University of Dublin

TRINITY COLLEGE

Electricity Trading Between Smart Nano-Grids: Matching Supply and Demand in the Face of Unpredictable Supply

Brian McNestry

B.A. (Mod.) Integrated Computer Science

Final Year Project May 2017

Supervisor: Dr. Donal O'Mahony

School of Computer Science and Statistics

O'Reilly Institute, Trinity College, Dublin 2, Ireland

Declaration

I hereby declare that this project is entirely my own work and that it has not been submitted as an exercise for a degree at this or any other university.

Brian McNestry, May 5 2017

Permission to lend

I agree that the Library and other agents of the College may lend or copy this report upon request.

Brian McNestry, May 5 2017

Abstract

As fossil fuels across the world are steadily depleted, the majority of the world's energy production will shift towards renewable sources. However, renewable energy sources do not always guarantee the same reliability in rates of production. Therefore new strategies must be developed to match supply and demand in the face of unpredictable supply. The growing popularity and proliferation of smart grid technology is one proposed method of solving this problem.

In this paper, a network implementation of a game theoretic approach to this problem is implemented in a smart nanogrid to prove whether or not such an approach is first feasible in a network, and secondly whether or not it would provide a better approach to the problem. A number of optimisation techniques, such as convex optimisation and hyperplane projection optimisation, are also employed to find a better solution.

Acknowledgements

- Dr Donal O'Mahony for being an excellent supervisor
- Luiz Da Silva and Georgios Iosfidis for their help with helping me to understand optimisation techniques

Table of Contents

Part I

Introduction

Currently in the field of smart grid technologies, there is an abundance of theoretical approaches to the problems of managing the transactions between consumers and suppliers of energy [1]. While there are multiple facets to this problem, this project examines the problem of matching supply and demand in the face of unpredictable supply. The world is slowly moving towards a future in which renewable energy sources will become our primary method of energy production, particularly with the inevitable depletion of fossil fuels [2]. This future, which will unfortunately be accompanied by a more unpredictable supply of energy, will require a smarter approach to the accumulation and distribution of energy. Due to the proliferation of decision making computers, the smart grid is a very popular solution. As mentioned previously, research has been conducted by hundreds of papers on theoretical solutions to this problem. However very few have real implementations and do not take into account the limitations and overheads of a real network.

The goal of this project is to implement a network solution to the problem of matching supply and demand in the face of unpredictable supply. The paper authored by Wayes Tushar [3] formed the basis of the project and therefore this report. The system will then be compared against a simple scenario of the current REFIT scheme [4] and an analysis will be provided to see whether or not this particular smart grid solution would be effective in a real world situation.

First a discussion of the technologies used and an explanation of the associated terms will be presented, including topics such as the smart grid, the current implementation of managing supply (REFIT), definitions of auctions and game theory, as well as a brief summary of the mathematical optimisation techniques employed in this project. Next, the design of the project will be examined and finally the actual implementation itself will be presented. To conclude there will be an assessment of the system designed and a number of potential continuations of the project will be discussed.

Part II

Background

Chapter 1

Decentralised Grid

For many years, in Ireland and in many other countries, the national electric grid infrastructure was controlled by a central body, namely the ESB. The various electricity providers all used the same distribution network. Fundamentally, the power was provided from each of the different providers and then routed into the same centralised hub belonging to the ESB. From there, each consumer (a household) received the energy that they paid for accordingly at a fixed rate through that same infrastructure belonging to the ESB.

This system has been in place for decades and lends itself very well to large companies providing a steady supply of energy by way of electricity plants that use both renewable and nonrenewable energy sources. Non-renewable energy sources, also known as fossil fuels, include resources such as coal, gas and oil. While these are finite resources, at present they can be burned at a steady rate in order to meet the demands of customers. Electricity from renewable sources can also be produced at a reasonably steady rate by placing large farms in areas that are particularly well suited to the type of renewable energy being produced. For example, large wind farms are set up in windy regions far removed from residential or urban areas and solar panels can be placed in regions that typically enjoy clearer skies than other areas.

However in the future, with the ongoing depletion of nonrenewable resources, increasing numbers of customers will turn to erecting solar panels and local wind farms on their own properties, regardless of whether or not they are living in a particularly sunny or windy area. Presently there are a few houses that use a solar panel to heat their water or other smaller tasks but soon more and more people will become more and more dependent on what they can produce either within their own home, or in a more collective sense in their own neighbourhood to power their houses [5].

The issue that then arises in these areas that aren't as sunny or windy is that supply of electricity is no longer steady. The current system could not be maintained as the energy produced on a local level would be small enough that it would not be worth passing this energy upstream to the central grid. The energy would instead be used at a local level to try to cover the demand for electricity of the house or business with which that particular device is associated.

The model of infrastructure that would then be required is that of a decentralised grid. This model would require a massive infrastructure overhaul in order to implement, so it would not exist until it is absolutely necessary and has been accepted by the major companies who would then go about implementing it. The rough idea of a distributed grid is described in figure 1.1. Throughout the rest of this report the phrases "distributed grid" and "decentralised grid" are used interchangeably.

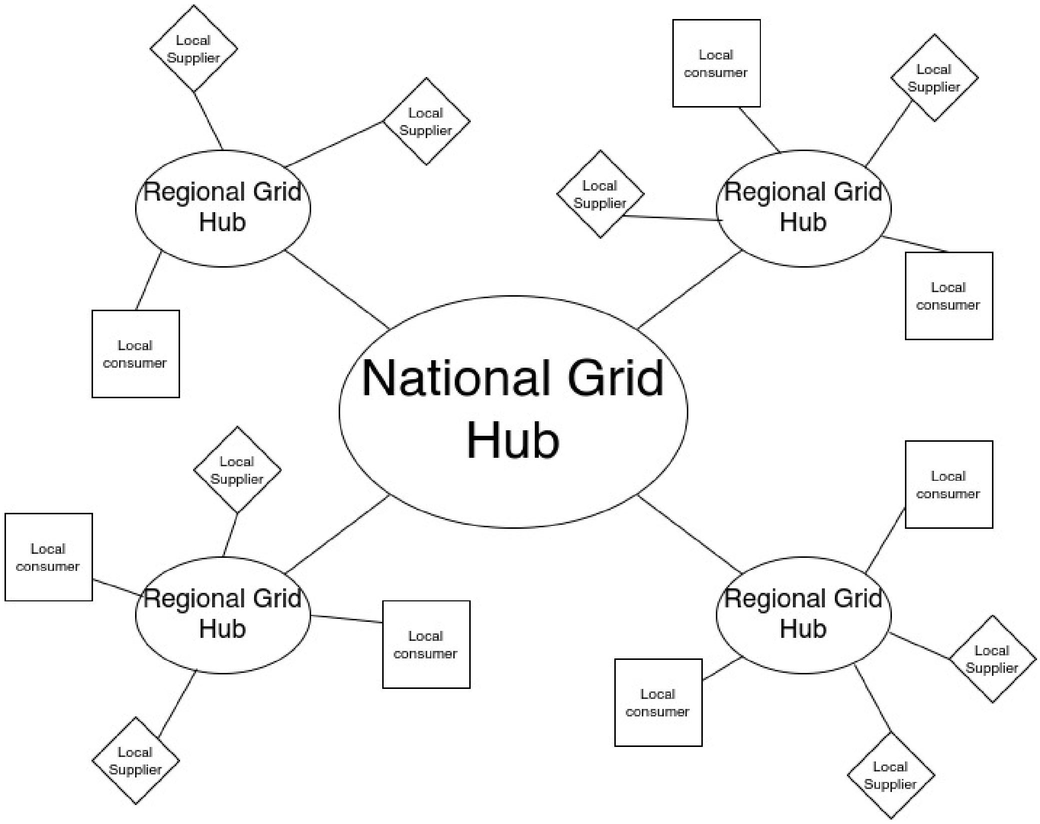

Figure 1.1: Each local consumer and supplier is attached to a regional grid hub which manages the allocation of electricity between suppliers and consumers. This is just a simple overview of the idea but conceivably a consumer or a supplier could be connected to two or more different regional grid hubs.

Chapter 2

Smart Grid

2.1 Overview

Due to advancements in networking technologies, and the field of sophisticated decision making technologies, the idea of a smart grid has become increasingly popular. Actors within a smart grid, be they individual consumers or suppliers, or groups thereof, can be fitted with small computers that perceive changes in the grid and then these actors can react accordingly. Several different types of management systems have been constructed in order to successfully, fairly and efficiently allocate resources for each of these different types of actors. The two primary types of management systems that were examined as part of this final year project were Auctions and Game Theory which will both be discussed in detail later on.

The smart grid is not only used in this manner, but has many other potential applications, some of which have been implemented already in several cities and regions throughout the world. These applications include energy consumption or production prediction, scheduling the use of consumers in order to reduce costs of operation, and smart reaction to disruptions or blackouts within the grid to reduce the damage that occurs as a result.

In this project it is assumed that the consumers within the system are outfitted with some kind of prediction technology. An example of such a system has been proposed by Garcia et al [6] where a device tries to time its own operation within a certain time-frame in accordance with when the price of energy is cheapest. It also attempts to predict how much energy the system will consume based on its own knowledge of previous experiences in buying power at that particular time of day, allowing the system to learn over time and make smarter decisions as time goes on.

2.2 Microgrids and Nanogrids

At present smart grids have generally been implemented at the level of microgrids. Microgrids are generally thought of having a consumer be a single house, or perhaps a group of houses, and a supplier being a small wind farm or solar farm, or perhaps a group of these together. Real world examples of these are campuses and industrial estates [7]. In the case of a microgrid, actors within the system are defined in similar terms to those involved in a centralised grid system, meaning that the transition from a centralised grid to the microgrid scheme is a relatively easy one.

An example of a real world implementation is that of the system in place in Japan. Due to the robust nature of the Sendai Microgrid Tohuku Fukushi University following the 2011 disaster of Fukushima [8], the microgrid has garnered ever increasing popularity. When the region was cut off from the central grid, the local generators attached to the local micro grid were able to supply the on-campus hospital with power while repair work was being carried out. This greatly helped the relief effort in the area by providing much needed medical aid to those injured by the earthquake and tsunami. Following this success of the microgrid system, several other developments have been made in creating more microgrids in Japan [9].

The company ENEL has also introduced a smart grid system in the region of Apulia in southern Italy [10] with great success. The system there allows customers to produce and network their own electricity as well as making them more aware of their consumption and any potential savings.

The concept of a nanogrid is much more modern one, having been introduced by Bruce Nordman in 2012 [11]. The nanogrid system is very similar to that of the microgrid system conceptually but is concerned with a much smaller scale. A nanogrid is one that operates within the confines of a single building, generally where each consumer is a single appliance such as a washing machine or an electronic vehicles (EV). Suppliers would also be very small scale perhaps a set of solar panels or a small wind turbine. A nanogrid system could also be adapted to aggregate a number of devices to act as one as a single actor within the nanogrid system, for example all the lights on one floor of a house could act as a single consumer and draw on a shared reserve of power.

Further extensions involve connecting multiple nanogrid systems together, such as having a nanogrid as a sub-node of a microgrid. This would create a hierarchy of distributed grids. This tree could also be adapted into a graph where a parent node in the tree could have multiple children and a child could have multiple parents. Another version of this would be to have a peer-to-peer network, where multiple nanogrids could trade electricity between one another. These will be discussed in more detail in the conclusion.

Chapter 3

REFIT Scheme

The REFIT scheme (Renewable Energy Feed In Tariff) is one of the most common ways in which countries around the world, including Germany, Spain and the state of Hawaii [12], try to incentivise renewable energy sources and suppliers to sell energy into the main grid for consumption by consumers. The primary tenet of the REFIT scheme is to guarantee a fixed price for energy provided by suppliers at particular times of the day. These prices are offered in a non-discriminatory fashion for every kWh produced by the supplier. The prices can be lower or higher based on the type of energy being produced. For example in Germany the price is higher for suppliers of solar energy than for suppliers of wind energy, according to the EU at the time of the writing of this report [13].

The main advantage of this type of a scheme is that firstly it incentivises companies to invest in renewable energy because they know they'll receive a good return on their investment. It also incentivises landowners and homeowners to invest, thereby creating a large infrastructure of renewable energy resources in a relatively small space of time and this has worked effectively in Germany. The payment also eventually covers the cost of constructing the solar panels or wind turbines for regular consumers over a period of 6-10 years [4].

The main downside to the REFIT scheme however is that because it provides a fixed amount based primarily on the type of energy produced and for how long it is being provided, which means that it is not worth it for a supplier to sell if it has a poor supply in reserve for example. In this case, the incentive to sell energy is quite low as selling any energy would drain the supplier of most of its power. Therefore a scheme involving a dynamic price model that incentivises all suppliers at all times to contribute to the demand and maximise their own utility in the system might be better.

Chapter 4

Auctions

4.1 Overview

The first type of node management systems considered as part of this project was that of auctions. Auctions generally have a number of different types of properties [14] and as such, can be classified into different groupings, including:

- Single- or multi-dimensional
- One- or two-sided
- Open-cry or sealed-bid
- First- or k th-price
- Single- or multi-unit
- Single- or multi-item

While all of these are examined in detail in the book by Simon Parsons, only the continuous double auction will be discussed here as it the only type of auction that was deemed suitable. The reasoning for the decision is explained in the next section along with a description of what the method itself entails.

4.2 Continuous Double Auction

The idea of a double auction is a simple one. Instead of trying to match multiple bidders to a single seller or multiple sellers to a single buyer, a double auction is where there are multiple sellers and multiple bidders.

Through combining the buy-side and the sell-side of an auction into a single process, we then have a two-sided or double action.

A continuous double auction is an extension and a refinement of a double auction where multiple rounds are conducted until as many bidders and sellers have been satisfied as is possible. The first stage attempts to match up as many bidders and sellers as possible who have compatible bids. After that both the sellers and the bidders attempt to adjust their respective ask and bid prices and then another round begins. This process continues iteratively until either all actors involved in the auction are satisfied or until all remaining actors have reached their respective buying or selling thresholds.

The reason why this particular style of auction was chosen to be investigated was that it matches the real world scenario of having multiple consumers within a nanogrid environment as well as multiple suppliers and as such proved to be a popular choice among many proposed auction based solutions to the smart grid problem [15]. It is also reasonable to assume that some kind of memory might be built into the consumers and suppliers so that they might remember what each other offered on previous occasions and submit bids in order to be accepted quicker. The iterative style of the continuous auction was appealing and realistic due to the nature of managing the bids and sales of so many different actors within one given system.

However, most of the auctions investigated as part of this project required the central controller having access to all the private information of all the other nodes. This, among other reasons, led to auctions not being implemented for this project and this will be discussed in further detail later.

Chapter 5

Game Theory

5.1 Overview

The field of game theory is one that has many different facets and versions depending on the situation in which this is used. In this section the nomenclature and jargon of game theory will be discussed, as will a short explanation about the decision to select the type of game implemented as part of this final year project. First the two main types of interactions between players in a game will be discussed and after that the two primary types of playing styles. However, before this, certain traits that are universal for any type of game that must first be explained in order to grasp the concept of game theory enough to understand some implementation decisions later in this report to grasp the general concept of game theory itself.

The concept of utility is an important one to grasp in game theory. Each player that takes part in the game has a utility function associated with it. This function takes the variables involved within the game as parameters and generally results in some kind of scalar value. This value, or utility, can be considered as the score of a particular player in a given turn of the game. In a game that involves money, such as the one proposed in this project, the utility function is modelled as the potential profit that a player can earn when offering different amounts of the resource they are selling. This profit is weighed against any potential loss that the player can incur by offering too much of the resource in a given turn.

In game theory, players within a game compete for a finite resource with the objective of maximising their own utility within the scope of that game. Each player within the game has an associated utility function that is generally the same for all players within that game. The player strategises to try to reach some maximum value for their utility function, on either an individual or collective level. There is generally some kind of manager node also involved, which helps to conduct the game between all of the players involved. Within any particular game, the players are all trying to maximise their own utility. However in different types of games they may also be conscious of the utilities of all the other players involved and try to react accordingly, whether to further their own goal or to further the goals of the collective group.

A well defined game has some form of state of equilibrium. This state of equilibrium is when the sum of utilities of all the players within the game reaches a maximum. The central managing node, if there is one, generally decides whether or not this state has been reached. This state is the success state of the game. In a well-designed game the utility function must be designed such that the state of equilibrium, that is the success state, not only can be reached but also that reaching that state is appealing to all players within the game.

5.2 Non-Cooperative Game Theory

Non-Cooperative games are the simplest types of games both to understand and design. The core component of a non-cooperative game is that all of the players are operating purely independently while trying to maximise their own utility. Each player within the game knows the best strategy to take in order to maximise their own utility. Because every player in a game has the same objectives and strategies available to them, each player knows what strategy will maximise its own utility, based on everyone else trying the same technique

This is where the concept of equilibrium comes into play. Equilibrium is the state in which there is the least disparity between the best player and the worst player, that is that each player performs the best that it can with the knowledge that all other players are similarly going to try to maximise their own utilities. In non-cooperative games, the term ”Nash Equilibrium” is used to mean equilibrium [16].

A simple example of a non-cooperative game is thought experiment called the prisoners' dilemma [17], where two prisoners are face with a choice of snitching on the other or keeping quiet. If both keep quiet, then they each receive a prison sentence of 1 year. If they both betray each other then they receive 2 years each. However if one stays silent and the other decides to betray them then the silent one receives 3 years and the snitch goes free. In this game, the players (prisoners) are both trying to maximise their own utility, in this case their freedom and are working purely selfishly. Each player should then try to always betray the other as this has the potential to lead to freedom and if not, then they avoid the worst sentence of 3 years.

With the knowledge of the strategies of the other players in the game, each player is then able to pick the strategy that maximises its own utility, taking into consideration that all other players are trying to do the exact same thing and therefore it picks an appropriate strategy. In a well designed game, there should also be no incentive for a player to change their strategy to try to undercut other players. If made correctly, such an action would have an adverse effect on the player in the game. In this case all other players would then be aware that this player's strategy had changed and would then react accordingly in order to maximise their own utility and decrease that player's utility.

5.3 Cooperative Game Theory

Cooperative game theory shares many similar traits with that of non-cooperative game theory as outlined in section 5.1. However the defining feature of cooperative game theory is that players within the game will form coalitions based on threats and incentives that occur between each other. The key component of cooperative game theory is the analysis of which coalitions are likely to form within any given game and what the projected outcomes are based upon these permutations of coalitions. In this way the study of cooperative games have two main facets. Firstly, they are concerned with what might cause different groups of players to act together in unison. Secondly they are concerned with the most likely outcomes of each of these games that happen when different groups form.

In this project, the nodes involved in the game are all of the energy suppliers who are trying to maximise their own profit based on the amount of energy that they are able to sell. The utility functions of the nodes and other details will be discussed later in the Implementation section of this report. The desired outcome of each player is therefore entirely selfish and because they are all trying to compete for a finite price, they each want to obtain as much of that money as possible. Therefore it does not follow to design this game in such a way that these players should be able to form coalitions, as any coalition would involve compromising and receiving less money which doesn't make sense in this game. Similarly due to the lack of communication between the players in the game, they can also never know if other players could change their strategies so are unable to even realise that cooperation is even possible at any given stage.

5.4 Cournot and Stackelberg Games

Cournot and Stackelberg games are two manners in which players participate in the game, in other words they constitute the structure of the game as opposed to how players react to one another and strategise within the game. Both of these are relatively easy concepts to understand so this section should be quite short. Because these different structures of games affect the way in which a player interacts with the other players in the game, different strategies can be better or worse based on whether the game is a Cournot game or a Stackelberg game and in some cases some strategies may not even be possible within different game structures.

A Cournot game is simply where all the players make their moves at the same time. For example, all players may submit their moves separately to a central manager node who then reveals all of the different moves at the same time and tries to work out and resolve all the different collisions and determine what exactly the outcome of the game was on that particular turn. In a Cournot game, the players all have to predict what the most likely turn of all the other players are and react accordingly for every round of the game.

A Stackelberg game is where there is a leader within the game who plays first, attempting to maximise its own utility first and then all other players in the game play in turn after that and are able to see the moves of all other players before them. Obviously in this kind of a game, where players are competing over a finite resource, whoever plays first has an immediate advantage over the over players in the game. This trickles down through the game, so that while any given player has a disadvantage compared the whoever had the preceding turn, they have a distinct advantage over all players who come afterwards.

The reasoning behind choosing a Stackelberg game over a Cournot game for this project will be discussed later in the Implementation section of this report.

Chapter 6

Optimisation Techniques

6.1 Overview

Optimisation techniques are an important part of the field of mathematics and are reasonably simple to understand, but can be extremely difficult to formulate. Optimisation problems concern themselves with a key problem that is relevant to many different fields of engineering and computer programming.

For a function $f\colon A \to \mathbb{R}^n$ for a particular set A, an optimisation problem is concerned with finding an element x_o of A where $f(x_o) <= f(x)$ for a minimisation problem or $f(x_o) >= f(x)$ for a maximisation problem, $\forall x \in A$. These optimisation problems manifest themselves in countless fields from economics [18], civil engineering [19] and of course as part of the smart grid [20]. The optimisation techniques involved in this particular project are used on each of the two utility functions involved in the process namely that of each of the game players and then the moderator actor process involved in the system. This will of course be discussed in more detail later on.

One of the main benefits of an optimisation technique is that it is often obtainable using linear algebraic methods which means that a computer can figure out the solution to the optimisation problem in polynomial time. Another benefit of this is that an optimisation technique can be used in tandem with any other problem solving technique in order to find a better solution much faster. If any problem fits the parameters of the optimisation as defined above then different optimisation techniques can be applied or at least the same one in multiple places.

While the basic premise and motivation behind every optimisation technique is the same, different types of sets of values can be used for the set A and as a result. Fortunately, different types of optimisation techniques have been developed in order to more efficiently solve problems in each of these areas. In some cases, the type of values in the set such as in a convex set, actually make other optimisation methods useless. In this project, two main optimisation methods were used, namely Convex Optimisation and Hyperplane Projection

Optimisation. Both techniques are involved with quickly and accurately solving for a maximum in the case of two different utility functions but operate with different types of sets, each one being suitable for the relevant type of problem.

6.2 Convex Optimisation

Convex optimisation is defined as the solving of minimisation problems that involve convex functions being applied to convex sets [21]. Due to the nature of the convexity of the sets involved in these sorts of problems, a term that I will discuss momentarily, the local minimum that is discovered is actually a global minimum. Basically this means that the curve of the graphed outputs from mapping the values of a convex set through a convex function, only has a single minimum as opposed to a situation where the curve could have multiple minimima or values that can be converged on which are not the true minimum of the curve. This property of a convex optimisation problem as well as the property of general optimisation problems of being able to solve the problem in polynomial time means that the true solution can be discovered relatively quickly.

A convex set is simply a region in which, if you draw a line between any two arbitrary points in the region, then all points on the line are also inside the region as outlined in the left side of Fig 6.1. The right side shows a non-convex set where there is a hollow section to the region.

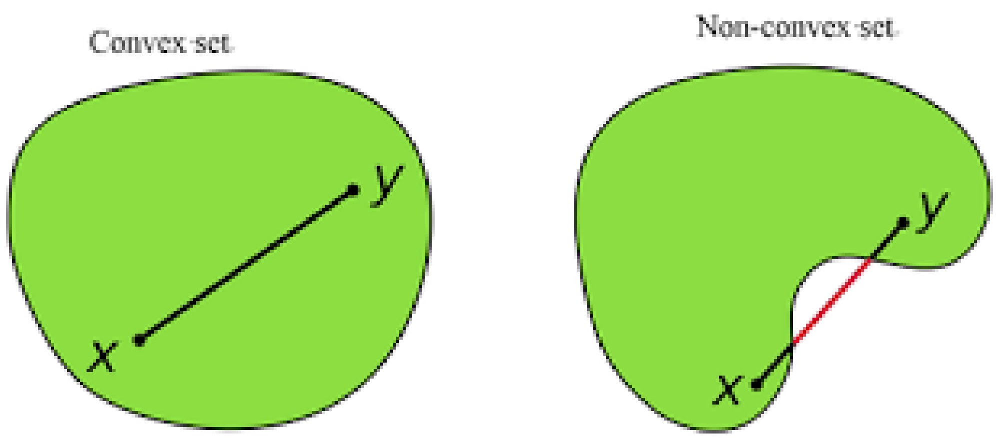

Figure 6.1: A convex set (gtMath March 2016) [22]

A convex function on the other hand is simply a function where the entire line segment between any two points on the graph is above or on the graph. This is the part of convex optimisation that determines the fact that the local minimum is a global minimum. Convex functions are extremely common in the field of mathematics such as the quadratic function x^2 and the exponential function e^x.

Convex optimisation is therefore a relatively simple concept to understand and is clearly seen to be a very useful and efficient method of accurately and quickly finding solutions to minimisation problems.

6.3 Hyperplane Projection

6.3.1 Variational Inequality Problem

The hyperplane projection method is a tool for solving problems that suit the criteria of a variational inequality problem (VI problem) so first that must be explained before moving onto the concept of the solution to such a problem.

One of the components of the VI problem [23] is that of a functional. A functional is a function that maps a vector space onto its underlying field of scalars. A scalar field is where there is an associated scalar value to every point in a space, in this case to every point in the vector. Often this vector space can be a series of functions, meaning that the functional takes a function as an argument and can be interpreted as a function of functions. This is similar to the idea of higher order functions, where a single higher order function can be used to operate on multiple functions and perhaps capture some other important piece of data for a given system.

A variational inequality is an inequality that involves a functional that must be solved for all variables in a set, usually a convex set. As a side note, although this problem also involves a convex set like the convex optimisation problem, the functional is not a convex function and therefore convex optimisation does not apply in this instance.

The origin of, and primary application of, variational inequality problems is in the field of finding solutions of equilibrium in a given system. As we'll see later on in the implementation section of this report, finding the state of Nash Equilibrium between the different suppliers that take part in the game requires a state of equilibrium. Therefore it can be easily inferred that the variational inequality problem is applicable and the problem can be solved as such using a method appropriate for such a problem.

The hyperplane projection method defined here also stipulates that the underlying functional involved in the problem must meet a certain monotonicity criteria. Monotonicity is a property of a function that says that the function must either be non-decreasing or non-increasing. The function does not have to be constantly increasing or decreasing but for example if it is increasing then it cannot decrease or vice versa in order to be deemed monotonic. This can be represented mathematically as $f(x) <= f(y) \forall x <= y$ or $f(x) >= f(y) \forall z <= y$. Functions that cleave to this mould are called monotonically increasing and monotonically decreasing respectively.

6.3.2 Hyperplane Projection Method

Having covered a number of the prerequisites for using a hyperplane projection method, the method itself can be explained. The version I looked at was developed by Solodov and Svaiter and is called the Solodov and Svaiter Hyperplane Projection Method (SSHPM) [24]. Figure 6.2 will be referred to as a part of the explanation.

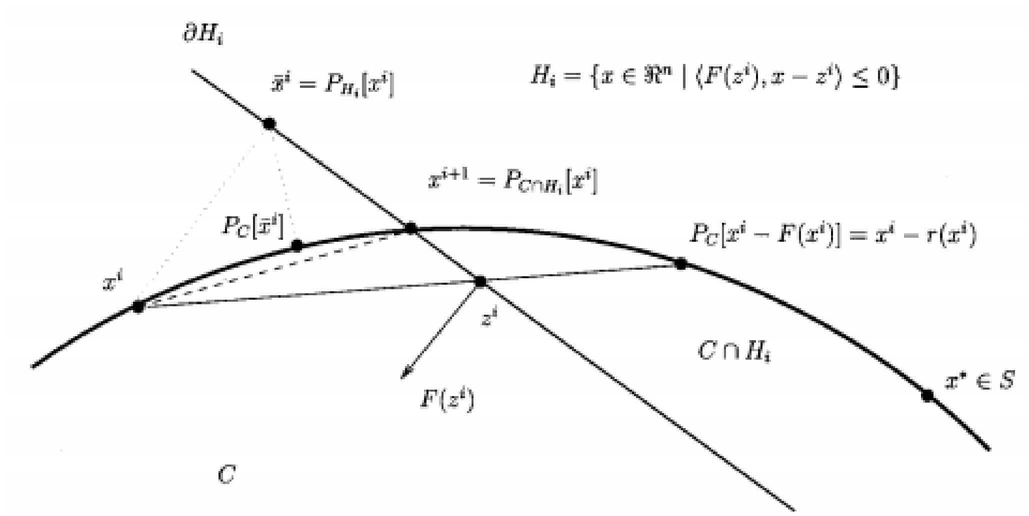

Figure 6.2: Solodov and Svaiter Hyperplane Projection Method

The curve in the figure describes the functional in the variational inequality (VE) problem. This method uses the projection operator $P_C[x] := argmin||y - x||$ where $y \in C$. Suppose we have a point x^i which is the current approximation of the solution to the VE problem involving the set C and the functional F. First we calculate a a projection point $P_C[x^i - F(x^i)]$. The segment between x^i and $P_C[x^i - F(x^i)]$ is searched for a point z^i, using a linesearch method like the Armijo linesearch method [25], such that a hyperplane δH_i (using the definition of H_i as defined in figure 6.2) strictly separates x^i from any solution x^* of the problem. The next approximation to the solution x^{i+1} is calculated by projecting x^i onto the intersection of the set C and the halfspace H^i that contains the solution set using $P_{C \cap H_i}$.

The benefit of this solution is that each iteration of the method only requires two projections which makes it computationally efficient, the first to calculate the hyperplane H_i and another onto the intersection $C \cap H^i$ to find the next iterate in finding the solution.

A good analogy for the hyperplane projection method is that of the binary search method. The space is divided in two and it is determined as to which half the solution resides in. That half is then searched in the same manner until the solution is eventually converged on. The difference with SSHPM is that it operates in n-dimensions whereas binary search only works in one dimension.

In this smart grid scenario, the value being searched for is that of the amount of energy that a particular supplier will provide to the system. The system tries to find the ideal value for the amount of energy to offer to the system based on the current incentives. Later on in the Implementation section, the application of this method will be discussed in further detail.

Part III

Implementation

Chapter 7

Design

7.1 Games vs Auctions

In the background section of this report both the concepts of Auctions and Game Theory as both were considered as potential candidates for the management system to match supply and demand in a nanogrid system. Ultimately however, a non-cooperative game was chosen as the prime candidate for the smart grid in this project. It is important to first consider the reasons as to why this choice was made before explaining how the game was designed.

In the process of investigation of auctions and game theory, certain similarities stood out between the two management systems. Ultimately all actors within either of these systems are trying to maximise their utility, a scalar value that is determined based on a number of key variables that each actor considers pertinent to their operation. In the case of a model such as this one, where a price value is involved, the utility of any given actor is usually modelled as a balance between any profit that the unit could make versus some kind of risk factor of selling too much at any one given time. In this regard, the modelling of any actors within the grid would end up being the same on a conceptual level and only the interactions between them would change based on what kind of system was chosen.

As has been outlined in previous sections, one of the main criteria for the nanogrid system, was that of minimal sharing of information between actors in the grid. This was to decrease the size of packets exchanged between nodes in the network as well as to hopefully decrease the number of packets sent between each other in order to improve the efficiency of such a system such that it might be practical for a real world scenario. Therefore the focus was on a system that would fit this design. Every auction that was investigated as part of this report had a crucial element of either all nodes being aware of the each others' private information or at the very least the central node needed to have all this information to hand. Therefore a non-cooperative game seemed more appropriate based off this particular design.

7.2 System Structure

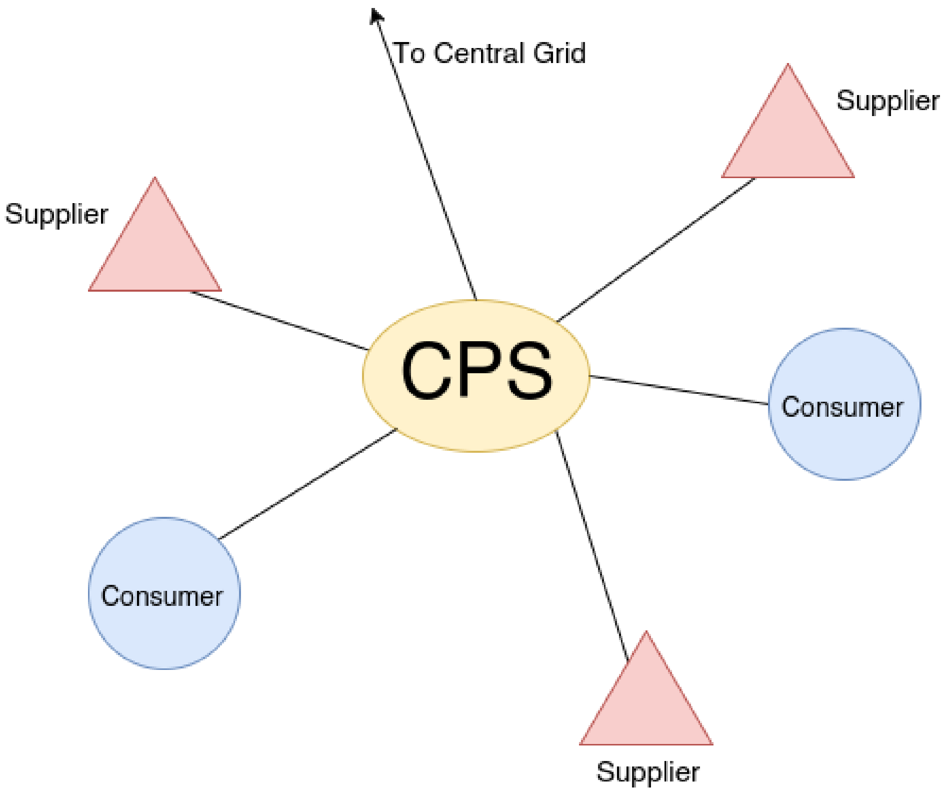

Figure 7.1: Simple diagram to understand the connections between the different actors involved in the system in a given iteration

Figure 7.1 above defines the connections that exist in the network in a given iteration of the system. This section introduces the different kinds of entities involved in this proposed scheme, namely the ECs and CPS.

The Central Power Station (CPS) is the central manager unit in the system. It is responsible for processing all of the energy estimates given by the ECs. It is also the entity that controls the amount of money supplied to each of the ECs. In a more formal sense, the CPS is the manager unit of the game conducted in the system and as such decides when the game has reached the state of Nash Equilibrium. Should the game be unable to provide appropriate supply to the demand specified by the system, the CPS is also connected to the central grid and is able to purchase extra power in order that the demand is still met.

The Energy Consumers (ECs) are the appliances and energy suppliers within the system. They were modelled here as the same entity and at the start of a timeslot in the system operation they notify the CPS as to whether they are consumers or suppliers. This decision was made to facilitate the situation where an appliance might have bought too much energy in advance and wants to sell its excess. It also facilitates extensions to the system where a nanogrid could buy or sell to and from other nanogrids, where a nanogrid could be of either type.

7.3 System Design

In this section I will discuss a brief overview of the operation of the system implemented in this project. Below in Figure 7.2 is a basic flowchart of a single iteration of the operation of the system, followed by a brief summary of each step. The summary below assumes that all the nodes within the network have connected with one another already, although in my code submission there is an extra step to ensure that the system process doesn't start until the user decides that it should so that the system can be monitored on a step by step basis. Figure 7.1 is a simple diagram of the connections between different actors within the system.

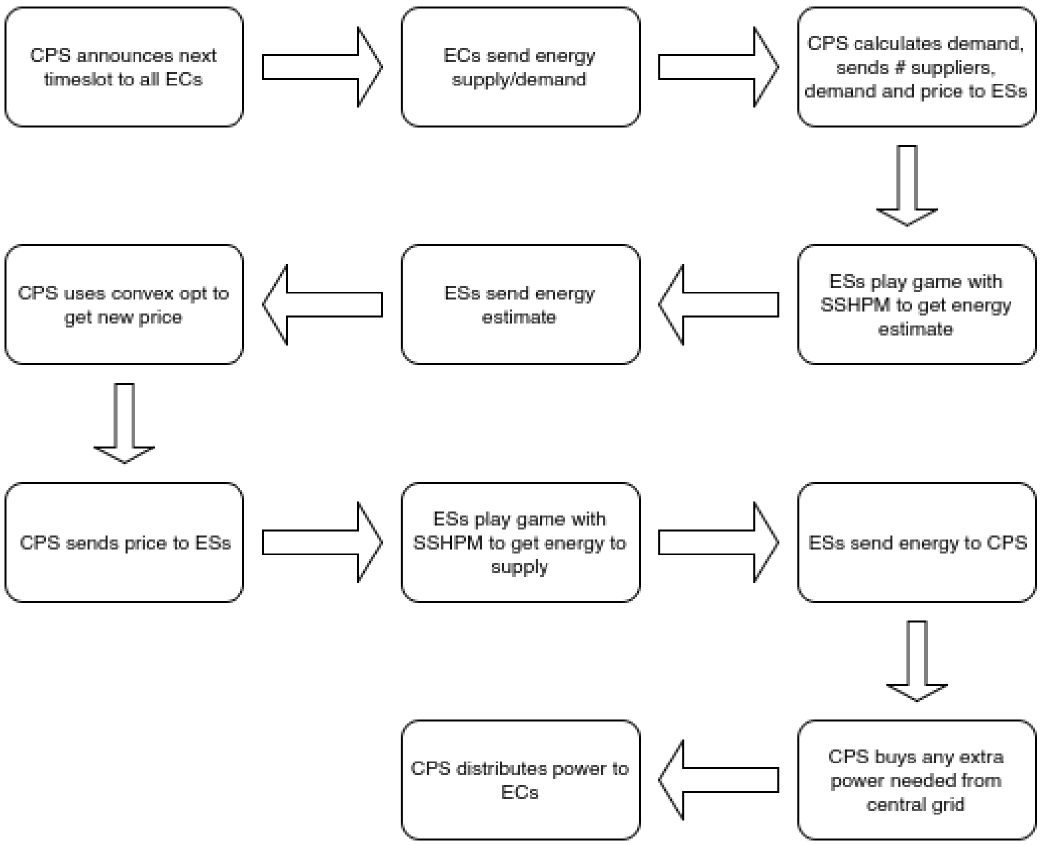

Figure 7.2: Flow chart depicting the operation of the system in terms of the Central Power Station (CPS), Energy Consumers (ECs) and Energy Suppliers (ESs) in a single iteration

The objective of the proposed system is to ensure that for each timeslot, the sum of utilities of the ECs is maximised, that is that each EC gives an appropriate amount of energy to the system based on the price incentive offered by the CPS. When the sum of utilities is maximised, the energy deficit for a given timeslot is more likely to be met and ECs are more likely to be able to continue to provide power into the future, as they will only deplete their power supply if the incentive to do so is very high.

An iteration of the system is conducted to match supply and demand for in a nanogrid situation for a given upcoming timeslot. Some kind of system where a consumer can predict their energy usage for the next timeslot is presumed to be in place. The suppliers of course know what their own supply of energy is as well as having a caution level. The caution level determines how willing they are to sell larger amounts of energy, a low caution value representing a willingness to sell more energy and a high value standing for a more conservative supplier.

The operation begins with the Central Power Station (CPS) announcing a timeslot to all consumers and all suppliers within the network. At the beginning the CPS doesn't know who is a consumer and who is a supplier in order to accommodate the situation where a consumer has proactively bought too much energy in anticipation of needing it or has been instructed by some logic to sell excess energy into the grid. Each Energy Consumer (EC) then notifies the CPS as to whether it is in need of energy or whether it has energy to sell and if it's the case of the former then it also sends how much energy it requires. Figure 7.1 shows the situation where ECs within the grid have already made it clear as to whether they are a supplier or a consumer for this particular timeslot.

For each timeslot the CPS has a fixed price per unit (per kWh) already associated with it before the game begins. This can be determined using some kind of real time price estimator such as the one proposed in [26]. When the game begins the "total price" P is determined by multiplying the total demand of the timeslot by the current price per unit. This total price is the finite resource for which the ECs are competing in the non-cooperative game. If the CPS was simply buying energy from the central grid, this is the same amount of money that it would be paying out. However, in this system, instead of paying the central grid, the CPS is paying the ECs for the energy they can provide and they are trying to get as much money as they can for the energy they can offer while maximising their utility.

The CPS then simply sums the total demand and can begin the game. It sends the total demand, the total amount of money it has available to give, also called the "total price" P, and the number of suppliers within the system to each Energy Supplier (ES). The total price is calculated naively by multiplying the current price per unit that is offered by the central grid by the number of units of energy required by the consumers within the nanogrid. A standard unit would be kWh. Each ES first calculates how much energy it can be offered by dividing the total price by the number of players. Each one then uses the SSHPM optimisation method to determine an estimate for the energy it is willing to give to the CPS at that price and sends that estimate to the CPS. The functional used as part of the SSHPM is the utility function of each EC and the set of values being mapped over is a one dimensional vector space that goes from zero to the total amount of power currently stored by each EC, that is the amount available for them to give.

The CPS then receives each ES's energy estimate. From this it is able to estimate how willing each ES is to giving more or less energy. It cannot work out the private store or the caution of each ES but rather understands the ratio that exists between all the different players involved. Each ES is thus able to keep their private information private but, using this estimate the CPS can infer the amount of energy that each ES is willing to give and can offer a higher incentive accordingly to those who have less to give and a lower incentive to those who have more to give. This maximises the utility of all ESs involved in the game. The CPS uses its own utility function and the vector of energy estimates from each ES as the inputs to a convex optimisation problem. A disciplined convex optimisation method is employed [27] as any standard convex optimisation technique is all that is required and the Python solver CVXPY [28] was readily available. A new vector of prices per ES is generated and each one is sent to each ES. This is the actual price that each ES receives.

The ESs then play another game using their utility functions and the new price that they have been offered by the CPS and try to find the actual amount of energy that they are willing to give away using SSHPM. This energy is then sent to the CPS. The CPS then sums the total of energy that has been provided at that time. If this energy matches the total demand of the consumers in the nanogrid, then the energy is simply supplied to those who need it, on a first come first serve basis. However, if the supply does not reach the demand then the CPS buys the extra power that is needed from the central power grid as seen in Figure 7.1. This system accepts the fact that it may not be able to supply all consumers within the nanogrid using solely local sources that exist within its own grid. Once the supply matches the demand, the power is then distributed as before. The process then starts again ahead of the next timeslot to ensure that everyone that needs power during that time is supplied.

At the end of each iteration of the system, the demand of the consumers within the system has been met without compromising the utilities of the the devices that supplied the energy. This means that no suppliers have been left with no energy without receiving proper compensation for it, meaning that they are more likely to have more power to sell later on when perhaps the demand might be higher or the supply of other ESs in the system might be lower. It must be noted that this system does not attempt to drive down the price for power but rather to distribute that money in such a way that the demand is more likely to be met within the local grid and to increase the chances of power continuing to be supplied in this fashion.

7.4 Game Design

First some of the key components of the game as well as a brief overview of how it is conducted will be explained. Following that, the game itself will be discussed in further detail. The game played between all of the ESs that are trying to receive remuneration for the energy they are willing to offer is played across two steps. First of all the ESs use their utility functions along with a number of other important variables such as their energy capacity E_n, caution c_n and the current price offer p_n in order to determine their new estimate for how much energy they are willing to offer to the CPS e_n, where $n \in N$, N being the set of all ESs taking part in the nanogrid.

Table 7.1: A description of each of the variables involved in the game

Variable	Description
e_n	The amount of energy the ES is currently offering (the variable that changes on each iteration)
E_n	The total energy stored by the ES. This is the maximum amount of energy that it is able to sell
c_n	The caution variable of the ES where $c_n \in (0, 1)$. This determines how cautious the ES is
ε_n	The slack variable of the current iteration. Used to determine Nash Equilibrium by the CPS

Next they use that energy estimate to calculate a slack variable ε_n which is a variable indicating the amount of energy it is willing to offer without giving up any private information. These slack variables are derived from the ES's utility functions which will be discussed in the next section. The slack variables are used by the CPS to determine Nash Equilibrium within the game, namely this is when all of the slack variables are equal. Once this state of equilibrium is reached, then the CPS asks for the energy offer from each of the ESs.

When the hyperplane projection is initially calculated there is a small piece of logic that determines what slack variable is sent to the CPS as well as what energy should be offered. If the hyperplane projection $P_C[e_n]$, as defined in section 6.3.2, is equal to zero then $\varepsilon_n = E_n - 2c_n e_n + p_n$. Otherwise the second part of the hyperplane projection method is run, where the halfspace is determined and from that a new projection is worked out. In this case the slack variable sent back to the CPS is $\varepsilon_n = E_n - e_n + p_n$. These slack variables are then sent to the CPS. If the slack variables are all equal, as previously mentioned, then the game has reached the state of Nash Equilibrium and the ESs are informed to end their iterations and they instead send back the amount of energy they are offering. If the slack variables are not equal then the CPS instructs the ESs to perform another iteration of the SSHPM.

7.5 Utility Functions

7.5.1 EC Utility Function

Each EC has a utility function that is used as the functional in the the hyperplane projection optimisation. The utility function in question takes into account the energy that EC n has stored E_n, the price being offered to it p_n, the caution value of that EC c_n and the energy that it is offering e_n.

$$U(e_n, E_n, p_n) = p_n e_n + (E_n - c_n e_n)e_n$$

This utility function is based on the profit that the EC could get when it is supplying energy, that is $p_n e_n$. $(E_n - c_n e_n)e_n$ represents the loss that the EC incurs by giving away a certain amount of power. Ultimately the system is trying to maximise the utilities of all ECs in the nanogrid, where the sum of all offered energies is less than or equal to the energy deficiency (demand) of the system for a given timeslot E_{def}, that is

$$\sum_n e_n <= E_{def}$$

.

The utility function defined for the EC is the the crux of this project in order to both structure the game itself and to determine the efficacy of the system. The utility function is defined such that each EC is better utilised for each timeslot but also does not expend too much electricity at one time unless the incentive, namely the price, for it to do so is very high. This means that at a later stage when there is perhaps a higher deficit, it can make more money in the future as opposed to potentially being depleted of energy for the times of high profit.

7.5.2 CPS Utility Function

The CPS has its own utility function that serves as the convex function for the convex optimisation problem in trying to find appropriate prices for each of the ESs that have submitted energy estimates for how much they are willing to offer. The function is represented as a minimisation problem in terms of the energy that each ES is offering e_n, the price that the CPS would offer for that energy p_n and two scalar values a_n and b_n that account for the costs associated with storing and transmitting the energy respectively.

$$min_p L(p, e) = min_{p_n} \sum_n (e_n p_n^r + a_n p_n + b_n), where \sum_n p_n = P, p_{min} <= p_n <= p_{max}$$

For each ES, the CPS is trying to find the value of p_n that will give the smallest value for the function L. However all values of p_n must sum to be equal to the value of P, the total price that the CPS is willing to pay. As can be seen in this model, the system doesn't pay any less for power overall, but rather incentivises all suppliers of electricity to try to match the demand in question. Another caveat of the minimisation problem is that p_n must be between the values of p_{min} and p_{max}. This simply means that there is a minimum and a maximum value that the CPS is willing to pay for energy.

Chapter 8

Application

8.1 Python Twisted Framework

In the course of this project the Python Twisted Framework [29] was used in order to implement a network between the CPS and the ECs. The Twisted Framework has a number of layers in order to abstract out the problem for the user so that they only need care about their own application [30]. It also has a number of inbuilt functions so that the programmer does not have to care about things like sockets that are very tricky and are far removed from the problem trying to be solved in this project.

In Twisted, both the Client and the Server have two main layers, the Factory layer and the Protocol layer. Essentially the Factory layer contains all of the persistent information of any given network actor and the Protocol layer contains actions and information for every connection made by that actor. In the code produced as part of this project, the factories of the CPS and the ECs were mostly used to store the values of variables pertinent to each of them such as the price vectors for the CPS and the energy storage for each of the ECs.

The majority of the logic that controls both the CPS and ECs was contained within separate Finite State Machine (FSM) files which were connected to their respective Protocol files the FSM only changes state based on the inputs it receives from any given connection. It was easiest to abstract the problem out in this fashion for ease of reading and understanding of the code for both the programmer and any potential readers. Each EC and the CPS is finally wrapped by a simple run script that just sets up the factory from which everything else is run. In each section below, the FSM of both the ECs and the CPS will be examined as the system is conducted in the same fashion for both, where the protocol calls a different function in the FSM depending on the state of the actor at that time.

The FSM logic is not a part of the Twisted Framework but was a separate class that was attached to it in order to better encapsulate the actions required at each step. Figures 8.1 and 8.2 respectively describe the

Finite State Machines of the EC and CPS while Figure 8.3 describes the interactions between the CPS and EC entities.

8.2 Client (EC)

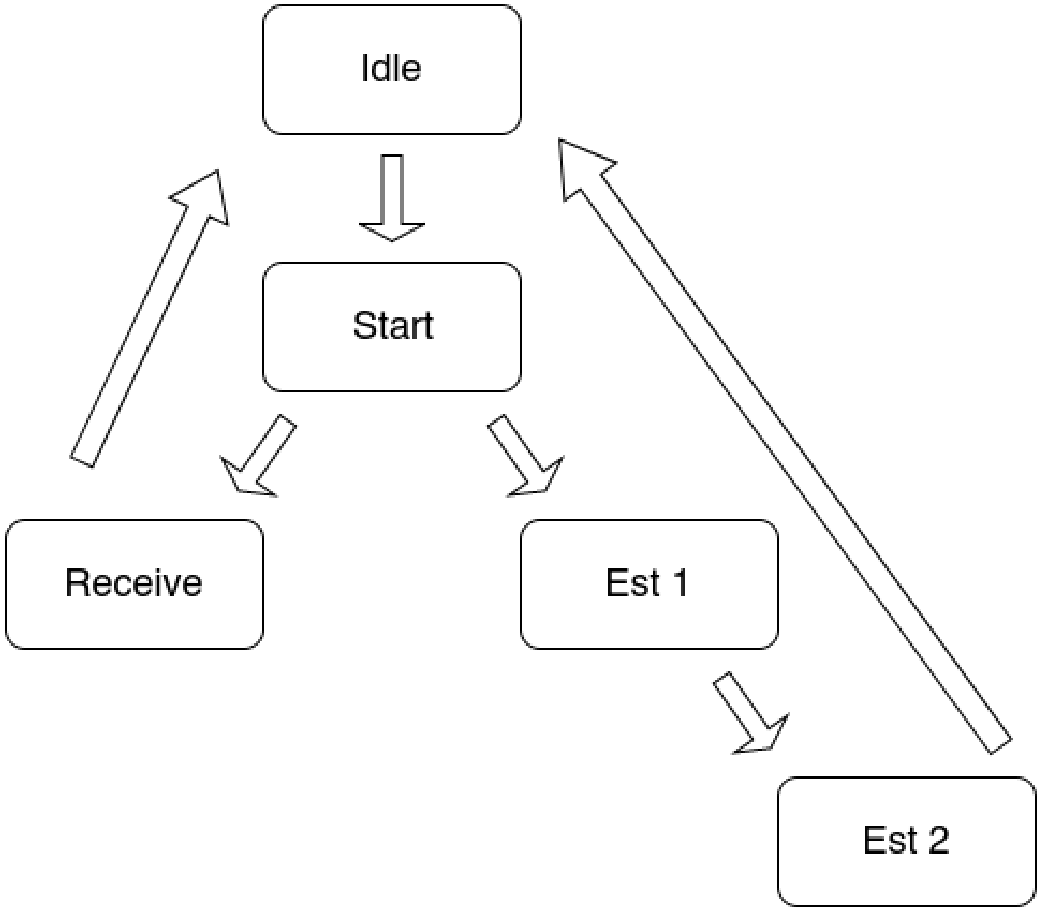

Figure 8.1: Finite State Machine for an EC

8.2.1 Idle State

The idle state is merely a state for in between operations of the system, where no game is being played. In this project it was also used as an initial state before the user decides to start the game. When a new game is started by the CPS, it notifies each EC to move into the start state.

8.2.2 Start State

The start state is where the EC sends a message to the CPS to inform it as to whether it is a supplier or a consumer for the upcoming timeslot. If it is a consumer then it also sends the amount of energy that it requires at that time. Also if an EC in the nanogrid requires no energy for the next timeslot, then it simply puts itself back into the idle state, awaiting the next timeslot when it might need or be able to supply energy. An EC moves to the Estimate 1 State if it is a supplier and to the receive state if it is a consumer.

8.2.3 Estimate 1 State

This state is used for when the ESs are playing the game and making their first estimate of how much energy they are willing to offer to the CPS. In this state, if an ES is told to ”End” its iterations then it moves to the second estimation state (Estimate 2 State). Otherwise it uses the hyperplane projection method solver (SSHPM Implementation) that was developed as part of this project and sends a slack variable to the CPS, used in determining Nash Equilibrium for the game.

8.2.4 Estimate 2 State

The Estimate 2 State is more or less the exact same as Estimate 1 State except that when it receives the ”End” message, it instead moves back to the idle state, having successfully supplied energy to the CPS and having been remunerated for that energy. If it doesn’t receive the end message then it uses SSHPM to calculate a new slack variable and continues playing the game.

8.2.5 Receive State

The Receive State is the state for any consumers for the current timeslot. An EC stays in this state until the operation of the system has been completed and the energy is distributed to it accordingly. Once it receives this energy, it returns to the Idle state in order to wait for the next timeslot.

8.2.6 SSHPM Implementation

The SSHPM implementation caused the greatest amount of difficulty as part of this final year project. The paper [24] details a complex and dense mathematical algorithm that was difficult to grasp and to implement. The functions with the SSHPM.py file follow the steps in the algorithm defined by Solodov and Svaiter.

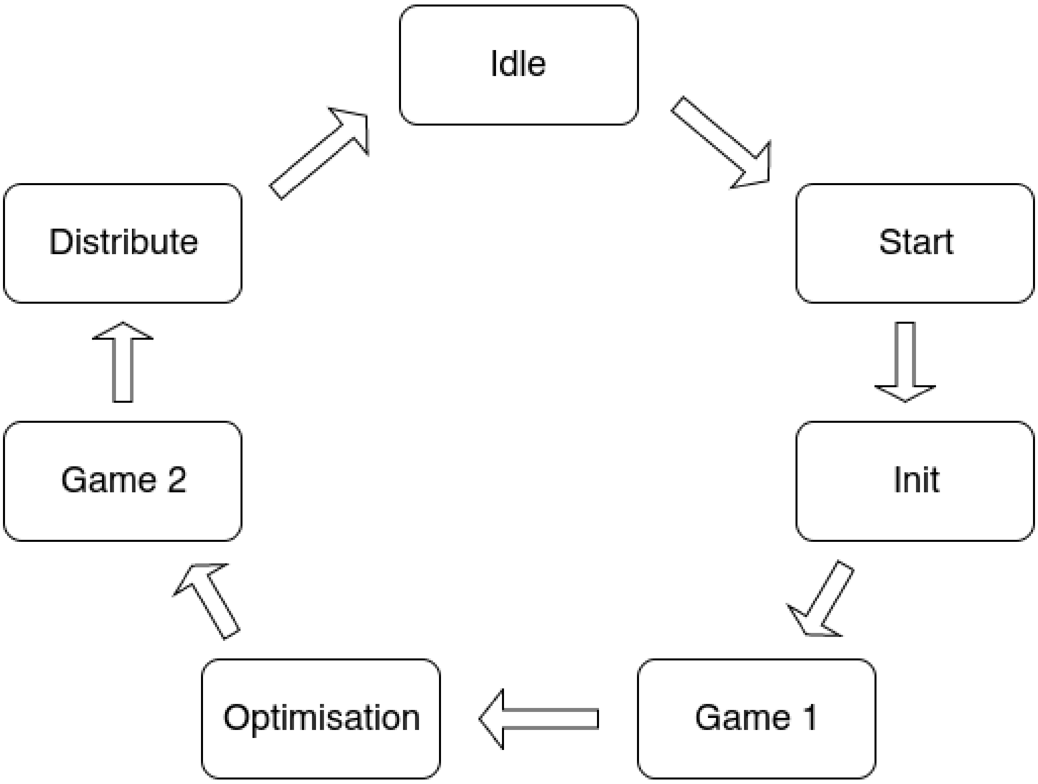

Figure 8.2: Finite State Machine for the CPS

8.3 Server (CPS)

8.3.1 Idle State

The Idle State is used while the there is no game happening and the CPS is idle. The CPS is then able to move into the start state when it wants to begin a game immediately preceding a new timeslot. In this implementation it's also used on start-up of the system so that the operation of the system doesn't begin until every EC is connected to the CPS. Each connected EC is stored in a Python dictionary with its relevant connection.

8.3.2 Start State

The Start state accepts incoming messages from each of the ECs about whether they will be a consumer or a supplier for the current timeslot or if they will be abstaining from the current round of operation. It stores each EC's role in the upcoming game and doesn't move to the next state until every single EC in the network has given an answer as to what their role shall be. The CPS then naively calculates the demand by simply summing the values given by each of the consumers and then moves to the Init State. Each EC also has a timer associated with it so that if the EC dies, then it times out and is not included in the game.

8.3.3 Init State

The Init State is used to allow the suppliers who will be taking part in the game to start the game by providing them with the values that they need. Each EC is sent the energy deficiency (E_{def}) for the current timeslot and price that the CPS is willing to offer to each EC. The price is calculated by multiplying E_{def} by the current price per unit of energy and dividing that by the number of ESs in the system. The CPS then moves to the Game 1 State.

8.3.4 Game 1 State

The Game 1 State is used for the first game that is played by the ECs that are supplying energy for the current timeslot. When all connected ESs have responded with their slack variables, the CPS runs a quick check as to whether or not the slack variables are equal. If they are then it tells the ESs to finish their iterations of SSHPM and to send the energy estimate that they used to calculate the last slack variable that they sent and then the CPS moves to the Optimisation State. If the slack variables are not all equal then the CPS tells the ESs to continue playing the game.

8.3.5 Optimisation State

The Optimisation state first waits to receive the energy estimates from each ES in the game of the current timeslot before beginning the convex optimisation. It formulates the problem and then solves it using the CVXPY [28] solver library. The new prices are then extracted from the solver and the CPS sends the relevant price to each ES before moving to the Game 2 State.

In order to use the CVXPY solver, the problem must first be condensed into a constructor. The problem has two parts, the objective and the constraints. The convex function is the objective, in this case the utility function of the CPS, namely L from section 7.4.2. It is passed a one-dimensional vector that represents the prices that will be offered to each of the ESs for their offers. The other variables such as those energy estimates are also added as appropriate in vectors or as scalar values. The constraints are then specified in an array and in this case ensure that each price does not exceed the minimum or maximum price that the CPS is willing to offer each ES and that the prices sum to the total price that the CPS has to give for the given timeslot.

8.3.6 Game 2 State

The Game 2 State is more or less identical to the Game 1 State except that it is concerned with managing the ESs who are trying to calculate the actual amount of energy that they will give to the CPS as opposed to an estimate. In this state, when the slack variables are equal, it similarly tells the ECs to stop the iterations of their game and then it moves the CPS to the Distribute State.

8.3.7 Distribute State

The Distribute State first makes sure that all ESs have submitted the amount of energy that they are going to provide to the CPS. It then sums these values and compares that to E_{def} and if it the supplied energy is insufficient, then it buys the extra energy needed from the central grid. It then disperses the energy needed to each of the consumers within the current timeslot and moves into the idle state, ready for the next game before the next timeslot.

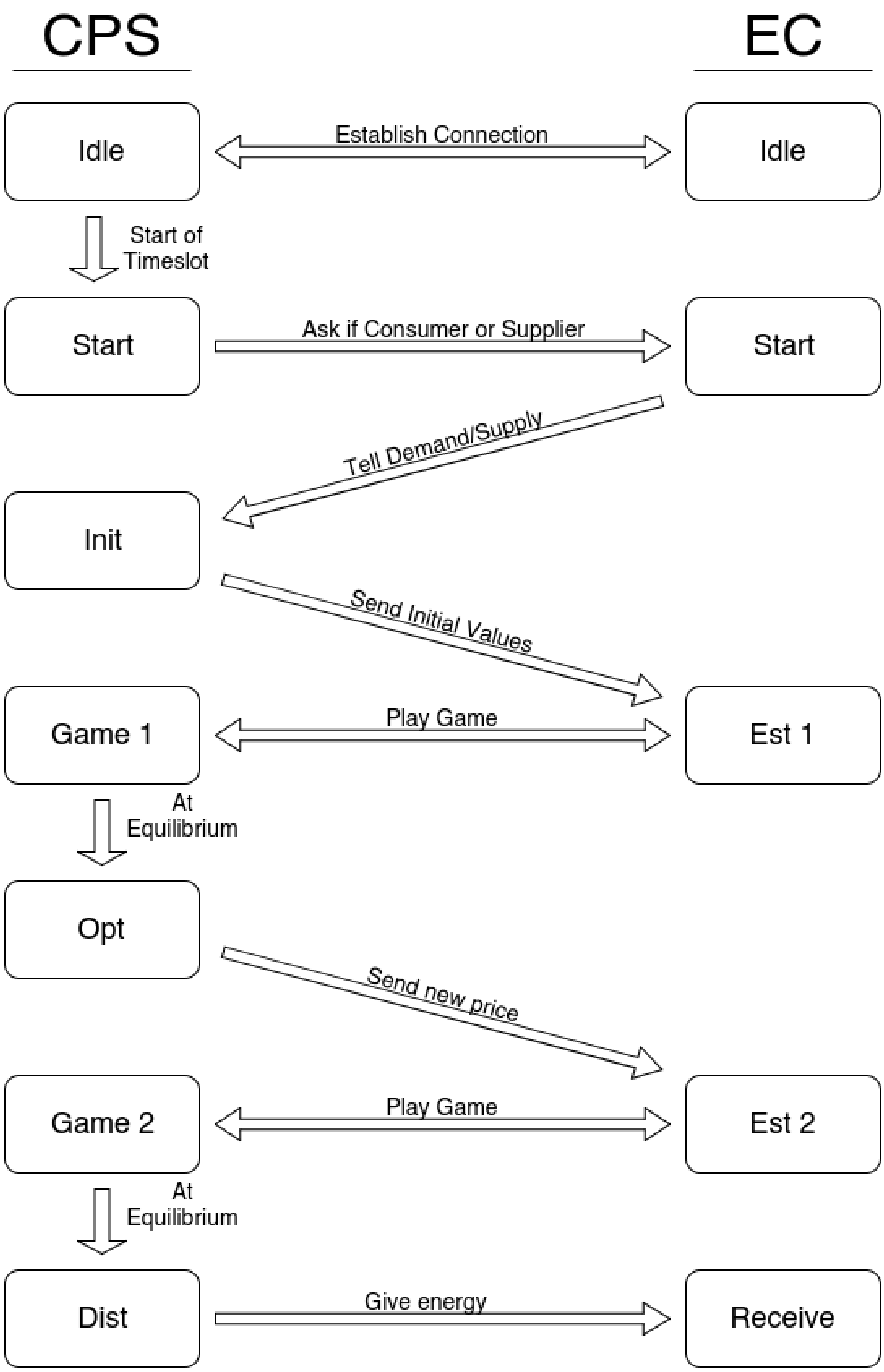

Figure 8.3: Diagram showing the interactions between the states of the two entities

Part IV

Results and Discussion

Ultimately the success of the system was unable to be determined due to difficulties with the implementation of the hyperplane projection method. Instead of converging on a value for the energy estimate, it instead approaches zero for every instance but never reaches it, and therefore runs infinitely. In order to solve for this, a number of lecturers within the school of computer science were contacted for assistance and the author of the paper, Wayes Tushar, was approached. However Mr Tushar was unavailable and unfortunately a proper understanding of the mathematics involved in SSHPM was not reached and consequently a correct code implementation could not be achieved.

However, a pen and paper comparison between the REFIT scheme and the proposed scheme was calculated to demonstrate a potential scenario of operation in order to compare the two systems. The slack variables were calculated on paper and were hard-coded into the current system in order that the code can be run as opposed to running indefinitely. The attempt at the correct solution is still present but is currently commented out. These sections are flagged in the code. This scenario was run between a single consumer and two suppliers. The consumer sets $Edef = 700$ and the two suppliers have $E1 = 1100$ and $E2 = 1000$. The price per unit (kWh) is set at 1.85. When the proposed system is run with these values the sum of utilities $nU_n = 1069618.928$. Compared to the REFIT scheme where the ratio between the capacities is used as a naive method of deciding how much energy should be given by each EC, in a system where they share private information. In this scenario, the sum of utilities $nU_n = 1068167.8$.

Unfortunately, due to the lack of conversion on correct values by the system, no further scenarios could be calculated as it is unclear as to what the values that the optimisation methods would reach in instances where the supply does not meet the demand. Optimisation techniques can often require several iterations before converging on a correct value and therefore are not feasible to be worked out on paper.

Chapter 9

Assessment

From the numerical results it can be easily seen that the proposed method in this instance is superior to the REFIT scheme as the sum of utilities is greater in the former approach. However the difference between the two utilities is minimal so this system may not be the worth implementing in reality. A working solution, however, would verify whether this statement is true or not.

Regardless, another equally important question to answer as part of this project, is about the likeliness of such an system being implemented in the real world. First of all, it must be noted that such a system could only be deployed following advancements in the creation of suitable prediction methods for predicting the amount of energy that a consumer will require within the next timeslot. Secondly, the installation of such a smart grid would require a huge overhaul of the current network if this system was focussed on immediately. Rather, a more realistic approach would be to first introduce smart meters into homes in conjunction with the current grid, like is in place in Italy as discussed previously and to follow that with a REFIT scheme to incentivise the installation and construction of local renewable energy sources as is in place in Germany. At this point it would then be feasible to introduce a system such as the one discussed in this report. The reason why a REFIT scheme would need to precede a game theoretic solution is that on a human level, people would need tangible and static amounts of money to ensure that they would see a return on their investment whereas the game theoretic approach does not yield a concrete and easy-to-grasp amount of money.

Part V

Conclusion

There are a number of ways in which this project could be continued, that were discovered during the course of the investigation of the field of smart grids and in the particular area around which this project is based. These fall under two distinct categories: the first being aspects in which this project could have been extended had there been more time, and the second being ways in which investigations could be made into pairing this project with other proposed smart grid technologies.

Originally it was intended that the project would include implementing a prototype for the supply and demand matching using a number of Raspberry Pis connected over a WiFi network. This could then be used to analyse network latencies as well as to create a real network and prove whether or not this system could be implemented on a larger scale.

A second potential extension would be to create a hierarchy of CPSs in a larger smart grid, where the CPS of a nanogrid in a house would then act as an EC within say the community or neighbourhood. The CPS controlling that region would then be an EC for a larger region, say a county or even an area code. However, a major change would need to take place in order to implement such a system. The current system which was examined in this project, only attempted to supply energy less than or equal to the upcoming energy deficiency. This extension however, would require the system to be able to produce a surplus of electricity for a single house at a given time period. If this were possible then a house could then act as a supplier in the game within its own neighbourhood and actually generate a further profit for that house.

As mentioned previously, as well as supply side management systems, there are also numerous papers concerned with demand side management. Most of these papers were primarily concerned with driving down the price of energy at any one given timeslot so it would be both beneficial and interesting to investigate pairing such a system with the system in this paper in order to see whether the low price generated is still enough to incentivise suppliers to give energy to the CPS. If successful, such an amalgamation would help consumers who had such a grid system set up in two different ways, first by driving down their costs and then by allowing them to sell energy to make further savings.

The final potential continuation of this project would be to pair any devices that would normally solely be suppliers in a nanogrid, such as wind turbines and solar panels, with some form of prediction software relating to weather patterns [31]. This data could then be used to inform an EC's caution value. For example if the EC was nearing capacity and it knew that it was going to be generating enough electricity that it would be unable to store it, then it could have a very low caution value. Conversely, low energy capacity and low production in the future could could be used to inform a higher caution value.

Bibliography

[1] R. Ambrosio, "Transactive energy systems [viewpoint]," *IEEE Electrification Magazine*, vol. 4, no. 4, pp. 4–7, 2016.

[2] G. Boyle *et al.*, *Renewable energy: Power for a sustainable future*. Taylor & Francis, 1997.

[3] W. Tushar, J. A. Zhang, D. B. Smith, H. V. Poor, and S. Thiébaux, "Prioritizing consumers in smart grid: A game theoretic approach," *IEEE Transactions on Smart Grid*, vol. 5, no. 3, pp. 1429–1438, 2014.

[4] V. Lauber, "Refit and rps: Options for a harmonised community framework," *Energy policy*, vol. 32, no. 12, pp. 1405–1414, 2004.

[5] N. Apergis and J. E. Payne, "Renewable energy consumption and economic growth: Evidence from a panel of oecd countries," *Energy policy*, vol. 38, no. 1, pp. 656–660, 2010.

[6] A.-H. Mohsenian-Rad and A. Leon-Garcia, "Optimal residential load control with price prediction in real-time electricity pricing environments," *IEEE transactions on Smart Grid*, vol. 1, no. 2, pp. 120–133, 2010.

[7] T. Markvart, "Microgrids: Power systems for the 21st century?" *Refocus*, vol. 7, no. 4, pp. 44–48, 2006.

[8] K. Hirose, J. Reilly, and H. Irie, "The sendai microgrid operational experience in the aftermath of the tohoku earthquake: A case study," *New Energy and Industrial Technology Development Organization*, vol. 308, 2013.

[9] T. Harbert. (Feb. 18, 2015). Post-fukushima, japanese companies build microgrids, [Online]. Available: `http://spectrum.ieee.org/energy/the-smarter-grid/postfukushima-japanese-companies-build-microgrids`.

[10] G. Sapienza, G. Bianco, G. Di Lembo, L. D. Carpini, G. Scrosati, and P. Paulon, "The enel smart grid test system: A real-time digital simulator-based infrastructure," 2013.

[11] B. Nordman, K. Christensen, and A. Meier, “Think globally, distribute power locally: The promise of nanogrids,” *Computer*, vol. 45, no. 9, pp. 89–91, 2012.

[12] T. Couture and Y. Gagnon, “An analysis of feed-in tariff remuneration models: Implications for renewable energy investment,” *Energy policy*, vol. 38, no. 2, pp. 955–965, 2010.

[13] S.-L. Nicola. (Feb. 20, 2017). Feed-in tariff (eeg feed-in tariff), [Online]. Available: `http://www.res-legal.eu/search-by-country/germany/single/s/res-e/t/promotion/aid/feed-in-tariff-eeg-feed-in-tariff/lastp/135/`.

[14] S. Parsons, J. A. Rodriguez-Aguilar, and M. Klein, “Auctions and bidding: A guide for computer scientists,” *ACM Computing Surveys (CSUR)*, vol. 43, no. 2, p. 10, 2011.

[15] B. Ramachandran, S. K. Srivastava, C. S. Edrington, and D. A. Cartes, “An intelligent auction scheme for smart grid market using a hybrid immune algorithm,” *IEEE Transactions on Industrial Electronics*, vol. 58, no. 10, pp. 4603–4612, 2011.

[16] A. Cournot, “Researches in the mathematical principles of the theory of wealth (translated by bacon, n.),” *London: Haffner*, 1960.

[17] W. Poundstone, *Prisoner’s Dilemma/John von Neumann, Game Theory and the Puzzle of the Bomb*. Anchor, 1993.

[18] A. K. Dixit, *Optimization in economic theory*. Oxford University Press on Demand, 1990.

[19] S. M. Piryonesi and M. Tavakolan, “A mathematical programming model for solving cost-safety optimization (cso) problems in the maintenance of structures,” *KSCE Journal of Civil Engineering*, pp. 1–10, 2017.

[20] M. Ahat, S. B. Amor, M. Bui, A. Bui, G. Guérard, and C. Petermann, “Smart grid and optimization,” 2013.

[21] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[22] G. Troderman. (Mar. 5, 2016). Convexity and jensen’s inequality, [Online]. Available: `http://www.gtmath.com/2016/03/convexity-and-jensens-inequality.html`.

[23] I. Konnov and J.-C. Yao, “On the generalized vector variational inequality problem,” *Journal of Mathematical Analysis and Applications*, vol. 206, no. 1, pp. 42–58, 1997.

[24] M. V. Solodov and B. F. Svaiter, “A new projection method for variational inequality problems,” *SIAM Journal on Control and Optimization*, vol. 37, no. 3, pp. 765–776, 1999.

[25] L. Armijo, “Minimization of functions having lipschitz continuous first partial derivatives,” *Pacific Journal of mathematics*, vol. 16, no. 1, pp. 1–3, 1966.

[26] Z. Yun, Z. Quan, S. Caixin, L. Shaolan, L. Yuming, and S. Yang, “Rbf neural network and anfis-based short-term load forecasting approach in real-time price environment,” *IEEE Transactions on power systems*, vol. 23, no. 3, pp. 853–858, 2008.

[27] M. Grant, S. Boyd, and Y. Ye, “Disciplined convex programming,” in *Global optimization*, Springer, 2006, pp. 155–210.

[28] S. Diamond and S. Boyd, “Cvxpy: A python-embedded modeling language for convex optimization,” *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[29] T. M. Labs. (Apr. 8, 2017). Twisted, [Online]. Available: `https://twistedmatrix.com`.

[30] K. Kinder, “Event-driven programming with twisted and python,” *Linux journal*, vol. 2005, no. 131, p. 6, 2005.

[31] A. M. Foley, P. G. Leahy, A. Marvuglia, and E. J. McKeogh, “Current methods and advances in forecasting of wind power generation,” *Renewable Energy*, vol. 37, no. 1, pp. 1–8, 2012.