Summer 2017

Survivability modeling for cyber-physical systems subject to data corruption

Mark James Woodard

Department: Electrical and Computer Engineering

SURVIVABILITY MODELING FOR CYBER-PHYSICAL SYSTEMS SUBJECT TO DATA CORRUPTION

by

MARK JAMES WOODARD

A DISSERTATION

Presented to the Graduate Faculty of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

in

COMPUTER ENGINEERING

2017

Approved by

Dr. Sahra Sedigh Sarvestani, Co-Advisor
Dr. Ali R. Hurson, Co-Advisor
Dr. Minsu Choi
Dr. Donald C. Wunsch II
Dr. Abhijit Gosavi

ABSTRACT

Cyber-physical critical infrastructures are created when traditional physical infrastructure is supplemented with advanced monitoring, control, computing, and communication capability. More intelligent decision support and improved efficacy, dependability, and security are expected. Quantitative models and evaluation methods are required for determining the extent to which a cyber-physical infrastructure improves on its physical predecessors. It is essential that these models reflect both cyber and physical aspects of operation and failure. In this dissertation, we propose quantitative models for dependability attributes, in particular, survivability, of cyber-physical systems. Any malfunction or security breach, whether cyber or physical, that causes the system operation to depart from specifications will affect these dependability attributes. Our focus is on data corruption, which compromises decision support — the fundamental role played by cyber infrastructure. The first research contribution of this work is a Petri net model for information exchange in cyber-physical systems, which facilitates i) evaluation of the extent of data corruption at a given time, and ii) illuminates the service degradation caused by propagation of corrupt data through the cyber infrastructure. In the second research contribution, we propose metrics and an evaluation method for survivability, which captures the extent of functionality retained by a system after a disruptive event. We illustrate the application of our methods through case studies on smart grids, intelligent water distribution networks, and intelligent transportation systems. Data, cyber infrastructure, and intelligent control are part and parcel of nearly every critical infrastructure that underpins daily life in developed countries. Our work provides means for quantifying and predicting the service degradation caused when cyber infrastructure fails to serve its intended purpose. It can also serve as the foundation for efforts to fortify critical systems and mitigate inevitable failures.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude for the academic guidance from my advisors Dr. Sahra Sedigh Sarvestani and Dr. Ali Hurson. They have both been of enormous help during my time as a Ph.D. student at the Missouri University of Science and Technology. Their guidance and support helped me to progress in my education and produce quality research.

I would also like to thank Dr. Donald Wunsch, Dr. Minsu Choi, and Dr. Abhijit Gosavi for serving on my Ph.D. committee. The knowledge and technical skills I gained from their courses have greatly contributed to my research and their insightful technical feedback helped to improve the quality of my work.

A special thanks goes to all of my colleagues and friends in the Sensor Networks and Dependable Computing (SeNDeComp) Research Group. Their company made the challenges of research an enjoyable experience.

The educational and research experience I have gained at Missouri S&T will help me to overcome obstacles in research and work in the near future. I will always cherish my experiences in Rolla.

Finally, I would like to extend my deepest appreciation to my parents Joan and James Woodard, my brother Thomas Woodard, and my dog Aspen. Their love and support have been unwavering.

The research presented in this dissertation was supported by the Department of Education through a Graduate Assistance in Areas of National Need (GAANN) fellowship. Financial support was also provided by the National Science Foundation, the US and Missouri Departments of Transportation, and the Missouri S&T Intelligent Systems Center and Center for Infrastructure Engineering Studies.

TABLE OF CONTENTS

Page

LIST OF ILLUSTRATIONS

LIST OF TABLES

NOMENCLATURE

Acronyms

CPSs Cyber Physical Systems

ITS Intelligent Transportation System

IWDS Intelligent Water Distribution System

MIS Markov Chain Imbeddable Structure

1. INTRODUCTION

Modern critical infrastructures are large networked cyber-physical systems (CPSs). CPSs improve efficacy, dependability, and other attributes by supplementing a traditional physical system with computation, networking, and control [1, 2]. In CPSs, sensors collect information about the operational state of the physical system and communicate this information in real-time to computers and embedded systems used for intelligent control. Examples of modern critical infrastructures CPSs include smart grids, intelligent water distribution networks, and intelligent transportation systems.

CPSs are deployed in unpredictable environments; as such, modeling and analysis of their behavior in response to disruptive events is a critical challenge. Modeling and evaluation are used to determine if a system design meets the specified levels of performance (as measured by functional attributes) or dependability (a non-functional attribute).

Traditional dependability attributes, such as reliability and availability, utilize a binary view of system functionality (i.e., operational or failed). However, the size and complexity of CPSs make it likely that component failures will place the system in a functionally degraded, but operational, state that is better captured by attributes such as survivability and resilience, each of which can capture degraded operation. One focus of the research presented in this dissertation is on modeling and evaluation of dependability attributes for CPSs. Specifically, we quantify survivability, which captures the ability of a CPS to deliver essential services despite being in a degraded operational state.

The intelligent control utilized by CPSs requires access to real-time and historical data from the vicinity of each control entity, as well as system-wide information, to calculate ideal control settings. The reliance on real-time field data makes CPS susceptible to consequences of data corruption — unintended changes to data that occur during writing, reading, storage, transmission, or processing. A functional control system that processes

corrupted data will produce incorrect control settings, defying the purpose of intelligent control and compromising decision support. Given the critical role of data, among the potential disruptions that can degrade the operation of a CPS, we focus on data corruption. The specific research problems addressed are the limited understanding of data dependencies and interdependencies among components of a system and the lack of quantitative models that can capture the propagation of corrupt data and its consequences for system operation. To our knowledge, as of the date of publication, no existing model directly evaluates systems based on the presence or propagation of corrupted data. Some models can account for corrupted data by modeling a component as "failed;" however, this does not account for the possibility of degraded or erroneous behaviors. Quantitative models and evaluation techniques for data corruption in CPSs are the subject of this work.

1.1. MOTIVATION

In most complex systems, data corruption is unavoidable. Erroneous data can be created through unintentional means, such as failures in sensors, processors, storage, or communication hardware, or intentionally, through a cyber or physical attack. Data corruption can have severe consequences for critical systems. Kirilenko et al. [3] describe a financial computing system failure that occurred in August of 2012. A software error ended up costing Knight Capital, a mid-size financial firm, $450 million, at a rate of $10 million per minute. In this case, a cyber malfunction led to corrupted output data — trading prices.

In CPSs, the tight coupling between the cyber and physical infrastructures increases the likelihood and exacerbates the consequences of data corruption. In addition to economic consequences, failures in critical infrastructure and manufacturing systems can result in the loss of life. One of these failures is presented by Miller et al. [4]. In June 1999, a Bellingham, WA gas pipeline ruptured and leaked 237,000 gallons of gasoline, which ignited and burned approximately 25 acres of forest, resulting in three deaths and eight documented injuries. The failure was exacerbated by the inability of the control systems to react, due to the

company's practice of performing database development work on the system while it was operating, which made real-time data unavailable. While this example is not the result of corrupted data, it demonstrates CPSs' reliance on accurate real-time data and the potential consequences of corrupted data - missing data caused a cyber malfunction.

Lastly, a very recent example of corrupted data causing a control failure is the ExoMars Schiaparelli Mars Lander crash on October 19, 2016 [5]. During the descent onto Mars, the lander's inertial measurement unit produced an erroneous measurement that persisted for about 1 sec. This erroneous data caused the navigation system to generate an estimated altitude that was below ground level. This, in turn, triggered the premature release of the parachute, as though the lander had already landed, and caused the lander to fall 3.7 km. While this example is not a critical infrastructure CPS, it does show the potential for severe consequences of erroneous data. In this case, corrupted data caused a physical malfunction. All of these examples demonstrate the dependence of CPSs on timely and accurate data.

1.2. RESEARCH CONTRIBUTIONS

This dissertation serves as the culmination of my research activities. The broad topical areas related to my research are CPS modeling, survivability evaluation, and data corruption. A taxonomy of related topics is depicted in Figure 1.1.

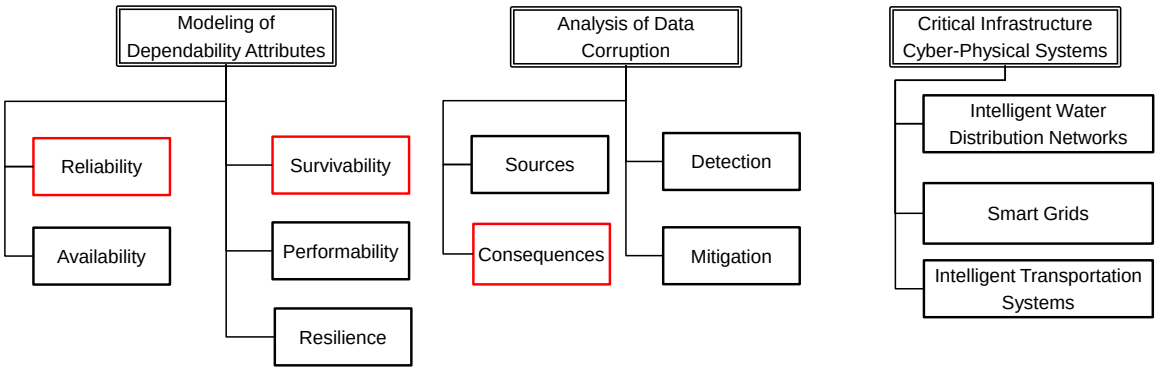


Figure 1.1. Taxonomy of research topics.

The original research contributions presented in this dissertation include:

1. Quantitative modeling of CPS reliability with Markov imbedded structures, where the system-level reliability of a CPS is evaluated as a function of the respective reliabilities of the control system and cyber infrastructure. The approach is demonstrated with a case study using an intelligent water distribution network.

2. Defining metrics and an evaluation method for CPS survivability, where a domain-specific figure-of-merit captures the extent and rate of degradation during a disruptive event. This approach is demonstrated with a case study on several smart grids.

3. Component prioritization approach for targeted hardening of CPSs, where components are ranked based on their fragility and criticality, which are calculated from the survivability metrics. This is demonstrated with a case study on several smart grids.

4. Survivability modeling for intelligent transportation systems, where manned, semi- and fully-autonomous vehicles operate in the same environment.

5. Qualitative and quantitative models for the propagation of corrupted data in CPSs, where the qualitative model elucidates the potential for propagation of corrupted data in the course of information exchange among control entities of a CPS. The quantitative model measures the state of data in the system as information is exchanged, providing a basis for examining the propagation of corrupted data. The quantitative model can be used in conjunction with the proposed survivability evaluation approach. This model and analysis are demonstrated with a case study using an IEEE 57-bus smart grid system.

The intellectual merit of my work is in a) advancing knowledge and enabling prediction of the consequences of data corruption, and b) creating quantitative representations of the effects of data corruption on system dependability. The broader impact of my work

is in enabling justifiable reliance on cyber-physical critical infrastructure, which can result in increased public safety and efficacy, as well as reduced cost.

The following publications have resulted from this research:

1. M. Woodard and S. Sedigh Sarvestani, "Modeling of autonomous vehicle operation in intelligent transportation systems," in *Proceedings of the 5th International Workshop on Software Engineering for Resilient Systems*, pp. 133–140, 2013.

2. K. Marashi, M. Woodard, S. Sedigh Sarvestani, and A. R. Hurson, "Quantitative reliability analysis for intelligent water distribution networks," in *Proceedings of the Embedded Topical Meeting on Risk Management for Complex Socio-Technical Systems, Annual Meeting of the American Nuclear Society*, November 2013.

3. M. Woodard, S. Sedigh Sarvestani, and A. R. Hurson, "A survey of research on data corruption in cyber-physical critical infrastructure systems," vol. 98 of *Advances in Computers*, pp. 59–87, Elsevier, 2015.

4. M. Woodard, M. Wisely, and S. Sedigh Sarvestani, "A survey of data cleansing techniques for cyber-physical critical infrastructure systems," vol. 102 of *Advances in Computers*, pp. 63–110, Elsevier, 2016.

5. M. Woodard, K. Marashi, and S. Sedigh Sarvestani, "Survivability evaluation and importance analysis for complex networked systems," *Submitted to IEEE Transactions on Network Science and Management*, 2017.

6. N. Jarus, M. Woodard, K. Marashi, S. Sedigh Sarvestani, J. Lin, A. Faza, and P. Maheshwari, "Survey on modeling and design of cyber-physical systems," *Submitted to ACM Transactions on Cyber-Physical Systems*, 2017.

7. M. Woodard, S. Sedigh Sarvestani, and A. R. Hurson, "A Petri-net model for propagation of corrupted data in a networked system," *In Preparation for Reliability Engineering & System Safety*, 2017.

This dissertation covers two related topics - modeling of dependability attributes for CPS, and modeling of CPS behavior. For ease of reference, the background and related work for each topic immediately precedes the description of the related research contribution. Section 2 discusses dependability attributes and the state-of-the-art in their modeling and evaluation. This section begins with a discussion of the background and related work in reliability modeling and evaluation in Section 2.1 followed by my work in CPS reliability modeling in Section 2.2. A discussion of related work in performability, resilience and survivability modeling is presented in Section 2.3, followed by my work in survivability evaluation and component importance analysis in Section 2.4.

Section 3 discusses system behavior modeling. This section begins with an overview of intelligent transportation systems in Section 3.1, followed by my proposed model for autonomous vehicle behavior, in Section 3.2. A discussion of related work in data corruption and data cleansing as it pertains to CPSs is presented in Section 3.3. My qualitative and quantitative models for the propagation of corrupted data in CPSs are articulated in Section 3.4. This dissertation concludes with Section 4, where future extensions to the work are also described.

2. MODELING AND EVALUATION OF DEPENDABILITY ATTRIBUTES FOR CPS

Two types of attributes characterize systems: *functional* and *non-functional*. The functional requirements of a CPS describe the operational and performance requirements for the cyber and physical infrastructure of that system. Security, interoperability, and reliability, which drive the design of all CPS infrastructures, are examples of non-functional attributes.

Dependability is the ability of a system to provide a justifiably trustable level of service [13]. It describes the behavior of a system over its lifetime: its ability to deliver services and to avoid and recover from faults. Avizienis et al. [13] provide a taxonomy of dependability aspects, which are summarized here. Dependability is a broad concept that encompasses many metrics, including availability, reliability, safety, integrity, and maintainability. Metrics may be *qualitative*, describing principles of system design and behavior, or *quantitative*, providing a means to compare different systems' operations.

In order to define various metrics, an understanding of the types of system events that may be measured is required. These definitions are from the work of [14]. At the lowest abstraction level is the system component, which may experience a *defect*. A *fault* occurs when a component (whether defective or simply improperly designed) ceases to perform its function perfectly. Faults are undetectable by system monitoring but may be found through thorough examination. An *error* occurs when one or more faults threaten to compromise system performance. A *failure* occurs when the system is unable to perform as intended; that is, the service the system provides is *degraded*. Failures may be localized to one area of a system (such as a power grid not serving some customers); a *complete failure* causes the system to cease functioning entirely.

System dependability was initially defined in terms of reliability, availability, and robustness. These metrics take a binary view of the system: either it is functional or

it has entirely failed. However, these metrics are considered to be too pessimistic to accurately model large-scale systems and thus cyber-physical systems. For example, a nation-wide power grid may experience a service outage in one area, but may still, be providing service to other areas. This led to the development of more granular metrics, such as performability, resilience, and survivability that take the level of service a system provides into consideration. Dependability metrics also differ based on which portion of the *system lifecycle* they measure. As such, no one metric can claim to entirely capture system dependability; several models must be used to judge the trustworthiness of a system's service. Figure 2.1 shows the portions of system lifetime modeled by these metrics.

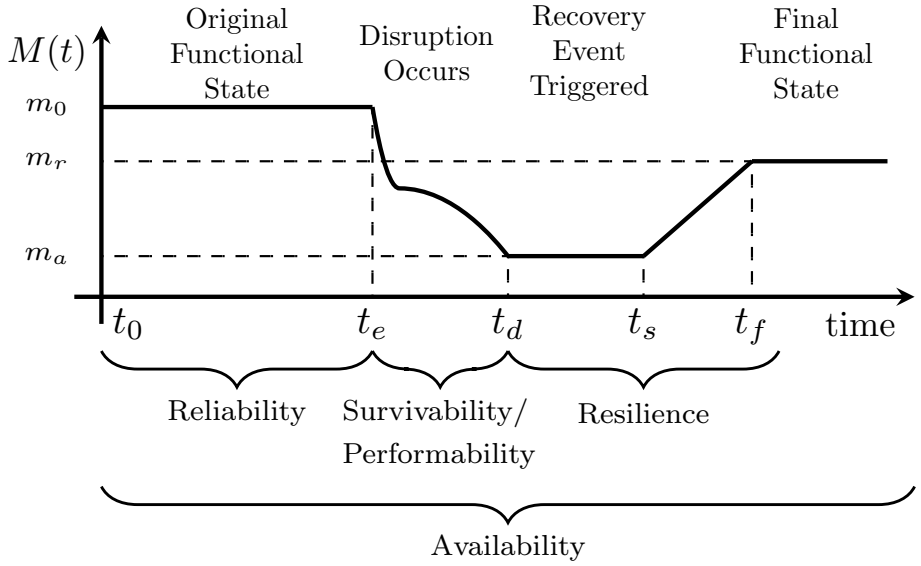


Figure 2.1. Interval of system lifetime covered by aspects of dependability.

2.1. RELIABILITY

Reliability is concerned with system behavior before a failure. *Reliability* is the ability of an item to perform a required function under given conditions for a given time interval [15]. However, once the system fails, reliability does not consider partial system functionality or the system's ability to recover; thus, it is a binary measure of continuous operation. In other words, reliability considers every system failure to be a complete

failure. Reliability is mathematically modeled using probability. Let X be a continuous random variable representing the system lifetime beginning at the time origin and ending at the instant of system failure. A system's reliability at time t can be expressed as in Equation 2.1.

$$R(t) = Pr\{X > t\} \tag{2.1}$$

If $F(t)$ denotes the cumulative distribution function (CDF) of X, reliability can be expressed as in Equation 2.2.

$$R(t) = 1 - F(t) \tag{2.2}$$

Or in other words, the reliability of a system is the probability of it not failing within some interval $[0, t]$.

One of the earliest reliability analysis tools is the fault tree [16]. Fault trees encode the connections among system components using logic gates. Different types of gates represent the different effects the failure of a component can have on the system. For instance, an AND gate indicates that both subsystems it relates must be functional; this would apply to a system of two components connected in series, among others. An OR gate would model a system with two parallel components that requires both to experience a fault before the system fails. The tree of gates can be analyzed using either analytical or numerical methods [16, 17]. They may also be converted to reliability block diagrams for analysis [18]. While fault trees are intuitive, it is difficult to capture some types of component interdependence with them. Thus, modeling some complex systems with fault trees can be labor-intensive.

Reliability block diagrams (RBDs) [19] provide another visual means of modeling reliability. Instead of the logic gates of the fault tree, system components are represented as switches in an electrical circuit. If the circuit remains complete between input and output, the system remains functional. As such, RBDs can be used to analyze the probability of the system being functional and thus the system's reliability. Additional analysis can

be performed by converting the RBD to a fault tree and applying appropriate analysis techniques [18]. In particular, Boolean algebra can be used to reduce the complexity of the RBD and thus simplify analysis [20].

Markov chains have been applied to reliability modeling in numerous ways [21]. A Markov chain consists of states in which a system may be and transitions among those states that are taken with some probability. The Markov assumption constrains these probabilities: assuming that the probability of transitioning to a given state depends only on the state the system is currently in. Formally, if X_n is a random variable denoting the state of the system at time step n, the probability can be expressed as in Equation 2.3.

$$P\{X_n = x_n | X_{n-1} = x_{n-1}, \ldots, X_1 = x_1\} = P\{X_n = x_n | X_{n-1} = x_{n-1}\} \tag{2.3}$$

One notable technique, the Markov chain imbeddable structure (MIS) method, is presented in [22, 23]. Each state in the Markov chain represents one of the possible permutations of system component failure. The states that result in overall system failure are identified. Transitions between states take place with a probability dependent on the reliability of individual components. System reliability is then defined as the likelihood of being in a functional system state after taking one step through the Markov chain for each component in the system. The MIS technique thus models system reliability as a function of individual component reliability.

Stochastic Petri nets (SPNs) are another tool used extensively for modeling reliability. A Petri net is a bipartite directed graph where the set of nodes consists of two disjoint sets: places and transitions [24]. Directed arcs connect places to transitions and transitions to places. Each place contains a non-negative number of tokens. The state of the system, referred to as the *marking*, is dictated by the distribution of tokens in various places within the Petri net. The change in the Petri net's marking is controlled by the firing condition of the transitions. For instance, a transition may fire once each place that has an arc to

that transition contains a token. When a transition fires, tokens are removed from places connected to the transition by an input arc and are added to places connected to the transition by an output arc. The transitions in an SPN utilize exponentially distributed firing times.

SPNs provide a graphical model of system behavior similar to Markov chains. For analysis, they can be reduced to continuous-time Markov chains (CTMCs) to obtain a steady-state representation of reliability [24]. SPNs provide a more concise representation of a system than traditional CTMC modeling, as each marking of the Petri net corresponds to a state in a CTMC. Extensions of SPNs have been proposed for modeling more complex systems, including high-level smart grid control centers [25] and subcomponents such as multi-source power systems [26].

2.2. QUANTITATIVE RELIABILITY ANALYSIS OF CPS

CPSs incorporate components of heterogeneous domains, hence, addressing the issue of developing a comprehensive model of CPS is a complex task and requires a thorough knowledge of the joint dynamics of embedded computers, software, networks, and physical processes [2]. Modeling issues of CPSs have been addressed by quite a few investigations, most of them focusing on the problem from a qualitative point of view. Faza et al. [27] is one example of the very few studies that address quantitative modeling of CPSs. It develops a mathematical reliability model that captures impairments of both physical and cyber processes and demonstrates its applicability on smart power grids. Similarly, few studies have presented quantitative reliability models that capture physical and cyber processes in Intelligent Water Distribution Networks (IWDN).

A completely satisfactory IWDN should supply water in the required quantities at desired residual heads throughout a specified period. How well an IWDN can satisfy this goal can be determined from water-supply reliability. However, evaluation of IWDN reliability is relatively complex because reliability depends on a large number of parameters, some of which are the quality and quantity of water available at the source, power outages,

pipe breaks and valve failures, variation in demands, and failure of networked computers that regulate the water flow through the widespread water supply network. However, the dependability of IWDNs has not been researched thoroughly. The reliability of WDNs, from a purely physical point of view, has long been a topic of interest to the civil engineering community.

Dasic and Djordjevic [28] used mechanical reliability (probability of pipe failure) and hydraulic reliability (probability of pressure and velocity being satisfactory) to present a method for reliability evaluation of water distribution systems. Jun et al. [29] aimed to quantitatively measure the failure impact in terms of expanding subnetworks and the number of customers out of service. Ezell et al. [30] proposed a probabilistic risk analysis model to capture a water distribution system's interconnections and interdependencies.

Wagner et al. [31] proposed analytical methods for computing the reachability (the case in which a given demand node is connected to at least one source) and connectivity (the case in which every demand node is connected to at least one source) as topological measures for water distribution system reliability. The study was complemented through stochastic simulations in which the system is modeled as a network whose components are subject to failure with given probability distributions. Probabilities of a specific shortfall, number of failure events in a simulation period, and inter-failure times and repair durations are used as reliability measures.

Gupta and Bhave [32] discuss different failure conditions, and obtain water flows for the part of the network that remained after the removal of pipes (contingency analysis), and use them in reliability assessment. Mays [33] presented a survey and reviewed methods for risk and reliability evaluation of water distribution systems. This survey discusses reliability and availability of repairable and nonrepairable WDNs and investigates algorithms for optimal solutions to reliability-based designs of WDNs. In a more recent study, Torii and Lopez [34] investigate the application of the adaptive response surface approach to the reliability analysis of WDNs.

In some of the aforementioned studies, simulation tools are utilized to develop models that describe the behavior of a system. Specialized models and simulation tools exist for the engineering domains represented in critical infrastructure, including power, water, and transportation. They have been created with the objective of accurately reflecting the operation of the physical system at high spatial and temporal resolution. However, intelligent control is usually not captured in these tools, leaving them incapable of representing cyber-physical infrastructure systems. Interdependencies among the cyber and physical components, in operation and failure, present a major challenge, as they invalidate simplified models that assume components will fail independently [35, 36].

This work differs from the above-mentioned studies in that it presents a quantitative model that spans the cyber and physical and captures the interdependencies (and tight coupling) inherent to any CPS. Ideally, a quantitative reliability model for an IWDN would present a mathematical form that encompasses impairments of the system, originating from both physical domain (e.g., pipes), and cyber processes (e.g., control software, communication links, and sensors). As such, the MIS technique was selected as the foundation the model.

2.2.1. Markov Chain Imbeddable Structure for CPS Reliability. The specific contribution of this section is a quantitative model that captures the overall reliability of a CPS. This work was published in [7]. The foundation of this model is the MIS technique, which classifies the states of a system into "functional" and "failed" states, in effect partitioning them based on their effect on reliability. This technique derives a metric for system reliability from component-level reliability measures. Each component can be in either a functional or a failed state. The combinations of component states that result in a failed system are identified. Then, a Markov chain is constructed where transitions between system states occur when components fail. Analysis of the Markov chain reveals the probability that the system remains functional after a certain number of component failures. Faza et. al [27] use this technique to compare changes to cyber control algorithms

and cyber device placements in a CPS, as well as the effect of physical faults on the control system. Their work provides methods for reducing model complexity, making the modeling of real-world CPSs feasible with the MIS technique.

This technique was applied to compare the effects of cyber and physical component failures on overall system reliability by modeling the cyber infrastructure as a single component with a specific reliability.

2.2.2. Markov Chain Imbeddable Structure. The Markov imbeddable structure (MIS) technique [37] models system reliability as a function of individual component reliability. The MIS technique involves the following steps. First, the set of vulnerable components of the system is recognized. Subsequently, the state of the system can be defined as a binary vector. The state of a system with n components can be represented by an n-dimensional binary vector, S, where its i^{th} element reflects the operational state of the corresponding component. There are 2^n such vectors exist, reflecting all possible states of the system. Next, each of the 2^n system states is inspected to determine if the overall system is "functional" or "failed". The MIS model then computes the system reliability as the probability of being in one of the "functional" system states.

The system reliability is computed as follows. Let Π_0 denote a vector of probabilities, where $Pr(Y_0 = S_i)$ is the probability of the system initially being in state S_i. In a normal system, the initial state would be S_0, which represents a system with no component failures as in Equation 2.4.

$$\Pi_0 = [Pr(Y_0 = S_0), Pr(Y_0 = S_1), \ldots, Pr(Y_0 = S_N)]^T \tag{2.4}$$

Furthermore, for a given component, l, the matrix Λ_l represents the state transition probabilities of the system as a function of l. In other words, each element $p_{ij}(l)$ in the matrix Λ_l represents the probability that the system will make a transition from state S_i to state S_j due to the failure of component l. Finally, a vector u is defined, with length equal to 2^n, where

each element has a value of 1 if the corresponding state is considered a "functional" state for the system, and 0 otherwise. The overall reliability of the n-component system can be expressed as in Equation 2.5

$$R = (\Pi_0)^T \left(\prod_{l=1}^{n} \Lambda_l \right) u \tag{2.5}$$

2.2.3. Case Study: Intelligent Water Distribution Network. In this section, the proposed approach is demonstrated by applying it to an intelligent water distribution network (IWDN). Figure 2.2 depicts a practical example of an IWDN. Information such as demand patterns, water quantity (flow and pressure head), and water quality (contaminants and minerals) is critical for providing a dependable supply of potable water and is beneficial in guiding maintenance efforts and identifying vulnerable areas requiring fortification and/or monitoring. Sensors dispersed in the physical infrastructure collect this information, which is fed to the distributed algorithms running on the cyber infrastructure. These algorithms provide decision support to hardware controllers (e.g., valves), which are used to control the quantity and quality of the water.

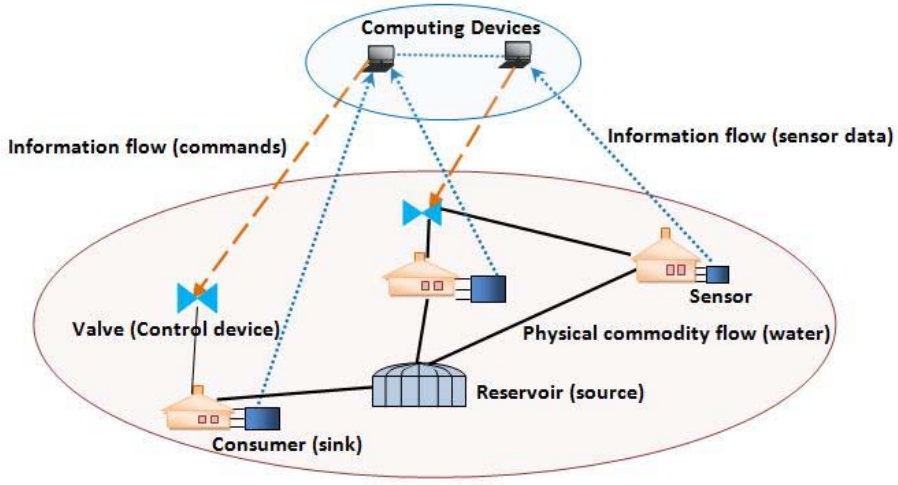


Figure 2.2. An intelligent water distribution network.

The case study IWDN used was based on the network described and analyzed in [31]. The case study IWDN, depicted in Figure 2.3, consisted of two sources (reservoir node

1, and tank node 11), nine demand nodes (2-10), four valves (96, 97, 98, 99), and thirteen pipes (1-11, 98a, 98b). Table 2.1 shows the parameters of some important nodes and pipes of this WDN as used in the simulations. Additional components such as a tank and multiple valves were added to create a more robust system but the basic structure (i.e., elevation of nodes and topology of the network), as well as the supply and demand specifications, were not altered.

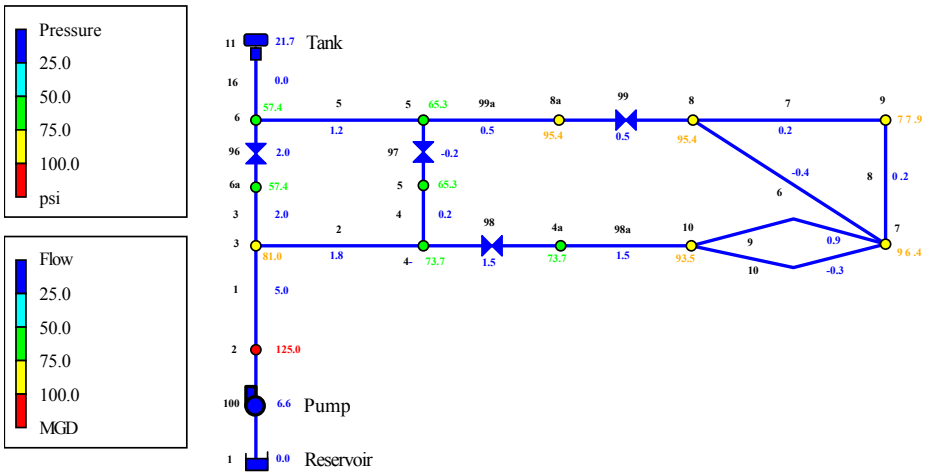


Figure 2.3. Topology of intelligent water distribution network.

The reservoir is capable of providing an infinite supply of water, while the tank's supply is limited by the tank diameter and water level, which are set at initialization. The system has a single pump located at the reservoir that maintains the flow and head. The pipes and valves are the major components that control the flow and provide water to the consumers represented by the demand nodes.

2.2.3.1. Component failures. The vulnerable physical components selected for this reliability analysis included pipes, valves, pumps, and tanks. The IWDN used in this case study had a total of 20 failable components, resulting in 20 single component failure cases and 190 double component failure cases. All failed components were simulated as being in a zero-state (i.e., failures of pipes, valves, pumps, and tanks were evaluated by essentially removing the component from the system). The decision support system consisting of sensors, communication infrastructure, and computing systems was modeled

Table 2.1. Parameters of intelligent water distribution network.

Node	Elevation (ft)	Demand (mgd)	Normal head (ft)	Minimum head
1	100	-6.625	100	-
2	100	0.7355	388.48	146
3	200	1.2	386.43	246
4	210	0.6	376.80	256
5	230	0.4	377.54	276
6	250	0.82	380.05	296
7	10	0.6	173.57	56
8	10	0.8	170.31	56
9	50	0.4	160.87	96
10	25	0.2	181.37	96
8a	10	0	-	-
4a	210	0	-	-
6a	250	0	-	-
5a	230	0	-	-
Pipes	**From node/To node**	**Length (ft)**	**Diameter (in.)**	**Roughness**
1	2/3	200	16	120
2	3/4	1500	12	120
3	3/6a	1800	14	120
4	4/5a	2000	10	120
5	6/5	1900	14	120
6	8/7	1000	8	120
7	8/9	2500	10	120
8	7/9	3500	8	120
9	10/7	1500	10	120
10	7/10	1500	6	120
11	11/6	1000	12	100
98a	4a/10	500	6	65
99a	5/8a	500	4	65

as a single component that was classified as either functional or failed. The failure of the decision support was evaluated as a purely physical system. While the physical and intelligent control components can fail into non-zero states, these states were not evaluated for simplicity and will be explored in future work.

2.2.3.2. System failure. The MIS technique requires the functional and failed system states to be defined. These definitions are based on the system's service parameters. In an IWDN, a system is considered "failed" if the pressure and/or flow drop below a specified

value at one or more nodes [38]. For this case study, a system failure is defined as one or more of the following three cases.

1. Negative pressure case: If the IWDN exhibits a negative pressure in any pipe or node. A negative pressure in a component will result in additional component failures and further system degradation.

2. Quality failure case: If the system fails to provide a minimum of 80% of the demand at every node. While there are many cases where only one node's demand is not met and the remaining nodes are supplied with the full demand, the system is still considered failed due to unsatisfactory supply.

3. Excessive fault case: If three or more components have failed. This case is the threshold case, set based on the size and number of components of the system.

2.2.3.3. System state inspection. A simulation was used to observe the operation of the IWDN and to determine the failed and functional states. Initially, a purely physical WDN was constructed and simulated using EPANET 2.0 [39]. EPANET is a WDN simulator developed by the US Environmental Protection Agency to simulate functional aspects of the system such as demand patterns, water quantity (flow and pressure head), and water quality (contaminants and minerals). However, it has no capability to simulate intelligent decision support, so to simulate the operation of the intelligent control, a MATLAB control algorithm described by Lin et al. in [40] was used.

The IWDN simulation was conducted as follows:

1. Set fault conditions

2. Run EPANET and generate report

3. Parse report and extract input for decision support

4. Run decision support algorithms to determine controller settings

5. Output control settings as a .inp file

6. Provide .inp file to EPANET and produce simulation results

In order to simulate the IWDN, the method introduced by Lin et al. [41] was used. The simulator produced a report which contained the flow and pressure at each node and in each component as well as a negative pressure warning. These reports were parsed and loaded into MATLAB to determine if the specified fault conditions resulted in a system failure. Figure 2.4 illustrates the procedure to simulate the function of IWDN as a CPS.

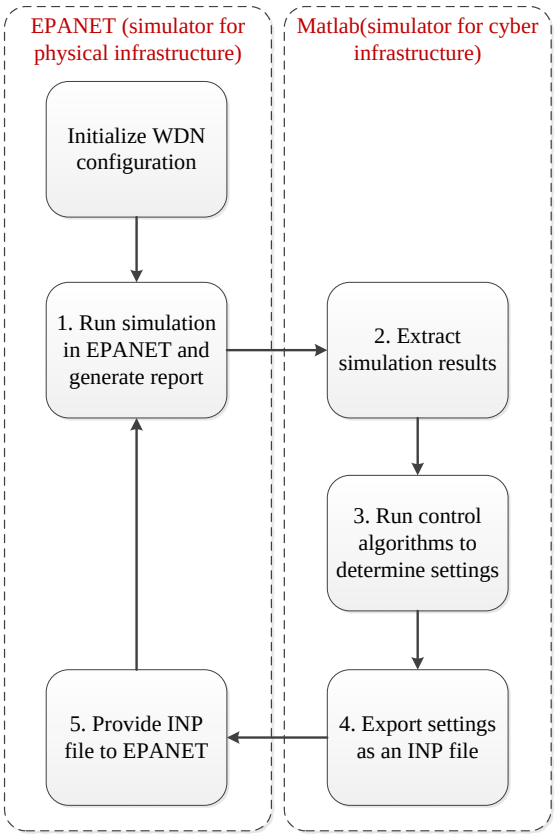


Figure 2.4. Procedure to simulate an IWDN using EPANET and MATLAB.

2.2.4. Results. The results from single-component and double-component failures were used to generate the mathematical reliability model. As previously stated, all triple-component failures were considered as failed states. Some of the failures are listed and briefly described in Table 2.2.

Table 2.2. List of some component failures and their consequences.

Failed Component	System State	Failure at hour	Failure description
Pump	Failed	3	negative pressures
Tank	Functional	-	
Pipe 1	Failed	6	negative pressures
Pipe 6	Functional	-	
Pipe 98a	Failed	0	negative pressures
Pipe 7 and Pipe 8	Failed	0	negative pressures
Valve 98	Failed	3	negative pressures
Valve 99	Functional	-	

The failure case observations were used to populate the u vector as described in Section 2.2.2. Using Equation 2.5, the mathematical model of reliability for the IWDN case study, R_{sys} , is developed as shown in Equation 2.6.

$$\begin{aligned} R_{sys} = {} & p_P.p_T.{p_L}^{13}.{p_V}^4.p_C + p_P.q_T.{p_L}^{13}.{p_V}^4.p_C + \\ & 8p_P.q_T.{p_L}^{12}.q_L.{p_V}^4.p_C + p_P.q_T.{p_L}^{13}.{p_V}^4.q_C + \\ & 10p_P.p_T.{p_L}^{12}.q_L.{p_V}^4.p_C + 31p_P.p_T.{p_L}^{11}.{q_L}^2.{p_V}^4.p_C + \\ & 16p_P.p_T.{p_L}^{12}.q_L.{p_V}^3.q_V.p_C + 5p_P.p_T.{p_L}^{12}.q_L.{p_V}^4.q_C + \\ & 3p_P.p_T.{p_L}^{13}.{p_V}^3.q_V.p_C + p_P.p_T.{p_L}^{13}.{p_V}^4.q_C \end{aligned} \tag{2.6}$$

In this mathematical model, p_P is the reliability of the pump, p_T is the reliability of the tank, p_L is the reliability of the pipes, p_V is the reliability of valves, and p_C is the overall reliability of the control algorithm. The unreliabilities are expressed as $q_T = 1 - p_T$, $q_L = 1 - p_L$, $q_V = 1 - p_V$, and $q_C = 1 - p_C$ for the corresponding components. This model is a simplified version and holds true when all similar components are equireliable (i.e., $p_{L_i} = p_L, \forall i$, and $p_{V_i} = p_V, \forall i$).

In Figures 2.5, 2.6, and 2.7, the comparison among the system reliability with different values for reliabilities of the control system, pump, and valves, respectively. Comparing these three figures illustrates that the effect of using reliable valves and pumps

on the overall reliability of the system is more important than using a reliable control system that implements the intelligent decision support. This is because the model distinguishes between basic rule-based control routines (which is conducted in EPANET) and intelligent decision support (which is conducted in MATLAB). Failure of control system does not eliminate the basic IF-THEN rules that control valves and pumps in the WDN. Hence, the effect of the reliability of the actuators on the overall reliability of the system as a whole is higher than that of the intelligent decision support.

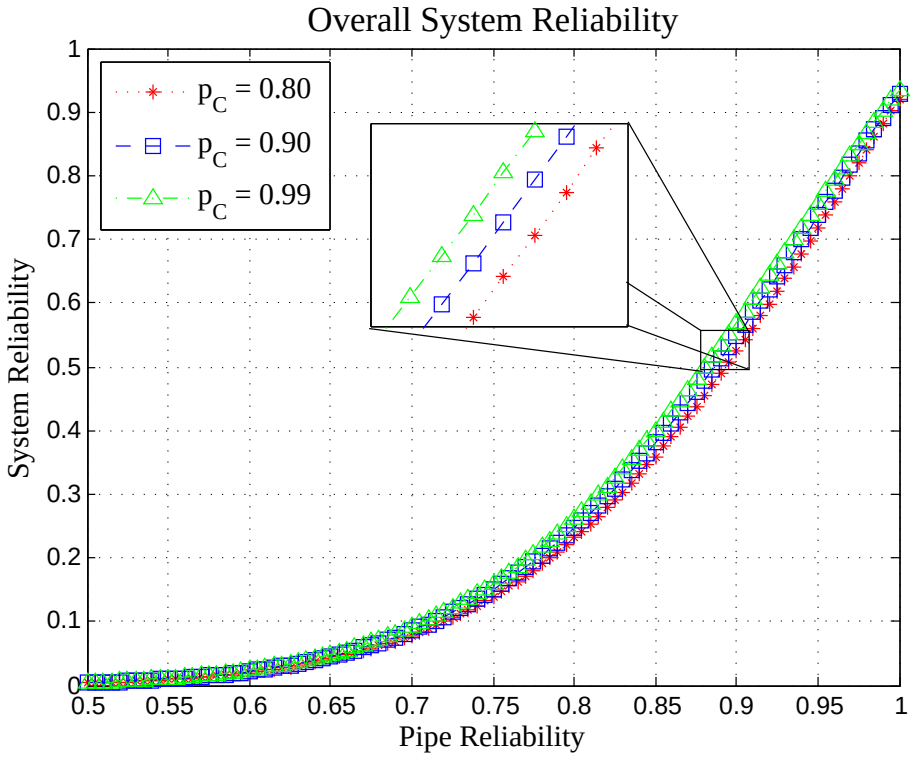


Figure 2.5. System reliability, comparison among control systems with different reliabilities. $p_P = p_T = p_V = 0.97$

2.2.5. Summary of Research Contribution. The research contribution presented in Section 2.2 is a system-level reliability model for critical infrastructure CPSs based on the MIS technique. This reliability model was demonstrated using an IWDN as a case study. The IWDN consisted of a control system, two sources, nine demand nodes, four valves, and thirteen pipes. The control system was added to the system to prevent against potential failures and increase the reliability of the water distribution system. The failed

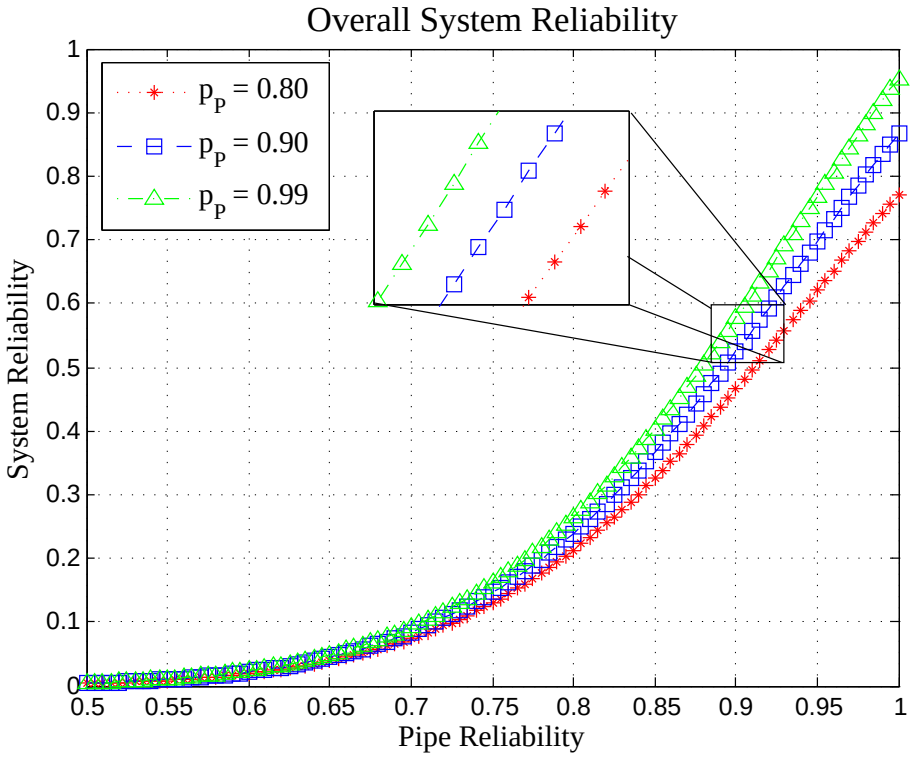


Figure 2.6. System reliability, comparison among pumps with different reliabilities. $p_C = p_T = p_V = 0.97$

and functional system states were determined by setting an acceptable service threshold and evaluating failure cases using EPANET and MATLAB as physical infrastructure and cyber infrastructure simulators, respectively. This state information was then used in the MIS technique to develop a mathematical reliability model for the IWDN. Comparison among the results of the simulations shows that the effect of impairments of low-level actuators (e.g., valves), on the functionality of the system is more intense than that of high-level decision support that finely adjusts the settings in order to reach the highly-efficient operation.

In future studies, additional cyber and physical component fault states would be included to incorporate a more realistic behavior of the system. Moreover, the decision support system will be modeled as multiple components in a hierarchical structure with sensors at the bottom moving up to the computing systems with communication links connecting these elements. Initially, each of these control subcomponents will be modeled

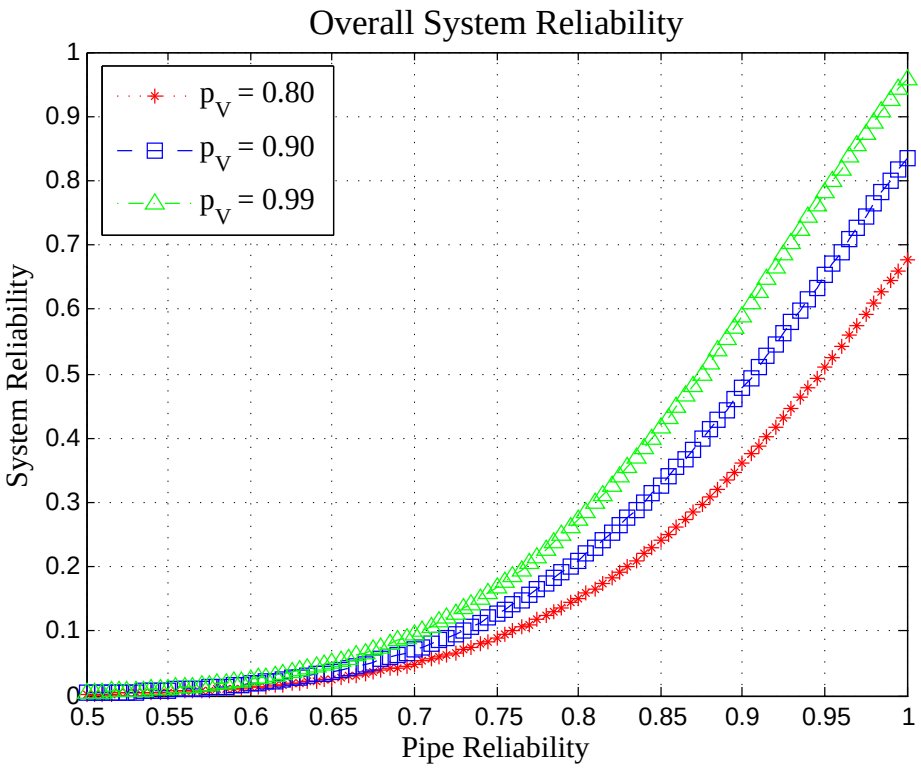


Figure 2.7. System reliability, comparison among valves with different reliabilities. $p_P = p_T = p_C = 0.97$

as a failed system and then expanded to analyze other faulty states to evaluate systems with degraded states.

2.3. SURVIVABILITY, PERFORMABILITY AND RESILIENCE

Although dependability attributes are well studied for small- to large-scale networks and systems, they fall short in describing characteristics of modern complex CPSs. For example, reliability and availability, two important aspects of dependability, consider the system state to be binary—*operational* or *failed*. This view is inadequate for critical infrastructures, which are expected to deliver uninterrupted service despite continual disturbances. It is expected that a large-scale system will spend time in functionally degraded

states without interruption of essential services. Consequently, additional dependability attributes are required to characterize these degraded, yet operational, states.

Performability, introduced by Meyer et al. [42], combines performance and availability to evaluate system effectiveness, taking into account behavior due to failures. In other words, a system can be in a fully functional state, a partially operational state with degraded performance, or a failed state. Performability evaluates the expected performance over a duration composed of alternating functional/degraded/failed periods. System capability quantifies the extent to which users can expect to benefit from a system when it is in a specific state. Iyer et al. [43] discuss developing performability models of fault-tolerant systems that use a capability function, $M(t)$, to relate the state of a system at time t to the overall system performance level.

The performability of a system from the time origin t_0 to time t is given by Equation 2.7.

$$Perf_{sys}(t) = \int_{t_0}^{t} M(x)\,\mathrm{d}x \tag{2.7}$$

In this equation, $M(t)$ is the reward function associated with performance per unit time and t is the mission time of the system. The mission time of the system is the duration over which the system is expected to be operational. Performability is focused on mission time and becomes difficult to calculate if repairs occur during operation since the mission time can then become unbounded. CPS performability evaluation techniques are presented in [44, 45] using Markov reward models—Markov chains that earn a reward dependent on the state the system is in. Performability is useful for evaluating systems that are initially in a perfect functional state but it fails to capture repairs after a complete failure and subsequently fails to capture long-term system operation.

Resilience takes a more refined view of system availability. Resilience is defined as the ability of a system to "bounce back" from failure [46, 47], but its application is limited to the recovery phase that follows a failure and not any period beforehand. Avizienis et al/ [13] mention resilience as a synonym for fault tolerance. Ouyang et al. [48] expand

this definition to include the ability of a system to resist different possible hazards, absorb initial damage from a failure or attack, and recover to normal operation. Mathematically, resilience is defined in Equation 2.8.

$$\Lambda(t) = \frac{M(t)}{M(t_0)} \tag{2.8}$$

As with performability, all quantitative resilience measures rely on some measure $M(t)$ of system functionality at time t, alternatively known as a capability function or a figure-of-merit [46, 49].

Ghosh et al. [50] outline a procedure for developing resilience metrics from reliability, performability, or availability models. First, a stochastic model for the chosen dependability attribute is developed. Next, a particular metric on this model is chosen as the measure of system functionality. Finally, structural or parametric changes, which may include component faults, are made to the system, and the resulting change in functionality is observed. Albasrawi et al. [51] apply this methodology to measure not only the loss of functionality resulting from a cascading failure in a power grid but also the rate at which functionality is regained by different recovery actions. The choice of recovery actions is governed by the maintainability of the system, demonstrating that maintainability and resilience are interrelated.

Nan et al. [52] identify three components of system resilience: ability to absorb faults, ability to adapt and reconfigure in order to reduce the impact of faults, and ability to restore service after degradation. The authors propose a figure-of-merit-based resilience metric that captures these components; it also captures the ability of systems to recover to a level of functionality higher than their initial state. They develop a method for applying this resilience metric to complex, interdependent systems where no unified figure-of-merit exists. The interdependent system is divided into subsystems for which a figure-of-merit can be defined, and the interdependencies between subsystems are then quantified using

input and output variables that allow for simultaneous simulation of subsystem models. Results from simulations can be used to identify the relative effect of each interdependency on overall system resilience and on components that can be improved in order to increase system resilience.

Resilience is extended to systems-of-systems by [53]. The authors develop models that incorporate resilience metrics from several systems, including models of the effects of one system's failure on the resilience of the others. For example, a blackout in part of the power grid may cut power to water distribution centers and trigger a failure in the water distribution network. They model this failure propagation using a deterministic cause-effect model. However, this may not capture all the effects of other systems on the resilience of the system under consideration since the state of other systems can affect both the likelihood of a component fault and the maintenance time required to recover the system after a failure.

Survivability is another dependability attribute that was introduced with a similar objective and is used to characterize degraded operation. Survivability, unlike resilience, can be used to describe degraded operation at any point after a disturbance occurs regardless of whether the disturbance is a fault tolerated by the system or a failure that actually causes degradation. The roots of survivability are in military applications that focus on mission fulfillment. Most definitions of survivability are qualitative; for example, [54] define it as the capability of a system to fulfill its mission in a timely manner in the presence of attacks, failures, or accidents. The mission of a system is a set of very high-level requirements or goals for that system; timeliness means the mission is fulfilled by a user-specified time. Queiroz et al. [55] define survivability as the capacity of essential services to provide their functionalities in cases of malicious attacks that compromise parts of the system. Such functionalities may rely on other services of the system that are not necessarily essential. The definition focuses on a specific service or component that must survive and how the interdependency between services affects that survivability.

Critical CPSs, including smart grids, are expected to autonomously defend against attacks, remediate the consequences of failure, and recover in a timely manner. Dependability attributes provide a coarse-grained characterization of these qualities, unlike survivability metrics, whose very purpose is to precisely characterize transient behavior after a disturbance. For this exact reason, it is used in several different domains including weapons systems engineering [56], telecommunication services [57], information systems [54], and software engineering [58].

No standard definition of survivability was identified at the time of publication, perspectives on the topic are diverse [59]. A qualitative and concise definition is presented by Heegaard and Trivedi [60]:

> "Survivability is the system's ability to continuously deliver services in compliance with the given requirements in the presence of failures and other undesired events."

The ANSI T1A1.2 working group [61] using a domain-specific FoM, as shown in Figure 2.1, to quantitatively defined survivability for networked systems:

> "Suppose a measure of interest M has the value m_0 just before a failure occurs. The survivability behavior can be depicted by the following attributes: m_a is the value of M just after the failure occurs; m_u is the maximum difference between the value of M and m_a after the failure; m_r is the restored value of M after some time t_r; and t_R is the time for the system to restore the value of m_0."

Comparison of survivability evaluation techniques is complicated by the lack of a common definition for this attribute. A number of approaches have been used to model survivability. Zhang et al. [62] present a qualitative approach using attack graphs. In this approach, an attack graph is created using known system vulnerabilities and their associated difficulty parameters. Each path represents a series of exploits leading to an undesirable state, each node represents the network states under attack, and each directed edge represents

an attack action. Survivability analysis is conducted by defining the states in the attack graph where the system fails completely and by determining the cost associated with each attack. In this case, survivability is associated with the difficulty and the destruction level of an attack, which quantitatively defines survivability as the minimal cost to compromise the system. This model captures the probability that a system will meet its mission requirements in the presence of an attack, but neglects the presence of survivable system enhancements. It also does not model the timeliness or ability of a system to recover or account for the graceful degradation of a system.

Knight et al. [63] present a survivability definition and model based on a service state-transition graph constructed from a six-tuple of service specification levels, service value factors, reachable environmental states, relative service values, set of valid transitions, and service probabilities. Ma [64] similarly quantifies survivability as a four-tuple of resistance, resilience, persistence, and failure count. In this study, *resistance* refers to the ability to withstand an attack, *resilience* refers to the mean recovery time from a catastrophic failure, *persistence* is the ability to maintain or exceed the minimal threshold of required functionality, and *failure count* describes the number of failures encountered over the duration of observation. These individual attributes are well-defined but disjoint, and as such the study does not lead to a practical approach for quantitative evaluation of survivability.

Liu and Trivedi [65] introduce a method of modeling survivability that uses continuous time Markov chains (CTMC) by combining a pure performance model and a pure availability model to construct a composite performance-availability model. The pure availability model for a system's resources is modeled as a birth-death process where each state represents the number of functioning assets. The pure performance model is created using task arrival rates and service rates for the system. This is also a birth-death process, but here each state represents the number of assets currently tasked. The two models are combined to create a composite model which is then truncated based on the survivability measure of

interest. The desired survivability measures are obtained using transient analysis. Many extensions to this model have been presented to incorporate different aspects of survivability; for instance, [66] developed a checking algorithm to decide whether a system is survivable. Continuous stochastic logic [67] is used to phrase survivability in a precise manner for CTMC models. Heegaard and Trivedi [60] expand and refine this method to determine the scalability of the model, as well as to model additional performance measures such as failure propagation and recovery using a phased recovery model.

Kim et al. [68] model the survivability of a wireless sensor network using a semi-Markov process (SMP) instead of a simple Markov chain. The SMP captures the behaviors of attacks, system responses to the attacks, intrusion detection, and repairing mechanism that cause sojourn time to be non-exponential.

System survivability has also been modeled using Petri nets. Castet and Saleh [69] explore the applicability of stochastic Petri nets for multi-state failures and survivability analysis. They model components with multiple operational states, allowing them to analyze survivability and focus on failure propagation in the system that results in either graceful degradation or catastrophic failure.

2.3.1. Alternative Survivability and Resilience Definitions. The majority of qualitative survivability and resilience definitions in the literature are similar to the ones presented in this survey. However, [70] use alternative definitions that are focused more on communication networks and are thus less applicable to CPSs. They define resilience as "the ability of the network to provide and maintain an acceptable level of service in the face of various faults and challenges to normal operation," which is analogous to the definitions of survivability used elsewhere in this paper.

Sterbenz et al. [70] classify resilience as a field of related disciplines categorized as *challenge tolerance* and *trustworthiness*. These disciplines target the dependability attributes outlined previously in this section with more emphasis on security and networking issues. Challenge tolerance captures network design issues such as traffic and disruption

tolerance and survivability. The survivability sub-discipline encapsulates resistance to multiple correlated failures, such as a disaster or attack, as well as fault tolerance and resistance to few random independent failures. The other half of their definition of resilience, trustworthiness, captures dependability [13], performability [42], and security.

2.4. SURVIVABILITY EVALUATION AND COMPONENT IMPORTANCE ANALYSIS OF NETWORKED SYSTEMS

A smart grid, which integrates cyber infrastructure with the traditional power grid, is a prime example of cyber-physical critical infrastructure. In traditional power grids, reliability and availability metrics, such as system average interruption duration index (SAIDI), could sufficiently describe a system's dependability. The integration of cyber infrastructure, however, has motivated the use of more complex metrics such as survivability.

Menasché et al. [71] introduce a new metric extended-SAIDI (ESAIDI) for analyzing survivability aspects of the smart grid. ESAIDI is an extended version of SAIDI for analysis of the consequences of failures in distribution automation in the power grid. Avritzer et al. [72] utilizes the same approach and extends the model to account for disruptions in the communication infrastructure. This work was later combined with power flow analysis to create a survivability model that facilitates optimal design for the automation system of the smart grid [73]. The work was again extended to account for concurrent failures of the power system [74]. All of these approaches [71, 72, 73, 74] use time-to-recovery as a measure of survivability of the system. However, time-to-recovery includes both the failure and the recovery processes, and thus the behavior of the system in each of these areas is indistinguishable.

Alobaidi et al. [75] propose a quantitative approach for survivability evaluation of smart grids by studying the relationship between system condition (in terms of a number of functional components) and system capacity (ability to provide power to customers). The authors also present recovery strategies that minimize the effect of failures on system

survivability and verify the applicability of their proposed methods on a test case. A limitation of [75] that is addressed in this work is the restriction to power grid systems.

In another study, Chopade and Bikdash [76] utilize a graph-theoretic method to model the survivability of a smart grid. Graph-theoretic measures such as degree distribution and clustering coefficient are utilized, but all buses (vertices) and lines (edges) are assumed to be identical. This is an unrealistic assumption, and the resulting survivability model fails to capture differences in reliability, among other features, of the buses and lines. Similar to the graph-theoretic approach, [77] quantitatively evaluates the survivability of mobile ad hoc networks (MANETs) by representing the probability that all active nodes are k-connected to the network. A semi-Markov model that captures state transitions of the network due to node failures and malicious attacks is used to determine this probability. The connectedness of MANETs is a measure of functionality of the system; thus, the probability of being k-connected can reflect survivability. While connectivity is appropriate for MANETs, the evaluation method is inapplicable to systems with more complex functionality.

A recent study by Avritzer et al. [59] presents common approaches for survivability evaluation of critical infrastructures (namely, water, gas, and electricity infrastructures). Stochastic hybrid models such as fluid stochastic Petri nets, hybrid Petri nets, and piecewise deterministic Markov processes, as well as graph-theoretic approaches, are recommended as methods for survivability evaluation of these systems.

For many systems, including critical infrastructures, the high availability required makes it infeasible to bring down the system for fault injection studies. Detailed reports of real-world failures are few and far between, and many of the potential failure scenarios have never actually occurred in practice, which necessitates the use of simulation tools. No simulation environment perfectly captures the characteristics of real-world entities; however, simulation does provide a good understanding of system behavior at minimal cost. The goal of this work is to enable survivability evaluation for complex networked systems with

an approach that can rely upon data from simulation, laboratory and/or field observations, and historical data about failure.

This work provides an approach for evaluating the survivability of networked systems and identifying the components most critical to a system's survivability. The survivability evaluation approach relies upon the identification of a domain-specific *figure-of-merit (FoM)* that is indicative of the extent to which one or more essential services are provided. A set of representative failure cases are selected, and the FoM is tracked over the duration of each disturbance in order to determine the rate and extent of service degradation. In this context, a *failure case* is defined as failure of one or more specific components.

This definition and each task associated with the approach are described in Section 2.4.2. This work was published in [10].

Survivability evaluation can be incorporated into the system design cycle as follows:

1. An appropriate FoM and a set of representative failure cases are identified during the initial design phase of the system.

2. Once the system is implemented, changes in the FoM are observed in a field, laboratory, or simulation environment over the duration of each failure case.

3. Survivability is quantified in terms of the *extent* and *rate* of degradation of the FoM.

4. Components whose failure is the most detrimental to survivability are identified and hardened.

5. The process is repeated as necessary until desired levels of survivability are attained.

The distinction of this work is its broad applicability to a range of domains; the FoM reflects domain-specific aspects of the system's operation. The same breadth facilitates analysis of cyber-physical systems, where features such as complexity and heterogeneity render other methods inapplicable. The study presented in [52] is similar to this work in extracting dependability features from a FoM; however, it is focused on comparing

resilience of different system topologies against a single disruptive event and finding an optimal recovery strategy.

2.4.1. Survivability Attributes. In [60], *graceful degradation* and *failure resistance* are described as two attributes essential to survivability. These attributes are defined as metrics for survivability (with reference to Figure 2.1) as follows:

- *Graceful degradation* is achieved when the *rate of degradation*

$$\left|\frac{dM(t)}{dt}\right| \tag{2.9}$$

  is considered to be slow after a disturbance, in the context of the time scale of the system domain.

- *Failure resistance* has occurred when the *extent of degradation*

$$|M(t_d) - M(t_e)| \tag{2.10}$$

  resulting from a disturbance (i.e., the loss in FoM incurred between the start of the disturbance and initiation of recovery), leaves the system functionality at a level considered to be acceptable.

The FoM is domain-specific, as it is intended to capture the extent to which a system is delivering essential services. In this work, the FoM represents a single service. The survivability evaluation approach can be used to represent more complex behavior by defining a FoM that is a composite metric (e.g., a weighted average of multiple FoM), that reflects different essential services.

These two attributes are pivotal to the proposed approach to survivability evaluation and importance analysis, which are described in Sections 2.4.2 and 2.4.3, respectively.

2.4.2. Evaluation of Survivability. The proposed approach for survivability evaluation has two phases:

1. A system-specific FoM and a set of representative failure cases are selected to evaluate the system. Each failure case is then observed or simulated, and the value of the FoM is recorded.

2. Based on the log of FoM values, the rate and extent of degradation of the FoM are calculated. The extent and rate of degradation are plotted on a two-dimensional figure in order to facilitate analysis.

In this context, a *failure case* is defined by the set of distinct components, the failure of which causes a disturbance to the system. In the broadest sense, the failure of this set of components leads to a set of failure cases that differ in the chronological order and exact timing of failures of the respective components. For simplicity and tractability, all component failures of a failure case are considered to have occurred simultaneously, and as such, failure case j, denoted as $\mathbf{F}_j$, can be represented as the set of components whose concurrent failure (at time t_e in Figure 2.1) has initiated the disturbance that is being examined. In this work component-level operation is considered to be binary (i.e., a component is either fully functional or has failed altogether). This assumption is justified where the system representation is fine-grained and the contribution of a single component to the delivery of an essential service cannot be further decomposed.

An exhaustive examination of failure cases is infeasible for large complex systems. Alternatively, the omission of failure cases with catastrophic consequences could render the survivability evaluation meaningless. This state-space explosion problem is common in any type of system evaluation, but its resolution is not within the scope of this work. Existing literature on software or system testing [78] can provide inspiration. In this work, a set of failure cases is assumed to be predefined.

Suppose a system with n components has m failure cases that have been designated as the basis for survivability evaluation. Each failure case is observed or simulated for a duration that begins with a fully functional system where all components are operational, continues through the disturbance caused by failure of the components in $\mathbf{F}_j$, and ends when

recovery efforts are initiated. In other words, observation or simulation of the failure case defined by $\mathbf{F}_j$ produces a record of the FoM, $M_j(t)$, for $t_0 \leq t \leq t_d$, where t_0 and t_d are as defined in Figure 2.1. It is worth noting that the failures of the components initiating the disturbance at t_e, $\mathbf{F}_j$, can lead to failures of other components. This larger set, denoted as $\mathbf{C}_j$, includes any component whose failure is observed between t_e and t_d.

Survivability analysis requires that $M_j(t)$ be examined in order to determine the extent and rate of degradation. The full extent of degradation, denoted as δ_j, is determined between the instant of disturbance (t_e) and the initiation of recovery (t_d). Over the same period, the most rapid rate of degradation is denoted as ρ_j. Equations (2.11) and (2.12) formalize these attributes.

$$\delta_j = \max_{t_0 \leq t \leq t_d} |M_j(t_0) - M_j(t)| \tag{2.11}$$

$$\rho_j = \max_{t_0 \leq t \leq t_d} \left| \frac{dM_j(t)}{dt} \right| \tag{2.12}$$

The rate and extent of degradation are depicted in Figure 2.8. The survivability of a system is determined by aggregating the extent and rate of degradation for all failure cases.

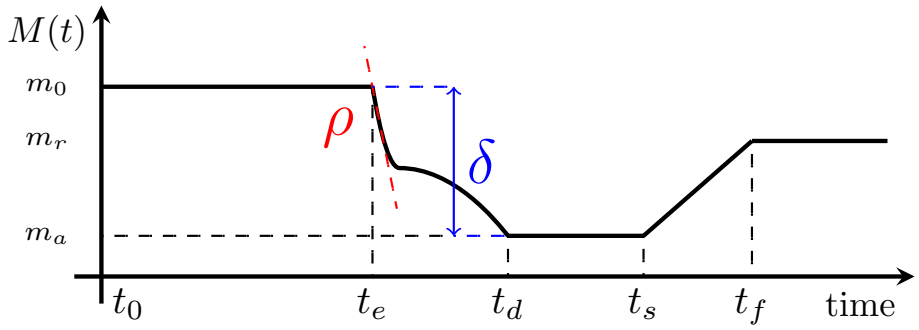


Figure 2.8. Rate and extent of failure depicted on figure-of-merit graph.

For each failure case, a degradation point, (δ_j, ρ_j), shown in Figure 2.9, is used to calculate a degradation index, which is the distance from the degradation point to the origin. The single degradation point (failure case) shown in Figure 2.9 has a δ of 0.6 and a ρ of 0.25, which gives it a degradation index, s, of 0.65. The degradation index facilitates

the comparison of failure cases and can be averaged across all failure cases to determine a single survivability index for the system.

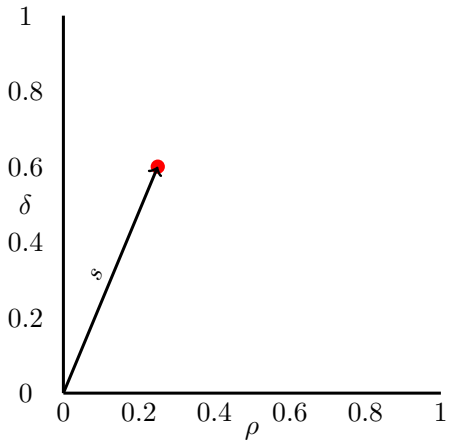


Figure 2.9. FoM degradation rate vs. extent histogram and degradation index. Color indicates the number of failure cases which resulted in the corresponding rate and extent of degradation. In this instance a single failure case is plotted.

Creating a two-dimensional color intensity histogram of these values for all failure cases can facilitate identification of clusters of degradation points, which are indicative of failure cases that are similar in consequence. In an ideal system, only one cluster would be evident, near the origin in the lower left corner of the plot. This cluster is characterized by slow and minimal degradation of the system. Clusters outside of this area represent failure cases that require further investigation, as they demonstrate non-survivable behavior.

2.4.3. Importance Analysis. Evaluation of survivability can illuminate weaknesses in a system. Specifically, the method presented in this work can facilitate the identification of components most in need of fortification (i.e., importance analysis), where the measure of importance is the contribution of a component to survivability. Two criteria for ranking components are proposed, namely *criticality* and *fragility*.

The *criticality* of a component is determined by the consequences of its failure on service degradation, which are evaluated over all failure cases in which the component experiences failure. Associated with each failure case $\mathbf{F}_j$ is a second set, $\mathbf{C}_j \supseteq \mathbf{F}_j$, that encompasses all components observed to fail during the failure case. As described in

Section 2.4.2, the highest degradation incurred during a given failure case, $\mathbf{F}_j$, which is denoted as δ_j. To determine the criticality of the i^{th} component, the failure cases in which the i^{th} component was observed to fail, must be identified. The set of these cases is $\mathcal{S}_i = \{ j \mid i \in \mathbf{C}_j \}$. Additionally, let $t_{i,j}$ denote the time at which component i has failed during failure case $\mathbf{F}_j$. The criticality of a component is thus the product of three terms.

The first term, shown in Equation (2.13), normalizes the extent of degradation to rank the severity of the failure case.

$$\omega_1 = \frac{\delta_j}{\Delta} \tag{2.13}$$

$$\Delta = \max_{\forall j} \delta_j \tag{2.14}$$

The second term, shown in Equation (2.15), normalizes the rate of degradation at the time of component i failure during the failure case.

$$\omega_2 = \frac{\left.\frac{dM_j(t)}{dt}\right|_{t=t_{i,j}}}{\max\limits_{\forall t} \frac{dM_j(t)}{dt}} \tag{2.15}$$

The third term, shown in Equation (2.16), normalizes the second derivative of the FoM at the time of component failure, which calculates the immediate impact of the component's failure on the FoM.

$$\omega_3 = \frac{\left.\frac{d^2M_j(t)}{dt^2}\right|_{t=t_{i,j}}}{\max\limits_{\forall t} \frac{d^2M_j(t)}{dt^2}} \tag{2.16}$$

The product is calculated and summed across all failure cases involving component i and divided by m, the total number of failure cases. The criticality, α_i, of component i is determined as shown in Equation (2.17).

$$\alpha_i = \frac{\sum_{j \in \mathcal{S}_i} (\omega_1 \times \omega_2 \times \omega_3)}{m} \tag{2.17}$$

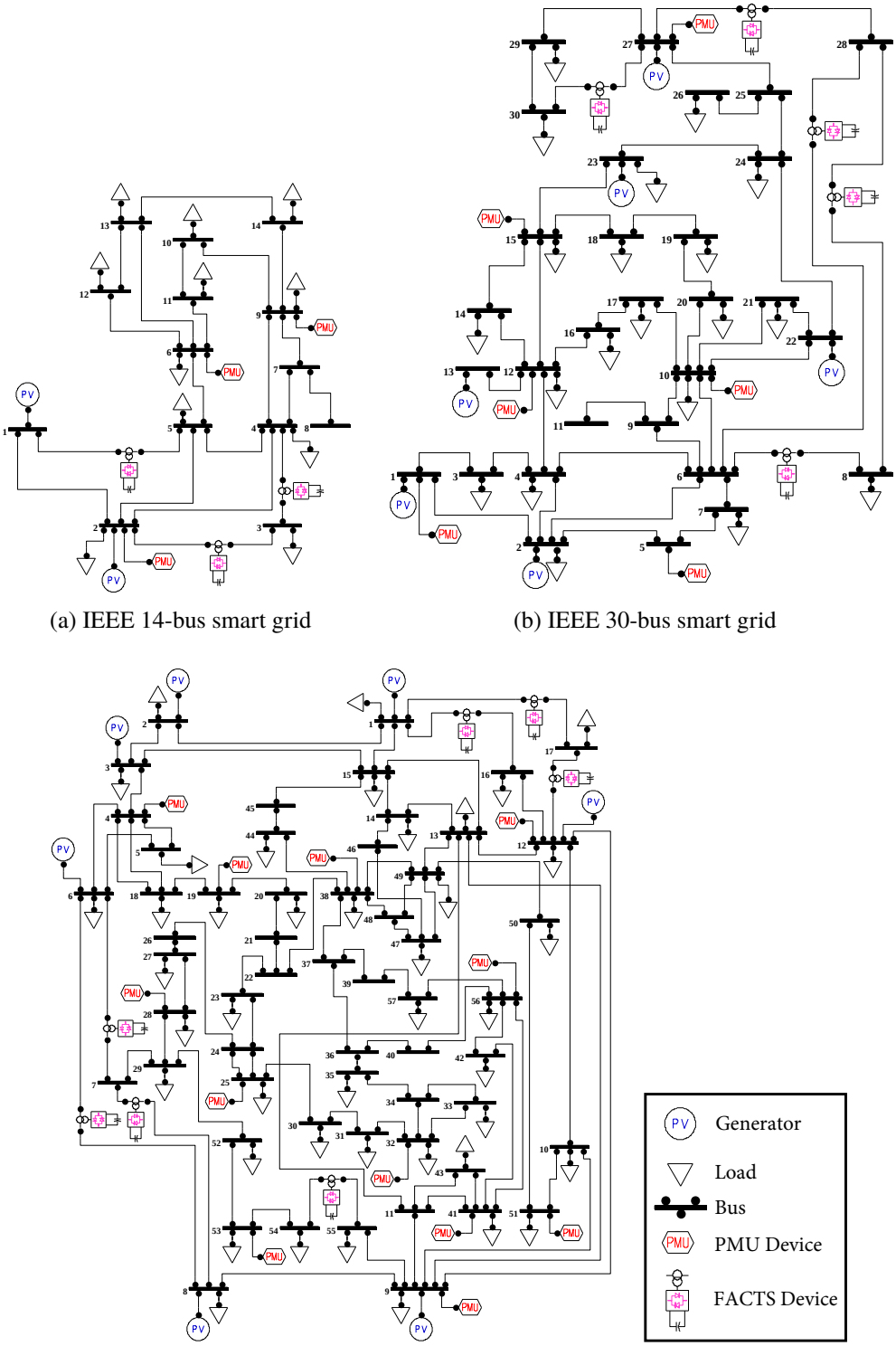
If the times of the component failures are not available, another importance analysis approach looks at the fragility of a component without incorporate survivability. The *fragility* of component i, denoted as β_i, is the fraction of observed or simulated failure cases in which the component has failed. The fragility of component i can be determined as shown in Equation (2.18), where m is the total number of failure cases.

$$\beta_i = \frac{|\mathcal{S}_i|}{m} \tag{2.18}$$

Fragility or criticality can be used to determine the priority of a component for hardening efforts.

2.4.4. Case Study: IEEE Smart Grid Test Systems. In this section, the proposed approach is demonstrated by applying it to a number of smart grids based on test systems commonly used in power engineering literature. Specifically, survivability evaluation is demonstrated for smart grids based on the IEEE 14-, IEEE 30-, and IEEE 57-bus test systems [79]. The IEEE 14-bus system example has been included in the interest of providing a concise and clear example. The two larger systems demonstrate the scalability of the method. All three systems are depicted in Figure 2.10.

The proposed approach to survivability evaluation is intended to be holistic and applicable to cyber, physical, and cyber-physical systems. To demonstrate this for a cyber-physical system, the classic IEEE test systems were supplemented with cyber infrastructure



(a) IEEE 14-bus smart grid

(b) IEEE 30-bus smart grid

(c) IEEE 57-bus smart grid

Figure 2.10. Single line diagrams of IEEE smart grid test systems.

in order to create corresponding smart grids. The cyber infrastructure is comprised of phasor measurement units (PMUs), which record and communicate GPS time-synchronized dynamic power system data; flexible AC transmission system (FACTS) devices, which adjust the flow of power in the transmission lines; and a decision support algorithm that determines optimal settings for the FACTS devices based on data from the PMUs. The placement of the PMUs and FACTS devices on each smart grid (shown in Figure 2.10) was determined using the methods presented in [80] and [81]. The decision support algorithm used was a neural network trained with $N - 1$ contingencies.

2.4.4.1. Selection of figure-of-merit. The *essential service* expected of a smart grid is the provision of stable power to customers. Two FoMs are used to quantify this provisioning: the customer service index and the average nominal voltage error. The *customer service index* (CSI) reflects the fraction of customers who have received this essential service, with a binary view — a customer is either served with adequate power or has not been served at all. In accordance with standards such as EN-50160 [82], the determination of whether a customer has been served is based on whether the voltage of the bus to which the customer is connected is within a predetermined range. For example, EN-50160 specifies a range of 0.9 to 1.1 per unit, where the per-unit representation denotes normalization by a base value, in this case, a base voltage. Equation (2.19) articulates the calculation of CSI.

$$\text{CSI} = \frac{\text{Number of customers served}}{\text{Total number of customers}} \tag{2.19}$$

The second FoM utilized for the evaluation of smart grid survivability is the *average nominal voltage error* (ANVE), which calculates the average voltage error over all load buses that experience undervoltage or overvoltage and subtracts that value from 1, where an ANVE of 1 indicates full service. Equation (2.20) articulates the calculation of ANVE. In contrast to CSI, which solely reflects blackouts, ANVE considers brownouts.

$$\text{ANVE} = 1 - \frac{\sum_i |\text{Rated voltage at bus } i - \text{actual voltage at bus } i|}{\text{Total number of customers}} \tag{2.20}$$

2.4.4.2. Selection of failure cases. As power grids are typically highly reliable and robust networks, most evaluations rely on $N-1$ or $N-2$ contingency analyses (i.e., a single failure or two concurrent failures). In this work, the consequences of an outage of a transmission line or a power regulator (FACTS device) in the presence of a fault in the cyber network are analyzed. The cyber faults injected to the smart grid are manifestations of data corruption: (i) incorrect data from PMUs, (ii) incorrect commands generated by the decision support algorithm, and (iii) undetected errors in the communications. Note that any one of these cyber faults alone can be tolerated by the system, however, if they are accompanied by an outage of a transmission line or failure of a power regulator, further propagation of the failures is likely. Table 2.3 lists the simulated failure cases and the number of simulations carried out for each case.

Table 2.3. Summary of faults injected.

	IEEE-14	IEEE-30	IEEE-57
single transmission lines	20	41	80
FACTS devices	3	5	7
number of hardware faults simulated	23	46	87
PMU devices	3	6	12
communication links	6	11	19
control units	1	1	1
number of cyber faults simulated	10	18	32
total number of simulation runs	**230**	**828**	**2,784**

For this study, failure cases were selected based on the failure rate of components. Failures of transmission lines and FACTS devices were selected because they have a relatively high rate of failure and are a major source of power outages [83, 84]. Additionally,

failures of PMUs, communication links, and decision support algorithms are included because their failure would impact the physical components of the system as well.

2.4.4.3. Simulation technique. A smart grid simulator capable of fault injection is utilized to observe the CSI and ANVE during each failure case. For the electric delivery system, there are a number of commercial and non-commercial computer simulation tools available. The PowerWorld Simulator [85] is a popular commercial tool for analysis of high voltage power systems as it supports common protection and control devices, provides an interactive environment and intuitive GUI, and is able to solve power flow equations for very large systems. However, PowerWorld does not provide the transparency needed for analysis of the sequence of failures. DIgSILENT [86] is another well-known professional tool that has the same shortcoming. Among the non-commercial packages, MATPOWER [87] and PSAT [88] are two MATLAB-based toolboxes for Windows machines. MATPOWER can solve load flow and optimal power flow problems in a command line interface. PSAT has a graphical interface and supports power regulators and basic monitoring and protection devices in addition to the capabilities of MATPOWER. In this work, PSAT is used for the simulation of the three IEEE bus systems. For the purpose of fault injection simulations, PSAT was enhanced in order to achieve the high resolution required for the analysis of smart grids. These enhancements include incorporating wide-area measurement capabilities by PMU devices, providing a platform for implementing a decision support algorithm, and integrating the power systems with communication technologies that are used in smart grid applications. This modified version of PSAT is interfaced with a MATLAB wrapper that acts as an adapter between libraries and orchestrates subroutine calls.

This simulation environment is used to determine power flows and voltages in the network during the failure cases. Figure 2.11 illustrates the procedure followed for simulating each failure case. In each outer loop, a data file that contains the topology of the system under test is loaded and a failure case (with index j) is executed at time t_e by injecting the corresponding faults and/or failures. In the inner loop, at each time step, PSAT

performs power flow analysis and determines active power flow on each line and voltage at each bus. PMU devices then measure phasor data (including active power and voltage) of corresponding lines and buses and send that data to the decision support algorithm where new settings for FACTS devices are calculated. Updated settings will regulate active and reactive power flow in the lines where FACTS devices are installed. At this point, PSAT power flow analysis is run once more to find the updated active power flows and bus voltages. In every iteration of the inner loop after instant t_e, active power flow of the lines are compared to their capacity. If any line is overloaded, it is considered failed, and the topology is updated accordingly.

The simulation continues until no further failures are detected. For the sake of consistency among the three IEEE bus systems and for the ease of comparing the plots, all simulations are continued for 25 time steps (denoted as t_{final} in Figure 2.11). However, all failure sequences terminate before the 25^{th} time step.

Note that since the time is discrete and is determined by the software simulation tool, the rate of degradation is limited to a maximum value. Additionally, minor changes in the rate of degradation due to time-specific variations may not be captured in the simulation environment.

2.4.4.4. Simulation results. Figure 2.12 depicts the simulation results for each of the three test systems, using CSI and ANVE as the FoMs. In Figure 2.12, each sub-figure depicts the change in one FoM over time after the injection of a failure. The intensity of the line indicates the number of failure cases in which the FoM demonstrated this behavior. Note that since CSI is discrete, it can only hold a finite set of values between 0 and 1.

These results indicate that the majority of the simulated failure cases result in minimal degradation, as the test systems are relatively robust. In the IEEE-14 smart grid results, shown in Figure 2.12a and 2.12b, a number of failures lead to total system failure with no customers served and a maximum error for CSI and ANVE, respectively (this is indicated by the FoMs reaching 0). Additionally, the results indicate that the initial

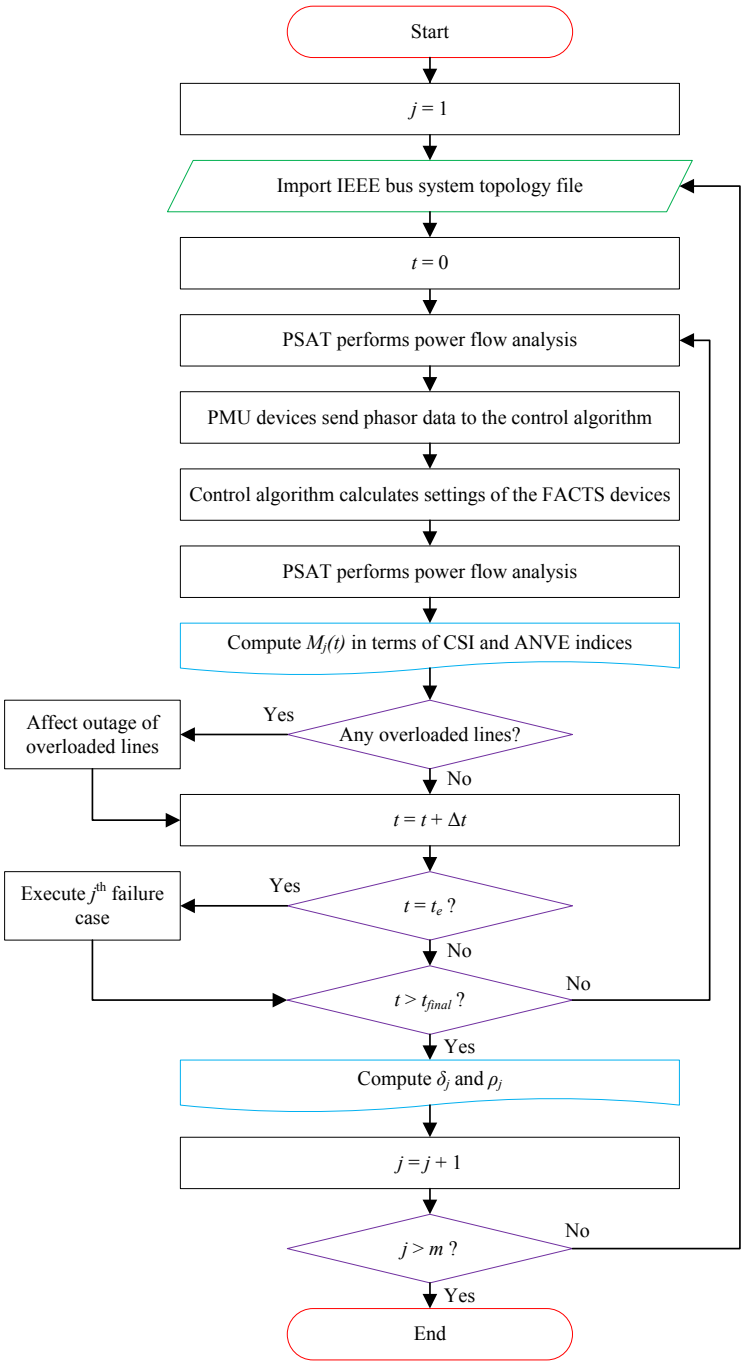
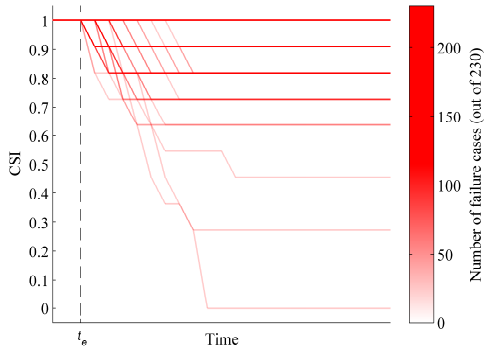
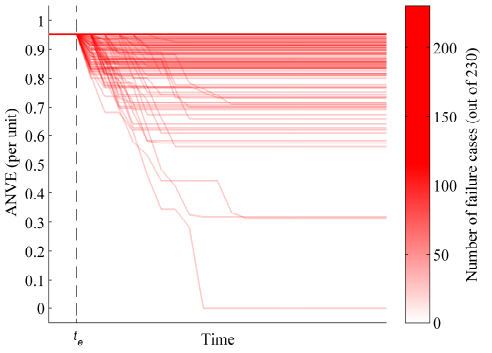


Figure 2.11. Survivability evaluation procedure.

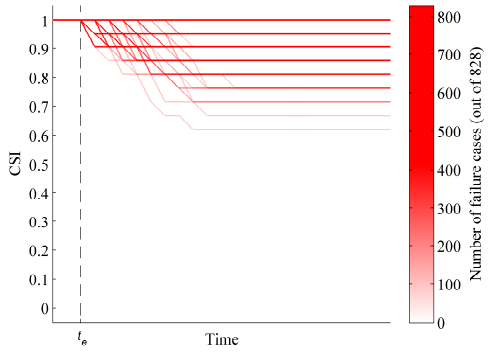
degradation of both CSI and ANVE for a number of failure cases is delayed by multiple time steps. Lastly, the results show that a number of failure cases have two phases of system degradation separated by a brief period of stabilization. The IEEE-30 and -57 smart grids



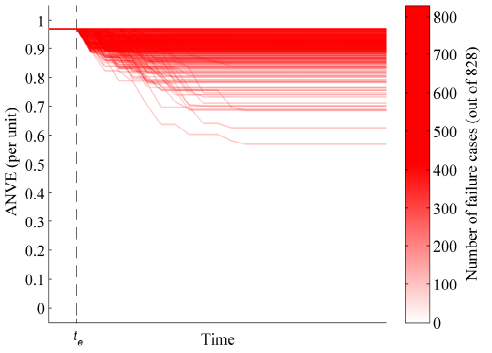
(a) CSI vs. time for IEEE-14



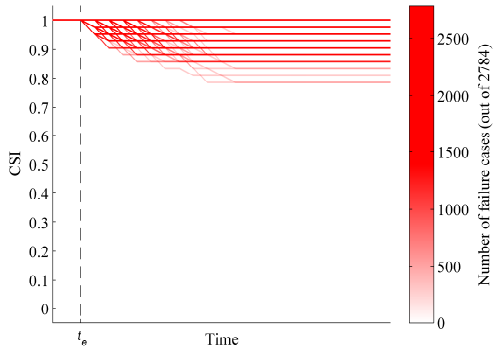
(b) ANVE vs. time for IEEE-14



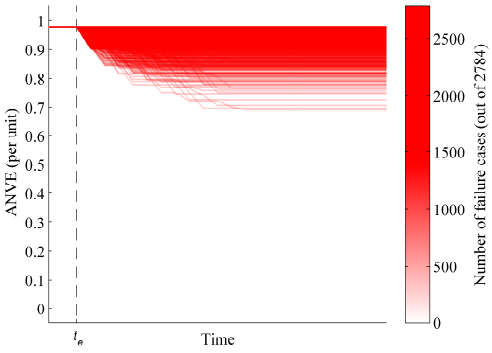
(c) CSI vs. time for IEEE-30



(d) ANVE vs. time for IEEE-30



(e) CSI vs. time for IEEE-57



(f) ANVE vs. time for IEEE-57

Figure 2.12. CSI and ANVE vs. time. The desired outcome is a value of 1 characterized by all customers being served and no voltage error for all customers, respectively, for CSI and ANVE.

incorporate more redundancy and can tolerate a greater number of failures, this is seen in Figure 2.12c, 2.12d, 2.12e, and 2.12f, where the FoMs never reach 0.

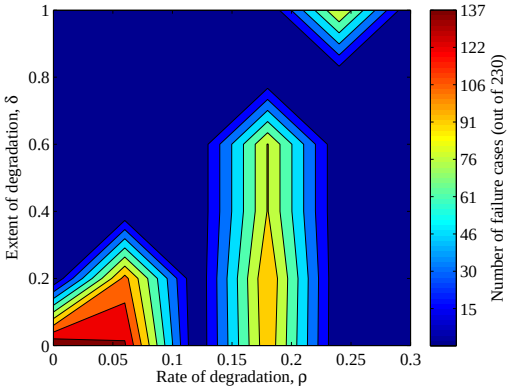
2.4.4.5. Survivable behavior evaluation. The maximum rate and extent of degradation were extracted from the log of each failure case. Figure 2.13 depicts a two-dimensional histogram of CSI and ANVE for each smart grid simulated.

In an ideal system, every one of these histograms would be dense near the origin and sparse elsewhere, reflecting slow and minimal degradation in response to failure. This expectation is realized for the IEEE smart grids evaluated. However, there are clusters of failure cases with higher rates and extents of failure which fall in the upper and/or right regions of the histogram. The presence of these clusters indicates that many of the failure cases have similar values of rates and extents of degradation. This is most likely caused by similar failure propagation paths through the power grid (i.e., different cascading failures involve the same vulnerable components).

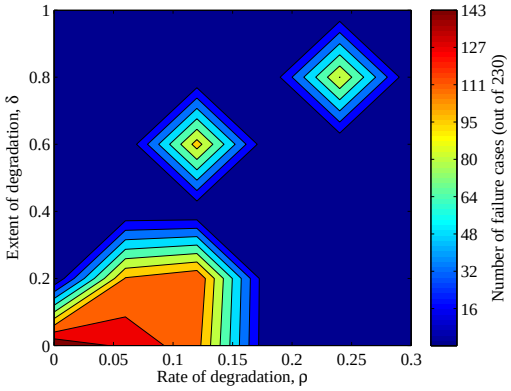
2.4.4.6. Importance analysis. The importance analysis technique is used to identify survivability bottlenecks and guide investments toward fortifying these systems. Criticality and fragility can be determined for each component of a grid, as described in Section 2.4.3.

Table 2.4. Transmission lines of IEEE 57-bus system with highest fragility and criticality. Only the top ten lines are shown.

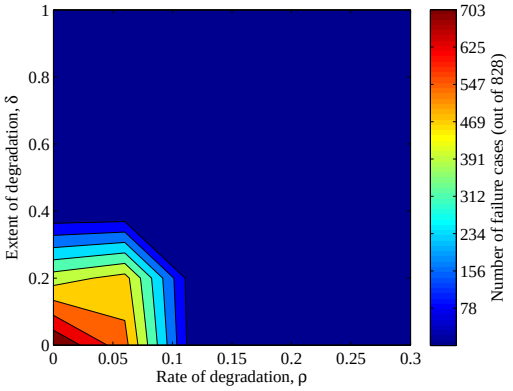
Rank	**Line**	**Fragility** ($\times 10$)	**Rank**	**Line**	**Criticality** ($\times 10^2$)
1	l_{4-18}	0.2801	1	l_{8-9}	0.1477
2	l_{1-2}	0.2729	2	l_{4-18}	0.1417
3	l_{3-4}	0.2514	3	l_{3-4}	0.1337
4	l_{1-15}	0.2370	4	l_{6-7}	0.1277
5	l_{1-17}	0.2370	5	l_{4-6}	0.1137
6	l_{4-6}	0.2227	6	l_{1-2}	0.1078
7	l_{8-9}	0.2227	7	l_{1-15}	0.1058
8	l_{1-16}	0.2227	8	l_{13-15}	0.0718
9	l_{6-7}	0.2155	9	l_{1-16}	0.0659
10	l_{2-3}	0.1939	10	l_{2-3}	0.0639



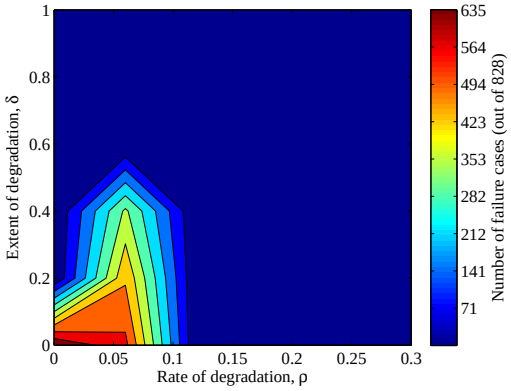
(a) CSI degradation rate vs. extent histogram for IEEE-14



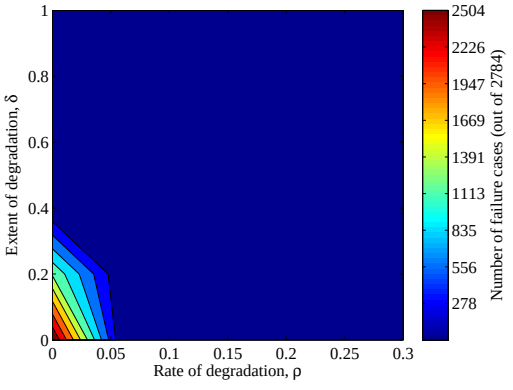
(b) ANVE degradation rate vs. extent histogram for IEEE-14



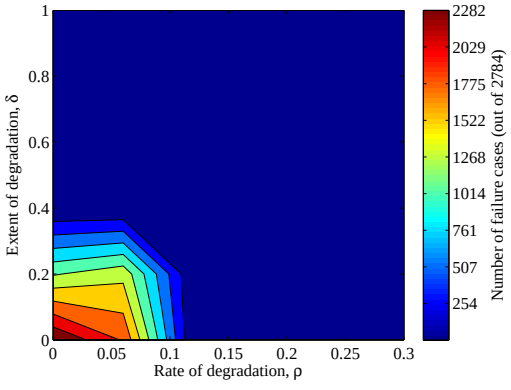
(c) CSI degradation rate vs. extent histogram for IEEE-30



(d) ANVE degradation rate vs. extent histogram for IEEE-30



(e) CSI degradation rate vs. extent histogram for IEEE-57



(f) ANVE degradation rate vs. extent histogram for IEEE-57

Figure 2.13. CSI and ANVE degradation rate vs. extent histogram. Color indicates the number of failure cases which resulted in the corresponding rate and extent of degradation. The desired outcome for both CSI and ANVE is a single cluster near the origin characterized by slow and minimal degradation of the FoM.

Table 2.4 shows the rankings of the top ten lines of IEEE 57-bus system using fragility and criticality as criteria for hardening prioritization. It can be seen that some lines have similar ranking in both (e.g., lines l_{4-18}, l_{3-4}, and l_{4-6}). However, a few lines have much higher priority when using criticality as the metric, such as lines l_{8-9} and l_{6-7}. These lines fail in fewer failure cases but have a very high impact on the system FoM when they do fail. Alternatively, a few lines have a much lower priority using criticality as a metric, such as lines l_{1-2} and l_{1-15}. These lines fail as a result of another more important line failing, but their failure is relatively insignificant in terms of system survivability.

2.4.4.7. Validation of approach. In this section, importance analysis technique is validated through the targeted hardening of an IEEE 57-bus smart grid. To harden the smart grid system, the five lines with highest priority metrics were fortified by increasing their power flow capacity by 50%, which is expected to increase the survivability of the system because it increases fault tolerance. The same hardening effect could have been achieved by adding redundant lines; however, this was avoided in order to maintain the topology of the system for ease of comparison. Once the system was hardened, the survivability analysis was rerun to compare the results with the original system.

First, fragility was used to select components for hardening. Lines l_{4-18}, l_{1-2}, l_{3-4}, l_{1-15}, and l_{1-17}, highlighted in yellow in Figure 2.14, were hardened. Next, criticality was used to select components for hardening. Lines l_{8-9}, l_{4-18}, l_{3-4}, l_{6-7}, and l_{4-6}, highlighted in blue in Figure 2.14, were hardened. Results of simulations are compared for original and hardened versions of IEEE-57 in Figure 2.15. The survivability results shown in Figure 2.16 verify the effectiveness of the hardening technique.

Comparing the survivability evaluation results of the two hardened and original IEEE 57-bus smart grids demonstrates an improvement in the survivable behavior of the system. Both importance analysis techniques resulted in an improvement in system survivability evident as is in Figure 2.15. However, using criticality as the metric leads to a more effective improvement over the original system.

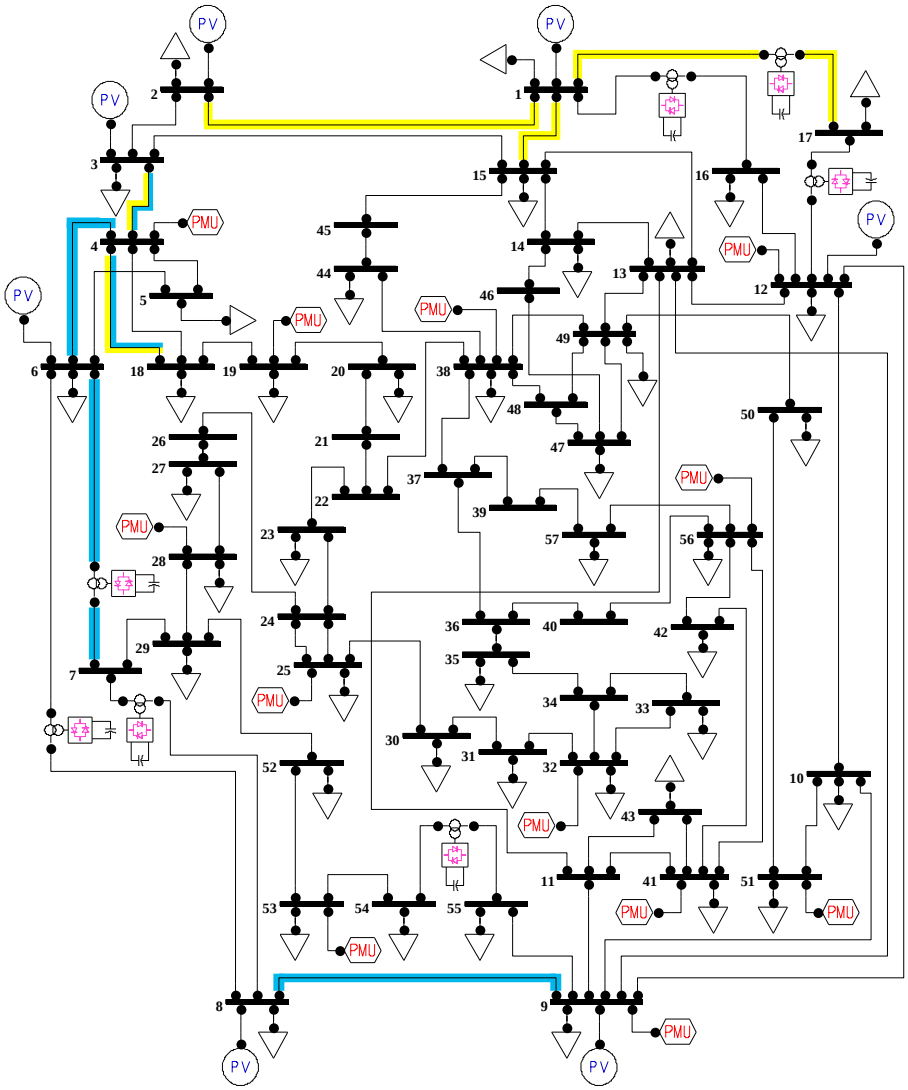
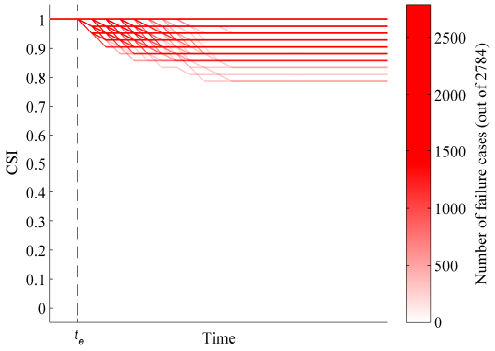
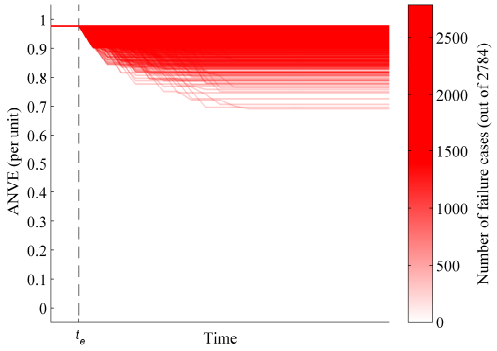


Figure 2.14. IEEE 57-bus smart grid test system. Lines highlighted in yellow have the highest fragility and those highlighted in blue have the highest criticality.

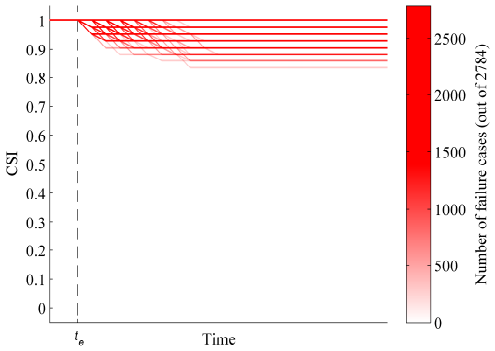
2.4.5. Summary of Research Contribution. The research contribution presented in Section 2.4 is an approach for evaluation of survivability for CPSs with arbitrary, fixed, and known topologies. This approach evaluates the survivability of a system by extracting the rate and extent of degradation of a domain-specific FoM during the observation or simulation of multiple failure cases. The results of the observation or simulation are used in importance analysis to identify critical components whose hardening would be most



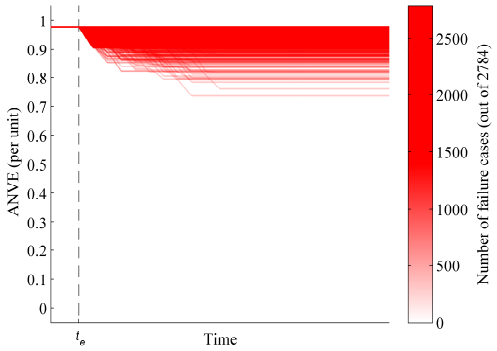
(a) CSI vs. time for original IEEE-57



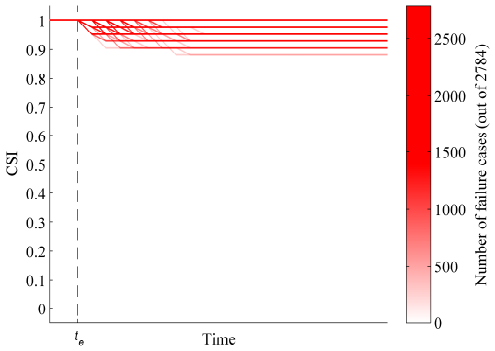
(b) ANVE vs. time for original IEEE-57



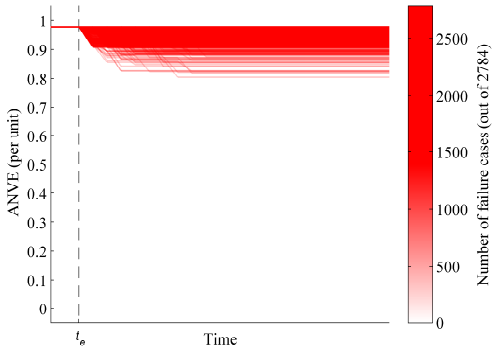
(c) CSI vs. time for IEEE-57 hardened based on fragility



(d) ANVE vs. time for IEEE-57 hardened based on fragility

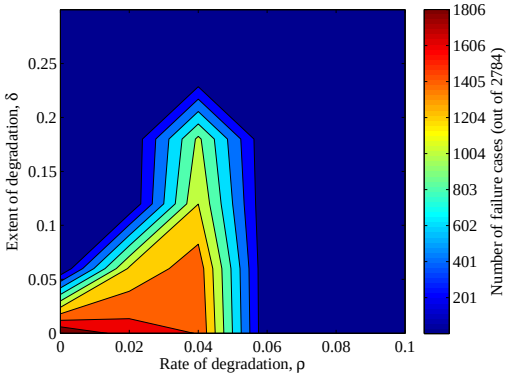


(e) CSI vs. time for IEEE-57 hardened based on criticality

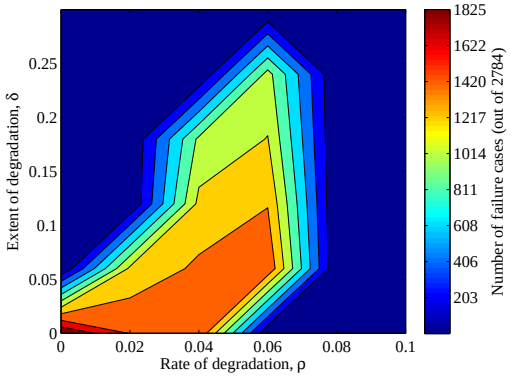


(f) ANVE vs. time for IEEE-57 hardened based on criticality

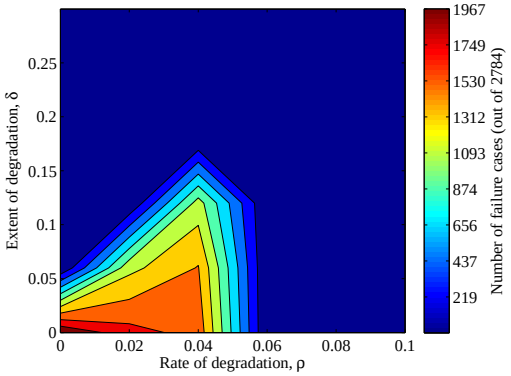
Figure 2.15. CSI and ANVE vs. time for hardened systems. Comparison of the FoM graphs for original IEEE-57, the IEEE-57 hardened based on fragility, and the IEEE-57 hardened based on criticality. Extent of degradations has reduced for the hardened systems.



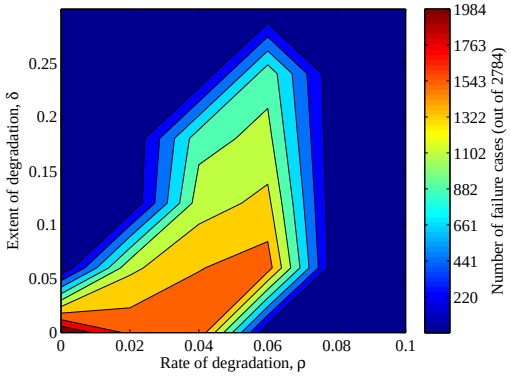
(a) CSI degradation rate vs. extent histogram for original IEEE-57



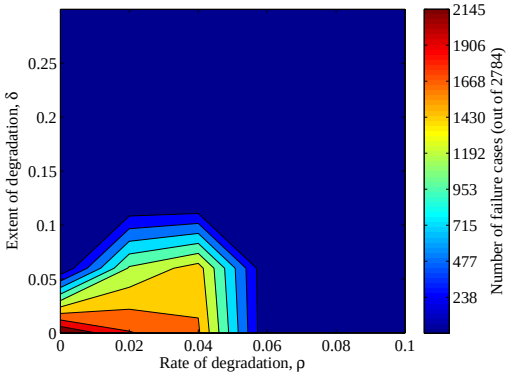
(b) ANVE degradation rate vs. extent histogram for original IEEE-57



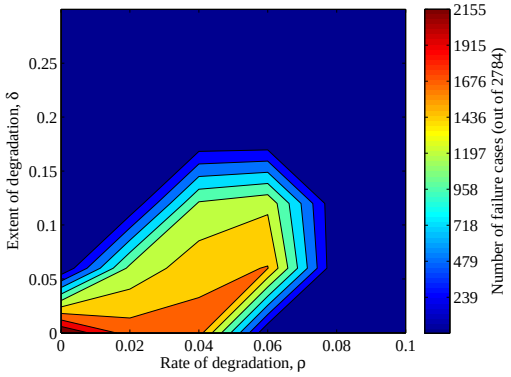
(c) CSI degradation rate vs. extent histogram for IEEE-57 hardened based on fragility



(d) ANVE degradation rate vs. extent histogram for IEEE-57 hardened based on fragility



(e) CSI degradation rate vs. extent histogram for IEEE-57 hardened based on criticality



(f) ANVE degradation rate vs. extent histogram for IEEE-57 hardened based on criticality

Figure 2.16. CSI and ANVE degradation rate vs. extent histogram. Comparison of the CSI and ANVE degradation rate vs. extent histograms for original IEEE-57, the IEEE-57 hardened based on fragility, and the IEEE-57 hardened based on criticality. Clusters of degradation points have moved towards the origin for the hardened systems.

beneficial to survivability. This technique was demonstrated using three smart grids based on conventional IEEE power test systems.

Future work will incorporate the simultaneous use of multiple FoMs to create a multi-dimensional FoM utilizing Pareto optimality. Additionally, the scalability of this approach will be improved by examining a more strategic selection of failure cases using superposition for evaluating systems with independent components.

3. MODELING AND EVALUATION OF CPS BEHAVIOR

The survivability evaluation approach described in Section 2.4 requires on observing system behavior after catastrophic events to determine if a system will continue to provide essential services. This observation can be conducted using a real system, a system simulator or using a system behavior model. There are a number of reasons a system behavior model would be preferred, including cost and time constraints. Behavior models are very domain specific and require an understanding of the various physical properties inherent to the type of CPS.

3.1. INTELLIGENT TRANSPORTATION AND AUTONOMOUS VEHICLES

Intelligent Transportation Systems (ITS) are CPSs aimed at improving performance and safety of transportation networks [89]. ITS refers to all modes of transport including road, rail, and air. All of these transportation systems have similar challenges [1]. However, road transportation systems are used as an example system in this work. All following references to ITS will refer to road transportation systems.

ITS technologies include everything from basic traffic management systems such as vehicle navigation and traffic signal control to more advanced systems that allow Vehicle to Everything (V2X) communication to improve control and information dissemination between vehicles, roadside units, infrastructure, pedestrians, and cyclists [90]. ITS technologies also include unmanned vehicle technologies including self-driving vehicles and automatic parking systems. The example ITS system used in this work focuses on Infrastructure to Vehicle (I2V), Vehicle to Vehicle (V2V), and Vehicle to Infrastructure (V2I) communication as well as information dissemination and the intelligent traffic control that communication facilitates [91].

These services can be classified based on where in the infrastructure the required computer processing occurs. Elements of ITS infrastructure can be classified as mobile infrastructure or static infrastructure.

3.1.1. Mobile Infrastructure. Mobile infrastructure consists of all ITS elements without a static network connection (i.e., vehicles). Vehicles on modern roadways range from classic cars with no digital systems to fully autonomous, unmanned vehicles. ITS systems must be designed to accommodate the full range of vehicles. Vehicles can be categorize based on their communication capability and level of automation.

Traditional vehicles are all vehicles without V2V or V2I communication capability. They do not provide data directly to the ITS. Vehicles in this category may or may not have I2V capabilities which would provide the driver with additional information about congestion, such as 2-way GPS traffic updates. This category also includes vehicles with adaptive cruises control or advanced collision avoidance systems such as blind spot sensors and backup sensors. While these technologies improve vehicle safety and control, they do not provide data to other components in an ITS.

Intelligent vehicles are vehicles with V2V or V2I communication capability that are controlled by a human driver. These vehicles are equipped with an onboard sensor suite with the capability to monitor the locations and actions of surrounding vehicles as well as detect road obstacles and conditions. These vehicles utilize on-board processing and storage systems to analyze collected data. Collected information is communicated to surrounding vehicles or the ITS infrastructure via roadside unit. The wireless communications capability falls into two categories based on the indented recipient. Short-range communication is used to communicate with neighboring vehicles and roadside units using the IEEE 802.11p protocol, which was specifically developed for ITS and mobile ad hoc or mesh networking. The second type of communication is longer range communications using IEEE 802.16, WiMAX, GSM, or 3G. This type of communication is used to communicate with a central traffic management center or to access other relevant data sources.

Unmanned vehicles are vehicles with the same capabilities as intelligent vehicles. However, the collected data is used to directly control the vehicle rather than to assist a human driver. In addition, collected data may be provided to other components of an ITS.

3.1.2. Static Infrastructure. Static infrastructure within ITS includes purely physical infrastructure including roads, highways, and bridges as well as the static, cyber-enhanced infrastructure. The static ITS infrastructure does not move during operation and includes devices such as traffic signals and road sensors.

The cyber layer of an ITS system is structured and functions similar to a sensor database architecture. Sensor database architectures are classified based on where the data is stored. These architectures range from traditional sensor databases, where data is stored in a centralized database, to distributed databases, where every sensor node has its own database.

Traditional sensor networks described by Akyildiz et al. [92] are not applicable to ITS due to large networking overhead and delay. Another sensor network architecture is the distributed sensor database system, which places databases closer to the controller and sensor nodes. This architecture can be thought of as a data logging network. In this type of sensor network, all sensors send all sensed data to secondary storage, which can be retrieved in bulk. This architecture permits duplication of stored data to improve performance. Distributed database architectures are not specific to sensor networks. Many approaches to distributed databases are summarized by Hurson et al. [93] including federated and multi-databases which address issues such as data distribution and transparency as well as query and transaction processing. A further distributed sensor network architecture is discussed by Bonnet et al. [94] is the sensor database model. In this architecture, each sensor node holds a database that can be dynamically queried. Tsiftes et al. [95] discuss this sensor network architecture and propose a database management system.

A practical ITS would use a combination of distributed and sensor database architectures at various hierarchical levels of the system. Combining these architectures may

improve performance by limiting the communication of raw data, energy, bandwidth, and scalability. Additionally, this architecture improves maintainability and fault recovery by storing performance data at the sensor nodes. Amadeo et al. [90] discuss the benefits of using a Named Data Networking model for ITS, which would require this type of database architecture. However, this architecture has challenges including the system updates and database management due to its distributed nature.

3.1.2.1. Road side units. A Road Side Unit (RSU) collects traffic data from a static sensing area along a road and transmits data to traffic control devices as well as a Central Traffic Management center. These devices also serve as an information source for intelligent vehicles to collect future traffic information [96].

RSU can sense traffic information using a number of methods. One method for collecting traffic information is the triangulation method. Triangulation uses mobile phones as anonymous traffic probes. The phones transmit presence announcement signals to the mobile phone network which can be observed by an RSU. This network data is collected and analyzed using triangulation and converted into traffic flow information. This method works for all types of vehicles, provided that a powered-on mobile phone is in the vehicle. Another method is vehicle re-identification. This method uses some unique identification from an in-vehicle device, such as Bluetooth MAC addresses or a RFID toll tags. As a vehicle travels along a route, multiple RSU detects a specific vehicle and record a time stamp. This information is shared and analyzed to determine speed, travel times, and traffic flow for a road segment. This method requires technology within the vehicle to transmit a unique id. Conveniently, most modern vehicles use wireless communication between components, which can be used to identify a vehicle. Lastly, V2I communication provided by intelligent vehicles can be used to collect traffic flow data. Many other techniques can be used to collect traffic flow data such as two-way GPS or satellite navigation systems, inductive loop detection, traffic video cameras, and audio detection.

RSU use information from multiple sources to create an accurate picture of traffic flow on a specific road segment by using data fusion based approaches to intelligently combine data. These data fusion techniques create a more accurate representation of the traffic than any single sensing method.

3.1.2.2. Traffic control. ITS allows for traffic control systems that are more advanced than traditional timed traffic signals [97]. One type of control device is intelligent traffic lights, which use traffic data collected at the local intersection, as well as future traffic information provided by RSUs, to create a dynamic time schedule to maximize the flow of traffic through an intersection. Another control system is variable speed limits. These systems work to minimize traffic density in congested areas by dynamically changing the speed limit of roads based on weather conditions, road conditions, or the presence of congestion areas. Lastly, dynamic lanes can be used to provide more inbound or outbound lanes depending on the flow of traffic as traffic in many metropolitan areas is not symmetric.

3.1.2.3. Central traffic management. A Central Traffic Management (CTM) system could be centralized or distributed over a control area. In either case, a CTM collects and analyzes data from intelligent vehicles, unmanned vehicles, and RSU to facilitate control decisions [98]. Each central traffic management office would have a server for data storage and processing. The processed data could be used for high-level coordination of the traffic control devices. The central office could then broadcast data back to vehicles to improve navigation and control.

3.2. MODELING AUTONOMOUS VEHICLE BEHAVIOR

The past decade has seen autonomous vehicles become the subject of considerable research and development activity with test vehicles already on the road. The majority of these advances have focused on individual vehicles, rather than the interactions that result when autonomous (unmanned) and conventional (manned) vehicles come together in an intelligent transportation system. The robustness of autonomous vehicles to contingencies

caused by unpredictable human behavior is a critical safety concern. Assuring the reliability, availability, security, and similar non-functional attributes of autonomous vehicles are just as critical. While many traffic models exist, no existing models incorporate manned and autonomous vehicles.

The research project proposed in this work centers on developing models capable of accurately representing environments where manned and unmanned vehicles coexist. This work proposes extending an established macroscopic transportation model to differentiate between manned and autonomous vehicles. Differentiating vehicles allows for the use of stochastic methods to reflect the non-determinism of the operating environment, especially as related to driver behavior. The goal of this research is to capture both basic operation of autonomous vehicles, as well as advanced capabilities such as platooning and robotic adaptation. The insights gained from these models will facilitate the design of intelligent transportation systems that are both safe and efficient. This work was published in [6].

3.2.1. Traffic Models. Traffic can be modeled at various levels of abstraction. The state of a traffic system is given by the number of vehicles present in a section of the transportation network at a given time. The most basic models are microscopic discrete-event models such as those in [99, 100, 101, 102, 103], which accurately describe traffic behavior at intersections or a single stretch of road or highway. When the roads are highly populated, these models suffer from state explosion, making analysis difficult. These models are useful for the design of individual intersections and roads and have been expanded to reflect human behavior.

Macroscopic models overcome this state expansion by disregarding individual vehicles. They use only three variables to describe local behavior: density, average speed, and flow rate [104]. Many macroscopic models are described in [105, 106, 107, 108, 109, 110, 111].

Many of these models utilize Petri nets for synchronous system evaluation. Petri nets represent a powerful modeling formalism that has been successfully used in different

application domains. A Petri net consists of places, transitions, arcs, and tokens. Arcs serve as connections between places and transitions and tokens represent some aspect of the system - in this case vehicles in a traffic system. Places hold the tokens until they are passed via an arc through a transition based on a set of firing rules. Many different types of Petri nets have been developed and tailored to model specific applications.

3.2.2. Traffic Model for Autonomous Vehicle Operation. This proposed traffic model for autonomous vehicle operation is built upon the model described in [104]. In the original model, the traffic system is modeled as a hybrid Petri net, with road sections modeled as continuous transitions and stop lights and intersections modeled as discrete transitions. Hybrid Petri nets allow for modeling both the continuous and discrete elements of a system while preventing the state space explosion that would result from a purely discrete model.

Roads are represented as a series of virtually-divided road sections that are described by the density $d(t)$ of cars at time t, their average speed $v(t)$, and the flow $f(t)$. The marking $m(t)$ of a place represents the number of cars present at time t, uniformly distributed along the length of the road section with an average speed $v(t)$. The modeled road sections have three different modes of operation, depending on the traffic conditions (i.e., the density of vehicles). If a section has low density, vehicles will travel at the free speed (free flow), where outflow increases proportionally to the density. When the density is higher, the average speed will decrease, but the outflow will remain constant (constant flow). And lastly, when the density is very high, the outflow decreases due to congestion.

The Continuous Petri net model of a single road section is shown in Figure 3.1. It has three places (p_1, p_2, p_3) and two transitions (t_{i-1}, t_i). The number of cars in a section is the marking of p_1. The flow of vehicles entering and leaving a section is dictated by t_{i-1} and t_i, respectively. Free-flow traffic is modeled by ignoring p_2 and p_3. Constant-flow traffic is modeled using p_3, which has a constant marking and imposes an upper bound on the flow of t_i. Lastly, when the density reaches the maximum, as a road section can hold

a finite number of vehicles, p_2 is used to ensure $m[p_1] + m[p_2]$ capacity of road section. The marking at p_2 represents the number of gaps in the section. To model a road, multiple sections are connected with transitions.

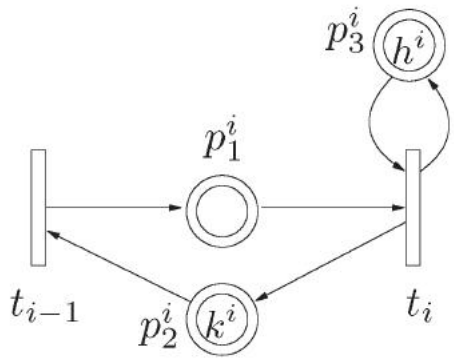


Figure 3.1. Single Road Section Model (from [104]).

Traffic lights are modeled as discrete events that can take one of three values: red, amber, or green. Each traffic light is modeled as a four-phase system, each represented by a place. The phases for an intersection of two roads R_1 and R_2 would be:

1. Phase 1: Green light for R_1 and Red light for R_2.
2. Phase 2: Amber to Red light for R_1 and Red light for R_2.
3. Phase 3: Red light for R_1 and Green light for R_2.
4. Phase 4: Red light for R_1 and Amber to Red light for R_2.

Phases 1 and 3 are when traffic is flowing on one of the two roads and phases 2 and 4 are the safety periods used to clear the intersection. This discrete Petri net has only one marking, so the system can be in only one state, with each phase being active when the corresponding place is marked. The road sections are joined to the intersection as follows. The flow through the intersection at any time is calculated by multiplying the flow of the continuous transition by the average velocity of the section. The velocity is dictated by the phase. The flow for R_1 during phase 1 is the same as the flow would be if there were no

traffic light. During phase 2, the flow decreases linearly to 0 and remains at 0 for phases 3 and 4.

Extending the Hybrid Petri net model developed by [104] to become a Colored Hybrid Petri net provides the mechanism to distinguish colors of tokens (i.e., types of vehicles). Distinguishing types of vehicles allows for transition firing rates dependent on the saturation of vehicle type at a location in the system. The extended model would be used to first build a city-level traffic model for analysis of traffic patterns as well as the study of failure in active transportation control systems. Specifically, the effect of intelligent traffic lights and dynamic speed limits on fault propagation from a single faulty controller across an urban traffic network will be studied.

This project was abandoned due to limitations in available data and validation. The pace of development in autonomous vehicles and the proprietary nature of the associated research prevented the continuation of this research. Without real traffic data from a system with manned and unmanned vehicles, validation will be impossible.

3.2.3. Summary of Research Contribution. The main contribution of the proposed model in Section 3.2 is a model for the behavior of manned and autonomous vehicles in an intelligent (urban) transportation system. The proposed model has the potential to contribute to better understanding the behavior of autonomous vehicles, which will be crucial in designing future transportation infrastructure systems that are both safe and efficient.

3.3. DATA CORRUPTION IN CPSS

The operation and behavior of CPSs are heavily dependent on the accuracy of the data processed by the control system. A functional control system which processes corrupted data will produce incorrect control settings. Data corruption is the unintended change to data that occur during writing, reading, storage, transmission, or processing. It can be created within a system through unintentional means, such as failures in sensors, processors, storage, or communication hardware, or through intentional means such as an

attack. Data corruption can manifest as missing or erroneous information. Both types of data corrupting can have negative effects on CPS operation and performance, however, these effects can be minimized by using a data cleansing process. Data cleansing is the process of detecting and mitigating corrupted data to ensure proper system operation.

Depending on the architecture of the cyber infrastructure, a CPS may have one or multiple control entities which received data created by sensors throughout the network. Additionally, data is exchanged between control entities in distributed control systems. This creates the potential for corrupted data to enter the control system and propagate to other control entities. Understanding the extent to which the corruption can propagate is essential designing fault tolerant and reliable systems.

The following sections provide an overview of the sources of data corruption, as well as corruption detection and mitigation techniques suitable for CPSs. These key topics are essential to understanding how undetected corrupted data can propagate through a CPS and are essential to designing robust CPS. This work was published in [8] and extended in [9].

3.3.1. Sources of Corrupted Data. An understanding of fault tolerance and dependability is necessary in order to discuss data corruption. Avižienis et al. [13] define failure, error, and fault to describe the state of a system in the presence of a disruptive event based on the system's ability to provide its specified service. A system *failure* occurs when the system does not comply with its specifications. An *error* is a system state that may cause a subsequent failure (i.e., a failure occurs when an error alters a service). And lastly, a *fault* is the adjudged cause of an error. Faults can be classified based on a number of factors, including persistence, activity, and intent. Therefore, data corruption can be classified as a failure, error, or fault, depending on its location in the system. Creation of corrupted data by a sensor is a failure; a system processing corrupted data is a system error; receipt of corrupted data as a system input is a fault. Therefore, in a networked system, corrupted data can be a classified as a fault, error, or failure.

Corrupted data can be created within a system in a number of different ways, both deliberate and non-deliberate. Deliberate data corruption is the result of an attack. Attacks can be classified as cyber, physical, or cyber-physical [112]. All three types of attacks have the potential to result in data corruption. As an example, consider a smart meter utilized in the smart grid. A cyber attack such as altering the software of a smart meter could cause the sensor to report erroneous consumption information. Alternatively, a physical attack such as physically bypassing the smart meter would result in the sensor not reading the correct consumption information and result in incorrect information reported. Similarly, a cyber-physical attack such as combining the above cyber and physical attacks could be employed to disguise the presence of meter bypassing by altering the smart meter's software.

Non-deliberate data corruption is the result of corruption during communication, processing, or storage, or could be due to inaccurate sensor readings [113]. Erroneous sensor readings can be caused by quantizing errors reading a noisy signal. Other errors can be introduced by external conditions or sensor aging. Re-calibrating a sensor can reduce these errors, but cannot prevent them. Additionally, data may be incomplete, due to periodic failures of sensors. Cebula et al. [114] provide a detailed taxonomy of related cyber and physical risks.

3.3.2. Detection of Corrupted Data. As stated in Section 3.3.1, corrupted data can be produced by a number of sources including miscalibrated or faulty sensor hardware and errors in processing, storage, and communication. In many large distributed systems the cause or source of corrupted data is difficult to determine, however, the same data cleansing techniques may be used regardless of the source of the data error. Corrupted data can be detected by locating anomalies in the system. Rajasegarar et al. [115] discuss the importance and challenges of anomaly detection in sensor networks as it pertains to fault diagnosis, intrusion detection, and monitoring applications. The main challenge in anomaly detection algorithm development is that sensor networks are highly application and domain dependent. Two domain specific techniques are proposed by Yin et al. [116] who

model wind turbine data and Freeman et al. [117] who model aircraft pilot-static probe data. Both of these examples propose anomaly detection techniques for data with a nonlinear and unknown distribution and significant measurement noise. However, these techniques are not suitable for other domains and do not scale to CPSs. Another challenge in anomaly detection for CPS is sensor node storage and processing limitations. Anomaly detection that does not hinder normal operation must be employed. Corrupted data is detected and mitigated while the data is still viable with minimal energy consumption.

3.3.2.1. Statistical detection. Corrupted data can be detected by locating data anomalies or statistical irregularities in the data. While faulty sensors typically report easily distinguishable extreme or unrealistic values, not all data anomalies are the result of data corruption. Extreme environmental variations can produce data anomalies that must be distinguished from corruption.

Jurdak et al. [118] classify data anomalies into three broad categories: temporal, spatial, and spatiotemporal. Temporal data anomalies are local to one node and can be detected by observing sensor values over time that have one of the following attributes: high variability in subsequent sensor readings, lack of change in sensor readings, gradual reading skews, or out-of-bound readings. Examples of failures that result in this type of anomaly are as follows. A sensor may fail into a locked state or fail to obtain new samples making the sensor reading remain the same over long periods of time. As a sensor loses calibration, its data values drift away from the true value. A major malfunction of the sensor could produce out-of-bound readings that are physically not possible. And lastly, high variability in sensor readings could arise from sensor voltage fluctuations but could also signify major changes in the sensed environment. The detection process requires the data stream from a single node as well as stored historical data. The process can be conducted locally at the node, provided the node is cable of storage and processing, or by a centralized process on either a sink node or a base station.

Spatial data anomalies occur when one sensor's data readings are significantly different from surrounding nodes' readings. Detecting this type of anomaly requires a network-aware algorithm and is thus usually performed by a sink node or base station. Data redundancy between sensors is exploited to determine which sensors may have faulty readings. This type of detection is only possible for certain types of data with low spatial variation, such as air temperature or humidity. In this type of data, a change in one area will affect the surrounding sensors' readings. Networks with high spatial variation, especially video and audio data, are usually incapable of detecting such anomalies.

Spatio-temporal anomalies combine attributes of both temporal and spatial anomalies. These anomalies are somewhat rare but also more difficult to detect. For example, a storm progressively moving through an area causing sensor nodes to fail would be a spatiotemporal anomaly. As with spatial anomalies, spatiotemporal anomalies require a network-wide detection algorithm.

A variety of techniques are employed to detect each of these classes of data anomaly. Statistical approaches assume or estimate some statistical distribution model which captures the distribution of the data and detect anomalies by checking how well the data fits the model. Statistical approaches can be further classified as rule-based, estimation-based, or learning-based. Zhang et al. [119], Chandola et al. [120] and Fang et al. [121] provide comprehensive overviews of statistical anomaly detection techniques. Below is a summary of these approaches and recent advances in anomaly detection and a discussion of their applicability to CPSs.

Rule-based statistical approaches are the simplest form of anomaly detection. An acceptable lower and upper limit for the data is set and any value outside of this range is an anomaly. This technique requires only the definition of an outlier to be set, making it inflexible and resulting in many false positives or undetected anomalies if the tolerance is set too low or high. The benefits of this technique are that it is fast, requires no additional storage capability, and can be implemented in few lines of code making it ideal for sensor

nodes. Another simple rule-based statistical approach to anomaly detection is statistical inference using the mean and variance of a data set. Ngai et al. [122] use a chi-square test performed over a sliding window. In this example, the system determines that at least one value in the sliding window is anomalous if the chi-square value falls outside of some range specified by the user. The acceptable level must be configured prior to operation. This node-local approach can detect temporal type anomalies of a single sensor while imposing no additional network overhead. However, each sensor will require more storage, depending on the window size, and processing power to carry out the statistical analysis. Statistical inference techniques cannot adapt to changing ranges, which are very common in long-term wireless sensor network installations. Panda et al. [123] propose another very simple rule-based statistical anomaly detection method which calculates the mean and variance of a set of neighboring sensors to determine if a sensor is faulty. This approach can detect spatial anomalies in a set of neighboring sensors. Rule-based statistical methods can be implemented on minimal hardware and detect anomalies very quickly provided the data is well behaved and the rules are set appropriately. As such, other approaches have been developed that do not rely on user-set parameters.

Estimation-based statistical approaches use probability distribution models of the data to detect anomalous values. Probability distribution models can be parametric or non-parametric based [119]. Parametric models assume knowledge of the data distribution (i.e., Gaussian-based model). Non-parametric models such as histograms and kernel density estimators, do not assume knowledge of the data distribution. Histogram models estimate the probability of data occurrences by counting the frequency of occurrence and detect anomalies by comparing the new data with each of the categories in the histogram. Kernel density estimators estimate the probability distribution function (pdf) for some normal data. An anomaly is detected if new data lies in the low probability region of the pdf. Fang et al. [124] propose an energy efficient detection method using an ARIMA model. The ARIMA model is a statistical model used time series analysis. It has three terms, auto-regression

(AR), integration (I), and moving average (MA) to represent the data. The auto-regression term compares the new value to historical data using linear regression. The integration term differences the original data series to make the process stationary. And the moving average term captures the influence of extreme values. Each sensor node maintains a matrix of all maximum and minimum differences between itself and its neighbors. Then, using a voting mechanism, values are marked as valid or erroneous. Estimation-based approaches are mathematically proven to detect anomalies if a correct probability distribution model is used. However, knowledge of the probability distribution is not available in many real-world applications, making non-parametric approaches more useful. However, these approaches require additional hardware and storage but execute very quickly to detect anomalies.

Learning-based statistical approaches utilize data mining clustering and classification algorithms to group data with similar behaviors [121]. An anomaly is detected when data does not belong to a group. These techniques have very high detection rates but require additional processing and storage hardware.

A decentralized clustering approach to anomaly detection is set forth by Rajasegarar et al. [125]. This approach was designed specifically for hierarchical (tree-based) networks. Leaf nodes take sensor readings and cluster them into fixed-width clusters. Each non-leaf node in the tree takes clusters from its children and merges them together. Anomaly detection is performed at the root node by finding clusters that are further away from other clusters by more than one standard deviation above the average cluster distance. Chang et al. [126] use an Echo State Network (ESN), a neural network in which all neurons are connected to each other, to perform anomaly detection. The ESNs are trained before the nodes are deployed, so they are not very flexible. They operate in a similar fashion to Bayesian networks where the sensor's value is compared to the value predicted by the ESN. The advantage of using a neural net, in this case, is that it has much lower CPU and RAM requirements than a Bayesian network. An improvement to this approach is put forth by Obst [127]. Instead of building recurrent neural networks beforehand, each node

communicates with its immediate neighbors to build a model of its sensors' values. This model is then used to estimate anomalies in the readings.

Classification approaches use a learned model to organize data into a class; in this case, normal or anomalous. One classification approach uses Bayesian networks to model sensor values and predict when values are anomalous. [121] Mayfield et al. [128] have developed a tool called ERACER that uses Relational Dependency Networks to correct anomalous data and fill in missing data. The tool runs on the base station and develops linear models of sensor data, taking into account readings from other sensors at that node and readings from neighbor nodes. Another example of Bayesian networks is [129], where the concentration of various gasses in a mine's atmosphere is monitored. The network models sensor values over time as well as physical relationships between sensors. The system learns a baseline for the mine's concentrations that adapts to the natural fluctuations in gas concentration. It can detect both single-sensor anomalies and multi-node anomalies and events.

Ni et al. [130] propose using a hierarchical Bayesian space-time model to detect trustworthy sensors. The disadvantage of this technique is the amount of work required to set up the model. This technique results in excellent anomaly detection if model accurately represents the data. However, as with all models, if the model is poorly matched to the data the system performance degrades. A more advanced classification approach is the nearest neighbor approach. This approach uses a distance metric, for example, Euclidean distance, to determine how similar a value is to its neighbors. An anomaly is detected if the distance between neighbors is more than a user specified threshold. Expanding on this approach, Branch et al. [131] use a distributed algorithm to detect outliers as data propagates through the sensor network. In this approach, each node maintains a set of outlier data points from itself and its neighbors. A ranking function is used to map data values to non-negative real numbers which indicate the degree to which the data value can be regarded as an outlier with respect to the dataset. Nodes transmit data they suspect will cause the outlier set of

their neighbors to change. This is similar to a distributed k-nearest-neighbors classification approach. This technique is flexible with respect to the outlier definition, allowing for dynamic updating and in-network detection, reducing bandwidth and energy consumption.

A method to improve the performance of learning-based approaches uses principal component analysis (PCA) to reduce the dimensionality of a data. PCA is a technique that uses spectral decomposition to find normal behavior in a data set. PCA is used to reduce dimensionality before detection by finding a subset of data which captures the behavior of the data, allowing for the detection of temporal, spatial, and spatiotemporal data anomalies. Chitradevi et al. [132] proposes a two-step algorithm. First, a PCA model is built that can be used for fault detection. Second, the Mahalanobis distance is used to determine the similarity between the current sensor readings against the developed sensor data model. However, conventional PCA approaches are sensitive to data anomaly frequency in collected data and fail to detect slow and long-duration anomalies. Xie et al. [133] addresses this problem by using a multi-scale principal component analysis (MSPCA) to detect anomalies and extract and interpret information. MSPCA uses both wavelet analysis and principal component analysis. The time-frequency information of the data is captured using wavelet analysis while principal component analysis is used to detect data anomalies. This technique allows for detecting gradual and persistent anomalies with different time-frequency features.

Lastly, a hybrid approach is proposed by Warriachet al. [134] to detect data anomalies based on the three methods. By combining rule-based methods, estimation-based, and learning-based methods, they are able to leverage domain and expert knowledge, sensor spatial and temporal correlations and inferred models for the faulty sensor readings using training data. This approach has the benefits of the above approaches but also requires more processing capability and power at sensor nodes.

One major issue in CPS data corruption detection is determining when anomalous data is erroneous. Tang et al. [135] investigates the trustworthiness of sensor data and propose a method called Tru-Alarm to eliminate false alarms by filtering out the noise and

false information. Tru-Alarm is able to estimate the source of alarm by constructing an alarm graph and conducting trustworthiness inference based on the graph links.

3.3.2.2. Behavioral detection. Behavioral approaches have also been implemented to detect the anomalous behavior of a system rather than the data produced. Many of these approaches are part of intrusion detection systems (IDS). Liao et al. [136] provide a comprehensive overview of IDS approaches for general computing, classifying them as signature-based detection, anomaly-based detection, and stateful protocol analysis. Signature-based detection, also known as knowledge-based detection, detects a pattern or string that corresponds to a known attack. This technique is limited to detecting known attacks. Anomaly-based detection determines the normal behavior of the system and detects anomalies by comparing the current behavior with the normal behavior model. Anomaly-based detection can monitor any type of activity, including network connections, number and type of system calls, failed login attempts, processor usage, the number of e-mails sent, etc. This approach can detect both known and unknown attacks. Lastly, stateful protocol analysis, also known as specification-based detection, compares a vendor-developed profile of specific protocols to current behavior. An example would be monitoring protocol states such as pairing requests and replies. Modi et al. [137] provide a survey of IDS techniques used for cloud computing. Many of the approaches use techniques similar to statistical anomaly detection, as well as neural networks and fuzzy logic. While some of these techniques are very computationally intensive, they can be implemented on sensor nodes without hindering the real-time access to data.

CPS specific IDS approaches have also been developed. Buttán et al. [138] discuss the WSAN4CIP Project which investigated a number of attack detection methods to determine if a sensor node is compromised. The project included intrusion detection and prevention techniques that were adapted to the wireless environment. A micro-kernel in the sensor node operating system supports multiple levels of security and determines if the code deployed on a sensor node is unchanged. Mitchell et al. [139] provide a detailed review of

CPS related IDS research. In addition to IDS, for traditional networked computing systems, CPS IDS monitors both the embedded components and the physical environment, which under attack may exhibit abnormal properties and behavior. However, this is complicated by legacy technology still used in many CPS. Some legacy components are based on mechanical or hydraulic control with no cyber component, making them difficult to modify or access. Thus CPS IDS must define acceptable component behavior based on sensor readings of the physical environment.

3.3.3. Mitigated of Corrupted Data. Detected erroneous or missing data can be mitigated in a number of ways depending on the criticality and valid time interval of the data. Mitigation can be accomplished by correcting, replacing or ignoring the corrupted data. In many CPS applications, the useful life of a single piece of data is very short making some correction or replacement techniques inappropriate. Additionally, many correction techniques require a great deal of computation making the energy consumption prohibitive. However, in other applications corrupted data minimizes the quality of information and ignoring these errors may cause a serious effect in data analysis. Gantayat et al. [140] provide a review of research on missing or incomplete data. A variety of techniques are used to generate predicted values. Many of these approaches are very similar to the anomaly detection techniques. The following are approaches for mitigating missing and corrupted data:

- Imputation: This technique replaces missing data values with an estimation based on the data stream's probabilistic model.
- Predicted Value Imputation: This technique replaces missing data with estimated values based on the data set. The estimation methods vary in complexity from mean or mode values to more complex estimates from training data.
- Distribution Based Imputation: This technique replaces missing data using a classification algorithm. A set of pseudo-instances is created when a missing value is

encountered. Each pseudo-instance is tested. The replacement value is selected using a weighted comparison.

- Unique Value Imputation: This technique replaces the missing value using simple substitution from historic information.

- Replacing Missing Data: This technique replaces the missing data with a value from a test case that resembles the current data set.

- Rough Sets: This technique uses lower and upper approximations to determine a replacement value. The benefit of this technique is that there is no need for preliminary or additional information about the data. A number of extensions to rough set have been proposed including tolerance relation, non-symmetric relation, and valued tolerance relation.

- Similarity Relation: This technique replaces the missing data after making generalized decisions based on the entire data set.

Each of these mitigation techniques can be deployed on the sensor nodes of a CPS depending on the storage and processing limitations of the sensor node. These techniques can be employed to replace corrupted or missing data allowing for correct execution.

In some higher level data cleansing activities, multiple cleansing alternatives are available on a system. In this case, automatic data cleansing requires a set of policies to determine the appropriate option. Mezzanzanica et al. [141] present a model-based approach for developing a policy for the data cleansing of a data set. In some cases, data cleansing requires a domain expert to be involved in the data cleansing effort. Gschwandtner et al. [142] presents an interactive visual analysis tool called TimeCleanser. This system is designed for data cleansing of time-oriented data. TimeCleanser combines semi-automatic data quality checks and data visualizations to assist the user.

3.3.4. Effects of Corrupted Data. Ayatolahi et al. [143] experimentally studies the effects of single and double bit errors in an instruction set architecture registers and main memory using fault injection. Fault injection is a method to test and assess the dependability (availability, reliability, and maintainability) and performance of fault-tolerant and fail-safe systems. One of its uses is benchmarking the error sensitivity of a system when it experiences hardware faults in the processor or main memory. To measure the error sensitivity, bit-flip errors are injected in the main memory and Instruction Set Architecture registers. The experiment consisted of nine campaigns of the single bit-flip and double bit-flip models each with 12,000 trial runs on 13 test programs. Each run was categorized as one of the following categories:

- No Impact: The program terminates normally and the error does not affect the output of the program.
- Hardware Exception: The processor detects an error by raising a hardware exception.
- Timeout: The program fails to terminate within a predefined time.
- Silent Data Corruption: The program terminates normally, but the output is erroneous and there is no indication of failure.

This experiment demonstrates the error sensitivity of different registers and memory locations. Their results showed that the most common effect of both single and double-bit errors is a hardware exception, with double-bit errors resulting in more than single-bit. Both single and double-bit errors produced corrupted results roughly 30% of the time. Timeout errors were rarely encountered. Overall double-bit errors more impact on results. Obviously, the error sensitivity varies depending on bit positions, registers and the software tested. This study was conducted on a diverse set of programs, with various implementation sizes, input types and sizes, and functionalities. While none of the programs tested were

CPS control systems, the results give estimates of the probability of correct incomplete, or corrupted results based on corrupted input.

In an attempt to improve a program's sensitivity to corrupted data, Sangchoolie et al. [144] study the impact of compiler optimizations on various programs. The study provides insight into the impact of different levels of GNU Compiler Collection compiler optimizations (-O1, -O2, -O3, -Os) on the corrupted data sensitivity of programs using twelve benchmark programs. These optimizations can be used to improve the performance of the compiled program. However, the results of the experiment show that the data corruption sensitivity of the optimized programs is only marginally lower than the non-optimized programs.

3.4. MODELING THE PROPAGATION OF CORRUPTED DATA

Despite the considerable body of work on areas such as data flow and failure propagation, no models have been specifically developed to capture the propagation of corrupted data and resulting state in a networked system. Therefore, no existing work can be used for direct comparison to this project. The closest related work is presented in [145], which attempts to quantify the dependency between electrical and information infrastructure by evaluating the impact of missing data as a result of an attack. However, it does not capture the potential for propagation of erroneous or missing data. The studies most relevant to the research presented in this paper are on the topics of data corruption and data cleansing.

In order to develop a model to capture the propagation of corrupted data between nodes in a CPS, a qualitative model was created to understand the data cleansing process and the potential for propagation. The qualitative model was used as the foundation for a quantitative Petri net model. The Petri net model abstracts the data storage, processing, and communication of a CPS. This model is intended to facilitate analysis of failures caused by data corruption. The abstraction allows the analysis to focus on information exchange and

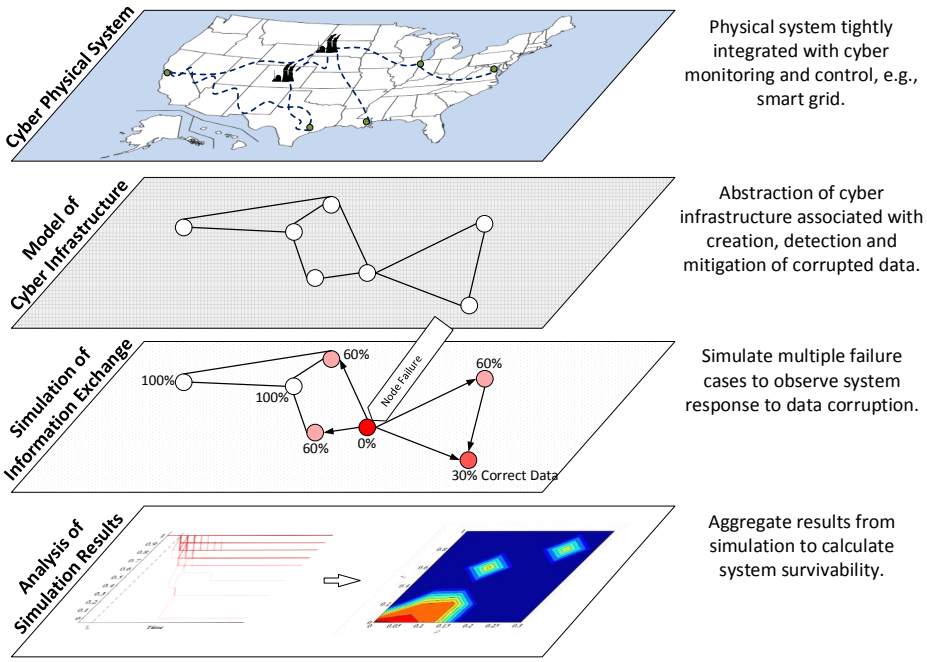


Figure 3.2. Overview of survivability evaluation process.

eliminates the need for simulation of the physical infrastructure of a CPS. The state of the system is quantified in terms of the percentage of system data that remains uncorrupted. Thus, the model enables analysis of data dependencies and facilitates survivability evaluation of a system in the presence of cyber faults producing corrupted data. This process is shown in Figure 3.2. The utility of the model is illustrated by application to an example monitoring system. This work was published in [12].

3.4.1. Qualitative Model for Propagation of Data Corruption. The qualitative model for the propagation of corrupted data between sensor/actuator nodes of a CPS is shown in Figure 3.3. This qualitative model focuses on the information processing steps of a single node in the network and determines if corrupted data results in corrupted control data and the propagation of corrupted data to neighboring nodes.

Initially, corrupted data enters the node from local sensors, neighboring nodes, or local storage causing a node fault. The corrupted data can be either erroneous or missing. Next, the corruption detection technique is employed. If corrupted data is

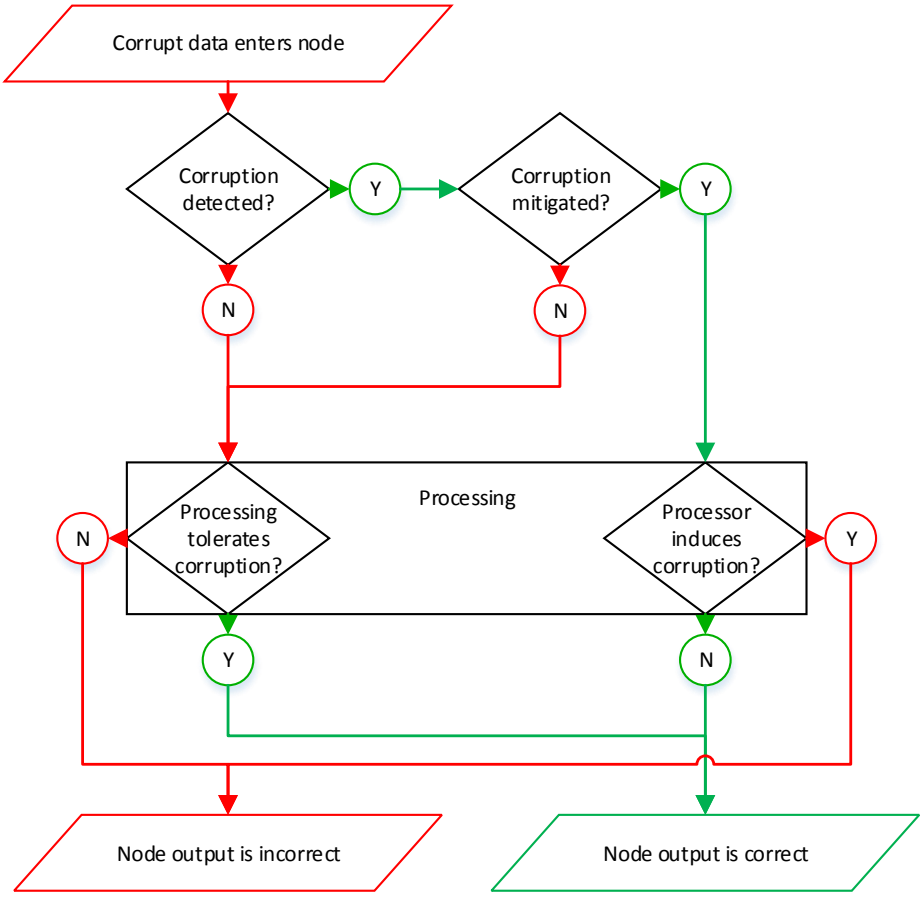


Figure 3.3. Qualitative model of the propagation of corrupted data in a CPS. Initially corrupted data enters a node; after the cleansing process the results in either a functional system of corrupted output data.

detected, mitigation is attempted. If the mitigation is successful, the node has recovered and regular execution takes place. If the corrupted data is not detected or mitigation fails, the node processes the corrupted data causing a node error. If processing of corrupt data results in correct output data, then the corruption has no impact on system operation. If the processing results in a hardware exception or timeout, then corrupted data is introduced in the form of missing data. However, if the processing results in an erroneous output, then silent data corruption occurs, introducing corrupted data in the form of an erroneous output. The propagation of corrupted data in the system may be limited if the corruption is detected

by the neighbors of a node. However, understanding the extent to which the corruption has propagated is essential for designing fault-tolerant and reliable systems.

3.4.2. Quantitative Model for Propagation of Data Corruption. The quantitative model, which is based on the qualitative model, is in the form of a colored stochastic discrete time Petri net (CDTSPN) [146, 147]. A basic Petri net is $N = (P, T, A, M_0)$, where P and T are disjoint finite sets of places and transitions, respectively, A is a set of arcs such that $A =\subset (P \times T) \cup (T \times P)$, and M_0 is the initial marking $M_0 = [p_1 = 1, p_2 = 0, \ldots p_n = 1]$ dictating the number of tokens in each place at time t_0. CDTSPNs are stochastic place/transition nets used in system modeling and consist of four primary components: transitions (signified by bars), places (signified by circles), arcs (signified by arrows), and colored tokens (signified by dots). Color provides the ability to distinguish among tokens. The Petri net is discrete-time stochastic due to the non-deterministic firing of transitions, which in the model represent data creation, transmission, and cleansing behaviors.

Formally a CDTSPN is defined as $N = (P, T, A, C, E, G, \rho, M_0)$, where P is a set of places, T is a set of transitions, A is a set of pre- and post-arcs, C is a color set function that maps each place to the set of possible token colors, E is a set of arc expression functions defined on A, G is a set of guard functions defined on T, ρ is the set of nonzero conditional probabilities for the geometrically distributed firing times defined on T, and M_0 is the initial marking of the net.

The model is constructed based on the system's specifications and logical topology of the cyber infrastructure of the networked system or CPS. The system specifications dictate the failure rates of each component in the system (e.g., sensor's probability of error or probability of communication channel failure). These probabilities dictate the firing rates in ρ. Each system node is modeled as a set of places and transitions which abstract the information exchange process including the sources of corrupted data and the data cleansing process performed at the node.

The model of a single sensor/actuator node in a networked system is depicted in Figure 3.4. The model consists of three data places (*input*, *detected*, and *database*), four transitions (*sensor*, *detect*, *mitigate*, *process*) for node execution, and multiple transitions (*input/output channels*) for communication between the nodes. The tokens represent discrete data elements in the modeled system. The token color set, C, corresponds to the three states defined for a data element (i.e., *correct* shown in green, *erroneous* shown in red, or *missing*, shown in blue). The state of the system is determined by the marking, M of the CSPN (i.e., the respective number of correct, erroneous, or missing data elements within each database), of the model after information is exchanged.

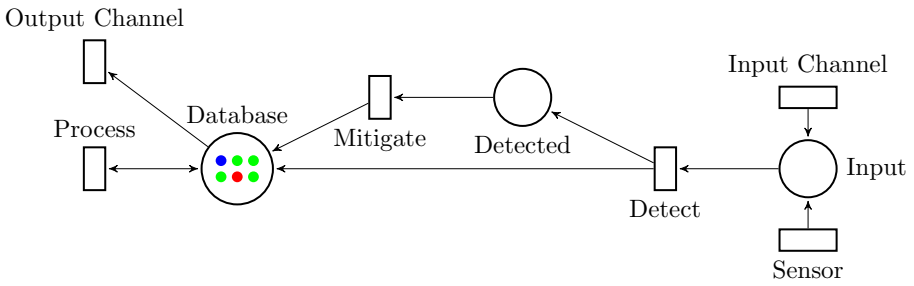


Figure 3.4. Petri net model of a single node of a CPS. Multiple nodes are connected via input/output transitions to model a CPS.

The information exchange process models the flow of data from data sources through the data cleansing process to the database of the sensor/actuator node.

3.4.2.1. Sources of data corruption. The primary sources of data corruption in a CPS are sensors, communication channels, and storage and processing hardware. Each of these sources is represented using transitions in the Petri net model.

Sensors located at a sensor/actuator node of a CPS are modeled with the sensor transition. The sensor transition fires to create a new data token that is deposited in the input place. The firing rate of this transition is determined by the probability of producing corrupted data (i.e., the probabilities of the sensor producing an erroneous reading or not reporting a sensor reading). The sensor transition firing rates are shown in Table 3.1.

Table 3.1. Probability of firing for sensor transition.

Produced Data	Probability	Consequence
Erroneous	P_{err}	Sensor reports an erroneous reading
Missing	P_{miss}	Sensor fails to report reading
Correct	$P_{corr} = 1 - P_{err} - P_{miss}$	Sensor functions correctly

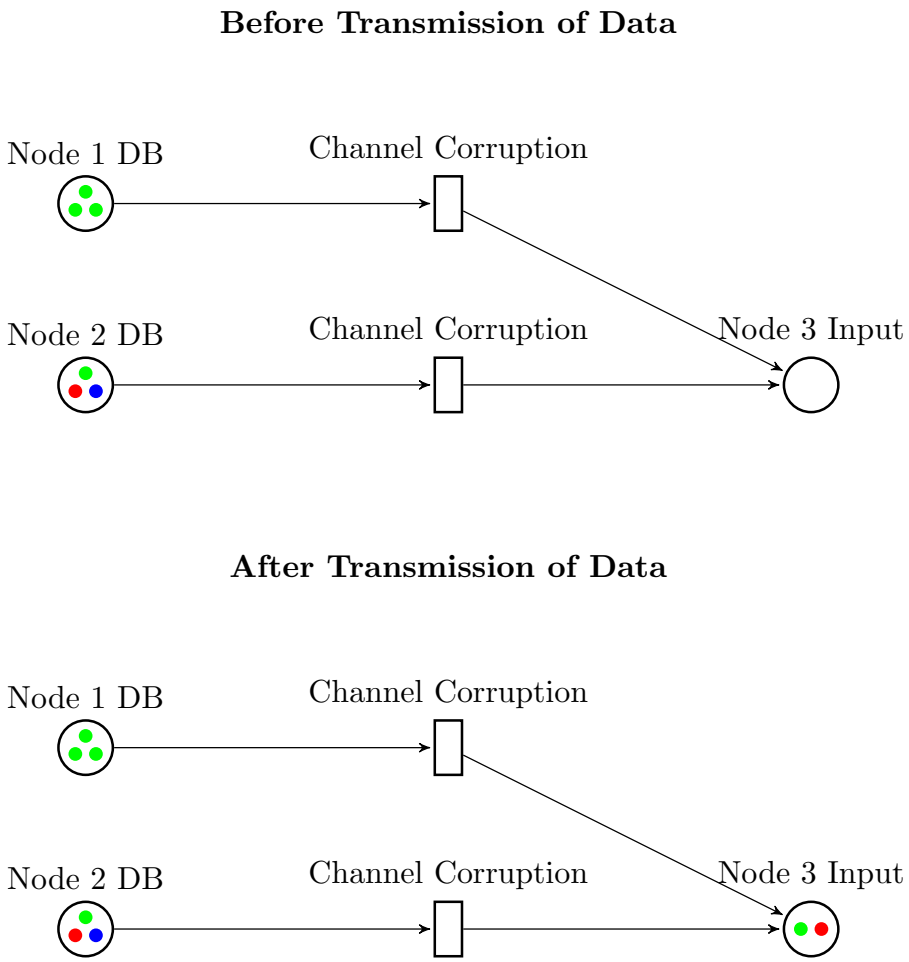


Figure 3.5. Petri net model of single direction data communication. Node 3 has two neighbors, Node 1 and Node 2, which transmit data for control purposes.

The channel transitions model the data corruption caused by communication channels between nodes of the CPS. These transitions connect the database place of neighboring nodes to the input place of the node as shown in Figure 3.5. In this case, Node 3 has two neighbors, Node 1 and Node 2, which transmit data for control purposes.

Each direction of a communication channel is modeled as a separate transition. The channel transition fires by copying one data token from the sending node's database and altering or discarding it in transit. The firing rate of this transition is determined by the probability of the channel corrupting data in transit either by dropping or altering a data element. The channel transition firing rates are shown in Table 3.2.

Table 3.2. Probability of firing for communication transition.

Sent Data	Received Data	Probability	Consequence
Correct	Missing	P_{miss}	Correct data is dropped in transit
	Erroneous	P_{err}	Correct data is altered in transit
	Correct	$P_{corr} = 1 - P_{err} - P_{miss}$	Data arrives correctly
Erroneous	Missing	P_{miss}	Erroneous data is dropped in transit
	Erroneous	$1 - P_{miss}$	Erroneous data is sent altered resulting in erroneous data being received
Missing	Missing	1	No data is sent so no data is received

The process transitions model the data corruption caused by the data processing and storage at a node in the CPS. This transition is a self-loop with the database place. This transition fires with every data element in the database and alters or discards the element based on the hardware specifications. The process transition firing rates are shown in Table 3.3.

Table 3.3. Probability of firing for process transition.

Processed Data	Result Data	Probability	Consequence
Correct	Missing	P_{miss}	Correct Data is lost of as a result of storage and processing
	Erroneous	P_{err}	Correct Data is altered of as a result of storage and processing
	Correct	$P_{corr} = 1 - P_{err} - P_{miss}$	Data is correctly stored and processed
Erroneous	Missing	P_{miss}	Erroneous data is lost of as a result of storage and processing
	Erroneous	$1 - P_{miss}$	Erroneous data is still erroneous as a result of storage and processing
Missing	Missing	1	No data is processed or stored

3.4.2.2. Data cleansing process. The detect and mitigate transitions and the intermediate detected place model the data cleansing process employed at a node in the CPS. The input place contains data newly arrived at the node that is to be added to a node's database. The cleansing process executes the following steps on data in the order it is received.

First, the detection process flags data as being corrupted. If the detection process is independent of the data currently stored in the database, the transition utilizes static probabilities of detection, $P_{truepositive}$, and false positives, $P_{falsepositive}$. The detection

process can detect or falsely detect corrupted data elements. The detection transition firing rates are shown in Table 3.4.

Table 3.4. Probability of firing for detection transition.

Input	Detection	Probability	Consequence
Correct	Flagged	$P_{falsepositive}$	Incorrectly flagged data by detection process
	Not Flagged	$P_{truenegative} = 1 - P_{falsepositive}$	Correct behavior
Erroneous	Flagged	$P_{truepositive}$	Correct behavior
	Not Flagged	$P_{falsenegative} = 1 - P_{truepositive}$	Undetected Erroneous data
Missing		1	Missing data is always flagged as missing.

However, if the probability of detection is dependent on the accuracy of the historical data stored in the node's database, an additional Petri net structure is used, as shown in Figure 3.6. In this case, the detection probabilities will change depending on the amount of correct and corrupted data in the database. The transitions fire changing the cleanser state when the percent of corruption in the database passes some system specified threshold. The database state is used as an additional firing condition in the detection transition to change the probabilities of detection.

In both the static or state-based detection process, the detection transition fires, moving the data token from the Input place and either detected place if the token is believed to be corrupted, or the Database place if it is not.

Second, the mitigation process attempts to correct the data that has been flagged as corrupted. Similar to the detection process, the mitigation process can be independent of the data currently stored in the database or utilize historic data. The mitigation process can replace the token with a correct token, P_{fix}, replace the token with a missing token, $P_{discard}$, thereby functionally discarding the data element, or it can replace the token with an erroneous token, $P_{fixfailure}$, if the mitigation is unable to repair the corruption. The mitigation transition firing rates are shown in Table 3.5.

The mitigate transition fires, moving the data tokens from the detected place, altering them based on the probabilities in Table 3.5, and placing them in the database place. The mitigation process corrects or discards corrupted data; however, if a correct token is falsely

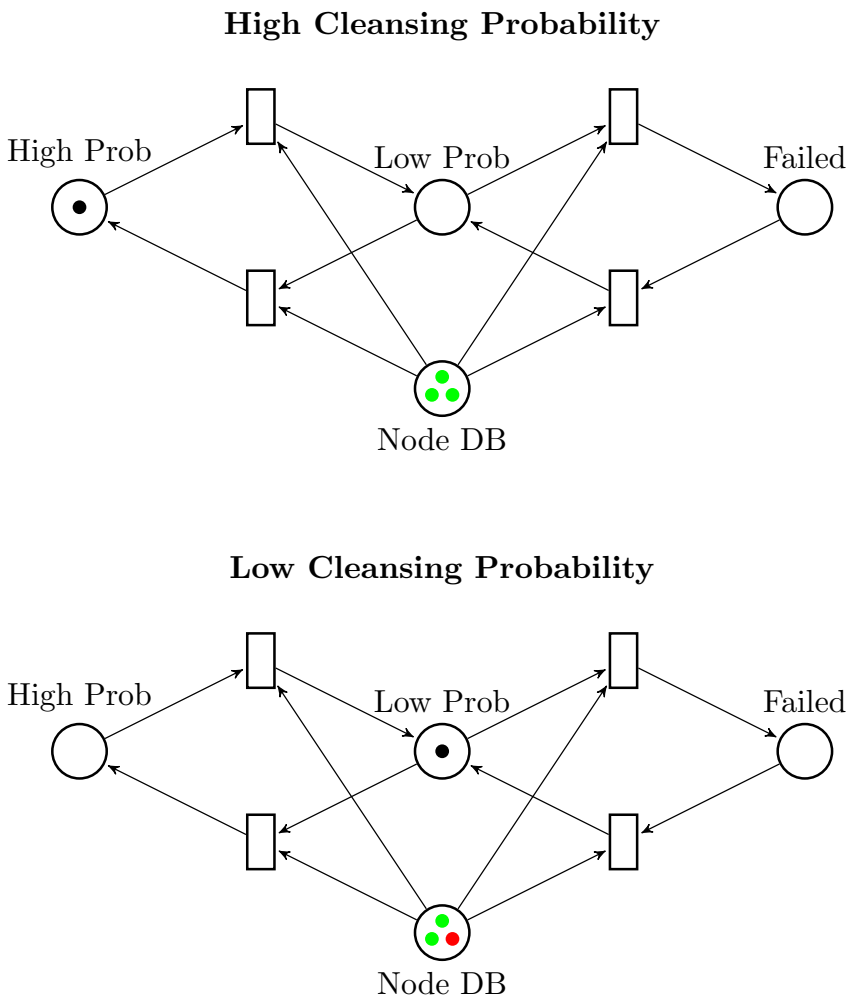


Figure 3.6. Petri net structure for non-static detection probabilities.

Table 3.5. Probability of firing for mitigation transition.

Input	Mitigation	Probability	Consequence
Correct	Discard (Missing)	$P_{discard}$	Incorrectly flagged data is discarded by mitigation process
	Mitigate Failed (Erroneous)	$P_{fixfailure} = 1 - P_{discard}$	Incorrectly flagged data is altered by mitigation process
Erroneous	Mitigate (Correct)	P_{fix}	Erroneous data is corrected or replaced by suitable data
	Discard (Missing)	$P_{discard}$	System aware that replacement data is not accurate
	Mitigation Failed (Erroneous)	$P_{fixfailure} = 1 - P_{fix} - P_{discard}$	System unable to correct or replace erroneous data
Missing	Mitigate (Correct)	P_{fix}	Mitigation able to replace with correct data
	Discard (Missing)	$P_{discard}$	System aware that replacement model is not accurate
	Mitigation Failed (Erroneous)	$P_{fixfailure} = 1 - P_{fix} - P_{discard}$	System unable to correct or replace erroneous data

detected as corrupted it will either be corrupted (by unnecessary mitigation) or discarded by the mitigation process. This can be the source of data corruption in a system.

In this model, the size of the databases is finite, which implies that as data elements are added to the database, older or unused elements will be removed. To accomplish this, the detect and mitigate transitions remove a random data token from the database place when a new data token is added. Tokens are chosen at random because data tokens in the database place are assumed to be aggregated information rather than individual sensor readings.

3.4.2.3. Simulation of model. The model is executed over time and the marking (number of correct, missing, and erroneous data tokens in each database place of the Petri net) is logged after each time step. Each time step represents the reading of sensors and exchange of information between nodes in the CPS. During each time step, each sensor creates a data element and all nodes exchange data elements via communication channels. The model is simulated synchronously at each time step by first firing all sensor and channel transitions, then firing the detection and mitigation transitions for each node, and finally firing the process transition. If data cleansing is dependent on the state of the data in the database, then cleanser state is updated by firing the enabled transitions.

This simulation process shows the effect of transient failures that cause data corruption. A transient failure is a temporary failure such as a packet being corrupted during transit across a wireless link. This is a transient failure because the transmission of one packet failed but the channel is still functional. In order to evaluate failure cases involving persistent failures that cause data corruption, the transition probabilities are altered. A persistent failure is a permanent failure requiring repair, such as a cut fiber optic cable. A persistent sensor failure in which the sensor produces erroneous readings would be ($P_{err} = 1, P_{miss} = 0, P_{corr} = 0$). Alternatively, a persistent sensor failure in which the sensor fails to produce a reading would be ($P_{err} = 0, P_{miss} = 1, P_{corr} = 0$).

The node execution is as follows:

1. The sensor and channel transitions fire and create data tokens in the input place.

2. The detection transitions fire, moving the data tokens from the input place to either the detected place or the database place.

3. The mitigate transition fires, moving the data tokens from the detected place, changing the state of the data token, and placing it in the database place.

4. The process transitions fire, replacing tokens in the database and altering the token's state to simulate corruption that occurs during processing and storage.

5. (If dependent cleansing) the cleanser state is updated.

3.4.3. Case Study: IEEE-57 Bus Smart Grid. In this section, the proposed approach is demonstrated by applying it to a smart grid based on a test system commonly used in power engineering. Specifically, the model for the propagation of corrupted data and survivability analysis technique are demonstrated using a smart grid based on the IEEE 57-bus test system [79]. The IEEE test system was supplemented with cyber infrastructure to create a smart grid. The cyber infrastructure was comprised of phasor measurement units (PMUs), which record and communicate dynamic power system data; flexible AC transmission system (FACTS) devices, which adjust the flow of power in the transmission lines, communication channels; and a four-zone distributed decision support system that determines optimal settings for the FACTS devices based on data provided by the PMUs, FACTs devices and generators. The methods presented in [80] and [81] were applied to determine where to place the PMUs and FACTS devices on the smart grid and utilized the four-zone distributed control zones presented in [148], shown in Figure 3.7. The distribution of components across the zones is shown in Table 3.6.

Table 3.6. Distribution of components across zones.

Zone	# of Buses	# of Generators	# of PMUs	# of FACTS	# of Loads
1	24	2	3	3	24
2	8	2	2	0	7
3	10	1	3	1	9
4	15	2	4	3	12

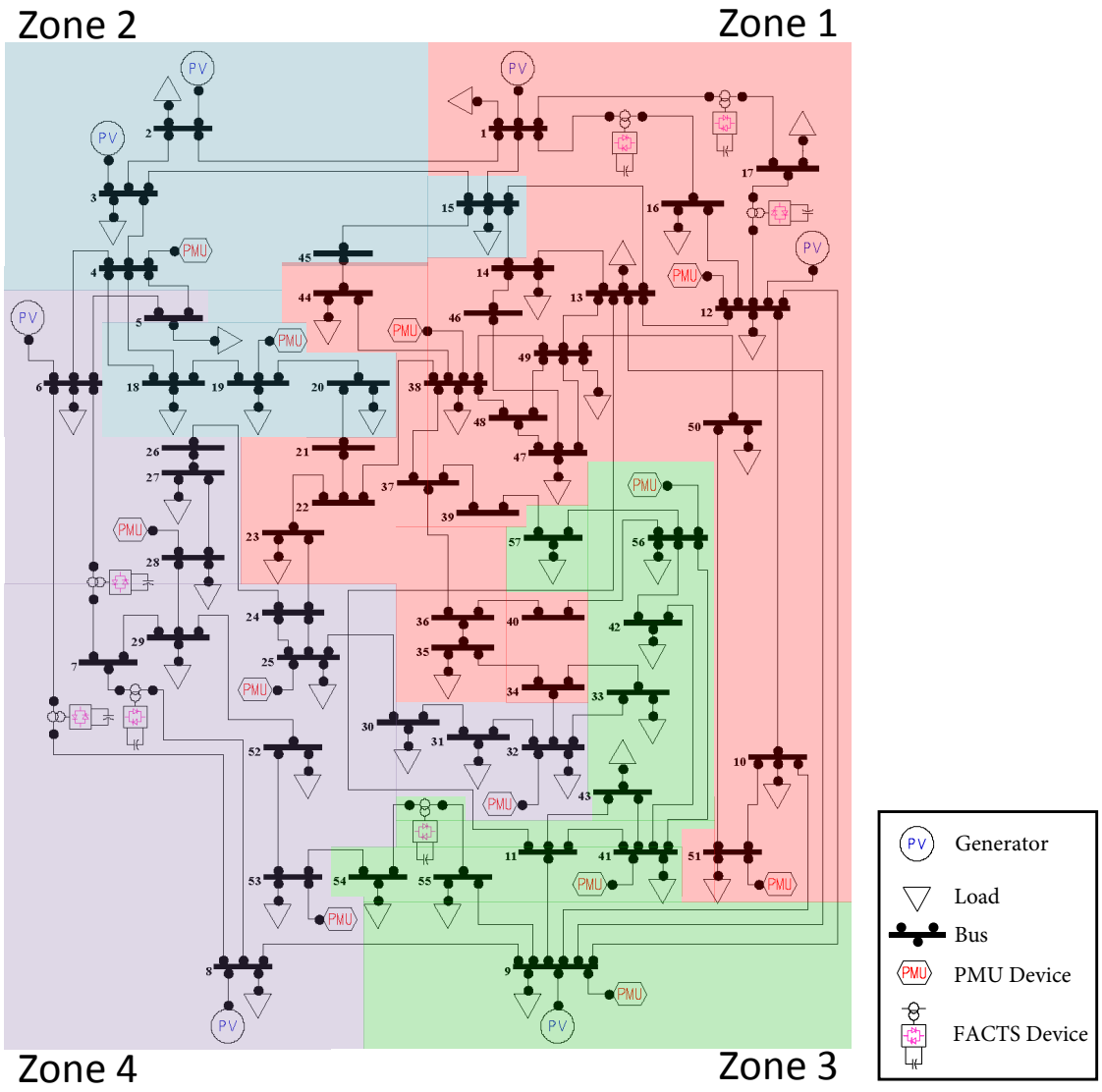


Figure 3.7. Single line diagrams of IEEE 57-bus smart grid test system.

The cyber infrastructure of each control zone is a database and processor that collects data from all of the data-generating components within the zone (i.e., generators, FACTS devices, and PMUs). Additionally, control zones exchange control information with neighboring zones that share a physical border. The logical topology of the cyber infrastructure is shown in Figure 3.8.

3.4.3.1. Experiment description and failure cases. The *essential service* expected of the cyber infrastructure of a smart grid is to collect and process data as a part

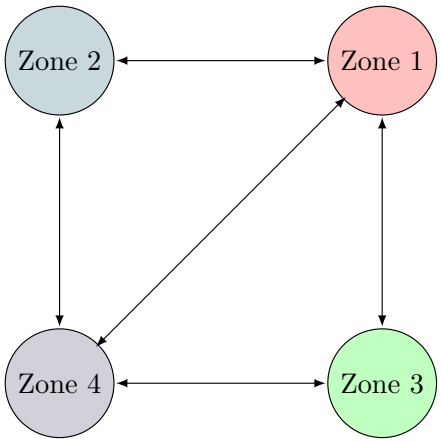


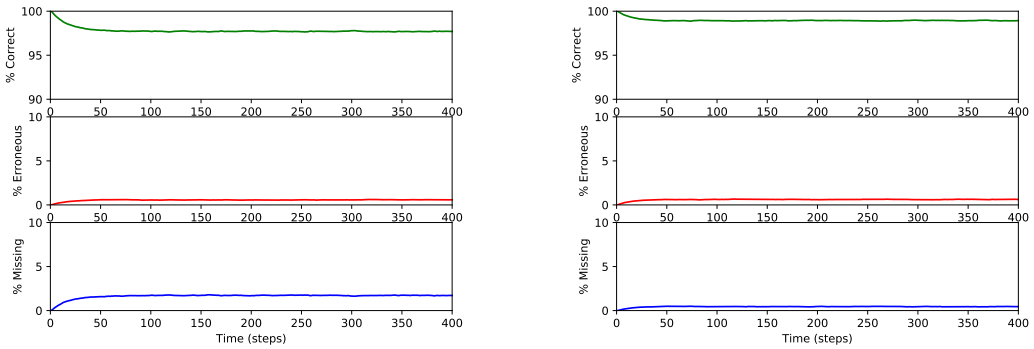
Figure 3.8. Logical topology of IEEE 57-bus smart grid test system.

of the decision support provided to the power system. Therefore, the FoM is defined as the percentage of correct data in all of the control zone databases. This reflects the overall quality of information being processed by the decision support algorithm. For this study, the consequences of the failure of a single data-generating element were analyzed and the survivability improvement of employing data cleansing systems on each control zone was determined.

The following parameters were used for the various cyber infrastructure components. The probabilities of producing an erroneous or missing data element by the data generating components are as follows: generators $P_{err} = 0.001$ and $P_{miss} = 0.001$, FACTS devices $P_{miss} = 0.002$ and $P_{err} = 0.002$, PMUs $P_{miss} = 0.01$ and $P_{err} = 0.01$. The probabilities of altering or dropping data by the communication channels is $P_{err} = 0.001$ and $P_{miss} = 0.03$, respectively. The data processing altered or lost the data with a probability of $P_{err} = 0.0001$ and $P_{miss} = 0.0001$, respectively. The parameters of the evaluated data cleansing system are a detection system with a true positive and false positive of $P_{truepositive} = 0.95$ and $P_{falsepositive} = 0.001$, respectively, and a mitigation system with a probability of correcting the data of $P_{fix} = 0.5$ and a probability of discarding the data of $P_{discard} = 0.25$.

The model was constructed and simulated using the SNAKES Petri net tool [149]. The results were averaged over 1000 simulations of each test case. The failure case was injected at $t = 200$, and the simulation continued until no further failures were detected. For the sake of consistency among the three IEEE bus systems and ease of comparing the plots, all simulations were continued for 400 time steps. In all tests, all nodes were initialized with no corrupted or missing data.

3.4.3.2. Results and survivability analysis. Figure 3.9 depicts the simulation results without any persistent component failures for the system without data cleansing capability (Figure 3.9a) and with data cleansing capability (Figure 3.9b). The desired outcome is a value of 100 characterized by the system having no corrupted data. However, in both systems, erroneous data exists as a result of transient failures of the sensors. The percentage of missing data is the system is high due to the high probability of dropped data elements by the communication channels.

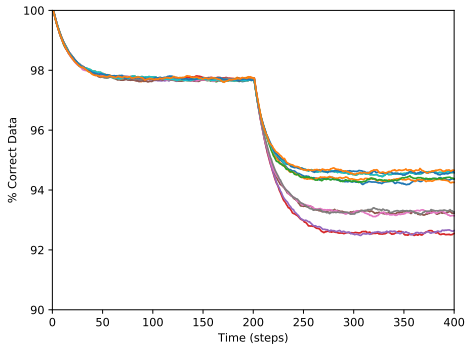


(a) Cyber Infrastructure without Data Cleansing (b) Cyber Infrastructure with Data Cleansing

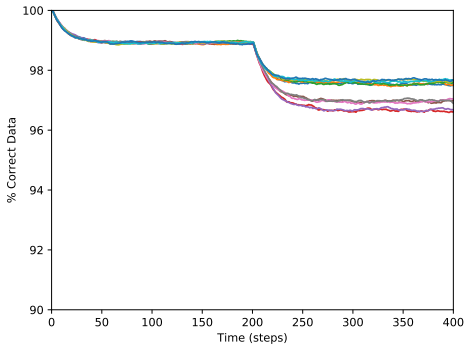
Figure 3.9. Correct data vs. time for IEEE 57-bus smart grid with failure event.

Figure 3.10 depicts the simulation results for each of the test systems using the percentage of corrupted data in the system as the FoM. These results indicate that while no failure case resulted in a catastrophic degradation in the FoM, all failure cases impacted the quality of information processed by the system. However, the results indicate a significant

improvement in the FoM as a result of utilizing data cleansing capability in the cyber infrastructure.



(a) Cyber infrastructure without data cleansing



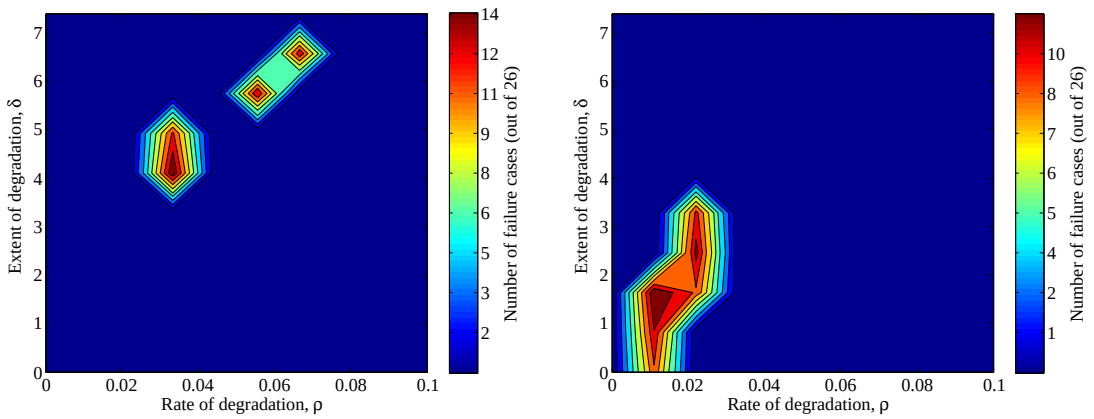
(b) Cyber infrastructure with data cleansing

Figure 3.10. Correct data vs. time for IEEE 57-bus smart grid with failure event.

The maximum rate and extent of degradation were extracted from the log of each failure case. Figure 3.11 depicts a two-dimensional histogram of the two systems simulated. In an ideal system, these histograms would be dense near the origin and sparse elsewhere, reflecting slow and minimal degradation in response to failure. However, there are clusters of failure cases with higher rates and extents of failure that fall in the upper and/or right regions of the histogram. The presence of these clusters indicates that many of the failure cases have similar values of rates and extents of degradation. When comparing the two systems, there is an obvious improvement in survivable behavior as a result of deploying data cleansing capabilities.

Averaging the degradation index (Cartesian distance from a degradation point, $(\delta_j,\ \rho_j)$) for each failure case produces the survivability indices of the system without cleansing capability of 0.1259 and with cleansing capability of 0.0501, showing a significant improvement in survivable behavior.

3.4.4. Summary of Research Contribution. The research contribution presented in Section 3.4 is a novel model to analyze the propagation of corrupted data in the cyber infrastructure of a CPS. This model abstracts data processing, communication, and cleansing



(a) Cyber infrastructure without data cleansing (b) Cyber infrastructure with data cleansing

Figure 3.11. Color indicates the number of failure cases which resulted in the corresponding rate and extent of degradation. The desired outcome is a single cluster near the origin characterized by slow and minimal degradation of the FoM.

in a networked system and produces a figure-of-merit over time in response to a failure case. These results are used for survivability analysis. This model and associated survivability analysis technique were demonstrated using the cyber infrastructure of a smart grid based on conventional IEEE power test systems. Future work will extend the data cleansing process of the model to study more complex cleansing techniques. Additionally, an analytical model for the propagation of corrupt data will be development based on the Petri net model.

4. CONCLUSIONS AND FUTURE WORK

The goal of the research presented in this dissertation is to illuminate data dependencies and interdependencies in CPSs and to quantitatively capture the effect of these dependencies by modeling the propagation of corrupted data. I have presented a number of metrics and models for the analysis of CPS with respect to dependability attributes, with a specific focus on survivability.

First, I demonstrated a model of system-level reliability for critical infrastructure CPSs, where reliability is determined as a function of the respective reliabilities of the control system and cyber infrastructure. This was demonstrated with a case study using an intelligent water distribution network. Future directions of this work are exploring additional fault states for various cyber and physical components, to represent more realistic behavior of the system; and to include degraded systems states to analyze the water distribution network from a survivability perspective.

Second, I justified the use of survivability as a metric for the evaluation of CPSs and presented an approach for evaluating survivability. This approach can be used to evaluate the survivability of any complex, networked system during a disruptive event, by observing the extent and rate of degradation of a domain-specific figure-of-merit. This evaluation approach facilitates targeted hardening of CPSs by prioritizing components based on their fragility and criticality. The entire approach, from evaluation to hardening, are demonstrated with a case study using smart grid systems. Future directions of this work are to improve the scalability of the approach by refining the selection of failure cases and investigating and the use of superposition for evaluating systems with independent components.

Third, I proposed a behavioral model for intelligent transportation systems that combine legacy (primarily manned) vehicles with their more modern semi- and fully-autonomous counterparts. The understanding gained from this model will be useful in

mitigating the results of failure, leading to increased safety. The next step for this research is validation on a testbed that allows for concurrent operation of vehicles with varying levels of autonomy. Given the emergence of driverless vehicles as a very active research area, this validation is expected to become feasible in the very near future.

Lastly, I presented qualitative and quantitative models for the propagation of corrupted data in CPSs. Qualitative modeling served as a precursor to quantitative modeling, by providing an understanding of the potential for the propagation of corrupted data as a result of information exchange between control entities of a CPS. The quantitative model measures the state of data in a system as information is exchanged and can be used in conjunction with survivability evaluation by enabling the use of correct (vs. corrupt or missing) data as a figure-of-merit. This application is illustrated with a case study on a smart grid based on the IEEE 57-bus test system. The propagation model abstracts information exchange in a CPS. It can be refined by considering domain-specific aspects of the physical infrastructure and cyber-physical interdependency that could facilitate (or complicate) detection or mitigation of data corruption.

BIBLIOGRAPHY

[1] R. Rajkumar, I. Lee, L. Sha, and J. Stankovic, "Cyber-physical systems: The next computing revolution," in *Proceedings of the 47th ACM/IEEE Design Automation Conference (DAC)*, pp. 731 –736, June 2010.

[2] P. Derler, E. Lee, and A. Vincentelli, "Modeling cyber-physical systems," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 13–28, 2012.

[3] A. A. Kirilenko and A. W. Lo, "Moore's law versus Murphy's law: Algorithmic trading and its discontents," *The Journal of Economic Perspectives*, pp. 51–72, 2013.

[4] B. Miller and D. Rowe, "A survey of SCADA and critical infrastructure incidents," in *Proceedings of the 1st ACM Conference on Research in Information Technology*, pp. 51–56, 2012.

[5] E. S. Agency, "Schiaparelli landing investigation completed." http://www.esa.int/Our_Activities/Space_Science/ExoMars/Schiaparelli_landing_investigation_completed. Accessed: 2017-05-02.

[6] M. Woodard and S. Sedigh Sarvestani, "Modeling of autonomous vehicle operation in intelligent transportation systems," in *Proceedings of the 5th International Workshop on Software Engineering for Resilient Systems*, pp. 133–140, 2013.

[7] K. Marashi, M. Woodard, S. Sedigh Sarvestani, and A. R. Hurson, "Quantitative reliability analysis for intelligent water distribution networks," in *Proceedings of the Embedded Topical Meeting on Risk Management for Complex Socio-Technical Systems, Annual Meeting of the American Nuclear Society*, November 2013.

[8] M. Woodard, S. Sedigh Sarvestani, and A. R. Hurson, "A survey of research on data corruption in cyber-physical critical infrastructure systems," vol. 98 of *Advances in Computers*, pp. 59–87, Elsevier, 2015.

[9] M. Woodard, M. Wisely, and S. Sedigh Sarvestani, "A survey of data cleansing techniques for cyber-physical critical infrastructure systems," vol. 102 of *Advances in Computers*, pp. 63–110, Elsevier, 2016.

[10] M. Woodard, K. Marashi, and S. Sedigh Sarvestani, "Survivability evaluation and importance analysis for complex networked systems," *Submitted to IEEE Transactions on Network Science and Management*, 2017.

[11] N. Jarus, M. Woodard, K. Marashi, S. Sedigh Sarvestani, J. Lin, A. Faza, and P. Maheshwari, "Survey on modeling and design of cyber-physical systems," *Submitted to ACM Transactions on Cyber-Physical Systems*, 2017.

[12] M. Woodard, S. Sedigh Sarvestani, and A. R. Hurson, "A Petri-net model for propagation of corrupted data in a networked system," *In Preparation for Reliability Engineering & System Safety*, 2017.

[13] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, pp. 11–33, January 2004.

[14] B. Parhami, "A multi-level view of dependable computing," *Computers & Electrical Engineering*, vol. 20, pp. 347–368, July 1994.

[15] The International Telegraph and Telephone Consultative Committee, "ITU-T recommendation E.800. Quality of telecommunication services: Concepts, models, objectives and dependability planning. Terms and definitions related to the quality of telecommunication services," September 2008.

[16] W. S. Lee, D. L. Grosh, F. A. Tillman, and C. H. Lie, "Fault tree analysis, methods, and applications: A review," *IEEE Transactions on Reliability*, vol. 34, no. 3, pp. 194–203, 1985.

[17] R. E. Barlow, J. B. Fussell, and N. D. Singpurwalla, *Reliability and fault tree analysis*, vol. 33. Society for Industrial and Applied Mathematics, 1975.

[18] M. Malhotra and K. S. Trivedi, "Power-hierarchy of dependability-model types," *IEEE Transactions on Reliability*, vol. 43, no. 3, pp. 493–502, 1994.

[19] M. Modarres, M. Kaminskiy, and V. Krivtsov, *Reliability Engineering and Risk Analysis: A Practical Guide*. CRC Press, August 1999.

[20] R. G. Bennetts, "Analysis of reliability block diagrams by Boolean techniques," *IEEE Transactions on Reliability*, vol. R-31, pp. 159–166, June 1982.

[21] H. Ohlef, W. Binroth, and R. Haboush, "Statistical model for a failure mode and effects analysis and its application to computer fault-tracing," *IEEE Transactions on Reliability*, vol. 27, no. 1, pp. 16–22, 1978.

[22] S. Chadjiconstantinidis and M. V. Koutras, "Measures of component importance for Markov chain imbeddable reliability structures," *Naval Research Logistics*, vol. 46, no. 6, pp. 613–639, 1999.

[23] M. V. Boutsikas and M. V. Koutras, "Reliability approximation for Markov chain imbeddable systems," *Methodology and Computing in Applied Probability*, vol. 2, no. 4, pp. 393–411, 2000.

[24] M. Malhotra and K. Trivedi, "Dependability modeling using Petri nets," *IEEE Transactions on Reliability*, vol. 44, pp. 428–440, September 1995.

[25] R. Zeng, Y. Jiang, C. Lin, and X. Shen, "Dependability analysis of control center networks in smart grid using stochastic Petri nets," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 9, pp. 1721–1730, 2012.

[26] Y. Katsigiannis, P. Georgilakis, and G. Tsinarakis, "A novel colored fluid stochastic Petri net simulation model for reliability evaluation of wind/PV/diesel small isolated power systems," *IEEE Transactions onSystems, Man and Cybernetics, Part A: Systems and Humans*, vol. 40, pp. 1296–1309, November 2010.

[27] A. Faza, S. Sedigh, and B. McMillin, "Reliability analysis for the advanced electric power grid: From cyber control and communication to physical manifestations of failure," in *Proceedings of the 28th International Conference on Computer Safety, Reliability and Security*, vol. 5775 of *Lecture Notes in Computer Science*, pp. 257–269, Springer, September 2009.

[28] T. Dasic and B. Djordjevic, "Method for water distribution systems reliability evaluation," in *Proceedings of the 2nd Biennial Meeting of the International Environmental Modeling & Software Society*, 2004.

[29] H. Jun, G. V. Loganathan, A. K. Deb, W. Grayman, and J. Snyder, "Valve distribution and impact analysis in water distribution systems," *Journal of Environmental Engineering*, vol. 133, pp. 790–799, August 2007.

[30] B. C. Ezell, J. V. Farr, and I. Wiese, "Infrastructure risk analysis model," *Journal of Infrastructure Systems*, vol. 6, pp. 114–117, September 2000.

[31] J. Wagner, U. Shamir, and D. Marks, "Water distribution reliability: Analytical methods," *Journal of Water Resources Planning and Management*, vol. 114, no. 3, pp. 253–275, 1988.

[32] R. Gupta and P. Bhave, "Reliability analysis of water distribution systems," *Journal of Environmental Engineering*, vol. 120, no. 2, pp. 447–461, 1994.

[33] L. W. Mays, "Review of reliability analysis of water distribution systems," in *Proceedings of the 7th International Association for Hydro-Environment Engineering and Research International Symposium*, pp. 53–60, July 1996.

[34] A. Torii and R. Lopez, "Reliability analysis of water distribution networks using the adaptive response surface approach," *Journal of Hydraulic Engineering*, vol. 138, no. 3, pp. 227–236, 2012.

[35] S. Rinaldi, "Modeling and simulating critical infrastructures and their interdependencies," in *Proceedings of the 37th Annual Hawaii International Conference on Systems Sciences (HICSS)*, 2004.

[36] P. Pederson, D. Dudenhoeffer, S. Hartley, and M. Permann, "Critical infrastructure and interdependency modeling: A survey of US and international research," tech. rep., Idaho National Laboratory, August 2006.

[37] W. Kuo and M. Zuo, *Optimal Reliability Modeling: Principles and Applications*. Wiley, 2003.

[38] V. Singh, S. Jain, and A. Tyagi, *Risk and Reliability Analysis: A Handbook for Civil and Environmental Engineers*. ASCE Press/American Society of Civil Engineers, 2007.

[39] United States Environmental Protection Agency, "EPANET2 User's manual." http://www.nepis.epa.gov/Adobe/PDF/P1007WWU.PDF. Accessed: 2017-05-02.

[40] J. Lin, S. Sedigh, and A. Hurson, "Ontologies and decision support for failure mitigation in intelligent water distribution networks," in *Proceedings of the 45th Hawaii International Conference on System Sciences*, January 2012.

[41] J. Lin, S. Sedigh, and A. Miller, "Towards integrated simulation of cyber-physical systems: A case study on intelligent water distribution," in *Proceedings of the 8th IEEE International Conference on Dependable, Autonomic and Secure Computing*, pp. 690–695, 2009.

[42] J. F. Meyer, "On evaluating the performability of degradable computing systems," *IEEE Transactions on Computers*, vol. 100, no. 8, pp. 720–731, 1980.

[43] B. Iyer, L. Donatiello, and P. Heidelberger, "Analysis of performability for stochastic models of fault-tolerant systems," *IEEE Transactions on Computers*, vol. C-35, pp. 902–907, October 1986.

[44] R. Smith, K. Trivedi, and A. Ramesh, "Performability analysis: measures, an algorithm, and a case study," *IEEE Transactions on Computers*, vol. 37, pp. 406–417, April 1988.

[45] G. Ciardo, R. A. Marie, B. Sericola, and K. S. Trivedi, "Performability analysis using semi-Markov reward processes," *IEEE Transactions on Computers*, vol. 39, pp. 1251–1264, October 1990.

[46] D. Henry and J. E. Ramirez-Marquez, "Generic metrics and quantitative approaches for system resilience as a function of time," *Reliability Engineering & System Safety*, vol. 99, pp. 114 – 122, 2012.

[47] S. Hosseini, K. Barker, and J. E. Ramirez-Marquez, "A review of definitions and measures of system resilience," *Reliability Engineering & System Safety*, vol. 145, pp. 47 – 61, 2016.

[48] M. Ouyang and L. Duenas-Osorio, "Time-dependent resilience assessment and improvement of urban infrastructure systems," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 22, no. 3, 2012.

[49] K. Barker, J. E. Ramirez-Marquez, and C. M. Rocco, "Resilience-based network component importance measures," *Reliability Engineering & System Safety*, vol. 117, pp. 89–97, September 2013.

[50] R. Ghosh, D. S. Kim, and K. S. Trivedi, "System resiliency quantification using non-state-space and state-space analytic models," *Reliability Engineering & System Safety*, vol. 116, pp. 109–125, September 2013.

[51] M. N. Albasrawi, N. Jarus, K. A. Joshi, and S. Sedigh Sarvestani, "Analysis of reliability and resilience for smart grids," in *Proceedings of the 38th IEEE Computer Software and Applications Conference*, pp. 529–534, 2014.

[52] C. Nan and G. Sansavini, "A quantitative method for assessing resilience of interdependent infrastructures," *Reliability Engineering & System Safety*, vol. 157, pp. 35 – 53, 2017.

[53] R. Filippini and A. Silva, "A modeling framework for the resilience analysis of networked systems-of-systems based on functional dependencies," *Reliability Engineering & System Safety*, vol. 125, pp. 82–91, 2014.

[54] R. J. Ellison, D. A. Fisher, R. C. Linger, H. F. Lipson, and T. Longstaff, "Survivable network systems: An emerging discipline," tech. rep., DTIC Document, 1997. Retrieved 2017-05-02.

[55] C. Queiroz, A. Mahmood, and Z. Tari, "A probabilistic model to predict the survivability of scada systems," *IEEE Transactions on Industrial Informatics*, vol. 9, pp. 1975–1985, November 2013.

[56] R. Ball, *The fundamentals of aircraft combat survivability analysis and design*. No. v. 1 in AIAA education series, American Institute of Aeronautics and Astronautics, 2003.

[57] National Communications System, Technology & Standards Division, "Telecommunications: Glossary of telecommunication terms - federal standard 1037C." http://www.its.bldrdoc.gov/fs-1037/fs-1037c.htm, 1996. Retrieved 2017-05-02.

[58] M. S. Deutsch and R. R. Willis, *Software Quality Engineering: A Total Technical and Management Approach*. Software Engineering, Prentice Hall, 1988.

[59] A. Avritzer, L. Carnevali, L. Happe, B. R. Haverkort, A. Koziolek, D. Menasché, A. Remke, and S. Sedigh Sarvestani, "Survivability evaluation of gas, water and electricity infrastructures," in *Proceedings of the 7th International Workshop on Practical Applications of Stochastic Modelling (PASM)*, (Newcastle upon Tyne, UK), May 2015.

[60] P. E. Heegaard and K. S. Trivedi, "Network survivability modeling," *Computer Networks*, vol. 53, no. 8, pp. 1215 – 1234, 2009.

[61] T1A1.2 Working Group, "Technical report on enhanced network survivability performance," tech. rep., T1A1.2 Working Group on Network Survivability Performance, February 2001.

[62] L. J. Zhang, W. Wang, L. Guo, Y. Wu, and Y. T. Yang, "A survivability quantitative analysis model for network system based on attack graph," in *Proceedings of the IEEE International Conference on Machine Learning and Cybernetics*, vol. 6, pp. 3211–3216, 2007.

[63] J. C. Knight, E. A. Strunk, and K. J. Sullivan, "Towards a rigorous definition of information system survivability," in *Proceedings of the DARPA Information Survivability Conference and Exposition*, vol. 1, pp. 78–89, 2003.

[64] Z. S. Ma, "A unified definition for reliability, survivability and resilience inspired by the handicap principle and ecological stability," *International Journal of Critical Infrastructures*, vol. 8, no. 2-3, pp. 242–272, 2012.

[65] Y. Liu and K. S. Trivedi, "Survivability quantification: The analytical modeling approach," *International Journal of Performability Engineering*, vol. 2, no. 1, p. 29, 2006.

[66] L. Cloth and B. R. Haverkort, "Model checking for survivability!," in *Proceedings of the 2nd IEEE International Conference on the Quantitative Evaluation of Systems*, pp. 145–154, 2005.

[67] A. Aziz, K. Sanwal, V. Singhal, and R. Brayton, "Model-checking continuous-time Markov chains," *ACM Transactions on Computer Logic*, vol. 1, pp. 162–170, July 2000.

[68] D. S. Kim, K. M. Shazzad, and J. S. Park, "A framework of survivability model for wireless sensor network," in *Proceedings of the 1st IEEE International Conference on Availability, Reliability and Security*, pp. 8–12, 2006.

[69] J.-F. Castet and J. H. Saleh, "On the concept of survivability, with application to spacecraft and space-based networks," *Reliability Engineering & System Safety*, vol. 99, pp. 123–138, 2012.

[70] J. P. Sterbenz, D. Hutchison, E. K. Çetinkaya, A. Jabbar, J. P. Rohrer, M. Schöller, and P. Smith, "Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines," *Computer Networks*, vol. 54, no. 8, pp. 1245–1265, 2010.

[71] D. S. Menasché, R. M. Meri Leao, E. de Souza e Silva, A. Avritzer, S. Suresh, K. Trivedi, R. A. Marie, L. Happe, and A. Koziolek, "Survivability analysis of power distribution in smart grids with active and reactive power modeling," *ACM SIGMETRICS Performance Evaluation Review*, vol. 40, pp. 53–57, January 2012.

[72] A. Avritzer, S. Suresh, D. S. Menasché, R. M. M. Leao, E. de Souza e Silva, M. C. Diniz, K. Trivedi, L. Happe, and A. Koziolek, "Survivability models for the assessment of smart grid distribution automation network designs," in *Proceedings of the 4th ACM/SPEC International Conference on Performance Engineering (ICPE)*, (New York, NY, USA), pp. 241–252, 2013.

[73] A. Koziolek, A. Avritzer, S. Suresh, D. Sadoc Menasché, K. Trivedi, and L. Happe, "Design of distribution automation networks using survivability modeling and power flow equations," in *IEEE 24th International Symposium on Software Reliability Engineering (ISSRE)*, pp. 41–50, November 2013.

[74] D. S. Menasché, A. Avritzer, S. Suresh, R. M. Leao, E. de Souza e Silva, M. Diniz, K. Trivedi, L. Happe, and A. Koziolek, "Assessing survivability of smart grid distribution network designs accounting for multiple failures," *Concurrency and Computation: Practice and Experience*, vol. 26, no. 12, pp. 1949–1974, 2014.

[75] I. A. Alobaidi, S. Sedigh Sarvestani, and A. R. Hurson, "Survivability analysis and recovery support for smart grids," in *Proceedings of the 4th International Symposium on Resilient Cyber Systems*, (Chicago, USA), August 2016.

[76] P. Chopade and M. Bikdash, "Modeling for survivability of smart power grid when subject to severe emergencies and vulnerability," in *Proceedings of IEEE Southeastcon*, pp. 1–6, March 2012.

[77] Z. Yi and T. Dohi, "A simulation approach to quantify network survivability on MANETs," in *Proceedings of the 39th IEEE Computer Software and Applications Conference (COMPSAC)*, vol. 3, pp. 268–273, July 2015.

[78] D. R. Cox and D. V. Hinkley, *Theoretical statistics*. CRC Press, 1979.

[79] University of Washington, "Power systems test case archive." http://www.ee.washington.edu/research/pstca/. Retrieved 2017-05-02.

[80] M. Asprou and E. Kyriakides, "Optimal PMU placement for improving hybrid state estimator accuracy," in *IEEE Trondheim PowerTech*, pp. 1–7, June 2011.

[81] N. Acharya and N. Mithulananthan, "Locating series FACTS devices for congestion management in deregulated electricity markets," *Electric Power Systems Research*, vol. 77, pp. 352–360, March 2007.

[82] "EN 50160, voltage characteristics of electricity supplied by public distribution systems," tech. rep., CENELEC, 2005.

[83] Office of Electricity Delivery & Energy Reliability, Department of Energy, "Electric disturbance events (OE-417) annual summaries." https://www.oe.netl.doe.gov/OE417_annual_summary.aspx. Retrieved 2017-05-02.

[84] Y. Song and B. Wang, "Survey on reliability of power electronic systems," *IEEE Transactions on Power Electronics*, vol. 28, no. 1, pp. 591–604, 2013.

[85] "PowerWorld Corporation." http://www.powerworld.com/products/simulator/overview. Retrieved 2017-05-02.

[86] "DIgSILENT GmbH." http://www.digsilent.com/. Retrieved 2017-05-02.

[87] "MATPOWER, a MATLAB power system simulation package." http://www.pserc.cornell.edu/matpower/. Retrieved 2017-05-02.

[88] F. Milano, "An open source power system analysis toolbox," *IEEE Transactions on Power Systems*, vol. 20, pp. 1199–1206, August 2005.

[89] I. Solutions, "Delivering intelligent transport systems driving integration and innovation," tech. rep., IBM Corperation, 2007.

[90] M. Amadeo, C. Campolo, and A. Molinaro, "Information-centric networking for connected vehicles: a survey and future perspectives," *IEEE Communications Magazine*, vol. 54, no. 2, pp. 98–104, 2016.

[91] K. N. Qureshi and A. H. Abdullah, "A survey on intelligent transportation systems," *Middle-East Journal of Scientific Research*, vol. 15, no. 5, pp. 629–642, 2013.

[92] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, pp. 102–114, August 2002.

[93] A. R. Hurson and Y. Jiao, "Database system architecture–A walk through time: From centralized platform to mobile computing," in *Advanced Distributed Systems*, pp. 1–9, Springer, 2005.

[94] P. Bonnet, J. Gehrke, and P. Seshadri, "Towards sensor database systems," in *Mobile Data Management*, pp. 3–14, Springer, 2001.

[95] N. Tsiftes and A. Dunkels, "A database in every sensor," in *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, pp. 316–332, ACM, 2011.

[96] G. Dimitrakopoulos and P. Demestichas, "Intelligent transportation systems," *IEEE Vehicular Technology Magazine*, vol. 5, pp. 77–84, March 2010.

[97] L. Li, D. Wen, and D. Yao, "A survey of traffic control with vehicular communications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, pp. 425–432, February 2014.

[98] V. Milanes, J. Villagra, J. Godoy, J. Simo, J. Perez, and E. Onieva, "An intelligent V2I-based traffic management system," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, pp. 49–58, March 2012.

[99] S. Mitsch, S. Loos, and A. Platzer, "Towards formal verification of freeway traffic control," in *Proceedings of the 3rd IEEE/ACM International Conference on Cyber-Physical Systems*, pp. 171–180, April 2012.

[100] M. Dotoli, M. Fanti, and G. Iacobellis, "A freeway traffic control model by first order hybrid Petri nets," in *Proceedings of the IEEE Conference on Automation Science and Engineering*, pp. 425–431, August 2011.

[101] L. Zhen-Long, "A differential game modeling approach to dynamic traffic assignment and traffic signal control," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, vol. 1, pp. 849–855, October 2003.

[102] W. Qianjiao, L. Rong, and L. Xianglong, "An intelligent traffic control model based on intersection agent," in *Proceedings of the International Conference on Information Engineering and Computer Science*, pp. 1–5, December 2009.

[103] A. Tzes, S. Kim, and W. McShane, "Applications of Petri networks to transportation network modeling," *IEEE Transactions on Vehicular Technology*, vol. 45, pp. 391–400, May 1996.

[104] J. Julvez and R. Boel, "A continuous Petri net approach for model predictive control of traffic systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, vol. 40, no. 4, pp. 686–697, 2010.

[105] H. Dezani, L. Gomes, F. Damiani, and N. Marranghello, "Controlling traffic jams on urban roads modeled in coloured Petri net using genetic algorithm," in *Proceedings of the 38th IEEE Conference on Industrial Electronics Society*, pp. 3043–3048, 2012.

[106] J. Li and Q. Li, "Modeling of urban traffic system based on dynamic stochastic fluid Petri net," in *Proceedings of the Workshop on Power Electronics and Intelligent Transportation Systems*, pp. 485–491, 2008.

[107] J. Wang, C. Jin, and Y. Deng, "Performance analysis of traffic networks based on stochastic timed Petri net models," in *Proceedings of the 5th IEEE International Conference on Engineering of Complex Computer System*, pp. 77–85, 1999.

[108] M. E. Ben-Akiva, S. Gao, Z. Wei, and Y. Wen, "A dynamic traffic assignment model for highly congested urban networks," *Transportation Research Part C: Emerging Technologies*, vol. 24, pp. 62–82, 2012.

[109] S. Lin, B. De Schutter, Y. Xi, and H. Hellendoorn, "An efficient model-based method for coordinated control of urban traffic networks," in *Proceedings of the International Conference on Networking, Sensing, and Control*, pp. 8–13, April 2010.

[110] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results," *Journal of Transportation Engineering*, vol. 129, no. 6, pp. 664–672, 2003.

[111] Y. Wang, Z. Yang, and Q. Guan, "Traffic coordination and control model of regional boundary based on fuzzy control," in *Proceedings of the International Conference on Intelligent Computation Technology and Automation*, vol. 1, pp. 946–950, October 2008.

[112] Y. Mo, T.-H. Kim, K. Brancik, D. Dickinson, H. Lee, A. Perrig, and B. Sinopoli, "Cyber–physical security of a smart grid infrastructure," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 195–209, 2012.

[113] C. C. Aggarwal, N. Ashish, and A. Sheth, "The Internet of things: A survey from the data-centric perspective," in *Managing and Mining Sensor Data*, pp. 383–428, Springer, 2013.

[114] J. L. Cebula and L. R. Young, "A taxonomy of operational cyber security risks," tech. rep., Defense Technical Information Center Document, 2010.

[115] S. Rajasegarar, C. Leckie, and M. Palaniswami, "Anomaly detection in wireless sensor networks," *IEEE Wireless Communications*, vol. 15, no. 4, pp. 34–40, 2008.

[116] S. Yin, G. Wang, and H. R. Karimi, "Data-driven design of robust fault detection system for wind turbines," *Mechatronics*, vol. 24, no. 4, pp. 298–306, 2014.

[117] P. Freeman, P. Seiler, and G. J. Balas, "Air data system fault modeling and detection," *Control Engineering Practice*, vol. 21, no. 10, pp. 1290–1301, 2013.

[118] R. Jurdak, X. R. Wang, O. Obst, and P. Valencia, "Wireless sensor network anomalies: Diagnosis and detection strategies," in *Intelligence-Based Systems Engineering*, pp. 309–325, Springer, 2011.

[119] Y. Zhang, N. Meratnia, and P. Havinga, "Outlier detection techniques for wireless sensor networks: A survey," *IEEE Communications Surveys & Tutorials*, vol. 12, no. 2, pp. 159–170, 2010.

[120] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys*, vol. 41, no. 3, p. 15, 2009.

[121] L. Fang and S. Dobson, "In-network sensor data modelling methods for fault detection," in *Evolving Ambient Intelligence*, pp. 176–189, Springer, 2013.

[122] E.-H. Ngai, J. Liu, and M. Lyu, "On the intruder detection for sinkhole attack in wireless sensor networks," in *Proceedings of the IEEE International Conference on Communications*, vol. 8, pp. 3383–3389, June 2006.

[123] M. Panda and P. M. Khilar, "An efficient fault detection algorithm in wireless sensor network," in *Contemporary Computing*, pp. 279–288, Springer, 2011.

[124] L. Fang and S. Dobson, "Unifying sensor fault detection with energy conservation," in *Self-Organizing Systems*, pp. 176–181, Springer, 2014.

[125] S. Rajasegarar, C. Leckie, M. Palaniswami, and J. C. Bezdek, "Distributed anomaly detection in wireless sensor networks," in *Proceedings of the 10th IEEE International Conference on Communication systems*, pp. 1–5, 2006.

[126] M. Chang, A. Terzis, and P. Bonnet, "Mote-based online anomaly detection using echo state networks," in *Distributed Computing in Sensor Systems*, pp. 72–86, Springer, 2009.

[127] O. Obst, "Distributed back propagation decorrelation learning," in *Proceedings of the Neural Information Processing Systems Workshop: Large-Scale Machine Learning: Parallelism and Massive Datasets*, December 2009.

[128] C. Mayfield, J. Neville, and S. Prabhakar, "Eracer: a database approach for statistical inference and data cleaning," in *Proceedings of the ACM International Conference on Management of Data*, pp. 75–86, ACM, 2010.

[129] X. R. Wang, J. T. Lizier, O. Obst, M. Prokopenko, and P. Wang, "Spatiotemporal anomaly detection in gas monitoring sensor networks," in *Wireless Sensor Networks*, pp. 90–105, Springer, 2008.

[130] K. Ni and G. Pottie, "Sensor network data fault detection with maximum a posteriori selection and Bayesian modeling," *ACM Transactions on Sensor Networks*, vol. 8, no. 3, p. 23, 2012.

[131] J. W. Branch, C. Giannella, B. Szymanski, R. Wolff, and H. Kargupta, "In-network outlier detection in wireless sensor networks," *Knowledge and Information Systems*, vol. 34, no. 1, pp. 23–54, 2013.

[132] N. Chitradevi, V. Palanisamy, K. Baskaran, and U. B. Nisha, "Outlier aware data aggregation in distributed wireless sensor network using robust principal component analysis," in *Proceedings of the IEEE International Conference on Computing Communication and Networking Technologies*, pp. 1–9, 2010.

[133] X. Ying-Xin, C. Xiang-Guang, and Z. Jun, "Data fault detection for wireless sensor networks using multi-scale PCA method," in *Proceedings of the 2nd IEEE International Conference on Artificial Intelligence, Management Science and Electronic Commerce*, pp. 7035–7038, 2011.

[134] E. U. Warriach, T. A. Nguyen, M. Aiello, and K. Tei, "A hybrid fault detection approach for context-aware wireless sensor networks," in *Proceedings of the 9th IEEE International Conference on Mobile Adhoc and Sensor Systems*, pp. 281–289, 2012.

[135] L.-A. Tang, X. Yu, S. Kim, Q. Gu, J. Han, A. Leung, and T. La Porta, "Trustworthiness analysis of sensor data in cyber-physical systems," *Journal of Computer and System Sciences*, vol. 79, no. 3, pp. 383–401, 2013.

[136] H.-J. Liao, C.-H. Richard Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 16–24, 2013.

[137] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan, "A survey of intrusion detection techniques in cloud," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 42–57, 2013.

[138] L. Buttyán, D. Gessner, A. Hessler, and P. Langendoerfer, "Application of wireless sensor networks in critical infrastructure protection: Challenges and design options," *IEEE Wireless Communications*, vol. 17, no. 5, pp. 44–49, 2010.

[139] R. Mitchell and I.-R. Chen, "A survey of intrusion detection techniques for cyber-physical systems," *ACM Computing Surveys*, vol. 46, no. 4, p. 55, 2014.

[140] S. Gantayat, A. Misra, and B. Panda, "A study of incomplete data–a review," in *Proceedings of the International Conference on Frontiers of Intelligent Computing: Theory and Applications*, pp. 401–408, Springer, 2014.

[141] M. Mezzanzanica, R. Boselli, M. Cesarini, and F. Mercorio, "A model-based approach for developing data cleansing solutions," *Journal of Data and Information Quality*, vol. 5, no. 4, 2015.

[142] T. Gschwandtner, W. Aigner, S. Miksch, J. Gärtner, S. Kriglstein, M. Pohl, and N. Suchy, "Timecleanser: A visual analytics approach for data cleansing of time-oriented data," in *Proceedings of the 14th ACM International Conference on Knowledge Technologies and Data-driven Business*, 2014.

[143] F. Ayatolahi, B. Sangchoolie, R. Johansson, and J. Karlsson, "A study of the impact of single bit-flip and double bit-flip errors on program execution," in *Computer Safety, Reliability, and Security*, pp. 265–276, Springer, 2013.

[144] B. Sangchoolie, F. Ayatolahi, R. Johansson, and J. Karlsson, "A study of the impact of bit-flip errors on programs compiled with different optimization levels," in *Proceedings of the 10th IEEE European Dependable Computing Conference*, 2014.

[145] M. Beccuti, S. Chiaradonna, F. D. Giandomenico, S. Donatelli, G. Dondossola, and G. Franceschinis, "Quantification of dependencies between electrical and information infrastructures," *International Journal of Critical Infrastructure Protection*, vol. 5, no. 1, pp. 14–27, 2012.

[146] C. Girault and R. Valk, *Petri nets for System Engineering: A Guide to Modeling, Verification, and Applications*. Springer, 2001.

[147] M. K. Molloy, "Discrete time stochastic Petri nets," *IEEE Transactions on Software Engineering*, vol. SE-11, pp. 417–423, April 1985.

[148] S. R. Islam, K. M. Muttaqi, and D. Sutanto, "A decentralized multiagent-based voltage control for catastrophic disturbances in a power system," *IEEE Transactions on Industry Applications*, vol. 51, no. 2, pp. 1201–1214, 2015.

[149] F. Pommereau, "SNAKES: a flexible high-level Petri nets library," in *Proceedings of Petri nets*, vol. 9115 of *Lecture Notes in Computer Science*, pp. 254–265, Springer, 2015.

VITA

Mark James Woodard was born in Walnut Creek, California. In May 2008, he received a Bachelor's of Science degree in Electrical and Computer Engineering from the Virginia Military Institute in Lexington, Virginia. Upon graduation, Mark served as a combat engineering officer in the US Army for four years. He began working toward his Ph.D. in Computer Engineering at the Missouri University of Science and Technology in August 2012 and received the degree in July 2017. During his time as a graduate student, Mark taught digital network design, circuits analysis I, and multiple sections of the calculus laboratory for engineers I and III. In addition to teaching and research, Mark completed a summer internship at Sandia National Labs while working towards his doctoral degree.