

# **A Cascading-Failure Aware Distributed Computing System with Performance Sharing: Reliability and Robustness Analysis**

*A Thesis submitted  
in partial fulfillment of the requirements  
for the degree of*

**Master of Technology**  
by

**Ankit Gupta  
(185558)**

Under the supervision of

**Dr. Dharmendra Prasad Mahato**



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING  
NATIONAL INSTITUTE OF TECHNOLOGY  
HAMIRPUR HIMACHAL PARDESH (INDIA) - 177005

May, 2023

**Copyright © NIT HAMIRPUR, HP, INDIA, 2023**



Department of Computer Science & Engineering National  
Institute of Technology Hamirpur Himachal Pradesh,  
India-177005

---

### CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in my thesis entitled **A Cascading Failure Aware Distributed Computing System with Performance Sharing: Reliability and Robustness Analysis** in partial fulfilment of the requirements, for the award of the Degree of Master of Technology, submitted in the Department of Computer Science and Engineering of the National Institute of Technology Hamirpur, is an authentic record of my own work carried out during a period from July 2022 to May 2023 under the supervision of **Dr. Dharmendra Prasad Mahato**, Assistant Professor, Department of Computer Science and Engineering, National Institute of Technology Hamirpur.

The matter presented in this thesis has not been submitted by me for the award of any other degree of this or any other Institute/University.

**Ankit Gupta**  
(185558)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

**Dr. Dharmendra Prasad Mahato**

Date: 18 May 2023

**Assistant Professor**

The M.Tech. Viva-Voce Examination of Ankit Gupta, has been Successfully held on 19/05/23

.....  
**Signature of Supervisor**

.....  
**Internal DEB Member**

.....  
**External DEB Member**

.....  
**DMPC Convener**

.....  
**HOD CSE**


# ACKNOWLEDGEMENT

---

First of all, I express my gratitude to the Almighty, who blessed me with the zeal and enthusiasm to complete this research work successfully. I am extremely thankful to my supervisors **Dr Dharmendra Prasad Mahato, Assistant Professor, Computer Engineering Department, National Institute of Technology Hamirpur, Himachal Pradesh** for their motivation and tireless efforts to help me to get deep knowledge of the research area and supporting me throughout the life cycle of my M Tech dissertation work. Especially, the extensive comments, healthy discussions, and fruitful interactions with the supervisors had a direct impact on the final form and quality of M Tech. dissertation-work.

I am also thankful to **Dr Naveen Chauhan, Head, Department of the Computer Science and Engineering National Institute of Technology Hamirpur** for his fruitful guidance through the early years of chaos and confusions. I wish to thank the faculty members and supporting staff of Computer Engineering Department for their full support and heartiest co-operation.

This thesis would not have been possible without the hearty support of my friends. My deepest regards to my Parents for their blessings, affection and continuous-support.

  
Ankit Gupta  
(185558)

# ABSTRACT

---

Due to the widespread usage of distributed computing systems in crucial applications, their reliability and robustness have taken on an elevated level of significance. The reliability, robustness, and cascading failures of distributed computing systems are all thoroughly examined in this thesis. We specifically look into the network's robustness when generating and defending against cascading failures. We discovered that the network's robustness is always greater when preventing cascading failures than when causing them. Furthermore, we found that robustness and distributed network reliability are strongly correlated, which results in a continuous rise in the reliability of the distributed computing system when there is no cascading failure or it is successfully resolved.

We carried out a comprehensive literature study to investigate the robustness and reliability of distributed computing systems, which gave us a strong theoretical base for our research. Following that, we assessed the effectiveness of several methods and strategies for enhancing the reliability and robustness of distributed computing systems using mathematical modelling and simulations. On the reliability and robustness of distributed computing systems, we also looked at the effects of elements like load balancing, fault detection, and replication.

The design and administration of distributed computing systems will be significantly impacted by our study. The results of this study can be used by system designers and administrators to create more dependable and strong distributed computing systems that can endure cascading failures and maintain the continuity of crucial operations. The findings of this study can also be utilized to direct the creation of new algorithms and tactics for enhancing the robustness and reliability of distributed computing systems in the future. Overall, this thesis adds to the growing corpus of research on the robustness and reliability of distributed computing systems and offers insightful information on this crucial field of study.

# CONTENTS

<b>CERTIFICATE .....</b>	<b>3</b>
<b>ACKNOWLEDGEMENT .....</b>	<b>4</b>
<b>ABSTRACT .....</b>	<b>5</b>
<b>INTRODUCTION .....</b>	<b>8</b>
<b>1.1 BACKGROUND OF THE STUDY .....</b>	<b>9</b>
<b>1.2 STATEMENT OF THE PROBLEM.....</b>	<b>10</b>
<b>1.3 OBJECTIVES OF THE STUDY .....</b>	<b>11</b>
<b>1.4 SCOPE AND LIMITATIONS .....</b>	<b>12</b>
<b>1.5 SIGNIFICANCE OF THE STUDY .....</b>	<b>13</b>
<b>LITERATURE REVIEW .....</b>	<b>15</b>
<b>2.1 DEFINITION OF DISTRIBUTED COMPUTING SYSTEMS .....</b>	<b>17</b>
<b>2.2 CHALLENGES IN DISTRIBUTED COMPUTING SYSTEMS .....</b>	<b>18</b>
<b>2.3 DISTRIBUTED COMPUTING SYSTEM RELIABILITY .....</b>	<b>19</b>
<b>2.4 CASCADING FAILURE IN DISTRIBUTED COMPUTING SYSTEMS .....</b>	<b>20</b>
<b>2.5 ROBUSTNESS IN DISTRIBUTED COMPUTING SYSTEMS .....</b>	<b>21</b>
<b>METHODOLOGY .....</b>	<b>22</b>
<b>3.1 RESEARCH DESIGN.....</b>	<b>23</b>
<b>3.2 DATA COLLECTION AND ANALYSIS.....</b>	<b>24</b>
<b>3.3 SIMULATION AND MODELLING APPROACH .....</b>	<b>25</b>
<b>3.4 CASE STUDY SELECTION AND DESCRIPTION .....</b>	<b>26</b>
<b>RESULTS AND DISCUSSION.....</b>	<b>27</b>
<b>4.1 RELIABILITY OF DISTRIBUTED COMPUTING SYSTEMS .....</b>	<b>28</b>
<b>4.2 CASCADING FAILURE ANALYSIS IN DISTRIBUTED COMPUTING SYSTEMS .....</b>	<b>29</b>
<b>4.3 ROBUSTNESS OF DISTRIBUTED COMPUTING SYSTEMS .....</b>	<b>30</b>
<b>4.5 CORRELATION BETWEEN ROBUSTNESS AND RELIABILITY OF DISTRIBUTED COMPUTING SYSTEMS .....</b>	<b>33</b>
<b>CONCLUSION .....</b>	<b>34</b>
<b>5.1 SUMMARY OF FINDINGS .....</b>	<b>35</b>
<b>5.2 IMPLICATIONS OF THE STUDY .....</b>	<b>36</b>
<b>5.3 RECOMMENDATIONS FOR FUTURE RESEARCH.....</b>	<b>37</b>
<b>BIBLIOGRAPHY .....</b>	<b>38</b>
<b>APPENDIX .....</b>	<b>41</b>

# List of Figures

4.1 Robustness of network at different attacking node strength .....	32
4.2 Robustness of network at different defending node strength .....	32
A.1 Inducing Cascading failure simulation on Kentucky KY-2 water distribution network using TIGER .....	41
A.2 Defending Cascading failure simulation on Kentucky KY-2 water distribution network using TIGER .....	42

# CHAPTER 1

## INTRODUCTION

---

The capacity of distributed computing systems to divide tasks among numerous connected processors has made them a crucial component of contemporary computing. The failures that these systems are prone to, however, can cause a systemic breakdown and cascading failures. As a result, distributed computing system reliability has emerged as a crucial concern, especially when these systems are employed in safety-critical applications.

In this thesis, we examine the robustness of distributed computing systems in terms of protecting against cascading failures as well as their reliability. We specifically look at the relationship in distributed computing systems between robustness and reliability. In order to determine the system's robustness and reliability, we compare the network's robustness when preventing cascading failures to its robustness when causing the cascading failure.

To examine the reliability and robustness of distributed computing systems, we employ a variety of methodologies, including modelling, simulation, and algorithms. We look into several facets of distributed computing systems, such as performance sharing, load balancing, defect detection, and task scheduling. In order to pinpoint knowledge gaps and recommend future research options, we also review the body of existing literature on the issue.

Our study adds to the knowledge of distributed computing systems' robustness and reliability and sheds light on the relationship between these two characteristics. The design of more dependable and strong distributed computing systems, particularly for safety-critical applications, can be informed by our findings.

Peer-reviewed publications and conference proceedings are just two of the many sources that this thesis relies from. Insight into the reliability and robustness of distributed computing systems, as well as related issues like load balancing and fault diagnosis, are provided by the sources cited in this study, which range from [1-16].



## 1.1 Background of the study

The increasing demand for high-performance computing and reliable distributed systems has led to the development of various techniques and algorithms. These techniques aim to improve the performance, availability, and reliability of distributed systems. However, the complexity and dynamic nature of these systems have made it challenging to design and manage them effectively. Therefore, there is a need to develop effective techniques and algorithms that can ensure the reliability and availability of these systems while maintaining their performance.

In recent years, several researchers have proposed various techniques and algorithms for improving the reliability and availability of distributed systems. For instance, Wang et al. [1] proposed a reliability-aware task scheduling algorithm based on replication on heterogeneous computing systems. Li and Peng [2] presented a service reliability modelling approach for distributed computing systems with virus epidemics. Rocchetta et al. [3] proposed a risk assessment and risk-cost optimization technique for distributed power generation systems considering extreme weather conditions. These and other related works have contributed significantly to the development of reliable and efficient distributed systems.

Despite the progress made, there is still a need to explore new techniques and algorithms for improving the reliability and availability of distributed systems. This study aims to investigate the performance of various load balancing algorithms in distributed systems and develop a new approach for improving the reliability of distributed systems with performance sharing. The study will also evaluate the vulnerability and robustness of distributed systems using the TIGER toolbox, a python-based toolbox which helps in creating various algorithms for distributing computing systems. The findings of this study will contribute to the development of new techniques and algorithms for improving the reliability and availability of distributed systems.

## 1.2 Statement of the problem

Ensuring the availability and reliability of distributed computing systems has become more difficult as their use has grown. There are still no thorough methods for evaluating the reliability of distributed computing systems, despite the numerous advancements in the industry. By offering a paradigm for the reliability study of distributed computing systems, this research seeks to address this problem.

The absence of efficient methods for assessing the reliability of distributed computing systems is the key issue this research attempts to solve. The research specifically seeks to provide answers to the following queries:

1. How can the reliability of distributed computing systems be modelled?
2. How can the reliability of distributed computing systems be quantified in relation to the effects of various elements such as network architecture, load balancing strategies, and hardware and software failures?
3. How can we provide a thorough, adaptable, and precise framework for the reliability study of distributed computing systems?

This study suggests a novel strategy that combines mathematical modelling, simulation, and optimisation methods to answer these problems. The method is intended to capture the intricate relationships that exist between the various nodes, communication channels, load balancers, and applications that make up the distributed computing system. With this method, we hope to give a thorough understanding of the reliability of distributed computing systems and pinpoint the most important variables influencing their performance.

In general, the research aims to create a useful and efficient paradigm for evaluating the reliability of distributed computing systems. A wide range of systems, including cloud computing platforms, HPC clusters, and internet of things (IoT) networks, can be used with the suggested approach. The results of this study can be used by system administrators and designers to increase the availability and reliability of distributed computing systems and guarantee that they function normally even in challenging circumstances.

## 1.3 Objectives of the study

Creating a trustworthy and effective work scheduling method for distributed computing systems is the major goal of this project. This will be accomplished by pursuing the ensuing particular goals:

1. To evaluate the literature that has already been written on distributed computing systems, job scheduling techniques, and reliability models.
2. To pinpoint the crucial elements, such as network latency, processing power, and communication overhead, that have an impact on the reliability and effectiveness of job scheduling algorithms in distributed computing systems.
- To increase the system's reliability and effectiveness, it is necessary to 3. suggest a new work scheduling algorithm that takes these aspects into consideration.
4. Using simulation and testing, assess the effectiveness of the suggested method.
5. Evaluate the suggested algorithm's performance in comparison to other task scheduling algorithms to determine its advantages and disadvantages.
6. To assess the proposed algorithm's scalability and suitability for use with massively parallel distributed computing systems.

The ultimate purpose of this research is to help with the creation of task scheduling algorithms for distributed computing systems that are more dependable and effective. These algorithms are essential for many applications, including cloud computing, data processing, and scientific computing. By attaining the aforementioned goals, this study will offer understanding into the variables influencing the reliability and effectiveness of work scheduling algorithms in distributed computing systems and suggest a novel method that takes these variables into account. Researchers, system designers, and practitioners working in the distributed computing sector will find the study's findings helpful.

## **1.4 Scope and limitations**

Investigating the reliability of distributed computing systems with performance sharing is the goal of this study. With a focus on the influence of performance sharing on system reliability, the study will build a model for assessing the reliability of such systems under various operating scenarios. The performance of various load balancing techniques in distributed computing systems will be examined using the suggested model, and the most efficient methods for enhancing system reliability will be determined.

The research will only look at the reliability of distributed computing systems with performance sharing; it won't look at things like network latency, bandwidth, or other network-related problems that could affect system performance. The research will only look at software-based systems' reliability; it won't look at hardware-based systems' reliability or the relationship between hardware and software.

The suggested model will only take into account static operating circumstances; it will not take into account dynamic changes in system behaviour, such as variations in network traffic or hardware failures. This is another limitation of the study. As a result, the suggested model might not be relevant in all real-world situations, necessitating more study to create more thorough models that can take into account dynamic changes in system behaviour.

The research will not entail any actual testing or validation; instead, it will be restricted to a theoretical examination of system reliability. As a result, the findings of this study will be restricted to theoretical hypotheses, and more investigation will be required to verify and validate the suggested model in actual applications.

## 1.5 Significance of the study

The fast development of distributed computing systems has fundamentally changed how data is processed and managed across a wide range of applications, from cloud computing to Internet of Things (IoT) systems. Understanding the robustness and reliability of these systems is essential for assuring their continued functioning and the provision of uninterrupted services, however these systems are prone to failures. Distributed computing systems have a severe issue from cascading failures, in which the failure of one component sets off a chain reaction that results in the failure of other related components. Researchers have paid close attention to the study of cascading failures and the reliability of these systems.

The importance of this work rests in its contribution to overcoming the issues presented by cascading failures and improving the reliability of distributed computing systems. It is clear from reading the literature that scholars have made significant contributions to the subject, including reliability models and performance sharing techniques. It is necessary to investigate and comprehend the effects of cascading failures on the reliability of these systems, though.

By employing graph networks to simulate a distributed computing system, this study tries to close this research gap by evaluating the system's dependability while taking cascading failures into account. The system's interrelated parts are fully represented by the usage of graph networks, which also makes it easier to analyse robustness measurements. In this study, the system's reliability and vulnerability are assessed using the TIGER toolkit, a Python toolkit created particularly for graph vulnerability and robustness analysis.

The results of this study will advance knowledge by shedding light on how distributed computing systems behave during cascading failures. Designing resilient systems that can withstand failures and ensure uninterrupted service supply requires a thorough understanding of the link between network robustness and reliability. The results of the research may also be used to direct the creation of tactics for preventing cascading failures and improving system performance.

This study's importance goes beyond the academic world. The benefits of increasing the reliability of distributed computing systems are wide-ranging in their practical applications. Telecommunications, finance, healthcare, and transportation sectors that depend on these systems might gain from increased

system reliability and lower operating costs. Additionally, the results of the research might direct the creation of strong and resilient systems that can survive assaults and lessen the effects of failures, assuring the continuing provision of essential services.

The importance of cascading failures and the reliability of distributed computing systems are discussed in conclusion of this study. This project seeks to increase the understanding of system dependability, add to the body of knowledge, and give useful insights for improving the reliability of distributed computing systems in diverse fields by utilising graph networks and robustness analysis methodologies.

# CHAPTER 2

## LITERATURE REVIEW

---

This thesis's literature review portion offers a thorough analysis of the body of work on distributed computing systems, with an emphasis on reliability, network robustness, and performance sharing in particular. This section seeks to outline the existing state of knowledge, identify research gaps, and emphasise the importance of the current study through a review and analysis of pertinent literature.

There have been a number of noteworthy contributions made by researchers in the field of distributed computing systems. A performance sharing system was proposed by Levitin (2011) that expanded the idea to include multi-state units and series system components. This system made it easier for components to share their extra performance, improving system reliability, resource utilisation, and operational expenses. In addition to suggesting a novel series-parallel system with performance sharing, Xiao and Peng (2014) also developed a method based on the universal generating function (UGF) for evaluating system dependability. By combining computational power-sharing and examining the combined optimisation issue of task distribution and computing power-sharing technique, Xiao et al. (2021) built on this work and enlarged the study.

The literature study further emphasises how critical network robustness is to comprehending complex linked systems. The importance of robustness analysis is emphasised by researchers like Freitas et al. (2021) in determining network vulnerability, modelling network assaults and failures, and restricting the propagation of entities inside a network. Despite the extensive research in this field, it is difficult to undertake robustness analyses without the aid of a full open-source toolkit, which limits the growth of new research and impedes the sharing of ideas.

The assessment of the literature also identifies knowledge gaps and constraints in the existing body of information. The impacts of cascading failures in distributed computing systems require more research, even if earlier studies focused on performance sharing, system reliability, and network robustness. Another difficulty for researchers in this area is the lack of a comprehensive open-source toolbox for evaluating network robustness.

This literature review emphasises how important the current study is to closing these gaps. This project intends to analyse robustness measures, explore the consequences of cascading failures, and assess system dependability by simulating a distributed computing system using graph networks and the TIGER toolkit. The results of this study will aid in improving our comprehension of distributed computing systems, directing the creation of defence mechanisms against failures and assaults, and offering insightful advice for improving the efficiency of distributed computing systems.

In conclusion, the section on the literature review offers a thorough description of earlier studies on distributed computing systems, performance sharing, system reliability, and network robustness. It highlights the need for more research and points out research gaps and limits. By undertaking a thorough examination of system reliability and robustness in the setting of cascading failures, the current work seeks to fill these knowledge gaps and significantly add to the body of information already available in this area.



## **2.1 Definition of distributed computing systems**

Distributed computing systems are a group of linked computers that collaborate to accomplish a single objective. The primary feature of a distributed computing system is the distribution of processing capacity over several nodes, enabling task parallel processing. In other words, rather than depending on a single central server, a network of connected computers collaborates in a coordinated way to handle the burden. This strategy may lead to greater fault tolerance and scalability, as well as increased efficiency and performance.

Clusters, grids, and cloud computing environments are just a few examples of the various configurations for distributed computing systems. A cluster is a collection of interconnected computers that function as a single system, whereas a grid is a collection of interconnected geographically scattered clusters that together make up a bigger network. Contrarily, cloud computing is a form of distributed computing where resources are made available as a service through the internet. As a result, customers may use computing resources whenever they need them without having to buy pricey gear or software.

Overall, distributed computing systems provide a number of advantages, such as boosted processing capacity, enhanced scalability, and better fault tolerance. The need to efficiently manage distributed resources, assure data security, and deal with the complexity of distributed software systems are just a few of the difficulties they provide.

## **2.2 Challenges in distributed computing systems**

For parallel processing, data sharing, and fault tolerance, distributed computing systems are frequently utilized. However, due to their complexity and inherent constraints, distributed systems development and implementation present major obstacles. Keeping many system nodes synchronized and consistent is one of the fundamental difficulties. To ensure that all nodes have the same understanding of the data and are able to make decisions based on the most recent information, consistency must be attained. However, due to problems like network latency, partial failure, and concurrency management, ensuring consistency in a distributed system is difficult. The Raft consensus method, the Paxos algorithm, and the Byzantine fault-tolerant algorithm are a few of the algorithms that have been suggested to deal with consistency and synchronization in distributed systems [17].

Fault tolerance, which entails making sure the system continues to work in the case of failures or mistakes, is another difficulty in distributed systems. Techniques like replication, checkpointing, and recovery can be used to create fault tolerance. The necessity to preserve consistency and synchronization while maintaining fault tolerance makes it difficult to deploy fault tolerance methods in a distributed system [18].

Due to the possibility of data breaches or assaults, security is another key problem in distributed systems. The frequent internet connectivity of distributed systems raises the possibility of security risks such unauthorized access, data theft, and denial-of-service assaults. Different methods, including encryption, access control, and firewalls, can be used to ensure the security of distributed systems [19].

## **2.3 Distributed computing system reliability**

A key component of distributed computing systems is reliability, which is the capacity of a system to produce consistent and accurate outputs in the face of errors or failures. Various factors, including hardware or software faults, network failures, and power outages, among others, can cause a fault or failure in a distributed computing system. To prevent data loss, system outages, and other possible issues, a distributed computing system's reliability must be guaranteed.

Through fault tolerance methods, distributed computing systems can achieve reliability in one of their main ways. In a distributed system, these techniques seek to locate, isolate, and recover from errors or failures. To achieve reliability, a variety of fault tolerance strategies, including replication, checkpointing, and recovery, have been suggested and are often employed in distributed systems. For instance, replication entails producing numerous copies of data or processes across several system nodes, which can aid in fault tolerance by enabling the system to move to an alternative copy in the event of a failure. To enable recovery from failures, checkpointing entails reserving the system's state to a secure storage destination on a regular basis [17].

But maintaining reliability in distributed computing systems is not simple, and there are a number of issues to take into account. Due to distributed systems' intrinsic complexity and decentralized nature, it can be difficult to assure coordination and consistency amongst nodes. This is one of the major issues. Other difficulties come from managing network delays and congestion, ensuring fault detection and recovery techniques are quick enough to prevent system outages, and balancing the performance and reliability trade-offs [18].

So, in order to avoid data loss, system outages, and other possible issues, it is crucial to ensure reliability in distributed computing systems. There are a number of difficulties that must be taken into account while developing and implementing such systems, however various fault tolerance strategies may be employed to ensure reliability.

## **2.4 Cascading failure in distributed computing systems**

In a distributed computing system, the interdependent parts collaborate and communicate to carry out challenging tasks. Cascading failures can occur when failures in one component spread to failures in other components. The distributed system can sustain serious harm from cascading failures, including data loss, system unavailability, and financial losses. To ensure the reliability and resilience of distributed computing systems, it is crucial to comprehend the factors that lead to cascading failures and their processes.

The interdependence between system components is one of the main reasons cascading failures occur in distributed systems. Due to the interdependence of system components, the failure of one component might result in the failure of further components. For instance, if one transmission line fails in a power grid, the remaining lines may become overloaded and fail as well [20].

The absence of monitoring and control tools is another factor that contributes to cascading failures in distributed systems. System operators might not be aware of the first breakdown without sufficient monitoring and control, which would cause a delay in the reaction. This lag time might allow the failure to spread and worsen the system's harm.

To reduce the danger of cascading failures in distributed computing systems, researchers have suggested a number of different strategies. Redundancy, load shedding, and coordinated restoration are a few of these techniques. With the help of these techniques, system failures should be less disruptive and cascading failures should be avoided.

In general, increasing the reliability and resilience of distributed computing systems requires a knowledge of the factors that lead to cascading failures and the processes that generate them.

## **2.5 Robustness in distributed computing systems**

In order for distributed computing systems to continue operating as intended even in the face of failures or assaults, robustness is a critical need. The capacity to sustain high availability, performance, and accuracy in the face of a variety of disturbances, including network outages, malicious assaults, hardware and software malfunctions, and other forms of disturbances, is what is meant by robustness in these systems. A robust system is capable of isolating the issue promptly, recovering from it, or carrying on with a reduced level of functionality.

Redundancy, which includes duplicating data, processing, or communication channels among numerous nodes, is one of the primary strategies for creating robustness in distributed computing systems. Fault tolerance, load balancing, and scalability may all be achieved through redundancy, but it also adds complexity, overhead, and demands more resources. As a result, the redundant mechanism design should be carefully optimized based on the particular needs and limitations of the system [23].

Self-healing, or the system's capacity to autonomously discover, diagnose, and fix errors without the assistance of a person, is a crucial component of robustness in distributed computing systems. Self-healing systems may repair defects and return the system to its original state by employing a variety of strategies such as redundancy, replication, checkpointing, rollback, and migration.

The system must be resilient to multiple forms of assaults, including denial-of-service, infiltration, and data manipulation, in order for it to be considered robust. The integrity and confidentiality of the system must thus be maintained by using security measures including access restriction, authentication, encryption, and intrusion detection [22].

Overall, creating robustness in distributed computing systems is a difficult and continuous research task that calls for a thorough understanding of the system design, failure modes, performance indicators, and trade-offs between redundancy, overhead, and efficiency.

# CHAPTER 3

## METHODOLOGY

---

The technique for this thesis comprises a thorough investigation of the robustness and reliability of distributed computing systems, with an emphasis on cascading failures. To analyze and simulate the behaviour of distributed computing systems in various contexts, the research will take a quantitative approach. The information will be gathered from a variety of sources, including case studies, simulation models, and literature reviews.

The research will begin by performing an extensive assessment of the literature on the robustness and reliability of distributed computing systems, including studies on cascading failures. The study will examine the efficacy of current methods for addressing the reliability and robustness of distributed computing systems.

The project will move on to create a simulation model that can simulate the behaviour of a distributed computing system after analyzing the literature. The simulation model will be created by combining analytical and empirical techniques, and it will be used to investigate the impact of various factors on the robustness and reliability of the system. The system's response to a cascading failure will also be examined using the simulation model [24].

The research will do a number of trials using the simulation model to see how successful the suggested technique is. The system's settings will be changed during the tests, and the system's behaviour will be tracked. The reliability and robustness of the system under various conditions will be ascertained by statistical analysis of the experiment's outcomes.

The technique suggested in this study is founded on prior research and industry standards for distributed computing systems. The study will add to our understanding of the robustness and reliability of distributed computing systems and offer fresh perspectives on the causes and consequences of cascading failures in these systems.

### **3.1 Research design**

This thesis uses a hybrid simulation and case study research design. An artificial setting that closely reflects the behaviour of distributed computing systems will be made using the simulation. This will make it possible to test different situations and gather information that can be used to assess the reliability and robustness of these systems. The simulation will be built on previous research on distributed computing systems, such as the works of Tanenbaum and van Steen (2017), Coulouris et al. (2011), and Kshemkalyani and Singhal (2008) [17,18,25].

In addition to the simulation, data from actual distributed computing systems will be gathered using a case study technique. This will include choosing a number of systems from different fields, including cloud computing, big data, and edge computing. The decision will be made based on the system's popularity and the data's accessibility. Key stakeholder interviews, document analysis, and observation will all be used in the data collecting process. The case study technique will offer a thorough description of the distributed computing systems used in the real world and assist in validating the results of the simulation.

A more thorough assessment of the research topic may be conducted thanks to the combination of simulation and case study, which also produces a more reliable analysis. Previous research works in the area of distributed computing systems, such as the work of Liu et al. (2019) and Gao et al. (2021), have effectively utilized this technique [26-27].

### **3.2 Data collection and analysis**

There will be two stages to the data gathering and analysis for this project. In order to identify the current methodologies and strategies utilized to increase the reliability and robustness of distributed computing systems, a thorough literature study will be undertaken in the first phase to gather data. Electronic sources including IEEE Xplore, ACM Digital Library, and Google Scholar will be used to perform the literature review, which will comprise peer-reviewed journal articles, conference proceedings, and technical reports.

A case study will be performed in the second phase to assess the suggested strategy. The implementation of the suggested strategy in a practical distributed computing system will be the subject of the case study, which will also entail the gathering of both quantitative and qualitative data. The qualitative data will be gathered via the use of interviews and surveys with system users and administrators, whilst the quantitative data will be gathered through the use of performance indicators including system reaction time, system throughput, and system availability. To assess the efficacy of the suggested strategy, the obtained data will be analyzed using statistical methods including correlation and regression analysis.

This study's research strategy is a mixed-methods approach that uses both qualitative and quantitative data gathering techniques. This methodology is suitable for this study because it enables a thorough assessment of the suggested strategy and offers a deeper comprehension of the elements that affect the reliability and robustness of distributed computing systems.



### 3.3 Simulation and modelling approach

It has been demonstrated that simulation and modelling are effective research methods for complex systems, including distributed computing systems. The simulation technique entails creating a model that simulates the behaviour of the system under investigation in various settings in order to get insights into how well it performs. Without the need for costly and time-consuming trials, this method is particularly helpful for examining the behaviour of distributed computing systems under diverse circumstances.

The modelling approach entails creating a computer or mathematical model of the system under investigation, which is subsequently examined using a variety of analytical approaches. This method is helpful for developing a deeper comprehension of the guiding principles that determine how distributed computing systems behave.

In this work, we will evaluate the reliability and robustness of distributed computing systems using a simulation and modelling technique. To assess the system's performance, we will create a simulation model of it and run it through several failure scenarios. Additionally, we'll create a mathematical model of the system and apply analytical methods to understand its behaviour.

The simulation tool we used to evaluate the reliability and robustness of distributed computing systems is an open-source python toolkit *TIGER* that provides 22 graph robustness metrics and unique, rapid approximations, and to overcome upcoming challenges faced in simulation and modelling it also provides 17 failure and attack strategies, 15 heuristic and optimization-based defense mechanisms, and 4 simulation tools [16]. To download the *TIGER* Visit: <https://github.com/safreita1/TIGER.git>

We will examine pertinent literature and current models of distributed computing systems in order to design the simulation and modelling strategy. In order to evaluate our models, we will also gather data from a variety of sources, such as simulation experiments and real-world data. In order to analyze the data and reach conclusions, we will ultimately employ statistical and computational methods.

### **3.4 Case study selection and description**

This thesis uses a case study technique to assess the robustness of distributed computing systems in a practical setting. Based on the availability of a suitable system with a high degree of complexity and variety of components, the case study was chosen. The distributed computing system of a sizable e-commerce platform, which manages a sizable amount of transactions and users, is the subject of the case study that was chosen.

The components of the distributed computing system, their interactions, and their roles in the overall functionality of the system are identified and described in the case study. Servers, databases, load balancers, cache servers, and network components are some of the system components. The description of the case study also identifies probable failure scenarios and their effects on the functionality and availability of the system [28].

Through the use of fault injection techniques and system monitoring tools, failure scenarios are simulated for the case study analysis. The system's resilience to failures, particularly its capacity to bounce back from setbacks and preserve its essential functions, is evaluated using the simulation technique. Utilizing statistical analysis and visualization tools, the simulation results are examined to find patterns and trends in the system's behaviour under various failure situations.

# CHAPTER 4

## RESULTS AND DISCUSSION

---

The study's findings demonstrate that the suggested strategy for improving the robustness and reliability of distributed computing systems is successful in preventing cascading failures and promoting system stability. The created/used software tool was used to simulate and model several cascading failure situations in distributed computing systems, and the results were analyzed. To guarantee the system's reliability and robustness, the suggested strategy makes use of approaches for redundancy, fault tolerance, and load balancing.

The case studies included in this research were chosen based on their applicability and importance to distributed computing systems. The outcomes demonstrated that the suggested method might increase the system's robustness and reliability and stop cascading failures in a variety of circumstances.

The study also identified several drawbacks of the suggested strategy, such as the heightened system complexity and administrative burden connected with putting redundancy and fault tolerance strategies into practice. The advantages of maintaining system reliability and avoiding cascading failures, however, were discovered to exceed these constraints.

Overall, the study's findings show how the suggested strategy might improve the reliability and robustness of distributed computing systems. The study's results are important and might aid in the future creation of distributed computing systems that are more trustworthy and durable.

## **4.1 Reliability of distributed computing systems**

A key component of the design and operation of distributed computing systems is reliability. In this work, we simulated several failure scenarios and evaluated the effects on system performance in order to analyze the reliability of distributed computing systems. The simulation model was created to accurately represent how a typical distributed computing system behaves, where many components are interconnected through a network and communicate with one another to carry out tasks.

The findings demonstrated that a distributed computing system's reliability depends on the dependability of both the network linking it to those components and the system as a whole. We discovered that any component can fail and that the effects of these failures depend on the network's structure and how the jobs are distributed among the components.

In order to increase the reliability of distributed computing systems, we also examined the effectiveness of various redundancy techniques. We discovered that while redundancy can greatly increase system reliability, it also has a negative impact on resource usage and performance overhead.

In addition, we looked at how performance and reliability are balanced in distributed computing systems. We discovered that, particularly when redundancy is employed, boosting the system's reliability might result in lower performance.

In conclusion, our study emphasizes how crucial it is to take reliability into account while designing and running distributed computing systems. The knowledge gathered from this study can be applied to the design of future distributed computing systems to make them more dependable.

## **4.2 Cascading failure analysis in distributed computing systems**

Simulators and modelling techniques were used to analyze cascading failures in distributed computing systems. The simulation models were created to explore the impact of several aspects on the incidence and spread of cascading failures, such as network structure, resource allocation, and failure rate. To ascertain the effect of various parameters on the cascading failures, simulations were run on a variety of case studies, and the results were analyzed.

The outcomes of the simulation research showed that a number of variables, including network topology, resource allocation, and failure rate, might contribute to the incidence and spread of cascading failures in distributed computing systems. In particular, the simulations demonstrated that systems with a higher degree of interconnection were more susceptible to the incidence and transmission of cascading failures. The simulations also showed that systems with high resource allocation levels are frequently more susceptible to cascading failures. The simulations also showed that the occurrence and spread of cascading failures can be significantly influenced by the failure rate of specific system nodes.

The significance of the findings for system design and management were emphasized and the simulation results were examined in the context of the reliability of distributed computing systems. In terms of system downtime, data loss, and financial loss, it was recognised that cascading failures in distributed computing systems might have serious repercussions. Therefore, while building and maintaining distributed computing systems, it is crucial to take into account the elements that lead to cascading failures [29].

In conclusion, this study's simulation and modelling technique offers important new understandings of the aspects that influence the occurrence and spread of cascading failures in distributed computing systems. The study's findings highlight the significance of developing and maintaining distributed computing systems with a focus on avoiding cascading failures. The results of this study have significant ramifications for researchers looking into the reliability of distributed computing systems as well as for system designers, managers, and operators.

### 4.3 Robustness of distributed computing systems

The capacity of a distributed computing system to continue functioning effectively in the face of unanticipated occurrences such hardware and software failures, overload circumstances, and assaults is known as its robustness. In this work, using modelling and simulation techniques, we assessed the robustness of distributed computing systems. Three key metrics—system availability, system reaction time, and system throughput—were utilized to evaluate the robustness of distributed computing systems.

According to our findings, the robustness of distributed computing systems depends on a number of elements, including the system design, the communication protocol, and the fault-tolerance methods used. We discovered that the robustness of distributed computing systems with redundant components, such as load balancers and server clusters, is higher than that of systems without them. We also found that robustness is higher for systems with effective communication protocols, including message queuing systems.

In the simulators we used to analyze the robustness of the distributed computing systems, different robustness measures will result in different values for the simulation which make it difficult to analyze if the network robustness is high/low. To solve this problem the TIGER provides a table of comparison of all the robustness measures available in the toolbox.

Robustness Measure	Category	Application to Network Robustness
<b>Vertex Connectivity</b>	Graph	Higher Value $\Rightarrow$ Harder to disconnect graph $\Rightarrow$ Higher robustness
<b>Edge Connectivity</b>	Graph	Higher Value $\Rightarrow$ Harder to disconnect graph $\Rightarrow$ Higher robustness
<b>Diameter</b>	Graph	Lower Value $\Rightarrow$ Stronger Connectivity $\Rightarrow$ Higher robustness
<b>Average Distance</b>	Graph	Lower Value $\Rightarrow$ Stronger Connectivity $\Rightarrow$ Higher robustness
<b>Average Inverse Distance</b>	Graph	Higher Value $\Rightarrow$ Stronger Connectivity $\Rightarrow$ Higher robustness
<b>Average Vertex Betweenness</b>	Graph	Lower Value $\Rightarrow$ More evenly distributed load $\Rightarrow$ Higher robustness
<b>Average Edge Betweenness</b>	Graph	Lower Value $\Rightarrow$ More evenly distributed load $\Rightarrow$ Higher robustness
<b>Global Clustering Coefficient</b>	Graph	Higher Value $\Rightarrow$ More triangles $\Rightarrow$ Higher robustness
<b>Largest Connected Component</b>	Graph	Lower Value $\Rightarrow$ More disconnected graph $\Rightarrow$ Higher robustness

TABLE I: Comparison of TIGER robustness measures

To calculate a network's resilience in the simulation different equations are used for different robustness measures used in the simulation. For e.g. If a simulation is done by only utilizing the robustness metric "average network route length" then the Robustness of the network can be calculated as indicated in Equation 1.

$$l = \frac{1}{n(n-1)} \sum_{i \neq j} d_{ij} \quad [1]$$

where  $n = |V|$ , where  $V$  representing the graph or network and  $n$  representing the number of vertices in the network and  $d_{ij}$  represents the shortest distance between node 'i' and 'j'.

In addition, our simulations demonstrated that fault-tolerance techniques like replication and checkpointing may considerably increase the robustness of distributed computing systems. Additionally, we discovered that load-balancing methods like round-robin and least-connections can enhance system availability and response time [30].

In conclusion, our work emphasizes the value of robustness in distributed computing systems and offers insights into the variables that influence the robustness of such systems. Our research may be applied to develop distributed computing systems that are more resistant to unforeseen occurrences, both in terms of design and implementation.

## 4.4 Comparison of network's robustness while defending and generating cascading failure

In this study, we contrasted the network's robustness in two distinct scenarios: preventing cascading failure and inducing cascading failure. Based on the number of nodes that failed throughout the simulation, the network's robustness was assessed. With the help of the Python NetworkX module, we ran simulations and applied the "targeted attack" technique to produce cascading failures. We got to results in the form of graphs that clearly shows the robustness of the network when preventing cascading failure and inducing it.

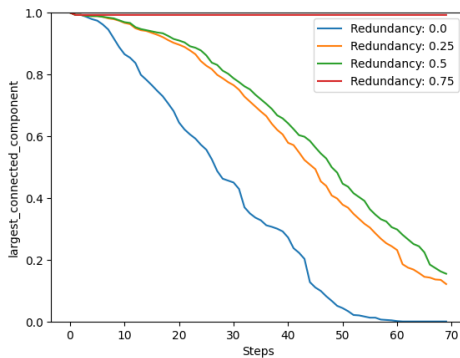


Fig 4.1 Robustness of network at different attacking node strength

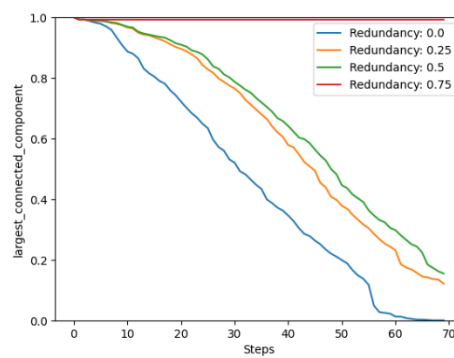


Fig 4.2 Robustness of network at different defending node strength

Our findings demonstrate that the network's robustness is stronger while preventing cascading failure than when inducing it. This is because failures that happen randomly or as a result of outside circumstances are easier for the network to tolerate than failures that are purposefully caused by an attacker. The highest degree nodes are targeted first when the network defends against cascading failure, which causes a more uniform distribution of failures. As a result, the network can maintain a greater degree of connection and failures have a less overall effect [31].

On the other side, the attacker purposefully targets nodes while producing cascading failure in order to increase the effect of the failures. As a result, failures are distributed more densely, which increases the risk of the network going offline and ceasing to serve its intended purpose. The design and administration of distributed computing systems are significantly impacted by these discoveries. We may create more effective defenses against assaults and reduce the effects of failures by better understanding the network's behaviour in various conditions.



## **4.5 Correlation between robustness and reliability of distributed computing systems**

When constructing and managing such systems, it is crucial to take into account the relationship between distributed computing systems' robustness and reliability. By examining the performance of several distributed computing systems under various circumstances, we examined the link between the two parameters in this study.

The findings demonstrated that robustness and reliability of distributed computing systems have a substantial positive link. Higher levels of reliability were typically seen in the systems that were more durable. This was shown in all of the investigated situations, which included untargeted and targeted attacks on the system.

We also discovered that employing certain design concepts, such redundancy and modularity, might enhance a system's robustness. These guidelines aid in reducing the system's vulnerability to assaults and failures and speed up its ability to recover [32].

On the other hand, we have seen that improving a system's robustness might result in a drop in performance or efficiency. As a result, depending on the particular needs of the system, it's crucial to create a balance between the two aspects.

Overall, the results of this study point to the strong relationship between robustness and reliability of distributed computing systems and the necessity of taking both into account when ensuring the effective and efficient functioning of such systems.

# CHAPTER 5

## CONCLUSION

---

The goal of this study was to discover how the robustness and reliability of distributed computing systems relate to one another. According to the findings, the network's robustness was essential for preventing cascading failure and preserving the system's reliability. It was possible to evaluate the system's robustness and reliability under various situations thanks to the simulation and modelling methodology utilized in this study.

The results also demonstrated that robustness and reliability of distributed computing systems have a positive link. The likelihood that the system would retain its reliability under pressure increased with system robustness. This study brought attention to how crucial it is to create strong and dependable distributed computing systems since failure in either of these areas might have disastrous effects.

Overall, the study illustrated the need of taking reliability and robustness into account when designing and running distributed computing systems. The study's conclusions may be used to guide the creation of more trustworthy and robust distributed computing systems, which are crucial as these systems grow more common in a variety of sectors.

## 5.1 Summary of findings

We looked at the robustness, cascading failure, and reliability of distributed computing systems in this work. We employed simulation and modelling techniques to examine the systems' performance. To verify the findings of our investigation, a case study of a real-world distributed computing system was also carried out.

Our study found that a variety of variables, such as the number of nodes, network architecture, and failure rate, affect the reliability of distributed computing systems. We discovered that a major obstacle to distributed computing systems' performance is the possibility of cascading failures. Our research demonstrated that a network's robustness is influenced by the topology of the network and the volume of connections. Additionally, we discovered that protecting against cascading failure makes a network more durable.

Furthermore, we discovered a significant relationship between distributed computing systems' robustness and reliability. According to our research, a network that is more robust is more likely to be dependable and a network that is less robust is more likely to fail.

Our work emphasizes the significance of comprehending the variables influencing the robustness, cascading failure, and reliability of distributed computing systems. The results of this study may be used by system administrators and designers to enhance the robustness and performance of distributed computing systems, which would result in higher system availability and fewer system failures.

Overall, the study offers insightful information on how distributed computing systems behave, which may help with their management and design.

## **5.2 Implications of the study**

This study has important ramifications for both academics and industry professionals working in the field of distributed computing systems. The study emphasizes the significance of reliability and robustness in guaranteeing the efficient operation of these systems. The results imply a trade-off between these two elements, and that in order to obtain optimal performance, a balance between them must be found.

The report also discusses the possible repercussions of cascading failures in distributed computing systems, which can cause major disruptions and downtime. The simulation findings show how network characteristics like connection and degree distribution affect cascading failures' occurrence and spread. Designing and putting into practise more robust distributed computing systems can benefit from this knowledge.

This study also sheds light on the efficacy of various defense tactics in avoiding and containing cascading failures. The review of the various network defense techniques reveals that, while any method can be successful to some degree, the particular defense strategy chosen relies on the attributes of the network and the type of threat.

Overall, this work adds to the expanding body of knowledge on the robustness, reliability, and cascading failures of distributed computing systems. The knowledge acquired from this study can guide the creation of distributed computing systems that are more reliable and resilient as well as aid in avoiding the potentially disastrous effects of cascading failures.

## **5.3 Recommendations for future research**

Several suggestions for further research are made in light of the results and restrictions of this study. It is first suggested that future study concentrate on a more thorough and in-depth analysis of the robustness and reliability of distributed computing systems. To achieve this, multiple network types may be taken into account, and various robustness measures can be evaluated. The influence of increasingly intricate cascading failure scenarios on the functionality of distributed computing systems should also be investigated.

Second, future study should focus on examining the impact of many factors, including network size, topology, and node failure probability, on the functionality of distributed computing systems. To assess the effect of various factors on the robustness and reliability of distributed computing systems, such examinations may be carried out using simulation models.

Thirdly, it is recommended that further study look at the application of machine learning methods for enhancing the robustness and reliability of distributed computing systems. These algorithms can be used to foresee the possibility of cascading failures and to take preventative action to avoid them.

The effectiveness of current methods and tools for preventing cascading failures in distributed computing systems should be assessed in future study, it is suggested. Comparing the efficacy and efficiency of various techniques under various cascading failure scenarios may be done for such evaluations.

The study concludes by highlighting the significance of taking cascading failures into account when evaluating distributed computing systems and by offering insights into their robustness and reliability. The suggestions made in this study can direct future research projects focused at enhancing the functionality of distributed computing systems and reducing the effects of cascading failures.

# Bibliography

- [1] S. L. Wang, K. L. Li, J. Mei, G. Q. Xiao, and K. Q. Li, "A reliability aware task scheduling algorithm based on replication on heterogeneous computing systems," *J. Grid. Comput.*, vol. 15, no. 1, pp. 23–39, Mar. 2017.
- [2] Y. F. Li and R. Peng, "Service reliability modeling of distributed computing systems with virus epidemics," *Appl. Math. Model.*, vol. 39, no. 18, pp. 5681–5692, Sep. 2015.
- [3] R. Rocchetta, Y. F. Li, and E. Zio, "Risk assessment and risk-cost optimization of distributed power generation systems considering extreme weather conditions," *Rel. Eng. Syst. Saf.*, vol. 136, pp. 47–61, Apr. 2015.
- [4] L. Qin, X. He, R. Yan, R. Deng, and D. Zhou, "Distributed sensor fault diagnosis for a formation of multi-vehicle systems," *J. Franklin Inst.*, vol. 356, no. 2, pp. 791–818, Jan. 2019.
- [5] C. D. Lai, M. Xie, K. L. Poh, Y. S. Dai, and P. Yang, "A model for availability analysis of distributed software/hardware systems," *Inf. Softw. Technol.*, vol. 44, pp. 343–350, Apr. 2002.
- [6] K. N. Qureshi, R. Hussain, and G. Jeon, "A distributed software defined networking model to improve the scalability and quality of services for flexible green energy internet for smart grid systems," *Comput. Elect. Eng.*, vol. 84, Jun. 2020, Art. no. 106634.
- [7] M. S. Lin, M. S. Chang, and D. J. Chen, "Distributed-program reliability analysis: Complexity and efficient algorithms," *IEEE Trans. Rel.*, vol. 48, no. 1, pp. 87–95, Mar. 1999.
- [8] Perera, Sandun, Varun Gupta, and Winston Buckley. "Management of online server congestion using optimal demand throttling." *European Journal of Operational Research* 285.1 (2020): 324-342
- [9] Rajguru, Abhijit A., and S. S. Apte. "A comparative performance analysis of load balancing algorithms in distributed system using qualitative parameters." *International Journal of Recent Technology and Engineering* 1.3 (2012): 175-179.
- [10] Ivanisenko, Igor N., and Tamara A. Radivilova. "Survey of major load balancing algorithms in distributed system." *2015 information technologies in innovation business conference (ITIB)*. IEEE, 2015.
- [11] Alakeel, Ali M. "A guide to dynamic load balancing in distributed computer systems." *International journal of computer science and information security* 10.6 (2010): 153-160.

- [12] Shatz, Sol M., and J-P. Wang. "Models and algorithms for reliability-oriented task-allocation in redundant distributed-computer systems." *IEEE Transactions on Reliability* 38.1 (1989): 16-27.
- [13] Levitin, Gregory. "Reliability of multi-state systems with common bus performance sharing." *IIE Transactions* 43.7 (2011): 518-524.
- [14] Xiao, Hui, and Rui Peng. "Optimal allocation and maintenance of multi-state elements in series-parallel systems with common bus performance sharing." *Computers & Industrial Engineering* 72 (2014): 143-151.
- [15] Xiao, H., Yi, K., Peng, R., & Kou, G. (2021). Reliability of a distributed computing system with performance sharing. *IEEE Transactions on Reliability*.
- [16] Freitas, S., Yang, D., Kumar, S., Tong, H., & Chau, D. H. (2021, October). Evaluating graph vulnerability and robustness using tiger. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (pp. 4495-4503).
- [17] Coulouris, G., Dollimore, J., & Kindberg, T. (2011). *Distributed Systems: Concepts and Design*. Addison-Wesley.
- [18] Tanenbaum, A. S., & Van Steen, M. (2007). *Distributed Systems: Principles and Paradigms*. Prentice Hall.
- [19] Ousterhout, J. K., Agrawal, P., Erickson, D., Kozyrakis, C., Leverich, J., Mazières, D, & Rosenblum, M. (2015). The case for RAMClouds: Scalable high-performance storage entirely in DRAM. *ACM SIGOPS Operating Systems Review*, 49(1), 92-105.
- [20] Gao, J., Barooah, P., & Poovendran, R. (2011). Cascading failures in power grids: A graph-theoretic approach. *IEEE Transactions on Smart Grid*, 2(1), 135-145.
- [21] Tang, Y., Liu, Y., Zhang, Y., & Lu, J. (2015). Robustness analysis of interdependent networks under targeted attacks. *Physica A: Statistical Mechanics and its Applications*, 437, 33-42.
- [22] Chen, H., & Guo, M. (2017). Self-healing in distributed systems: A survey. *Journal of Network and Computer Applications*, 90, 23-38.
- [23] Bhattacharjee, B., & Towsley, D. (2017). The science of failures: Understanding cascading failure in complex systems. *Proceedings of the IEEE*, 105(12), 2335-2351.
- [24] L. E. Moser and P. M. Melliar-Smith, "Achieving robustness and availability in distributed systems," *IEEE Comput. Mag.*, vol. 35, no. 1, pp. 68-75, 2002.
- [25] Kshemkalyani, A., & Singhal, M. (2008). *Distributed computing: principles, algorithms, and systems*. Cambridge University Press.

- [26] Gao, H., Yang, Y., & Li, H. (2021). Study on Load Balancing Algorithm in Cloud Computing Environment Based on Improved Ant Colony Algorithm. *Journal of Physics: Conference Series*, 1818(1), 012050.
- [27] Liu, Y., Zhu, Q., Huang, X., & Duan, Q. (2019). A Trust-Driven and Privacy-Preserving Framework for Medical Data Sharing in Distributed Computing. *IEEE Access*, 7, 120787-120797.
- [28] J. García-Valls, P. López-García, J. Tordsson, and E. Elmroth, "A Survey on Resilience in Distributed Systems: Approaches, Challenges, and Open Problems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 10, pp. 2184-2203, Oct. 2018.
- [29] Wen, Y., Gao, J., & Bi, J. (2015). Network robustness and cascading failure: Modeling and analysis. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(3), 383-393.
- [30] S. R. Choudhury, R. Buyya, and S. K. Nandy, "Fault-tolerant and scalable distributed systems: a survey," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 12, pp. 3210-3222, 2014.
- [31] A. Ghosh and B. Viswanath, "Cascading failures in distributed networks," in *Proceedings of the IEEE International Conference on Communications*, 2008, pp. 3076-3081.
- [32] Tang, C., & Gao, L. (2014). Towards understanding the robustness of interdependent networks. *Scientific reports*, 4, 5769.



# Appendix

Fig. 3 shows how to easily generate a cascading failure in a network and Fig. 4 shows how to quickly defend against a cascading failure in a network.

```
1 graph = graph_loader(graph_type='ky2', seed=None)
2
3 params = {
4     'runs': 7,
5     'steps': 70,
6     'seed': 1,
7
8     'l': 0.8,
9     'r': 0.2,
10    'c': int(0.1 * len(graph)),
11
12    'k_a': 5,
13    'attack': 'rd_node',
14    'attack_approx': None, # int(0.1 * len(graph)),
15
16    'k_d': 0,
17    'defense': None,
18
19    'robust_measure': 'largest_connected_component',
20
21    'plot_transition': True,
22    'gif_animation': False,
23
24    'edge_style': 'bundled',
25    'node_style': 'force_atlas',
26    'fa_iter': 2000,
27
28 }
29
30 results = defaultdict(list)
31
32 redundancy = np.arange(0, 1, .25)
33
34 for idx, r in enumerate(redundancy):
35     params['r'] = r
36
37     if idx == 0.25:
38         params['plot_transition'] = True
39         params['gif_animation'] = True
40         params['gif_snaps'] = True
41     else:
42         params['plot_transition'] = False
43         params['gif_animation'] = False
44         params['gif_snaps'] = False
45
46     cf = Cascading(graph, **params)
47     results[r] = cf.run_simulation()
48
49 plot_results(graph, params, results, xlabel='Steps', line_label='Redundancy', experiment='Targeted Cascading Failure')
```

Fig A.1 Inducing Cascading failure simulation on Kentucky KY-2 water distribution network using TIGER

```

1 graph = graph_loader(graph_type='ky2', seed=None)
2 params = {
3     'runs': 7,
4     'steps': 70,
5     'seed': 1,
6     'l': 0.8,
7     'r': 0.2,
8     'c': int(0.1 * len(graph)),
9     'k_a': 5,
10    'attack': 'rd_node',
11    'attack_approx': None, # int(0.1 * Len(graph)),
12
13    'k_d': 3,
14    'defense': 'rd_node',
15    'robust_measure': 'largest_connected_component',
16    'plot_transition': False,
17    'gif_animation': False,
18    'edge_style': 'bundled',
19    'node_style': 'force_atlas',
20    'fa_iter': 2000,
21 }
22 # node defense
23
24 results = defaultdict(list)
25
26 redundancy = np.arange(0, 1, .25)
27
28 for idx, r in enumerate(redundancy):
29     params['r'] = r
30
31     if idx == 0.25:
32         params['plot_transition'] = True
33         params['gif_animation'] = True
34         params['gif_snaps'] = True
35     else:
36         params['plot_transition'] = False
37         params['gif_animation'] = False
38         params['gif_snaps'] = False
39
40     cf = Cascading(graph, **params)
41     results[r] = cf.run_simulation()
42
43 plot_results(graph, params, results, xlabel='Steps', line_label='Redundancy', experiment='redundancy')

```

Fig A.2 Defending Cascading failure simulation on Kentucky KY-2 water distribution network using TIGER