enabled network dependency analysis models in Section 5.2. During the investigation, it became apparent that the aggregated model did not extend as a good predictive model of disrupted behavior as it tended to predict results that were above the operational capacity of the port. Therefore, the non-aggregated rNDA model will be used for the system network in the results that are presented here.

The four risk scenarios from Section 5.1.2 were simulated stochastically over 5 runs each. Results were assessed statistically to detect disruption, and the aggregated results will be discussed. In addition, the results were considered using a warning threshold at $\pm 5\%$ operability and a hypervulnerability threshold at $\pm 10\%$ operability for the number of trucks served. This is consistent with the definition of hypervulnerability in Section 5.3.1 where X is 1 and criteria 5 is used with a threshold of $\pm 10\%$ operability (or performance) as the expected impact on the number of trucks served. For service time, a warning threshold

at +10% operability and a hypervulnerability threshold at +20% operabilty was employed. These threshold values support the identification of nodes that require additional analysis due to being in a hypervulnerable state.

Except the disruptions applied during normal operation which had a probability of occurrence of 1%, all disruptions were applied with a likelihood of 100%. The reason the probability of occurrence was selected to be certain was that the risks were known to be occurring and therefore the nodes were expected to see at least some level of disruption. The average start time for disruptions was 20 time steps after the start of each run (each time step is 20 minutes, so 6.67 hours). The expected time in the absorptive, responsive, and recovery states was 5, 40, and 5 time steps, respectively. Each disruption was applied consistent with the methodology developed in Chapter 4, and the maximum magnitude of the disruption was drawn from a triangular distribution. For disruptions to trucks served nodes, the triangular distribution had a minimum of 45%, mode of 50%, and maximum of 55%. For service time nodes, the triangular distribution was modeled to have a minimum of 195%, mode of 200%, and maximum of 205%. These values were selected so that the internal health of the trucks served nodes and service time nodes were approximately halved and doubled, respectively.

5.4.1 Dynamic Response

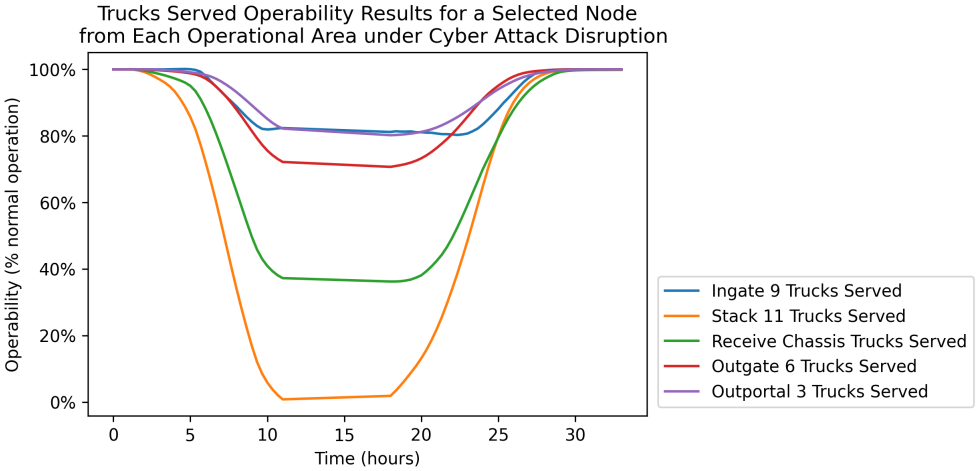
Before discussing the aggregated model results, examples of the behavior for individual nodes over time will be considered. A single node was selected from each operational area from each model layer. The operability of these nodes during each of the three disrupted scenarios, i.e., cyber attack, truck driver shortage, and coastal flooding, is shown in

Figs. 65, 66, and 67, respectively.

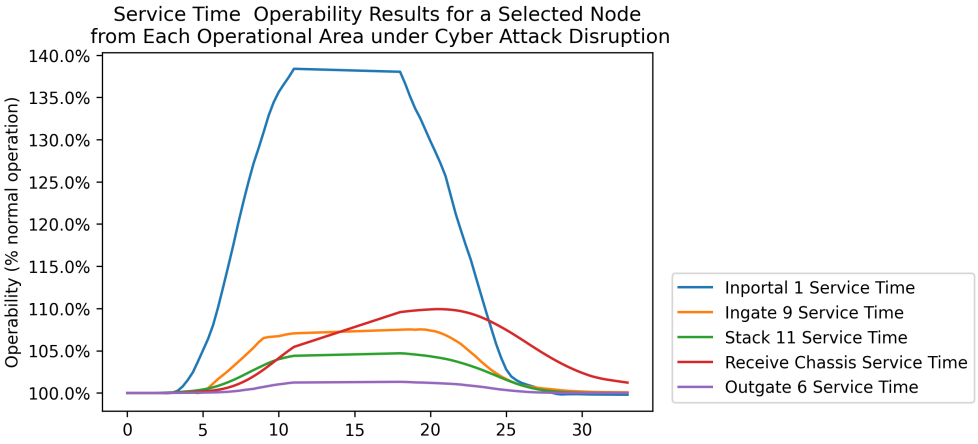
For the cyber attack disruption (Fig. 65), the in portal and in gate nodes were disrupted. The results, for example system nodes, show that these disruptions rippled through the port in the forward direction in the trucks serviced layer causing a maximum loss in operability to be exhibited by the stack node. The out gate and out portal nodes were also negatively affected, but the magnitude of disruption was smaller. In the service time layer, the disruptions caused increases in service times which were most severe at the in portal layer.

For the driver shortage disruption (Fig. 66), the in portal, in gate and stack nodes were disrupted. As more disruptions are interacting, the overall shape of the operability curves changes from the response seen previously for the cyber attack disruption. The stack node is still the most disrupted in the trucks served layer. In this case, it is also the most disrupted in the service time layer with the service time increasing to slightly more than double before returning to normal.

Finally, for the coastal flooding disruption (Fig. 67), only the stack nodes were disrupted. This only represents one functional area of the port; the stacks contain nearly double the number of nodes of any other area and are arguably the center of port operations. Still, the results are the simplest in this case with the stacks having the worst disruption. There is an interesting ripple effect response in the service time curve of in gate 9 which is clearly bimodal. This is because there was variation in the length of the stack disruptions with some lasting longer than the disruption shown at Stack 11.



(a)

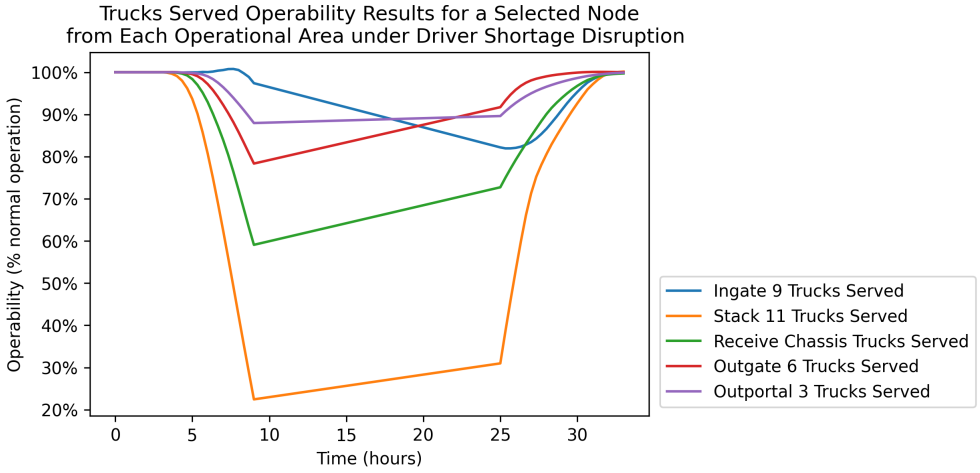


(b)

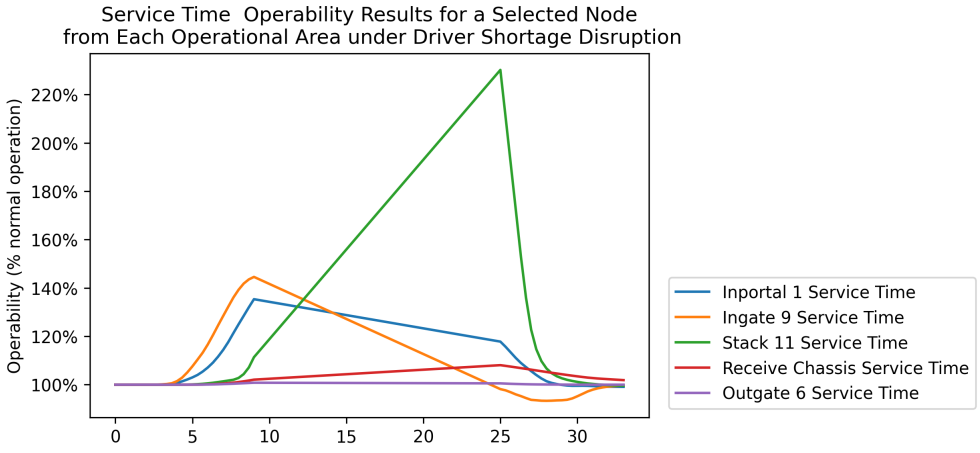
Fig. 65. Plots showing dynamic response for selected port nodes under cyber attack risk for (a) the number of trucks served layer and (b) the service time layer.

5.4.2 Normal Operation

Each node had a 1% chance of being semi-disrupted with a maximum disruption impact drawn from a triangular distribution with a minimum of 40%, mode of 50%, and maximum of 60% for both the trucks served and service time layers. This was implemented



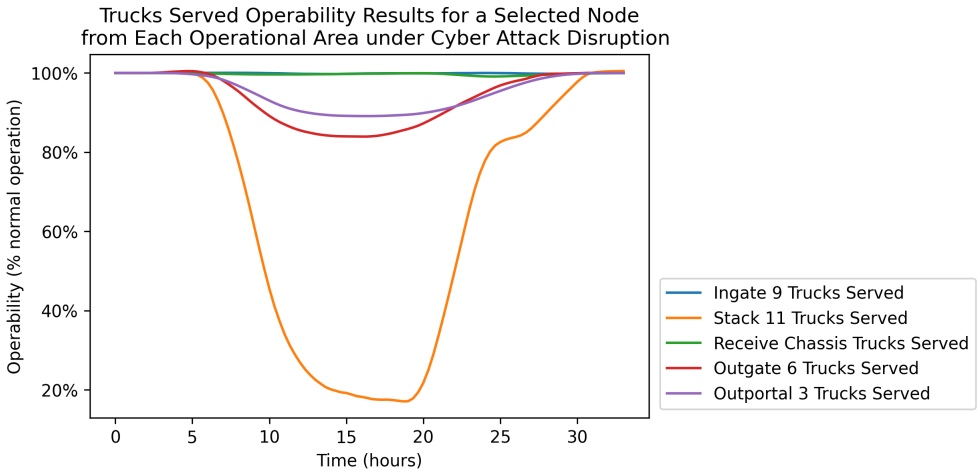
(a)



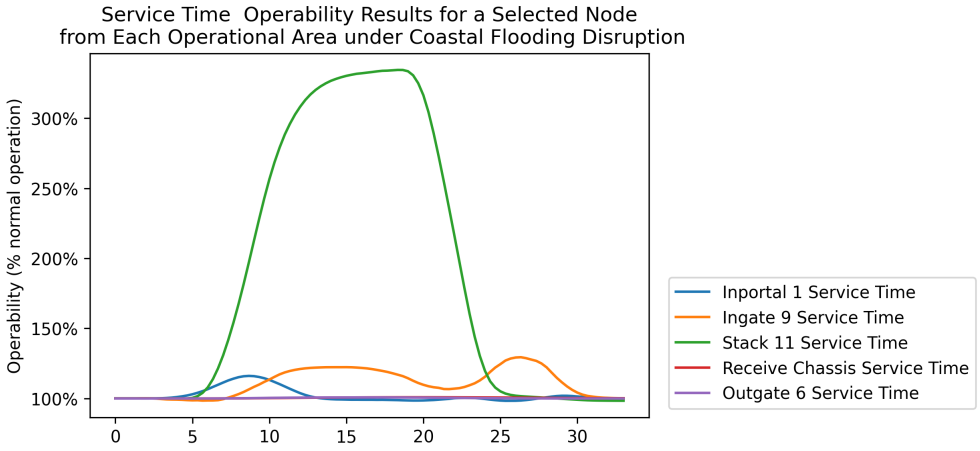
(b)

Fig. 66. Plots showing dynamic response for selected port nodes under driver shortage risk for (a) the number of trucks served layer and (b) the service time layer.

in the same manner as the normal operation case from Chapter 4 in Section 4.3.2. In particular, each of the 67 nodes in both layers, i.e., trucks serviced and service time, has a 1% probability of being disrupted in each of the 100 runs. If the node is disrupted, the disruption is applied to the internal health using a Markov chain model, and the node



(a)

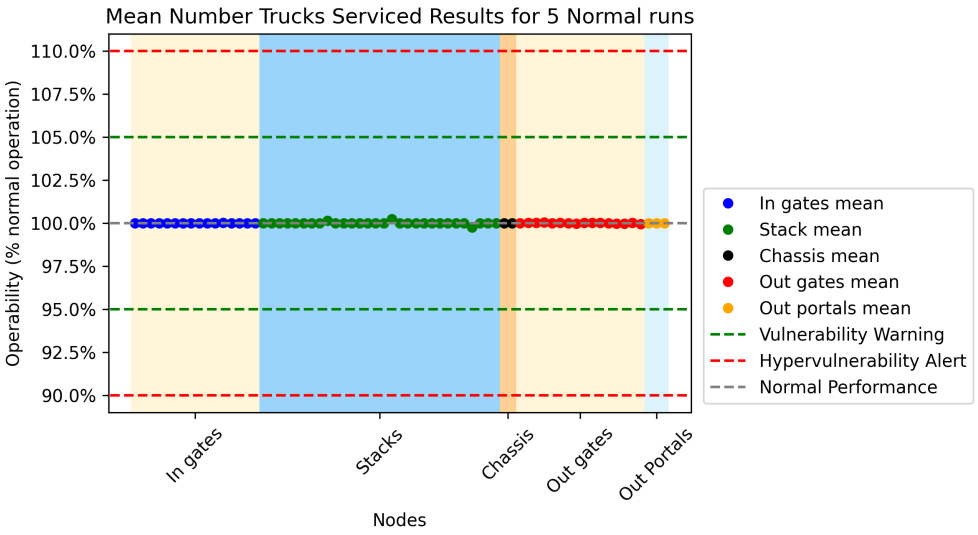


(b)

Fig. 67. Plots showing dynamic response for selected port nodes under coastal flooding risk for (a) the number of trucks served layer and (b) the service time layer.

moves through the stages of absorptive, responsive, and recovery stages of disruption before returning to normal.

Fig. 68 shows mean values for each node in the port network over 5 simulated runs. For both the number of trucks served and service time layers, all nodes lie in an



(a)

(b)

Fig. 68. Plots showing mean operability all nodes in port case study under normal operation for (a) the number of trucks served layer and (b) the service time layer.

approximately flat line at 100% operability. Additionally, Table 21 provides a numeric summary of the mean values of the port nodes averaged by operational area for all disruption

scenarios. For both the trucks served and service time layers, the values of operability for all nodes are very close to 100%.

TABLE 21. Mean operability results for all four disruption scenarios averaged by area of port operation in units of percent of normal operability.

Port Area	Layer	Normal	Cyber attack	Driver Shortage	Coastal Flooding
In gates	Trucks Served	100.0008	92.8194	98.1939	99.9888
Stacks	Trucks Served	100.0057	64.2254	88.7329	85.5335
Chassis	Trucks Served	100.0007	68.6200	91.5620	100.0319
Out gates	Trucks Served	100.0013	89.07793	96.3805	91.9920
Out portals	Trucks Served	100.0007	92.9137	97.6709	94.9719
In portals	Service time	99.9782	117.6823	108.4668	106.3250
In gates	Service time	99.9652	100.5440	105.2368	110.1603
Stacks	Service time	99.9546	101.1717	117.3168	191.8289
Chassis	Service time	100.0000	102.4045	101.2160	100.2334
Out gates	Service time	100.0009	99.2691	99.8054	99.5342

5.4.3 Cyber Attack

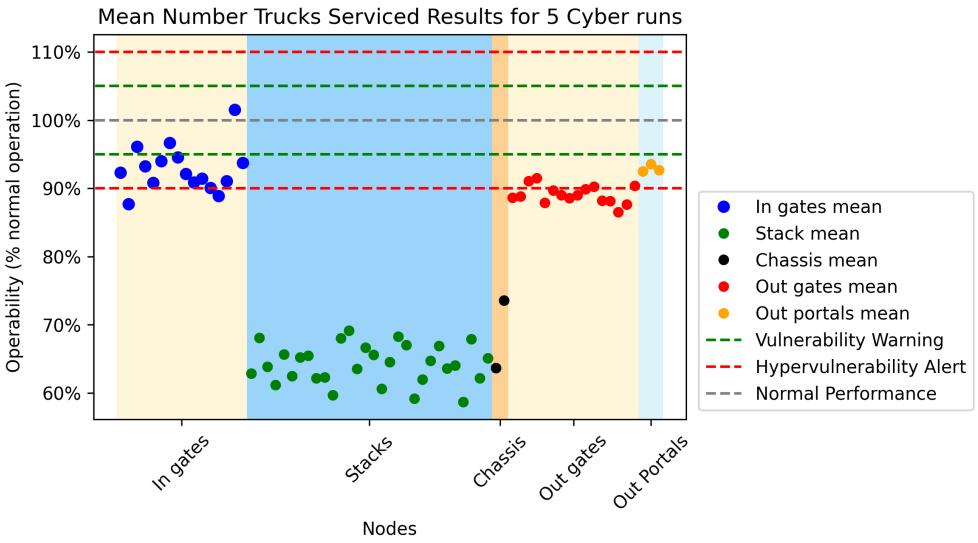
For the cyber attack disruption scenario, there is an active threat to the industrial control systems at the in portal and in gate nodes. The overall magnitude of the disruption, mean disruption start time, and disruption time stages were the same as they were above. The disruption was applied to each of the in portal and in gate nodes at the internal health variable using the same procedure developed in Chapter 4.

Fig. 69 shows the mean operability results for all nodes in both layers for the cyber attack over 5 runs. Comparing the trucks served layers there is a decrease in the number of trucks moving through all areas of the port. The stack and chassis nodes exhibit the

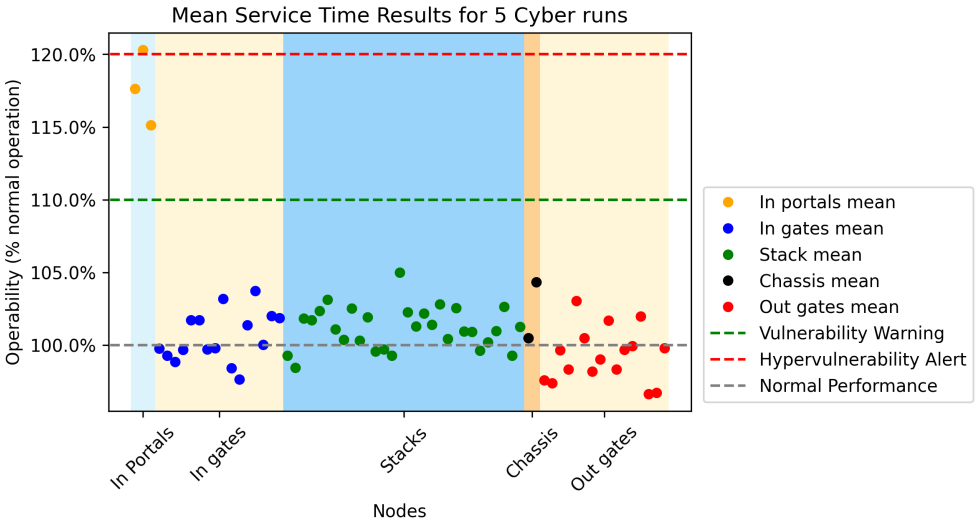
largest magnitude of decrease in performance. Considering the service time layer, there is a slight increase in the operability profiles for the stack, chassis, and in gate nodes and a larger increase at the in portal nodes.

In addition to comparing the plots visually and comparing the means, statistical analyses can be performed to determine if the disruptions can be detected statistically. The data were checked for normality using q-q plots and failed the test for normality. Therefore, a Kruskal-Wallis test was used to compare the disrupted results under the risk of cyber attack, with the normal operation results. The Kruskal-Wallis test indicates whether or not two samples come from the same population when they may not meet the test for normality [117]. The Kruskal-Wallis test was performed for all nodes in each model layer. Variables were aggregated using mean value per run prior to the performance of the statistical test.

Fig. 70 shows the results of the Kruskal-Wallis test comparing the runs for the cyber attack scenario against the runs for normal operation. Thresholds were plotted on the figure for three levels of significance with the most significance being 0.01 and the least significant being 0.1. The result shows that nearly all nodes are exhibiting behavior different from normal operation at the $p = 0.01$ confidence level. Combining these results with the results from above showing that the stack and chassis nodes performance had decreased below the hypervulnerability threshold for trucks served (Fig. 69(a)) indicated that there is a hypervulnerability caused by this disruption.



(a)



(b)

Fig. 69. Plots showing mean operability all nodes in port case study under disruption from a cyber attack for (a) the number of trucks served layer and (b) the service time layer.

5.4.4 Truck Driver Shortage

For the truck driver scenario, there is a shortage of truck drivers that is causally linked to an overall supply chain disruption. The same values were used to model the disruptions applied to the in portal, in gate, and stack nodes.

Fig. 71 shows the mean results for all nodes in both layers for the truck driver shortage scenario. In the trucks served layer, the mean values for the stacks are decreased. In addition, the variation in the behavior of the stacks increased meaning that the increase in number of disruptions has increased the uncertainty in the system. The service time for the stack nodes has also increased by nearly 20% on average. These results can be further investigated numerically in Table 21. Reviewing these results shows that the in portal, in gate, and chassis service times all seemed to increase significantly in addition to the results that were already stated. Finally, Fig. 71 shows the results of a Kruskal-Wallis test where under the driver shortage scenario nearly all nodes are reported to be statistically different from normal operation at the $p = 0.01$ level of significance.

5.4.5 Coastal Flooding

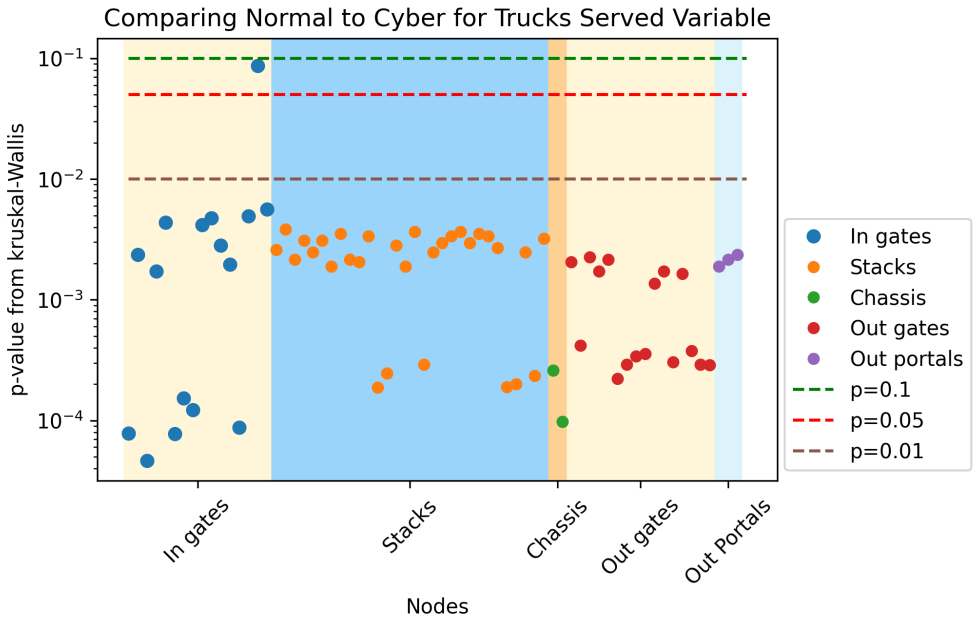
For the coastal flooding scenario, it was assumed that all stacks were disrupted during each run. The magnitude of disruption, mean disruption start time, and disruption time stages were the same as they were previously.

Fig. 73 shows the mean results for all nodes for the coastal flooding scenario. The stacks, out gates, and out portals all experience a decrease in the number of trucks served. The stacks experience a corresponding large increase in service time that is consistent with the overall slowdown experienced due to the flooding. Smaller slowdowns are experienced at

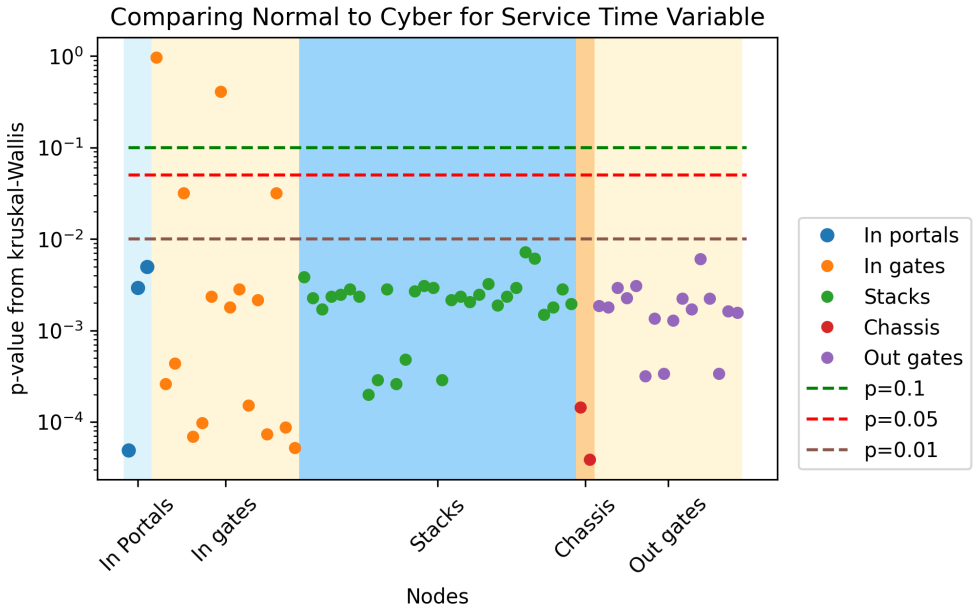
the in portals and in gates. Again, a Kruskal-Wallis test was performed (Fig. 74) and, like the driver shortage scenario, the majority of the nodes show behavior that is statistically different from the normal scenario at the $p = 0.01$ level.

5.5 Summary

This chapter presented detailed analysis of a maritime port using the ARNDA methodology. This included theoretical background for the definition of hypervulnerability along with a conceptual visualization. The port under study along with diagrams detailing the topology of system and risk networks with two functional layers were illustrated and modeled. Preliminary data analysis was conducted to identify underlying distributions from a real data set, and results were presented. Further, vulnerability and its relationship to risk and resilience were carefully studied and explained. Dynamic responses to disruptions were presented and discussed for various risk scenarios. Hypervulnerability thresholds were selected, and results were generated and plotted along with stochastic results showing which nodes were hypervulnerable using the full ARNDA methodology including results from RNN-enabled network dependency analysis.

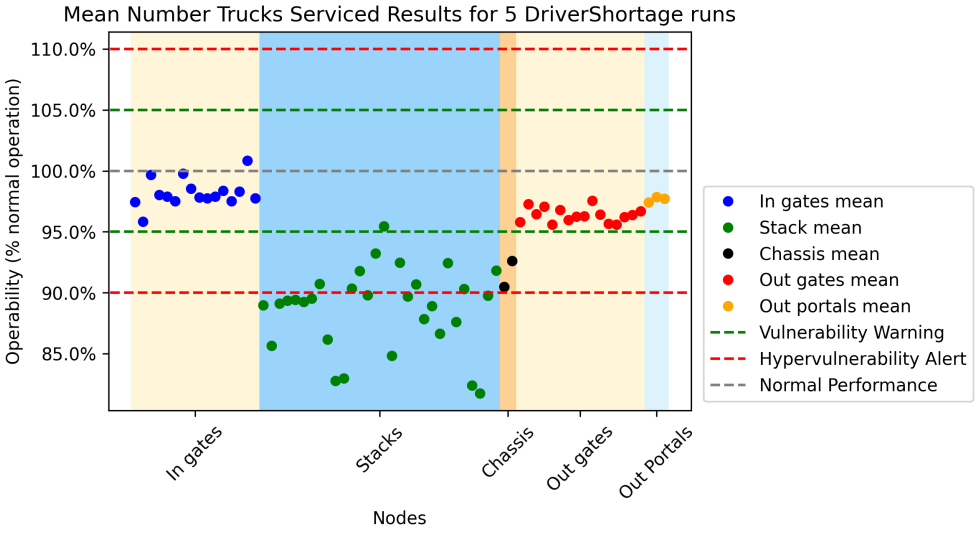


(a)

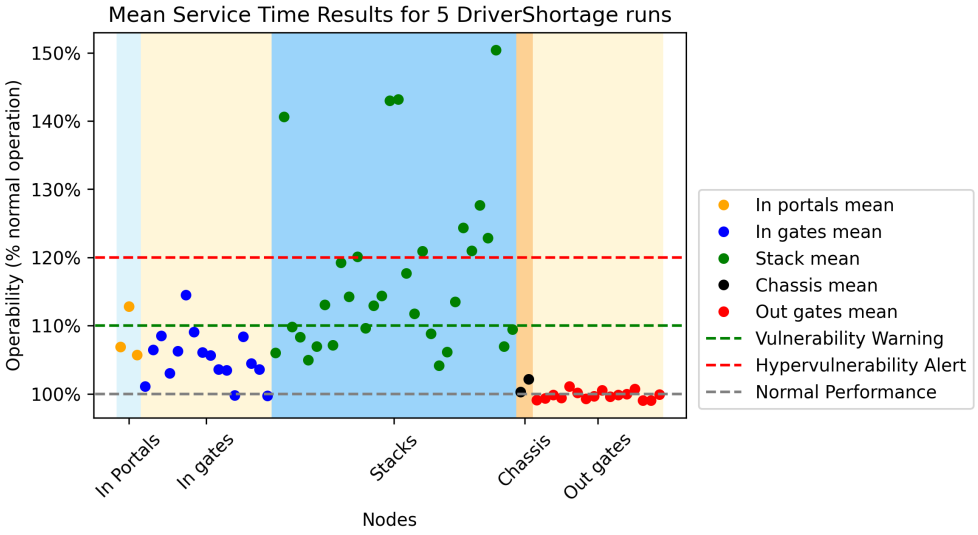


(b)

Fig. 70. Plots for Kruskal-Wallis test p-values comparing cyber attack results to normal results for (a) the number of trucks served layer and (b) the service time layer.

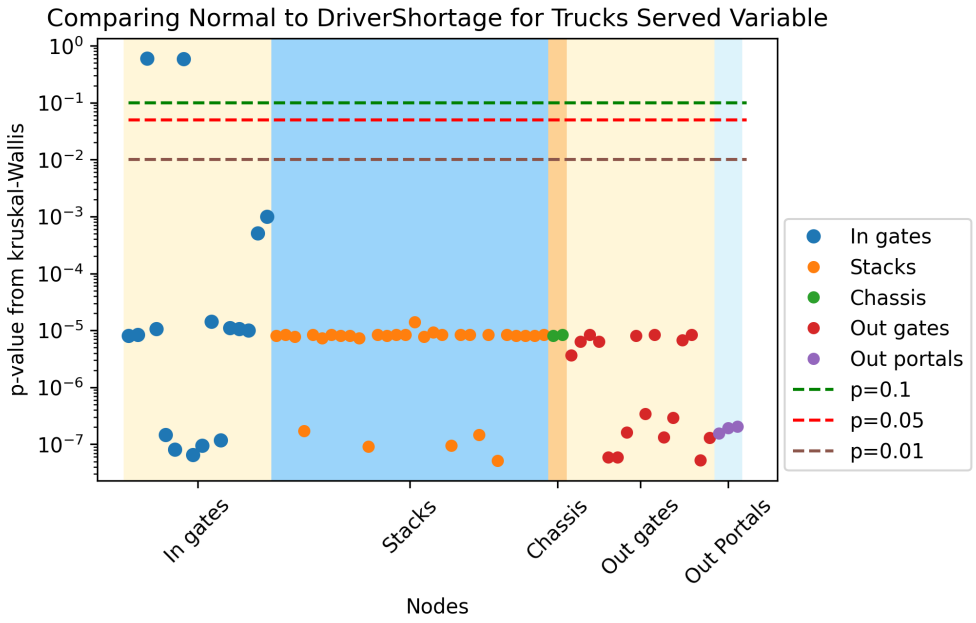


(a)

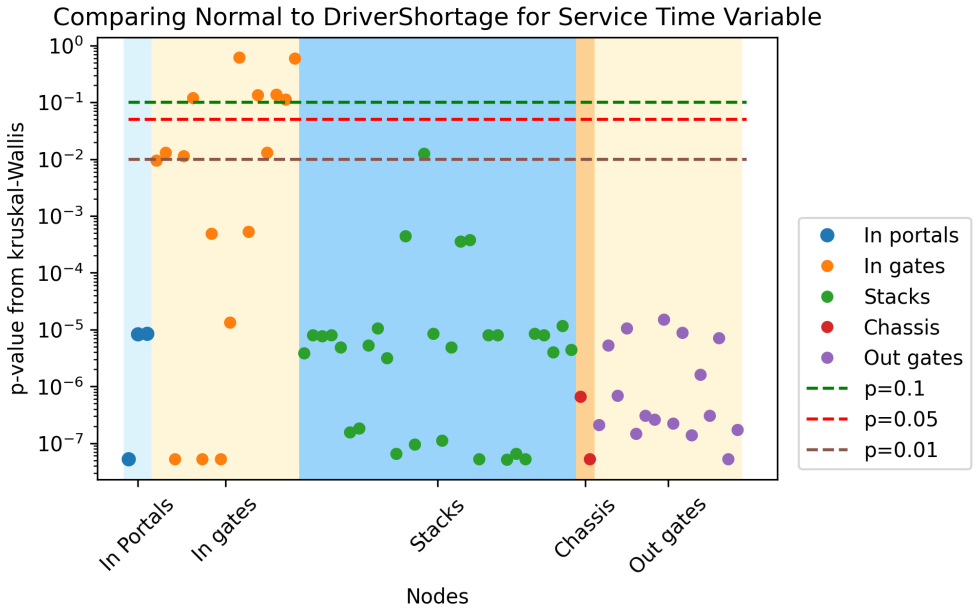


(b)

Fig. 71. Plots showing mean operability all nodes in port case study under under supply chain disruption and subsequent truck driver shortage for (a) the number of trucks served layer and (b) the service time layer.

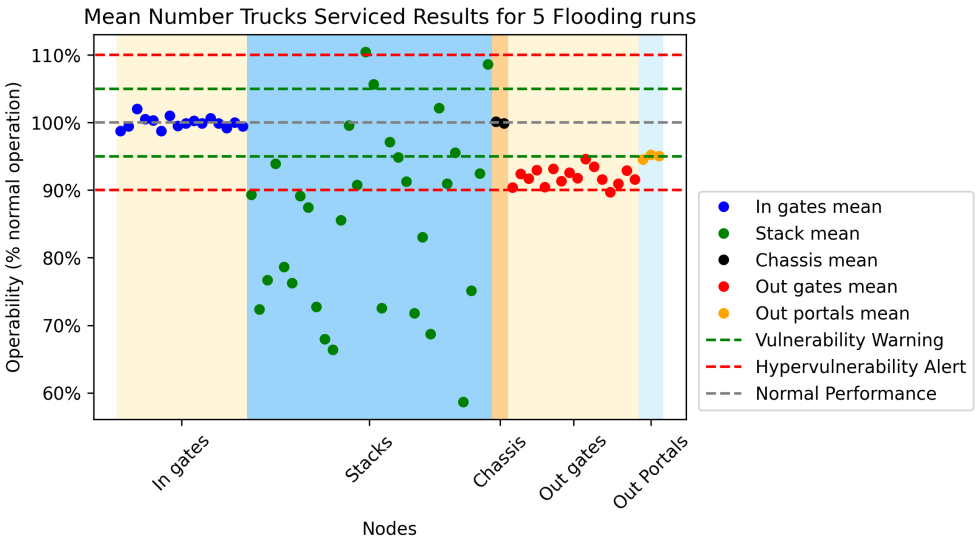


(a)

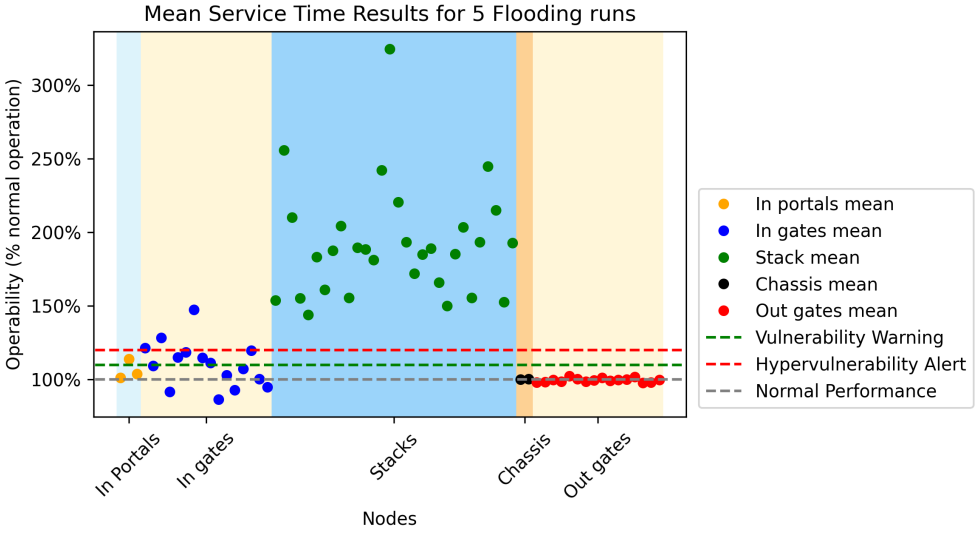


(b)

Fig. 72. Plots for Kruskal-Wallis test p-values comparing driver shortage results to normal results for (a) the number of trucks served layer and (b) the service time layer.

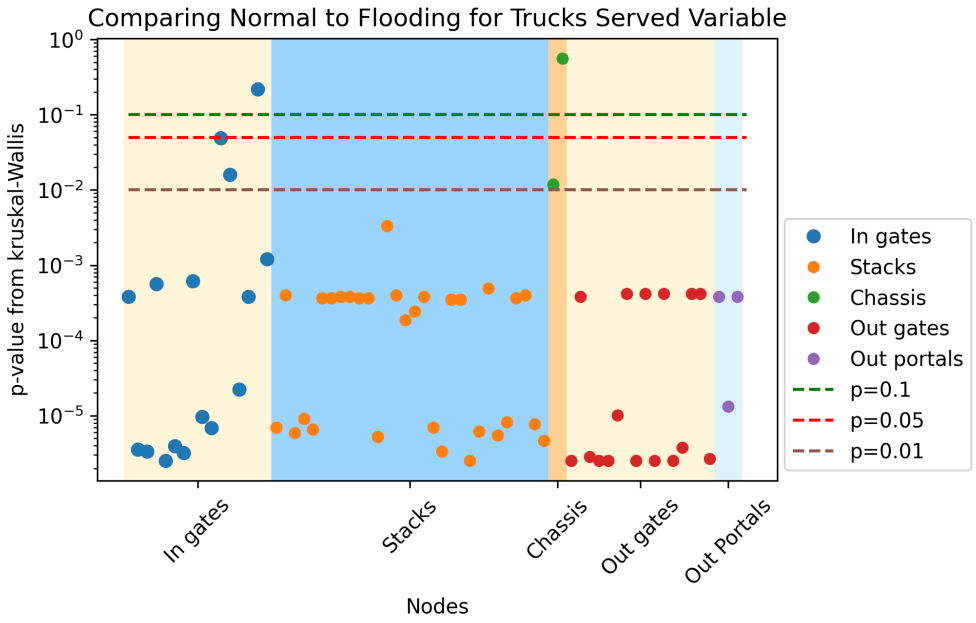


(a)

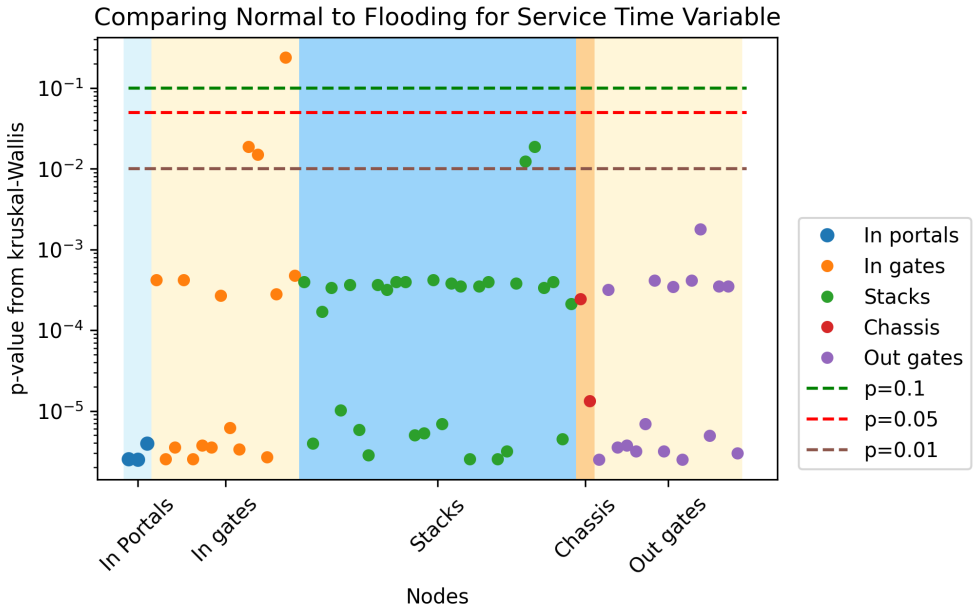


(b)

Fig. 73. Plots showing mean operability all nodes in port case study under under coastal flooding for (a) the number of trucks served layer and (b) the service time layer.



(a)



(b)

Fig. 74. Plots for Kruskal-Wallis test p-values comparing coastal flooding results to normal results for (a) the number of trucks served layer and (b) the service time layer.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

This dissertation presented a novel modeling and simulation methodology to analyze risk, resilience, and vulnerability of complex systems by combining a dynamic Bayesian network approach and Recurrent Neural Network (RNN)-enabled network dependency algorithms on directed graphs. Applications were provided in supply chain management and port logistics to demonstrate and validate the methodology.

This work makes several important contributions. First, it allows for the incorporation of cyclic features into a generalized analysis methodology for the analysis of system risk and resilience. These cyclic features were enabled by incorporating a multi-layered network representation of the system and with communication between layers that have edges traveling in opposite directions. It was demonstrated that ripple effects initiated by disruptions traveled through the graphical networks in the forward and backward directions.

In addition, temporal-dependence was incorporated into the methodology by fitting RNNs as the transfer functions in network dependency analysis. This allowed the system to capture dynamic response across multiple layers while learning system information from time-varying, dynamic data sets for supply chain and port logistics networks. The overall result was a reconfigurable network of RNNs that successfully learns the underlying dynamic behavior from the time series data.

Further, continuous system variables were used to capture the system behavior and validation was performed against a discrete event simulation for a supply chain network

under normal and disrupted conditions. The results show that Adaptive Risk Network Dependency Analysis (ARNDA) achieves similar accuracy to a discrete event simulation while needing to know much less about the internal workings of each entity in the complex network. This benefit comes from applying a more data-driven approach rather than a more knowledge driven approach.

Finally, approaches for using generalized metrics to support thresholds used to quantify performance with respect to risk, resilience, and vulnerability were presented and discussed. The overall RNN-enabled network dependency analysis approach is trained as a set of RNNs instead of a single RNN to allow for model reconfiguration with different areas able to be modeled at varying levels of detail or replaced altogether with a set of RNNs having a different structure.

An important avenue of ongoing work includes continuing to obtain and incorporate additional data sets into the training data for the existing methodology. In Chapter 4, ARNDA was trained on a data set that included examples that were typical of normal and disrupted operation. However, in the port case study in Chapter 5, disrupted data was not available, so the model was only trained on normal operation data. While the modeling approach was validated for a smaller model on disrupted data and then extended, it is important to validate on disrupted data in the future when it becomes available. Options for doing so include using real system data, data from a digital twin, or data from a cyber-physical scale model. Each of these is under consideration as future work in order to continue this work and further validate the modeling approach.

There are many fruitful directions available for future work. Building on the contributions from Chapter 3, different implementations and architectures for RNNs can be considered in order to further optimize the performance of the methodology. In particular, bidirectional Long-Short Term Memorys (LSTMs) [118] and adding a convolutional layer [119] have shown particular promise. Another direction for improvement would be to preprocess windows of time series data to classify it as disrupted or non-disrupted before using RNN models specifically trained on data exhibiting those behaviors.

Extending the work from Chapter 4, it would be valuable to have additional visualization methods to convey the results. The visualization methods presented are informative, but as the number of nodes becomes large, the amount of data becomes overwhelming very quickly. Developing and testing interactive dashboards for supply chain and port logistics managers to be able to view and respond to information in close to real time would be one potential way to improve static time history visualizations.

Finally, there are many interesting and relevant extensions of the work in Chapter 5. The first item of note is that the data-driven nature of the model implies that it can only be validated on cases where training data is available. As time passes and more disrupted training data becomes available, it is important to continue online training of the overall model to increase its accuracy over time. Other extensions include combining the ARNDA methodology with other methodologies that are used to model disruptions. Given that autonomous trucks will soon be flowing through ports alongside human-driven ones, integrating disruptions from models for autonomous vehicle safety and characterizing risks and disruptions related to these specific issues would increase the overall fidelity of the

ARNDA model.

REFERENCES

[1] S. M. Wagner and N. Neshat, "Assessing the vulnerability of supply chains using graph theory," *International Journal of Production Economics*, vol. 126, no. 1, pp. 121–129, 2010, ISSN: 0925-5273.

[2] C. Domonoske, "If world's battery supply doesn't scale up, automakers will be in trouble," *NPR*, 2021. [Online]. Available: https://www.npr.org/2021/04/08/985253463/if-worlds-battery-supply-doesnt-scale-up-automakers-will-be-in-trouble.

[3] D. Wu and T. Mochizuki, "Why shortages of a $1 chip sparked crisis in global economy," *Bloomberg Technology*, 2021. [Online]. Available: https://www.bloomberg.com/news/articles/2021-04-05/why-shortages-of-a-1-chip-sparked-crisis-in-the-global-economy.

[4] J. Kauffman, Electronic Article, 2021. [Online]. Available: https://www.npr.org/2021/04/09/985586760/betting-big-on-electric-vehicles-biden-faces-fraught-decision-on-ga-battery-plan.

[5] U. Nations, *Review of Maritime Transport*. New York, NY, 2020, ISBN: 978-92-1-112993-9. [Online]. Available: https://unctad.org/system/files/official-document/rmt2020_en.pdf.

[6] B. Klayman, *Reuters*, 2021. [Online]. Available: https://www.reuters.com/article/us-autos-semiconductors-supply-chain-ins-idUSKBN2BO4ZW.

[7] E. Hofmann, "Big data and supply chain decisions: The impact of volume, variety and velocity properties on the bullwhip effect," *International Journal of Production Research*, vol. 55, no. 17, pp. 5108–5126, 2017.

[8] K. Smith, R. Diaz, and Y. Shen, "Development of a framework to support informed shipbuilding based on supply chain disruptions," *Procedia Computer Science*, vol. 200, pp. 1093–1102, 2022.

[9] K. Smith, R. Diaz, Y. Shen, and F. Longo, "Conceptual development of a probabilistic graphical framework for assessing port resilience," *Proceedings of the 23rd International Conference of Harbor, Maritime, and Multimodal Logistic Modeling & Simulation*, 2021.

[10] K. Smith, R. Diaz, Y. Shen, and F. Longo, "Definition and detection of hypervulnerabilities using a framework for assessing port resilience," *Proceedings of the 24rd International Conference of Harbor, Maritime, and Multimodal Logistic Modeling & Simulation (under review)*, 2022.

[11] Encyclopedia, 2011. [Online]. Available: https://www.britannica.com/event/Japan-earthquake-and-tsunami-of-2011.

[12] D. Cyranoski, "Japan's tsunami warning system retreats," *Nature*, 2011. [Online]. Available: https://www.nature.com/news/2011/110811/full/news.2011.477.html.

[13] S. Shead, "The global chip shortage is starting to have major real-world consequences," *CNBC Tech*, 2021. [Online]. Available: https://www.cnbc.com/2021/05/07/chip-shortage-is-starting-to-have-major-real-world-consequences.html.

[14] K Katsaliaki, P Galetsi, and S Kumar, "Supply chain disruptions and resilience: A major review and future research agenda," *Annals of Operations Research*, pp. 1–38, 2021.

[15] J. Blackhurst, K. S. Dunn, and C. W. Craighead, "An empirically derived framework of global supply resiliency," *Journal of business logistics*, vol. 32, no. 4, pp. 374–391, 2011, ISSN: 0735-3766.

[16] M. J. Saenz, X. Koufteros, N.-O. Hohenstein, E. Feisel, E. Hartmann, and L. Giunipero, "Research on the phenomenon of supply chain resilience," *International Journal of Physical Distribution & Logistics Management*, 2015, ISSN: 0960-0035.

[17] S Hosseini, D Ivanov, and J. Blackhurst, "Conceptualization and measurement of supply chain resilience in an open-system context," *IEEE Transactions on Engineering Management*, 2020. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9248022/.

[18] A. Dolgui, D. Ivanov, and B. Sokolov, "Ripple effect in the supply chain: An analysis and recent literature," *International Journal of Production Research*, vol. 56, no. 1-2, pp. 414–430, 2018, ISSN: 0020-7543.

[19] S. Hosseini and D. Ivanov, "Bayesian networks for supply chain risk, resilience and ripple effect analysis: A literature review," *Expert systems with applications*, vol. 161, p. 113 649, 2020, ISSN: 0957-4174.

[20] S. Hosseini, A. Al Khaled, and M. Sarder, "A general framework for assessing system resilience using bayesian networks: A case study of sulfuric acid manufacturer," *Journal of Manufacturing Systems*, vol. 41, pp. 211–227, 2016, ISSN: 0278-6125.

[21] S. L. Lauritzen and D. J. Spiegelhalter, "Local computations with probabilities on graphical structures and their application to expert systems," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 50, no. 2, pp. 157–194, 1988, ISSN: 0035-9246.

[22] H. Guo and W. Hsu, "A survey of algorithms for real-time bayesian network inference," in *Join Workshop on Real Time Decision Support and Diagnosis Systems*, 2002.

[23] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2010.

[24] S. Hosseini, D. Ivanov, and A. Dolgui, "Ripple effect modelling of supplier disruption: Integrated markov chain and dynamic bayesian network approach," *International Journal of Production Research*, vol. 58, no. 11, pp. 3284–3303, 2020, ISSN: 0020-7543.

[25] Oxford English Dictionary, *Vulnerability, n.* https://www-oed-com.proxy.lib.odu.edu/view/Entry/224871, Accessed: 2021-06-13.

[26] B. Brown, *Courage and vulnerability part i: Definitions and myths*, https://brenebrown.com/wp-content/uploads/2019/08/Integration-Ideas_Courage-and-Vulnerability-Part-1-Definitions-and-Myths.pdf, Accessed: 2021-06-13.

[27] United Nations Office for Disaster Risk Reduction, *Vulnerability*, https://www.undrr.org/terminology/vulnerability, Accessed: 2021-06-13.

[28] R. Ross, M. McEvilley, and J. C. Oren, "System security engineering," National Institute of Standards and Technology, Tech. Rep., Accessed: 2021-06-13. [Online]. Available: https://www.undrr.org/terminology/vulnerability.

[29] D. Mustafa, S. Ahmed, E. Saroch, and H. Bell, "Pinning down vulnerability: From narratives to numbers," *Disasters*, vol. 35, no. 1, pp. 62–86, 2011.

[30] E. Tate, "Social vulnerability indices: A comparative assessment using uncertainty and sensitivity analysis," *Natural Hazards*, vol. 63, no. 2, pp. 325–347, 2012.

[31] C. G. Burton, "A validation of metrics for community resilience to natural hazards and disasters using the recovery from hurricane katrina as a case study," *Annals of the Association of American Geographers*, vol. 105, no. 1, pp. 67–86, 2015.

[32] Z. Charrahy, M. R. Yaghoobi-Ershadi, M. R. Shirzadi, *et al.*, "Climate change and its effect on the vulnerability to zoonotic cutaneous leishmaniasis in iran," *Transboundary and emerging diseases*, 2021.

[33] P. S. Pandit, M. M. Doyle, K. M. Smart, C. C. Young, G. W. Drape, and C. K. Johnson, "Predicting wildlife reservoirs and global vulnerability to zoonotic flaviviruses," *Nature communications*, vol. 9, no. 1, pp. 1–10, 2018.

[34] S. Platto, Y. Wang, J. Zhou, and E. Carafoli, "History of the covid-19 pandemic: Origin, explosion, worldwide spreading," *Biochemical and biophysical research communications*, vol. 538, pp. 14–23, 2021.

[35] R. C. Basole and M. A. Bellamy, "Supply network structure, visibility, and risk diffusion: A computational approach," *Decision Sciences*, vol. 45, no. 4, pp. 753–789, 2014.

[36] T. Bier, A. Lange, and C. H. Glock, "Methods for mitigating disruptions in complex supply chain structures: A systematic literature review," *International Journal of Production Research*, vol. 58, no. 6, pp. 1835–1856, 2020.

[37] Y. Li, K. Chen, S. Collignon, and D. Ivanov, "Ripple effect in the supply chain network: Forward and backward disruption propagation, network health and firm vulnerability," *European Journal of Operational Research*, vol. 291, no. 3, pp. 1117–1131, 2021.

[38] W. W. Leontief, "The structure of american economy, 1919-1939: An empirical application of equilibrium analysis," Report, 1951.

[39] I. Arto, V. Andreoni, and J. M. Rueda Cantuche, "Global impacts of the automotive supply chain disruption following the japanese earthquake of 2011," *Economic Systems Research*, vol. 27, no. 3, pp. 306–323, 2015, ISSN: 0953-5314.

[40] P. Jiang and Y. Y. Haimes, "Risk management for leontief-based interdependent systems," *Risk Analysis: An International Journal*, vol. 24, no. 5, pp. 1215–1229, 2004, ISSN: 0272-4332.

[41] P. R. Garvey and C. A. Pinto, "Introduction to functional dependency network analysis," in *Second International Symposium on Engineering Systems,*, vol. 5, 2009. [Online]. Available: https://www.researchgate.net/profile/Paul_Garvey4/publication/267690543_INTRODUCTION_TO_FUNCTIONAL_DEPENDENCY_NETWORK_ANALYSIS/

links/54b912400cf2c27adc4911cc/INTRODUCTION-TO-FUNCTIONAL-DEPENDENCY-NETWORK-ANALYSIS.pdf.

[42] C. Guariniello and D. DeLaurentis, "Communications, information, and cyber security in systems-of-systems: Assessing the impact of attacks through interdependency analysis," *Procedia Computer Science*, vol. 28, pp. 720–727, 2014, ISSN: 1877-0509.

[43] C. Guariniello and D. DeLaurentis, "Supporting design via the system operational dependency analysis methodology," *Research in Engineering Design*, vol. 28, pp. 53–69, 2017. DOI: 10.1007/s00163-016-0229-0.

[44] J. L. Hein, *Discrete structures, logic, and computability*. Jones and Bartlett Publishers, 2011, vol. 1.

[45] E. Çınlar, *Probability and stochastics*. Springer Science & Business Media, 2011, vol. 261.

[46] O. Knill, "Probability and stochastic processes with applications," *Havard Web-Based*, p. 5, 1994.

[47] C. Kooli and M. Lock Son, "Impact of covid-19 on mergers, acquisitions & corporate restructurings," *Businesses*, vol. 1, no. 2, pp. 102–114, 2021.

[48] D. Heckerman, D. Geiger, and D. M. Chickering, "Learning bayesian networks: The combination of knowledge and statistical data," *Machine learning*, vol. 20, no. 3, pp. 197–243, 1995.

[49] M. Kaushik, Electronic Article, 2021. [Online]. Available: https://www.marineinsight.com/marine-safety/watertight-doors-on-ships-a-general-overview/.

[50] Press Release, 2010. [Online]. Available: {https://www.dco.uscg.mil/Portals/9/DCO%20Documents/5p/CG-5PC/CG-CVC/CVC3/notice/alerts/Marine_Safety_Alert-Watertight_Doors.pdf}.

[51] U.S. Navy, *Naval ships' technical manual chapter 600 structural closures*, Government Document, 2005. [Online]. Available: https://www.hnsa.org/wp-content/uploads/2014/07/ch600-1.pdf.

[52] *Watertight Doors miscellaneous parts for individual and quick acting doors*, https://piersidesupply.com/watertight-doors/, Accessed: 2021-07-10.

[53] Rockwell Automation, 2020. [Online]. Available: https://www.arenasimulation.com/.

[54] S. H. Huan, S. K. Sheoran, and G. Wang, "A review and analysis of supply chain operations reference (scor) model," *Supply chain management: An international Journal*, 2004.

[55] H. Carvalho, A. P. Barroso, V. H. Machado, S. Azevedo, and V. Cruz-Machado, "Supply chain redesign for resilience using simulation," *Computers & Industrial Engineering*, vol. 62, no. 1, pp. 329–341, 2012.

[56] K. J. Arrow, T. Harris, and J. Marschak, "Optimal inventory policy," *Econometrica: Journal of the Econometric Society*, pp. 250–272, 1951.

[57] J. Lee Rodgers and W. A. Nicewander, "Thirteen ways to look at the correlation coefficient," *The American Statistician*, vol. 42, no. 1, pp. 59–66, 1988.

[58] J. E. Hutchinson and R. J. Loy, "Introduction to mathematical analysis," *School of Mathematical Sciences*, 1995.

[59] E. Framstad, S. Engen, and N. Chr, "Regression analysis, residual analysis and missing variables in regression models," *Oikos*, pp. 319–323, 1985.

[60] J. Olhager, "The role of the customer order decoupling point in production and supply chain management," *Computers in industry*, vol. 61, no. 9, pp. 863–868, 2010.

[61] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006, ISBN: 0387310738.

[62] J. T. Connor, R. D. Martin, and L. E. Atlas, "Recurrent neural networks and robust time series prediction," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 240–254, 1994.

[63] F. Rosenblatt, "Perceptron simulation experiments," *Proceedings of the IRE*, vol. 48, no. 3, pp. 301–309, 1960.

[64] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[65] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," *arXiv preprint arXiv:1506.00019*, 2015.

[66] R. Carbonneau, K. Laframboise, and R. Vahidov, "Application of machine learning techniques for supply chain demand forecasting," *European Journal of Operational Research*, vol. 184, no. 3, pp. 1140–1154, 2008.

[67] M. Pacella and G. Papadia, "Evaluation of deep learning with long short-term memory networks for time series forecasting in supply chain management," *Procedia CIRP*, vol. 99, pp. 604–609, 2021.

[68] G. H. Ribeiro, P. S. d. M. Neto, G. D. Cavalcanti, and R. Tsang, "Lag selection for time series forecasting using particle swarm optimization," in *The 2011 International Joint Conference on Neural Networks*, IEEE, 2011, pp. 2437–2444.

[69] H. C. Tijms, *Analysis of (s, S) inventory models*. Mathematisch Centrum Amsterdam, 1972.

[70] Y. W. Park, J. Blackhurst, C. Paul, and K. P. Scheibe, "An analysis of the ripple effect for disruptions occurring in circular flows of a supply chain network," *International Journal of Production Research*, pp. 1–19, 2021.

[71] F. Chollet *et al.*, *Keras*, https://keras.io, 2015.

[72] M. Abadi, A. Agarwal, P. Barham, *et al.*, *TensorFlow: Large-scale machine learning on heterogeneous systems*, Software available from tensorflow.org, 2015. [Online]. Available: https://www.tensorflow.org/.

[73] A. M. Law, W. D. Kelton, and W. D. Kelton, *Simulation modeling and analysis*. Mcgraw-hill New York, 2007, vol. 3.

[74] S. Siami-Namini, N. Tavakoli, and A. S. Namin, "A comparison of arima and lstm in forecasting time series," in *2018 17th IEEE international conference on machine learning and applications (ICMLA)*, IEEE, 2018, pp. 1394–1401.

[75] Z. He, J. Zhou, H.-N. Dai, and H. Wang, "Gold price forecast based on lstm-cnn model," in *2019 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big*

Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech), IEEE, 2019, pp. 1046–1053.

[76] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," *Advances in neural information processing systems*, vol. 29, 2016.

[77] Y. Duan, L. Yisheng, and F.-Y. Wang, "Travel time prediction with lstm neural network," in *2016 IEEE 19th international conference on intelligent transportation systems (ITSC)*, IEEE, 2016, pp. 1053–1058.

[78] J. Hu, Y. Lin, J. Tang, and J. Zhao, "A new wind power interval prediction approach based on reservoir computing and a quality-driven loss function," *Applied Soft Computing*, vol. 92, p. 106 327, 2020.

[79] C. Chatfield, *The analysis of time series: an introduction*. Chapman and hall/CRC, 2003.

[80] F. Altché and A. de La Fortelle, "An lstm network for highway trajectory prediction," in *2017 IEEE 20th international conference on intelligent transportation systems (ITSC)*, IEEE, 2017, pp. 353–359.

[81] D. J. Phillips, T. A. Wheeler, and M. J. Kochenderfer, "Generalizable intention prediction of human drivers at intersections," in *2017 IEEE intelligent vehicles symposium (IV)*, IEEE, 2017, pp. 1665–1670.

[82] S. Bozkurt Keser and Y. Buruk Sahin, "Response surface methodology to tune artificial neural network hyper-parameters," *Expert Systems*, vol. 38, no. 8, e12792, 2021.

[83] E. D. Vugrin, D. E. Warren, and M. A. Ehlen, "A resilience assessment framework for infrastructure and economic systems: Quantitative and qualitative resilience analysis of petrochemical supply chains to a hurricane," *Process Safety Progress*, vol. 30, no. 3, pp. 280–290, 2011.

[84] S. Chopra and M. Sodhi, "Supply-chain breakdown," *MIT Sloan management review*, vol. 46, no. 1, pp. 53–61, 2004.

[85] Y. Li and C. W. Zobel, "Exploring supply chain network resilience in the presence of the ripple effect," *International Journal of Production Economics*, vol. 228, p. 107 693, 2020.

[86] W. D. Kelton, *Simulation with ARENA*. McGraw-hill, 2002.

[87] N. Bugert and R. Lasch, "Supply chain disruption models: A critical review," *Logistics Research*, vol. 11, no. 5, pp. 1–35, 2018.

[88] T. G. Schmitt, S. Kumar, K. E. Stecke, F. W. Glover, and M. A. Ehlen, "Mitigating disruptions in a multi-echelon supply chain using adaptive ordering," *Omega*, vol. 68, pp. 185–198, 2017.

[89] M. C. Wilson, "The impact of transportation disruptions on supply chain performance," *Transportation Research Part E: Logistics and Transportation Review*, vol. 43, no. 4, pp. 295–320, 2007.

[90] C. Otto, S. N. Willner, L. Wenz, K. Frieler, and A. Levermann, "Modeling loss-propagation in the global supply network: The dynamic agent-based model acclimate," *Journal of Economic Dynamics and Control*, vol. 83, pp. 232–269, 2017.

[91] A Thangam and R Uthayakumar, "A two-level supply chain with partial backordering and approximated poisson demand," *European Journal of Operational Research*, vol. 187, no. 1, pp. 228–242, 2008.

[92] K. D. Cattani, F. R. Jacobs, and J. Schoenfelder, "Common inventory modeling assumptions that fall short: Arborescent networks, poisson demand, and single-echelon approximations," *Journal of Operations Management*, vol. 29, no. 5, pp. 488–499, 2011.

[93] M. J. Evans and J. S. Rosenthal, *Probability and statistics: The science of uncertainty*. Macmillan, 2004.

[94] T. Sawik, "Integrated selection of suppliers and scheduling of customer orders in the presence of supply chain disruption risks," *International Journal of Production Research*, vol. 51, no. 23-24, pp. 7006–7022, 2013.

[95] S. Chopra, P. Meindl, and D. V. Kalra, *Supply chain management: strategy, planning, and operation*. Pearson Boston, MA, 2013, vol. 232.

[96] T. Sawik, "A portfolio approach to supply chain disruption management," *International Journal of Production Research*, vol. 55, no. 7, pp. 1970–1991, 2017.

[97] J. Fang, L. Zhao, J. C. Fransoo, and T. Van Woensel, "Sourcing strategies in supply risk management: An approximate dynamic programming approach," *Computers & Operations Research*, vol. 40, no. 5, pp. 1371–1382, 2013.

[98] S. Hosseini and D. Ivanov, "A multi-layer bayesian network method for supply chain disruption modelling in the wake of the covid-19 pandemic," *International Journal of Production Research*, pp. 1–19, 2021.

[99] J. L. Hintze and R. D. Nelson, "Violin plots: A box plot-density trace synergism," *The American Statistician*, vol. 52, no. 2, pp. 181–184, 1998.

[100] A. Dolgui, D. Ivanov, and M. Rozhkov, "Does the ripple effect influence the bullwhip effect? an integrated analysis of structural and operational dynamics in the supply chain," *International Journal of Production Research*, vol. 58, no. 5, pp. 1285–1301, 2020.

[101] The Port of Virginia, "2065 master plan," Report, 2016. [Online]. Available: https://www.portofvirginia.com/wp-content/uploads/2016/02/TPOV-master-plan-2065-final-020316.pdf.

[102] Google, *Satellite image of vig portsmouth and surrounding area*, [Online]. [Online]. Available: https://www.google.com/maps/@36.8761386,-76.3541166,2325m/data=!3m1!1e3.

[103] United Nations, "Port performance indicators," Tech. Rep., 1976.

[104] R. Easley, N. Katsikides, K. Kucharek, D. Shamo, and J. Tiedman, "Freight performance measure primer," Tech. Rep., 2017.

[105] A. Varma, "Measurement sources for freight performance measures and indicators," Tech. Rep., 2008.

[106] Virginia Port Authority, *Virginia international gateway (vig)*, [Online; accessed 01-July-2022], 2022. [Online]. Available: {https://www.portofvirginia.com/facilities/virginia-international-gateway-vig/}.

[107] U.S. Department of Homeland Security, *Consequences to seaport operations from malicious cyber activity*, Press Release, 2016. [Online]. Available: {https://info.publicintelligence.net/DHS-SeaportCyberAttacks.pdf}.

[108] D. Zeiger, *Continuing port disruptions cause 'bullwhip' concerns*, [Online; accessed 02-July-2022], 2021. [Online]. Available: {https://www.ismworld.org/supply-management-news-and-reports/news-publications/inside-supply-management-magazine/blog/2021/2021-08/continuing-port-disruptions-cause-bullwhip-concerns/}.

[109] P. Goodman, *The real reason america doesn't have enough truck drivers*, Electronic Article, 2022. [Online]. Available: https://www.nytimes.com/2022/02/09/business/truck-driver-shortage.html.

[110] D. Romero, *Truckers, port workers vent as supply chain frustration mounts: 'a lot of us are willing to work'*, Electronic Article, 2022. [Online]. Available: https://news.yahoo.com/as-supply-crisis-worsens-truckers-and-dock-workers-all-say-dont-blame-us-125007621.html.

[111] J. Donnelly, *Major european ports hit by cyberattack*, Electronic Article, 2022. [Online]. Available: https://www.porttechnology.org/news/major-european-ports-hit-by-cyberattack/.

[112] O. Oshin, *Major us port target of attempted cyber attack*, Electronic Article, 2022. [Online]. Available: https://thehill.com/homenews/state-watch/573749-major-us-port-target-of-attempted-cyber-attack/.

[113] R. Asariotis, *Climate change impacts on seaports: A growing threat to sustainable trade and development*, Electronic Article, 2021. [Online]. Available: https://unctad.org/news/climate-change-impacts-seaports-growing-threat-sustainable-trade-and-development/.

[114] Python Software Foundation, *The Python Language Reference*. 2021. [Online]. Available: https://docs.python.org/3.8/reference/.

[115] I. Chakravati, R. Laha, and J. Roy, *Handbook of methods of applied statistics, volume i*, 1967.

[116] N. A. Heckert, J. J. Filliben, C. M. Croarkin, *et al.*, "Handbook 151: Nist/sematech e-handbook of statistical methods," 2002.

[117] R. E. Walpole, R. Myers, S. Myers, and K Ye, *Probability and Statistics for Engineers and Scientists, 9th*. Pearson, 2011.

[118] M. Khan, H. Wang, A. Riaz, A. Elfatyany, and S. Karim, "Bidirectional lstm-rnn-based hybrid deep learning frameworks for univariate time series classification," *The Journal of Supercomputing*, vol. 77, no. 7, pp. 7021–7045, 2021.

[119] R. Rick and L. Berton, "Energy forecasting model based on cnn-lstm-ae for many time series with unequal lengths," *Engineering Applications of Artificial Intelligence*, vol. 113, p. 104 998, 2022.

VITA

Katherine L. Smith

Department of Modeling and Simulation Engineering

Old Dominion University

Norfolk, VA 23529

Katherine L. Smith is a Research Associate in Digital Shipbuilding at Old Dominion University's Virginia Modeling, Analysis, and Simulation Center and a Ph.D. Candidate in the Computational Modeling and Simulation Engineering (CMSE) Department at Old Dominion University (ODU). She has previous experience as a senior lecturer in the Mathematics and Statistics Department at Old Dominion University and an aerospace engineer at NASA Langley Research Center. She graduated Summa Cum Laude from Old Dominion University as the top graduate from the Batten College of Engineering and Technology with B.S. degrees in Mechanical Engineering and Mathematics in December 2008. She earned an M.S. in Computational and Applied Mathematics and a Graduate Certificate in Modeling and Simulation from Old Dominion University in December 2009. Her research is focused in modeling, simulation, and visualization of complex systems, data analytics and visualization, predictive analytics, game-based learning, and promoting STEM accessibility across a spectrum of diverse groups. Ms. Smith is expected to graduate with her Ph.D. in August 2022.