Model Checking the Reliability of Smart Grid

Thesis submitted in accordance with the requirements of the University of Liverpool for the degree of Master in Philosophy by

Kai Zheng

February 2019

Abstract

In recent years, air pollution in China become more serious than before. Thermal power generation is one of the main pollution sources, so the Chinese government wants to develop distributed energy systems (DESs) to solve the problem.

However, the monitoring and control of the DESs become a challenge. In order to solve the problem, the concept of smart microgrid is introduced. Smart microgrid can monitor and optimize the running of DESs in an intelligent way. Smart microgrid systems usually include the following components: DESs, converters, inverters, sensors, gateways, and servers. The control algorithms are imported into the inverters and converters to realize the optimal control of DESs. The internet of things (IoT) network in a smart microgrid is used for monitoring the operation of the DESs.

However, the reliability of smart microgrid is still a challenge. In recent years, some of the researchers also focus on the reliability of smart microgrid[53]. But the research about the reliability of the smart microgrid is still not enough. Most of the researchers focus on the power quality reliability of the microgrid. However, few research concentrates on optimizing the structure design of smart microgrid.

In this project, we will optimize the architecture design of smart microgrid. Continuous-time Markov chain (CTMC) models will be used to evaluate the reliability of smart micro-grid. The architectures of the IoT system and DC microgrid will be evaluated respectively. Then the analysis results will show our optimized architecture is better. The optimized design of smart microgrid in this project will help the designer to improve the architecture design of smart microgrid in real cases. In this project, Monte Carlo method, reliability block diagram (RBD) method and case study system are used as benchmarks.

Acknowledgements

I would like to express my gratitude to my supervisor Dr.Xin Huang. He is a respectable people for the supervision. He helps me to build the structure of the thesis and give me the advice for the experiment design. His knowledge provides a very good basis for my thesis. Next, I will express my secondary supervisor Professor Alan Marshall, he was patient for teaching me how to make my project into a real research. My third supervisor Dr Paul Craig also give me some useful advice for the thesis. Whats more, some master students and bachelor students also help me a lot for the experiments.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

In the last decades, air pollution in China became serious. What's more, coal resources are dwindling. Thus, distributed energy systems has attracted increasing attention. For distributed energy system, energy is generated or stored by a variety of small, grid-connected devices referred to as distributed energy resources (DER) or distributed energy resource systems[13]. Distributed energy systems use some renewable energy sources such as small hydro, biomass, biogas, solar power, wind power, geothermal power. The distributed energy sources help to solve the air pollution problem. In order to effectively control the DER, the concept of the microgrid is introduced to coordinate the operation of the DERs. For distributed energy systems (DESs), the most widely used one is photovoltaic (PV) energy system. They are widely used for providing energy to domestic appliances, charging piles[15][32].

However, the monitoring of microgrid become a challenge [6]. According to [27], in 2009 and 2010, domestic PV energy systems caused 2 fire disasters in Germany. In recent months, in Fujian province and Shanxi province, 2 fire disasters happened in rooftop PV systems. The disasters reveal the monitoring problems of DES. So in recent years, the concept of the smart microgrid is introduced for monitoring the operation of the microgrid in an intelligent way [21]. The smart microgrid systems usually have distributed energy sources, converters, inverters, and the sensor network for monitoring the whole microgrid system. However, the reliability of the DC microgrid is still a challenge.

1.2 Some Current Research About Smart Microgrid

1.2.1 Microgrid

Microgrid gets energy from DES and provides the energy to the grid. Take the PV based microgrid as an example, The PV panels fed an input voltage to the DC/DC converters, the DC/DC converter gives a suitable voltage for the inverter. The function of the inverter is transferring the DC electricity to AC electricity. In this way, the microgrid can provide energy to the main grid. Microgrid is divided into 2 types, DC microgrid and AC microgrid. The typical DC microgrid is shown in Fig. 1.1. The typical AC microgrid is in Fig. 1.2. In recent years, the application of DC microgrid is increasing.

The typical microgrid includes the following components:

- PV panel: gets the luminous energy and transfer it into electric energy.
- DC/DC converter: adjusts the input voltage into a suitable one.
- DC/AC inverter: transfers the DC electricity into AC electricity.
- AC load: consumes AC energy.
- DC load: consumes DC energy
- AC/DC converter: transfers the AC electricity into DC electricity.

1.2.2 Control Methods of Microgrid

The microgrid usually includes the following components: distributed energy sources (DES), converters and inverters. The most widely used DES are PV energy system and wind energy system[2]. We will first concentrate on the PV energy system. PV energy system is an energy system designed to supply usable solar power by means of photovoltaic. For the domestic PV energy system, they are connected to the main grid[38]. For the control of PV energy system, the researchers focus on maximum power point tracking (MPPT) to improve the efficiency of PV energy system[31][34][35]. Some other researchers concentrate on the voltage stabilization for the PV energy system [23][24][49]. The control of the PV energy system is realized by importing control algorithms to the converters and inverters. For example, the maximum power point tracking (MPPT) method is used to improve the efficiency of the PV energy system to get more energy. The control algorithms are realized by adjusting the duty cycle of the converter or inverter. Take the converter as an example, the output voltage of the converter is shown as follows:

$$V_{\text{out}} = V_{\text{in}} \times D \tag{1.1}$$

V_{in} is the input voltage, V_{out} is the output voltage, D is the duty cycle. In Fig. 1.3, we can see that the output power of the PV panel is related to the output voltage of the converter. For the first half, the output power is proportional to the output voltage of the converter. For the latter half, the output power is inversely proportional to the output power. So the MPPT control algorithms are used to track the maximum power point.

For the wind energy system, some of the researchers also focus on the MPPT control of it, these control algorithms contribute to improving the efficiency of the wind energy system [10]. The low voltage ride through (LVRT) method is used to solve the transient fault problem of the power system.

For microgrid, the coordinate control of the microgrid is also very important to absorb the energy of DES. Typical control method include hierarchical control and droop control [36][5].

1.2.3 Monitoring System for Microgrid

In recent years, the number of domestic DC microgrid is increasing. The fire disaster for these microgrids is also increasing. So the monitoring of such microgrid become a challenge to avoid such disasters [36][5].

Many researchers also give some research about monitoring system for DES [14][19][40]. They usually focus on the monitoring of the domestic microgrid. A typical IoT system for DES is shown in Fig.1.4.

In this monitoring system, power monitoring modules and solar radiation modules are used to monitor the output power and real-time irradiation. The data is transmitted to the controller (PC) via wireless modules. The controller can provide data to users. Based on the output power and irradiation data, the controller will give commands to the control switch modules to adjust DC loads. Some researchers also give some other monitoring systems for the PV energy system, some other functions are added to the democratic monitoring system.

1.2.4 The Evaluation Methods for Sensor Network in Power System

The sensor network is widely used in many areas. For example, it can be used to get environmental data and health data. In a smart microgrid, the sensor network is used to meter the irradiation, temperature, wind speed and so on. The parameters will be used to predict the output power of the microgrid.

However, the sensor network also has failure problems. The failures mainly include nodes failure. For the server node and sensor node in the IoT system, the node failure problem may be caused by software and hardware problem.

The most widely used way to evaluate the reliability of wireless sensor network (WSN) is conducting experiments via real WSNs system. The conditions of the real WSNs are collected for analysis, the reliability also can be analyzed via the data. However, the time for collecting the data is too long. What's more, the cost of the devices is also a challenge. For these reasons, the methods are not very suitable for optimizing the design of the architecture of the IoT system.

Thus, the researchers provide some methods to evaluate the reliability of WSN by using simulation method or analytical modeling method[43]. The reliability analysis helps the designer to optimize the architecture design of WSN. In this way, the cost of maintenance will be reduced.

The typical widely used method for optimizing the architecture design of WSN are simulation method and analytical modeling method[11][50]. For the simulation method, Monte Carlo (MC) method is a popular example[20]. MC can be used to simulate the experiment about real WSNs reliability analysis. However, MC takes too much time. For analytical modeling method, the example is Reliability Block Diagram (RBD) method. RBD is based on math models[33]. RBD also can solve the disadvantages of the simulation method.

1.2.5 The Evaluation Methods for Microgrid

PV energy systems are widely used in many areas, however, the reliability of PV energy system is also very important. The failure of PV energy systems will influence the normal operating of the power system. For these reasons, many researchers focus on the reliability of PV energy system. The most widely used methods are the Monte Carlos method and RBD method.

1.2.6 Some Other Research About Reliability of IoT and Microgrid

In previous research, some researchers also focus on the reliability of the remote reprogramming system and IoT system for the PV energy system. In [48] and [47], problematic model checking is used for model checking the reliability of remote reprogramming system. However, the remote reprogramming is still an experiment platform and not widely used, so the meaning of research about its reliability remains to be discussed. In [29], model checking is used to analyze the reliability of the IoT system in the domestic PV energy system. However, the structure of it is still an experiment platform, so it is more simple than the real system. According to [41], the author gives the reliability analysis of a mobile edge network. However, the case study system was not built for proving the correctness of the models. What's more, the failure rate of the components is too high (the failure of the sensor occurs several days), this does not match reality in real cases.

In this project, I focus on the smart microgrid scenario, the IoT system, and DC microgrid are widely used in some remote islands and villages. The failure rate of each component is supported by some reference books, so the reliability analysis contribute to the real architecture design of smart microgrid. What's more, the case study system is built to analyzing the failure process of the smart microgrid. These are the main difference between my project and previous work.

1.3 Motivation

The overall aim of the project is to investigate the architecture design optimizing of the smart microgrid, which can help the designer to optimize the design of smart microgrid in real cases. In this project, the reliability of smart microgrid is also very important. The failure of it will influence the normal operating of the power system. The reliability analysis helps the designer to optimize the architecture design of the smart microgrid.

In recent years, some of the researchers also focus on the reliability of microgrid. However, the research about the reliability of the smart microgrid is still not enough. Most of the researchers focus on the power quality reliability of the microgrid. However, few of them concentrate on the structure reliability of smart microgrid system [51][22] [3]. If the reliability of the smart microgrid system can be simulated to find out the main influencing factors and optimal architectures. The cost of field testing will be saved. The strategy for improving the reliability of the smart microgrid will help the designer in real cases.

1.4 Proposed Solution

The overall aim of the project is to investigate the architecture design optimizing of the smart microgrid, which can help the designer to optimize the design of smart microgrid in real cases. In this project, the reliability of smart microgrid is also very important. The failure of it will influence the normal operating of the power system. The reliability analysis helps the designer to optimize the architecture design of the smart microgrid.

For this situation, this Mphil research will solve the current problems. In this project, we will give optimal architectures of the DC microgrid and its IoT system. Then CTMC models will be built in PRISM to simulate the reliability of the DC microgrid and its monitoring system. For the reliability of the IoT system, the comparison between the reliability of different architectures of the IoT system will be analyzed first. Then the influence of different influencing factors will be analyzed include (1)impact of the failure rate of smart sockets and smart meters;(2)impact of smart sockets and smart meters numbers; (3)impact of the failure rate of gateways; (4)impact of gateway numbers. For the reliability of DC microgrid, the comparison between the reliability of different architectures of DC microgrid will be analyzed first. Then the influence of different influencing factors will be analyzed include (1)impact of the failure rate of PV panels;(2)impact of PV panel numbers;(3)impact of inverter numbers; (4)impact of the failure rate of inverters. The Monte Carlo (MC) method and reliability block diagram (RBD) will be used as benchmarks to ensure the correctness of the CMTC models. For the experiment part, we build similar architectures to show our architecture is the better one.

1.5 Aim and Specific Objectives

This project aims to use a probabilistic model checking technique with the tool of PRISM to addresses the performance (reliability) analysis of smart microgrid. The specific objectives are shown as follows:

- Build CTMC models to model the IoT system and DC microgrid of it.
- Use Continuous Stochastic Logic to analyze the properties of the smart microgrid.
- Use Monte Carlo method and RBD method as benchmarks for the correctness of our CTMC model.

- Build case study system to simulate the failure process of the smart microgrid.

1.6 Contribution

Overall, this project makes some contributions:

- CTMC models are used to model the DC microgrid and its IoT system.
- Continuous Stochastic Logic is used to analyze the properties of the smart microgrid.
- Monte Carlo method and the RBD method are used as benchmarks for the correctness of our CTMC model.
- Case study system is built to simulate the failure process of the smart microgrid.

1.7 Publications

The related publication is shown below:

- Wu S, Zheng K, Huang X. Model Checking PV Energy System with Remote Reprogramming Function[C], published by International Conference on Information Technology in Medicine and Education. IEEE Computer Society, 2016:606-610.
- A Denial of Service Attack Methods For an IoT System, published by 8 th International Conference on IT in Medicine and Education, Fuzhou, 2016.
- Model Checking of IoT System in Microgrid, published by 8 th International Conference on IT in Medicine and Education, Fuzhou, 2016.
- Liang L, Zheng K, Sheng Q, et al. A Denial of Service Attack Method for IoT System in Photovoltaic Energy System[C], published by International Conference on Network and System Security. Springer, Cham, 2017:613-622.
- Model Checking PBFT Consensus Mechanism in Healthcare Blockchain Network, published by 9 th International Conference on Information Technology in Medicine and Education. IEEE Computer Society, 2018.

1.8 Organization of The Thesis

The thesis is divided into the following parts. Chapter II will give the theory of reliability analysis. Chapter III shows the optimal architecture of smart microgrid. Chapter IV gives the process of model construction for the 2 parts. The benchmarks are shown in chapter V. In chapter VI, we build case study similar architectures of the smart grid to show our analysis is correct. In the final part, the conclusion and future work are given.

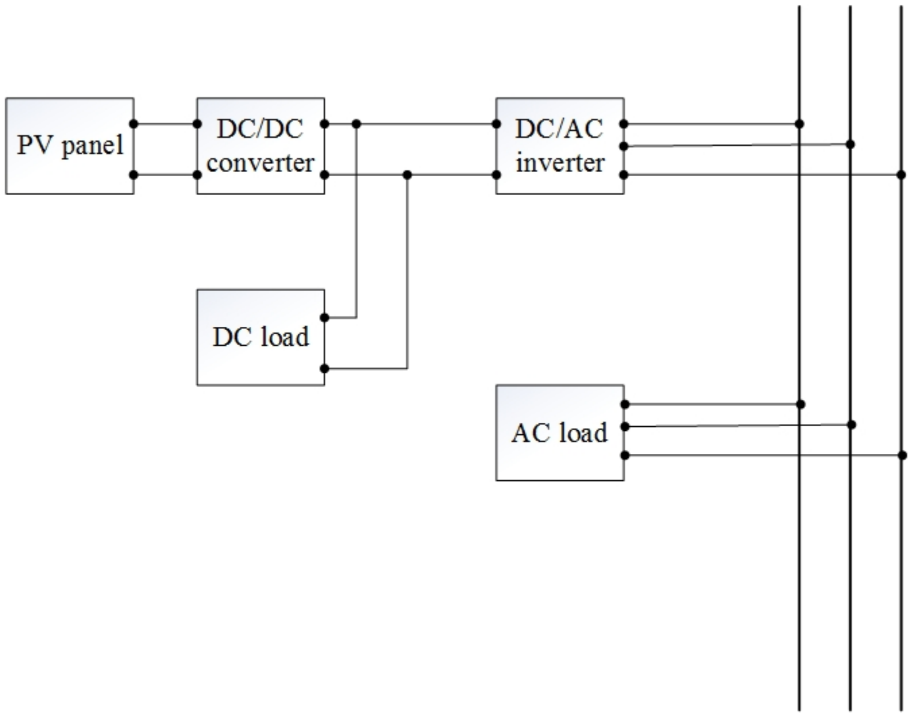


Figure 1.1: DC microgrid.

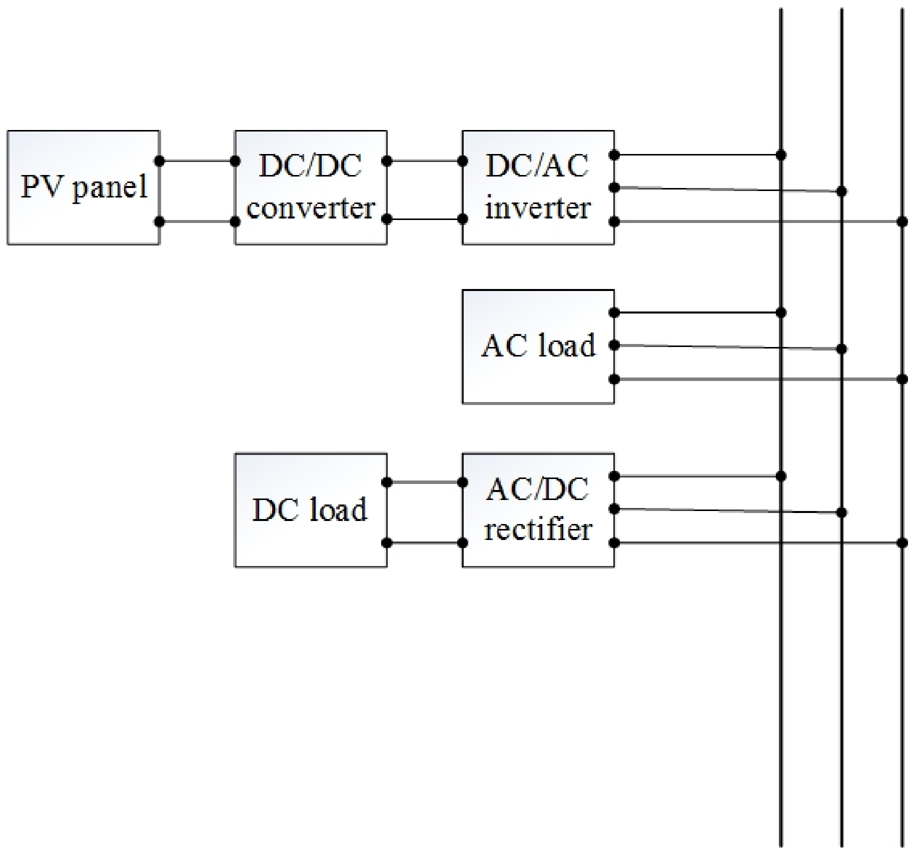


Figure 1.2: AC microgrid.

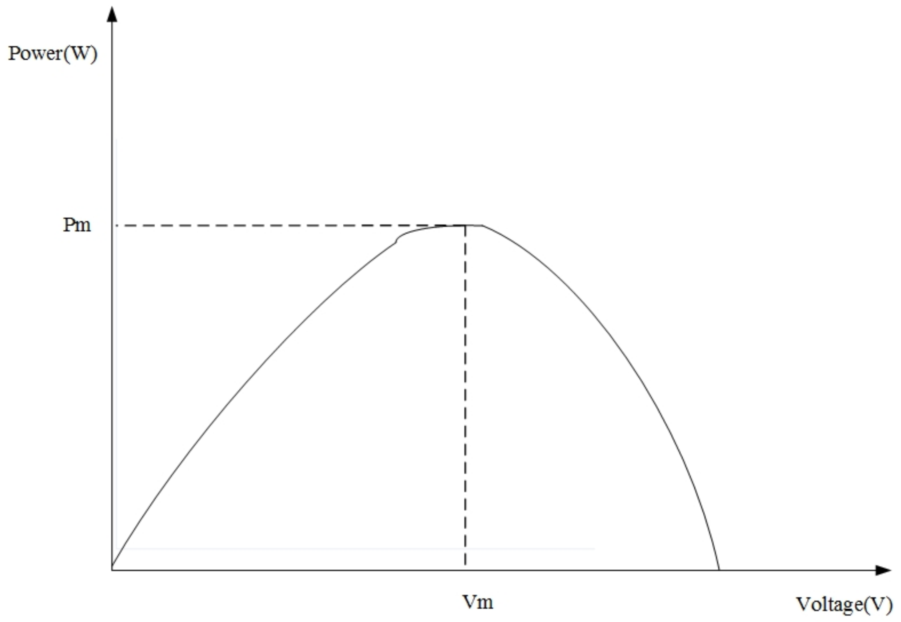


Figure 1.3: MPPT algorithm.

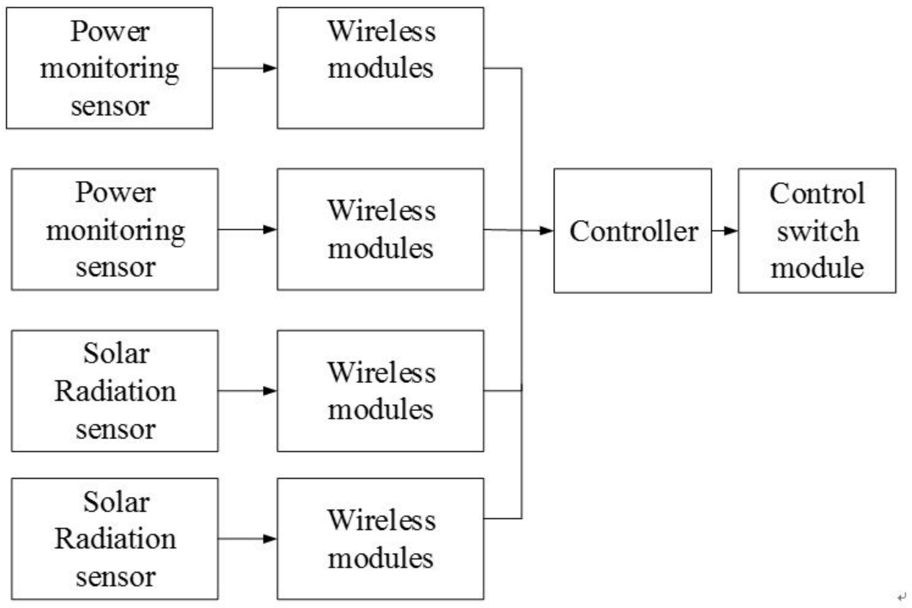


Figure 1.4: IoT system for DC microgrid.

Chapter 2

Basic Knowledge About Reliability Analysis

In this chapter, some basic knowledge about reliability analysis and probabilistic model checking will be shown. The basic knowledge will help us to design CTMC models for the reliability analysis of smart microgrid. The typical methods for reliability analysis are also shown in this chapter.

2.1 The Introduction of RBD Method

RBD method is a typical method for evaluating the reliability of variable systems. A typical example of RBD is shown in Fig.2.1.

Suppose the lifetime of all the components (X1 to X7) is 2000 hours, we want to calculate the failure rate of the whole system in 100 hours. The process is shown below. The failure rate of 1 components is shown below.

$$F_0 = 1 \text{ - } e^{\text{ - } \lambda t} = 1 \text{ - } 0.951 = 0.049 \tag{2.1}$$

Then calculate the failure rate of X1 to X3:

$$F_{123} = {F_0}^3 = 0.000118 \tag{2.2}$$

Then calculate the failure rate of X5 to X6:

$$F_{56} = F_0 + F_0 - {F_0}^2 = 0.0956 \tag{2.3}$$

Then calculate the failure rate of X4 to X6:

$$F_{456} = F_4 * F_{56} = 0.004684 \tag{2.4}$$

Then calculate the failure rate of whole failure rate:

$$F_{\text{whole}} = F_{123} + F_{456} - F_{123} * F_{456} = 0.0998 \tag{2.5}$$

2.2 The Introduction of Monte Carlo Method

Monte Carlo Method is also a typical method for evaluating the reliability of variable systems. A typical example of Monte Carlo method is shown in Fig.2.2.

In Fig.2.2, λ is the failure rate of one component. We do many experiments (for example, 1000 times). If the point is behind the curve, we think the system is down. Otherwise, the system is still ok."n_{nth_day}" is the experiment times for nth day. "$n_{fail\_nth\_day}$" is the failure times in nth day.The failure rate of the components in nth days is shown as follows.

$$F_{nth\_day} = \frac{n_{fail\_nth\_day}}{n_{nth\_day}} \tag{2.6}$$

The calculation of the failure rate of the whole system is similar to RBD method.

2.3 Introduction of Probabilistic Model Checking

Probabilistic model checking is a formal verification technique. It can be used for analyzing and modeling systems which has exhibit probabilistic behavior[25]. The model checking can be used to verify some properties of the systems. For example, the model checking can be used to build the models the following systems or procedures[1].

- Operating system
- Communication protocols
- Bio-manufacturing procedures

- Grid World Robot
- Network communication security analysis

Now model checking is widely accepted in industrial practice or software tools examination. The process of model checking is shown as follows:

1. Use the modeling language to construct the model of the target system or procedures. Markov model is widely used in probabilistic model checking[9]. Some typical Markov models include: Discrete-time Markov Chain (DTMC)[28], continuous-time Markov chain (CTMC)[37], MDP (Markov decision processes)[7]and so on.

2. Construct some properties of the system by using some logic. The logic includes Probabilistic temporal logics (PCTL) and Continuous Stochastic Logic (CSL). PCTL is mainly used for DTMC model and MDP model, CSL is mainly used for CTMC model.

3. Use the models and properties to conduct model checking, the experiment results can be used for analyzing the properties of the system.

2.4 Markov Chain

A Markov chain is a stochastic model that can be used to describe a sequence of possible events. The probability of each event depends only on the state attained in the previous event. In other words, the Markov chain has a property called memorylessness. For the target system, if its current state is known, then the future states of this target system will be independent of its past states. Markov chains are widely used in many areas like embedded system, smart control system in motor vehicles, the customs lines in a railway station, the exchange rate of currencies, population growth and some storage system. Some Markov chain models include: discrete-time Markov chains (DTMCs), continuous-time Markov chains (CTMCs), Markov decision processes (MDPs) and so on.

For the research about the analysis of system performance, in past decades, reliability block diagram (RBD) and Monte Carlo method are popular. However, the models are not suitable to describe the reliability of the system. What's more, the relationship between time and reliability is not clearly described in previous models. So in recent years, the research about Markov chain is increasing. The application range of the Markov model is also increasing.

2.5 Continuous-time Markov Chain

In this thesis, Continuous-time Markov chain (CTMC) model is used to study smart microgrid reliability recently. A CTMC is a four-tuple (S, s_0, R, L):

- S is the set of states, but it is finite. $S = \{s_0, s_1, s_2, s_3..., s_{\mathrm{n}}\}$
- s_0 stands for the initial state.
- R stands for the transition rate matrix $|S| \times |S|$
- L is the function for labeling.

For example, we can suppose an example of IoT system for the microgrid. It has 3 end devices and 2 gateways, the end devices can upload data to one gateway, another gateway is a backup gateway. We call the number of working end devices nd (nd=3) and working gateway ng (ng=2). The example IoT system is shown in Fig.2.3. The failure of end devices occurs in every 5 days, the failure of end devices occurs in every 30 days.

This example IoT system fails if (1) the number of working sensors is less than one, (2) all the 2 gateways fail. The failure process of the example IoT system can be modeled by the CTMC model.

The process is shown in Fig.2.4.

For this example, the CTMC model is shown as follows:

- $ES = \{es_0, es_1, es_2, es_3..., es_{\mathrm{n}}\}$, stands for the set of states.
- es_0 is the initial state.
- The transition matrix is shown in Fig.2.5.
- L = Fail,OK is the labeling function.

2.6 Continuous Stochastic Logic

Continuous stochastic logic (CSL) is the extension version of non-probabilistic continuous temporal logic[4]. CSL can be used to verify the properties of CTMC models. CSL can be defined by 2 kinds of syntaxes: state formula (F) and path formula (Y). The state formula is described as follows:

- True.
- $L(s_i)$: is the atomic proposition.
- $\Phi_1 \wedge \Phi_2$: means the Φ_1 and Φ_2 are true.
- $\neg\Phi$ means Φ is not true.

The path formula is used to describe the set of all possible paths starting from the state. The path formula is described as follows:

- $X\Phi$: means that the next is state formula.
- $\Phi_1 \cup^{[a,\infty]} \Phi_2$ remain in Φ_1 until time a, then Φ_2 is true.
- P bound [pathprop]: it stands for probabilistic operator.
- S bound [prop]: steady state operator.

The logic CSL can be used to verify the reliability of IoT. For CSL, "P" is the probability, "true" means the system is working well. "U" means "until". "Down" means the system fails. For example, in this example IoT system. The failure conditions are shown as follows:

$$Down = (ng < 1)|(nd < 1) \tag{2.7}$$

This means that all the gateways and end devices fail.

Some useful properties are shown in TABLE.2.1.

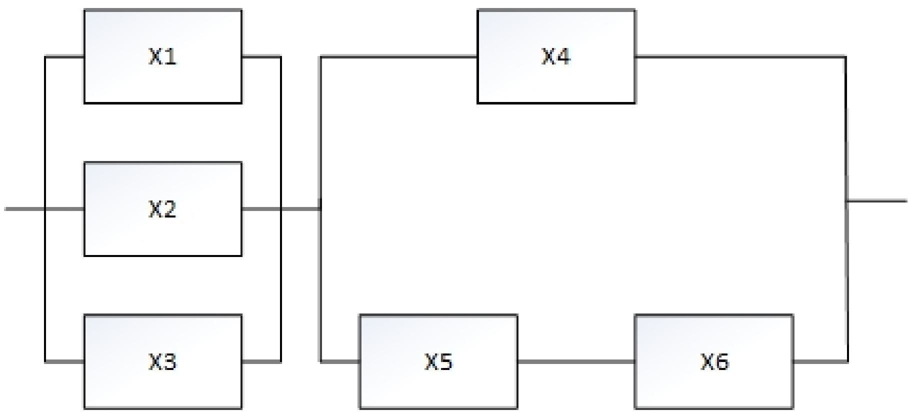


Figure 2.1: An example of RBD method.

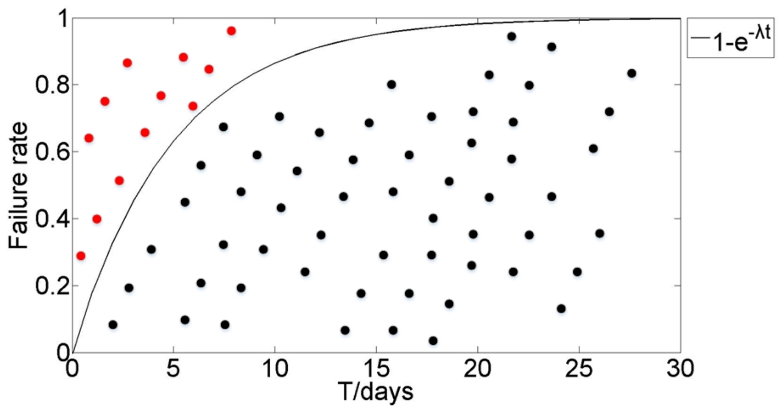


Figure 2.2: An example of Monte Carlo method.

Table 2.1: Properties for evaluating the reliability of IoT system

The Statements of CSL	**Meaning**
$P = ?[true U <= 24 * 3600 Down]$	The failure rate of the system within 1 day
$P = ?[true U <= 30 * 24 * 3600 Down]$	The failure rate of the system within 30 days

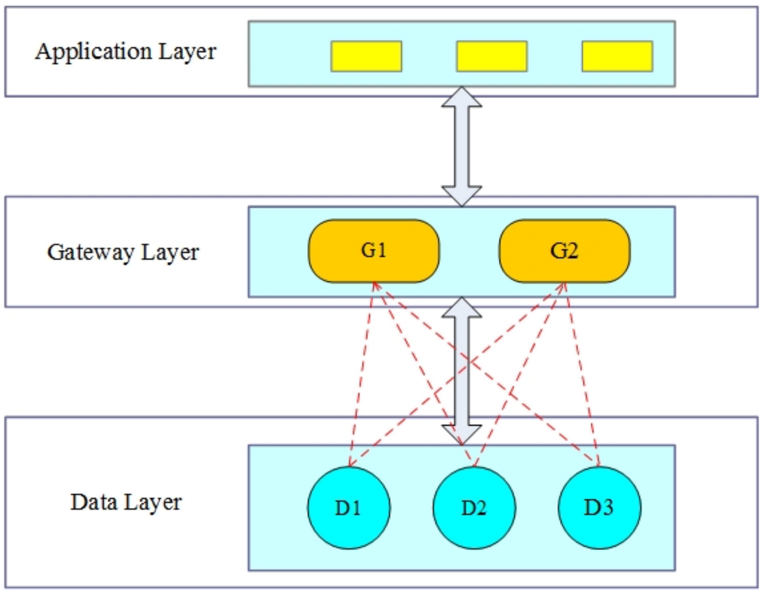


Figure 2.3: The architecture of example IoT system.

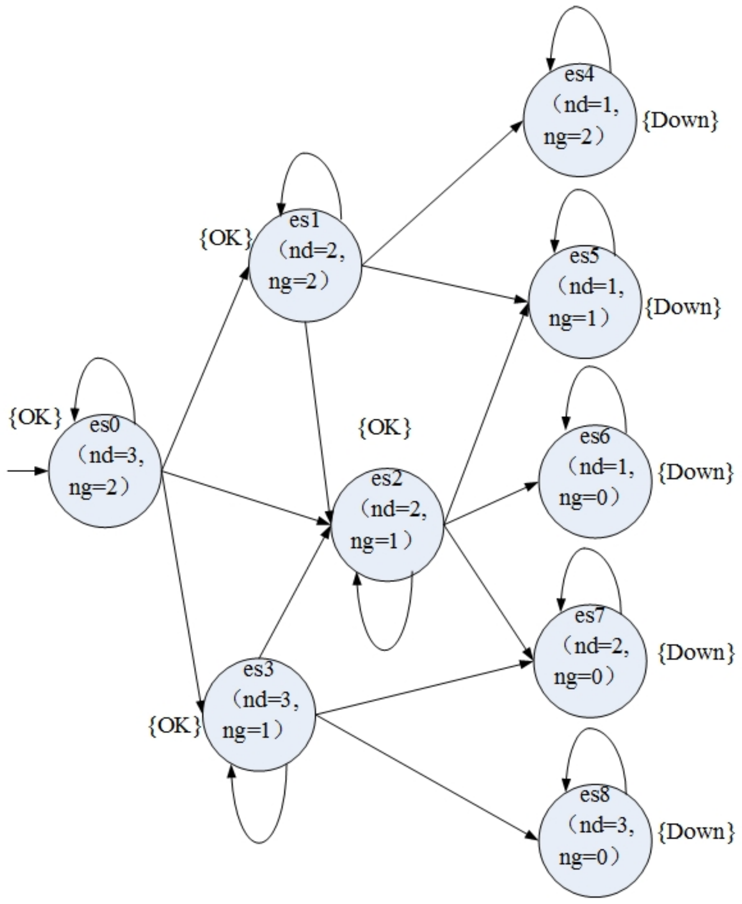


Figure 2.4: The state transition diagram for the failure process of the example IoT system.

$$\begin{bmatrix}
\frac{86399}{86400}\times\frac{86399}{86400} & \frac{1}{86400}\times\frac{86399}{86400} & \frac{1}{86400}\times\frac{1}{86400} & \frac{86399}{86400}\times\frac{1}{86400} & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{86399}{86400}\times\frac{86399}{86400} & \frac{86399}{86400}\times\frac{1}{86400} & 0 & \frac{1}{86400}\times\frac{86399}{86400} & \frac{1}{86400}\times\frac{1}{86400} & 0 & 0 & 0 \\
0 & 0 & \frac{86399}{86400}\times\frac{86399}{86400} & 0 & 0 & \frac{1}{86400}\times\frac{86399}{86400} & \frac{1}{86400}\times\frac{1}{86400} & \frac{86399}{86400}\times\frac{1}{86400} & 0 \\
0 & 0 & \frac{1}{86400}\times\frac{86399}{86400} & \frac{86399}{86400}\times\frac{86399}{86400} & 0 & 0 & 0 & \frac{1}{86400}\times\frac{1}{86400} & \frac{86399}{86400}\times\frac{1}{86400} \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}$$

Figure 2.5: The transition matrix for the example IoT system.

Chapter 3

The Optimal Architecture Design of Smart Microgrid

In this thesis, an optimal architecture of the IoT system in the smart microgrid is proposed. The IoT system includes 3 layers: data layer, gateway layer, and applications layer. The optimal architecture of the IoT system is shown in Fig.3.1.

3.1 The Components of The Optimal Architecture

The IoT system includes three layers. The functions of the different layer are shown as follows:

- Data layer: this layer involves many smart meters and smart sockets. Smart meters are used to monitor the generated energy of PV panels. Smart sockets are used to monitor the energy consumption of smart buildings. For each group, the backups are used in case of failures.

- Gateway layer: the gateway layer has some gateways. The sensors can upload data to the applications via gateways. In this layer, backup gateways are used in case of gateway failures.

- Application layer: the application layer contains applications for users to handle devices in the data layer.

In this experiment, we call the optimal architecture Architecture I. We will use other typical architectures to show our architecture is better.

3.2 Some Exiting Architectures

For current architectures, some of them use the centralized gateway. They usually do not have backup gateways. Distributed architectures usually mean one gateway connects certain sensor groups. Some architectures also use master-slave gateway architecture for the IoT system [17] and [16]. The architectures are shown in Fig.3.2.

3.2.1 Architecture II, III, and IV

Architecture II is a centralized gateway architecture, which is shown in Fig.3.2a. In Architecture II, the number of gateways is 1. All the sensor groups upload the data to the applications via centralized gateways.

Architecture III is shown in Fig.3.2b, Compared with the first two architectures, this one is modified in the following aspects:

- Each group has its own gateway. Gateway 1 manages the smart meters, gateway 2 manages smart sockets.

Architecture IV is shown in Fig.3.2c. The differences between Architecture III and Architecture IV are shown below:

- There are 2 kinds of gateways: a master gateway and a local gateway. Gateway 1 is the master gateway. Gateway 2 and 3 are local gateways.
- The sensors are divided into 2 groups, local gateway 2 and 3 connect smart meters and smart sockets respectively.

3.2.2 The Comparison Between the 4 Architectures

The comparison of the 4 architectures is shown in TABLE.3.1.

The difference description is shown below:

- Sensor number: All the four architectures have 2 smart meters and 20 smart sockets.

- Root gateway: The first three architectures do not have root gateway. Architecture IV has one root gateway.
- Gateways number: Architecture I and III has 2 gateways, Architecture II has 1 gateway. Architecture IV has 3 gateways.

3.3 The Introduction of Modeling IoT System

Our work focus on how to provide methods to improve the reliability of the IoT. The IoT system is divided into 3 parts: data layer, gateway layer, and application layer. The modeling of the 3 architectures is shown below.

3.3.1 Architecture I

The model design of Architecture I is shown as below:

Smart Meters Module: The smart meters module shows the status of the smart meters. The states of the smart meters module are shown below:

- nm: is the number of workable smart meters.
- nm=2: is the initial state, the number of smart meters is 2.

Smart Sockets Module: The smart sockets module shows the status of the smart sockets. The states for the smart sockets layer is shown below:

- ns: is the number of workable smart sockets.
- $ns = 20$: is the initial state, the number of working smart sockets is 20.

Gateway Module: The gateway module shows the status of the gateways. The states for the gateway layer is shown below:

- ng: is the number of workable gateways.
- $ng = 2$: is the initial state, the number of the working gateways are 2.

Failure Conditions of DC Microgrid: The failure conditions for the whole system is shown below:

- $nm < 2$.

- $ns < 15$.
- $ng < 1$.

3.3.2 Architecture II, III, and IV

The model design of Architecture II, III and IV are similar to Architecture I. However, the 3 architectures are different, so the formal model design of the 3 architectures is also different. The difference between the formal model design is shown below:

- S: The states of Architecture IV is more than the other 3 architectures. Architecture II has the minimum states, architecture IV has the maximum states.
- s: The initial state means all the components in the 3 architectures are OK.

3.3.3 The Comparison of Optimal Architecture (Architecture I) and Other Typical Architectures

For the Optimal Architecture (Architecture I), it has backup gateways in case of failure of the gateways. If one gateway fails, the backup gateway will replace it and manage the sensors. In this way, the reliability of the IoT system will be improved. The reliability of the 4 architectures will be evaluated respectively. The experiment results will show the strategy to improve the reliability of the IoT system obviously.

3.4 3 Architectures of DC Microgrid

Now the application of the microgrid is increasing. However, how to improve the architecture design of the microgrid is still a challenge. If the design of the microgrid can be improved. The reliability of the microgrid will also be further enhanced. In this project, we will give out how to optimize the architecture design of the microgrid.

In this project, we give the optimal architecture of the microgrid. Then use some existing architectures for the microgrid. Then the reliability of the microgrid will be analyzed. According to the experiment results, we will show our designed architecture is better than the others.

However, how to improve the architecture design is still a challenge. Firstly, how to build models for the architectures of the microgrid. Secondly, how to figure out the impact of different components. Thirdly, how to find out which structure is more reliable. Finally, how to improve the structures for microgrid by using the analysis results. In this project, we solve these challenges. These will be illustrated later.

In this project, we will first analyze the influencing factors of the reliability of the system. Secondly, we will design 3 models for different architectures of the microgrid. Architecture I is an optimal architecture, Architecture II is a centralized structure, Architecture III is a mixed structure. Thirdly, according to the 3 models, we will design 3 CTMC models in PRISM. Fourthly, the reliability of the 3 architectures will be evaluated. Finally, the impact of each component will be analyzed.

3.4.1 The Optimal Architecture of DC Microgrid

The Architecture I of the microgrid is shown in Fig 3.3. In normal operation of the microgrid, The PV panels fed an input voltage to the DC/DC converter. Then DC/AC inverter transfer the DC electricity into AC electricity. In this way, the PV energy system provides energy to the main grid. We call the optimal architecture Architecture I. The function of different components is shown below:

- PV panels: provide energy to the load.
- Inverter: transfer the DC electricity to AC electricity.
- Main grid: accept the energy provided by PV panels.

3.4.2 Architecture II

For Architecture II, each PV panel is connected to the centralized inverter. The distributed microgrid architecture is shown in Fig.3.4. In Architecture II, each converter is controlled by a single inverter.

3.4.3 Architecture III

The Architecture III is shown in Fig.3.5. For this architecture, the DC/DC converter is used for adjust the DC voltage.

3.4.4 Summary of the 3 Architectures

For the 3 architectures, the number of controller and converter are different. Architecture I has 4 inverters. Architecture II has 1 centralized inverter. Architecture III has 4 converters and 1 inverters. The Comparison of the 3 architectures is shown in TABLE.3.2.

3.5 The Introduction of Modeling DC Microgrid

Our work focus on how to provide methods to improve the reliability of the DC microgrid. The DC microgrid is divided into 3 parts: PV panels, inverters, and the main grid. This is supported by [52] and [8]. The modeling of the 3 architectures is shown below.

3.5.1 Architecture I

The model of Architecture I is shown as below:

PV Layer Module: The PV layer module shows the status of the PV panels. The states of the PV module is shown below:

- np: is the number of workable PV panels
- np=4: is the initial state, the number of PV panels is 4.

Inverter Layer Module: The inverter layer module shows the status of the inverters. The states for the inverter layer is shown below:

- ni: is the number of workable inverters
- $ni = 4$: is the initial state, the number of working inverters is 4.

Failure Conditions of DC Microgrid: The failure conditions for the whole system is shown below:

- $np < 2$.
- $ni < 2$.

3.5.2 Architecture II and III

The model design of Architecture II, III and IV are similar to Architecture I. However, the 3 architectures are different, so the formal model design of the 3 architectures is also different. The difference between the formal model design is shown below:

- S: The states of Architecture III is more than the other 2 architectures. Architecture II has the minimum states.
- s: The initial state means all the components in the 3 architectures are OK. However, the devices of different architectures are different.

3.5.3 The Comparison of Optimal Architecture (Architecture I) and Other Typical Architectures

For the Optimal Architecture (Architecture I), it is a distributed architecture. It has more distributed inverters than other architectures to improve the reliability of it. The experiment results will show the distribute architecture improve the reliability of the IoT system obviously.

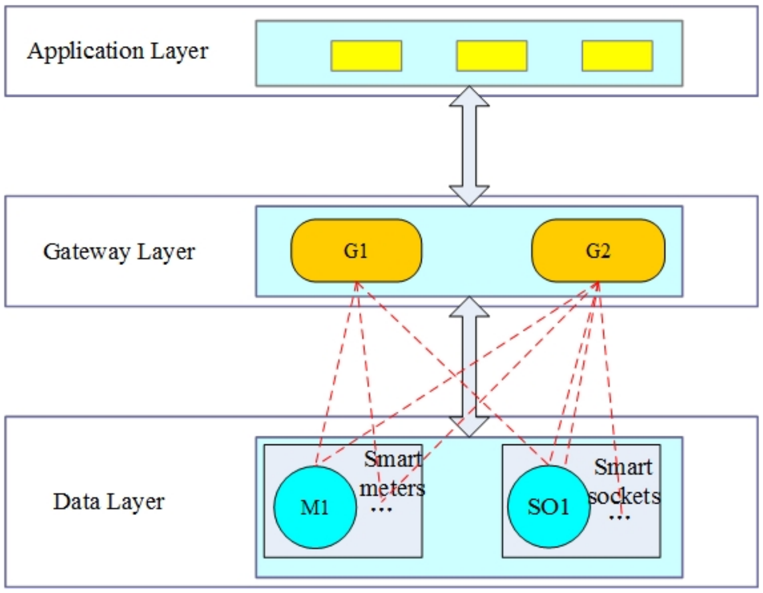


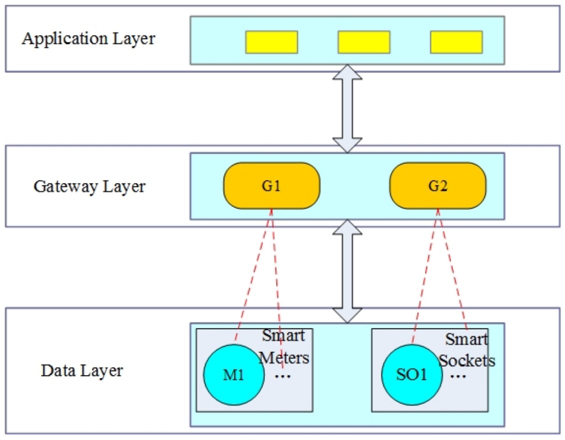
Figure 3.1: The optimal architecture for the IoT system.

Table 3.1: CTMC models comparison (IoT system)

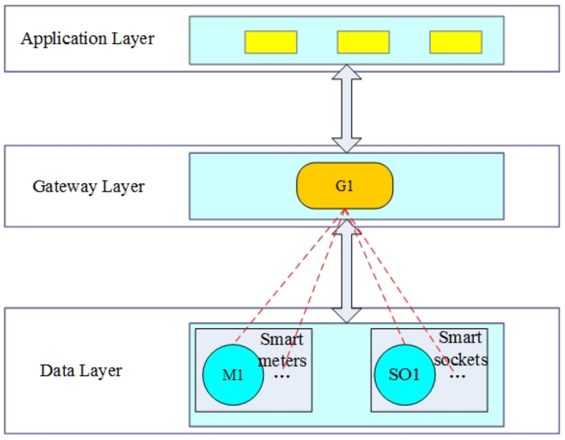
	Architecture I, Fig. 3.1	**Architecture II, Fig. 3.2a**	**Architecture III, Fig. 3.2b**	**Architecture IV, Fig. 3.2c**
Number of smart sockets nodes	20	20	20	20
Number of smart meter nodes	2	2	2	2
Root gateway	None	None	None	1 root gateway
Number of gateways	2 gateways	1 gateway	2 gateways	3 gateways
Gateway sectors	1 gateway sector	1 gateway sectors	2 gateway sectors	3 gateway sectors

Table 3.2: CTMC models comparison (DC microgrid)

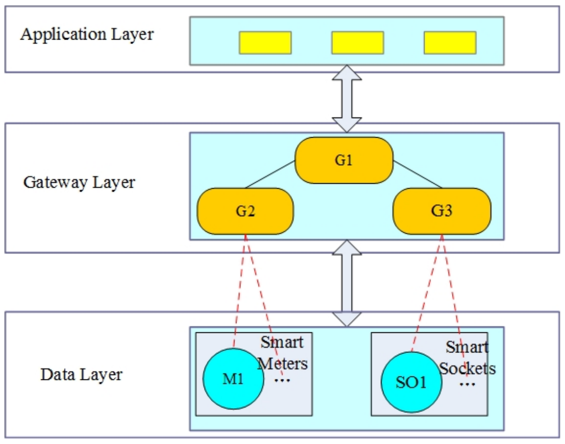
	Architecture I, Fig. 3.3	**Architecture II, Fig. 3.4**	**Architecture III, Fig. 3.5**
PV	4	4	4
DC/DC converter	0	0	4
Inverter	4	1	1



(a) Architecture II



(b) Architecture III



(c) Architecture IV

Figure 3.2: The typical 3 architectures.

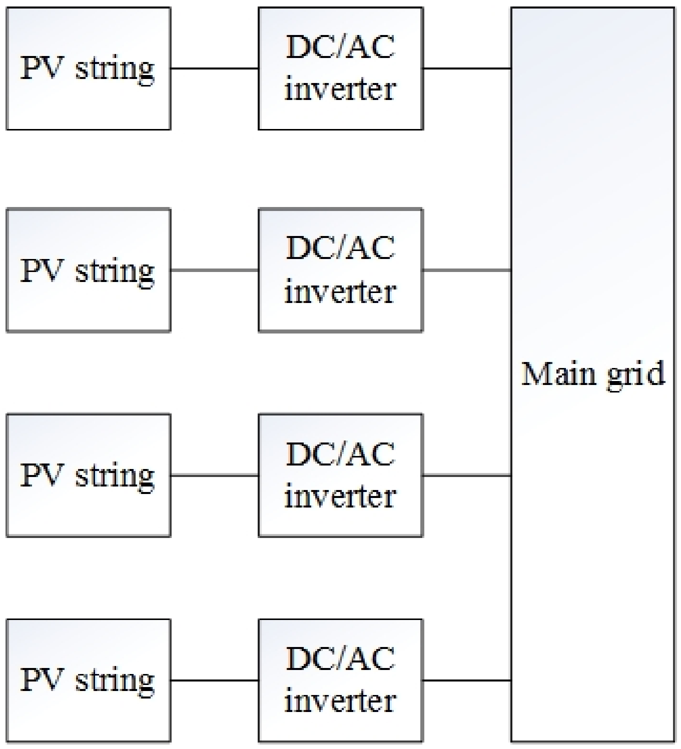


Figure 3.3: The Architecture I of DC microgrid.

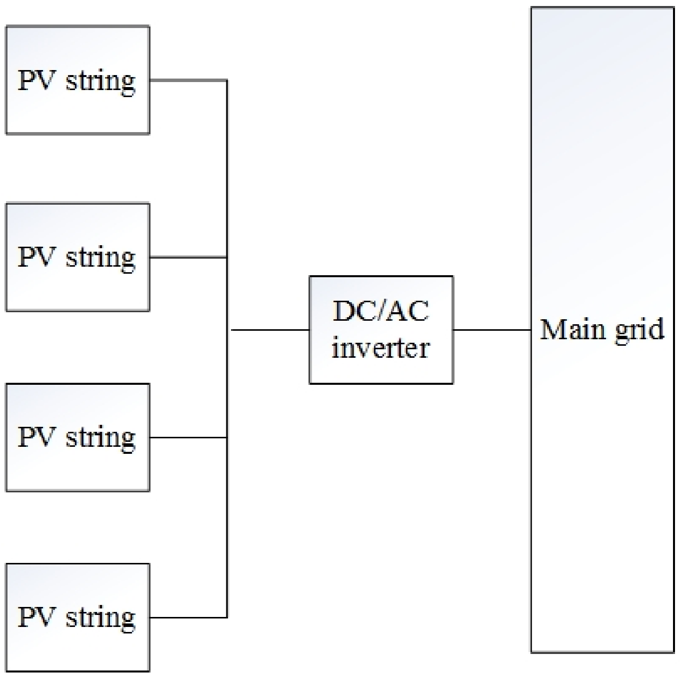


Figure 3.4: Architecture II of DC microgrid.

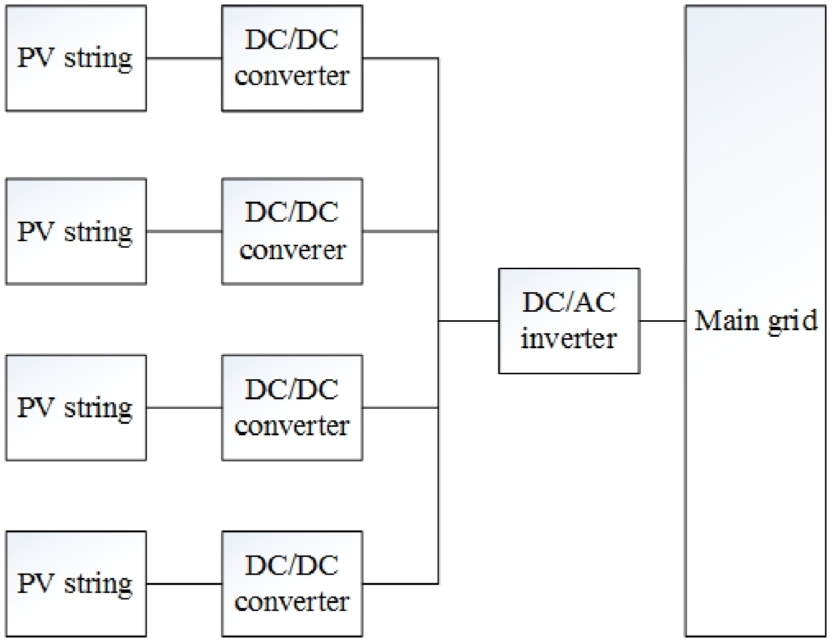


Figure 3.5: Architecture III of DC microgrid.

Chapter 4

Model Design of The Smart Microgrid and Its Analysis

Our work focuses on the network structures of the smart microgrid, we will use CTMC models to model checking the reliability of the IoT system and DC microgrid respectively. The chapter will include the following parts:

- Problematic model checking technique and the tool PRISM.
- CMTC model design of the IoT system in smart microgrid
- CMTC model implementation of the IoT system.
- Analysis of the reliability of the IoT system.
- CMTC model design of DC microgrid
- CMTC model implementation of the DC microgrid.
- Analysis of the reliability of the DC microgrid.

4.1 Problematic Model Checking Technique and The Tool PRISM

In this project, the model checking technique is used for evaluating the reliability of the smart microgrid. The process for the model checking technique is shown in Fig.4.1

For our model checking technique in my project, according to the structure of smart microgrid and failure conditions, the probabilistic model and probabilistic temporal logic specification are built in PRISM. Then the model checker PRISM will use the probabilistic model and probabilistic temporal logic specification to give out the quantitative results.

PRISM is an open-source and free probabilistic model checker. It also acts as a tool for formal modeling and analysis of systems which has random or probabilistic behaviour[18]. PRISM has been used to analyze systems from many different areas, including communication and multimedia protocols[26], network security analysis[39], bio-manufacturing systems[12], and communication cells and so on.

Some probabilistic models can be modeled and analyzed include:

- Probabilistic Automata (PAs)
- Probabilistic timed automata (PTAs)
- Discrete-time Markov chains (DTMCs)
- Continuous-time Markov chains (CTMCs)
- Markov decision processes (MDPs)

In these models, the cost and rewards can also be calculated and analyzed. In PRISM, the Models can be described by using the PRISM language. It is a state-based and simple language. PRISM enables the automated analysis about a wide range of quantitative properties of these PRISM models, e.g. "what is the probability of a shutdown causing the embedded system to shut down in 30 days?

For the property specification language, it helps to construct some temporal logics such as PCTL, CSL, and LTL. The quantitative specifications for cost and rewards can also be constructed.

Experiment environment and parameters are described in Table 4.1 for ease of reading.

4.2 CMTC Model Design of The IoT System

4.2.1 Architecture I

For Architecture I, four-tuple (S, s_0, R, L) is shown as follows:

- S:$S = \{s_0, \mathrm{s}_1, \mathrm{s}_2, s_3..., s_n\}$, it stands for the states of the IoT system.

- s0: initial states, all the components are applicable.
- R: the transition matrix is similar to the example.
- L: Fail means the IoT system fails. OK means the IoT system is OK.

Smart Meter Module: this module represents the number of workable smart meters. "λ_m " is the failure rate of smart meters. For initial state, if 1 smart meter fails, the process will be shown as:

$$nm \overset{\lambda_m}{\to} nm - 1 \tag{4.1}$$

Smart Socket Module: this module represents the number of workable smart sockets. "λ_s " is the failure rate of smart sockets. For initial state, if 1 smart meter fails, the process will be shown as:

$$ns \overset{\lambda_s}{\to} ns - 1 \tag{4.2}$$

Gateway Module: The controller layer has 2 gateways . Both the gateways have failure rate "λ_g". For initial state, if 1 gateway fails with failure rate "λ_g", the process is shown as:

$$ng \overset{\lambda_g}{\to} ng - 1 \tag{4.3}$$

The failure conditions of Architecture I are shown as follows:

- $nm < 1$
- $ns < 15$
- $ng < 1$

4.2.2 Architecture II, III and IV

The construction of Architecture II, III and IV are similar to Architecture I. For the model design, the differences for the 4 architectures are shown in TABLE.4.2. For Architecture II, the states are less than Architecture I because architecture II just has 1 centralized gateway. Architecture III has more states because gateways, smart sockets and smart meters are divided into several sectors.

4.3 CMTC Model Implementation of The IoT System.

4.3.1 Architecture I

The model of Architecture I is shown as follows. The model includes 3 parts: Smart Meter Module, Smart Socket Module, and Gateway Module.

Smart Meter Module: "nm" is the available smart meter numbers, the initial number of "nm" is 12. If 1 smart meter fails with failure rate "*lambda_m*", "nm" will minus 1. The code for the data layer is shown below:

```
nm  : [ 0 . . 2 ] init 2 ;
[] nm >0 -> nm  * lambda_m  : ( nm' = nm  - 1 ) ;
```

Smart Socket Module: "ns" is the available smart meter numbers, the initial number of "ns" is 20. If 1 smart meter fails with failure rate "*lambda_s*", "ns" will minus 1. The code for the data layer is shown below:

```
ns  : [ 0 . . 20 ] init 20 ;
[] ns  >0 -> ns  * lambda_s   : ( ns' = ns  - 1 ) ;
```

Gateway Module: In Architecture I, the number of controllers is 2. "ng" is the available controller number. If 1 gateway fails with a failure rate "*lambda_g*", "ng" will minus 1. The code for the controller layer module is shown below:

```
ng : [ 0 . . 2 ] init 2 ;
[] ng>0 -> ng * lambda_g : ( ng' = ng - 1 ) ;
```

The Condition for Failure of The Whole System: If "nd" is smaller than lower bound, or "ng" is smaller than lower bound of gateways, "ns" is smaller than lower bound of smart sockets, the system fails. The code for failure conditions is shown as follows:

```
formula down =(nd<1) | (ng<1)|(ns <15) ;
```

4.3.2 Architecture II, III, and IV

Architecture II, III and IV are also constructed in PRISM. The modeling of other architectures are similar to Architecture I, the differences are shown in TABLE.4.3.

- Firstly, all architectures has 2 sensor sectors(nd1, nd2).

- Secondly, Architecture I and III have 2 gateway nodes(g=2), architecture II has 1 gateway nodes(g=1).
- Thirdly, Architecture IV has 1 root gateway(g3=1).

4.4 Results Analysis of IoT System in Smart Microgrid

After the CTMC models for the IoT system are built in PRISM, we design different experiments to show our architecture is better than other architectures. What's more, for the optimized architecture, the impact of different components are analyzed. $fiot$ is the failure rate of the IoT system. The experiment is divided into 7 parts:

- Experiment 1: The reliability of different architectures are analyzed to show architecture I is better.
- Experiment 2: The impact of the number of smart meters in the IoT system is analyzed.
- Experiment 3: The impact of the failure rate of smart meters in the IoT system is analyzed.
- Experiment 4: The impact of the number of smart sockets in IoT system is analyzed.
- Experiment 5: The impact of the failure rate of smart sockets in IoT system is analyzed.
- Experiment 6: The impact of the number of gateways in the IoT system is analyzed.
- Experiment 7: The impact of the failure rate of gateways in the IoT system is analyzed.

For experiment 1, the parameters for the 4 architectures are shown in TABLE.4.4. The analysis results will be given later. In this table, "$fm = 1/10$" means the failure of smart meters occurs every 10 years. "$fs = 1/10$" means the failure of smart sockets occurs every 10 years. "$fg = 1/10$" means the failure of smart gateway occurs every 10 years. "nm","ns" and "ng" are the number of smart meters, smart sockets, and gateways respectively.

For experiment 2-7, the impact of different components will be analyzed. Architecture I will be chosen for analysis. The parameters for experiment 2-7 is shown in TABLE. 4.5, the failure rates of the components are supported by [46]and [42].

Experiment 1: The reliability analysis of the 4 architectures: The 4 CTMC models are built in PRISM to analyze the reliability of them. The experiment results are shown in Fig. 4.2 to show the reliability of the 4 architectures.

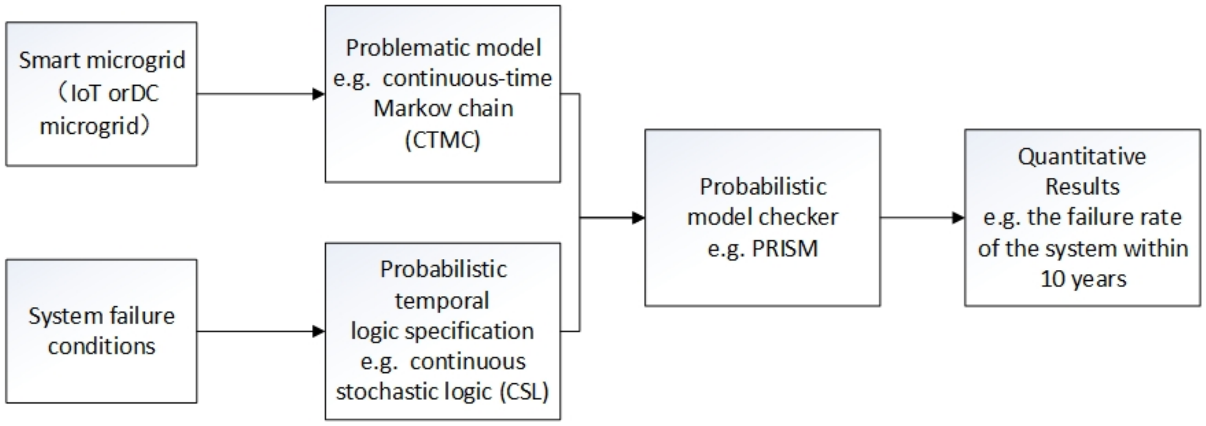

Smart microgrid （IoT orDC microgrid）
Problematic model e.g. continuous-time Markov chain (CTMC)
System failure conditions
Probabilistic temporal logic specification e.g. continuous stochastic logic (CSL)
Probabilistic model checker e.g. PRISM
Quantitative Results e.g. the failure rate of the system within 10 years

Figure 4.1: Model checking technique.

Table 4.1: Experiment environment

Experiment methods	**CPU**	**RAM**	**Operating system**	**Software**
CTMC	3.20GHz (i5-4460)	16G	Windows 7, 64-bit	PRISM

Table 4.2: The construction of different CTMC models

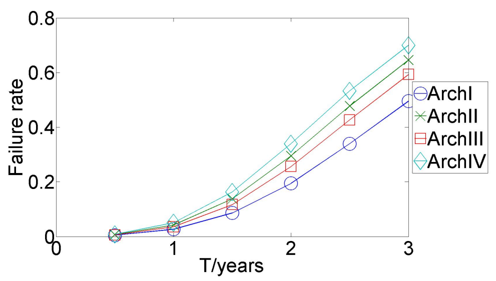
	Architecture I	**Architecture II**	**Architecture III**	**Architecture IV**
S	$S = \{s_0, \mathrm{s}_1, \mathrm{s}_2, s_3..., s_n\}$	$S = \{s_0, \mathrm{s}_1, \mathrm{s}_2, s_3..., s_n\}$	$S = \{s_0, \mathrm{s}_1, \mathrm{s}_2, s_3..., s_n\}$	$S = \{s_0, \mathrm{s}_1, \mathrm{s}_2, s_3..., s_n\}$
s_0	all smart sockets, smart meters, and gateways are available	all smart sockets, smart meters, and gateways are available	all smart sockets, smart meters, and gateways are available	all smart sockets, smart meters, root gateway and local gateways are available
L	OK,Fail;	OK,Fail ;	OK,Fail	OK,Fail

Table 4.3: The difference between the model code of the 4 architectures

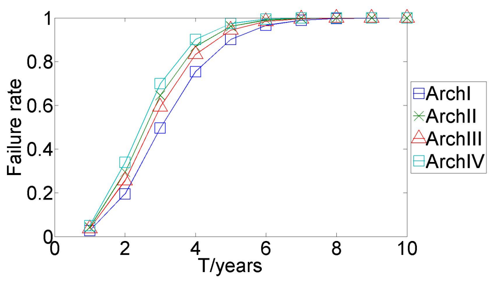
	Architecture I	**Architecture II**	**Architecture III**	**Architecture IV**
Smart meter sectors	nm : [0..2] init 2 ;	nm : [0..2] init 2 ;	nm : [0..2] init 2;	nm : [0..2] init 2;
Smart socket sectors	ns : [0..20] init 20 ;	ns : [0..20] init 20 ;	ns : [0..20] init 20;	ns : [0..20] init 20;
Number of gateways	ng : [0..2] init 2 ;	ng : [0..2] init 2 ;	ng1 : [0..1] init 1 ;ng2 : [0..1] init 1 ;	ng1 : [0..1] init 1 ;ng2 : [0..1] init 1 ; ng3 : [0..1] init 1 ;

Table 4.4: The paremeter of experiment 1

	Architecture I	**Architecture II**	**Architecture III**	**Architecture IV**
Smart meters	nm=2,fm=1/10;	nm=2,fm=1/10;	nm=2,fm=1/10	nm=2,fm=1/10;
Smart sockets	ns=20,fs=1/10;	ns=20,fs=1/10;	ns=20,fs=1/10;	ns=20,fs=1/10;
Gateways	ng=2,fg=1/10;	ng=2,fg=1/10;	ng=2,fg=1/10;	ng=2,fg=1/10;



(a) The architecture comparison of the 4 architectures in 3 years .

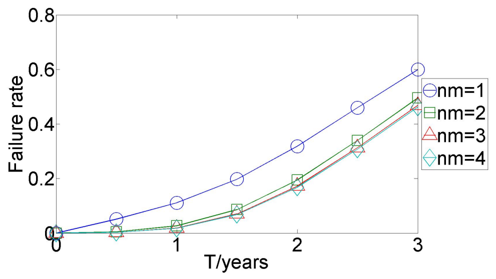


(b) The architecture comparison of the 4 architectures in 10 years.

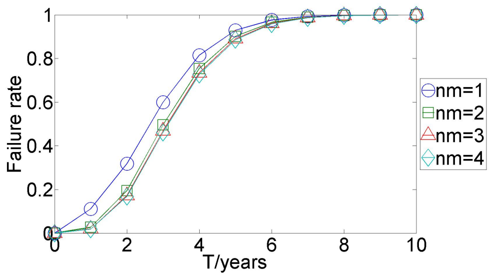
Figure 4.2: Experiment 1: The reliability analysis of the 4 architectures.

Table 4.5: Premeters for experiment 2 to 7

	Smart meters	**Smart sockets**	**Gateways**
Experiment 2	$nm = 1, 2, 3, 4; fm = 1/10$	$ns = 20, fs = 1/10$	$ng = 2, fg = 1/10$
Experiment 3	$nm = 2; fm = 1/5, fm = 1/10, fm = 1/15, fm = 1/20;$	$ns = 20, fs = 1/10$	$ng = 2, fg = 1/10$
Experiment 4	$nm = 2; fm = 1/10$	$ns = 18, 20, 22, 24, fs = 1/10$	$ng = 2, fg = 1/10$
Experiment 5	$nm = 2; fm = 1/10$	$ns = 20, fs = 1/5, fs = 1/10, fs = 1/15, fs = 1/20$	$ng = 2, fg = 1/10$
Experiment 6	$nm = 2; fm = 1/10$	$ns = 20, fs = 1/10$	$ng = 1, 2, 3, 4, fg = 1/10$
Experiment 7	$nm = 2; fm = 1/10$	$ns = 20, fs = 1/5, fs = 1/10, fs = 1/15, fs = 1/20$	$ng = 2, fg = 1/5, 1/10, 1/15, 1/20$

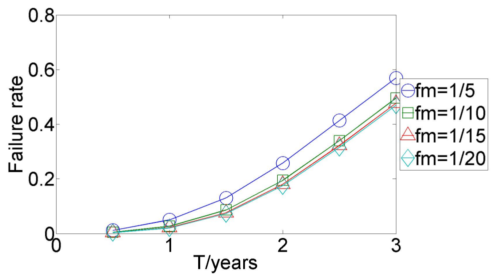


(a) The failure rate of IoT system in 3 years with different number of smart meters.

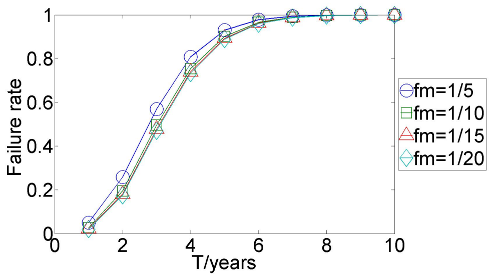


(b) The failure rate of IoT system in 10 years with different number of smart meters.

Figure 4.3: Experiment 2: The impact of nm.



(a) The failure rate of IoT system in 3 years with different failure rate of smart meters.



(b) The failure rate of IoT system in 10 years with different failure rate of smart meters.

Figure 4.4: Experiment 3: The impact of fm.

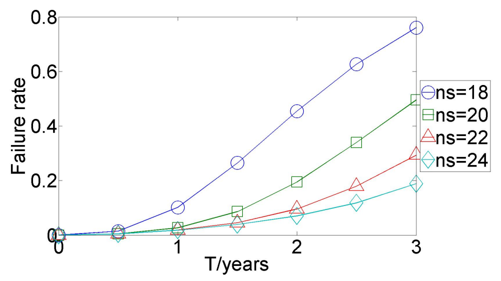
Finding 1 (the reliability of the 4 architectures): Architecture I is better than other architectures. The failure rate of architecture IV is highest.

Experiment 2: The relationship between nm and $fiot$: In this experiment, nm in Architecture I is changed to analyze the impact of it. The experiment results are shown in Fig. 4.3 to show impact of nm.

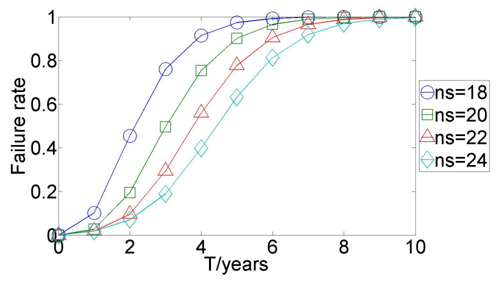
Finding 2 (The relationship between nm and $fiot$): $fiot$ is inversely proportional to nm. However, after nm increase to 3, the impact of nm is not obvious.

Experiment 3: The relationship between fm and $fiot$: In this experiment, fm in Architecture I is changed to analyze the impact of it. The experiment results are shown in Fig.4.4 to show the impact of fm.

Finding 3 (The relationship between fm and $fiot$): $fiot$ is proportional to fm. However, after fm increase to 15 years once, the impact of fm is not obvious.

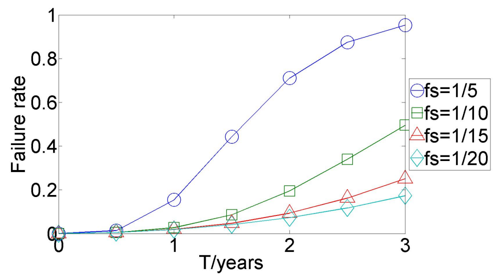


(a) The failure rate of IoT system in 3 years with different number of smart sockets.

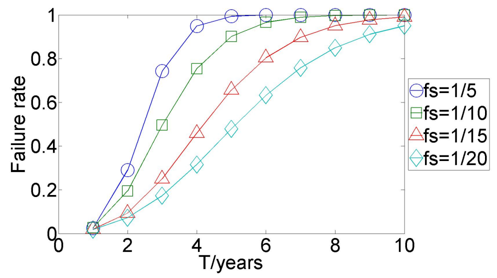


(b) The failure rate of IoT system in 10 years with different number of smart sockets.

Figure 4.5: Experiment 4: The impact of ns.



(a) The failure rate of IoT system in 3 years with different failure rate of smart sockets.



(b) The failure rate of IoT system in 10 years with different failure rate of smart sockets.

Figure 4.6: Experiment 5: The impact of fs.

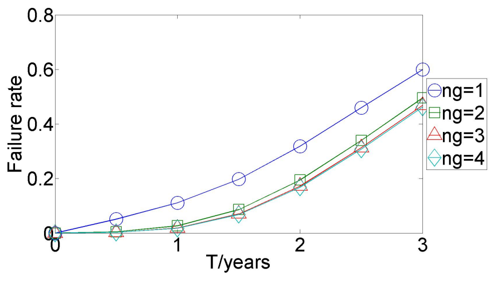
Experiment 4: The relationship between ns and $fiot$: In this experiment, ns in Architecture I is changed to analyze the impact of it. The experiment results are shown in Fig.4.5 to show the impact of ns.

Finding 4 (The relationship between ns and $fiot$): $fiot$ is inversely proportional to ns. The impact of fs is very obvious.

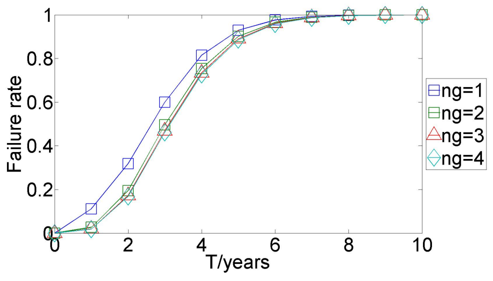
Experiment 5: The relationship between fs and $fiot$: In this experiment, fs in Architecture I is changed to analyze the impact of it. The experiment results are shown in Fig.4.6 to show the impact of fs.

Finding 5 (The relationship between fs and $fiot$): $fiot$ is proportional to fs. The impact of fs is very obvious.

Experiment 6: The relationship between ng and $fiot$: In this experiment, ng in Architecture I is changed to analyze the impact of it. The experiment results are shown

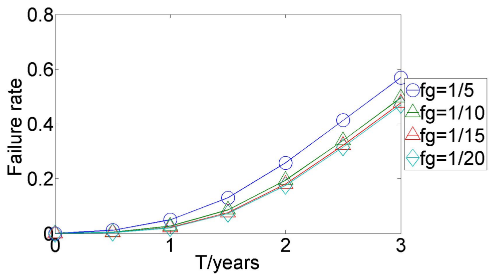


(a) The failure rate of IoT system in 3 years with different number of gateways.

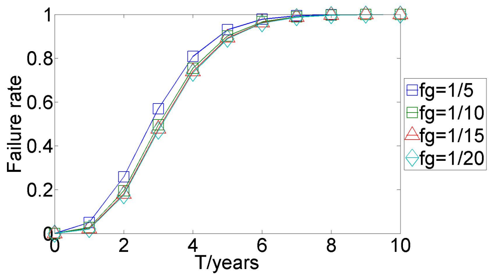


(b) The failure rate of IoT system in 10 years with different number of gateways.

Figure 4.7: Experiment 6: The impact of ng.



(a) The failure rate of IoT system in 3 years with different number of gateways.



(b) The failure rate of IoT system in 10 years with different number of gateways.

Figure 4.8: Experiment 7: The impact of fg.

in Fig.4.7 to show the impact of ng.

Finding 6 (The relationship between ng and $fiot$): $fiot$ is inversely proportional to fs. However, if ng increase to 3, the impact of it is not obvious.

Experiment 7: The relationship between fg and $fiot$: In this experiment, fg in Architecture I is changed to analyze the impact of it. The experiment results are shown in Fig.4.8 to show the impact of fg.

Finding 7 (The relationship between fg and $fiot$): $fiot$ is proportional to fs. However, if fg increase to 15 years once, the impact of it is not obvious.

4.5 CMTC Model Design of the DC Microgrid System

4.5.1 Architecture I

PV Layer Module: this module represents the number of workable sensor nodes. "λ_p " is the failure rate of PV panels. For initial state, if 1 PV panel fails, the process will be shown as:

$$np \xrightarrow{\lambda_p} np - 1 \tag{4.4}$$

Inverter Layer Module: The inverter layer has 4 inverters. Both the inverters have failure rate "λ_i". For initial state, if 1 inverter fails with failure rate "λ_i ", the process is shown as:

$$ni \xrightarrow{\lambda_i} ni - 1 \tag{4.5}$$

Failure Conditions: The failure conditions of Architecture I are shown as follows:

- $ni < 2$
- $np < 2$

4.5.2 Architecture II and III

For architecture II and III, the CTMC models are similar to architecture I. Architecture II just have 1 centralized inverter, architecture III has DC/DC converters and centralized inverters.

4.6 CMTC Model Implementation of the DC Microgrid System

4.6.1 Architecture I

PV Layer Module: "np" is the available PV panel numbers, the initial number of "np" is 4. If 1 PV panel fails with failure rate "*lambda_p*", "np" will minus 1. The code for the PV panel layer is shown below:

```
np : [ 0 . . 4 ] init 4 ;
[] np>0 -> np * lambda_p : ( np' = np - 1 ) ;
```

Inverter Layer Module: "ni" is the available inverter numbers, the initial number of "ni" is 4. If 1 inverter fails with failure rate "*lambda_i*", "ni" will minus 1. The code for the inverter layer is shown below:

```
ni : [ 0 . . 4 ] init 4 ;
[] ni>0 -> ni * lambda_i : ( ni' = ni - 1 ) ;
```

The Condition for Failure of The DC Microgrid : If "np" is smaller than lower bound, or "ni" is smaller than lower bound of gateways, the system fails. The code for failure conditions is shown as follows:

```
formula down =(np<2) | (ni<2);
```

4.6.2 Architecture II and III

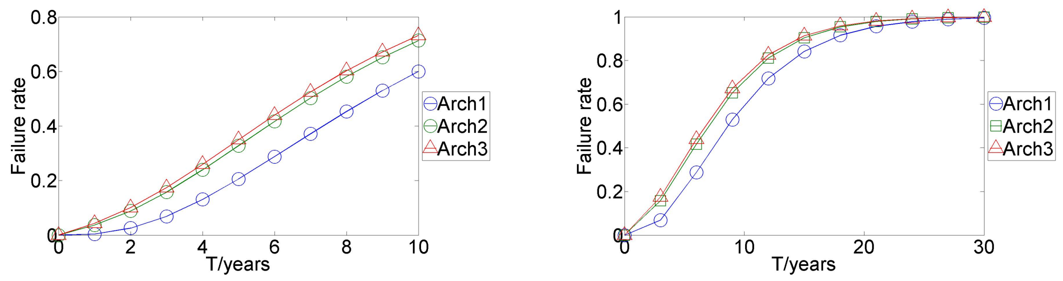
Architecture II and III are also constructed in PRISM. The modeling of other architectures are similar to Architecture I, the differences are shown in TABLE.4.6.

- Firstly, all architectures have 4 PV panels (np=4).
- Secondly, Architecture I has 4 inverters (i=2), architecture II has 1 inverter (i=1). Architecture III has 4 converters(c=4) and 1 inverter(i=1).

4.7 Results Analysis of DC Microgrid

After the CTMC models for DC microgrid are built in PRISM, we design different experiments to show our architecture is better than other architectures. What's more, for the optimized architecture, the impact of different components are analyzed. fdc is the failure rate of the DC microgrid. The experiments are divided into 5 parts:

- Experiment 1: The reliability of different architectures of DC microgrid are analyzed to show architecture I is better.
- Experiment 2: The impact of PV panels numbers in DC microgrid is analyzed.
- Experiment 3: The impact of the failure rate of PV panels in DC microgrid is analyzed.
- Experiment 4: The impact of inverter numbers on DC microgrid is analyzed.



(a) The architecture comparison of the 3 architectures in 10 years. (b) The architecture comparison of the 3 architectures in 30 years.

Figure 4.9: Experiment1: The architecture comparison of the 3 architectures.

- Experiment 5: The impact of the failure rate of inverters on DC microgrid is analyzed.

For experiment 1, the parameters for the 3 architectures are shown in TABLE.4.7. The failure rate of the components are supported by [45] and [44]. The analysis results will be given later. In this table, "$fp = 1/20$" means the failure of a PV panel occurs every 20 years. "$fi = 1/10$" means the failure of inverters occurs every 20 years. "np" and "ni" are the number of PV panels and inverters respectively.

For experiment 2-5, the impact of different components will be analyzed. Architecture I will be chosen for analysis. The parameters for experiment 2 to 5 is shown in TABLE. 4.8

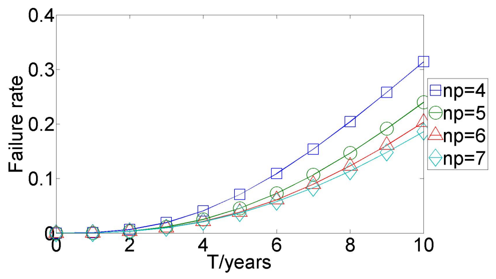
Experiment 1: The reliability analysis of the 3 architectures: The 3 CTMC models are built in PRISM to analyze the reliability of them. The experiment results about the reliability of IoT is shown in Figure.4.9.

Finding 1 (The reliability comparison of the 3 architectures) : Architecture I is the most reliable one. The failure rate of Architecture III is the highest.

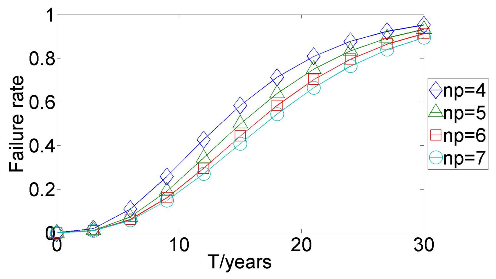
Experiment 2: The relationship between np and fdc: In this experiment, np in Architecture I is changed to analyze the impact of it. The experiment results are shown in Fig.4.10.

Finding 2 (The relationship between np and fdc): fdc is inversely proportional to np. However, if np increase to 6, the impact of it become not obvious.

Experiment 3: The relationship between fp and fdc: In this experiment, fp in Architecture I is changed to analyze the impact of it. The experiment results are shown in Fig.4.11.

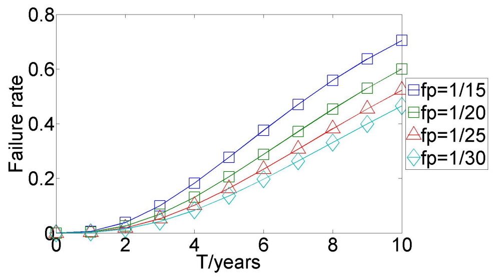


(a) The failure rate of DC microgrid in 10 years with different number of PV panels.

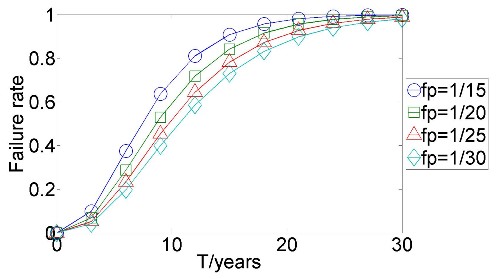


(b) The failure rate of DC microgrid in 30 years with different number of PV panels.

Figure 4.10: Experiment 2: The impact of np.



(a) The failure rate of DC microgrid in 10 years with different failure rate of PV panels.



(b) The failure rate of DC microgrid in 30 years with different failure rate of PV panels.

Figure 4.11: Experiment 3: The impact of fp.

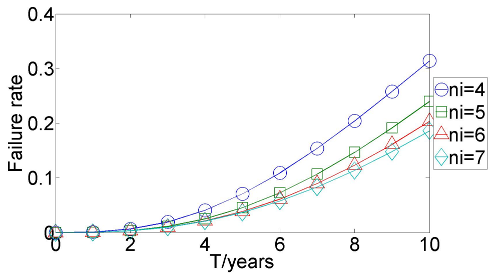
Finding 3 (The relationship between fp and fdc): fdc is inversely proportional to fp. The impact of fp is not obvious.

Experiment 4: The relationship between ni and fdc: In this experiment, ni in Architecture I is changed to analyze the impact of it. The experiment results are shown in Fig.4.12.

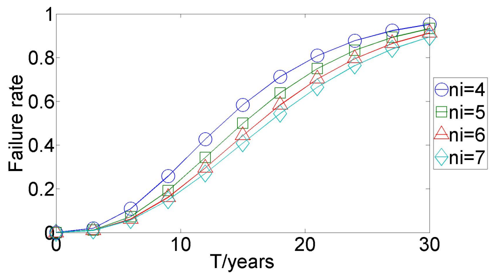
Finding 4 (The relationship between ni and fdc): fdc is inversely proportional to ni. The impact of ni is not obvious.

Experiment 5: The relationship between fi and fdc: In this experiment, fi in Architecture I is changed to analyze the impact of it. The experiment results are shown in Fig.4.13.

Finding 5 (The relationship between fi and fdc): fdc is proportional to fi. The impact of fi is obvious.

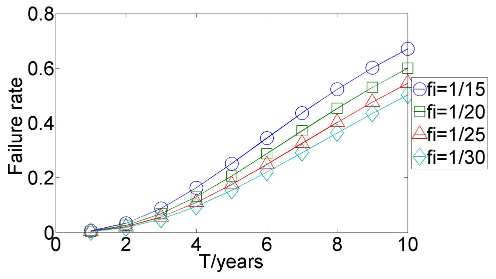


(a) The failure rate of DC microgrid in 10 years with different failure rate of inverters.

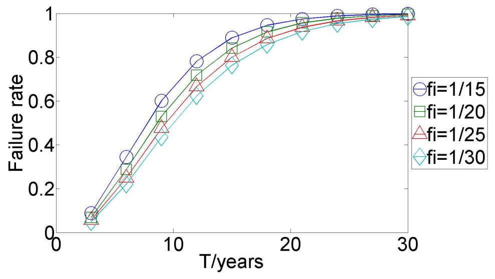


(b) The failure rate of DC microgrid in 30 years with different failure rate of inverters.

Figure 4.12: Experiment 4: The impact of ni.



(a) The failure rate of DC microgrid in 10 years with different failure rate of inverters.



(b) The failure rate of DC microgrid in 30 years with different failure rate of inverters.

Figure 4.13: Experiment 5: The impact of fi.

Table 4.6: The difference between the model code of the 3 architectures

	Architecture I	**Architecture II**	**Architecture III**
PV panel sector	np : [0..4] init 4 ;	np : [0..4] init 4 ;	np : [0..4] init 4;
Inverter sectors	ni : [0..4] init 4 ;	ni : [0..1] init 1 ;	ni : [0..1] init 1;
Converter sectors	/	/	nc : [0..4] init 4 ;

Table 4.7: The parameters of Experiments 1

	Architecture I	**Architecture II**	**Architecture III**
PV Pannels	np=4,fp=1/20;	np=4,fp=1/20;	np=4,fp=1/20;
Inverters	ni=4,fi=1/20;	ni=1,fi=1/20;	ni=1,fi=1/20;
Converters	/	/	nc=4,fc=1/20;

Table 4.8: Premeters for experiment 2 to 5

	PV Pannels	**Inverters**
Experiment 2	$np = 4, 5, 6, 7; fp = 1/20;$	$ni = 4, fi = 1/20;$
Experiment 3	$np = 4; fp = 1/15, 1/20, 1/25, 1/30;$	$ni = 4, fi = 1/20;$
Experiment 4	$np = 4, fp = 1/20;$	$ni = 4, 5, 6, 7; fi = 1/20;$
Experiment 5	$np = 4; fp = 1/20;$	$ni = 4; fi = 1/15, 1/20, 1/25, 1/30;$

Chapter 5

The Benchmarks for The Model Checking of The Smart Microgrid

After we build CMTC models for the smart microgrid, the benchmarks are needed to verify the correctness of the CTMC models. In this project, we use RBD method and Monte Carlo method as benchmarks to show our CTMC models are correct. The RBD models and Monte Carlo methods are shown as follows:

5.1 Benchmark 1: Reliability Block Diagram Method

Reliability Block Diagram (RBD) is a widely used method for reliability analysis[30]. Based on RBD, we can evaluate the IoT system failure rate $fiot(t)$ of ArchI to ArchIV by using RBD algorithms.

5.1.1 The Implementation of RBD Models for IoT System

The algorithm for the RBD model of IoT system is shown as follows:

The RBD models are built in Matlab, the process of RBD model design is shown as follows:

```
%% parameters
% the failure rate of a single smart meter
lambda_m = 1/10;
% the number of smart meters
```

```
num_m = 2;
```

"*lamda_m*" is the failure rate of the smart meter,"*num_m*" is the number of the smart meters. The smart meters fail with a failure rate of "*lamda_m*".

The parameters for the smart meter is shown as follows:

```
% the failure rate of a single smart socket
lambda_s = 1/10;
% the number of smart socket nodes
num_s = 20;
```

"*lamda_s*" is the failure rate of the smart socket,"*num_s*" is the number of smart sockets. The smart sockets fail with a failure rate "*lamda_s*".

The parameters for the gateways are shown as follows:

```
% the failure rate of a single gateway
lambda_g = 1/10;
% the number of gateways
num_g = 20;
```

The "*lamda_g*" is the failure rate of gateways,"*num_g*" is the number of gateways. The gateways fail with a failure rate of "*lamda_g*".

The failure rate of smart meters, smart sockets and gateways are between 0 and 1. They are shown as follows:

```
fp_m = zeros(1,duration);
fp_s = zeros(1,duration);
fp_g = zeros(1,duration);
failure_iot = zeros(1,duration);
```

The simulation time is 10 years, which is shown as follows:

```
% total simulation time
duration = 10;
day = (1:duration);
```

The failure rate of the whole system is calculated as follows:

```
for t = 1 : duration
    % failure probability of smart meter layer on each day
```

```
        fp_m(t) = (1-exp(-lambda_m*t))^num_m;
        % failure probability of smart socket layer on each day
        fp_s(t) = (1-exp(-lambda_s*t))^num_s;
        % failure probability of gateway layer on each day
        fp_g(t) = (1-exp(-lambda_g*t))^num_g;


        failure_iot(t) = fp_m(t) + fp_s(t)+ fp_g(t) - fp_m(t)*fp_s(t
            )-fp_m(t)*fp_g(t)-fp_s(t)*fp_g(t)+fp_m(t)*fp_s(t)*fp_g(t)
            ;
    end

  plot(day, failure_iot);
```

After the failure rate of the whole system is calculated, then Matlab will plot it. According to the 7 experiments about reliability analysis of IoT system, we can modify the parameters to get the experiment results. For the reliability analysis of another 3 architectures, the RBD models are similar.

5.1.2 The Experiments Results of RBD for IoT System

According to the 7 experiments about reliability analysis of IoT system. The experiment results of RBD models are shown as below as benchmarks. The experiment results are shown in Fig. 5.2 to Fig. 5.8.

For experiment 1 to 7, we can see that the experiment results are similar to our CMTC models.

5.1.3 The Implementation of RBD Models for DC Microgrid

The RBD models are built in Matlab, the process of RBD model design is shown as follows. The PV panel module is shown below.

```
%% parameters
% the failure rate of a single PV pannel
lambda_p = 1/20;
% the number of PV pannels
```

```
num_p = 4;
```

"*lamda_p*" is the failure rate of one PV panel,"*num_p*" is the number of PV panels. The PV panels fail with a failure rate "*lamda_p*".

The inverter module is shown below.

```
%% parameters
% the failure rate of a single inverter
lambda_i = 1/20;
% the number of inverter
num_i = 4;
```

"*lamda_i*" is the failure rate of one inverter,"*num_i*" is the number of inverters. The inverters fail with a failure rate of "*lamda_i*".

The failure rate of PV panel and inverter are between 0 and 1. They are shown as follows:

```
fp_p = zeros(1,duration);
fp_i = zeros(1,duration);
failure_dc = zeros(1,duration);
```

The simulation time is 10 years, which is shown as follows:

```
% total simulation time
duration = 10;
day = (1:duration);
```

The failure rate of the DC microgrid in tth day is shown as follows:

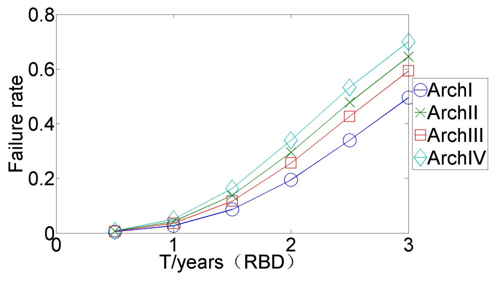
```
for t = 1 : duration
    % failure probability of PV panel layer on each day
    fp_p(t) = (1-exp(-lambda_p*t))^num_p;
    % failure probability of inverter layer on each day
    fp_i(t) = (1-exp(-lambda_i*t))^num_i;

    failure_dc(t) = fp_p(t) + fp_i(t)- fp_p(t)*fp_i(t);
  end

  plot(day,failure_dc);
```

After the failure rate of the whole DC microgrid is calculated, then Matlab will plot it. According to the 5 experiments about reliability analysis of IoT system, we can modify the parameters to get the experiment results. For the reliability analysis of another 2 architectures, the RBD models are similar.

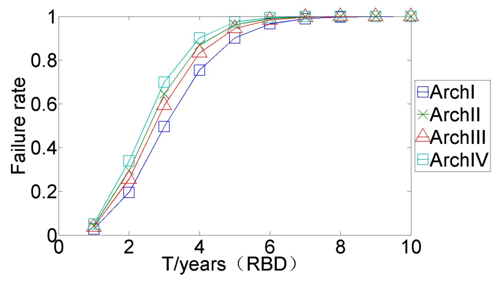
According to the 5 experiments about reliability analysis of DC microgrid system. The experiment results of RBD models are shown below as benchmarks. The experiment results are shown in Fig. 5.10 to Fig.5.14.

```
Algorithm 1 RBD for ArchI
for time $t = 1$ to $T$ ($T$ will be 3 years or 10 years in experiments.) do
    1. Compute failure rate of smart meter layer $fm(t) = (1 - e^{-fm \times t})^{nm}$;
    2. Compute failure rate of smart socket layer $fs(t) = (1 - e^{-fs \times t})^{ns}$;
    3. Compute failure rate of gateway layer $fg(t) = (1 - e^{-fg \times t})^{ng}$;
    4. Compute failure rate of the IoT system $fiot(t) = fm(t) + fs(t) + fg(t) - fm(t) \times fs(t) - fm(t) \times fg(t) - fs(t) \times fg(t) + fm(t) \times fs(t) \times fg(t)$;
end for
```

Figure 5.1: The RBD algorithm for the IoT system(Architecture I).

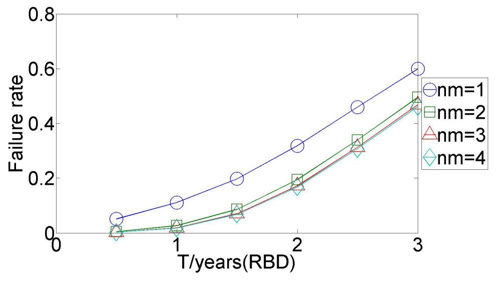


(a) The reliability analysis of the 4 architectures in 3 years (RBD)

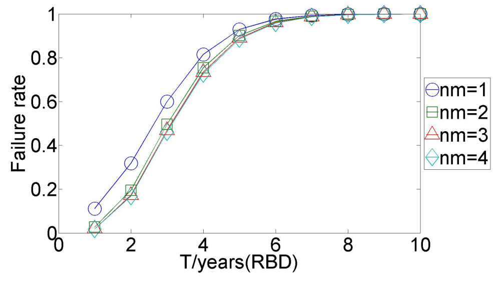


(b) The reliability analysis of the 4 architectures in 10 years (RBD)

Figure 5.2: Experiment 1: The reliability analysis of the 4 architectures (RBD method).

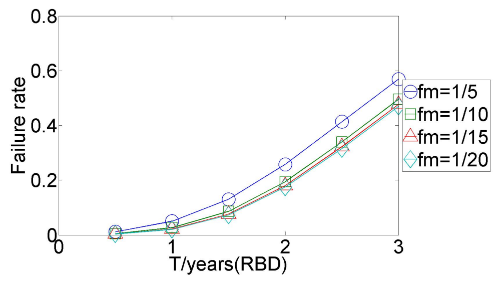


(a) The failure rate of IoT system in 3 years with different number of smart meters (RBD).

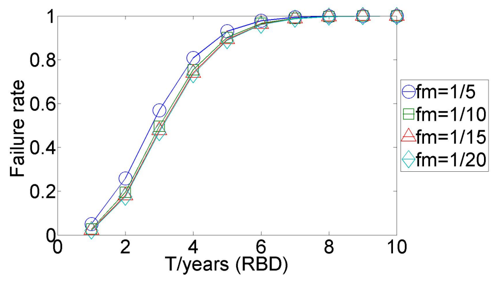


(b) The failure rate of IoT system in 10 years with different number of smart meters (RBD).

Figure 5.3: Experiment 2: The impact of nm (RBD method)

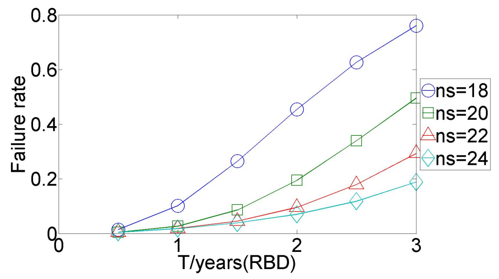


(a) The failure rate of IoT system in 3 years with different failure rate of smart meters(RBD).

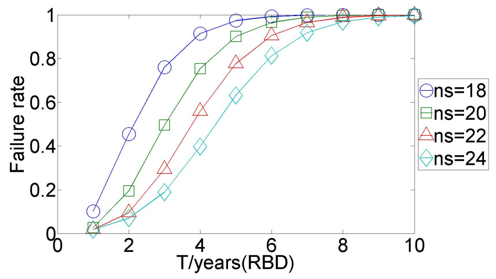


(b) The failure rate of IoT system in 10 years with different failure rate of smart meters(RBD).

Figure 5.4: Experiment 3: The impact of fm (RBD method).

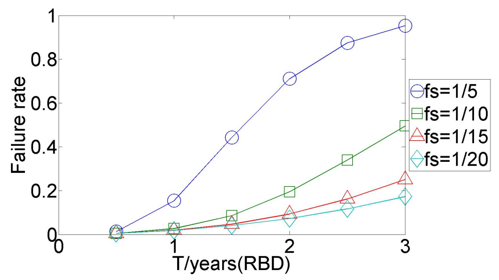


(a) The failure rate of IoT system in 3 years with different number of smart sockets (RBD).

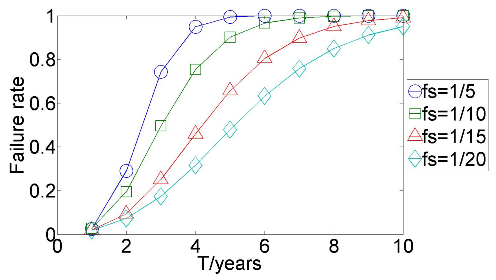


(b) The failure rate of IoT system in 10 years with different number of smart sockets (RBD).

Figure 5.5: Experiment 4: The impact of ns (RBD method).

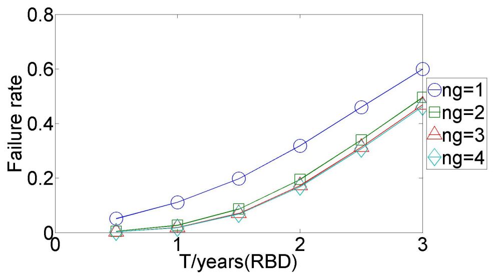


(a) The failure rate of IoT system in 3 years with different failure rate of smart sockets (RBD).

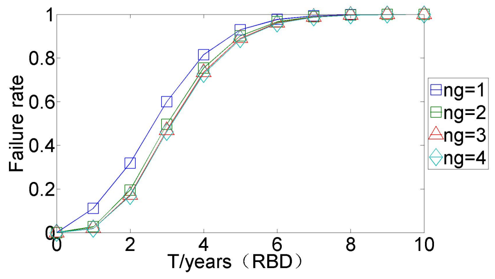


(b) The failure rate of IoT system in 10 years with different failure rate of smart sockets (RBD).

Figure 5.6: Experiment 5: The impact of fs (RBD method).

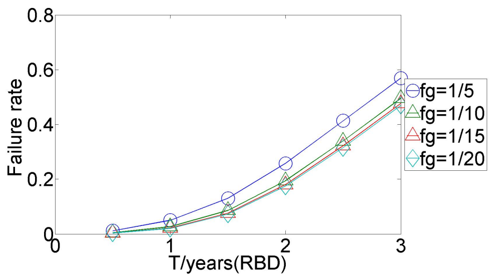


(a) The failure rate of IoT system in 3 years with different number of gateways (RBD)

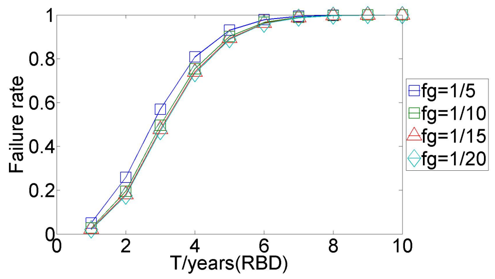


(b) The failure rate of IoT system in 10 years with different number of gateways (RBD)

Figure 5.7: Experiment 6: The impact of ng (RBD method).

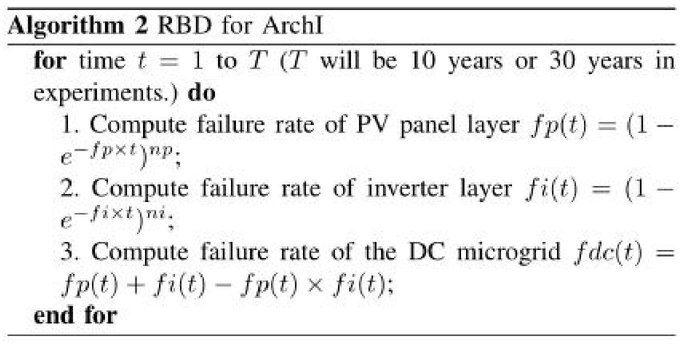


(a) The failure rate of IoT system in 3 years with different failure rate of gateways (RBD)



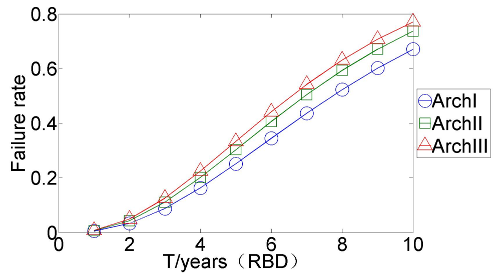
(b) The failure rate of IoT system in 10 years with different failure rate of gateways (RBD)

Figure 5.8: Experiment 7: The impact of fg (RBD method).

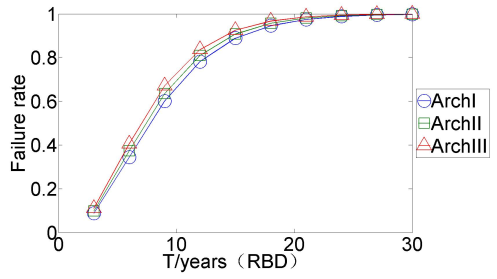


```
Algorithm 2 RBD for ArchI
  for time $t = 1$ to $T$ ($T$ will be 10 years or 30 years in experiments.) do
    1. Compute failure rate of PV panel layer $fp(t) = (1 - e^{-fp \times t})^{np}$;
    2. Compute failure rate of inverter layer $fi(t) = (1 - e^{-fi \times t})^{ni}$;
    3. Compute failure rate of the DC microgrid $fdc(t) = fp(t) + fi(t) - fp(t) \times fi(t)$;
  end for
```

Figure 5.9: The RBD algorithm for the DC microgrid(Architecture I).

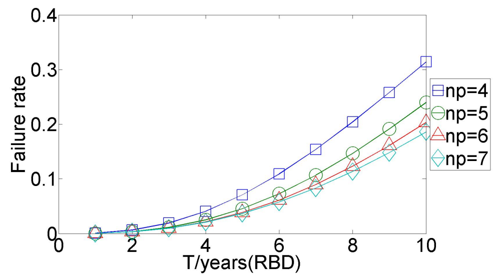


(a) The reliability analysis of the 3 architectures in 3 years (RBD)

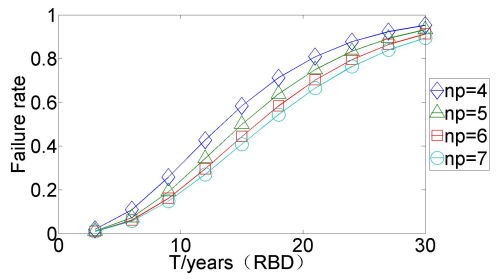


(b) The reliability analysis of the 3 architectures in 10 years (RBD)

Figure 5.10: Experiment 1: The reliability analysis of the 3 architectures of DC microgrid (RBD method).

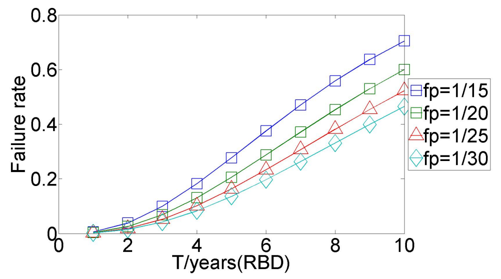


(a) The failure rate of DC microgrid system in 3 years with different number of PV panels

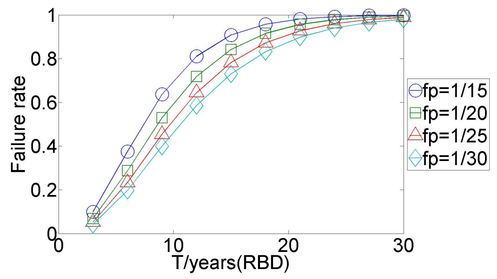


(b) The failure rate of DC microgrid system in 10 years with different number of PV panels

Figure 5.11: Experiment 2: The impact of np (RBD method).

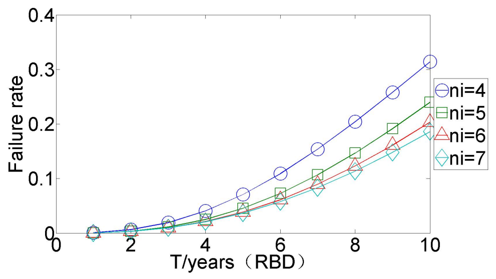


(a) The failure rate of DC microgrid system in 10 years with different failure rate of PV panels

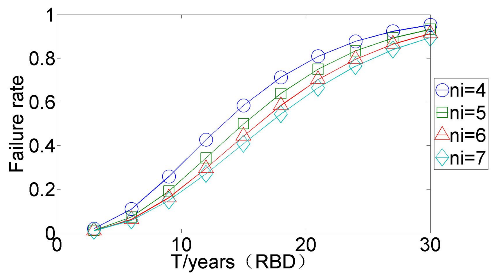


(b) The failure rate of DC microgrid in 30 years with different failure rate of PV panels

Figure 5.12: Experiment 3: The impact of fp (RBD method).

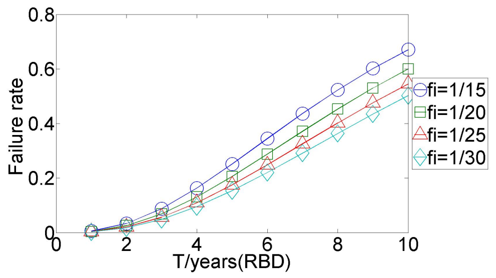


(a) The failure rate of DC microgrid in 10 years with different number of inverters

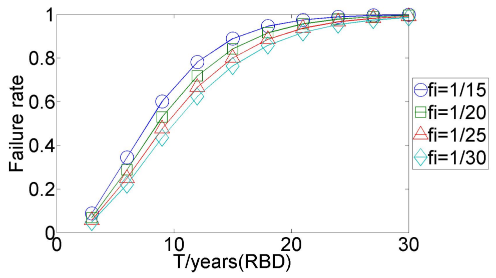


(b) The failure rate of DC microgrid in 30 years with different number of inverters

Figure 5.13: Experiment 4: The impact of ni (RBD method).



(a) The failure rate of DC microgrid in 10 years with different failure rate of inverters



(b) The failure rate of DC microgrid in 30 years with different failure rate of inverters

Figure 5.14: Experiment 5: The impact of fi (RBD method).

5.2 Benchmark 2: Monte Carlo Method

Monte Carlo method is also a widely used reliability analysis method. Based on MC, we can compute the failure rate of the smart microgrid; Monte Carlo method is also used as a benchmark to show our CMTC model is correct. The model implementation and experiment result analysis are shown below.

5.2.1 The Implementation of Monte Carlo Models for IoT System

Monte Carlos method is also a widely used method for the reliability analysis of IoT system.

```
The implementation of the Monte Carlo models is shown as follows.
    The parameters for the smart meters are shown as follows.
\begin{lstlisting}
%% parameters
% the failure rate of smart meters
lambda_m = 1/10;
% the number of smart meters
num_m= 2;
```

The "*lamda_m*" is the failure rate of smart meters,"*num_s*" is the number of smart meters. The smart meters fail with a failure rate of "*lamda_s*". The failure of the smart meters occurs once in 10 years.

The parameters about the smart sockets are shown as below:

```
%% parameters
% the failure rate of smart sockets
lambda_s = 1/10;
% the number of smart sockets
num_s= 20;
```

The "*lamda_s*" is the failure rate of the smart socket,"*num_s*" is the number of smart sockets. The smart sockets fail with a failure rate of "*lamda_s*". The failure of the smart sockets occurs once in 10 years.

The parameters about the gateways are shown as below:

```
%% parameters
% the failure rate of gateways
lambda_g = 1/10;
% the number of smart gateways
num_g= 2;
```

The "*lamda_g*" is the failure rate of gateways,"*num_g*" is the number of gateways. The smart gateways fail with a failure rate of "*lamda_g*". The failure of the gateways occurs once in 10 years.

The duration of the simulation time is shown as follows:

```
% total simulation time
duration = 10;
```

```
day = (1:duration);
```

The simulation times each day is also shown as follows. In this experiment, the simulation time is 10000.

```
% simulation times each year
n = 10000;
```

The parameters for the random number of smart meter, smart sockets and gateways are shown below.

```
rand_m = zeros(1,num_m);
normal_num_m = zeros(1,num_m);
critical_value_m = 1;

rand_s = zeros(1,num_s);
normal_num_s = zeros(1,num_s);
critical_value_s = 15;

rand_g = zeros(1,num_g);
normal_num_g = zeros(1,num_g);
critical_value_g = 1;

failure_iot = zeros(1,duration);
```

The failure rate of smart meter layer is shown as follows:

```
for i = 1 : duration
    count = 0;
    for j = 1 : n

        rand_m = rand(1,num_m);
        for k = 1 : num_m
            if ( rand_m(k) < 1-exp(-lambda_m*i) )
                % kth smart meter  fails
                normal_num_m(k) = 0;
            else
                % kth smart meter does not fail
                normal_num_m(k) = 1;
            end
        end
        if ( normal_num_m * normal_num_m' < critical_value_m )
            flag_m = 0;
        else
            flag_m = 1;
        end
```

The failure rate of the smart socket layer is shown as follows:

```
rand_s = rand(1,num_s);
        for k = 1 : num_s
            if ( rand_s(k) < 1-exp(-lambda_s*i) )
                % kth smart socket fails
                normal_num_s(k) = 0;
            else
                % kth smart socket does not fail
                normal_num_s(k) = 1;
            end
        end
        if ( normal_num_s * normal_num_s' < critical_value_s )
            flag_s = 0;
        else
            flag_s = 1;
        end
```

The failure rate of the gateway layer is shown as follows:

```
rand_g = rand(1,num_g);
        for k = 1 : num_g
            if ( rand_g(k) < 1-exp(-lambda_g*i) )
                % gateway fails
                normal_num_g(k) = 0;
            else
                %  gateway does not fail
                normal_num_g(k) = 1;
            end
        end
        if ( normal_num_g * normal_num_g' < critical_value_g )
            flag_g = 0;
        else
            flag_g = 1;
        end
```

The failure rate of the IoT system is calculated as follows:

```
if ( flag_m==1 && flag_s==1 && flag_g==1  )
            count = count + 0;
        else
            count = count + 1;
        end

        failure_iot(t) = count/n;
```

After the failure rate of the IoT system in t days is calculated, Matlab will plot the failure rate of the IoT system each day.

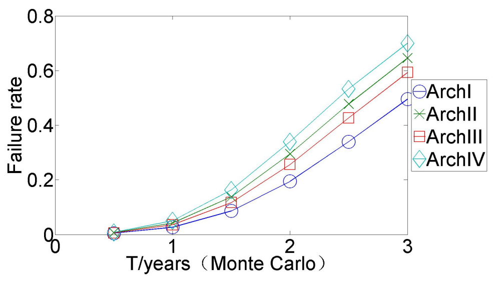
5.2.2 The Experiments Results of Monte Carlo Method for IoT System

According to the 7 experiments about reliability analysis of IoT system. The experiment results of Monte Carlo models are shown as below as benchmarks. The experiment results are shown in Fig. 5.16 to Fig.5.22.

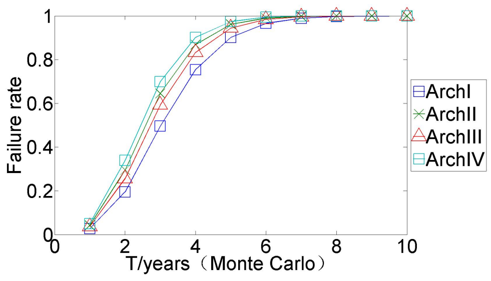
Algorithm 3 Monte Carlo method for ArchI

```
for time $t = 1$ to $T$ do
  Set the number of system failure $count = 0$.
  for number of experiment $n_{experiment} = 1$ to $N$ ($N$ will typically be 10000 in experiments.) do
    1. Compute the state of the $k$th smart meter $m_k$ .
    for $k = 1$ to $m_k$ do
      Generate a random value $rand\_m_k \in [0, 1]$.
      if $rand\_m_k < 1 - e^{-fm \times t}$ then
        $m_k = 0$
      else
        $m_k = 1$
      end if
    end for
    2. Compute the state of the $k$th smart socket $s_k$.
    for $k = 1$ to $ns$ do
      Generate a random value $rand\_s_k \in [0, 1]$.
      if $rand\_s_k < 1 - e^{-fs \times t}$ then
        $s_k = 0$
      else
        $s_k = 1$
      end if
    end for
    3. Compute the state of the $k$th gateway $g_k$.
    for $k = 1$ to $ng$ do
      Generate a random value $rand\_g_k \in [0, 1]$.
      if $rand\_g_k < 1 - e^{-fs \times t}$ then
        $g_k = 0$
      else
        $g_k = 1$
      end if
    end for
    4. Compute the state of the system. Assume $m'$ ,$s'$ and $g'$ are the transposed matrixes of $m$,$s$ and $g$ respectively. $m \times$
    is the number of working smart meters. $s \times s'$ is the sum of working smart sockets.$g \times g'$ is the sum of working sm
    sockets.
    if $m \times m' < 1$ or $s \times s' < 5$ or $g \times g' < 1$ then
      $count = count + 1$
    end if
  end for
  $fiot(t) = count/N$;
end for
```

Figure 5.15: The Monte Carlo algorithm for the IoT system(Architecture I).

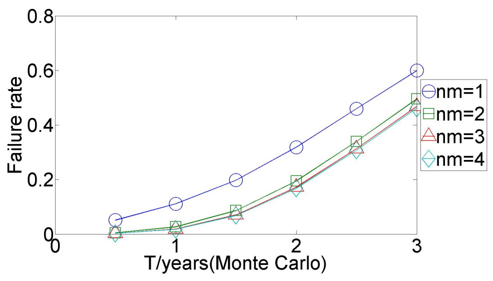


(a) The reliability analysis of the 4 architectures in 3 years (Monte Carlo)

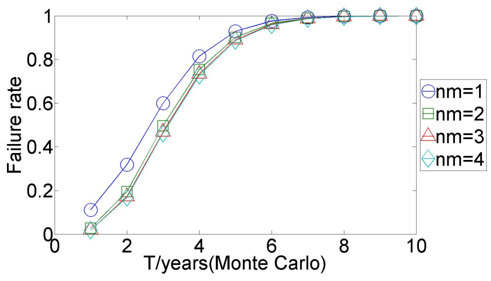


(b) The reliability analysis of the 4 architectures in 10 years (Monte Carlo)

Figure 5.16: Experiment1: The reliability analysis of the 4 architectures (Monte Carlo method).

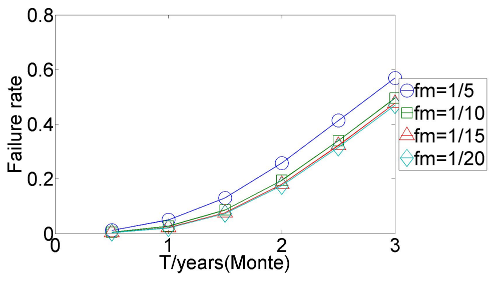


(a) The failure rate of IoT system in 3 years with different number of smart meters

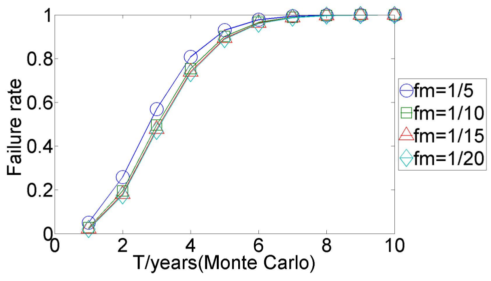


(b) The failure rate of IoT system in 10 years with different number of smart meters

Figure 5.17: Experiment2: The impact of nm (Monte Carlo method).

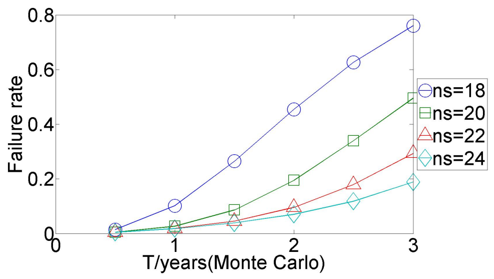


(a) The failure rate of IoT system in 3 years with different failure rate of smart meters

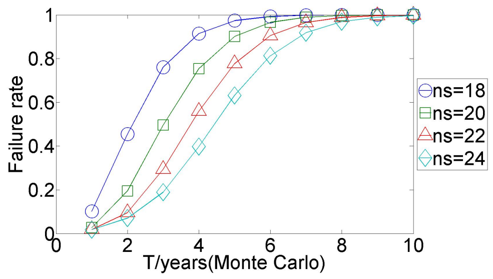


(b) The failure rate of IoT system in 10 years with different failure rate of smart meters

Figure 5.18: Experiment3: The impact of fm (Monte Carlo method).

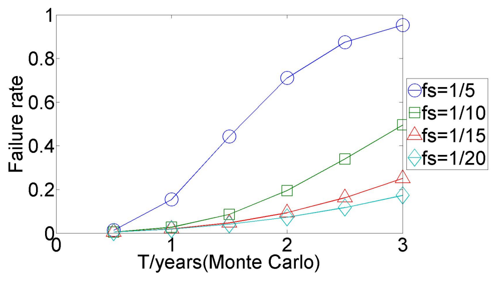


(a) The failure rate of IoT system in 3 years with different number of smart sockets

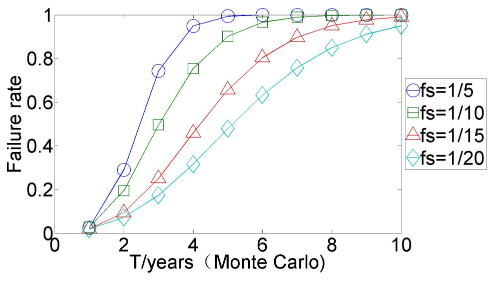


(b) The failure rate of IoT system in 10 years with different number of smart sockets

Figure 5.19: Experiment4: The impact of ns (Monte Carlo method).

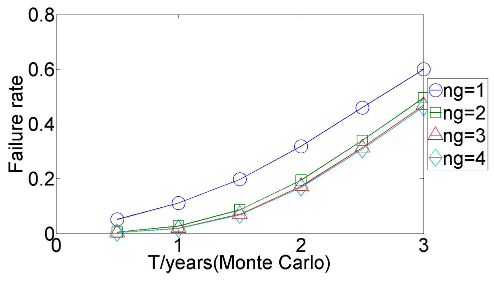


(a) The failure rate of IoT system in 3 years with different failure rate of smart sockets

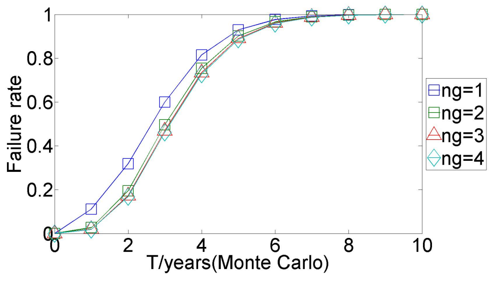


(b) The failure rate of IoT system in 10 years with different failure rate of smart sockets

Figure 5.20: Experiment5: The impact of fs (Monte Carlo method).

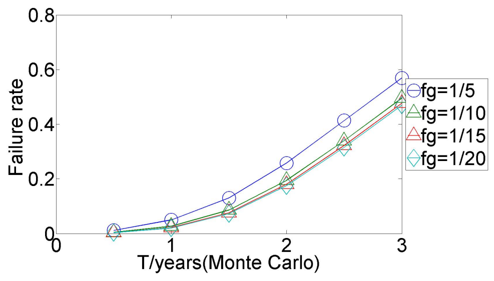


(a) The failure rate of IoT system in 3 years with different number of gateways

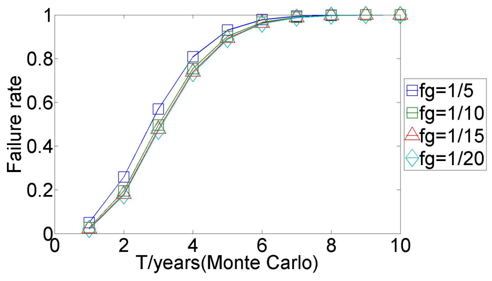


(b) The failure rate of IoT system in 10 years with different number of gateways

Figure 5.21: Experiment6: The impact of ng (Monte Carlo method).



(a) The failure rate of IoT system in 3 years with different failure rate of gateways



(b) The failure rate of IoT system in 10 years with different failure rate of gateways

Figure 5.22: Experiment7: The impact of fg (Monte Carlo method).

The experiments show that the results are similar to RBD models and our CTMC models are correct. We can see that our CTMC models are correct.

5.2.3 The Implementation of Monte Carlo Models for DC Microgrid

In this project, Monte Carlo models are also benchmarks for the reliability analysis of the DC microgrid.

```
 The parameters for the PV panels are shown as follows.
\begin{lstlisting}
%% parameters
% the failure rate of PV panels
lambda_p = 1/20;
% the number of PV panels
num_p= 4;
```

The "*lamda_p*" is the failure rate of PV panels,"*num_p*" is the number of PV panels. The PV panels fail with a failure rate of "*lamda_p*". The failure of the PV panels occurs once in 10 years.

The parameters for the inverters are shown as follows:

```
%% parameters
% the failure rate of inverters
lambda_i = 1/20;
% the number of inverters
num_i= 4;
```

The duration of the simulation time is shown as follows:

```
% total simulation time
duration = 10;
day = (1:duration);
```

The simulation times each day is also shown as follows. In this experiment, the simulation time is 10000.

```
% simulation times each year
n = 10000;
```

The parameters for the random number of PV panels, inverters are shown below.

```
rand_p = zeros(1,num_p);
normal_num_p = zeros(1,num_p);
critical_value_p = 1;

rand_i = zeros(1,num_i);
normal_num_i = zeros(1,num_i);
critical_value_i = 1;
failure_dc = zeros(1,duration);
```

The failure rate of PV panel layer is shown as follows:

```
for i = 1 : duration
    count = 0;
    for j = 1 : n

        rand_p = rand(1,num_p);
        for k = 1 : num_p
            if ( rand_p(k) < 1-exp(-lambda_p*i) )
                % kth PV panel fails
                normal_num_p(k) = 0;
            else
                % kth PV panel does not fail
                normal_num_p(k) = 1;
            end
        end
        if ( normal_num_p * normal_num_p' < critical_value_p )
            flag_p = 0;
        else
            flag_p= 1;
        end
```

The failure rate of the inverter layer is shown as follows:

```
for i = 1 : duration
    count = 0;
    for j = 1 : n

        rand_i = rand(1,num_i);
        for k = 1 : num_i
            if ( rand_i(k) < 1-exp(-lambda_i*i) )
                % kth inverter fails
                normal_num_i(k) = 0;
            else
                % kth inverter does not fail
                normal_num_i(k) = 1;
            end
        end
        if ( normal_num_i * normal_num_i' < critical_value_i )
            flag_i = 0;
        else
            flag_i= 1;
        end
```

The failure rate of the IoT system is calculated as follows:

```
if ( flag_p==1  && flag_i==1  )
            count = count + 0;
        else
            count = count + 1;
        end

        failure_iot(t) = count/n;
```

After the failure rate of DC microgrid in t days is calculated, Matlab will plot the failure rate of DC microgrid on each day.

5.2.4 The Experiment Results of Monte Carlo Models for DC Microgrid

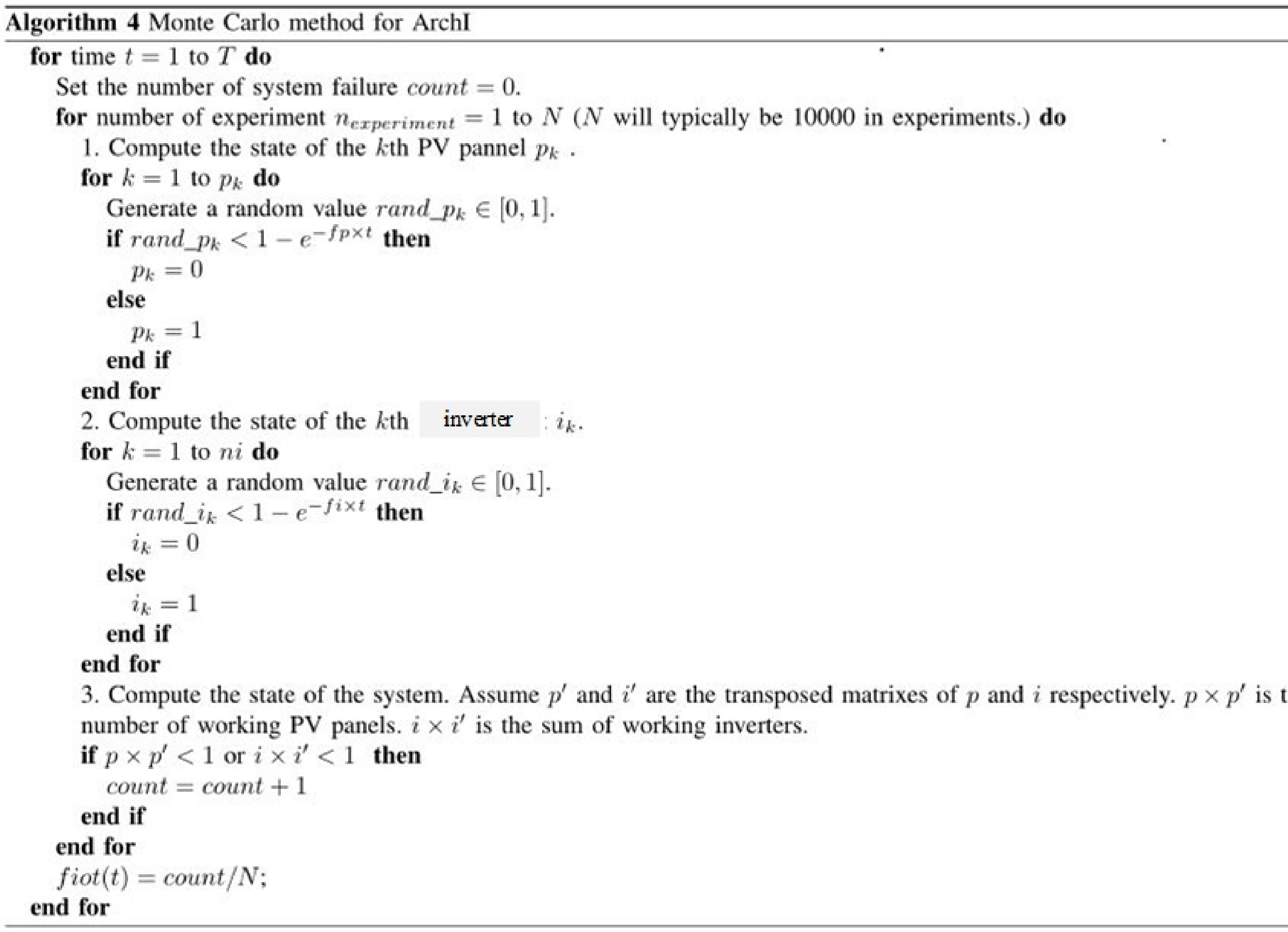
According to the 5 experiments about reliability analysis of IoT system. The experiment results of Monte Carlo models are shown below as benchmarks. The experiment results are shown in Fig. 5.24 to Fig.5.28.

The experiment results show that the results are similar to CTMC models, so our CTMC models are correct.

5.2.5 The Comparison Between Our CTMC models and Other Methods

The comparison of our CTMC models and other methods (IoT system) are shown in TABLE. 5.1. The experiment results in this table shows that the time for our CTMC models is shorter than other methods,so our CTMC models are more efficiency than the other methods. The comparison of our CTMC models and other methods (DC microgrid system) are shown in TABLE. 5.2. The experiment results in this table shows that the time for our CTMC models is shorter than other methods,so our CTMC models are more efficiency than the other methods.

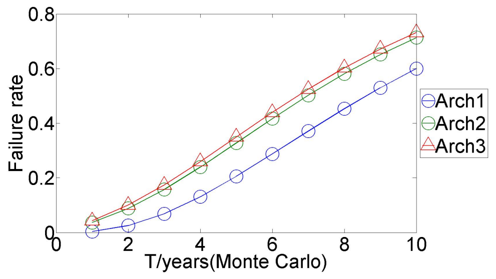
CTMC, RBD, and Monte Carlo method is used in reliability analysis of the network. For smart microgrid, the architecture of it is similar to networks, especially for IoT system. For CTMC, the efficiency of it is faster than other methods. What's more, CTMC is easier to build more complex models for more complex networkit is easier to connect the different modules and set the failure conditions. For other methodthe equations for a more complex model will be very complex and easy to cause errors. So CTMC is suitable for current research and further research about complex networks. For the smart grid, the structure of it is more complex than smart microgrid, so our CTMC is suitable for evaluating the reliability of the smart grid in further research.



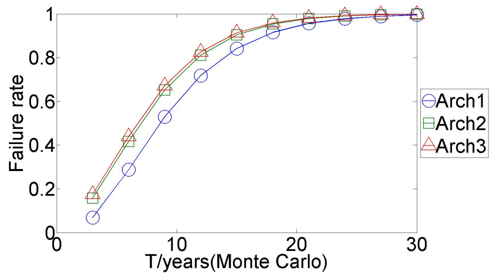
Algorithm 4 Monte Carlo method for ArchI

```
for time $t = 1$ to $T$ do
  Set the number of system failure $count = 0$.
  for number of experiment $n_{experiment} = 1$ to $N$ ($N$ will typically be 10000 in experiments.) do
    1. Compute the state of the $k$th PV pannel $p_k$ .
    for $k = 1$ to $p_k$ do
      Generate a random value $rand\_p_k \in [0, 1]$.
      if $rand\_p_k < 1 - e^{-fp \times t}$ then
        $p_k = 0$
      else
        $p_k = 1$
      end if
    end for
    2. Compute the state of the $k$th inverter : $i_k$.
    for $k = 1$ to $ni$ do
      Generate a random value $rand\_i_k \in [0, 1]$.
      if $rand\_i_k < 1 - e^{-fi \times t}$ then
        $i_k = 0$
      else
        $i_k = 1$
      end if
    end for
    3. Compute the state of the system. Assume $p'$ and $i'$ are the transposed matrixes of $p$ and $i$ respectively. $p \times p'$ is t
    number of working PV panels. $i \times i'$ is the sum of working inverters.
    if $p \times p' < 1$ or $i \times i' < 1$ then
      $count = count + 1$
    end if
  end for
  $fiot(t) = count/N$;
end for
```

Figure 5.23: The Monte Carlo algorithm for the DC microgrid (Architecture I).

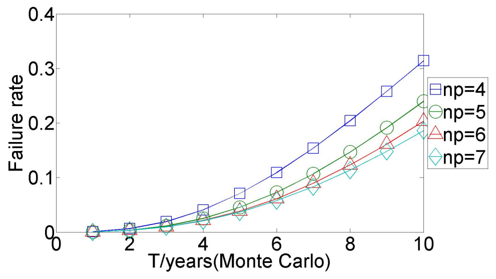


(a) The reliability analysis of the 4 architectures in 10 years (Monte Carlo)

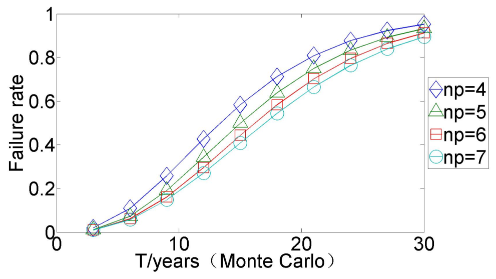


(b) The reliability analysis of the 4 architectures in 30 years (Monte Carlo)

Figure 5.24: Experiment1: The reliability analysis of the 3 architectures (Monte Carlo method).

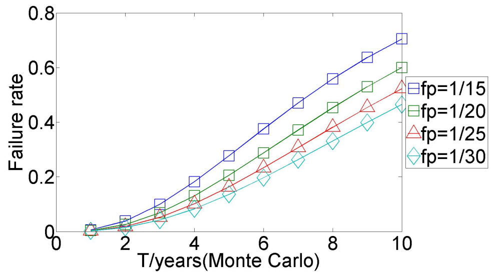


(a) The failure rate of DC microgrid in 10 years with different number of PV panels

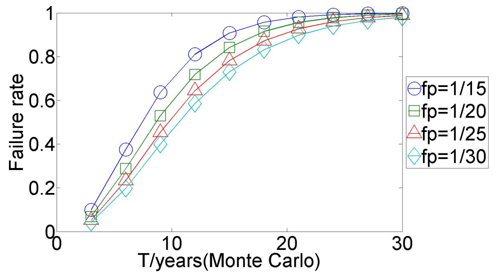


(b) The failure rate of DC microgrid in 30 years with different number of PV panels

Figure 5.25: Experiment2: The impact of np (Monte Carlo method).

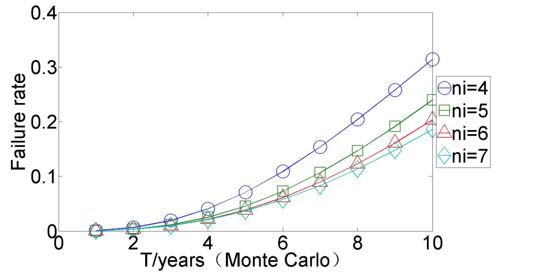


(a) The failure rate of DC microgrid in 10 years with different failure rate of PV panels

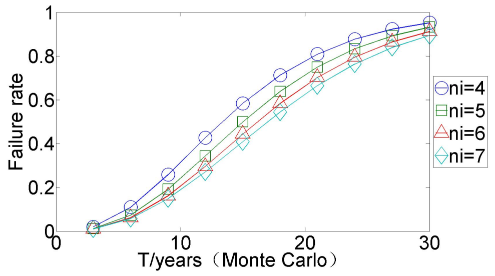


(b) The failure rate of DC microgrid in 30 years with different failure rate of PV panels

Figure 5.26: Experiment3: The impact of fp (Monte Carlo method).

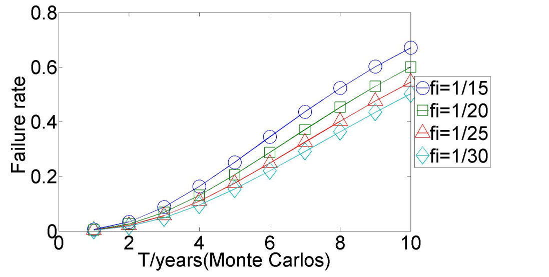


(a) The failure rate of DC microgrid in 10 years with different number of inverters(Monte Carlo method)

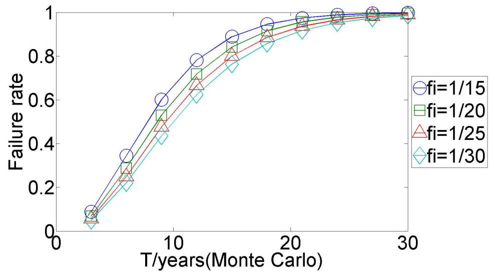


(b) The failure rate of DC microgrid in 30 years with different number of inverters (Monte Carlo method)

Figure 5.27: Experiment4: The impact of ni (Monte Carlo method).



(a) The failure rate of DC microgrid in 10 years with different failure rate of inverters (Monte Carlo)



(b) The failure rate of DC microgrid in 30 years with different failure rate of inverters (Monte Carlo)

Figure 5.28: Experiment5: The impact of fi (Monte Carlo method).

Table 5.1: The method comparison of the 4 architectures for IoT system

	CTMC	**RBD**	**Monte Carlo**
Experiment 1(3 years)	0.004s;	0.078s;	8.338s;
Experiment 1(10 years)	0.006s;	0.082s;	19.585s;
Experiment 2(3 years)	0.002s;	0.065s;	6.326s;
Experiment 2(10 years)	0.004s;	0.056s;	17.576s;
Experiment 3(3 years)	0.002s;	0.061s;	6.238s;
Experiment 3(10 years)	0.005s;	0.059s;	16.355s;
Experiment 4(3 years)	0.002s;	0.061s;	6.138s;
Experiment 4(10 years)	0.004s;	0.059s;	17.585s;
Experiment 5(3 years)	0.002s;	0.063s;	6.338s;
Experiment 5(10 years)	0.005s;	0.054s;	17.556s;
Experiment 6(3 years)	0.002s;	0.061s;	6.325s;
Experiment 6(10 years)	0.004s;	0.064s;	17.284s;
Experiment 7(3 years)	0.002s;	0.061s;	6.326s;
Experiment 7(10 years)	0.004s;	0.059s;	17.282s;

Table 5.2: The method comparison of the 4 architectures for DC microgrid system

	CTMC	**RBD**	**Monte Carlo**
Experiment 1(3 years)	0.002s;	0.045s;	6.318s;
Experiment 1(10 years)	0.003s;	0.036s;	17.565s;
Experiment 2(3 years)	0.001s;	0.032s;	6.317s;
Experiment 2(10 years)	0.001s;	0.038s;	13.573s;
Experiment 3(3 years)	0.001s;	0.036s;	6.288s;
Experiment 3(10 years)	0.001s;	0.038s;	16.325s;
Experiment 4(3 years)	0.001s;	0.042s;	6.158s;
Experiment 4(10 years)	0.001s;	0.037s;	14.575s;
Experiment 5(3 years)	0.001s;	0.031s;	6.318s;
Experiment 5(10 years)	0.001s;	0.042s;	14.536s;
Experiment 6(3 years)	0.001s;	0.032s;	6.336s;
Experiment 6(10 years)	0.001s;	0.034s;	14.214s;
Experiment 7(3 years)	0.001s;	0.035s;	6.827s;
Experiment 7(10 years)	0.001s;	0.032s;	14.272s;

Chapter 6

The Case Study for Reliability Analysis

In my Mphil project, I also design case study system to simulate the failure process of the IoT and DC microgrid.

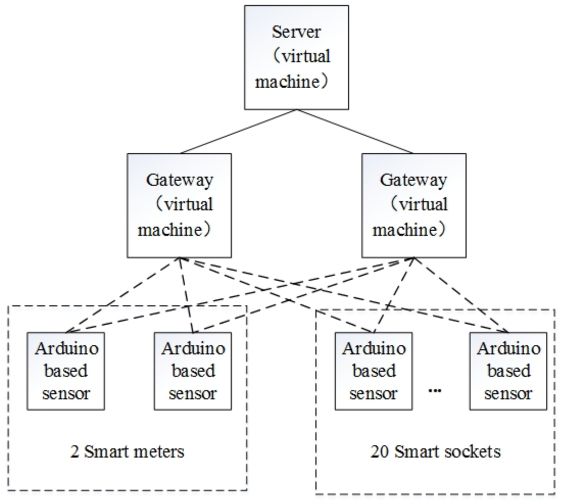
6.1 Case Study IoT System for Reliability Analysis

In this project, we also build the case study platform to test the reliability of the 4 Architectures of IoT system. The case study IoT system for photovoltaic (PV) system includes the following components:

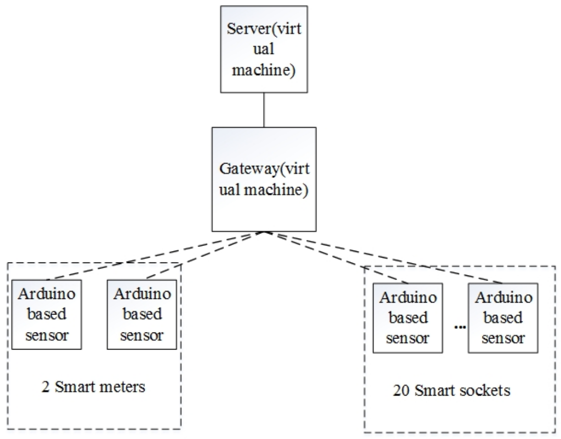
- Server: Get the data from 2 gateways.
- Gateway: get the data from 2 smart meters and 20 smart sockets.
- Sensors: the sensors are used to monitor the operation of PV energy system, we divided them into 2 groups: smart sockets and smart meters.

The schematic diagram of the platform for the IoT system is shown in Fig.6.1. The 4 architectures are similar with the previous architectures for IoT system. The figure for the case study IoT platform is shown in Fig.6.2. In this platform, another 18 smart sockets are not shown in this figure.

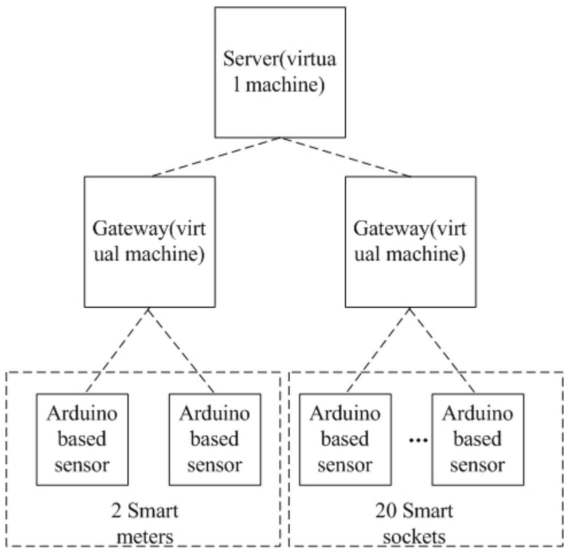
The server can monitor the operation of gateways, smart sockets, and smart meters. The monitoring results are shown in Fig.6.3. If one gateway fails, the monitoring results



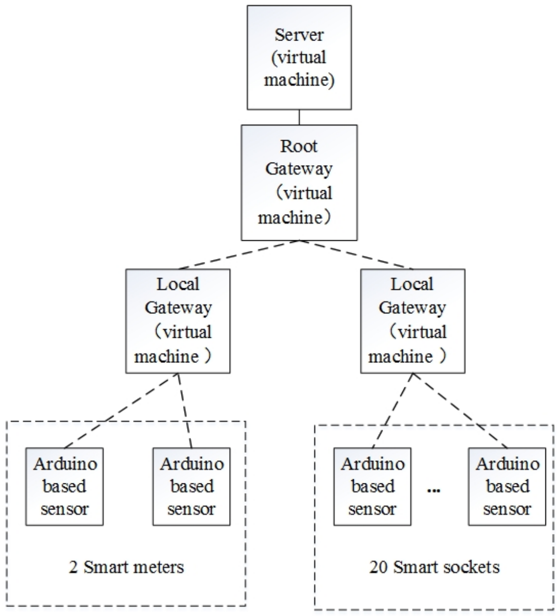
(a) Architecture I



(b) Architecture II



(c) Architecture III



(d) Architecture IV

Figure 6.1: The case study IoT system.

will be different,which is shown in Fig.6.4.

For the smart sockets and smart meters, the failure rate of them is realized by setting the sleep rate of the Arduino. In order to get more data to calculate the failure rate of the system to match the simulation results of CTMC models, we use 10 minutes to stand for 10 years. For example, the failure of smart sockets occurs in 10 years. In this experiment, the sleep rate of Arduino is set by using a random number. Get the random number from 1 to 120 every 5 seconds, if the random number equals to 1, the Arduino-based smart sockets entry sleep mode. For the failure rate of the gateway, the method is similar. For example, the failure of the gateway occurs in 10 years. The random number is also used to simulate the failure of the gateway. In the Python program of the gateway (virtual machine), get a random number from 1 to 120 every 5 seconds. If the random number equals to 1, the Python program exits, so the server can not get the data.

In order to check the state of smart sockets, smart meters, and gateways. Jmeter (a software) is used to send HTTP requests to the smart socks, smart meters, and gateways because each component is able to answer HTTP requests. The interface of Jmeter is shown in Fig.6.5.

The interface of HTTP requests are shown in Fig.6.6. In order to check the failure of the smart meters and smart sockets, the gateways(virtual machines)are also used to get the data of Arduino-based smart sockets and smart meters. The information is shown in Fig.6.7 and Fig.6.8.

I use JMeter to send HTTP requests to each component. According to the failure record of each component, we can judge when the system is down. After the system is down, this experiment finishes once. We conduct 300 experiments and calculate the failure rate on each year and plot it.

For the architecture comparison of the case study IoT system is also shown in Fig.6.9 and Table.6.1, the simulation results of the case study IoT system is similar with the CTMC models. We can also prove that our CTMC model is correct.

6.2 Case Study PV Energy System for Reliability Analysis

We also use sensors and virtual machine to simulate the failure process of the smart DC microgrid. The architecture of the case study PV system for simulating the failure process of the DC microgrid is shown in Fig.6.10.

The comparison of the results of the case study PV system and our CTMC models are shown in Fig.6.11 and Table6.2. The results show that the simulation results of the case study PV system is similar to CTMC models, so our CTMC models are correct.

6.3 Limitation of The Case Study System and Further Research

In this project, the reliability of domestic smart microgrid is evaluated by building case study systems. However, the scale of the case study IoT system and DC microgrid is small due to the limitation of time. In further research, the scale of the smart microgrid should be intended to make the research more meaningful. What's is more, in this project, the case study just simulate the failure process of 2 experiments. Due to the time limitation of the project, some other experiments are not conducted to match the results of CTMC models. These will be included in future research.

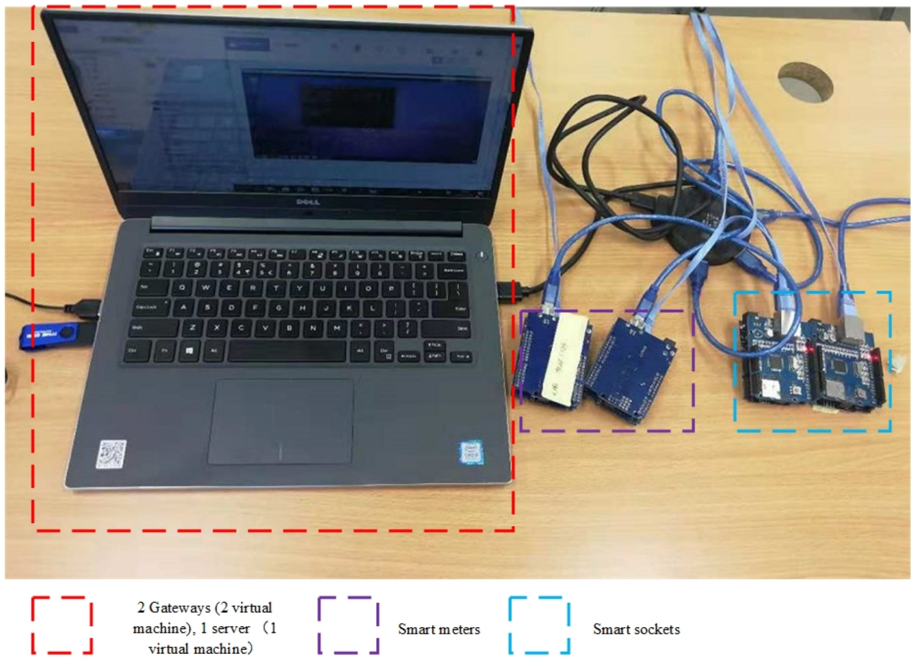
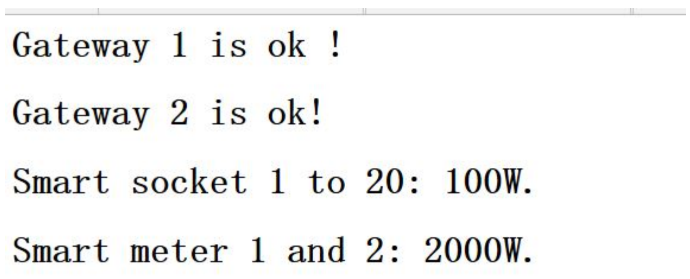


Figure 6.2: The case study IoT platform for Architecture I.

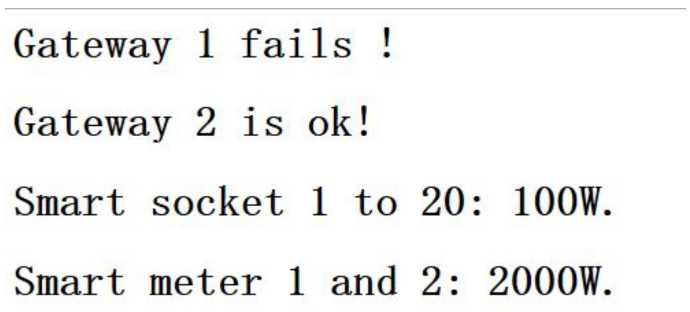
Table 6.1: Difference between case study system simulation and CTMC results (IoT system)

Time(years)	CTMC	Case Study system	CTMC-Case study system
1	0.1213	0.1045	0.0168
2	0.3283	0.3048	0.0235
3	0.5793	0.5964	0.0171
4	0.8152	0.8012	0.014
5	0.9291	0.9361	−0.0007
6	0.9663	0.9713	−0.0005
7	0.9929	1	0.0007
8	0.997	1	0.0003
9	0.9995	1	0.0005
10	0.9999	1	0.0001



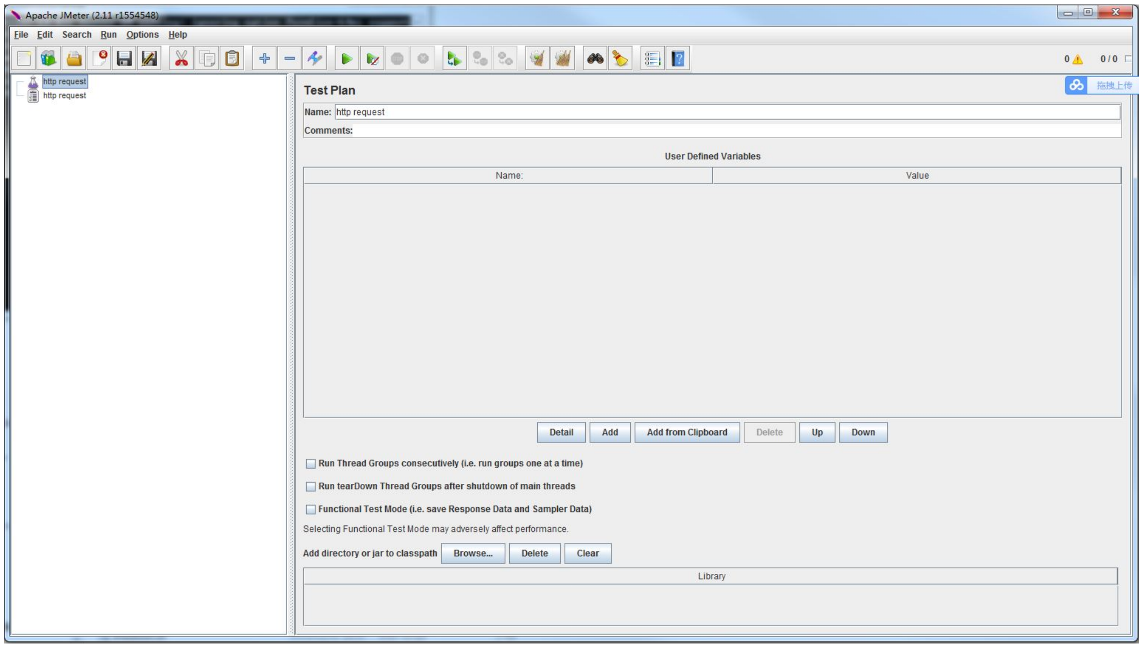
```
Gateway 1 is ok !

Gateway 2 is ok!

Smart socket 1 to 20: 100W.

Smart meter 1 and 2: 2000W.
```

Figure 6.3: The monitoring results on the server.



```
Gateway 1 fails !

Gateway 2 is ok!

Smart socket 1 to 20: 100W.

Smart meter 1 and 2: 2000W.
```

Figure 6.4: The monitoring results on the server (failure).



Apache JMeter (2.11 r1554548)

Test Plan

Name: http request

Comments:

User Defined Variables

Name:	Value

☐ Run Thread Groups consecutively (i.e. run groups one at a time)

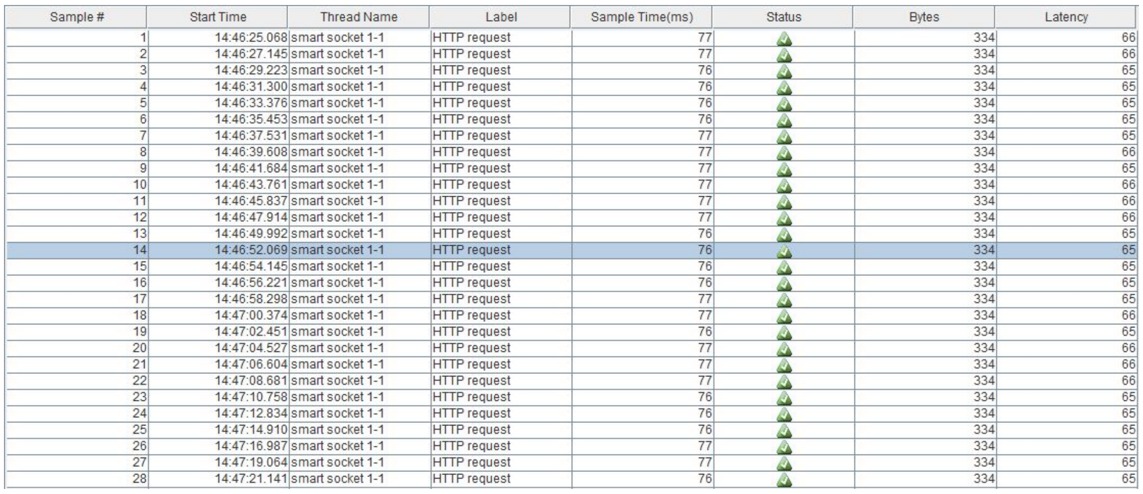
☐ Run tearDown Thread Groups after shutdown of main threads

☐ Functional Test Mode (i.e. save Response Data and Sampler Data)

Selecting Functional Test Mode may adversely affect performance.

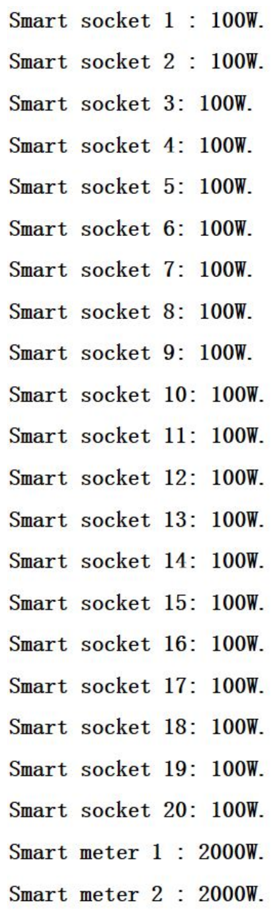
Add directory or jar to classpath

Figure 6.5: The interface of JMeter.



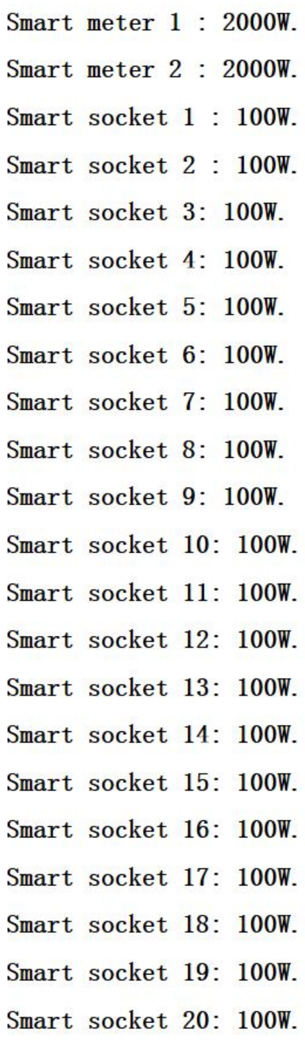
Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Latency
1	14:46:25.068	smart socket 1-1	HTTP request	77	✓	334	66
2	14:46:27.145	smart socket 1-1	HTTP request	77	✓	334	66
3	14:46:29.223	smart socket 1-1	HTTP request	76	✓	334	65
4	14:46:31.300	smart socket 1-1	HTTP request	76	✓	334	65
5	14:46:33.376	smart socket 1-1	HTTP request	76	✓	334	65
6	14:46:35.453	smart socket 1-1	HTTP request	76	✓	334	65
7	14:46:37.531	smart socket 1-1	HTTP request	77	✓	334	65
8	14:46:39.608	smart socket 1-1	HTTP request	77	✓	334	66
9	14:46:41.684	smart socket 1-1	HTTP request	77	✓	334	65
10	14:46:43.761	smart socket 1-1	HTTP request	77	✓	334	66
11	14:46:45.837	smart socket 1-1	HTTP request	77	✓	334	66
12	14:46:47.914	smart socket 1-1	HTTP request	77	✓	334	66
13	14:46:49.992	smart socket 1-1	HTTP request	76	✓	334	65
14	14:46:52.069	smart socket 1-1	HTTP request	76	✓	334	65
15	14:46:54.145	smart socket 1-1	HTTP request	76	✓	334	65
16	14:46:56.221	smart socket 1-1	HTTP request	76	✓	334	65
17	14:46:58.298	smart socket 1-1	HTTP request	77	✓	334	65
18	14:47:00.374	smart socket 1-1	HTTP request	77	✓	334	66
19	14:47:02.451	smart socket 1-1	HTTP request	76	✓	334	65
20	14:47:04.527	smart socket 1-1	HTTP request	77	✓	334	66
21	14:47:06.604	smart socket 1-1	HTTP request	77	✓	334	66
22	14:47:08.681	smart socket 1-1	HTTP request	77	✓	334	66
23	14:47:10.758	smart socket 1-1	HTTP request	76	✓	334	65
24	14:47:12.834	smart socket 1-1	HTTP request	76	✓	334	65
25	14:47:14.910	smart socket 1-1	HTTP request	76	✓	334	65
26	14:47:16.987	smart socket 1-1	HTTP request	77	✓	334	65
27	14:47:19.064	smart socket 1-1	HTTP request	77	✓	334	65
28	14:47:21.141	smart socket 1-1	HTTP request	76	✓	334	65

Figure 6.6: The HTTP requests.



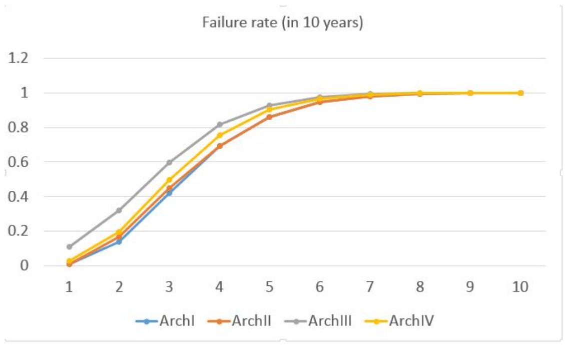
```
Smart socket 1 : 100W.
Smart socket 2 : 100W.
Smart socket 3: 100W.
Smart socket 4: 100W.
Smart socket 5: 100W.
Smart socket 6: 100W.
Smart socket 7: 100W.
Smart socket 8: 100W.
Smart socket 9: 100W.
Smart socket 10: 100W.
Smart socket 11: 100W.
Smart socket 12: 100W.
Smart socket 13: 100W.
Smart socket 14: 100W.
Smart socket 15: 100W.
Smart socket 16: 100W.
Smart socket 17: 100W.
Smart socket 18: 100W.
Smart socket 19: 100W.
Smart socket 20: 100W.
Smart meter 1 : 2000W.
Smart meter 2 : 2000W.
```

Figure 6.7: Monitoring information of smart sockets and smart meters (gateway 1).

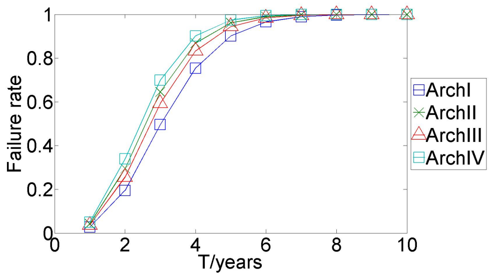


```
Smart meter 1 : 2000W.
Smart meter 2 : 2000W.
Smart socket 1 : 100W.
Smart socket 2 : 100W.
Smart socket 3: 100W.
Smart socket 4: 100W.
Smart socket 5: 100W.
Smart socket 6: 100W.
Smart socket 7: 100W.
Smart socket 8: 100W.
Smart socket 9: 100W.
Smart socket 10: 100W.
Smart socket 11: 100W.
Smart socket 12: 100W.
Smart socket 13: 100W.
Smart socket 14: 100W.
Smart socket 15: 100W.
Smart socket 16: 100W.
Smart socket 17: 100W.
Smart socket 18: 100W.
Smart socket 19: 100W.
Smart socket 20: 100W.
```

Figure 6.8: Monitoring information of smart sockets and smart meters (gateway 2).



(a) Case study IoT system



(b) CTMC model (IoT system)

Figure 6.9: The comparison between simulation results of case study IoT system and CTMC models (IoT system).

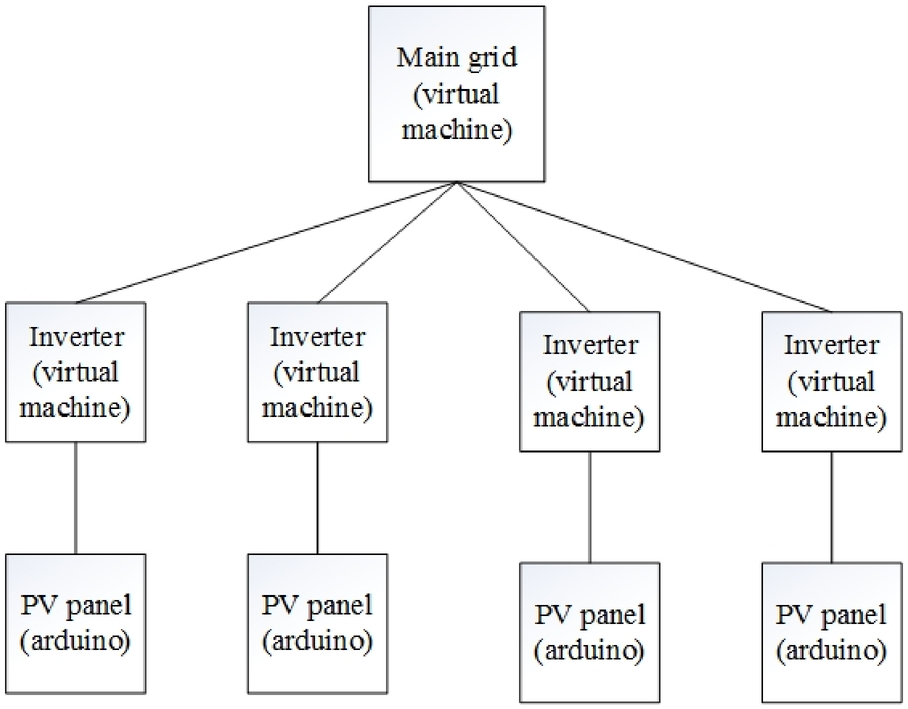
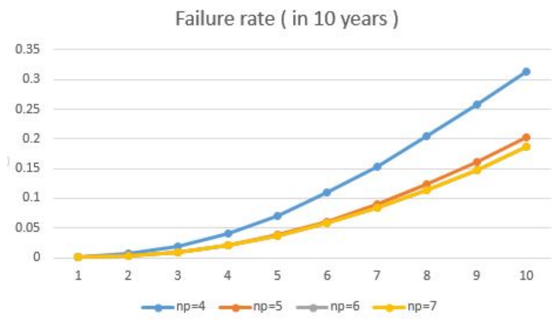
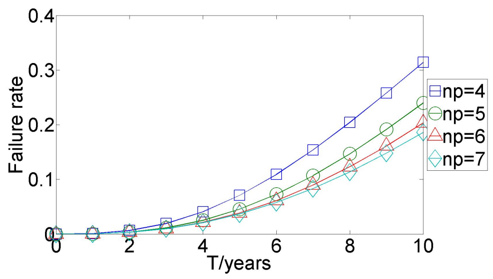

Main grid (virtual machine)
Inverter (virtual machine)
Inverter (virtual machine)
Inverter (virtual machine)
Inverter (virtual machine)
PV panel (arduino)
PV panel (arduino)
PV panel (arduino)
PV panel (arduino)

Figure 6.10: The simulated DC microgrid system.



(a) Case study PV system



(b) CTMC model(DC microgrid)

Figure 6.11: The comparison between simulation results of case study system and CTMC models (DC microgrid).

Table 6.2: Difference between case study system simulation and CTMC results (DC microgrid)

Time(years)	**CTMC**	**Case study system**	**CTMC-Case study system**
1	0.0001	0	0.0001
2	0.0064	0	0.0064
3	0.0193	0.0298	−0.0105
4	0.0408	0.0304	0.0104
5	0.0709	0.0609	0.0100
6	0.1091	0.0853	0.0238
7	0.1540	0.1433	0.0107
8	0.2041	0.2146	−0.0105
9	0.2580	0.2686	−0.0106
10	0.3140	0.3350	−0.0210

Chapter 7

Conclusion and Future Work

7.1 Conclusion

In this Mphil project, we analyze and propose an optimized architecture for the smart microgrid. The CTMC models are built in PRISM to analyze the reliability of it. The experiments are divided into 2 parts: reliability of IoT system and DC microgrid. The optimized architecture of IoT system and DC microgrid are analyzed respectively to show our architecture is better than the other typical architectures. Then the influence of different components of the IoT system and DC microgrid are analyzed.

In order to show our CMTC models are correct, RBD method and Monte Carlo method are used to as benchmarks. The RBD models and Monte Carlo models are built in Matlab to analyze the reliability of the smart microgrid. The experiment results are similar to the CMTC models, so our CTMC models are correct.

What's more, the case study IoT platform is built in our lab to verify the reliability of different architectures. The experiment results also show that our architecture is the most reliable one.

7.2 Future Work

In this project, I focus on the reliability analysis of the smart microgrid. For the model checking part, in the real case, some of the components are reparable. However, for the CTMC models in this project, the reparable nodes are not considered. The architectures should be extended to meet the demand of smart grid. The reparable nodes bring more states, the CTMC models and Matlab models still should be improved. What's more, for IoT system and DC microgrid, there are also some other failure types, these will also be included in further research.

Bibliography

[1] Martn Martin Abadi, Cdric Fournet, Andrew D Gordon, Yasmina Abdedda?m, Eugene Asarin, Oded Maler, Sanjeev Arora, Adnan Aziz, K Sanwal, and Vigyan Singhal. Probabilistic model checking. In *IEEE Workshop on Design and Diagnostics of Electronic Circuits and Systems*, pages 1 – 6, 2011.

[2] K. Agbossou, M. Kolhe, J. Hamelin, and T. K. Bose. Performance of a stand-alone renewable energy system based on energy storage as hydrogen. *IEEE Transactions on Energy Conversion*, 19(3):633–640, 2004.

[3] Mohammadreza Aghaei, Francesco Grimaccia, Carlo A. Gonano, and Sonia Leva. Innovative automated control system for pv fields inspection and remote control. *IEEE Transactions on Industrial Electronics*, 62(11):7287–7296, 2015.

[4] E. G. Amparore and S. Donatelli. Efficient model checking of the stochastic logic csl ta. *Performance Evaluation*, 2018.

[5] E. Barklund, Nagaraju Pogaku, Milan Prodanovic, C. Hernandez-Aramburo, and Tim C. Green. Energy management in autonomous microgrid using stability-constrained droop control of inverters. *IEEE Transactions on Power Electronics*, 23(5):2346–2352, 2008.

[6] Paolo Bellagente, Paolo Ferrari, Alessandra Flammini, and Stefano Rinaldi. Adopting iot framework for energy management of smart building: A real test-case. In *IEEE International Forum on Research and Technologies for Society and Industry Leveraging A Better Tomorrow*, pages 138–143, 2015.

[7] Nicole B?Uerle and Ulrich Rieder. Markov decision processes. *Jahresbericht der deutschen Mathematiker-Vereinigung*, 112(4):217–243, 2010.

[8] M. Buresch. *Photovoltaic energy systems: Design and installation.* McGraw-Hill, 1983.

[9] Juan A. Carrasco. Two methods for computing bounds for the distribution of cumulative reward for large markov models. *Performance Evaluation*, 63(12):1165–1195, 2006.

[10] Zakariya M. Dalala, Zaka Ullah Zahid, Wensong Yu, Younghoon Cho, and Jih Sheng Lai. Design and analysis of an mppt technique for small-scale wind energy conversion systems. *IEEE Transactions on Energy Conversion*, 28(3):756–767, 2013.

[11] Antnio Damaso, Nelson Rosa, and Paulo Maciel. Reliability of wireless sensor networks. *Sensors*, 14(9):15760–15785, 2014.

[12] Frits Dannenberg, Marta Kwiatkowska, Chris Thachuk, and Andrew J. Turberfield. Dna walker circuits: Computational potential, design, and verification. In *International Workshop on DNA-Based Computers*, pages 31–45, 2013.

[13] A. L. Dimeas and N. D. Hatziargyriou. Operation of a multiagent system for microgrid control. *IEEE Transactions on Power Systems*, 20(3):1447–1455, 2005.

[14] Zhixin Fu, Zhen Liu, Yue Yuan, Min Zhao, and Qiaomu Li. Coverage restoration method for wireless sensor networks of distributed pv system. *International Journal of Distributed Sensor Networks*, 2015:1–7, 2015.

[15] Ji Zhen Guan, Yi Sheng Wang, Pu Lan, L. I. Wei, Zhi Jiang Wang, P. U. Yong, Dong Wen Liu, L. I. Jian-Feng, and Liang Qiao. New solar led traffic signs for minibus lanes. *Advanced Display*, 21(10):20–23, 2010.

[16] Yuanxiong Guo, Miao Pan, and Yuguang Fang. *Optimal Power Management of Residential Customers in the Smart Grid.* IEEE Press, 2012.

[17] Lajos Hanzo. *Smart Grid: Fundamentals of Design and Analysis.* Wiley, 2012.

[18] John Heatha, Marta Kwiatkowskab, and Gethin Normanb. Tymchyshyn: Probabilistic model checking of complex biological pathways. *Theoretical Computer Science*, 391(3):239–257, 2008.

[19] Tao Hu, Minghui Zheng, Jianjun Tan, Li Zhu, and Wang Miao. Intelligent photovoltaic monitoring based on solar irradiance big data and wireless sensor networks. *Ad Hoc Networks*, 35(C):127–136, 2015.

[20] Yan Liang Jin, Hao Jie Lin, Zhu Ming Zhang, and Zhen Zhang. Estimating the reliability and lifetime of wireless sensor network. In *International Conference on Wireless Communications, NETWORKING and Mobile Computing*, pages 1–4, 2008.

[21] Hristiyan Kanchev, Di Lu, Frederic Colas, Vladimir Lazarov, and Bruno Francois. Energy management and operational planning of a microgrid with a pv-based active generator for smart grid applications. *IEEE Transactions on Industrial Electronics*, 58(10):4583–4592, 2011.

[22] V. Katsioulis, E. Karapidakis, M. Hadjinicolaou, and A. Tsikalakis. *Wireless Monitoring and Remote Control of PV Systems Based on the ZigBee Protocol.* Springer Berlin Heidelberg, 2011.

[23] K. Kawabe and K. Tanaka. Impact of dynamic behavior of photovoltaic power generation systems on short-term voltage stability. *IEEE Transactions on Power Systems*, 30(6):3416–3424, 2015.

[24] Kenichi Kawabe, Yutaka Ota, Akihiko Yokoyama, and Kazuyuki Tanaka. Novel dynamic voltage support capability of photovoltaic systems for improvement of short-term voltage stability in power systems. *IEEE Transactions on Power Systems*, PP(99):1–1, 2016.

[25] Marta Kwiatkowska, Gethin Norman, and David Parker. Probabilistic symbolic model checking with prism: a hybrid approach. *International Journal on Software Tools for Technology Transfer*, 6(2):128–142, 2004.

[26] Marta Kwiatkowska, Gethin Norman, and Jeremy Sproston. Probabilistic model checking of the ieee 802.11 wireless local area network protocol. In *Joint International Workshop on Process Algebra and Probabilistic Methods, PERFORMANCE Modeling and Verification*, pages 169–187, 2002.

[27] Hermann Laukamp, Georg Bopp, Robin Grab, Christof Wittwer, Heinrich H?berlin, Bettina Van Heeckeren, Severin Phillip, Florian Reil, Heribert Schmidt, and Annett

Sepanski. Pv fire hazard - analysis and assessment of fire incidents. *Canadian Journal of Fisheries and Aquatic Sciences*, 63(7):1515–1525, 2013.

[28] Howon Lee and Dong Ho Cho. Capacity improvement and analysis of voip service in a cognitive radio system. *IEEE Transactions on Vehicular Technology*, 59(4):1646–1651, 2010.

[29] Lulu Liang, Kai Zheng, Zilong Wei, Yanmei Wang, Sihan Wu, and Xin Huang. Model checking of iot system in microgrid. In *International Conference on Information Technology in Medicine and Education*, pages 601–605, 2017.

[30] Cheng Min Lin, Hui Kang Teng, Cheng Chih Yang, and Hwei Li Weng. A mesh network reliability analysis using reliability block diagram. In *IEEE International Conference on Industrial Informatics*, pages 975–979, 2010.

[31] Fangrui Liu, Shanxu Duan, Fei Liu, Bangyin Liu, and Yong Kang. A variable step size inc mppt method for pv systems. *IEEE Transactions on Industrial Electronics*, 55(7):2622–2628, 2008.

[32] Xiaosen Liu and Edgar Snchez-Sinencio. A highly efficient ultralow photovoltaic power harvesting system with mppt for internet of things smart nodes. *IEEE Transactions on Very Large Scale Integration Systems*, 23(12):3065–3075, 2015.

[33] Ali Mahani, Mohammad Saeed Ansari, and Yousef S. Kavian. Reliability or performance: A tradeoff in wireless sensor networks. In *International Symposium on Communication Systems, Networks and Digital Signal Processing*, pages 1–5, 2012.

[34] Emilio Mamarelis, Giovanni Petrone, and Giovanni Spagnuolo. Design of a sliding-mode-controlled sepic for pv mppt applications. *IEEE Transactions on Industrial Electronics*, 61(7):3387–3398, 2014.

[35] Qiang Mei, Mingwei Shan, Liying Liu, and Josep M. Guerrero. A novel improved variable step-size incremental-resistance mppt method for pv systems. *IEEE Transactions on Industrial Electronics*, 58(6):2427–2434, 2011.

[36] Lexuan Meng, Adriana Luna, Enrique Rodrguez Daz, Bo Sun, Tomislav Dragicevic, Mehdi Savaghebi, Juan C. Vasquez, Josep M. Guerrero, Moiss Graells, and Fabio Andrade. Flexible system integration and advanced hierarchical control architectures

in the microgrid research laboratory of aalborg university. *IEEE Transactions on Industry Applications*, 52(2):1736–1749, 2016.

[37] A. Yu. Mitrophanov. Stability and exponential convergence of continuous-time markov chains. *Journal of Applied Probability*, 40(4):970–979, 2003.

[38] Farzam Nejabatkhah, Saeed Danyali, Seyed Hossein Hosseini, and Mehran Sabahi. Modeling and control of a new three-input dccdc boost converter for hybrid pv/fc/-battery power system. *IEEE Transactions on Power Electronics*, 27(5):2309–2324, 2012.

[39] Gethin Norman and Vitaly Shmatikov. *Analysis of Probabilistic Contract Signing*. Springer Berlin Heidelberg, 2003.

[40] Miguel J. Prieto, Alberto M. Perna, Fernando Nu?o, Juan Daz, and Pedro J. Villegas. Development of a wireless sensor network for individual monitoring of panels in a photovoltaic plant. *Sensors*, 14(2):2379–2396, 2014.

[41] Meixun Qu. *Probabilistic model checking on reliability analysis of mobile edge networks*. PhD thesis, Xi'an Jiaotong-Liverpool University, 2018.

[42] Angle Reinders, Pierre Verlinden, Wilfried Van Sark, and Alexandre Freundlich. *11.4. PV System Monitoring and Characterization*. John Wiley Sons, Ltd, 2017.

[43] Ivanovitch Silva, Luiz Affonso Guedes, Paulo Portugal, and Francisco Vasques. Reliability and availability evaluation of wireless sensor networks for industrial applications. *Sensors*, 12(1):806–38, 2012.

[44] R Teodorescu, M Liserre, and P Rodr?Guez. *Grid Converters for Photovoltaic and Wind Power Systems*. Wiley, 2011.

[45] Remus Teodorescu, Marco Liserre, and Pedro Rodrguez. *Control of Grid Converters under Grid Faults*. John Wiley Sons, Ltd, 2010.

[46] Giuseppe Marco Tina, Fabio Cosentino, and Cristina Ventura. *Monitoring and Diagnostics of Photovoltaic Power Plants*. Springer International Publishing, 2016.

[47] Sihan Wu. *Probabilistic Model Checking of Software Defined Distributed Energy System (SDDES) Using PRISM*. PhD thesis, Xi'an Jiaotong-Liverpool University, 2017.

[48] Sihan Wu, Kai Zheng, and Xin Huang. Model checking pv energy system with remote reprogramming function. In *International Conference on Information Technology in Medicine and Education*, pages 606–610, 2016.

[49] Ruifeng Yan and Tapan Kumar Saha. Investigation of voltage stability for residential customers due to high photovoltaic penetrations. *IEEE Transactions on Power Systems*, 27(2):651–662, 2012.

[50] Jianfeng Yang, Jianfeng Yang, Ming Zhao, Qi Li, and Yan Liu. Wireless sensor network reliability modelling based on masked data. *International Journal of Sensor Networks*, 17(4):217–223, 2015.

[51] Mohamed Zahran, Abdullah Alhosseen, Abdullah Alhosseen, and Ihab El-Sayed. Wired and wireless remote control of pv system. *Wseas Transactions on Systems and Control*, 5(8):656–666, 2010.

[52] He Zhang, Arnaud Davigny, Jonathan Sprooten, Benoit Robyns, Frederic Colas, and Yvan Poste. *Energy Management Strategy for Commercial Buildings Integrating PV and Storage Systems.* Springer Berlin Heidelberg, 2012.

[53] Xuan Zhou, Rongpeng Li, Tao Chen, and Honggang Zhang. Network slicing as a service: enabling enterprises' own software-defined cellular networks. *IEEE Communications Magazine*, 54(7):146–153, 2016.