Simulating Network Adapted Market Controller in a Smart Grid Scenario

Networks and Distributed Systems
Group 1020

Aalborg University, 4. semester spring 2013

School of Information and Communication Technology
Networks and Distributed Systems
Frederik Bajers Vej 7
Telefon +45 99 40 86 00
http://www.sict.aau.dk
E-mail webinfo@es.aau.dk

AALBORG UNIVERSITY
DENMARK

Title:

Simulating Network Adapted Market Controller in a Smart Grid Scenario

Theme:

Master's Thesis

Project Period:
P10, Spring semester 2013

Project group:
13gr1020

Group members:
Jacob Theilgaard Madsen
Thomas le Fevre Kristensen

Supervisor:
Rasmus Løvenstein Olsen
Tatiana Kozlova Madsen
Kartheepan Balachandran

Number of copies: 5

Number of pages: 49

Appended documents: 1 appendix

Finished: 06-06-13

Abstract:

The emergence of smart grids requires a systematic approach to verifying certain scenarios. We wish to make a framework where the effects of an imperfect network can be seen on the performance of a control algorithm. We will then attempt to improve the performance by taking network conditions into account.
In this report we define a scenario for the smart grid, wherein a general architecture for the communication, control, and power distribution is defined. We then describe the control algorithm that will be used in the defined scenario. This control algorithm is a market controller that attempts to regulate the power consumption during peak hours, by storing thermal energy in refrigerators when there is low power demand and using this stored energy when there is a high power demand. A performance metric have been defined which shows the impact of an imperfect network in the simulations.
In order to improve performance of the control adaption two methods have been considered, one where the control parameter value is scaled according to the packet loss, and one where the control message is transmitted several times, based on the probability of information mismatch. The first method shows that performance can be increased compared to not using adaptation of the control algorithm, and the second method shows that although there is improvements in transmitting the control message several times it does not make sense to use mismatch probability in this scenario. The first method is very specific to the defined scenario, while the second method can be generalized.

School of Information and Communication Technology
Netværk og Distribuerede Systemer
Frederik Bajers Vej 7
Telefon +45 99 40 86 00
http://www.sict.aau.dk
E-mail webinfo@es.aau.dk

AALBORG UNIVERSITY
DENMARK

Titel:

Simulering af Netværkstilpasset Markedskontrolenhed i et Smart Grid Scenarie

Tema:

Masterafhandling

Projekt Periode:
P10, forårs semester 2013

Projekt gruppe:
13gr1020

Gruppe medlemmer:
Jacob Theilgaard Madsen
Thomas le Fevre Kristensen

Vejledere:
Rasmus Løvenstein Olsen
Tatiana Kozlova Madsen
Kartheepan Balachandran

Antal kopier: 5

Antal sider: 49

Vedlagte dokumenter: 1 appendiks

Afleveret: 06-06-13

Resume:

Fremkomsten af smart grid kræver en systematisk tilgang til at verificere visse scenarier. Vi ønsker at lave en simulerings ramme hvor effekten af et uperfekt netværk kan ses påydelsen af en kontrol algoritme. Vi vil forsøge at forbedre ydelsen ved at tage netværks forhold i betragtning.
I denne rapport definerer vi et scenarie for smart grid hvori en general arkitektur for kommunikationen, kontrollen og strøm distribueringen er defineret. Derefter beskriver vi kontrol algoritmen som vil blive brugt i det definerede scenarie. Denne kontrol algoritme er en markeds kontroller som forsøger at regulere strøm forbruget i timer hvor det er højt, ved at opbevare termisk energi i køleskabe når der er lav efterspørgsel påstrøm, og bruge denne opbevarede energi når der er høj efterspørgsmål påstrøm. En ydelsesevne kvantificering er blevet defineret, og denne viser indflydelsen a et ikke perfekt netværk.
For at forbedre ydelsen af kontrol algoritmen to metoder er blevet undersøgt, en hvor kontrol parameteren er skaleret i forhold til pakketabet, og en hvor kontrol beskeden er sendt flere gange. Den første metode viser at ydelsen kan forbedres sammenlignet med ikke at adaptere kontrol algoritmen. Den anden metode viser at der er forbedring af kontrollen ved at transmitere kontrol beskeden flere gange, men brugen af mismatch probability giver ikke mening i dette scenarie. Den første metode er meget specific for det valgte scenarie, mens den anden metode er mere generel.

Preface

This report is made by a group of 4th semester students of the Network and Distributed Systems masters programme at Aalborg University.
This report consists of nine chapters. The first three chapters will describe the problem, and be used to find a specific scenario to evaluate. Chapters 4 and 5 describes the implementations done in order to make the simulations, and chapter 6 and 7 discuss the simulations and the results of them. The Conclusion and Assessment is used to conclude on the entire project, as well as discussing what was achieved, and what should be done in the future.
Throughout the report external references will be displayed as numbers, an example of this is: [1]. If the report is read digitally these references will be interactive, and they can be used to jump directly to the reference in the reference list found on page 49.
This report is a continuation of the work conducted in [1], a resume of that work is therefore created and can be found in appendix A on page 52.
The project group would like to extend a special thanks to Luminita Cristiana Totu from Automation and Control on Aalborg University, for supplying insight into control algorithms used within smart grids, and for supplying simulations of these algorithms. A special thanks is also extended to the smart grid projects SmartC2Net and EDGE, for discussions on the subject.
This report contains a list of acronyms, a list of figures and a list of tables. The list of acronyms can be found on page 3, where acronyms used in the report are defined. These glossaries will be interactive in the same manner as the references. The list of figures can be found on page 5, and the list of tables can be found on page 6. The picture on the front page was found in [2].

If it is wished to access the simulation data, the developed software, or a digital copy of the report, it can be found on the group homepage:`http://kom.aau.dk/group/13gr1020/`.

Jacob Theilgaard Madsen

Thomas le Fevre Kristensen

Table of contents

Acronyms

ADSL Asymmetric Digital Subscriber Line. 9, 10

AN Access Network. 9–11

BER Bit Error Rate. 18, 23, 36, 37, 40

DL DownLoad. 9, 10

DN Distribution Network. 9–11, 37

EDGE Efficient Distribution of Green Energy. 4

GPRS General Packet Radio Service. 9, 10

IP Internet Protocol. 16, 20, 44

mmPr Mismatch Probability. 4, 5, 21, 32–34, 36, 37, 40–44

NB Narrow Band. 9, 10

PLC Power Line Communication. 9, 10, 15, 22, 36, 37, 42–44

QoS Quality of Service. 4, 9, 10, 15–17, 32, 42, 43

SmartC2Net Smart Control of Energy Distribution Grids over Heterogeneous Communication Networks. 4

SNR Signal to Noise Ratio. 43

TCP Transmission Control Protocol. 16, 17, 20–22, 26, 28, 30, 31, 34, 42–44, 47

UDP User Datagram Protocol. 16, 17, 19, 20, 22, 28, 30–32, 34, 36, 42, 44

UL UpLoad. 9, 10

WB Wide Band. 9, 10

List of Figures

List of Tables

1 Introduction

The smart grid is the idea to make the power grid more intelligent, and making it adaptable to the current production and consumption of power. This will require that the energy companies knows what the consumers consume, and the controllers must have the ability to change the consumption to some degree. Development of smart grid faces three main challenges: Controlling the power grid(voltage and frequency control), balancing power consumption with power production, and enabling communication to all entities in the system.

Meeting these challenges requires engineers from energy, control and communication to work together. As these disciplines are not independent in the smart grid, development of one part must incorporate the other two parts. This means that an analysis or implementation of the communication network must also take into consideration the controller of the smart grid, and the structure of the power grid. This project will study the communication networks and how to make the smart grid adaptive to the state of the communication network.

To determine the network conditions, this project proposes to develop a network monitor. This network monitor will measure different Quality of Service (QoS) parameters, and adapt the smart grid accordingly. If the smart grid is experiencing poor QoS from the network, it can be changed using two basic strategies. The first strategy is change the network to improve the QoS. This can be done by rerouting some packets to lessen the load on key points in the network. It can also be achieved by changing routing priorities so packets from slow parts will be routed first. If the network used in this strategy is rented from an other company(eg. 3G), the smart grid has no control over the routing, and this strategy will require a complete redundant network to switch to, if poor QoS is experienced.

The other strategy is to reduce the demand of the network. This can be achieved by changing the time interval of the controller, so information will have to be distributed less often. Lower demand can also be achieved by transmitting less information, this will, however, mean that the information not transmitted will have to be assumed by the receiver.

There are two groups working with the development of the smart grid which will be considered in this report. The first is the Efficient Distribution of Green Energy (EDGE) group, which considers the development of a simulation framework for the smart grid, and the other is the Smart Control of Energy Distribution Grids over Heterogeneous Communication Networks (SmartC2Net) group, which is more concerned with emulation and implementations of the smart grid. This project will be a work in collaboration with these two groups, and inspiration regarding use cases and implementation strategies will be collected from them. EDGE have given the main inspiration in regards to the framework and the control algorithm, while SmartC2Net have given inspiration regarding Mismatch Probability (mmPr) and the scenario identified in this report.

A framework for simulating a communication network together with a controller is developed in [1]. This framework was developed so the controller can be simulated with a communication network, but still separating the control discipline from the network discipline as much as possible. This is done by letting the network be simulated in OMNeT++, and the control algorithm run in MATLAB. A clear interface is then defined between the two simulators.

This project explores the possibilities of monitoring the communication networks in smart grids, and adapt the smart grid controller according to the monitored network conditions. The framework introduced in [1] will be extended to include a network monitor, and the smart grid controller will be simulated in this framework together with the network. The power grid dynamics will be omitted in this project, as these are not included in the framework, and is outside the scope of this project to develop. This leads to three key questions that must be answered:

Question 1: How should the communication network of the smart grid be defined?

Question 2: How should a network monitor be designed?

Question 3: How should the smart grid control be adapted based on the network conditions?

1.1 Problem formulation

In this project it is desired to modify a control algorithm to better match the conditions of the communication network. At first it is attempted to modify the control signal by a factor found from the packet loss experienced in the network. This method is very specific to the used control algorithm, and is unlikely to be useful for other control algorithms. The second approach is more generic, and uses the probability of information mismatch between the controller and its controlled units to modify the controller. We call this probability mmPr, and it will be explained in detail in chapter 7. The controller modification will in this case be done by sending the control signal several times in the same control loop, to counteract high packet loss probabilities. These modifications are performed to increase the performance of the control algorithm. This leads to the main question answered in this project:

Is it possible to modify a given control algorithm to the packet loss or mmPr of the communication network, to increase performance of the controller?

2 | Smart Grid

This chapter will give an overview of the architecture of the smart grid, which will be referred to in the rest of the report. We will answer part of question 1 from the introduction: *How should the communication network of the smart grid be defined?*, by answering the following sub questions:

Question 1a: What is the network architecture of the smart grid?

Question 1b: Which entities are there in the smart grid?

The smart grid will be implemented as an upgrade to the power grid, which means that many of the elements in the power grid will be reused, including the power lines and the transformer stations. Because of this the smart grid will have an over all architecture that seems similar to that of the current power grid. The large change is the addition of controllers that can optimize the power consumption, and a communication network that can transfer the needed information for these controllers to be effective.

Figure 2.1 on the next page gives a overview of the power grid. The high voltage layer in Denmark is 132 kV to 400 kV[4], and is where the large electricity suppliers are placed. It is used for transmitting the power to the entire country, and is called the transmission layer. The medium voltage layer is operating a voltages between 10 kV and 60 kV[4]. Its primary goal is to distribute the power. Some of it is going to consumers on the medium voltage level, but a large part of it is going to the low voltage layer. The main part of the power production in Denmark is added on the medium voltage layer, for example from wind farms. The low voltage layer is 400 V[4], and is the final leg of the power grid to households. It, together with the medium voltage layer, is referred to as the distribution grid, as that is their primary function. The low voltage grid is where most consumers are placed, both households and retail.

2.1 Entities in smart grid

When considering entities in smart grid, there is a distinction between nodes at the medium voltage layer and nodes on the low voltage layer. This can then be separated into consumers and producers, which again can be separated into flexible and inflexible consumers. This section will list the different entities on the voltage layers, and go into a more detailed description regarding communication and flexibility. Consumers will be marked either flexible or inflexible, depending on whether or not it is possible to regulate their consumption of power.

2.1.1 Medium Voltage entities

The entities operating on the medium voltage level are generally larger producers and consumers, requiring and generating more power than on the entities on the low voltage layer. They are, however, also fewer, and with greater physical distance between them. The consumers and producers on the medium voltage layer all need to communicate with the power level controllers and the market controllers, as can be seen on figure 2.2 on page 14. It can further be seen that they only communicate with the primary controller and the market controller, not the secondary controller nor the market aggregators. Any information they have which should be delivered to the secondary controllers or aggregators must be send from either the primary controller or market controller.

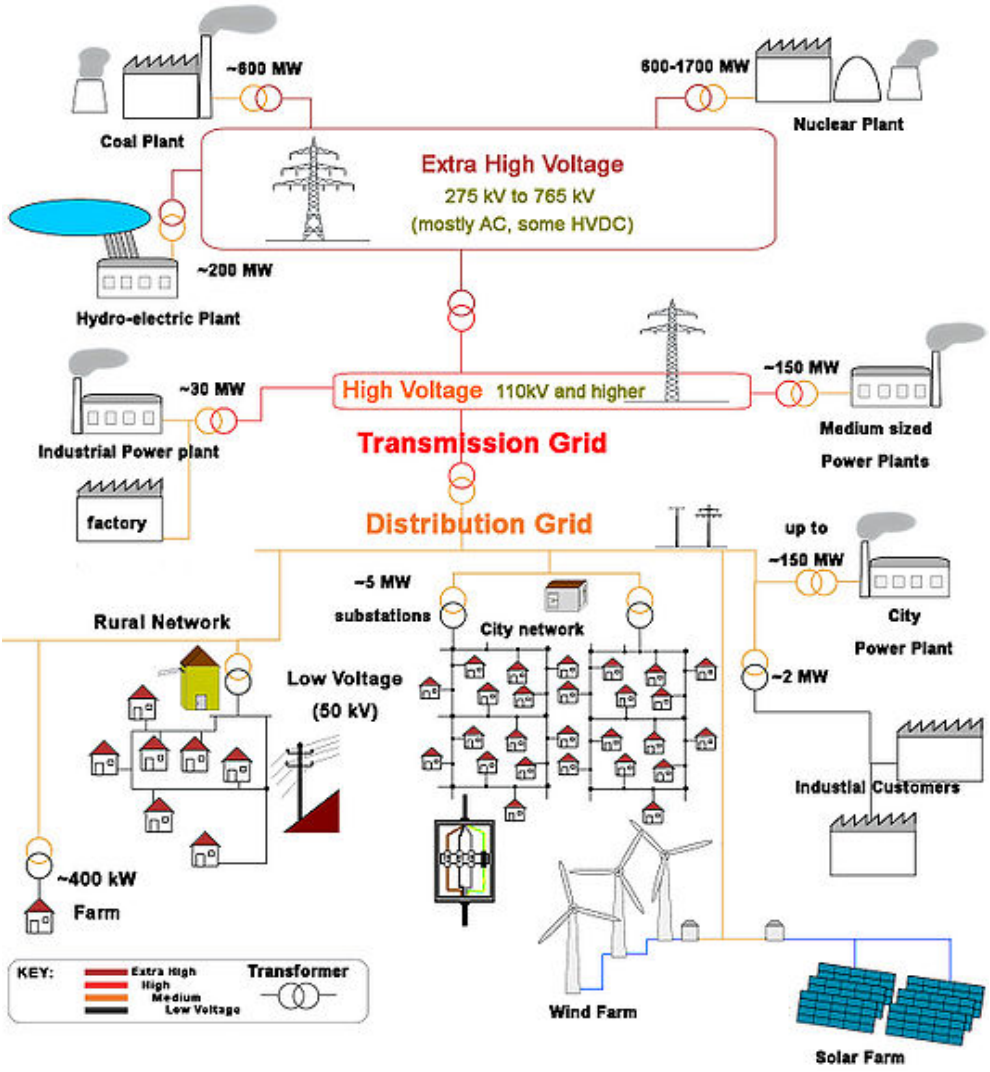

Figure 2.1: Overview of entities in the power grid [3].

Consumers:

Industrial consumers:
: These include shopping centers, and some smaller factories. Factories are inflexible consumers, as their production units must have power at all times. They may have some minor flexible units in the factory, but the major part of the power consumption is inflexible. Large shopping centers are flexible loads, as a large part of their power consumption comes from temperature controlling units, which can be regulated to a certain degree.

Producers:

Wind farms:
: Wind farms power production can vary greatly depending on the weather. Because their power production can change over time, it is necessary to have some back up in case they do not produce enough power.

Solar farms:
: While solar farms also depend on the weather, the power production can vary just as for wind farms.

City power plants:
: City power plants can generate a steady amount of power, and does additionally have the ability to increase or decrease the power output.

2.1.2 Low Voltage entities

While the entities on the low voltage layer consume less power than the entities on the medium voltage layer, they are also more numerous. The production at the low voltage layer are at the moment coming from personal power production units, and are relatively rare for households. They are, however, on the rise, and will constitute a larger part of the energy production in the future. This can cause issues with the current implementation of the power grid, as it does not support the transfer of energy from low to medium voltage level. So if the power production in a sub grid exceeds that of the consumption there is no place for it to go, which will cause problems. All nodes on the low voltage layer will communicate with the market and power level controllers. As can be seen on figure 2.2 on page 14 their communication is with the secondary controllers and the aggregators.

Consumers:

Households:
: Households can be either flexible or inflexible, depending on the electric appliances in the house. Households make up the majority of the nodes in the low voltage layer. There are both households in the rural parts and in cities, the main difference being the number of houses on the line, and how far apart they are. Households may have smaller energy producing units, such as a wind turbine or a photo-voltaic unit.

Farm:
: Farms generally behave as inflexible consumers, as the equipment taking care of animals must be able to run whenever it is needed. Farms are more likely than households to have a power generator, for example a wind turbine or a photo-voltaic unit.

Producers:

Photo-voltaic:
: The same as for solar farms, however, on a much smaller scale. Solar panels have become increasingly popular in Denmark the last few years.

Wind turbine:
: Weather dependent as a wind farm, again on smaller scale. Wind turbines are not as common as photo-voltaic units.

The amount of data, and what kind of data sent between the controllers and the consumers, greatly depends on what kind of control algorithm is used to regulate the power consumption. It is assumed that the current and expected future power consumption/production will be needed.

2.2 Communication in the smart grid

While the addition of market controllers, and the upgrade of technical controllers, allows better regulation of the power in the grid, it does require that there is some form of communication in the smart grid. This communication should relay information between entities within the smart grid, and possibly to entities outside the smart grid as well. The information transmitted depends on the controllers and entities on the network because controllers require different information with different timing requirements.
There are several concerns that should be considered when implementing a communication network for the smart grid. There should be sufficient security to ensure that data is received correctly, and is not accessible for anyone other than the controller needing it. The network should furthermore be reliable, preferably with a backup network in case the main network experiences faults.
The communication network is split into two parts, the Distribution Network (DN) and the Access Network (AN). The AN is the last hop to the end user, and the DN is the network from the AN to the controller. This distinction is made as it can be beneficial to use different communication technologies for these two networks. Several different communication technologies are considered for use in the smart grid, an overview of these can be seen in table 2.1 on the facing page. We consider three QoS parameters, delay, throughput, and packet loss. It has, however, not been possible to find packet loss measurements for all technologies considered in this section, and it will therefore be omitted in this part.
The Power Line Communication (PLC) technology is split into two, Wide Band (WB) and Narrow Band (NB). While WB provides higher throughput, the WB hardware is considerably more expensive than NB hardware. Asymmetric Digital Subscriber Line (ADSL) provides already implemented infrastructure, but the fact that it is a public network, can cause security risks that are not present in other networks. ADSL also provides significantly different throughput in UpLoad (UL) compared to DownLoad (DL). General Packet Radio Service (GPRS) is like ADSL already implemented, it does, however, have high latencies. Renting the GPRS network can also be expensive in the long run. Unlicensed wireless technologies(eg. Zigbee or Bluetooth) are a very cheap way to implement the AN, however, the limited range of these technologies means that the DN will have to be extended.

2.3 Market controllers versus technical controllers

There is a distinct difference between controllers in the smart grid. One type of controllers, the technical controllers, attempts to ensure that the voltage level does not rise or drop drastically, and that the frequency on the power lines remains close to 50 Hz. These controllers requires information regarding consumers and producers power requirements within as little as 5-6 ms [12]. The technical controllers are mainly considered by the energy department, as it is tied closely to the power grid.
The market controllers attempt to regulate the power consumption and production over greater time spans than the technical controllers. They attempt to ensure that the energy production matches the energy consumption by regulating flexible units at the consumers. They require information regarding energy production forecasts, and estimations of the power consumption of the coming time period.

Technology	**Throughput [Mb/s]**	**Delay [ms]**	**AN/DN**	**Comments**
PLC	NB:<0,128[5] WB: <40[6]	∼5[6]	AN	Utilizes the power grid for communication, but must, therefore, adhere to the same topology as the power grid.
GPRS	0,023[7]	2300[7]	AN	A GPRS subscription for each house in a country can be expensive in the long term. Relies on the reliability of the service provider.
ADSL	DL: 24[8] UL: 3,3	∼10[9]	AN/DN	Uses a public network which means no control over the network. Public networks can also cause security risks. Very changing traffic loads from other users can change the QoS.
Fiber optics	$14 \cdot 10^6$[10]	<1	DN	Implementation of this infrastructure is very expensive.
Unlicensed Wireless	1[11]	15[11]	AN	Very short range means that if used as AN the DN must go very close to the end user.

Table 2.1: Table of communication technologies considered for smart grid.

Because the regulation happen over a greater time period than for technical controllers, the timing requirements are more lax. Some controllers have loop durations of up to 15 minutes[13], which communication wise is a very long time.
Both of these controller types requires communication between them and entities on the smart grid.

The information found in this chapter leads to a general architecture for the smart grid, as seen on figure 2.2 with nodes, power lines, communication and controllers. The communication is shown as

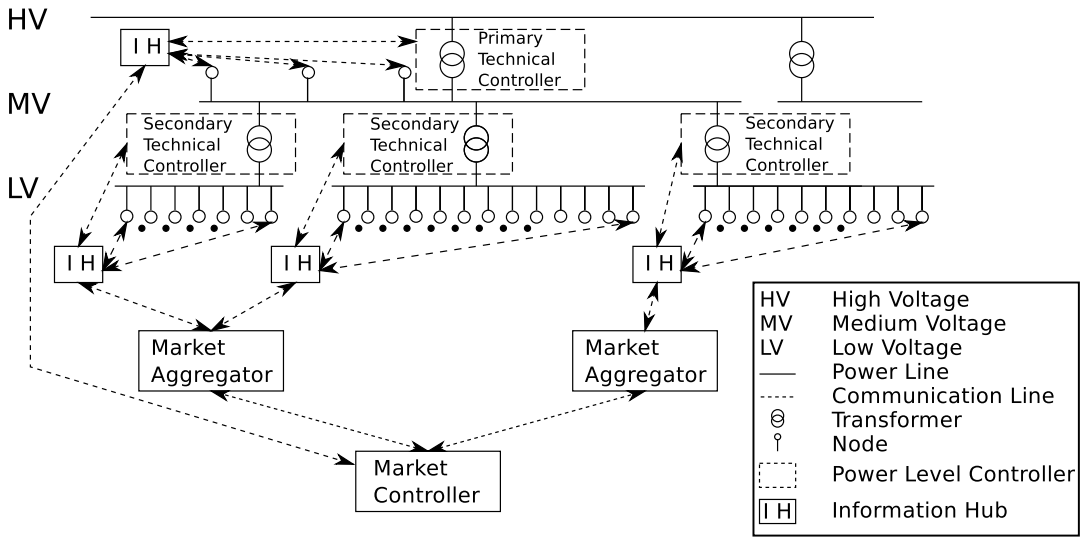

Figure 2.2: Overview of smart grid architecture, based on [14].

separate lines, but in some cases it can run on the power lines. Furthermore, the position of the market controllers could be the same physical location as the power level controllers. While it is not shown there is also communication between the primary technical controller and the secondary technical controllers.
The informations hubs shown in figure 2.2 indicates that in some cases it might be necessary to change the communication technology for increased range. The information hub will thereby serve as a link between the DN and the AN.

3 | Scenario

In this chapter we will answer the rest of question 1: *How should the communication network of the smart grid be defined?*, by answering the sub questions:

Question 1c: Which entities are communicating?

Question 1d: How often are entities communicating?

Question 1e: What is a realistic communication network for the smart grid?

To investigate the network design in smart grids, a scenario have been defined. This scenario will be imposed on the smart grid architecture defined previously. For this scenario a control algorithm is needed to show the difference between assuming perfect network conditions and not. It is chosen to consider a centralized demand response market controller. This controller is chosen as it was sufficiently simple for network engineers to fully understand. Choosing a market controller sets different requirements to the network than choosing a technical controller. Technical controllers in smart grids will typically have short control loops meaning that network delays becomes an important factor. The control loop time for market controllers is typically above one minute, causing packet losses to be of higher importance than network delays. In this chapter we will provide a technical description of the chosen control algorithm, as well as discussing communication technologies relevant for smart grid networks. Lastly the considered scenario will be specified.

3.1 Control algorithm

This section is based on the paper "Control approaches for residential demand response in Smart Grids"[15] and discussions with the lead author Luminita Totu[16]. This controller is a market controller that attempts to regulate the measured power consumption, by regulating controllable devices, in this case refrigerators. The regulation will depend on an estimation of the future power consumption. The current simulation implementation simulates two spikes in the power consumption over the course of the day, and it is during these spikes that the power consumption is attempted to be reduced.

The algorithm assumes N controllable units, which can be turned on or off, and M non-controllable units that have a stochastic time-varying on-off behavior. The N controllable units have some local constraints, which they use to determine whether it is on or off, they are: the temperature in the refrigerator, and how long it has been since it was last turned on or off. This means that no matter what the control signal is received at a refrigerator, its temperature can not go outside its bounds, and the refrigerator can not change state more often than every five minutes, unless the temperature constraint is not met.
It is not possible to put power back on the grid for this algorithm. The non-controllable units will have a higher chance of turning on during the morning and evening, and lower chance during the rest of the day. Because the only power consuming entities in the simulation are the M+N units, it is known that the power consumption will spike during two different time periods of the day.

The controller will get the total power consumption, and consider the referenced power consumption during the next few hours of the day. It will then, if there is expected to be a spike, send a signal to all controllable units informing them to increase their power consumption a few hours before each peak. This will store thermal energy in the refrigerators, thus when there is a spike in the power consumption the controllable units can be turned off, reducing the total power consumption during the spike. The

signal sent by the controller to the controllable units will be between -1 and 1. If the number is negative, for example $-0,5$, means that the controller expects that 50% of the units currently on will turn off. If it is positive it means that units currently off should turn on. The units will then decide internally if they actually turn on or off, based on the value received.

The general topology of the control algorithm can be seen on figure 3.1, where ϵ is the control signal, $\tilde{N}$ is an estimate of the amount of refrigerators in the on state if ϵ is below zero and an estimate of the amount of refrigerators in the off state if ϵ is above zero, r is the control reference, p_i is the average power consumption of a single refrigerator, and Z is the total power consumption. The supervisor control and estimation blocks are not in the scope of this paper, which means that $\tilde{N}$ is not an estimate but the true value, and r predetermined manually. The dispatch runs every 60 seconds.

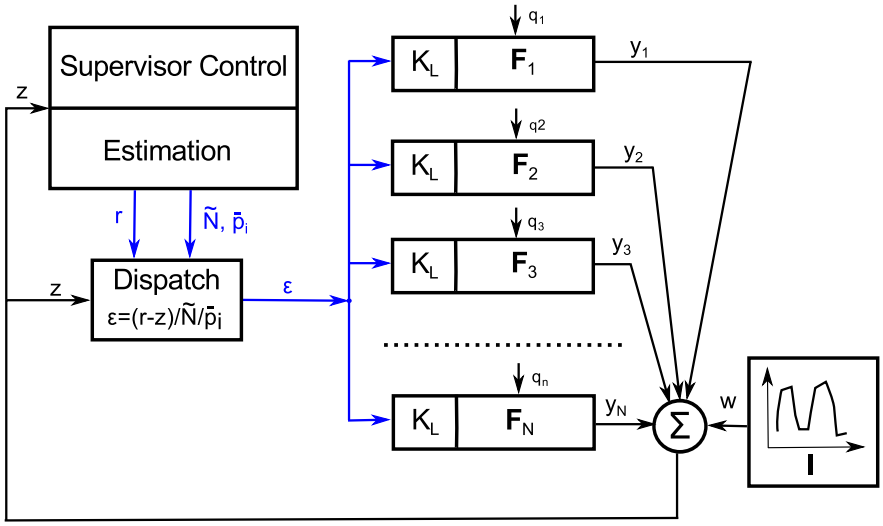

Figure 3.1: Overview of the topology of the control system[15].

ϵ is calculated from the formula:

$$\epsilon = \frac{f(r-z)}{N \cdot p_i}. \tag{3.1}$$

N is the amount of units in the appropriate state. The function f in equation 3.1 takes the difference between the reference consumption and the actual consumption as input, and multiplies it by -1 for its output. If the total power consumption is above the reference(ϵ is negative), ϵ is multiplied with $1,65$ to make the power consumption drop faster than it rises. This number is chosen by the algorithm designer by tuning it until the wanted effect is achieved. This is done as it is assumed that consuming too much power is worse than consuming too little. If the calculated ϵ value is positive, but so low that the expected number of refrigerators to turn on is less than 3, ϵ will be set to zero. The expected behavior of the control is, therefore, that the total power consumption should be slightly smaller than the reference. This behavior is shown in equation 3.3 to 3.4.

$$\epsilon = \frac{f(r-z)}{N_{\text{OFF}} \cdot p_i}, \quad \text{for} \quad \frac{f(r-z)}{p_i} > 0, \frac{f(r-z)}{N_{\text{OFF}} \cdot p_i} \cdot N > 3 \tag{3.2}$$

$$\epsilon = 0, \quad \text{for} \quad \frac{f(r-z)}{p_i} > 0, \frac{f(r-z)}{N_{\text{OFF}} \cdot p_i} \cdot N < 3 \tag{3.3}$$

$$\epsilon = \frac{f(r-z)}{N_{\text{ON}} \cdot p_i} \cdot 1,65, \quad \text{for} \quad \frac{f(r-z)}{p_i} < 0 \tag{3.4}$$

3.1.1 Assumptions

This control assumes that the number of currently turned on controllable units is known, and that average consumption of a controllable unit is known, and assumes that the reference regarding future power consumption is known. It is also assumed that the control algorithm knows the current power consumption whenever it needs it, and that there is no delay between requesting it and getting it. There is no network implemented in this control system which means that ϵ is assumed to reach all refrigerators within the loop time. Furthermore, the control algorithm assumes that all refrigerators are homogeneous.

3.2 Scenario specification

It is chosen to work with the control algorithm described in section 3.1 on page 15. It is a relatively simple control algorithm, and it was possible to get the implementation from [16]. It is furthermore an interesting scenario to consider from the network side. Figure 3.2 is a sub-figure of figure 2.2 on page 14, and details what will be focused on in this project.

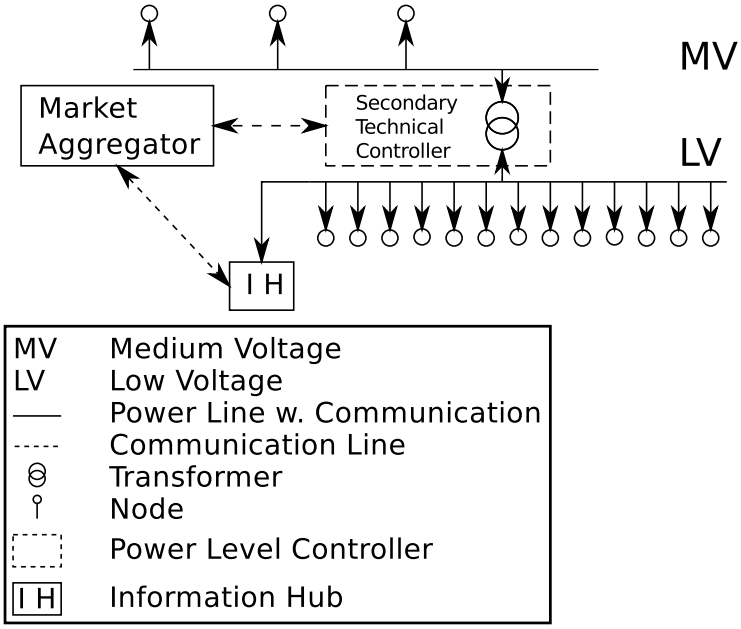

Figure 3.2: Limited scenario overview.

The provided control algorithm only consider lights (inflexible units), and refrigerators (flexible units), at households. As it is outside the scope of this project to add more entities to the controller, all other entities in smart grid are disregarded. This means that all power generators, and all other consumers are not simulated. The flexible units will receive a control message every $60s$.
As described in section 2.2 there are several technologies which can be used to ensure communication in smart grids. This project will only focus on some of these technologies, as it is out of the scope of this project to consider all of them.
In [17] the main communication interface described for smart grids in Europe is PLC. Because PLC is commonly used, and it fulfills all the requirements for smart grids, it will be used as the communication interface from consumers on the low voltage level to a information hub. From the information hub, another technology will transfer the data to the market controller. Due to the poor QoS parameters of PLC it is assumed that this link will not provide a restriction for the communication.
It is assumed that additional information will be needed to be transmitted in the network, which is not used by the aggregator. This information will generate cross traffic, which will reduce the networks QoS parameters. This information could be information used by the secondary controller.

The market controller will provide information to the market aggregator enabling it to calculate the its control reference defining the allowed power usage of the particular subgrid. This information is needed by the market aggregator to make a correct control decision when informing entities to either increase or decrease power consumption. Currently there is no support for changing this information in the control algorithm, and so this communication is not considered.

The market aggregator uses the total power consumption, z, in the sub grid for the control. z is measured at the secondary technical controller, and will then be send to the market aggregator. If this information does not reach the market controller it will decide ϵ on wrong or old information. This communication link is not implemented in the considered control algorithm for the market aggregator, and will, therefore, not be considered in this project. It will be assumed that this information always reaches the market controller on time.

As seen on figure 3.2 on the previous page, only a single market aggregator controlling the refrigerators under a single secondary technical controller is considered in this project. Since only the market aggregator will be simulated, this will be referred to as ”the controller” for the rest of this report.

4 | Network Monitoring

This chapter will discuss question 2: *How should a network monitor be designed?*. The following sub questions should help answering this question:

Question 2a: How is network monitoring performed on a network?

Question 2b: How/if does the control algorithm affect the monitoring?

As explained in chapter 1 on page 7 it is wished to adapt the control algorithm to the network conditions. This requires an estimation of the network parameters, which is found by monitoring the packets in the network.

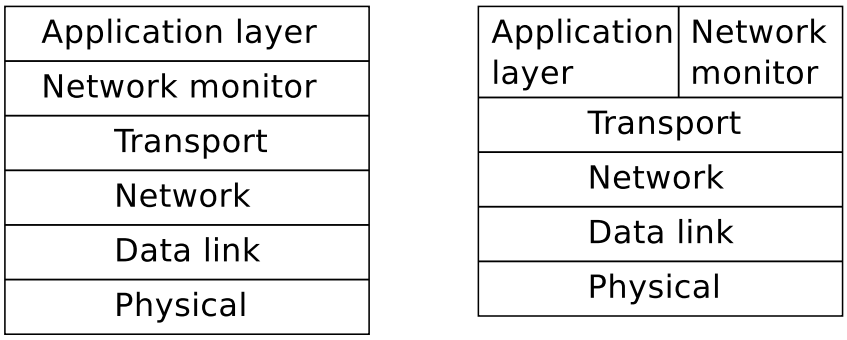

Figure 4.1: There are two considered approaches regarding placement of the network monitor.

Generally there are two approaches towards the placement of a network monitor in the OSI model. It can either be placed below the application layer, or it can be placed on the application layer. If it is placed below the application layer all packets must pass through it, which can potentially increase the delay experienced, this method will, however, enable passive measurements. If it is placed on the application layer only packets designated to the monitor will be received, and only active measurements can be done. It can then be chosen to place it on either side of the communication network, or only on one side. If the network monitor is placed on both sides of the network it is possible to measure one way QoS parameters, providing a more precise picture of the network state. As the network state is used in the controller, making measurements from both sides of the network means that measurement results must be transfered to the controller. If the network monitor is only placed on the one side of the communication network only two way QoS parameters can be measured, and one way QoS parameters must be estimated from that.

This chapter explores how to achieve an estimate of the network QoS, that can be used to adapt the controller. As QoS parameters we consider packet loss, latency and throughput, all considered from the application layer, as this is what will impact the controller. The impact of these parameters on the control performance will be discussed, and estimation methods of the parameters will be discussed. For the purpose of this project, we consider an Internet Protocol (IP) network using Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). The goal of the network, is to get the control signal from the controller to the refrigerators, and get the response from the refrigerator to the controller before the $60s$ control loop time is exceeded.

When using TCP, lost packets will be retransmitted causing increased delays. However, in networks with a very high packet loss probability, the probability of enough packets being lost for TCP to drop the connection can not be neglected, and application layer packet loss will still be experienced. A message sequence diagram showing the communication between the controller and a refrigerator is shown

on figure 4.2. The figure shows a scenario where at the first try the data packet is lost, it is then retransmitted this time, however, the acknowledgement is lost. In the third try both packets gets through the network, and the refrigerator sends its response which is acknowledged by the controller. Several packets can, however, be lost, until the packet is successfully acknowledged or the retransmission limit is reached.
Throughput will also cause additional delay, which for low throughput values will be significant compared to the latency. The TCP control messages will also experience these delays, which will worsen the overall delay further. The fact that all QoS parameters will cause increased delay when using TCP, means that the network delay can be long enough for it to exceed T_s, the $60s$ control loop time.
UDP has no error correction, and the delay will only be a sum of the latency and the throughput. As seen in the three possible scenarios for the UDP transmission on figure 4.3, the chance of this delay exceeding $60s$ is quite small, even with the low values of throughput considered in this project. This means that, when using UDP, the main contributer to decreased control performance is packet loss.

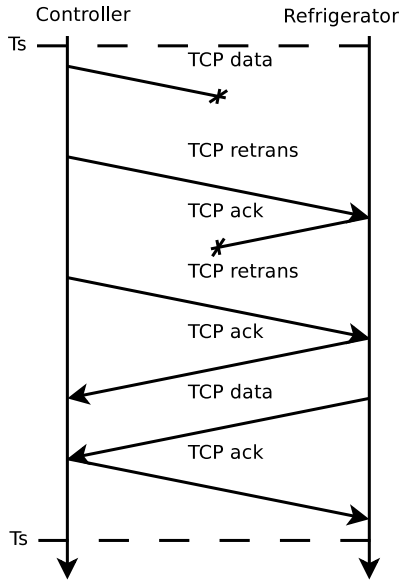

Figure 4.2: Message sequence diagram showing the controller-refrigerator TCP communication with non-zero packet loss.

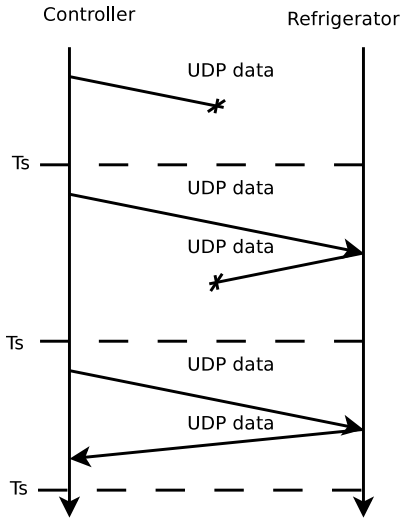

Figure 4.3: Message sequence diagram showing the controller-refrigerator UDP communication with non-zero packet loss.

4.1 Measuring Delay

Measuring delay can be done as active or passive measurements. By making active measurements, a detailed picture of the delay can be achieved, meaning that it is possible to determine which part of the delay is caused by throughput and which are caused by propagation delay. Active measurements does, however, introduce additional network traffic, which can be problematic when considering slow networks. When measuring delay actively it is necessary to consider if the packets send introduces extra delay for the other packets on the network, causing it to degrade the service and rendering the measurements incorrect, and if so how much of an impact it actually has on the measurements. It must also be considered if the packets send are representative of the packets for which it is desired to measure the delay. Some protocols may have higher priority in certain routers, thus making measurements with the wrong protocol can cause the measured delay to be incorrect.
Using passive measurements causes almost no additional traffic. It can, however, be necessary to put

additional information, like time stamps, in the packets transmitted. Passive measurements are bounded by the traffic patterns of the controller, which means that it may not be possible to achieve the full delay picture. Throughput estimation can also be problematic with passive measurements, as packet size can not be fully controlled, however, the measured delay with passive measurements will include the delay from the throughput on a packet of that specific size. This means that if packet size remains constant, a combined delay and throughput profile can be measured. As the important delay is the delay from the controller transmits the control signal until the controller receives the response from the refrigerator, and the packet sizes remain constant, the combined delay will be sufficient. This also means that the delay measurement can be both started and ended at the controller, and time synchronization between the controller and the refrigerators are not necessary.

4.2 Measuring Packet loss

The considered controller calculates a control signal based on the amount of refrigerators that will receive this signal. Therefore, packet losses from the controller to the refrigerator will cause the system to react differently than the controller expects it to. If packets are lost from the refrigerators to the controller, a fairly accurate estimate of the ratio between refrigerators in the on state and refrigerators in the off state can still be achieved. Therefore, the packet loss from the controller to the refrigerators are more important than from the refrigerators to the controller.

Like the delay, the packet loss can also be measured both actively and passively. Active measurement will again cause additional network traffic, but will provide a more detailed picture of the packet loss. As most networks defined by Bit Error Rate (BER) instead of packet loss, the packet loss will be dependent on the packet size, meaning that passive packet loss measurements will only apply to packets of the same size, whereas active measurements can change packet sizes, and achieve a better estimate of the BER. Active measurements are taken by sending a predetermined number of packets from the controller to some of the refrigerators, the refrigerators will then estimate the packet loss in one direction, and transmit the results to the controller. In the considered scenario, both packets from controller to refrigerator, and from refrigerator to controller is the same size, further more every time the controller sends a packet to the refrigerator, a response packet is expected. By knowing this, the controller can count the amount of responses received when it transmits its control signal and get the 2-way packet loss. From the 2-way packet loss the 1-way packet loss can be estimated under the assumption that the packet loss is the same in both directions, and that all packet losses are independent. The 1-way packet loss estimation is seen in equation 4.4.

$$1 - P_{2way} = (1 - P_{1way})(1 - P_{1way}) \tag{4.1}$$
$$= 1 + P_{1way}^2 - 2 \cdot P_{1way} \tag{4.2}$$
$$0 = 2 \cdot P_{1way} - P_{1way}^2 - P_{2way} \tag{4.3}$$
$$P_{1way} = 1 - \sqrt{P_{2way}} \tag{4.4}$$

This estimates precision is very dependent on the amount of samples it can get. If there is only 10 refrigerators it would need information from almost all of them to ensure a precise enough estimate, whereas with a 1000 it would only need information from a few percent of the total number.

4.3 Scenario modification

The overview, given on figure 4.4 on the next page, is an abstraction level of the system. It shows a modification to the control algorithm described in chapter 3 on page 15. Instead of estimating the

amount of refrigerators that are turned on from measured power consumption, the refrigerates send back their current state, 1 for on and 0 for off. This is used to estimate the amount of refrigerators that are on (not the true number due to packet loss), and allows measuring the packet loss passively. This is by counting the number of packets received at the controller and then using equation 4.4 on the previous page to find the one way packet loss. While this modification of the control algorithm will increase the network load, it is deemed worthwhile due to the increased accuracy it will give the estimation of the refrigerator state, which will improve the performance of the control algorithm. The network monitor

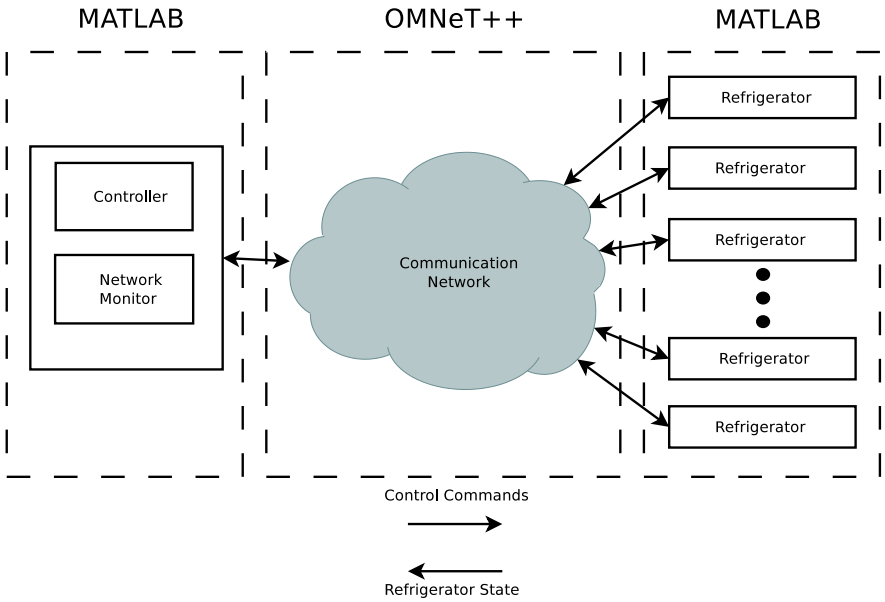

Figure 4.4: Entire system with network monitor shown.

will only measure the packet loss, and not the delay. This is because for UDP, which is the only place the measured network parameters are used, the delay is assumed inconsequential compared to packet loss.

5 | Simulation framework

The results in this project is based on simulations done in the simulation framework developed in [1]. Since work was done as a semester project at Aalborg university, and is not published, a resume of the work is provided in appendix A. In this chapter we provide a short overview of the framework, identify the parts of the original simulation framework that does not suit the needs of this project, and provide a new framework solution that can be used instead. This chapter will also describe any other implementations necessary for this project.

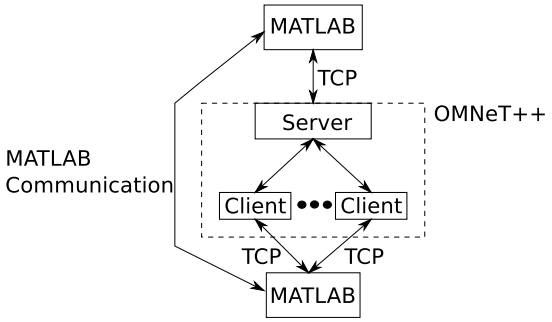

Figure 5.1: Overview of how the framework combines MATLAB and OMNeT++ for the simulations.

In [1] a framework was developed to simulate control algorithms in a smart grid scenario, and imposing imperfect network conditions on the algorithm. This was done by combining the preferred control simulator MATLAB with the preferred network simulator OMNeT++. These two where linked together by developing a communication protocol between them based on a TCP connection. An overview of this structure is shown on figure 5.1, where two instances of MATLAB is shown. This is because the framework allows for the controller to be simulated in one instance of MATLAB and the controlled units in another. The network simulator only uses the amount of data to simulate and not the actual data that should be sent on the network. By running two instances of MATLAB, the exchange of data will have to be handled by the two MATLAB instances and not the framework. The two MATLAB instances can also be merged, so all data is kept in the same workspace, and this data exchange becomes unnecessary.

As seen on figure 5.2 on the next page, the simulation time is controlled by OMNeT++, and MATLAB will be notified by OMNeT++ when events happen where MATLAB need to perform simulations. A protocol for how these notifications should be send was developed, which resulted in the following packet definition:

$$\begin{bmatrix} \text{Packet type} \\ \text{Server id} \\ \text{Client id} \\ \text{Size of data} \\ \text{Delay} \end{bmatrix}.$$

To show the use of the framework, [1] also develops a TCP based application for the framework, so a control algorithm could be simulated over an IP based network. As the TCP application is already developed, we continue working with IP based networks, and add a UDP application.

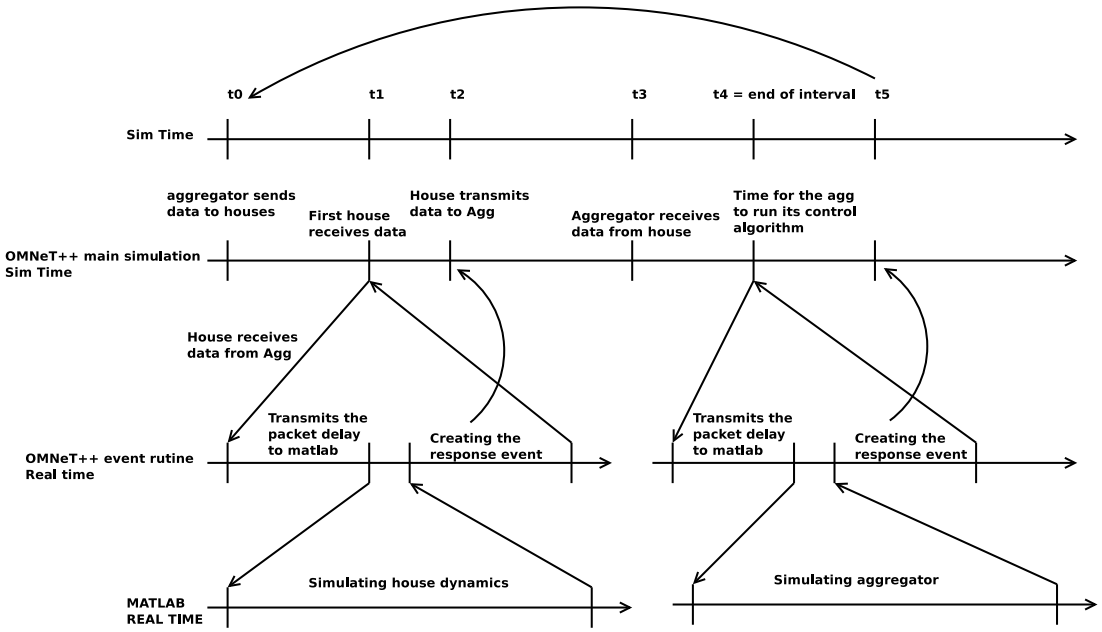

Figure 5.2: Overview of how time progresses through the simulation.

5.1 Framework modifications

The original framework makes a new connection from OMNeT++ to MATLAB every time information should be exchanged. This was done to allow each OMNeT++ node to use different MATLAB instances, for this project only one MATLAB instance will be used, and the many TCP connections being opened and closed increases simulation complexity. The `MatlabCommunication` class in OMNeT++ is changed to be a singleton class that is only created once per simulation. The MATLAB script `mainCA` is also changed to accept one connection during initialization, and use that connection for all communication.

A complete data set of each flexible unit and each inflexible unit was stored in a log file for each simulation minute. Using the performance metric defined in section 5.3 on page 26 requires quite long simulations, and this extensive data logging proved too space consuming. Therefore, only the cumulative power consumption is stored for both flexible and inflexible units.

To be able to add network monitoring and adaptive control, changes have also been made to the arguments the methods in the control class take. The updated class definition is shown below.

```
1  classdef ControlAlgorithm
2      properties
3          Variables
4      end
5
6      methods
7          function initControl
8          end
9          function houseSim(timestep,activeClients)
10         end
11         function controlSim(timestep,activeClients,packetloss,numTransmissions)
12         end
13     end
14 end
```

This project explores the use of mmPr as decision metric for the adaptive control. The reasons for using mmPr and a description of it can be found in chapter 7 on page 35. To determine models for the mmPr, the `mainCA` is changed to record mismatching states. The adaptive control explores the effect of multicasting the control signal several times to counteract poor mmPr. To be able to facilitate this, the amount of transmissions is added to the MATLAB/OMNeT++ protocol, so both programs agree how many times the control signal is transmitted. The revised MATLAB/OMNeT++ protocol definition is shown below.

$$\begin{bmatrix} \text{Packet type} \\ \text{Server id} \\ \text{Client id} \\ \text{Size of data} \\ \text{Delay} \\ \text{Number of Transmissions} \end{bmatrix}$$

The original framework came with a server and client application for TCP. As the considered control algorithm is well suited exploiting UDP multicasting, a server and client application is developed for UDP. When implementing the application in OMNeT++, a function that handles incoming messages is developed. This function can also send messages to itself if it needs to perform a task at a later point in the simulation.
This was developed with a server representing the controller, that will contact MATLAB at every control interval, get the control signal, and multicast it to all clients. Every time it receives a response from a client it will send it to MATLAB. When the control interval is over, it will signal to MATLAB that no more client responses will be received in time, so MATLAB knows how many responses was received in the time interval.
The client, simulating a house, is implemented so every time a packet is received from the server, it informs MATLAB, and send the response back to the server. Flowcharts showing the UDP server and client implementation can be seen on figure 5.3 and 5.4.

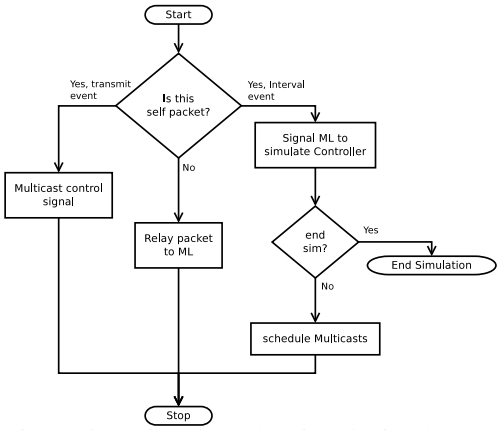

Figure 5.3: Flowchart showing the implementation of the UDP server in OMNeT++, simulating the controller.

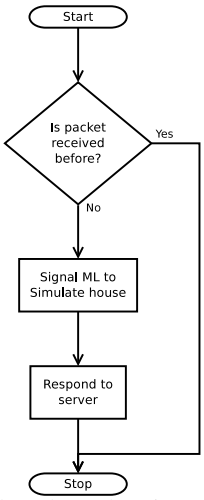

Figure 5.4: Flowchart showing the implementation of the UDP client in OMNeT++, simulating a refrigerator.

5.2 Framework additions

The OMNeT++/inet framework does not provide a bus implementation suitable for simulating PLC. A simple bus is, therefore, implemented in OMNeT++. All nodes connected to the bus will be distributed uniformly along the bus with re-definable limits. A input buffer is attached to each node connected to the bus, allowing for packets to wait for the bus to be available. The first node connected to the bus will be designated master node, and upstream will be defined as traffic flowing towards this node and downstream as traffic going away from this node. Upstream and downstream is handled separately for full-duplex communication. Nodes waiting to transmit on the bus will be handled in a round robin manner. Values for BER and throughput can be defined as configuration parameters in the OMNeT++ framework. The bus implementation is illustrated on figure 5.5.

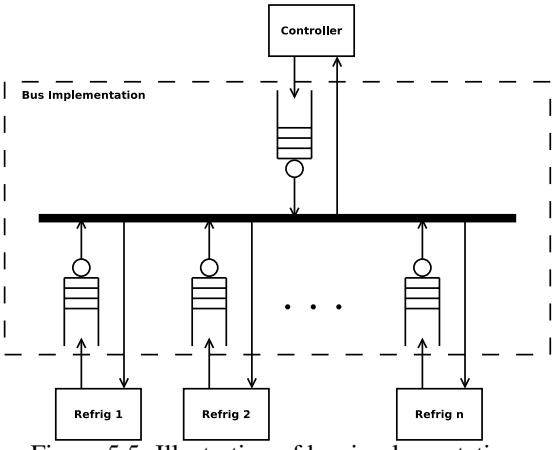

Figure 5.5: Illustration of bus implementation.

5.3 Performance metric

The control algorithm provided by [15], does not have a single metric with which it is possible to measure performance. The current way to measure the control algorithms performance is by visual inspection of the resulting graphs. This is, however, not sufficient when considering the performance of the network, and thus a new performance metric must be found. In this section several performance metrics will be discussed, and a single metric will be chosen. We define e as the performance parameter, $P_T(t)$ as the total power consumption at time t, shown as z on figure 3.1 on page 16, $P_R(t)$ as the control reference at time t, shown as r on figure 3.1 on page 16, T_H as the simulation termination time, and H as the Heaviside step function.

An obvious performance metric is to calculate the average euclidean distance between the total power consumption and the control reference, as shown in equation 5.1 on the next page. This distance can, however, not be seen as an error, as the controller is designed to get the power consumption close to the reference without going above it. Calculating the reference is out of the scope of [15], and is, therefore, chosen manually, and artificially large distances can occur, when the reference is unreachable, which could hide the differences in the different simulations. This metric would, however, be a measure of the controller quality, and could, therefore, be used not only to determine the impact of the network, but also to compare the controller to other controllers provided with the same scenario.

$$e = \frac{1}{T_H} \cdot \sum_{t=0}^{T_H} |P_R(t) - P_T(t)| \tag{5.1}$$

As the controller is designed to stay under the reference, the amount of time where consumption is above the reference could be used as performance metric. This can be seen in equation 5.2. This metric could also be defined as the average distance between total power consumption and the reference for samples above the reference. This is shown in equation 5.3. This performance metric will show the quality of the provided control, the impact on the controller caused by network degradation will, however, only be measured in the few samples in which the controller fails. This means that to get enough samples to definitively conclude on the results, the simulation will have to run for a longer time than with a performance metric that uses every sample.

$$e = \frac{1}{T_H} \sum_{t=0}^{T_H} H(P_T(t) - P_R(t)) \tag{5.2}$$

$$e = \frac{1}{T_H} \sum_{t=0}^{T_H} (P_T(t) - P_R(t)) \cdot H(P_T(t) - P_R(t)) \tag{5.3}$$

By running the simulation under perfect network conditions for many simulation days, the average total power consumption can be estimated for each minute of the day. A performance metric can then be defined as the difference between the total power consumption and the expected total power consumption. This is shown in equation 5.4. This metric will not measure the quality of the controller, but will only show the impact the network have on this specific controller. This method utilize every sample to find the performance of specific network conditions, allowing for shorter simulations than using the previous metric.

$$e = \frac{1}{T_H} \sum_{t=0}^{T_H} |E[P_T(t)] - P_T(t)| \tag{5.4}$$

It is chosen to use the last performance metric for this project. This is chosen as it is believed to show the most about the impact of imperfect network. The fact that this metric can not be used to compare different control algorithms, or scenarios, is less important for this project as we consider the same algorithm under different network conditions.
To create the performance metric, the control algorithm is simulated with 100 refrigerators for 1000 simulation days. The average is then calculated for each minute. The metric is plotted with the simulation values for a single day and the control reference on figure 5.6 on the next page.

By implementing these changes the framework is ready to use for the scenario defined in chapter 3 on page 15.

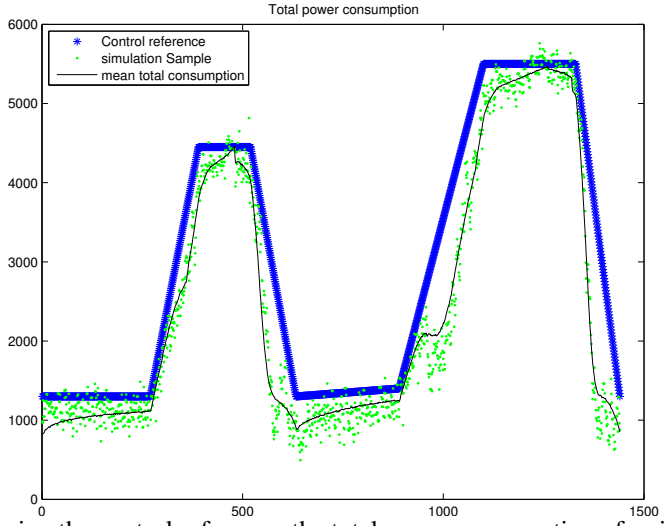

Figure 5.6: Plot showing the control reference, the total power consumption of a single simulation, and the total power consumption averaged over 1000 days.

6 | Adaption of the control algorithm by modifying epsilon

This chapter will explain how the measured packet loss of the communication can be used to adapt the control algorithm by modifying ϵ. It will be described how the control algorithm have been modified, and the results of the simulation will be shown. As this control adaptation works directly on the ϵ value, it will only work for this specific control algorithm. A more generalizable method will be described in chapter 7 on page 35. We consider the third question posed in the introduction: *How should the smart grid control be adapted based on the network conditions?*. The following questions elaborate further on the question.

Question 3a: Can the measured network parameters be used to adapt the control algorithm?

Question 3b: What adaption methods are there?

Question 3c: Does the control algorithm gain anything by being adaptive?

6.1 Unadapted controller results

First the results of the control algorithm without adaptation will be shown. All the following results have been simulated over the course of 15 simulation days, for illustrative purposes plots showing results is, however, only showing a single day. On figure 6.1 to 6.3, the green line indicates the power consumption from inflexible units (lights), the red flexible units (refrigerators), the black line the total power consumption, and the blue line is the reference used by the control algorithm to calculate ϵ. Figures 6.2 to 6.3 are all simulated with the same network parameters which can be seen in table 6.1.

Parameter	Value
PLC BER	$7 \cdot 10^{-6} \left[\frac{1}{m}\right]$
PLC throughput	50 [Kbps]
PLC range	0-400 [m]
PLC propagation speed	$200 \cdot 10^{6} \left[\frac{m}{s}\right]$
DN throughput	1 [Mbps]
DN delay	15 [ms]
Refrigerators	100
Control interval	60 [s]
Simulation Time	15 [days]

Table 6.1: Parameters used for the simulations.

Figure 6.1 on the next page shows a single day simulation run with perfect network conditions. This simulation is used to give a baseline for the other results.

On figure 6.2 on the following page are shown the result for the simulation using TCP. While TCP would usually ensure that packets always comes through the network, the network is in this scenario so poor that the TCP packets must be retransmitted several times, and in some cases the connection even time out. However, the poor network conditions also means that TCP is unable to get an estimate of the round trip time, causing the retransmit time to increase significantly. This can cause ϵ to be received after the loop duration is done, causing the refrigerator to make a decision on old data. This effect is seen by the controlled power consumption being more volatile than that of the perfect network.

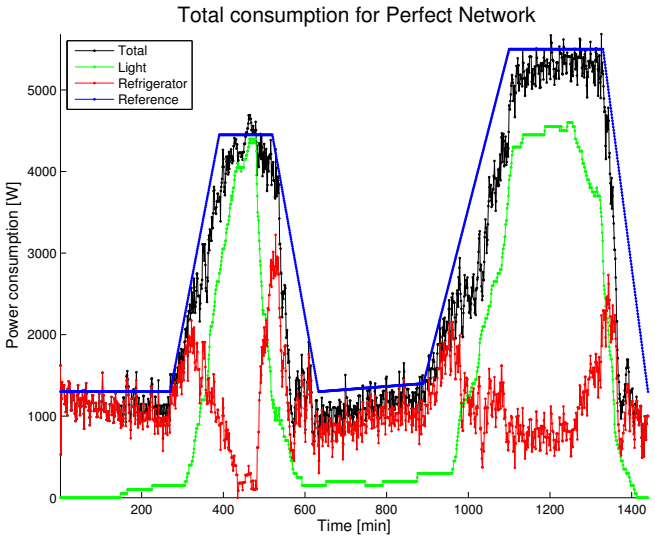

Figure 6.1: Results of the simulation with perfect network.

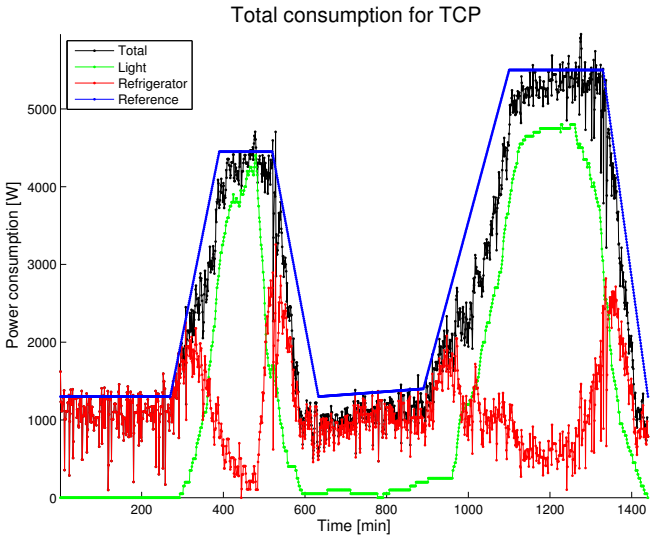

Figure 6.2: Results of the simulation with TCP.

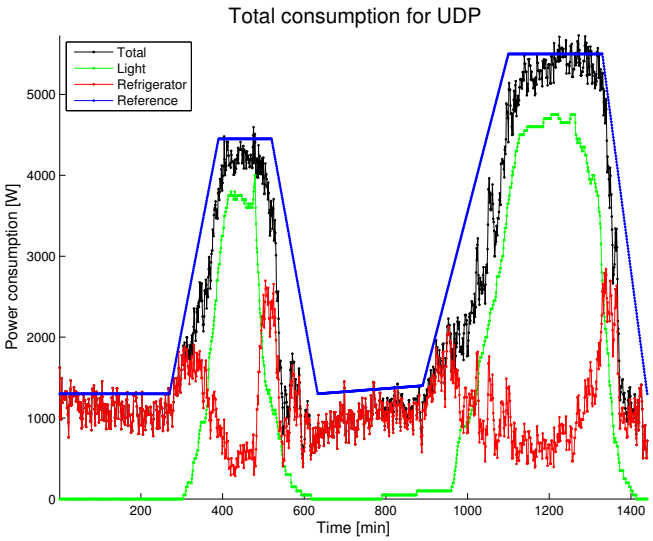

Figure 6.3: Results of the simulation with UDP without modification of epsilon.

Instead of using TCP the results in figure 6.3 uses UDP, which have the advantage of being able to multicast reducing the network load compared to TCP. This is seen by the UDP simulation being close to indistinguishable from the simulation of perfect network.

Network	**Mean error**			**STD of error**	
	Value	Degradation	95 % Confidence interval	Value	Degradation
Perfect network	164,5	0%	160,3 to 168,7	122,0	0%
TCP	185,7	12,9%	180,0 to 191,4	143,0	17,3%
UDP	173,8	5,6%	169,2 to 178,2	127,5	4,5%

Table 6.2: Performance of the TCP and UDP simulation compared to the simulation results of perfect network conditions.

The results in table 6.2 show the results of the three simulations. A more thorough description of this table can be found in section 6.3 on page 33. The error of the simulations have been plotted over the course of a single day, and can be seen on figure 6.4 on the following page. The graph has been smoothed using moving average of 75 timesteps. It shows that the main contribution to the error happens when the power consumption changes rapidly.

6.2 Control adaptation

Adapting a controller while it is running can be very tricky, as the control parameters are often chosen based on a fixed scenario. By changing the controller while it is running control stability can no longer be guaranteed. [18] proves that under certain conditions, the control loop interval can be changed and still ensure stability. It is, however, out of the scope of this project to detail how to ensure this stability. Instead we will focus on a relatively simple adaption of the control algorithm, and not analyze the effects of the adaption on the control, or the recalculating of parameters for increased performance. This does, however, result in a control algorithm that may not be optimal from the point of view of the controller, or in extreme cases may become an unstable controller.

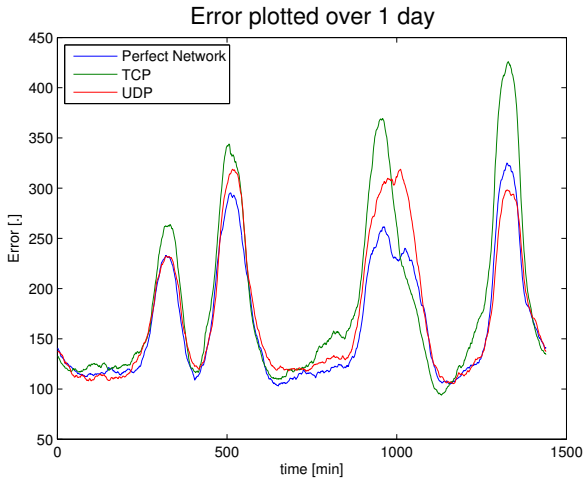

Figure 6.4: The error of the three simulations, plotted over a single day.

This adaptation of the control algorithm is to modify ϵ with regards to the 1-way packet loss measured by the network monitor described in chapter 4 on page 19. The loss of packets leads to the controller having fewer flexible units it can regulate, essentially reducing the number of flexible units in the system. When sending the ϵ value the controller expects that a certain percentage of the flexible units will change state, however, this percentage have been calculated from the assumption of a fixed number of flexible units. If this number is reduced then the amount of flexible units being changing state is changed, leading to the controller not performing optimally. We attempt to reduce this effect by scaling the ϵ value according to packet loss measurement.

For positive ϵ, it is calculated from:

$$\epsilon = \frac{f(r-z)}{N_{\mathrm{OFF}} \cdot p_i} \tag{6.1}$$

However, the true value for N_{OFF} is lowered by packet loss. We define ϵ' as ϵ calculated using the proper value for N_{OFF}

$$\epsilon' = \frac{f(r-z)}{(N_{\mathrm{OFF}} \cdot (1 - packetloss)) \cdot p_i} \tag{6.2}$$

Inserting the ϵ into this formula by isolating $f(r-z)$ from equation 6.2 gives:

$$\epsilon' = \frac{\epsilon \cdot N_{\mathrm{OFF}} \cdot p_i}{(N_{\mathrm{OFF}} \cdot (1 - packetloss)) \cdot p_i} \rightarrow \tag{6.3}$$

$$\epsilon' = \epsilon \cdot \frac{1}{(1 - packetloss))} \tag{6.4}$$

For negative values of ϵ, N_{OFF} will be interchanged with N_{ON}, and the resulting ϵ' remains unchanged.

The measured packet loss might not be the same as the actual packet loss. This can lead to ϵ being increased to much and become suboptimal. This error can lead to decrease in the performance of the control algorithm. It is, however, assumed that a more accurate ϵ value will be of greater importance than risking ϵ being too high.

6.3 Adapted controller results

In this section we show the results for the simulation of the control algorithm using adaptation of ϵ. The results are with the same parameters as for the unadapted controller, those in table 6.1 on page 29, and over the same time period. The graph uses the same color scheme, that is green for inflexible units (lights), red for flexible units (refrigerators), black for total power consumption and blue for the reference.

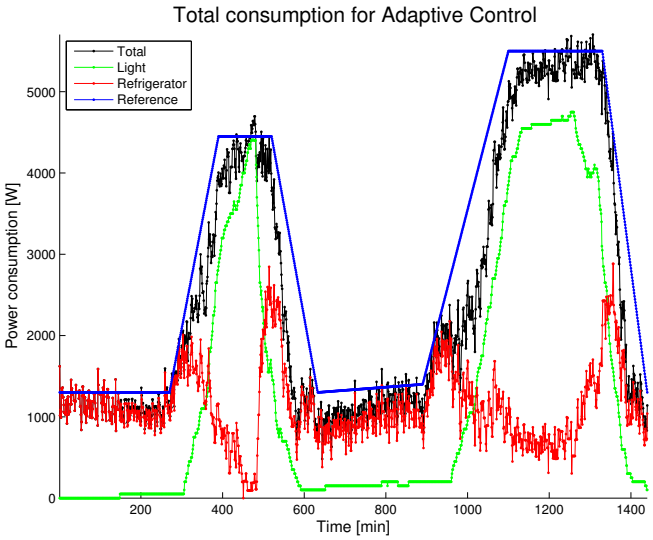

Figure 6.5: Results of the simulation with UDP with modification of epsilon.

The results on figure 6.5 also uses UDP, however, these results also use a network monitor to estimate the packet loss. The packet loss is then used to modify ϵ, as previously explained.

The results of the simulations performance can be seen in table 6.3. The performance of the perfect network when compared to the performance metric is not zero due to the randomness in the simulation. The simulation with perfect network conditions is used to compare all other results. As can be seen the adaptive control performs better than UDP without adaption, which again performs better than TCP.

Network	**Mean error**			**STD of error**	
	Value	Degradation	95 % Confidence interval	Value	Degradation
Perfect network	164,5	0%	160,3 to 168,7	122,0	0%
TCP	185,7	12,9%	180,0 to 191,4	143,0	17,3%
UDP	173,8	5,6%	169,2 to 178,2	127,5	4,5%
Adaptive Control	166,5	1,2%	161,8 to 171,2	123,2	1,0%

Table 6.3: Performance of each simulation compared to the simulation results of perfect network conditions.

6.4 Discussion

It is through out the simulations noticed that the probabilistic design of the control algorithm, results in a high tolerance for packet losses, as well as setting very low throughput requirements. This type of

control strategy, therefore, can be very beneficial when working under poor network conditions.
While there are several different methods for adapting a control algorithm, it was chosen to focus on a simple implementation. We show that it is possible to take the network conditions into account when transmitting the control signal. The results show that 4% performance can be gained, compared to using an unmodified controller, with UDP.

After completing the simulations, an updated version of the inet framework for OMNeT++ was released. This update corrected a bug in the implementation of TCP. This error meant that when a connection was lost due to packet loss, the TCP state was not updated. This resulted in when the connection was to be used again, a new connection was not established, but the old one was simply used as if it was not closed down. This reconnection procedure would have added additional traffic on the network, and increase the delays experienced by TCP packets going through the network. It is, therefore, believed that simulations with a proper TCP implementation would worsen the TCP performance further. Due to time constraints the application layer implementation using TCP was not reimplemented to fit the updated version of TCP. For these reasons, TCP is excluded from the simulations during the rest of the report.

7 | Adaptive control based on information mismatch probability

This chapter will explore the usage of the mmPr as decision metric for the adaptive control algorithm. The mmPr is the probability of a node in a network having incorrect information about other network nodes at a critical time T_c. As it is the primary objective of the network to provide nodes with information, the probability of information mismatch can, therefore, be seen as a network performance metric that combines all QoS parameters into a single metric. If a node in a network must make a decision based on the state of other nodes there is a potential for mmPr. If the node receives all information there are no problems, however, if some of the information is lost the node may take its decision on old information. There is a risk that this old information is no longer representative of the actual state of the other nodes on the network, giving a mismatch in the information. This is what is regarded as mmPr. For further information on the concept of mmPr we refer to [19]. In this chapter we focus on the same questions as in chapter 6 on page 29, question 3: *How should the smart grid control be adapted based on the network conditions?*. Two of the subquestions are also considered:

Question 3a: Can the measured network parameters be used to adapt the control algorithm?

Question 3c: Does the control algorithm gain anything by being adaptive?

This topic will be described analytically, then investigated based on simulations, finally simulation results will be discussed and reflected upon. Throughout this chapter all calculations will be based on communication using UDP.

Through a single control loop, two things happen on the network, as illustrated on figure 7.1. First the controller sends the control signal to all refrigerators, then all refrigerators responds with their updated state. If imperfect network conditions causes the control signal to be lost for some refrigerators, the system will essentially become less flexible, and it will not be possible to control the system with as high precision. If some of the responses are lost, the controller will still be able to estimate the state of the remaining refrigerators fairly accurately depending on the packet loss. It is attempted to improve the mmPr by retransmitting the control signal a number of times dependent on the current network conditions, but the refrigerators will only transmit their state once per control loop. This is implemented so that for k control signal transmission, the control loop will be divided into k equal parts, and the control signal will be transmitted at the beginning of each part.

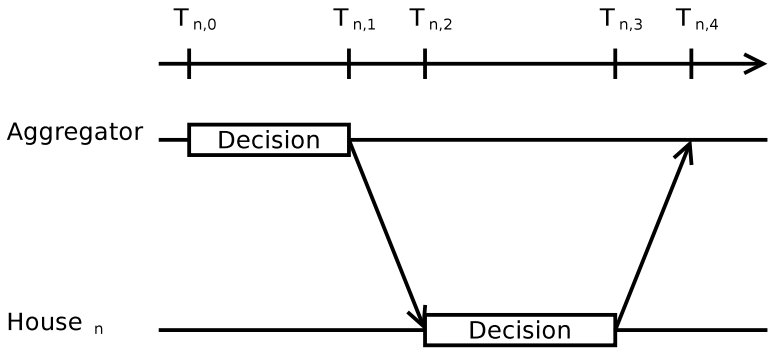

Figure 7.1: Overview of the timings considered when calculating the mmPr for a single house.

7.1 Passive packet loss estimation

In the scenario where the control signal is transmitted more than once, the one way packet loss estimation found in section 4.2 on page 21 is no longer valid. Instead a new method must be used that takes into account the number of transmissions used. The development of the one way packet loss estimation from the packets received is shown equation 7.1 to 7.6, where P_L is the packet loss, n is the number of responses received at the controller, and N is the total amount of refrigerators in the system. In equation 7.1 to 7.5, the ratio between received responses and the number of refrigerators is shown as a function of the packet loss probability, for a number of transmissions between 1 and 4. In these equations, an extra term is added for each number of transmissions. In equation 7.6 the expression is generalized to any number of control signal transmissions.

$$1: \frac{n}{N} = (1-P_L)(1-P_L) \tag{7.1}$$
$$2: \frac{n}{N} = ((1-P_L)+P_L(1-P_L))(1-P_L) \tag{7.2}$$
$$3: \frac{n}{N} = ((1-P_L)+P_L(1-P_L)+P_L^2(1-P_L))(1-P_L) \tag{7.3}$$
$$4: \frac{n}{N} = ((1-P_L)+P_L(1-P_L)+P_L^2(1-P_L)+P_L^3(1-P_L))(1-P_L) \tag{7.4}$$
$$\frac{n}{N} = (1-P_L)^2(1+P_L+P_L^2+P_L^3) \tag{7.5}$$
$$\mathrm{M}: \frac{n}{N} = (1-P_L)^2 \sum_{i=0}^{M-1} P_L^i \tag{7.6}$$

We insert the geometric series in equation 7.6, and a general expression for the packet loss estimation is found in equation 7.8.

$$\mathrm{M}: \frac{n}{N} = (1-P_L)(1-P_L^M) \tag{7.7}$$
$$\frac{n}{N} = P_L^{M+1} - P_L^M - P_L + 1 \tag{7.8}$$

To estimate the one way packet loss, the $(M+1)$'th order polynomial from equation 7.8 must be solved.

7.2 Analytical overview of mismatch probability theory

The mmPr is based on the information a specific node has at a specific time. For this specific case we look at the information the controller have at the time it has to run its control algorithm. At this point in time, the controller will have a state vector for the estimated states of the different refrigerators in the system. We define this vector as V_{ctrl} for N refrigerators as shown in equation 7.9. We then define the state vector of the actual states of the refrigerators in the system, V_{true} as shown in equation 7.10.

$$V_{\mathrm{ctrl}} = [V_{ctrl1}\ V_{ctrl2}\ V_{ctrl3}\ ...\ V_{ctrlN}] \tag{7.9}$$
$$V_{\mathrm{true}} = [V_1\ V_2\ V_3\ ...\ V_N] \tag{7.10}$$

The mmPr can then be defined as:

$$mmPr = P(V_{\mathrm{ctrl}} \neq V_{\mathrm{true}}) \tag{7.11}$$

If we look at the connection between an controller and house number n, illustrated on figure 7.1 on the previous page. At time $T_{n,0}$ the controller runs its control algorithm and decides how the house should

behave, this is send to the house at $T_{n,1}$ and received at $T_{n,2}$. The house then decides how to react, and does so at time $T_{n,3}$. And at time $T_{n,4}$ the controller is updated with the state of the house, $S_{n,t}$.
The control algorithm on the controller will run at constant intervals, and it is therefore critical that this information update is done before the controller must run again, the critical time T_c is therefore set to the interval of the control algorithm. This means that to avoid information mismatch, between the controller and house n, when the controller must run, it must either hold that

$$T_n \leq T_c \tag{7.12}$$

Where $T_n = T_{n,4} - T_{n,0}$, or that house n does not to change state. This means that the mmPr between the controller and house n caused by the transmission delay will be

$$mmPr_{D,n} = 1 - P(T_n \leq T_c)) - P(S_{n,t} = S_{n,t-1}) + P(S_{n,t} = S_{n,t-1}) \cdot P(T_n \leq T_c)) \tag{7.13}$$

This mmPr is dependent on the delay time, for which a reasonable distribution should be obtainable. The probability of the refrigerator state is, however, more complex, as it will be dependent on the control signal for that time instance.
An information mismatch can also be caused by the packet loss, P_L. When using reliable transmission protocols, like TCP, the packet loss is transformed into an increased delay, and the packet loss probability will approach zero. For protocols without packet correction, the mmPr will be dependent on the packet loss. We define P_L to be the probability of experiencing any packet loss in the system, which could be either when the controller transmits the control signal, or when the refrigerator responds.

$$mmPr_n = mmPr_{D,n} + P_L - P_L \cdot mmPr_{D,n} \tag{7.14}$$

Under the assumptions that the mmPr for each house is independent of each other the total mmPr becomes

$$mmPr_A = 1 - \prod_{n=1}^{N} (1 - mmPr_n) \tag{7.15}$$

The final mmPr depends on the network delay, the packet loss, and the probability of changing state. As this scheme is intended for a power balancing controller for smart grids, the critical time, T_c, will typically be $60s$ or above. This means that, when using UDP, the network delay will not be a substancial part. The high packet loss in the system will, however, be of great importance to the mmPr. The probability of changing state will have impact on the mmPr although the controller uses the number of refrigerators in a specific state, and two inverted state mismatches can, therefore, cancel each other out. Two mmPr definitions can, therefore, be made, a strict mmPr where all refrigerators must match the state information at the controller, and a mmPr where the estimated amount of refrigerators in the on state must be correct. These two mmPr definitions will be referred to as "the strict mmPr" shown in equation 7.16 and "the lenient mmPr" shown in equation 7.17 respectively.

$$mmPr_{\text{strict}} = P(V_{\text{ctrl}} \neq V_{\text{true}}) \tag{7.16}$$

$$mmPr_{\text{lenient}} = P\left(\sum V_{\text{ctrl}} \neq \sum V_{\text{true}}\right) \tag{7.17}$$

Producing a mathematical model for the mmPr is out of the scope of this project. The problem is, however, addressed from a mathematical point of view by [20], from where three models are received. As the mathematical model becomes rather complex, some simplifying assumptions are made.

- Mismatch is considered in the strict sense as described in equation 7.16.
- Refrigerators responds to all packets received.

- Refrigerator states are not saved at the controller, so all refrigerators must get a response successfully through the network to avoid an information mismatch.
- Transmission time and packet loss for packets from refrigerators to the controller are all independent of each other.

In practice, refrigerators responding to all received packets may not be beneficial, as previously explained. Assuming no transmission dependencies for response packets, may not be accurate as the packet are send on the same medium with in a short time interval, but it will likely be a close approximation, and is also assumed for all other work in this project. Additionally, the three models have different assumptions regarding the transmission of the control signal.

1. Transmission delay and packet loss for control signal packets are independent for all refrigerators. For this assumption to be accurate, the control signal should be send as independent information flows. Practically this could be done by using different communication lines for each house, essentially removing the ability to broadcast.
2. Transmission delay is independent for control signal packets, packet loss events are the same for all refrigerators. For a packet loss event to be shared between all refrigerators, the packet must be broadcasted on a shared medium. For transmission delays to be independent the network routes for each connection must be different, a practical case for this assumption have not been found.
3. Transmission delay and packet loss are the same for all refrigerators. If the control signal packet is broadcasted on a single medium, packet loss could be assumed to impact the message for all refrigerators. Under the assumption that the delay caused by a node being further down the bus than another node is insignificant compared to the rest of the delays, this would also cause shared delay between all refrigerators.

The assumption that transmission delay and packet losses are independent for the control signal for all refrigerators, is likely not valid as the control signal is broadcasted to the refrigerators. For simplicity, this is, however, the assumption used for this project. Assuming that if a control signal packet is lost, no refrigerators will be able to receive it, will probably be inaccurate as it is the signal strength will decrease with transmission distance. Assuming independent packet loss will, however, also be inaccurate as the the control signal is broadcasted in the same packet on the same medium. As the physical propagation delay will be small compared to the throughput delay, assuming the same transmission delay for all refrigerators might be more accurate than assuming independent transmission delays. Based on the assumptions, it is, therefore, believed that the third model will be the most accurate, and the first one the least accurate.
Notation for the models are defined by [20] and shown in table 7.1 on the facing page.

The three models are shown in equation 7.18, 7.19 and 7.20.

Case 1:

$$mmPr(S) = \left(1 - \prod_{j=1}^{m}(1 - p_u p_d(F_u * f_d)(T_j))\right)^n . \tag{7.18}$$

Case 2:

$$mmPr(S) = \sum_{s_1^u,\ldots,s_m^u \in \{0,1\}^m} \left(1 - \prod_{j=1}^{m}\left(1 - s_j^u p_d(F_u * f_d)(T_j)\right)\right)^n p_u^{m_u}(1 - p_u)^{m - m_u}. \tag{7.19}$$

Parameter	Description
T_c	The time of the deadline.
n	The number of refrigerators, indexed by $i = 1, ..., n$.
m	The number of times the control signal is transmitted, indexed by $j = 1, ..., m$
S_j	The transmission time of the particular control signal.
T_j	The remaining time from the j'th transmission to the deadline given by $T_j = T - S_j$.
p_u	The probability of the control signal being received successfully(i.e. $1-p_u$ is the packet loss).
F_u	The distribution function of delay from controller to refrigerator.
f_u	The density function of delay from controller to refrigerator.
s_{ij}^u	Indicating a successful transmission of the control signal. $s_{ij}^u = 1$ for success, and 0 otherwise.
$t_i j^u$	The time the control signal is received at the refrigerator.
m_u	The total number of successful control signal transmissions.
p_d, f_d, F_d, s_{ij}^d,t_{ij}^d	Is defined for response transmissions similarly to the definitions for control signal transmissions.

Table 7.1: Mathematical notation for mmPr models provided by [20].

Case 3:

$$mmPr(S) = \sum_{s_1^u,\ldots,s_m^u \in \{0,1\}^m} \int_0^{T_1} \cdots \int_0^{T_m} \left(1 - \prod_{j=1}^{m} \left(1 - s_j^u p_d F_d(T_j - t_j^u)\right)\right)^n \times \prod_{j=1}^{m} f_u(t_j^u) dt_m^u \cdots dt_1^u p_u^{m_u} (1-p_u)^{m-m_u}. \tag{7.20}$$

Though it is expected that the accuracy of the models increases for each case, the mathematical complexity is also greatly increased. The first case is possible to solve analytically, while there is no closed form analytical solution for case 3.

7.3 Mismatch probability simulations

In this project we investigate the mmPr by simulation, these simulations will later be compared to the case 1 analytical model. The considered controller will be simulated under different network conditions, and effects of the network will be investigated from the data gathered during the simulations. We will also simulate the impact of transmitting the control signal more than once. All mmPr simulations are done using UDP, and as explained in chapter 4 on page 19 this makes the delay measurements insignificant, and they are, therefore, omitted in the simulations. The parameters shared by all simulations are shown in table 7.2 on the following page, and an overview of parameters changed in the different mmPr simulations are shown in table 7.3 on the next page. Each simulation is set to simulate the system for a given amount of days, after which the number of times the control signal is transmitted is incremented.

Two types of simulations are done. For the first type each number of transmissions are simulated for a single day, and will be referred to as "the short simulations". Simulating for a single day reduces the simulation time sufficiently to simulate for a higher number of transmissions, as well as doing more simulations varying the BER of the PLC network. Using a single day does, however, mean that control performance estimation will not be precise enough to be meaningful. A single simulation day

Parameter	Value
PLC throughput	50 [Kbps]
PLC range	0-400 [m]
PLC propagation speed	$200 \cdot 10^6 \left[\frac{m}{s}\right]$
DN throughput	1 [Mbps]
DN delay	15 [ms]
Refrigerators	100
Control interval	60 [s]

Table 7.2: Parameters shared by all simulations.

Simulation number	PLC BER [x10^{-3}]	Simulation days	Transmissions
1	0	1	1-9
2	0,35	1	1-9
3	0,70	1	1-9
4	1,04	1	1-9
5	1,39	1	1-9
6	1,74	1	1-9
7	2,08	1	1-9
8	2,43	1	1-9
9	0,35	15	1-6
10	1,04	15	1-6
11	1,74	15	1-6
12	2,43	15	1-6

Table 7.3: Parameters used for the different simulations. Simulation days is for each number of transmissions.

will, however, be sufficient to estimate the mmPr meaningfully. The second simulation type will be simulated for 15 days, and will be referred to as ”the long simulations”. For these simulations, control performance can be estimated, and the effects on retransmissions on the control performance will be investigated.

From the short simulations, the average amount of response packets arriving at the controller for each control loop is found. This is shown on figure 7.2 on the facing page. As expected, the amount of packets received at the controller is inversely proportional to the BER. It is also seen that transmitting the control signal more than 6 times will not increase network performance for any of the considered BER. For this reason the long simulations will be run with up to 6 transmissions.

For the short simulations the mmPr is plotted with regards to amount of control signal transmissions and the BER. This is seen on figure 7.3 for the lenient mmPr and on figure 7.4 for the strict mmPr. As expected the strict mmPr is significantly higher than the lenient mmPr. It is also seen that transmitting the control signal more than 6 times does not decrease the mmPr noticeable.

From the long simulations, the control performance is shown together with the mmPr for a BER of $1,74 \cdot 10^{-3}$ of figure 7.5 on page 42. From this plot it can be concluded that the control performance is not changed noticeably by retransmitting the control signal. This trend was also experienced with lower values for the BER, and these results are, therefore, not shown here.

On figure 7.6 it is seen that by increasing the BER to $2,43 \cdot 10^{-3}$, the network conditions are poor enough for the control performance to be affected, and a positive effect of the retransmissions are clear.

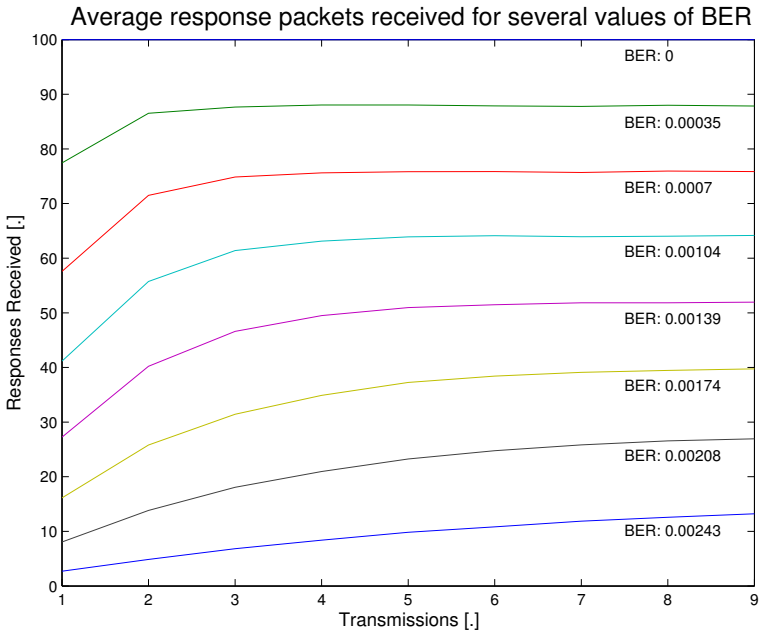

Figure 7.2: Plot showing the average amount of responses received in a single control loop, for different network conditions and number of transmissions.

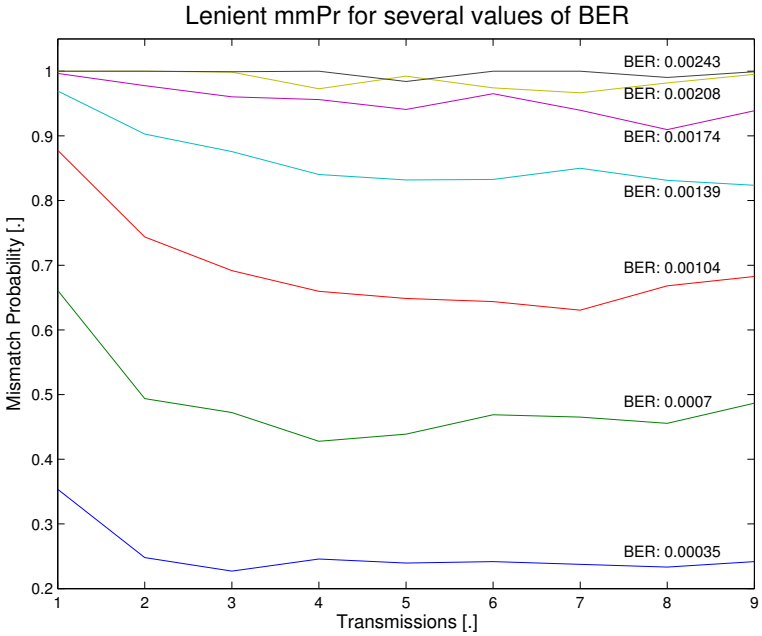

Figure 7.3: The lenient mmPr plotted with respect to BER and number of transmissions.

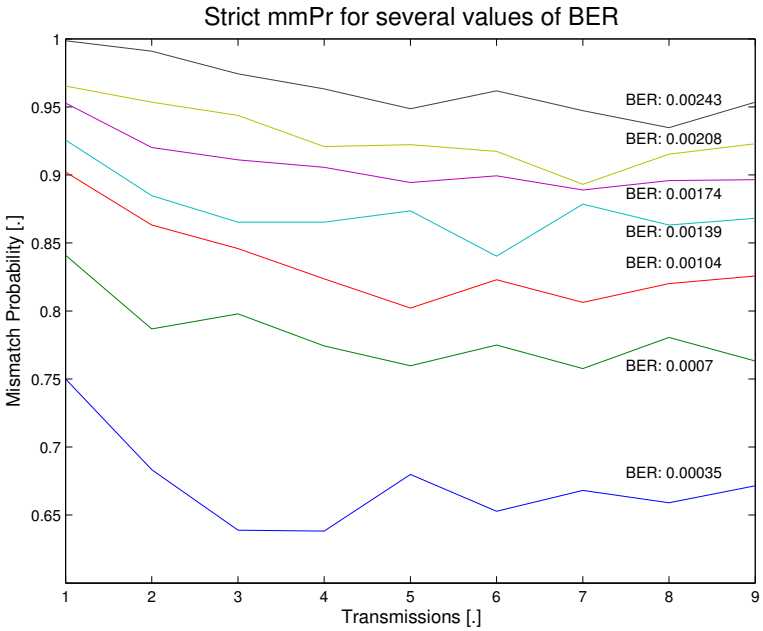

Figure 7.4: The strict mmPr plotted with respect to BER and number of transmissions.

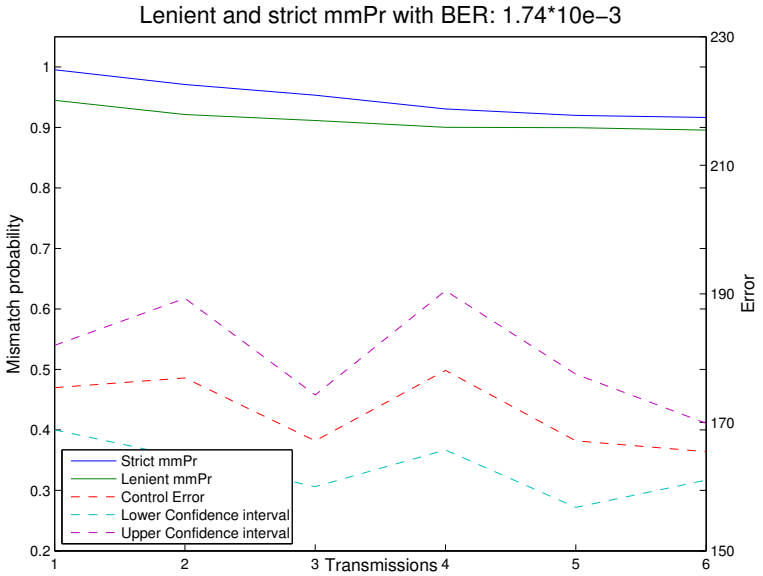

Figure 7.5: Both mmPr cases shown together with the control error with respect to the amount of transmissions. Shown for a BER of $1,74 \cdot 10^{-3}$.

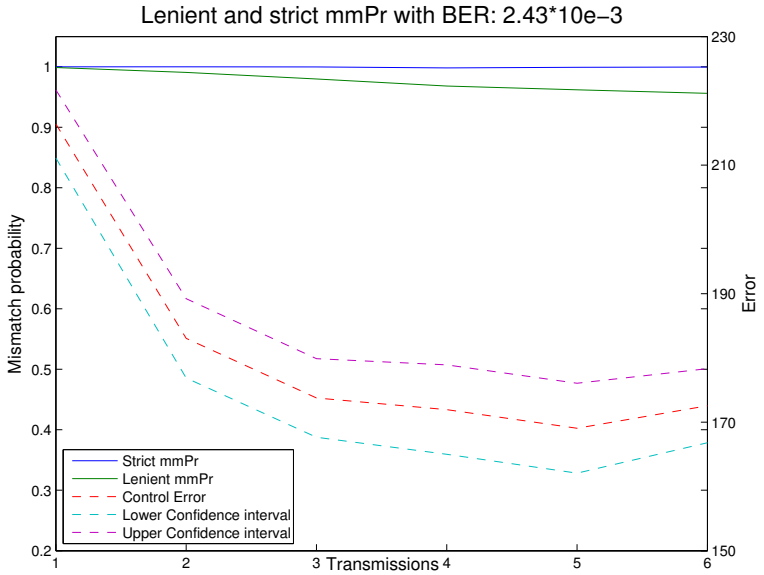

Figure 7.6: Both mmPr cases shown together with the control error with respect to the amount of transmissions. Shown for a BER of $2,43 \cdot 10^{-3}$.

7.4 Comparison between simulations and analytical mmPr

To be able to compare the analytical model to the simulations, the case 1 analytical model shown in equation 7.18 on page 38 is evaluated. It is chosen to compare it to the simulation with a BER of $0,35 \cdot 10^{-3}$, and the parameters for the analytical model is, therefore, taken from this simulation. In the simulation, the packet size for the control message and the response message are both 344 bits, making the probability of a packet successfully being transmitted:

$$p_u = p_d = 1 - 344 \cdot 0,35 \cdot 10^{-3} = 1 - 0,1204 = 0,8796 \tag{7.21}$$

When evaluating the term $(F_u * f_d)(T_j)$ it will become the distribution function for the sum of the delay in both direction. The distributions for the network delay is, in both directions, assumed to be exponentially distributed, and the mean value of the two distributions are measured in the simulation. The two exponential distributions are combined into an Erlang distribution with parameter λ calculated from the mean value of the two distributions. The parameters of the model is shown in table 7.4.

Parameter	Value
m	1-6
n	100
p_u	$0,8796$
p_d	$0,8796$
$F_u * f_d$	erlang(k,λ)
k	2
λ	$\frac{1}{(131+140)/2} = \frac{1}{135,5}$

Table 7.4: Parameters shared by all simulations.

The resulting mmPr is plotted together with the mmPr found by simulation. The result can be seen on figure 7.7 on the following page. From the results it is seen that the two mmPrs differs a lot from each other.

As the simulation only will send responses once for each control loop, it makes sense that the simulated strict mmPr does not decrease to the same extend as the mathematical model. Since the simulation have the refrigerator state saved from the last control loop, it is possible for the simulation to have correct information if the refrigerators, from which packets where lost, did not change state. This will cause lower mmPr from the simulation for low amounts of control signal transmissions. Though the differences between the two mmPr calculations are explainable, the difference is too great to conclude on the data. It can, therefore, be concluded that the assumptions of both methods should be reconsidered if a meaningful comparison is to be done.

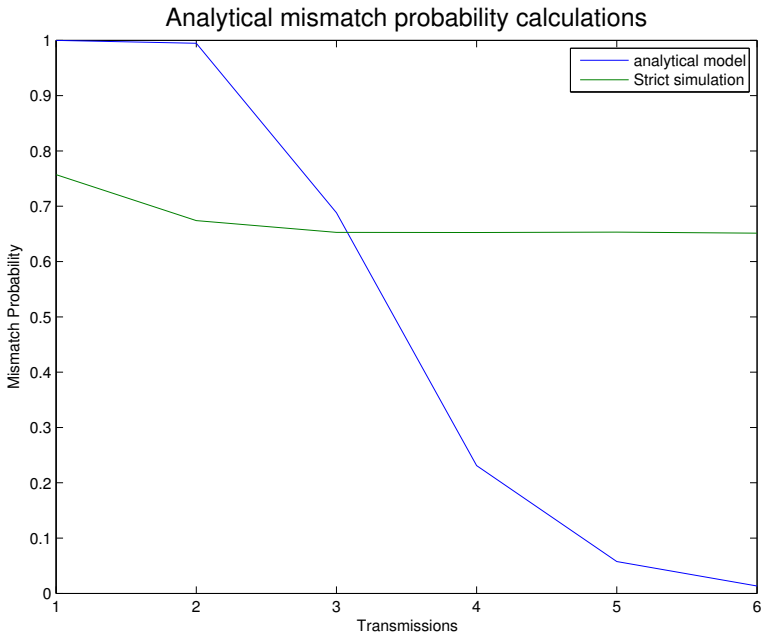

Figure 7.7: mmPr for the mathematical model, and the simulation with respect to the amount of control signal transmissions.

7.5 Discussion

From the simulations it is shown, that for sufficiently poor network conditions, retransmitting the control signal will be beneficial. The high resilience to network imperfections shown by the controller means that the mmPr becomes very close to 1 before the control adaptation makes sense. Using a decision metric that only takes values within a small interval may not be practical. If the method was to be used on a controller less resilient to network imperfections, the interval, within witch the mmPr would lie, would increase and the mmPr would be more useful as a decision metric. This claim is backed up by the fact that the results show that retransmitting the control signal causes lower mmPr.

8 | Conclusion

In this project we defined a scenario, and within this scenario a suggestion of a communication network for the smart grid was proposed. This suggestion was to have each entity in the smart grid connected to a controller via PLC, which means that the architecture follows that of the current power grid. The communication was illustrated using both TCP and UDP. It was furthermore illustrated in chapter 6 on page 29 that an imperfect network degrades the performance of a control algorithm.

It was found that a network monitor may be necessary for adapting the control algorithm dependent on network conditions. Using the chosen scenario it was possible to estimate the network conditions using only passive measurement techniques. The network monitor was placed at the controller and was measuring on the messages the controller receives. It was argued that when using TCP the key network QoS parameter was the network delay. For UDP the key QoS parameter was packet loss.

The main focus of this project was to adapt the control algorithm based on the measured network conditions. It was found that there exist several ways to modify a control algorithm, but most modifications would require better insight into the control discipline than was possessed by the project group. Two simple approaches was chosen to adapt the controller, one was to modify the control parameter value send by the controller, and the other to send the control value more times during a single control loop. Due to the supplied control algorithm not having a readily made performance parameter, a performance metric was designed to show changes in the specific control algorithm due to network errors.

Using the performance metric it was found that TCP showed a $12,9\%$ degradation, UDP showed $5,6\%$ degradation, and the adaptive control a $1,2\%$ degradation compared to the perfect network. This gives a $11,7\%$ performance increase from TCP to adaptive control. Likewise there was a $4,4\%$ increase in performance from UDP to adaptive control. From this it can be seen that there is an improvement to gain from modifying the control signal based on the packet loss, but it can also be seen that using UDP the control algorithm is very resilient towards network imperfections.

As modifying the control parameter requires a specific type of control algorithm, an approach more generalizable was developed. This approach investigated if transmitting the control signal several times during the control loop could increase control performance. To decide the number of times the control signal should be transmitted it was proposed to calculate the probability of having information mismatch in the network at critical times. This was chosen as this probability is a single metric showing the network performance taking all QoS parameters into account. This was mainly approached by simulating the mmPr using different network conditions and different amount of control signal transmissions.

From the mmPr simulations, it was concluded that the high resilience to network imperfections, shown by the control algorithm, meant that the mmPr became very close to 1 in most scenarios, making the mmPr impractical as a decision metric. It was, however, argued that a control algorithm less resilient to network imperfections would benefit from mmPr calculations.

9 | Assessment

Using simulations is one of the preliminary step towards implementing smart grid. Simulations are used to test the feasibility of certain scenarios, models, architectures etc. The simulations run in this project can be used to verify the assumptions regarding adaptation of control algorithms with regards to network parameters. This does not have to involve smart grid scenarios, as it should be possible to modify any networked control problem to this framework. While the adaptation of ϵ is only possible for problems with this specific type of controller, while retransmitting the control signal based on mmPr is more general and should be usable for most networked control problems. The main drawback with using the framework for other purposes is currently the rather long simulation times. This can become increasingly problematic as more complex networks would add even more complexity, and thus longer simulation times.
The mmPr estimation could be generalized further by adding active measurements of the network conditions, allowing the calculation of mmPr even on networks where passive measurements are not feasible.
When calculating the performance of the control algorithm, it is assumed that each day is independent, in terms of the error. This assumption is not entirely correct as the start of a day is Dependant on the end of the day that came before it. The dependence does, however, only stretch into a few minutes of each day, compared to 1440 minutes on a day, the assumption does not give rise to a large error.
When calculating the one way packet loss it is assumed that the packet loss is the same in both directions, which is indeed how it have been implemented in the bus. This assumption is, however, probably not a realistic assumption, as the packet loss in the downstream does not have nearly as much cross traffic that can cause collisions on the bus, as the upstream. Furthermore, the packet loss in the bus is assumed to be completely independent of other packets lost. A real bus would have some correlation, as packet loss is most likely due to noise and attenuation of the signal, increasing the Signal to Noise Ratio (SNR) to unacceptable levels.
The current implementation have no packet loss on the network between the information hub and the controller. This is because it is assumed that this connection have a significantly better QoS than the PLC connection. The information hub could furthermore be programmed to take a TCP packet from the controller, and convert to a broadcast packet to the refrigerators. Thus the assumption of no packet loss seems reasonable. The delay introduced in the information hub and on the communication line would increase, but this delay is deemed insignificant.
While some of the assumptions taken in this project could be drawn into dispute, with regard to their realism, it is firmly believed that the results found in chapter 6 on page 29 and chapter 7 on page 35 gives a useful insight into the workings of a control algorithm constricted by an imperfect network. While the results would be improved by longer simulations, the conclusions drawn in this report are still valid.

9.1 Future Work

There are many aspects of the smart grid that have not been considered in this project which could improve on the realism of the simulations. Here are some subjects which we would consider interesting to have include in the project.

- As the considered control algorithm allowed it, only passive network measurements was used. For a more generalized approach active QoS measurements could be added.
- The information in the smart grid are rather sensitive, and security of this data would be required. This includes security from cyber attacks against the controller, security against other people gaining access to private information, and ensuring that the entities only reacts on control messages from the controller.

- The current implementation of the PLC bus is rather simplistic. It would be interesting to see how it behaves with a more complex bus implementation, which have queues with finite buffers, a different physical layer protocol. The current mac protocol used is the point to point protocol, and it only encapsulates the packet. A more realistic implementation should be more aware of collisions and when to send packets.
- This project only consider UDP and TCP for the communication. There might be some non-IP based protocols that could be used to transmit the data. These protocols might have some advantages over UDP and TCP, possibly in the forms of security.
- While there are discussed and tested two methods for adapting the control algorithm, they might not be optimal from a control performance point of view. Other control adaptations would require greater knowledge within the control discipline, with this knowledge other control adaptations could be investigated.
- The power grid have a rather significant role in the smart grid, and simulations testing smart grid scenarios should have a model of it. The power grid gives extra constraints to the market controller, as it assumes that information regarding the power consumptions is available at all times, and the refrigerators have an instant effect on the power grid once they receive the control signal.
- Including the power grid in the simulations makes the inclusion of technical controllers, which regulate the voltage and frequency on the power lines, the logical next step. This could give rise to some interesting results because there might be conflicting control signals within the smart grid, because the market controller could wish to reduce power consumption overall, while the technical controller might wish to reduce the voltage in the network.
- These simulations have been tested under a specific scenario, but it could prove beneficial to test it against other scenarios to find the best suited controller for the smart grid. This would require a general performance metric for all controllers, allowing comparisons between simulations. Changing the scenario could be by trying different communication methods, or different control algorithms for the regulation.
- We make a comparison in chapter 7 on page 35 between the mmPr found analytically and the mmPr found from the simulations. However, as these are not using the same assumptions for the network they are difficult to compare. It would interesting to see how the simulation mmPr behaves under the same assumptions as for the analytical part, of if it is possible to find a analytical solution using the same assumptions as the ones used in the simulations.
- As described in the report some of the simulations have been done with cross traffic generated on the communication network. The model for this should be more detailed, but it was unfortunately not possible to get estimates of the communication of other controllers. This lead to the model for the cross traffic to be overly simplified.
- The simulations in this scenario have been run with 100 flexible units, and for 15 simulation days due to time constraints. The simulations should be done with more flexible units, we know that the controller have been tested with 1000 flexible units, and for a longer time period. Unfortunately the simulation frameworks complexity and the time given for the project did not allow such simulations.

Bibliography

[1] Jacob Theilgaard Madsen and Thomas le Fevre Kristensen. Framework combining network & control simulations in smart grids, 2012.
http://kom.aau.dk/group/12gr920/.

[2] SAP. Leveraging the 'smart grid' for smart decisions on network asset replacements, 2013.
http://cdn.blog-sap.com/analytics/files/2012/11/273207_l_srgb_s_gl.jpg.

[3] David A. Culler Randy H. Katz et al. An information-centric energy infrastructure: The berkeley view, 2009.
http://www.eecs.berkeley.edu/~ychen2/professional/InfoCentricEnergyInfrastructure.pdf
http://upload.wikimedia.org/wikipedia/commons/5/56/Electricity_grid_schema-_lang-en.jpg.

[4] climateminds.dk. Smart grid concept, 2007.
http://www.climateminds.dk/index.php?id=686.

[5] F. Aalamifar, H.S. Hassanein, and G. Takahara. Viability of powerline communication for the smart grid. In *Communications (QBSC), 2012 26th Biennial Symposium on*, pages 19 –23, may 2012.

[6] S. Galli, A. Scaglione, and Zhifang Wang. For the grid and through the grid: The role of power line communications in the smart grid. *Proceedings of the IEEE*, 99(6):998 –1027, june 2011.

[7] J. Korhonen, O. Aalto, A. Gurtov, and H. Lamanen. Measured performance of gsm, hscsd and gprs. In *Communications, 2001. ICC 2001. IEEE International Conference on*, volume 5, pages 1330 –1334 vol.5, 2001.

[8] Billion. Annex m technology enabling higher upstream speed by increasing the available frequency range, 2009.

[9] Nessoft. What is "normal" for latency and packet loss?, 2011.
http://www.nessoft.com/kb/42.

[10] NTT Science and Core Technology Laboratory Group. 14 tbps over a single optical fiber: Successful demonstration of world's largest capacity, 2006.
http://www.ntt.co.jp/news/news06e/0609/060929a.html.

[11] N. Baker. Zigbee and bluetooth strengths and weaknesses for industrial applications. *Computing Control Engineering Journal*, 16(2):20 –25, april-may 2005.

[12] Discussion during a meeting for smartc2net, 2013.
Meeting held between 6th to 8th of May.

[13] Klaus Trangbaek, Mette Pedersen, Jan Bendtsen, and Jakob Stoustrup. Predictive smart grid control with exact aggregated power constraints. *Smart Power Grids 2011*, 2011. From the book Smart Power Grids 2011.

[14] Discussion regarding use-cases for smartc2net, 2013.
Discussion had on the 12th of February.

[15] Luminita Totu, John Leth, and Rafael Wisniewski. Control for large scale demand response of thermostatic loads. *American Control Conference*, june 2013. To be published.

[16] Discussion with Luminita Totu from Control and Automation at AAU, 2012. Discussions had between 1st of October and 1st of December 2012.

[17] Rob Conant. Toward a global smart grid - the u.s. vs. europe, 2010. http://www.elp.com/index/display/article-display/2702271845/articles/utility-automation-engineering-td/volume-15/Issue_5/Features/Toward_a_Global_Smart_Grid_-_The_US_vs_Europe.html.

[18] J. Colandairaj, G.W. Irwin, and W. G. Scanlon. Wireless networked control systems with qos-based sampling. *Control Theory Applications, IET*, 1(1):430–438, 2007.

[19] Martin Bøgsted, Rasmus L. Olsen, and Hans-Peter Schwefel. Probabilistic models for access strategies to dynamic information elements. *Performance Evaluation*, 67(1):43 – 60, 2010.

[20] Discussion with Jakob G. Rasmussen from Department of Mathematical Sciences at Aalborg University, 2013. Discussions had the 23st of May and the 30st of May 2012.

Appendix

A | Interdisciplinary Simulations in Smart Grids - Resume

This chapter provides a resume of the report "Framework Combining Network & Control Simulations in Smart Grids"[1], which deals with the development of the simulation framework used in this project.

When control engineers design control algorithms, perfect network conditions are often assumed. Smart grids are a control system distributed across entire countries, and perfect network conditions is therefore a bold assumption. A framework allowing control simulations to be run with imperfect network conditions is therefore developed, and a test example is set up.

Control simulations are often done using the Simulink toolbox for MATLAB. MATLAB is however not suitable for complex network simulations, and it is therefore chosen to design the framework so that control engineers can design and implement their control algorithm in MATLAB, and network engineers can work in OMNeT++. This means that for the framework to work, proper interfaces has to be defined between MATLAB and OMNeT++. As this framework aims to provide interdisciplinary simulations, these interfaces will have to be defined so control engineers can use it without having deep knowledge about the field of network engineering, and vice versa.

By examining different market controllers, it is determined that a centralized control scheme will not be computationally feasible. The framework is therefore designed to support two basic control topologies seen on figures A.1 and A.2.

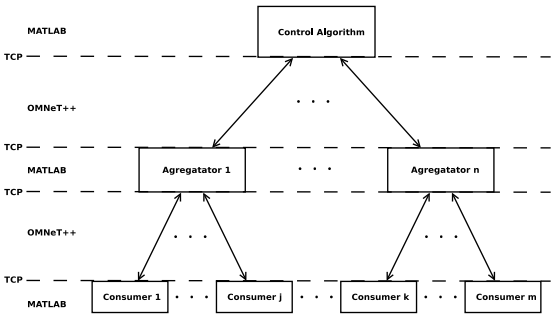

Figure A.1: Hierarchical control topology.

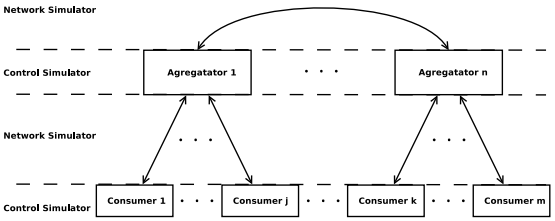

Figure A.2: Distributed control topology.

The framework consists of MATLAB platform on which control simulation should be build, and a similar platform for OMNeT++. Therefore three interfaces is defined, one from the control simulation to the MATLAB platform (MATLAB/MATLAB), one from the network simulation to the OMNeT++ platform(OMNeT++/OMNeT++), and one between the two platforms(MATLAB/OMNeT++). In addition a fourth interface is considered but not developed. This interface deals with the communication between different instances of the control simulation, if for example the simulation of the controller is done in another MATLAB instance than the simulation of the consumers. This interface is completely within the control simulations, and it is therefore chosen not to develop this interface. An overview of the simulation framework can be seen on figure A.3 on the facing page.

The MATLAB/MATLAB interface consists of a class definition that control simulations will have to adhere to. By doing that the MATLAB platform will be able to handle the simulation. This interface was developed with the specific test example in mind, as a deep analysis of the needs of different control simulations was out of scope. The class definition can be seen below.

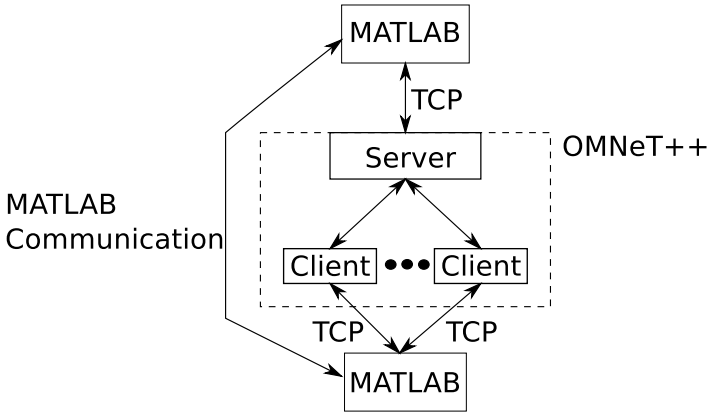

Figure A.3: Overview of the simulation framework,

```
classdef CA
    properties
        Variables
    end
    methods(Static)
        function initControl
        end
    end

    methods
        function clientsim(timestep,activeClients)
        end
        function serversim(timestep,activeClients)
        end
    end
end
```

The MATLAB/OMNeT++ Interface is based on TCP sockets. An application layer protocol is defined with eight different packet types defined that the framework adheres to. The packet definition for this protocol is shown below. The OMNeT++/OMNET++ interface is implemented as a class that handles the MATLAB communication.

$$\begin{bmatrix} \text{Packet type} \\ \text{Server id} \\ \text{Client id} \\ \text{Size of data} \\ \text{Delay} \end{bmatrix}$$

As a test example, two application layer nodes was developed for OMNeT++, a server node and a client node. The nodes can set up a TCP connection, and communicate based on instructions provided by a MATLAB control simulation. A simple network was set up using the developed nodes, and an existing control algorithm was fitted to the interfaces defined. The effects of the network on the control algorithm was then explored. Figure A.4 on the next page shows the control simulation without the network. The figures A.5 and A.6 on the following page shows the control simulation with network using extreme values of throughput and packet loss respectively.

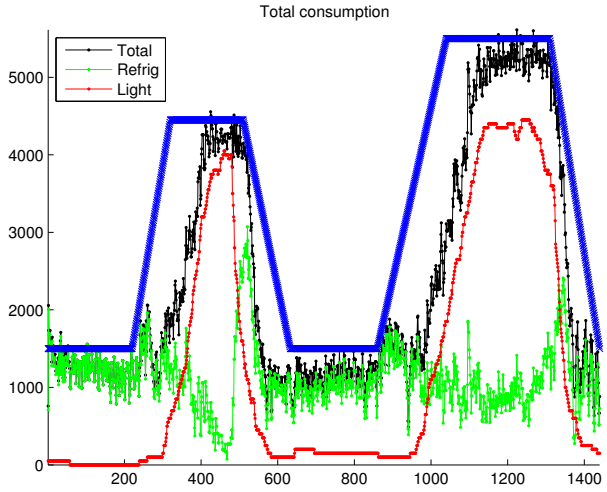

Figure A.4: Simulation of the control algorithm without network.

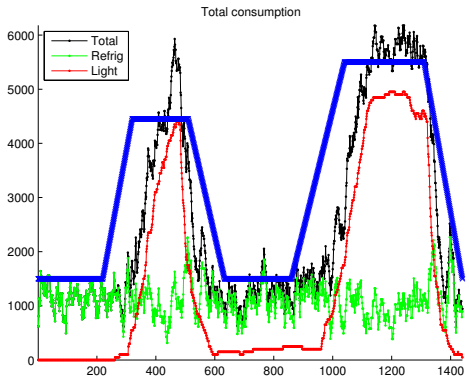

Figure A.5: Simulation with network using 750bps

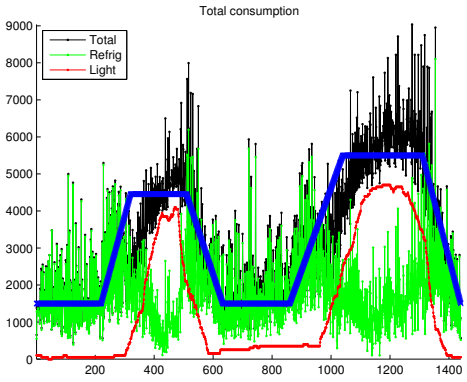

Figure A.6: Simulation with network using 36% packet loss.

It was concluded that assuming perfect network conditions can be very dangerous when developing smart grid, and that using a joint framework can provide insight into how the two effects each other.