Summer 7-17-2014

Measuring Autonomy And Solving General Stabilization Problems With Multi-Agent Systems

Rasheed A. Rajabzadeh
University of Nebraska-Lincoln, rasheed1361@live.com

MEASURING AUTONOMY AND SOLVING GENERAL STABILIZATION PROBLEMS WITH MULTI-AGENT SYSTEMS

By

Rasheed Ali Rajabzadeh

A Thesis

Presented to the Faculty of

The Graduate College at the University of Nebraska

In Partial Fulfillment of Requirements

For the Degree of Master of Science

Major: Computer Science

Under the Supervision of Professor Leen-Kiat Soh

Lincoln, Nebraska

July, 2014

MEASURING AUTONOMY AND SOLVING GENERAL STABILIZATION PROBLEMS WITH MULTI-AGENT SYSTEMS

Rasheed Ali Rajabzadeh, M.S.

University of Nebraska, 2014

Advisor: Leen-Kiat Soh

Many distributed complex problems address a particular form of resource scheduling where proper resource management can cut costs by stabilizing a set of stochastic fluctuating parameters. Wireless sensor network communication, supply chain management, stock trading, intelligent traffic management, and smart grid systems are examples of these problems. Among the various solutions, a common strategy often used to address this type of problems is fluctuation reduction via resource buffering combined with load shifting. Respectively, stable wireless communication, demand for supplies, liquidity, traffic speed, and power demand reduce cost and can be achieved by properly managing sensor data buffers, warehouses, capital, distance between vehicles, and power storage units. Although on the surface, the differences between such problems appear to warrant completely different multi-agent solutions, they can be rephrased or approximated in common terms that enable the generalization of various solutions and techniques.

This thesis is concerned with generalizing fluctuation reduction problems and their solutions. To that end, this thesis defines the fluctuation problem class in a multiagent framework, provides a general solution, applies the general solution to the smart grid problem, investigates the solution dynamics with respect to common multi-

agent system techniques, and finally defines a set of solution approach autonomy measurements. The resulting conceptual framework and applied investigation are directed at synthesizing currently disparate MAS research efforts which address fluctuation stabilization.

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION

A broad class of multi-agent problems is concerned with improving the efficiency and effectiveness of each agent and/or the system as a whole by managing and reducing fluctuations in production and consumption of resources where production, consumption, and resources are potentially abstracted concepts. Often, each problem provides limited information regarding future production and/or consumption, means to throttle current production and/or consumption, and means to buffer or store resources.

Generally these problems define various entities which are subject to information, communication, and processing restrictions that render an otherwise straightforward centralized optimal deterministic solution impractical. On the other hand multi-agent system built around simple concepts are inherently appropriate for solving such problems.

The general agent strategy is to use the available limited information to predict production/consumption in order to manage and schedule buffering and throttling resources in an effort to stabilize production/consumption which results in improved agent and/or systemic efficiency and effectiveness. We refer to this class of MAS problems as the stabilization problem class.

Most related multi-agent work has been concerned with various approaches and solutions to individual problems of the stabilization problem class. These solutions are mostly not generalizable to the class and as a result of being

entanglement with problem specifics, they are prone to being complicated and potentially less effective.

The intention of this thesis is to create a basis for generalizing and understanding multi-agent problems and solutions related to the stabilization problem class. As such, this thesis presents a generalized approach and methodology to developing multi-agent solutions to problems of the mentioned class, a detailed multi-agent solution to the smart grid problem, the analysis of various standard multi-agent approaches in combination with the proposed methodology, and a new metric to measure agent autonomy versus efficiency.

1.1 Background

The stabilization problem class is broad; particularly, since many seemingly unrelated problems can be reduced to, or approximated in terms of a fluctuation minimization problem. Contrary to appearances, at heart many high profiles MAS problems such as intelligent traffic control, and high frequency stock trading, are in fact in instances of the stabilization problem class. Over the following subsections we introduce some background on the problem class and the premise of our research through several examples. For the sake of clarity we avoid exotic examples in this chapter; however, we do provide such examples in Subsections 9.1.2.1 and 9.1.2.2.

1.1.1 The Bullwhip Effect

In the field of business, the bullwhip effect increases demand fluctuation at various stages of a distribution channel starting from the end customer to the production source [1]. Consider a distribution channel consisting of the consumer, retail,

wholesale, and manufacturer stages. As a result of the bullwhip effect, a small fluctuation in consumer demand increases demand fluctuation at the retail stage, which subsequently further increases demand fluctuation at the wholesale stage and ultimately the manufacturer stage (see Figure 1.1).

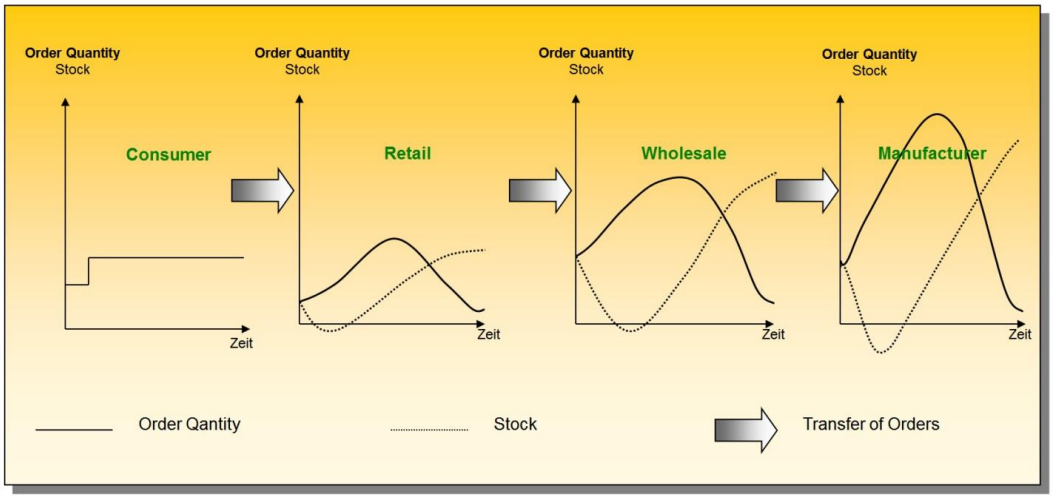


Figure 1.1: The bullwhip effect on demand at various stages of a distribution channel (Wikipedia). Ziet is German for time.

The manufacturer suffers the greatest demand fluctuation as a result of a relatively minor consumer demand fluctuation. In most cases demand fluctuation introduces a large amount of overhead to each stage of the distribution channel resulting in higher product costs. The problem of reducing the bullwhip effect is generally approached in terms of improving demand forecast quality and improving production scheduling; however, it can also be considered in terms of minimizing demand fluctuation through the distribution channel [1] [2] [3]. The general solution approach it to predict demand in order to throttle production, manipulate stage-intermediate storage space and prioritize orders in the distribution channel such that demand fluctuation is reduced at each stage. Prediction, throttling, and storage are a

recurring solution theme discussed by this thesis. The storage space at any stage in a channel can act as a buffer absorbing some demand fluctuations imposed by higher channel stages making it possible to impose a more stable demand on lower channel stages without compromising sales. The same effect can be achieved by spreading the response to demand received from a higher stage over time by delaying and prioritizing the demand. Much work has been devoted to developing MAS solutions to the bullwhip effect problem [4] [5] [2] [6] [7]. In short, a generalization of the bullwhip effect problem to the entire problem class becomes apparent when considering the problem in terms of reducing demand fluctuation.

1.1.2 The Cluster Load Balancing Problem

Another example from the problem class is related to server cluster load balancing. Consider a cluster supporting a large scale social media website, a search engine, or an online video streaming website. At certain times a particular region may place a lot of demand or even possibly an overwhelming demand on the clusters while at other times and for other region the cluster may be underutilized. The fluctuation in demand degrades the efficiency of the cluster by introducing cost overhead from idling, short upgrade cycles, customer dissatisfaction, and even reliability issues. The problem would not exist if users did not have fluctuating demand.

It is possible for the cluster to considerably even out demand by caching media and media associations useful to each region and time when customer demand is low. The process of caching imposes demand on the cluster where otherwise it would be relatively idle. When demand increases a portion can be supported by the

cache effectively reducing end demand on the cluster at those times. By predicting the type of demand, relevant content can be cached increasing the cache effectiveness. Furthermore, the cluster may postpone some low priority operations such as loading new advertisement content in place old content to a time when demand is lower; such tasks will eventually complete within their required timeframe, but in this way demand fluctuation on the cluster can be reduced.

1.1.3 The Smart Grid Problem

Another instance of the problem class that this thesis develops on extensively is the smart grid problem. The smart grid is a conceptual power system in which the resources that are responsible for generation, transmission and distribution of electricity are not only decentralize and distributed across the system but their management and the management of customer demand leverages recent advances in communication and computation to maximize system operation. The distributed resources include (but are not limited to) power lines, generators, and electrical storage systems. Recently, MASs are being recognized as well suited for smart grid resource management. In this thesis we define the smart grid problem to be that of minimizing the individual expenses of satisfying the load of electricity customers while maximizing system effectiveness and efficiency. As discussed later the effectiveness and efficiency of a power system has a direct relation with the demand fluctuation of electricity customers and ultimately electricity rates. Although finding a solution to the smart grid problem is extremely complex and potentially infeasible at best, it is possible to approximate the problem and solution in terms of minimizing

global demand fluctuation in place of minimizing customer expenses and maximizing system effectiveness and efficiency. The smart grid problem is an example problem which has a good approximation contained in the fluctuation problem class while possibly not being a problem class instance itself. Similar to the bullwhip effect problem, the general approach to the smart problem approximation is to shift customer demand in order to minimize demand fluctuation and to improve the prediction of customer demand. Demand shifting is accomplished by advancing demand to charge a storage system such as a battery or is accomplished by postponing demand by prioritizing customer loads such that customer activity is not disturbed.

Many other examples exist which for the most part have been investigated independently by the MAS community. As suggested by the examples, there are many similarities in the MAS solution approaches and the problem instances themselves making it possible to generalize the problems and solution approach and investigate the generalizations as to what extent common MAS techniques, such as coalitions, can contribute.

1.2 Problem Statement

The stabilization problem class consists of problems in which the main concern is the efficient production and consumption of a set of resources. Common to these problems are a set of producers and consumers each interested in maximizing the efficiency of their production and consumption respectively. In order to do so, a subset of the producers and consumers can modify their supply and demand

respectively using a buffer resource. The producers can meet varying demands; however, they are substantially more efficient at nearly any if not all steady production rates than equal or lower but fluctuating production rates. Consequently the cost of production in such problems is predominantly dependent on production fluctuation. Inherent to these problems is that consumption costs are directly related to production costs. The direct relationship may be imposed by an abstract cost function by the producer or it may be intrinsic. It is often the case that an abstract cost function may not capture the actual production costs but rather a best effort estimation in which case consumers may leverage the estimations to their advantage at the cost of the producer. An example of an abstract cost function would be amortizing the cost of producing and storing perishable commodities to supply fluctuating market demand. An example of an intrinsic production and consumption cost relations is the fuel consumption of a car being driven in the stop-and-go environment of a city compared to driving steadily on the highway, where the producer is the car, the product is displacement, the consumer is a driver with fluctuating displacement demands, and the cost is fuel.

Another key property of the problem class is that producers and consumers are imposed restrictions on what they can observe and manipulate in the environment; in particular, they do not know of the current global supply or global demand. In other words, the environment which the class instances describe are inherently distributed. In summary the problem class has the following key properties:

1. Producers and consumers are interested in being efficient

2. Production is generally most efficient at a steady rate
3. Consumption costs are directly related to production efficiency
4. Producers and consumers can adjust their demand via a buffer

It is important to note that producers may or may not have the capacity to adjust their production rate independent of demand; however, in general over or under production is inefficient; consequently, producers will avoid such situations as much as possible if the option to over or under produce exists. We refer to the problem class having the properties above as fluctuation problems.

It is worth emphasizing that the four fluctuation problem properties listed above imply that the problem of optimizing cost is strongly rooted in reducing fluctuation. Although reducing fluctuation does not necessarily guarantee cost reductions, particularly as a consequence of property 2, reducing fluctuation does in all significant cases reduce cost. Where such a general rule fall short of an optimal direct solution, MAS solutions, which are intended to address otherwise intangible problems, are intended to leverage such rules to produce the most effective practical solution.

The general solution to any fluctuation problem is for producers and consumers to modify their supply and demand rate such that both fluctuation is minimized and their functionality, namely, providing the product demanded by consumers in the case of producers, and consuming the required product in the case of consumers, is not sacrificed.

Section 1.1 demonstrated the extent of the problem class by described various instances. Our intention is to generalize a solution for all instances of the problem class. We focus on the smart grid problem, in order to evaluate and investigate the proposed multi agent solutions to the entire problem class; for that reason, the following section develops on this problem more extensively.

1.3 Smart Grid Power Network Problem

The smart grid is a future intelligently managed distributed power system which is slowly replacing the current centralized power system. As a power system, the smart grid is concerned with the scalable, reliable and efficient transmission and distribution of power. The smart grid, like micro-girds, leverages heterogeneous low capacity distributed power system components in order to accomplish its objectives with higher scalability and reliability. However, where micro-grids suffer from the complexities of managing distributed resources, the smart grid harnesses advances in computation and communication in order to efficiently and even in some cases intelligently manage resources.

The smart grid leverages advanced technologies such as the wireless sensor networks, communication networks, local direct current power, to name a few, in order to address real time pricing, and responsiveness requirements associated with power storage and renewable energy.

The smart grid consists of power plants, utility companies, and the transmission and distribution networks, which for simplicity we collectively refer to

as the grid, and various residential, commercial and industrial customers. In terms of the problem class the grid is the producer and the customers are the consumers.

Consumers, particularly those of the same class, have similar demand profiles. For instance, most customers are likely to turn their lights on at night while in the early morning they are mostly inactive. As a result global consumer demand has extreme peaks and dips. Enough investment in gird capacity must be made in order to reliably support peak global consumer demands. Most of the time this investment is underutilized since peak demands are short lived and infrequent. Other than low investment utilization, the producers also suffer inefficiency due to higher investment depreciation and increased unreliability as a result of fluctuating demand and production rates. Producers incorporate the costs they incur as a result of inefficient production into electricity rates. As a result consumers suffer higher electricity rates and lower reliability when production is inefficient. As a result both produces and consumers satisfy property 1, 2, and 3 of the problem class.

Producers and consumers can advance their supply and demand respectively to an earlier time by using storage devices such as batteries or mechanical displacements. Consumers can additionally postpone inessential loads to a later time. These adjustments act as a controllable buffer between producers and consumers fulfilling property 4 of the problem class.

In this example production is adjusted to be almost the same as demand since over producing will harm equipment and under producing will halt the electricity system. Nevertheless, some power plants may elect to over produce and discard

excess power in order to avoid fluctuating production rates although this is rare since discarding excess power is very costly. Often the power plant pays external sources to dispose of such excess power since it is cheaper [8] [9].

We refer to the problem of efficiently managing buffering resources by individual producers and consumers in order to minimize supply and demand fluctuation and thereby minimizing costs, as the smart grid problem.

1.4 Solution Approach

Common to the stabilization problem class is the notion that:

Steady utilization of resources produces more optimal results than overwhelming than idling resources in short bursts.

This tendency results in a desire to schedule and manage resources accordingly. For instance, a wireless sensor network modeled as an MAS may be concerned with stabilizing network traffic so as to minimize contention and wasted energy at each node. The objective of stabilizing a set of global properties is not unique to scheduling systems. For instance, a modular robot modeled as an MAS may be concerned with stabilizing the center of gravity to insure the robot as a whole remains balanced despite dynamically changing external forces.

By definition of the problem class, stabilizing global demand is central to achieving efficient production and consumption. Demand in this sense is an abstract concept with interpretations that vary from problem instance to problem instance. Global demand is the net effect of a distributed deficit which must be satisfied; as

such, global demand presents itself as a set of aggregate environment properties[1]. We differentiate between distributed local observations of demand as local demand and their aggregation as aggregate demand and the absolute demand of a disjoint system component as the global demand of that component.

In order to stabilize global demand without sacrificing consumption, global demand must be offset and spread as evenly as possible over time. In order to offset global demand it must be predicted accurately.

1.5 Investigation through Autonomy

In the field of MAS, autonomy, in its most simple and common form, is understood as the amount in which an agent can realize its goals independent of anything external to itself. As such autonomy provides great insight into important internal MAS dependencies. Over our investigation into the smart grid problem and the generalized solution for the stabilization problem we strongly rely on the notion of autonomy to uncover key internal properties. In particular we explore the dynamics of the smart grid problem and solution by comparing the results of varying autonomy with one another.

Finally we generalize our research process itself by defining two measures of autonomy which aid comparative investigation into goal dependencies of various

[1] In most cases demand presents itself as one aggregate property such as citywide electricity demand, but this need not be the case. For example consider the problem of workers in a city. Workers may work from home and reducing demand on city roads as their demand for internet bandwidth is more satisfied. In this example, demand for roads and internet bandwidth form a set which describe a more general demand for connectivity.

classes of MAS solutions. We revisit the idea of autonomy and how it relate to uncovering core MAS solution dynamics more over subsequent chapters.

1.6 Contributions

Our key vision is to establish a conceptual framework which unifies many MAS problems and solutions, which until now have been considered independently, under the stabilization problem class. In the direction of realizing the first step towards that vision we define the simulation problem class, propose a general solution methodology, and apply the methodology in producing a solution approach for the smart grid problem. We then investigate the dynamics of the solution approach and indirectly the solution methodology by creating an MAS simulation framework that implements the smart grid and solution approach. In order to gauge the flexibility of the solution approach to MAS techniques we introduce and implement an ad-hoc coalition enhancement to the solution approach. We then investigate the solutions by simulating them under conditions where autonomy is controlled. Finally we generalize our investigation process itself by defining two measures of autonomy which capture the autonomy inherent to MAS techniques such as coalitions, learning, auctioning, negotiating, voting, etc., when applied under a set of reasonable restrictions to a baseline solution approach. In summary our research:

1. Proposes a general solution that applies to all instances of the stabilization problem class
2. Applies and evaluates the general solution using the smart grid problem

3. Produces an MAS simulation framework for stabilization problems and customizes the framework to simulate the smart grid problem
4. Describes a relative and absolute measure of autonomy inherent to an MAS technique, such as learning or coalitions, when applied to a solution approach

1.7 Overview

In Chapter 2, we review the current state of related literature and the details of some MAS solutions related to the bullwhip problem and smart grid problem. In Chapter 3, we define the terminology and formulate the methodology that we propose as the basis of our solution to all stabilization problems. Chapter 4 elaborates the components of the smart grid and describes the approach we derive from the methodology to target the smart grid problem. Chapter 5 describes the details of our simulation framework and the configuration options it provides that are relevant to the smart grid and this thesis. Chapter 6 walks through the process of some preliminary simulations with the objective of finding and justifying configurations for 3 simulations cases, namely the solo, neighborhood, and unity simulation cases. Chapter 7 interprets and justifies the results observed from each of the simulations configured in the previous chapter. Chapter 8 generalizes the investigation process covered in Chapter 6 and Chapter 7 by defining two measures of autonomy and using those measures to recapture our findings in Chapter 7. In Chapter 9 we summarize our results and their significance, and finally, as future work we propose two revisions to our solution methodology and two high profile MAS problems which despite appearances are in fact likely to be instances of the stabilization problems.

CHAPTER 2 LITERATURE REVIEW

Our investigation covers defining and solving the stabilization problem, a new solution to the smart grid problem, and new autonomy measures. We have reviewed a large body of work done in each area. The following sections summarize our understanding of this body of work grouped by the areas covered in this thesis.

2.1 Reviewing Stabilization Problem

In Chapter 1 we have provided several examples of stabilization problems which have been addressed by MAS solutions. Over our literature review, we did not find any work addressing the generalized stabilization problem class as a whole. Most of the relevant MAS work are disparate studies and solutions on particular instances of the stabilization problem class. Although it is not feasible to enumerate through the MAS research addressing all subject categories that the stabilization problem class spans, we have reviewed a large body of work done covering fractional reserve banking, reservoir water management, stock trading, grid computing, communication network traffic, supply chain management (SCM), smart grid networks, intelligent traffic management. Although the problems posed in many of these subjects are good candidates for the stabilization problem class, the research work addressing the problems posed SCM and smart grid networks were most explanatory of the current state of relevant work relevant to the stabilization problem class. Except for the subject of grid computing, the relevant research in other subjects was difficult to exemplify in order to provide a good representation of the current state of related work; mostly this is due to the sparseness of related work relative to the body MAS

work in each of the other subjects. The following subsections summaries our understanding of the current state of relevant work through exemplifying the subjects of SCM and the smart grid.

2.1.1 Supply Chain Management

Extensive MAS researcher has been focused on SCM in which the primarily concern is establishing a stable supply chain (SC). This research predominantly builds on SCM domain concepts by applying various communication and collaboration protocols, learning and intelligence strategies, and other MAS techniques and principles. This has led to many disparate approaches and strategies which all share management of warehouse or inventory facilities to buffer fluctuations in supply and demand [10] [3] [5] [2] [6] [7]. For example, [2] proposes SC agents communicate both the amount of supplied needed to meet the demand they are facing but also the supplies needed to replenish inventory changes during the period it takes for the supplies to arrive. In this manner agent can communicate their over/under-order for a product down the SC allowing for each supplying agent to recognize short lived requests from its upstream demanding agents. The authors of [5] propose the use of genetic algorithms for forecasting demand at each inventory echelon having agent share information and demand forecasts using a communication protocol. The authors of [6] propose capturing typical SCM dynamism as a constraint network model and having agent optimize over the model by sharing information and following a genetic algorithm.

Ultimately, most of these approaches take a domain specific top down approach which is difficult to generalize and does not directly address the objective of reducing demand fluctuations in a SC.

2.1.2 The Smart Grid

There have been spanning developments in smart grid MAS [11] [12] [13] [14]. Some of these developments address the smart grid problem. But, even in the subject of applying MAS to solving the smart grid problem, a myriad of approaches have been taken. Although all of these independent solutions ultimately reduce demand fluctuation to some extent, they do so indirectly under the restrictions of the approach as opposed to under the restrictions of the actual objective. As such, the approaches lead to a diversification of solutions as opposed to a generalization. We will cover some popular approaches.

2.1.2.1 Utility Based Approaches

In the utility based approach agents are prescribed various utility functions describing what action to take and to what extent to take them. The agents then optimize over the utility space as opposed to the space measuring the final goals. For instance, consider the possible flow of energy for a smart grid agent from Figure 2.1. An agent may opt for a subset of the energy flows each based on a set of associated utility functions, say the utility of consuming and the utility of selling [15].

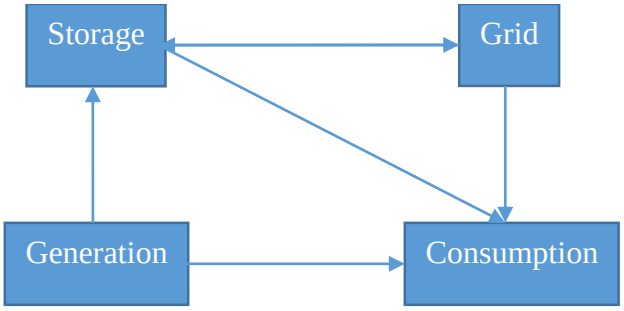


Figure 2.1: A possible set of energy flow options for a simple smart grid entity. In general generation cannot be directly consumed; however, some forms of generation (thermal collector enhanced solar panels) and some forms of consumption (water heating) allow the option of direct power consumption.

In a similar approach, the difference in market utility at two points in time, where market utility is defined of the attractiveness of buying power from the market, has been used [16]. Other solutions focus more on general utility optimization that rely on multi-attribute utility theory and incorporate Monte Carlo and logistic regression methods with various agent learning techniques [17]. The final objective of the systems in each of the solutions is to reduce demand fluctuation. However, the utility function approach quickly leads to the definition of a solution space that does not capture the objective naturally and as such does not capture the interest of the agent. In the case of the smart grid the agents benefit from solving the systemic problem of demand fluctuation.

2.1.2.2 Game Theoretic Solutions

Many solution are based on game theoretic frameworks. The general approach is to design at least one game where the agents as players each try to receive the highest score or pay-off. Usually the pay-off is based on the cost of satisfying the agent's load over a particular time [18] [19] [20] or in more complicated games the pay-off is the

difference in value of the obtained power and the cost of obtaining the power [21]. Finally, as evidence of correctness it is demonstrated that optimum performance is achieved as the Nash equilibrium. Although these techniques do provide elegant easy to explore and verify solutions, the solutions are again very much specific to the smart grid problem.

2.2 Defining Autonomy

Autonomy is one of the key properties and motivations for MASs. In its general sense, autonomy is a very broad concept making it difficult to define or quantify. Consequently, autonomy has predominantly been studied subject to a particular definition and in a relational manner where always a set of agents are the object of the relation. The consensus is that autonomy is a directional spectrum which at its maximum presupposes independence or the absence of relationship and additionally requires goals to be minimally achievable [22] [23] [24] [25]. At the minimum end of the spectrum, the absence of autonomy assumes full dependence such that any level of functionality is not possible [22] [23] [24]. For instance, in the case of a teenage child and parent, the child is considered somewhat autonomous with respect to her parent since she is able to achieve some level of survival regardless of the parent but flourishes with the added care of the parent, where here we assume surviving well is the agent's goal. The directionality of autonomy suggests that the degree of autonomy entity *A* has from entity *B* does not have to be the same visa-versa [24]. The directional nature of autonomy is evident from the child parent example. Concisely, autonomy is the degree in which a set of goals can be independently realized.

2.3 Autonomy Research Branches

Research on agent autonomy can be split into two broad categories:

1. Agent-user autonomy: Autonomy of agents from users
2. Agent-agent or agent-environment autonomy: Autonomy of agents from non-user entities

Much research on autonomy targets understanding the autonomy of a set of agents from a set of users. A large application of such research is agent driven robotic technology designed to coexist or serve users such as astronomers [26] [27]. It is common in such research that a component of an agent goal be carrying out the preferences and will of the user set. In this research category, a common desire, and indeed one of key defining factors of the category, is to maximize autonomy of agents from users, since maximizing autonomy would result in agents serving their goals with minimal dependence to users [24]. This branch benefits from most of attention to research MAS autonomy.

Another large yet less considered branch of autonomy research is concerned with understanding autonomy among various sets of agents or between a set of agents and a subset of the system environment not including users. The common desire in such research is to minimize complexity while maximizing agent and/or system effectiveness. In particular, the desire is to understand the relationship between autonomy and rules and policies, system distributiveness, and system or agent reliability [28] [29]. But perhaps more fundamental to this area of research, is

understanding the trade-off autonomy often exhibits with respect to performance, accuracy, and complexity (overhead) of agents or the system as a whole [30].

Most of the progress made in this branch does not target its fundamentals and furthermore is accomplished through independent studies which are indirectly related to the relationship between autonomy and solution performance/complexity. Our thesis is more interested in the agent-agent autonomy.

2.4 Motivating Agent-Agent Autonomy

The relationship between agents is often indirect and difficult to capture; autonomy measures such as those discussed in Section 2.5 provide a means, not only to detect but also measure the impact agents have on one another's ability to achieve their goal, whether their goals be completely the same or not.

To underline the importance of autonomy and what role it plays in MAS it is helpful to look at MAS from a general vantage point. In essence MAS are practical approximations to otherwise intangible solutions. As a particular form of Autonomy Oriented Computation (AOC) [31], the general approach enabling MAS approximations can be viewed as:

- Eliminating or relaxing dependencies and entanglements that restrict and complicate the proper solution such that the new approximate solution space can adequately represent the original solution space
- Breaking down the intangible goal of the complete problem into smaller tangible goals which composed together cover most of the intangible goal

When designing a MAS to approximate a difficult problem, the designer consciously or subconsciously prunes away dependencies in the original solution and decomposes the original goal into sub-goals such that each sub-goal can be achieved with little to no dependence to the other. Each mechanism that undertakes a sub-goal forms an autonomous component of one agent or a set of similar or dissimilar interdependent agents (i.e., a coalition). By design such an autonomous component has some degree of autonomy with respect to other autonomous components which each are undertaking their corresponding sub-goals. The ability to detect and measure the autonomy of various solution components allows us to fine tune how the original problem solution is broken up to create a MAS approximation. The ability to detect and measure autonomy not only allows us to fine tune a MAS during design time but also opens the door for a dynamically adjusting MAS approximation particularly of interest in problem which has a dynamic not easily predictable or generalizable goal [32] [33] [25] [34] [35] [23] [22].

2.5 Measuring Autonomy

Let $\mathbb{A}$ be the set of all agents in a MAS and $S \subseteq \mathbb{A}$ a nonempty subset and $i \in \mathbb{A}$ an agent. For some ratio scale measure of performance, let v_S^i be the performance corresponding to agent i in the presence of only the agents in set S, and let v_i^i be the performance of agent i in the presence of only itself. Table 2.1 lists various autonomy measures and their formalization as reasoned and defined by Bryanov and Hexmoor [24].

Table 2.1: Formulation of some autonomy definitions.

Definition	**Formulation**
Autonomy of agent *i* with respect to agent set *S*	$A_S^i = \frac{v_S^i}{v_i^i}$
Group autonomy of set *S*	$A^S = \sum_{i \in S} A_{S\setminus\{i\}}^i$
Autonomy of agent set *S* with respect to agent *k*	$A_k^S = \frac{\sum_{i \in S} A_{(S \cup \{k\})\setminus\{i\}}^i}{A^S}$

Such measurements open the door to many research opportunities in MAS; however, most research building on these measures is directed toward dynamic adjustment of autonomy, particularly, that of dynamically adjusting agent-user autonomy. Not much work leverages autonomy measures in order to understand the role agent-agent autonomy plays is defining limits on approximation effectives and complexity. For instance, using the measures from Table 2.1 and similar derived measures, given only a preconfigured black box MAS, and an agent performance metric, it is possible for one to produce a directed weighted graph describing the autonomy of each agent in the system with respect to others at a given moment. Given a set of autonomy graphs capturing snapshots of the MAS, in the case where the MAS is heterogeneous or composed of collaborative units, one could potentially classify each agent type and collaborating component. This classification can be done by measuring the network flows across various graph cuts. Since only a black box MAS is needed one could apply such an approach to reverse engineer naturally

occurring MASs or uncover implicit agent collaboration not stipulated by the MAS design. However, most importantly perhaps, it is possible to understand the flow of information and effects in the system and to identify points where autonomy, accuracy, and complexity can be fine-tuned. We will subsequently investigate some ideas posed here; however, most of them will be left for future work.

Aside from there not being much research leveraging agent-agent autonomy, current measures are subjective to particular instances of a simulation. It is not easy to compare agent autonomy across multiple similar simulations each leveraging set of varying MAS techniques such as coalitions, learning, auctioning, and etc.

In this thesis we introduce two measures which are independent of the relationship between individual agents corresponding of different MASs. The proposed measures are absolute across all MAS enhancements possible for a given solution approach. We further show how these measures can simplify finding relations between autonomy, computational complexity, and accuracy for a given solution approach.

CHAPTER 3 METHODOLOGY

As discussed in Section 1.2 and 1.4, our approach to solving the class of problems posed in Section 1.2, is reducing demand fluctuation. This Chapter describes a general methodology for reducing demand fluctuation by leveraging the buffering resources described in Section 1.3. The methodology is general in the sense that it is independent of consumer collaboration and relies only on modifying demand by partially advancing and/or postponing.

3.1 Basics and Demand Profile

For the sake of simplicity, let *consumer* refer to a collection of one or more consumers and let *demand* refer to the demand of a consumer. Let a *property profile* refer to the expected behavior of a stochastic and periodic property over and expected period; in particular, let *demand profile* refer to the expected demand over the duration of the expected demand period. Recall demand is periodic and stochastic (see Section 1.2); therefore, a consumer will know only an estimation of its demand profile and in particular demand period. In most of this Chapter, we consider property profiles in place of actual properties to explain the general problem structure and describe the proposed solution methodology without being limited by the complexities of the stochastic nature of these properties. Figure 3.1 illustrates a simple demand profile. We will use the demand profile in Figure 3.1 to illustrate the demand stabilization methodology described in this Chapter.

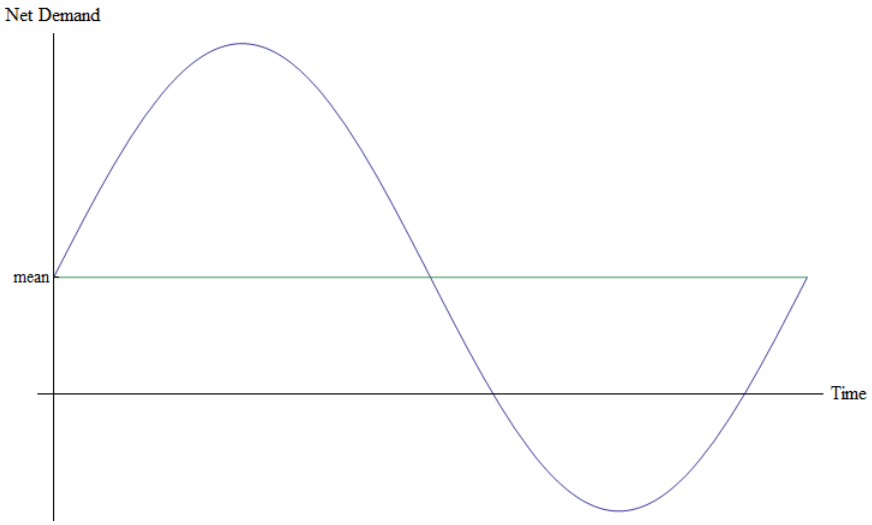


Figure 3.1: An illustration of an example demand profile. The blue curve indicated the expected demand at a given moment in the expected demand period of a consumer collection. The green curve indicates the mean of the demand over all time. The region where demand is negative corresponds to production or an excess of product.

It is important to note that demand can be negative in which case the interpretation is that the consumer is in fact a producer or provider; this has no ramification on the problem objectives since a producer or provider is just as interested in stabilizing production as a consumer is interested in stabilizing demand. As long as an entity has a positive (negative) mean demand we refer to it as a consumer (producer). Although the following methodology only refers to consumers the methodology applies to both consumers and producers.

3.2 Demand Stabilization

If demand adjustments were optimal and were able to successfully eliminate demand fluctuations, then the resulting demand profile would be the constant mean of the original demand over all time; in terms of Figure 3.1, such an adjustment would

modify the demand profile curve in blue to be the mean line in green. Our proposed demand stabilization methodology is motivated by this line of reasoning.

Let $d(t)$, $\delta(t)$, $\tilde{d}(t)$, and $\tilde{\delta}(t)$ respectively be the demand, demand modification, demand profile, and demand modification profile of a particular consumer at time t. In general we will decorate a periodic stochastic variable with a tilde to indicate its profile. Let T be a time interval starting from 0 spanning one period of $\tilde{\delta}(t)$; $|T|$ is then the expected period duration of the demand of the consumer. The mean demand of the consumer is:

$$\bar{d} = \frac{1}{|T|} \int_T \tilde{d}(t)dt \tag{3.1}$$

Since demand adjustment is limited to advancing and/or postponing demand and since collectively a consumer does not consume or produce more or less after adjusting its demand, then stochastically the total adjustment over T should satisfy:

$$\int_T \tilde{\delta}(t)dt = 0 \tag{3.2}$$

All demand modifications are accomplished via buffering resources which either advance or postpone demand. Consequently, $\delta(t)$ is limited by the total buffering resource capacity R_c as follows:

$$\left| \int_{t_0}^{t_1} \delta(t)dt \right| \leq R_c \quad \forall\, (t_0, t_1) \tag{3.3}$$

where (t_0, t_1) is any possible time interval; as a result this is also true for the demand modification profile as well. The interpretation of (3.3) is simply that a consumer may not adjust its demand over any time interval more than the consumer's buffering resources support.

The demand resulting after applying the adjustment $\delta(t)$ is:

$$d_\delta(t) = d(t) + \delta(t) \tag{3.4}$$

From (3.2) it is clear that the mean of any modified demand $d_\delta(t)$ is still $\bar{d}$. As discussed earlier, the optimal demand modification, $\delta^\star(t)$, must satisfy:

$$\bar{d} = d(t) + \delta^\star(t) \tag{3.5}$$

Therefore, with information about the mean demand and demand, the optimal demand modification can be computed by:

$$\delta^\star(t) = \bar{d} - d(t) \tag{3.6}$$

The interpretation of $\delta^\star(t)$ is to advance or postpone demand exceeding the average to some other point in time where demand is short of average. In other words, the demand modification redistributes peak demands to moments with relatively low demand (see Figure 3.2). It is important to note demand is not compromised; that is, all consumer demand is always satisfied at the moment when the consumer makes the demand; demand modifications simply act as a buffer between the stabilizing consumer demands. Figure 3.2 illustrates some of defined terms and the central methodology concept.

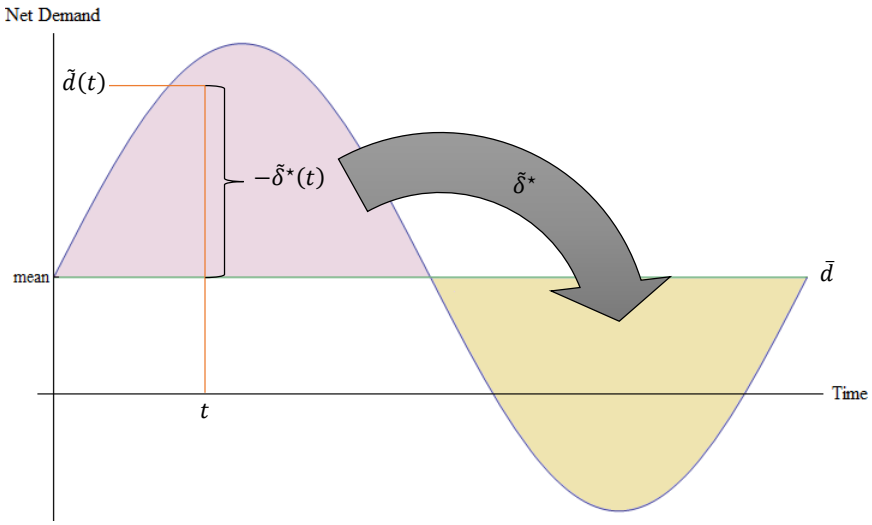


Figure 3.2: The red area is the total amount of demand exceeding the mean demand (total over demand) which is equal to the yellow area which is the total demand short of the mean demand (total under demand). Ideally, $\tilde{\boldsymbol{\delta}}^{\star}(\mathbf{t})$ redistributes the over and under demand of $\tilde{\mathbf{d}}(\mathbf{t})$ such that the resulting demand profile is precisely $\bar{\mathbf{d}}$.

It is important to note that finding and exploiting $\delta^{\star}(t)$ is challenging. In practice the $\tilde{d}(t)$ and $\bar{d}$ are generally not known beforehand, therefore $\tilde{d}(t)$ and $\bar{d}$ can, in general, only be estimations from $d(t)$ –we do not consider other information that may be available in particular problem instances. Consequently only estimations of $\delta^{\star}(t)$ are possible. Further complicating the matter is that generally $d(t)$ is only known at or after moment t has come to pass. Although, in some problem instances, it is possible that demand is unknown even until sometime in the past, and in other problem instances demand is known in advance; we will consider the case that demand is known only at the current time t and before since the other cases can be reduced to this case by applying other methodologies when precise demand in the future is known beforehand. Since it is not possible to compute $\delta^{\star}(t)$ we can only

rely on the periodic stochastic nature of demand to predict or estimate demand in the future. As such, aside from external information inherent to particular problem instances, in general, it is only possible to estimate $\delta^{\star}(t_f)$ based on $d(t_p)$ where t is the current moment in time and $t_p \leq t < t_f$. Estimations based on local observations are likely to be less effective than aggregate information, and consequently some form of consumer collaboration would be beneficial for estimating $\delta^{\star}(t)$ more effectively. Proposed prediction and collaboration techniques and their evaluations are not related to the general methodology and are discussed in Chapter 4. Finally, only a limited amount of resources for advancing or postponing demand are available. Therefore, applying $\delta_i^{\star}(t)$ is unlikely to be within means and thus achieving optimality is unlikely. In the following, we present a distributed, approximated approach to minimize demand fluctuations based on the above optimality criterion.

3.3 Minimizing Demand Fluctuations

Until now we referred to $\delta^{\star}(t)$ as optimal since it completely removes all demand fluctuation. However, when considering only a prediction of $\delta^{\star}(t)$ is available, the notion of optimal demand modification must capture the likelihood of any demand at time t_f occurring such that the buffering resource state at t_f can be prepared to accommodate arising circumstances accordingly. As such, the problem of minimizing demand fluctuation is difficult to solve deterministically, particularly when considering the interference of multiple customers on global demand. As a result, the methodology developed in this Chapter suffices to approximate the solution in an efficient distributed manner so that it can be applied as a MAS. We will still consider

the same definitions to further describe the methodology; however, in practice, particularly when describing an application in Chapter 4, all value must be approximated except for $d(t_p)$ where $t_p \leq t$.

In order to reduce demand fluctuation and consequently costs, some $\delta(t)$must be found such that it is as similar to $\delta^\star(t)$ as possible while satisfying restrictions dictated in (3.2) and (3.3). The interpretation of $\delta(t)$ being similar to $\delta^\star(t)$ is dependent on the particular problem being addressed, but in general, it can be interpreted as minimizing the inherent cost function of the problem; however, since directly minimizing such a cost function is generally impractical, other similarity or dissimilarity based cost functions are more desirable. Since we are guaranteed that reducing demand fluctuation reduces costs, minimizing or maximizing any similarity- or dissimilarity-based cost functions respectively will result in minimizing costs as well. Using such cost functions in place of the inherent problem cost function could result in different priorities when allocating buffering resources. Therefore, although we are guaranteed given enough buffering resources, both cost functions limit to the same cost optimum, we are not guaranteed the same when resources are a limiting factor. Examples of possible simple difference cost functions are the mean squared error and the Minkowski distance; and an example of a possible similarity cost function is the cosine distance.

Although a cost function could be used to produce an algorithm to find $\delta(t)$ from $\delta^\star(t)$, we instead directly describe an algorithm which guarantees $\delta(t)$ will

converge to $\delta^{\star}(t)$ as resources are increased. This algorithm (see Section 4.4) will have an associated cost function which is not directly of interest in this thesis.

Our proposed methodology uses a greedy heuristic to find $\delta(t)$ from $\delta^{\star}(t)$. In order to describe the heuristic we formalize the resource capacity restrictions on find $\delta(t)$. As previously mentioned in this section and discussed in Section 1.3, there are two forms of resources: one advances and one postpones demand, referred to as R^a and R^p, respectively. R^a and R^p act as a buffer to positive and negative amounts of product each having capacity R_c^a and R_c^p respectively. Therefore, $R_c = R_c^a + R_c^p$, which is consistent with the earlier definition of R_c. When demand is advanced, some product is buffered into R^a prior to being needed. When demand is postponed, the absence of some product (or the interest in some product) is buffered into R^p for a later time. For a resource R^x, where $x \in \{a, p\}$, the amount of free and used buffer space at time t is $R_f^x(t)$ and $R_u^x(t)$ respectively. Let $R_x(t)$ be defined as $R_x(t) = R_x^a(t) + R_x^p(t)$ where $x \in \{f, u\}$ (free capacity, and used capacity respectively).

Let the expected demand exceeding average and its total over a time interval be referred to as *over demand*, $\hat{d}_i(t)$, and *total over demand*, $\hat{\Sigma}_i(t)$, respectively. Similarly, let the expected demand short of average and its total over a time interval be referred to as *under demand*, $\check{d}_i(t)$, and total *under demand*, $\check{\Sigma}_i(t)$, respectively. Currently we define the time interval to be $(t, t + \Delta t)$ for some constant foresight Δt; however, in a more general manner, it is possible to define the interval as $\big(t, f(t)\big)$ for some foresight function f as discussed in later chapters. Table 3.1 provides the formulation for these definitions.

Table 3.1: The formulation of over and under demand at instant **t** and total over and total under demand at an instant **t** over some predefined foresight **Δt**.

	Instantaneous	**Total**
Over demand	$\hat{d}(t) = \max\left\{-\tilde{\delta}^{\star}(t), 0\right\}$	$\hat{\Sigma}(t) = \int_{t}^{t+\Delta t} \hat{d}(x)dx$
Under demand	$\check{d}(t) = \max\left\{\tilde{\delta}^{\star}(t), 0\right\}$	$\check{\Sigma}(t) = \int_{t}^{t+\Delta t} \check{d}(x)dx$

Total over demand indicates how much resources must be buffered in order to level demand over the foreseeable future Δt; it also indicates how much buffering resources are needed to accomplish the task.

The heuristic used to find $\delta(t)$ is to ration out the used buffer resources R_u proportional to the amount of over demand when demand is over average and to ration out the free buffer resources R_f proportional to the amount of under demand when demand is lower than average. In other words, when demand is over average, ration out the buffered product proportional to the over demand such that any foreseeable unit of over demand gets the same unit of buffered product; when demand is under average buffer product proportional to the under demand such that any foreseeable unit of under demand receive the same unit of free buffer space. The following equations formalizes the description of $\delta(t)$:

$$\delta(t) = \begin{cases} \delta^\star(t)\left(\min\left\{\dfrac{R_u(t)}{\hat{\Sigma}(t)}, 1\right\}\right), & \delta^\star(t) < 0 \\ \delta^\star(t)\left(\min\left\{\dfrac{R_f(t)}{\check{\Sigma}(t)}, 1\right\}\right), & \delta^\star(t) > 0 \\ 0, & \delta^\star(t) = 0 \end{cases} \tag{3.7}$$

Let $\mu(t)$ be defined as:

$$\mu(t) = \begin{cases} \min\left\{\dfrac{R_u(t)}{\hat{\Sigma}(t)}, 1\right\}, & \delta^\star(t) < 0 \\ \min\left\{\dfrac{R_f(t)}{\check{\Sigma}(t)}, 1\right\}, & \delta^\star(t) > 0 \\ 0, & \delta^\star(t) = 0 \end{cases} \tag{3.8}$$

then (3.7) can be simplified as $\delta(t) = \mu(t)\delta^\star(t)$.

Figure 3.3 illustrates the demand redistribution process. Figure 3.4 illustrates the resulting demand profile after modifying the profile in Figure 3.3 by $\delta(t)$.

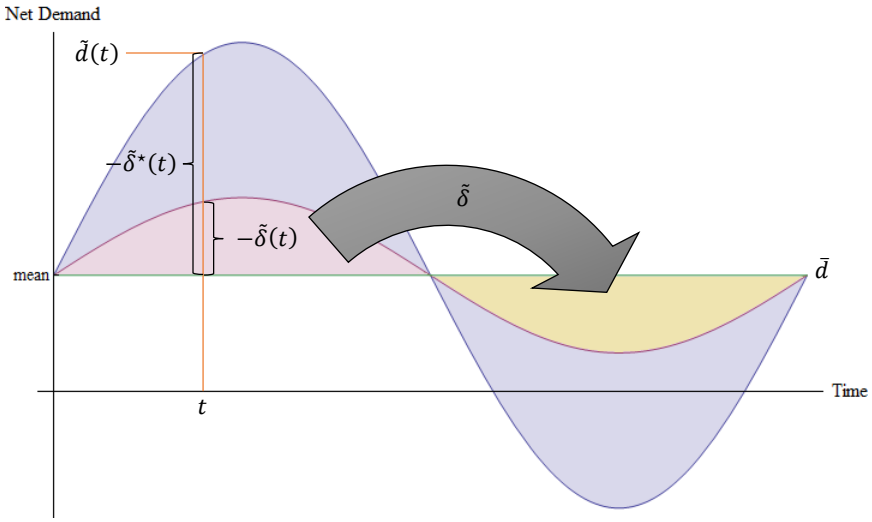


Figure 3.3: The process of modifying the demand profile indirectly with $\boldsymbol{\delta(t)}$. The area in red is the expected amount displaced by $\boldsymbol{\delta(t)}$ to the statistically equal area in yellow. The redistributed area is less than or equal to the total resource capacity $\mathbf{R_c}$.

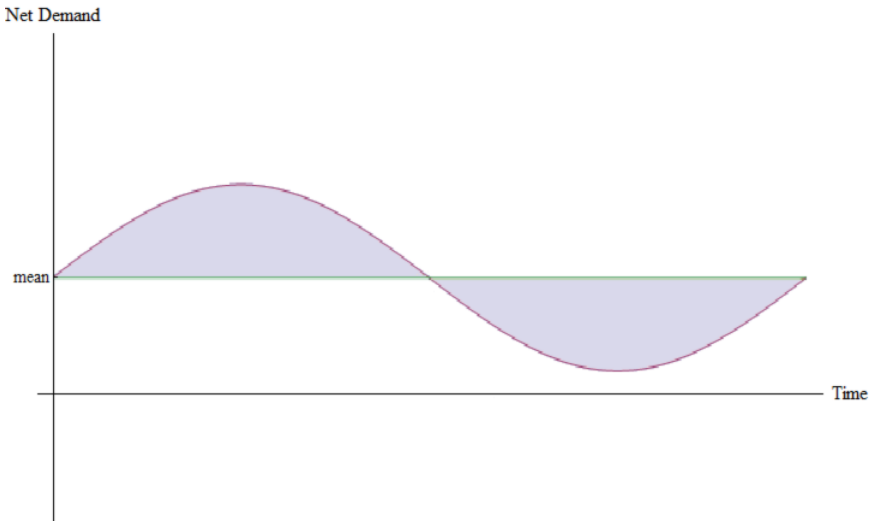


Figure 3.4 The resulting demand profile after applying the demand profile, $\tilde{\boldsymbol{\delta}}(\mathbf{t})$, in Figure 3.3.

CHAPTER 4 METHODOLOGY APPLICATION TO THE SMART GRID PROBLEM

This chapter describes several multi-agent solutions based on the methodology posed by this thesis to the smart grid problem described in Section 1.2. As discussed earlier, the smart grid problem is an instance of a larger problem class. Our solutions to the smart grid problem described in this chapter serve to demonstrate and investigate the application and methodology posed by this thesis to the entire problem class. The only assumptions of the methodology are the four properties of the problem class described in Section 1.2; therefore, the methodology applies to the entire problem class as exemplified by the solutions in this section.

4.1 Smart Grid Problem Simulation

The smart grid simulation is a multi-agent system consisting of electricity customers which each can potentially have their own electricity generation and storage facilities. All customers are connected to the grid where the grid simulates the electricity distribution and transition network and power plants. It is also possible that many customers have peer-to-peer electricity lines where direct peers are referred to as neighbors.

For the purposes of this study we consider the case where the environment is simulated at intervals of 1 hour and the demand profile captures expected customer behavior over 24 hours.

The following subsections describe the components of the smart grid problem simulation in more detail; however, many details considered in the simulation are beyond the scope of this study.

4.1.1 Random Model

Key stochastic simulation parameters are simulated by *Random Models* (RM). Depending on requirements, a RM can be based on either a function or statistical data. The function or data captures the expected value of the stochastic parameter being modeled in the form of $f(t)$ at given time t; in other words the function or data represents the profile of the simulated variable. In its simplest form, the RM uses $f(t)$ as a seed to find the simulated stochastic parameter value $\Gamma(t)$ as:

$$\Gamma(t) = c\psi(f(t), \sigma) \tag{4.1}$$

where ψ is the normal distribution with mean $f(t)$ and standard deviation σ and c is a scaling constant.

4.1.2 Generators

Power generators of any type may be simulated. The main differentiating factor of a generator is its generation profile. For our purposes we will focus on wind turbine and solar cell generators with the profiles shown in Figure 4.1 and Figure 4.2.

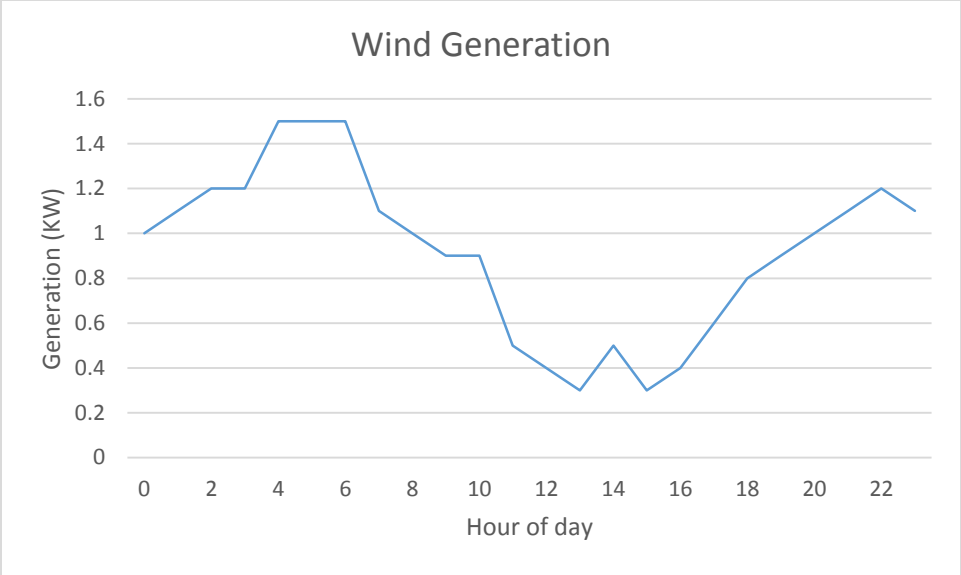


Figure 4.1: The generation profile of an average wind generator; the profile is primarily dependent on wind speed.

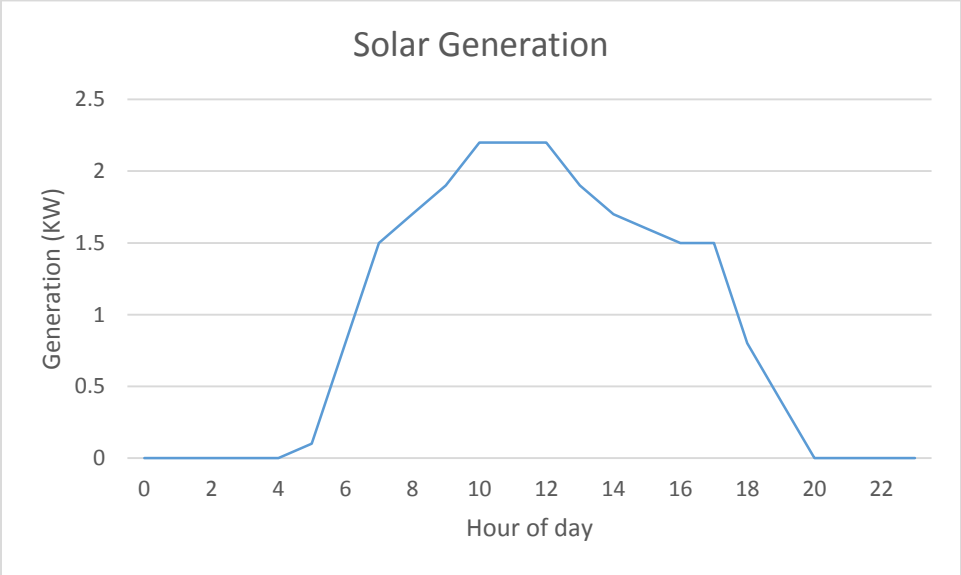


Figure 4.2: The generation profile of an average solar cell grid; the profile is primarily dependent on sunlight intensity.

Generators are simulated by RM. Let $g_i(t)$ refer to the generation of the generator of customer i at time t.

4.1.3 Storage Devices

The storage device is the primary buffering resource in the smart grid. Storage devices simply store electrical power for a later time and can displace demand by forwarding it to an earlier time. Storage devices of any type, which include batteries, fuel cells, mechanical displacers, can be simulated. For the purposes of this thesis we focus only on the capacity S_i^c, free capacity S_i^f, and used capacity S_i^u parameters of a storage device. These parameters are related by:

$$S_i^c = S_i^f + S_i^u \tag{4.2}$$

4.1.4 Customers

Each customer may have up to one generator and/or one storage device. Every customer in the smart grid has an electrical management system which is responsible for autonomously managing the customer's resources such that the customers load is satisfied with minimal cost. Every management system is modeled by an agent in the simulation.

Every customer has a load profile which describes the load behavior of the customer. The load of customer i at time t is simulated by the RM, $l_i(t)$. The load of a customer consists of a mission critical component, which is essential to the customer such that it must be satisfied immediately upon request, and an uncritical component, which can be postponed to a later time without interrupting the activities of the customer but must be eventually satisfied. Examples of critical loads are turning on lights at night time, turning on the water heater just before the use of

heated water, and recording a TV program. Examples of uncritical loads are turning on the water heater or washing and drying cloth despite there being no intended use of heated water or clean cloths anytime soon. It is assumed that all uncritical loads must be satisfied statistically over the duration of the expected demand period. This thesis only investigates the simple 24 hour daily periodic behavior of customer demand. For the sake of simplicity, instead of a RM we use a constant λ_i to identify the critical fraction of the customer load. Therefore the critical and uncritical load of a customer are $l_i^c = \lambda_i l_i(t)$ and $l_i^u = (1 - \lambda_i) l_i(t)$ respectively.

Different classes of customers, in particular commercial, industrial, and residential customers are differentiated based on their load profiles, generators, and storage devices. The load profiles of the various customer classes are as shown in Figure 4.3, Figure 4.4, and Figure 4.5.

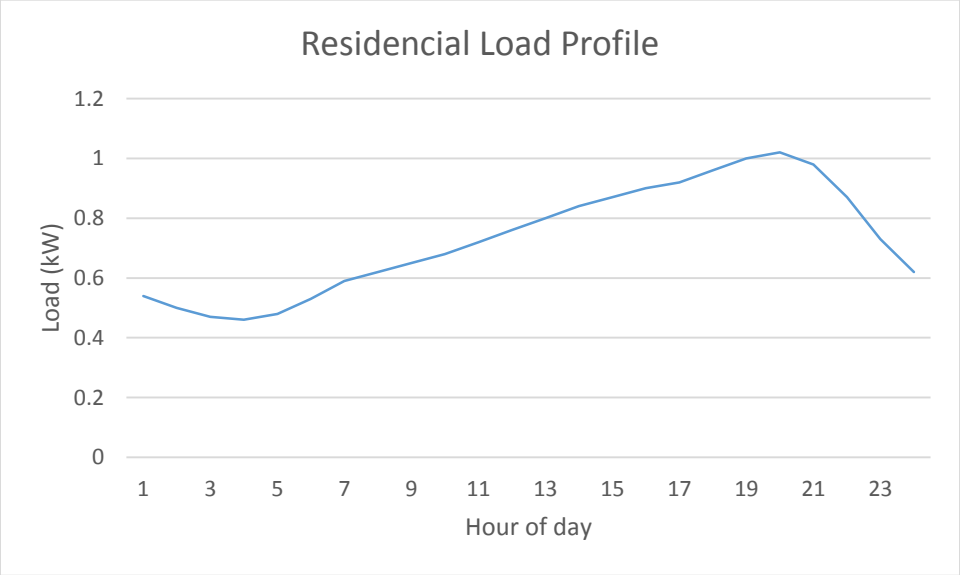


Figure 4.3: The residential load profile.

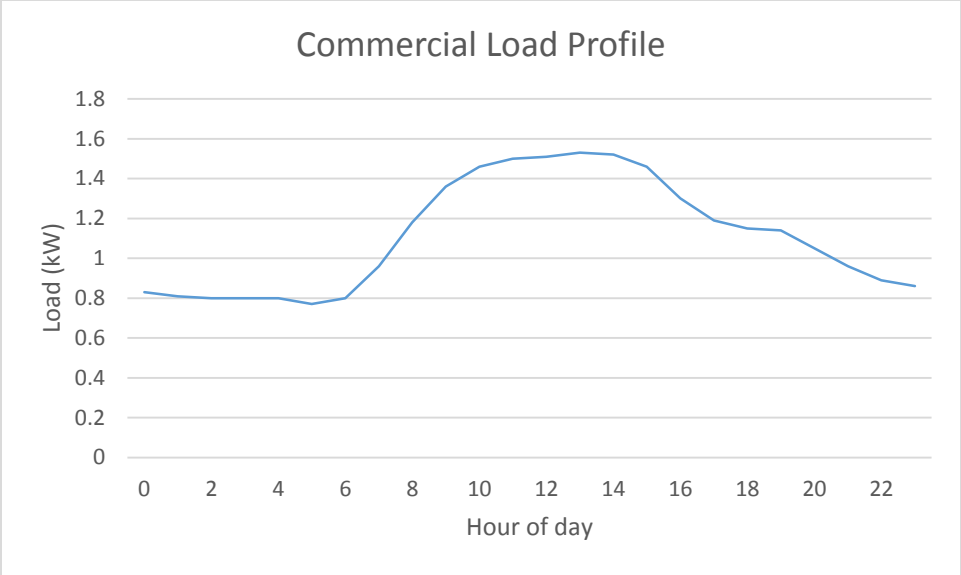


Figure 4.4: The commercial load profile.

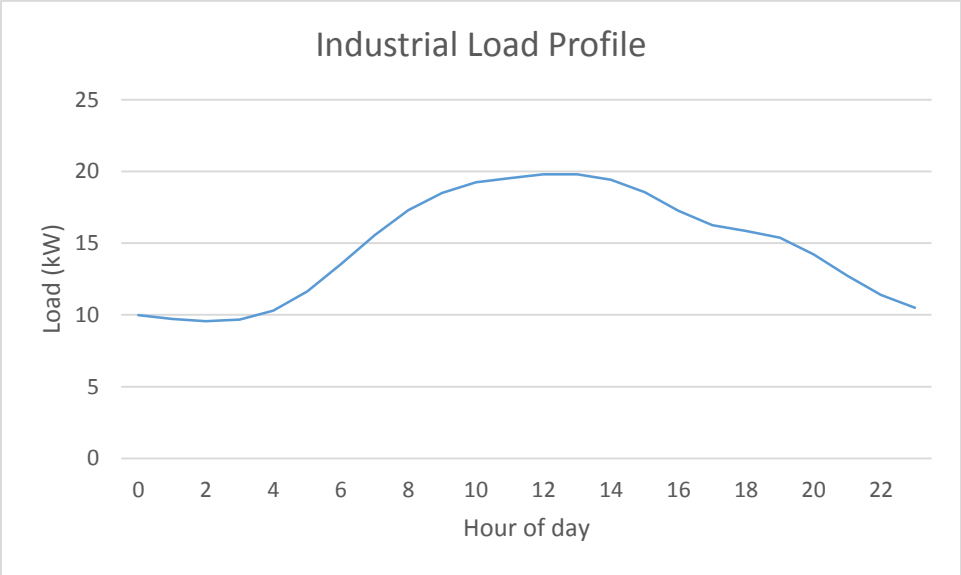


Figure 4.5: The industrial load profile.

4.1.5 Neighbors

Customers may be directly connected by a limited number of private peer-to-peer transmission lines to other customers effectively forming a graph where customer

agents are nodes and connections are edges. Let a_i be an agent i in such a graph, then $N^h(a_i)$ is the graph theoretic h-neighborhood of a_i which is a set containing all agents reachable from a_i using at most h hops. For convenience let $N(a_i) = N^1(a_i)$.

The simulation supports direct power trades between any a_i and the agents in $N(a_i)$. Trade rates in the neighborhood are advantageous over trade rates with the grid for both buyer and seller.

4.1.6 The Grid

The purpose of the grid is to simulate the electricity rate of the system as well as be a final source and destination for demand. The grid represents all elements of the power system remaining from the customers and their resources. The grid includes the transmission and distribution networks, power plants, load serving entities (LSE), independent system operators (ISO) etc. All demand which is not managed by the agents themselves must be managed by the grid: as a result all excess electricity or electrical deficiency is respectively bought by or from the grid. Details beyond how the electricity rates are determined by the grid are beyond the scope of this thesis.

It has been shown that electricity valuation is dominated by a direct exponential relationship with global demand [18]. Let the global demand modification be:

$$\Delta(t) = \sum_i \delta_i(t) \tag{4.3}$$

and the global demand be:

$$D(t) = \sum_i d_i(t) \tag{4.4}$$

and the global modified demand or global demand imposed on the grid be:

$$D_\Delta(t) = \sum_i d_i(t) + \delta_i(t) \tag{4.5}$$

where i indicates a particular agent. The grid must compute the base electricity rate $r(t)$ before any trades can be made; however, $D_\Delta(t)$ cannot be determined until all trades in hour t are complete. As a results the gird must compute $r(t)$ based on an estimation of the expected demand $\widetilde{D}_\Delta(t)$. The manner in which the simulation computes estimation will be described shortly. For the purposes of this thesis we consider the following simple base electricity rate function:

$$r(x) = c_1 e^{r_1 x} + c_2 e^{r_2 x} \tag{4.6}$$

where $x = \widetilde{D}_\Delta(t)$ and the constants c_1, r_1, c_2 and r_2 are selected so $r(t)$ fits average observed electricity rates as a function of demand. Equation (4.6) does not capture the expected electricity rate behavior when $\widetilde{D}(t) < 0$ nor is this condition expected since as such the grid would become a consumer for any t satisfying the condition. Therefore, the grid is simulated such that at any moment the total amount of power the grid buys from the agents is never more than what the grid expects to sell to the agents.

As a result of the direct exponential relationship of $r(x)$ with respect to global demand, $r(x)$must be monotonically increasing, and therefore after fitting $r(x)$ to observations, c_1, r_1, c_2 and r_2 must, without loss of generality, satisfy:

$$r_1 \geq r_2 \quad r_1 \geq 0 \quad c_1 > 0 \quad c_1 + c_2 > 0 \tag{4.7}$$

Figure 4.7 shows a general graph of $r(x)$.

Since the number and type of customers may vary over simulation instances, the average amount of demand from the grid will also vary as a result. Therefore, c_1, r_1, c_2 and r_2 must be recomputed for each simulation instance so that $r(x)$ fits observations despite variations in expected total periodic global demand for each simulation. Refitting $r(x)$ for each simulation instance allows for the results of different instances to be comparable. Refitting $r(x)$ can be interpreted as assigning a generation capacity to the grid proportional to the expected customer demand of any particular simulation instance.

In general the grid profits when trading with agents. The grid buys electricity from agents at a lower rate than the rate at which it sells electricity to agents. For simplicity, the system simulates the grid's buying price from the base price $r(x)$ as:

$$r_b(x) = (1 - \kappa)r(x) \tag{4.8}$$

and the selling price as:

$$r_s(x) = (1 + \kappa)r(x) \tag{4.9}$$

where $0 < \kappa < 1$. As explained earlier agents trade at an electricity rate which is advantageous to the grid, otherwise there is little motivation for the agents to invest in a peer-to-peer network and no motivation to trade amongst each other when the grid is functioning. Therefore, any trade among agents has an electricity rate greater than $r_b(x)$ and less than $r_s(x)$. For simplicity we let all agent-agent trades to be conducted at the base electricity rate $r(t)$.

4.2 Confirming Problem Properties Are Satisfied

This sections confirms that the simulation environment indeed captures the key properties of the problem class we are interested in investigating. The smart grid problem inherently captures properties 1 and 2 as discussed in Section 1.3; as such, the simulation also captures these properties. Since agents of the smart grid can either advance their load using their storage equipment or can postpone low priority loads to a future time, property 4 is also satisfied by the simulation. Property 3 requires that the expenses of satisfying customer electricity loads in general be lower when global demand from the grid and demand from other agents is more stable. The remainder of this section provides a high level discussion of why (4.6), (3.8) and (4.9) satisfy property 3.

Since electricity rates increase exponentially as global demand increases, it is intuitive that reducing peak global demand will reduce electricity rates at those times drastically while increasing global demand at times where global demand is relatively lower results in a relatively small increase in electricity rates. Let us consider that the total amount of any agents demand over time is not compromised. Then, reducing

peak global demands requires increasing periods with low global demand, intuitively suggesting that reducing global demand fluctuation without reducing the net demand of any agent over time can drastically reduce electricity rates.

Global demand fluctuation can be measured in many ways. Regardless of the measure being used, global demand fluctuation decreases as the distance between the maximum and minimum global demand decreases.

Let ϕ be an upper bound on the arithmetic[2] range of $D_\Delta(t)$ over T as follows:

$$0 \leq \max_{t\in T} D_\Delta(t) - \min_{t\in T} D_\Delta(t) \leq \phi < \max_{t\in T} D(t) - \min_{t\in T} D(t) \tag{4.10}$$

In order for the total global demand to be conserved after demand modifications it must be:

$$\bar{D}|T| = \int_T \widetilde{D}(t)dt = \int_T \widetilde{D}_\Delta(t)dt \tag{4.11}$$

as a result of which follows:

$$\bar{D} = \frac{1}{|T|}\int_T \widetilde{D}(t)dt = \frac{1}{|T|}\int_T \widetilde{D}_\Delta(t)dt \tag{4.12}$$

Consequently:

$$\min_{t\in T} \widetilde{D}_\Delta(t) \leq \bar{D} \leq \max_{t\in T} \widetilde{D}_\Delta(t) \tag{4.13}$$

and therefore:

[2] Not to be confused with the range of a function.

$$\lim_{\phi \to 0} \widetilde{D}_{\Delta}(t) = \overline{D} \quad \forall\, t \in T \tag{4.14}$$

That is the global demand resulting from demand modifications that remove global demand fluctuation must be exactly the mean of the original unmodified global demand. This result is general and applies to any problem in the problem class under consideration.

For illustrative purposes let the global demand $D(t)$ and the base electricity rate $r(x)$ over T be as in Figure 4.6 and Figure 4.7 respectively.

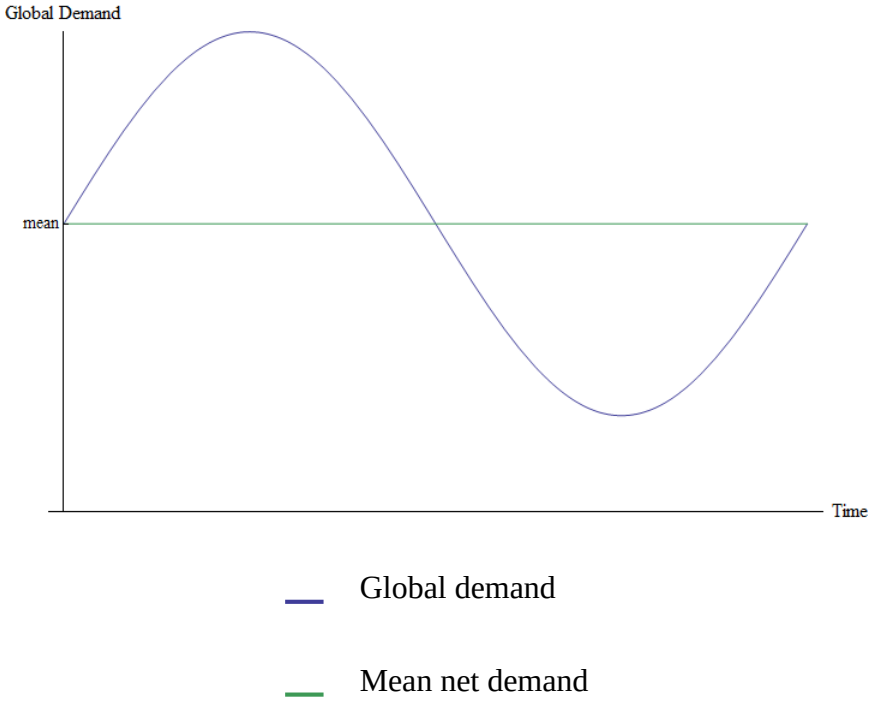


Figure 4.6: A simple sinusoidal global demand **D(t)** over **T** for illustrative purposes.

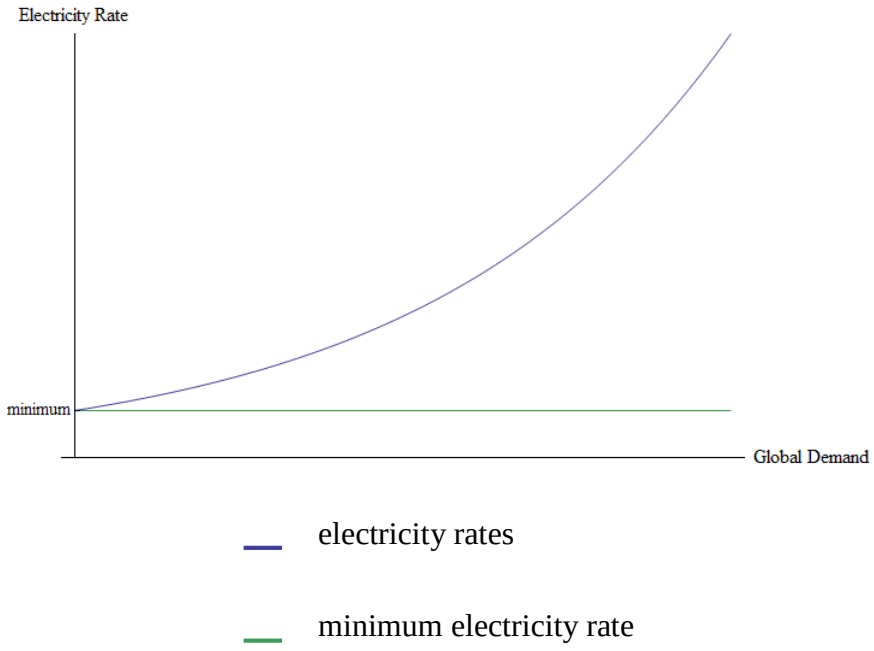


Figure 4.7: An illustration of a general electricity valuation function.

Let $\hat{T}$ and $\check{T}$ be defined as the times when global demand is over average and under average respectively:

$$\hat{T} = \{t \in T | D(t) > \bar{D}\} \tag{4.15}$$

$$\check{T} = \{t \in T | D(t) < \bar{D}\} \tag{4.16}$$

Let $\Delta(t) = \mu$, then $D_\Delta(t) = D(t) + \mu$. For any $\hat{t} \in \hat{T}$ and $\check{t} \in \check{T}$, since $r(x)$ is monotonically increasing, it must be $r\big(D(\hat{t})\big) > r\big(D(\check{t})\big)$. Considering restriction (4.7) and the expected fact that $D(t) \geq 0$, it follows that:

$$c_1 e^{r_1(D(\hat{t})+\mu)} + c_2 e^{r_2(D(\hat{t})+\mu)} > c_1 e^{r_1(D(\check{t})+\mu)} + c_2 e^{r_2(D(\check{t})+\mu)}$$

$$\rightarrow c_1 e^{r_1\mu}\left(e^{r_1 D(\hat{t})} - e^{r_1 D(\check{t})}\right) > c_2 e^{r_2\mu}\left(e^{r_2 D(\check{t})} - e^{r_2 D(\hat{t})}\right) \quad (4.17)$$

$$\rightarrow c_1 r_1 e^{r_1\mu}\left(e^{r_1 D(\hat{t})} - e^{r_1 D(\check{t})}\right) > c_2 r_2 e^{r_2\mu}\left(e^{r_2 D(\check{t})} - e^{r_2 D(\hat{t})}\right)$$

As a result it must be:

$$c_1 r_1 e^{r_1(D(\hat{t})+\mu)} + c_2 r_2 e^{r_2(D(\hat{t})+\mu)} > c_1 r_1 e^{r_1(D(\check{t})+\mu)} + c_2 r_2 e^{r_2(D(\check{t})+\mu)} \quad (4.18)$$

Since:

$$\frac{\partial r}{\partial \mu} = c_1 r_1 e^{r_1(D(\hat{t})+\mu)} + c_2 r_2 e^{r_2(D(\hat{t})+\mu)} \quad (4.19)$$

it follows from (3.8):

$$\frac{\partial r\left(D(\hat{t})\right)}{\partial \mu} > \frac{\partial r\left(D(\check{t})\right)}{\partial \mu} \quad (4.20)$$

The interpretation of (4.20) is that the rate of change in electricity prices with respect to adjusting global demand when global demand is above average is higher than that of when global demand is under average. As a result by redistributing over average global demand to under average global demand not only does the average cost of electricity decrease but the amount of electricity demanded by agents when prices are high also gets replaced by demand when prices are lower.

Inequality (4.17) and (4.20) together indicate electricity rates must reduce when shifting demand from any time $\hat{t}$ to any time $\check{t}$ as long as $D_\Delta(\hat{t}) \geq \overline{D} \geq D_\Delta(\check{t})$ after shifting. Figure 4.8 shows the electricity rate associated with the global demand

profile of Figure 4.6 as determined by the electricity valuation function of Figure 4.7 illustrating how higher-than-average electricity demands have a much higher associated electricity rate while lower-than-average electricity rates have a relatively negligible lower electricity rate and as such illustrated why property 3 holds.

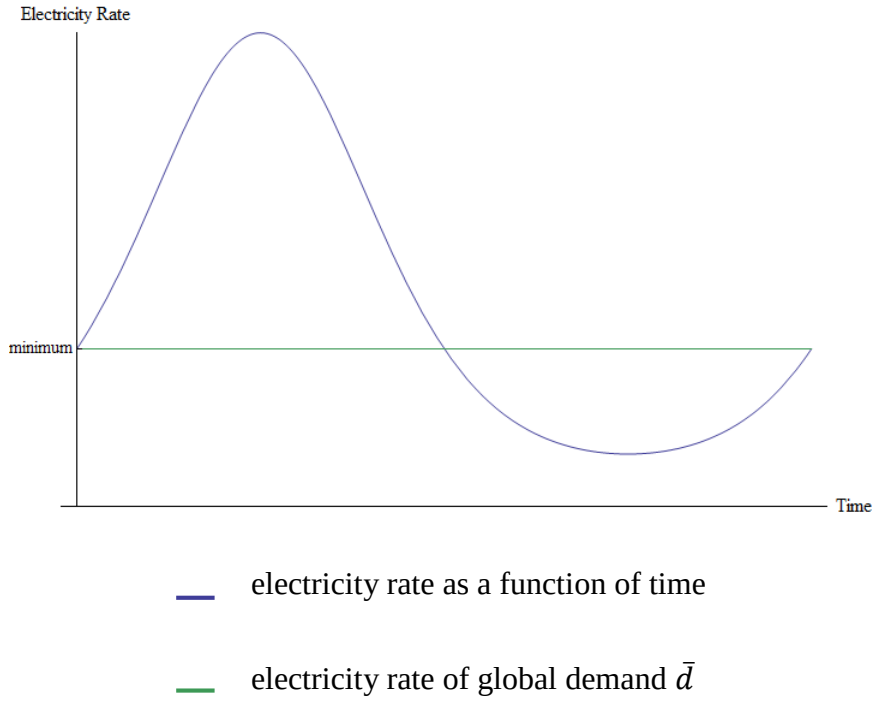


Figure 4.8: The electricity rate associated with the global demand profile. Under the claim restrictions the electricity rate associated with $\bar{\mathbf{d}}$ is minimum.

CHAPTER 5 SMART GRID SIMULATOR

This chapter describes the relevant details of the multi agent smart grid simulator developed for this thesis. The simulator was developed in a general manner as to allow investigations in various aspects of multi agent smart grids scenarios and MASs as a whole. In particular, the simulator supports various electricity and resource auctions, infrastructure failure, agent algorithms, and social structures, in addition to configuring various agent classes and resources. The simulation framework was built on repast symphony [36].

This chapter is concerned with describing the configuration parameters which determine a smart grid simulation environment. This is key in understanding the context of the results and observation in the following chapters.

5.1 Random Model

Central to describing the simulation environment is the concept of *random models*. Random models are used to model time dependent stochastic properties such as the amount of load each home needs to satisfy over at a given moment in time. The random model can be based on any one of the following models:

1. *data model*: statistical data that models a properties behavior over time
2. *function model*: a function of time that models a properties behavior
3. random distribution

A random model uses a normal distribution with a mean corresponding to some scaling of the model's basis in order to generate a random value corresponding

to a property of interest. This allows the simulation to mimic the chaos of the real world while still preserving the macroscopic properties of interest.

Each model describes a function $f(t)$ relating the simulation moment t to a random value as described in subsection 0. A random model is constructed from $f(t)$ by:

$$\Gamma(t) = s\psi(f(t), \sigma) \tag{5.1}$$

where s is a scaling factor and $\psi(\mu, \sigma)$ is an instance from a normal distribution if standard deviation σ about the mean μ. Figure 5.1 illustrates the values generated by a random model describing wind power generation for an agent.

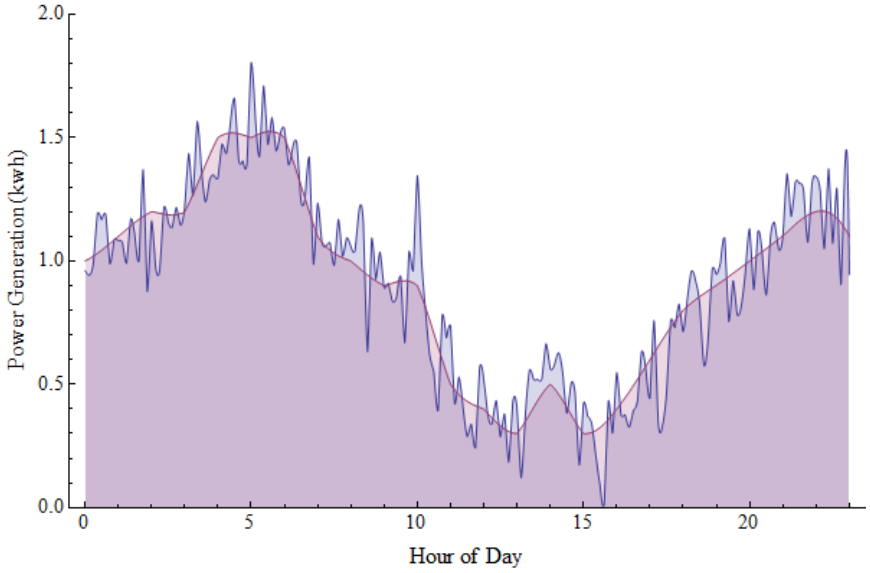


Figure 5.1: A graph of the interpolated values of a random model with $\sigma = 0.15$ describing the wind power generation profile of a particular agent (blue), and the expected wind power generation profile for any agent described by f(t) from a data model (red).

In some simulator versions, it is possible to synthesize an agent's prediction of $\Gamma(t + \Delta t)$ at Δt in the future by:

$$\Gamma_{a_n}(t, \Delta t) = \psi(\Gamma(t), k \cdot \log(\Delta t + 1)) \quad (5.2)$$

where $k > 0$ is constant selects at simulation design. Synthesizing agent prediction can be replaced in favor of having agents actually performing predictions. Figure 5.2 illustrates the increase in prediction error as agents make predictions more distant into the future.

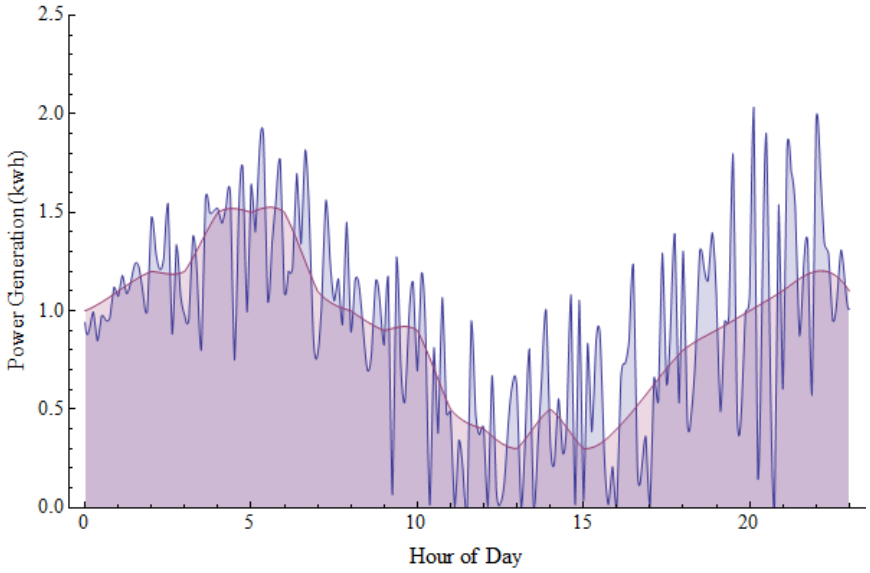


Figure 5.2: A graph of the interpolated values predicted by an agent from the random model (blue) and the expected values (red) shown in Figure 5.1. The horizontal axis indicates the number of hours into the future the prediction is made. The error in prediction increases logarithmically as the agent makes more distant predictions into the future.

5.2 Models

5.2.1 Data Model

As the name suggests, statistical data is used to directly define $f(t)$. Table 5.1 and Figure 5.3 provide an example data model describing expected wind generation for a given hour of a day.

Table 5.1: Data model describing the expected wind generation for a given hour of any day.

Hour of day	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
Generation	1	1.1	1.2	1.2	1.5	1.5	1.5	1.1	1	0.9	0.9	0.5	0.4	0.3	0.5	0.3	1	1.1	1.2	1.2	1.5	1.5	1.5	1.1

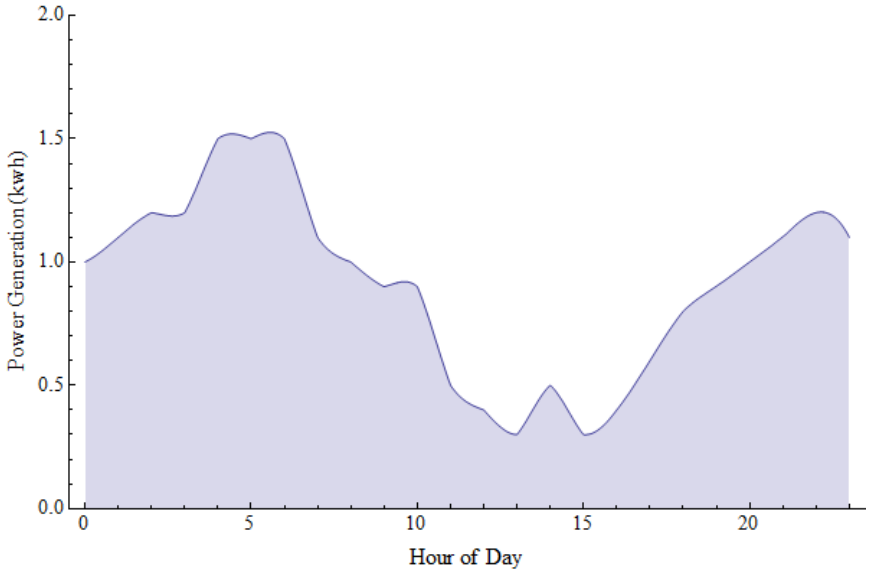


Figure 5.3: Graph of the interpolation of the data model in Table 5.1.

5.2.2 Function Model

A function model is defined by:

$$f(t) = (g \circ h)(t) \tag{5.3}$$

where $h(t)$ can be any property during a simulation at time t, such as average agent demand, and $g(x)$ is any function describing the relationship between h and the property being modeled. Eq. 5.4 and Figure 5.4 provide an example function model describing the electricity rate for a given time.

Eq. 5.4: A function model f(t) describing the price at time t where h(t) is the average power demand during a simulation and $g(x) = c_1\, e^{r_1 x} + c_2\, e^{r_2 x}$ with $c_1 = 2.298$, $r_1 = 0.5805$, $c_2 = 0.3028$, $r_2 = 2.537$ describes the relation between average demand and electricity rate.

$$f(t) = (g \circ h)(t) = c_1 e^{r_1 h(t)} + c_2 e^{r_2 h(t)}$$

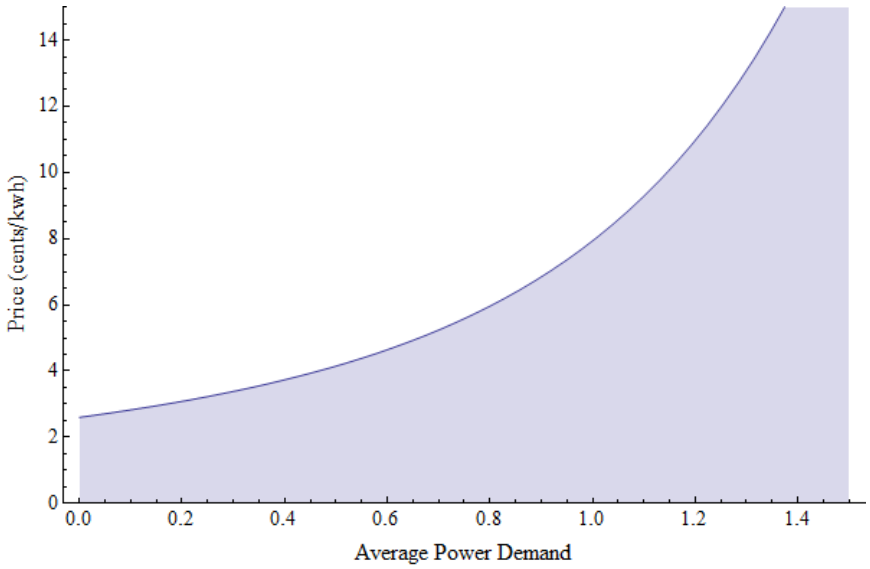


Figure 5.4: A graph of g(x) from Eq. 5.4.

5.2.3 Radom Distribution

As the name suggests, $f(t)$ is defined by a random distribution γ. The distribution is used to construct an infinite random set of values S, and f randomly maps $T \to S$. A random distribution can be any one from Table 5.2.

Table 5.2: The list of supported random distributions from which a random model may be constructed.

B through E	E through N	N though Z
Beta	Exponential	Normal
Binomial	Exponential Power	Poisson
Breit Wigner	Gamma	Poisson Slow

Breit Wigner Mean Square	Hyperbolic	StudentT
Chi Square	Hyper Geometric	Uniform
Constant	Logarithmic	Von Mises
Empirical	Log Normal	Zeta
Empirical Walker	Negative Binomial	

5.3 Load Factor

The key measurement we us to capture the degree in which an agent realizes its goals is the load factor (LF). LF captures the amount of fluctuation in demand over a given period of time. As captured by Definition 5.1, LF is the average demand over the maximum demand for a given period; as such, for nonnegative demand the LF measure is confined between 0 and 1 indicating minimum and maximum performance respectively.

Definition 5.1: Load Factor

$$\boldsymbol{LF = \frac{\int_{t_0}^{t_1} d(t)dt}{(t_1 - t_2) \max\limits_{t \in (t_0, t_1)} d(t)}}$$

5.4 Simulation Time

The simulation is based on discrete units of time; however, as indicated in the following sections, the simulations environment can be configured to support any unit of time limited only to simulation hardware. In the case where environment volatility must be captured the simulation time interval can be configured as small as necessary

to provide result accuracy. For the purposes of this thesis we configure the simulations to support 1 hour intervals.

5.5 Simulation Environment Configuration

The simulation environment is very flexible leading to complicated configuration scenarios. Since we are interested in isolating most of the environment parameters in our investigations, it is important to know the simulators general configurability but details are omitted.

It is possible to configure the simulator to either simulate supplier entities using individual agents or to synthesize supplier entity dynamics through a singular grid agent (see Table 5.3).

Table 5.3: General configurations options governing the simulation of supplier side dynamics.

Simulated Suppliers	**Synthetic Suppliers**
several power plants responsible for generating power	A singular grid agent responsible for power generation, pricing, and distribution
power auctioning agents responsible for pricing and distribution	

We omit the details of configuring simulated suppliers and focus only on describe general configurability of the grid agents. Simplified this way, every simulation environment is constructed from a grid agent and a nonempty set of customer agents.

Most of the configurable components are constructed from random models, random distributions and value adjusters. The simulation components are configured by XML as described in the following subsections.

5.6 Configuring Random Models

Random models are not defined individually but instead are defined in classes. When a simulation starts, the necessary random models are randomly drawn from instances of random model classes. Random model classes are defined using the following XML structure of Pseudo Code 5.1.

```
<[PropertyName]>
   <Failure>…</Failure>
   <Repair>…</Repair>
   <Mean>…</Mean>
   <MeanFactor>…</MeanFactor>
   <StandardDeviation>…</StandardDeviation>
   <PredictionError>…</PredictionError>
   <CostFactor>…</CostFactor>
   <Adjuster>…</Adjuster>
</[PropertyName]>
```

Pseudo Code 5.1: The configuration template for a random model.

The *[PropertyName]* is a placeholder for the XML friendly name of the environment property being simulated by the random model (i.e. *PowerRate* simulating electricity rates, *GenerationModel* simulating power generation, etc).

The element:

1. *Failure*: is a random distribution of failure events
2. *Repair*: is a random distribution of repair time durations
3. *Mean*: is a model describing $f(t)$ for random model instances

4. *MeanFactor*: a random distribution of s for random instances
5. *StandardDeviation*: a random distribution of σ for random model instances
6. *PredictionError*: a random distribution of k for random model instances
7. *CostFactor*: a random distribution of unit cost per final value for random model instances
8. *Adjuster*: a collection of nested adjusters each adjusting the final random model values

5.6.1 Configuring Data Models

The following XML structure describes a data model:

```
<Mean type="DataModel" format="json">
   [1, 1.1, 1.2, 1.2, 1.5, 1.5, 1.5, 1.1, 1, 0.9, 0.9, 0.5, 0.4,
    0.3, 0.5, 0.3, 0.4, 0.6, 0.8, 0.9, 1, 1.1, 1.2, 1.1]
</Mean>
```

Code Snippet 5.1: An example of configuring a data model.

The *type* attribute must have a value of *DataModel*. The *format* attribute indicates the format in which the data is provided, the value can be *XML*, *json*, *etc.* The example provided above describes the data model for wind generation presented in Table 5.1.

5.6.2 Configuring Function Models

The following XML structure describes function models:

```
<Mean type="ExponentialDynamicFunctionModel">
   <A value="2.298"/>
   <B value="0.5808"/>
   <C value="0.3028"/>
   <D value="2.537"/>
   <DynamicFunctionList>
      <DynamicFunction value="getNormalPredictedDemandByPeriod">
         <Adjuster>…</Adjuster>
       </DynamicFunction>
   </DynamicFunctionList>
</Mean>
```

Code Snippet 5.2: An example of configuring a function model; in particular this example configures and exponential function model.

The *type* attribute indicates the Java class which describes the function $g(x)$ of the model. Depending on the function various parameters may be configurable each in an element (i.e. A is c_1, B is r_1, C is c_2, D is r_2 from Eq. 5.4). The dynamic function list, provides a list of Java methods from the simulation which are composed together to construct $h(t)$; as in the example above, usually one such function is used. The *Adjuster* elements adjust the value returned by the dynamic function they are nested.

5.6.3 Configuring Random Distributions

The following XML is an example definition for a normal random distribution:

```
<[PlaceHolder] type=”Normal”>
   <Mean value="2" />
   <StandardDeviation value="0.2" />
   <Adjuster>…</Adjuster>
</[PlaceHolder]>
```

Code Snippet 5.3: An example of configuring a random distribution. The [Placeholder] should be replaced by the property name being modeled by the distribution.

The element name *[PlaceHolder]* is a placeholder for the component being described by a random distribution (i.e. *MeanFactor*, *CostFactor*, etc). The *type* attribute indicates the type of distribution, which can be any of the distributions in Table 5.2. Depending on the type of distribution various elements are used to provide distribution parameter values such as mean, and standard deviation for the normal distribution. The *Adjuster* elements adjust any final values as described in the following subsection.

5.6.4 Configuring Adjusters

Some randomly generated values may not be directly applicable to the property they describe. For example a normal distribution may infrequently produce negative values in which case a clamp adjuster may for such values to be mapped to zero. Adjusters are simple function which refine value and can be nested in one another.

5.7 Grid Agent Configuration

The grid agent calculates electricity rates based on its prediction of power demand for a given moment. Predictions can be done is several ways; however, we will focus on using expected aggregate demand given a set of samples from the past as the prediction mechanism.

The grid can simulate failures in distribution, generation capacity, etc. The frequency and type of failures, their extent, and their repair can be configured using the corresponding XML elements. All failures including grid failures are disabled in our simulations and their configurations will not be explained further.

The grid is defined by the *GridModel* XML element as follows:

```
<GridModel>
   <Failure>…</Failure>
   <Repair>…</Repair>
   <BlackoutRadius>…</BlackoutRadius>
   <PowerRateRandomModel>…</PowerRateRandomModel>
   <PriceAdjustment>…</PriceAdjustment>
</GridModel>
```

Code Snippet 5.4: A template for configuring a grid agent.

The element:

1. *Failure:* is a random distribution defining the failure frequency
2. *Repair:* is a random distribution defining repair durations
3. *BlackoutRadius:* is a random distribution describing the graph theoretic neighborhood of an agent from where a blackout originates
4. *PowerRateRandomModel:* a random model describing electricity rates
5. *PriceAdjustment:* a constant value m such that given base electricity price p, the grid by back price is $(1 - m)p$ and the sales price to customers is $(1 + m)p$

5.8 Customer Agent Configuration

Customer agents are not configured individually but in classes of similar agents. Each customer agent class is defined by an *AgentGnerator* XML element which has a *name* and *population* attribute (see: Code Snippet 5.5). Each agent has the following configurable components:

1. Load profile
2. Load suspension profile
3. Generation profile

4. Storage system
5. Foresight

Each of these components are in turn selected from a corresponding component classes. Each component class is specified by a corresponding element in the *AgentGenerator* element as shown in Code Snippet 5.5. At start up, the simulation generates the number of customer instances specified by the population attribute, each time selecting an instance of each agent component from the corresponding component class.

```
<AgentGenerator name="Commercial Agents" population="50">
   <Failure>…</Failure>
   <Repair>…</Repair>
   <SuspendableModelGenerator>…</SuspendableModelGenerator>
   <LoadModelGenerator>…</LoadModelGenerator>
   <GenerationModelGenerator>…</GenerationModelGenerator>
   <StorageGenerator>…</StorageGenerator>
   <ForesightRandomParameter>…</ForesightRandomParameter>
</AgentGenerator>
```

Code Snippet 5.5: Example XML describing a class of commercial agents which may represent hospitals, shopping malls, and other such business, from which 50 instances will be randomly generated and added to the environment.

The element:

- *Failure:* is a random distribution defining the failure frequency
- *Repair:* is a random distribution defining repair durations
- *SuspendableModelGenerator*: is a random distribution and defines the class of load suspension profiles
- *LoadModelGenerator*: is a random model and defines a class of load profiles

- *GenerationModelGenerator*: is a random model and describes generation profiles
- *StorageGenerator*: describes a storage class, see 5.9 for more details
- *ForesightRandomParameter*: describes a class of values from which an agent's foresight is drawn

5.9 Storage System Configuration

Instances of a customer agent's storage are not directly configured, instead a storage class from which storage instances are randomly drawn from are configured. The following XML structure describes a storage class:

```
<StorageGenerator>
   <Failure>…</Failure>
   <Repair>…</Repair>
   <Capacity>…</Capacity>
   <Efficiency>…</Efficiency>
   <Retention>…</Retention>
   <CostFactor>…</CostFactor>
</StorageGenerator>
<StorageGenerator>
   <Failure>…</Failure>
   <Repair>…</Repair>
   <Capacity>…</Capacity>
   <Efficiency>…</Efficiency>
   <Retention>…</Retention>
   <CostFactor>…</CostFactor>
</StorageGenerator>
```

Code Snippet 5.6: A template for configuring a storage system.

There element:

- *Failure:* is a random distribution defining the failure frequency

- *Repair:* is a random distribution defining repair durations
- *Capacity*: is a random distribution defining possible storage capacities
- *Efficiency:* is a random distribution defining the charging/discharging efficiency of a storage unit
- *Retention:* is a random distribution defining the fraction of stored power retained at any moment
- *CostFactor:* is a random distribution defining the cost per capacity of maintaining the storage unit at any momenta

5.10 Network Configurations

As described in previous chapters, the network distributing power and communication among agents forms a graph. In general all customer agents are connected to the grid agents. The graph determining how customer agents are connected to one another is generated randomly and is governed by the following options:

- *Minimum Connectivity:* the minimum number of customer agents any particular customer agent must be connected.
- *Maximum Connectivity:* the maximum number of customer agents any particular customer agent can possibly be connected.
- *Strongly Connected:* whether or not there is a path between any pair customer agents which do not involve the grid agents.

5.11 Algorithm Specific Configurations

Depending on personal and collaborative strategies many other configurations exist. These configurations will be discussed when the simulation evaluating a particular strategy is considered. These configuration include the preferences of a customer agent when conducting business with others, the amount and type of agent memory and etc.

CHAPTER 6 SIMULATIONS

In order to understand the dynamics and tradeoffs of the proposed approach, we investigate the dependencies the system and agents have toward achieving their goals. In other words, we investigate the autonomy of the agents while they follow the proposed approach. In that direction, we carefully craft and configure three simulation cases each following the proposed approach but with varying levels of agent collaboration. More specifically, in this chapter we will:

1. identify a set of values for the simulation variables such that they minimally limit the effectiveness of each simulation case
2. define simulation cases such that they only vary in autonomy

To accomplish each of the aforementioned items respectively we:

1. conduct some preliminary investigations into the simulation sensitivity of problematic variables
2. we define each simulation case such that the agents of each vary only on what their goals can possibly depend

The next section informally clarifies some concepts that are used in accomplishing item 2 of the above lists. We will revisit these concepts in more detail at the beginning of Chapter 8.

6.1 Controlling Autonomy

To summarize what was mentioned in Section 2.2, agent autonomy is simply the degree in which an agent is dependent on external factors in order to achieve its goals to some appreciable extent. To control autonomy without changing the approach, we

can only vary the dependencies since the approach defines the goals. It is worth emphasizing that the notion of goal being used here is general in the sense that it does not change; particularly, a goal is not dependent on any particular state of the environment. The external dependencies an agent has in achieving its goals irrespective of its environmental state is limited to the information and control over its external environment. We refer to the information and control and agent has over the environment external to itself as the agent's *authority* over its environment or simply the agent's authority. We will revisit the concept of authority in Chapter 8, but until then we will be explicit about agent information and control when referring to the notion of authority.

Agent authority in terms of the smart grid simulation cases considered in this thesis, refers to:

1. the information an agent has regarding the demand profile of other agents
2. the control the agent has over buffering resources other than its own

The reason other forms of information and control are not relevant in the smart grid simulations is because the methodology and approach the simulations follow only rely on demand profiles and buffering resources.

Therefore, we control the autonomy of the simulation cases by controlling agent access to external demand profiles and buffering resources. This is possible since an agent must achieve its goals while only relying on the authority it has available. Consequently, in the most extreme case, if an agent is given no authority, it

will be forced to achieve its goals independent of everything external to itself; consequently, such an agent must be fully autonomous.

6.2 Simulation Case Overview

Table 6.1 describes the three elected simulation classes each corresponding to a general category of agent cooperation.

Table 6.1: Simulations grouped by collaboration category of the agents.

	Autonomy	**Distribution**	**Resource Complexity**	**Communication Overhead**
Solo **Agents are completely autonomous**	Independent	Distributed	Low	None
Neighbors **Agents form ad-hoc bonds**	Loosely coupled without commitment	Distributed	Medium	Low
Unity **Agents merge into one entity**	Tightly coupled	Fully centralized	Generally Intractable	Generally Intractable

The results of these simulations give insight into the relationship between autonomy and optimality. The following subsections describe the simulations and

their configurations. Each configuration includes both environment and agent algorithm configurations.

Across all simulations the reduction of demand fluctuation based on the approach described in Chapter 4 is held constant. Only the degree of collaboration, particularly that defined by the exchange of information and the exchange of impact on the environment (i.e. agent *A* acts on its local environment on behalf of agent *B* who cannot directly or efficiently act on the local environment of *A*) is allowed to vary.

6.3 Common Configuration

The environment in all the simulations is composed of the agents in Table 6.2.

Table 6.2: Agent composition of the simulations.

Number	Agent Type	Description
1	Grid	Synthesizes the market, distribution, suppliers
200	Smart homes	Simulates a household who have installed a system to manage their load, storage, generation, and transactions with the grid and possibly neighbors
200	Normal homes	Simulates a traditional household who manually manages their load and purchases from the grid but may install a simple device to manage purchases from neighbors as well

50	Modern commercial entities	Simulates commercial environments like shopping malls, banks, and hospitals which manage their load, storage, generation, and transactions with the grid and possibly neighbors through system similar to smart homes
50	Traditional commercial entities	Simulates commercial environments like shopping malls, banks, and hospitals which manually manage their load and purchases from the grid but may install a device to manage purchases from neighbors as well
25	Industrial entities	Simulates industries heavily dependent on electricity such as aluminum and chemical plants; these industries can only operate by consuming power from the grid and possibly neighbors and cannot rely on storage

6.3.1 Smart Home and Normal Home Configurations

Smart homes are configured as described in Code Snippet 6.1. The normal home follows the same configurations except that it is lacking corresponding configurations for storage and generation since normal homes do not have generation and storage. Also normal homes are unable to effectively manage their load and therefore the amount of load they can suspend for later is set to 10% of that possible by a smart home. The relationship between the smart home and normal home suspension capabilities is not relevant to the study and its effects are canceled by holding the relationship constant across simulations.

```
<AgentGenerator name="Homes Batt/Win" population="200">
   <SuspendableModelGenerator type="StaticRandomModel">
      <Mean type="ConstantModel" format="json">
         <Constant value="0.2"/>
      </Mean>
   </SuspendableModelGenerator>
   <LoadModelGenerator name="LoadModel">
      <Mean type="DataModel" format="json">
         [0.54, 0.5, 0.47, 0.46, 0.48, 0.53, 0.59, 0.62, 0.65,
          0.68, 0.72, 0.76, 0.8, 0.84, 0.87, 0.9, 0.92, 0.96,
          1, 1.02, 0.98, 0.87, 0.73, 0.62]
      </Mean>
      <StandardDeviation type="Constant">
         <Constant value="0.1"/>
      </StandardDeviation>
      <Adjuster type="Clamp">
         <Min value="0"/>
      </Adjuster>
   </LoadModelGenerator>
   <GenerationModelGenerator>
      <Mean type="DataModel" format="json">
         [1, 1.1, 1.2, 1.2, 1.5, 1.5, 1.5, 1.1, 1, 0.9, 0.9,
          0.5, 0.4, 0.3, 0.5, 0.3, 0.4, 0.6, 0.8, 0.9, 1, 1.1,
          1.2, 1.1]
      </Mean>
      <StandardDeviation type="Constant">
         <Constant value="0.1"/>
      </StandardDeviation>
      <CostFactor type="Constant">
         <Constant value="3.6"/>
      </CostFactor>
      <Adjuster type="Clamp">
         <Min value="0"/>
      </Adjuster>
```

```
    </GenerationModelGenerator>
    <StorageGenerator>
        <Capacity type="Normal">
            <Mean value="1.5"/>
            <StandardDeviation value="0.2"></StandardDeviation>
            <Adjuster>
                <Min value="1.0"/>
                <Max value="2.0"/>
            </Adjuster>
        </Capacity>
        <CostFactor type="Constant">
            <Constant value=".48"/>
        </CostFactor>
    </StorageGenerator>
    <ForesightRandomParameter type="Constant">
        <Constant value="24"/>
    </ForesightRandomParameter>
</AgentGenerator>
```

Code Snippet 6.1: Configuration of smart homes.

Figure 6.1 and Figure 6.2 shows the defining characteristic of homes; the general load and generation profile. As indicated by Figure 6.1 a typical home starts in the day with gradually increasing power requirements until it peaking around dark when the whole family is at home and lights are needed. The power requirement typically dwindle down as people rest reaching a minimum early in the morning. The home generation profile is based on that of wind turbines. No particular generation profile impacts our comparison of performance among simulations since the profile is constant over the simulations; however, the existence of generation adds to the dynamism of the environment allowing for a larger range of stabilization performance results. A more dynamic environment provides a more challenging optimization task

for the agents of the various simulations ultimately increasing possible result spreads. The wind generation profile and home load profile are more similar than that of solar generation profiles particular when considering a phase shift in the wind profile. A phase shift requires less storage and load suspension making wind energy more suitable for homes.

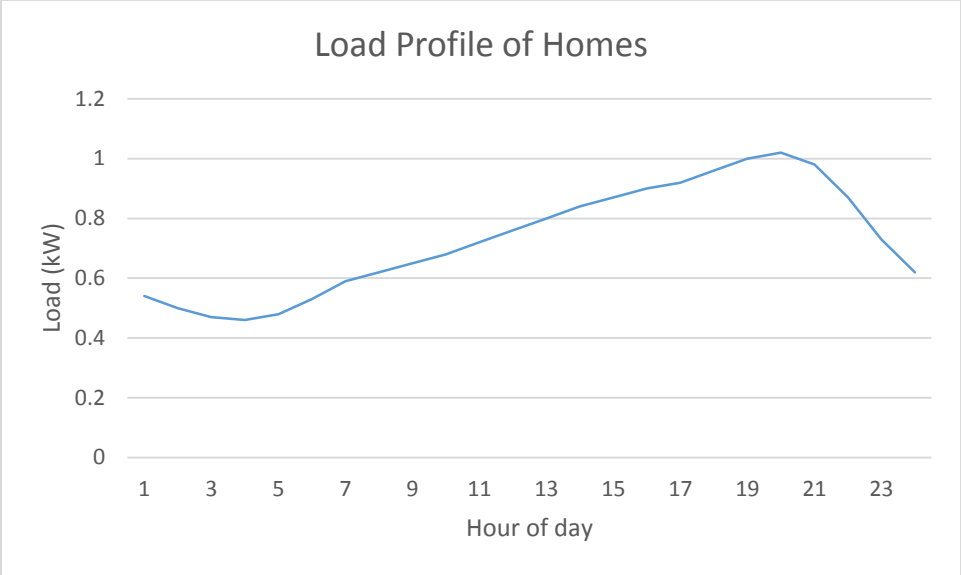


Figure 6.1: The expected home load over a day.

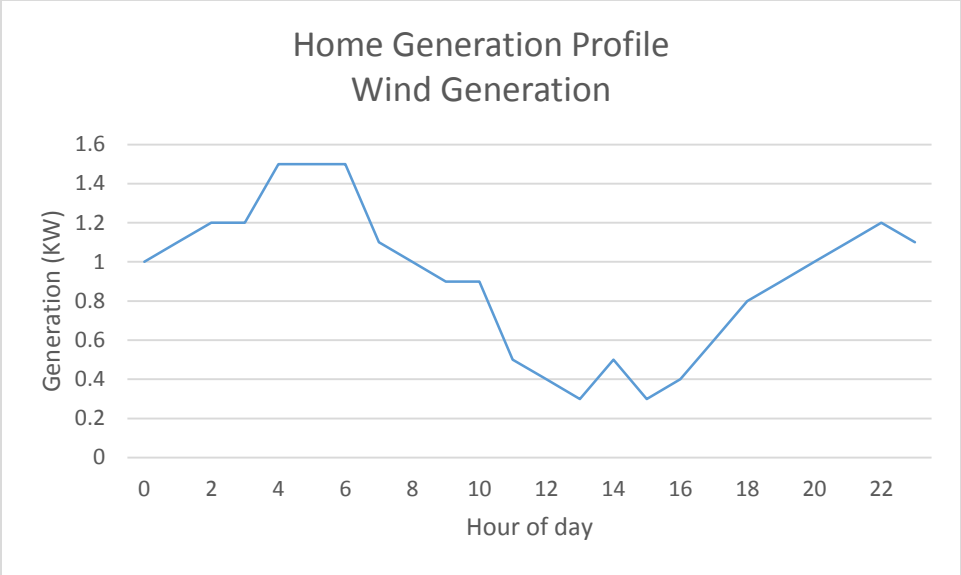


Figure 6.2: The expected home generation over a day. The generation profile is based on wind generation.

6.3.2 Modern and Traditional Commercial Entity Configurations

The modern commercial entity configurations appear in Code Snippet 6.2. Normal commercial entities are configured the same except that they lack storage and generation and their associated configurations. Load suspension capability of commercial entities are less than homes due to the more mission critical nature of the loads of commercial entities. The particular value selected for suspension capacity is of no relevance to the study and cancels out since the configurations is held constant.

```
<AgentGenerator name="Commercial Batt/Sol" population="50">
   <SuspendableModelGenerator type="StaticRandomModel">
      <Mean type="ConstantModel" format="json">
         <Constant value="0.1"/>
      </Mean>
   </SuspendableModelGenerator>
   <LoadModelGenerator name="LoadModel">
      <Mean type="DataModel" format="json">
```

```
        [0.83, 0.81, 0.8, 0.8, 0.8, 0.77, 0.8, 0.96, 1.18, 1.36,
         1.46, 1.5, 1.51, 1.53, 1.52, 1.46, 1.3, 1.19, 1.15,
         1.14, 1.05, 0.96, 0.89, 0.86]
      </Mean>
      <StandardDeviation type="Constant">
      <Constant value="0.2"/>
      </StandardDeviation>
      <Adjuster type="Clamp">
         <Min value="0"/>
      </Adjuster>
   </LoadModelGenerator>
   <GenerationModelGenerator>
      <Mean type="DataModel" format="json">
         [0, 0, 0, 0, 0, 0.1, 0.8, 1.5, 1.7, 1.9, 2.2, 2.2, 2.2,
          1.9, 1.7, 1.6, 1.5, 1.5, 0.8, 0.4, 0, 0, 0, 0]
      </Mean>
      <StandardDeviation type="Constant">
         <Constant value="0.2"/>
      </StandardDeviation>
      <CostFactor type="Constant">
         <Constant value="3.6"/>
      </CostFactor>
      <Adjuster type="Clamp">
         <Min value="0"/>
      </Adjuster>
   </GenerationModelGenerator>
   <StorageGenerator>
      <Capacity type="Normal">
         <Mean value="25"/>
         <StandardDeviation value="0.2"></StandardDeviation>
         <Adjuster>
            <Min value="20.0"/>
            <Max value="30.0"/>
         </Adjuster>
```

```
      </Capacity>
      <CostFactor type="Constant">
         <Constant value=".42"/>
      </CostFactor>
   </StorageGenerator>
   <ForesightRandomParameter type="Constant">
      <Constant value="24"></Constant>
   </ForesightRandomParameter>
</AgentGenerator>
```

Code Snippet 6.2: Modern commercial entity configuration.

Figure 6.3 and Figure 6.4 show the commercial load and generation profile respectively. Similar to homes opting for commercial agents to use solar energy has not impact on our study other than to increase the amount of dynamism and optimization challenge for agents. The load of commercial entities drastically as businesses and their customers start their daily routine; the load slowly dwindles people return home carrying their loads with them. As clear from Figure 6.3 and Figure 6.4 the commercial load profile and solar generation profile matches very well reducing storage and load management requirement.

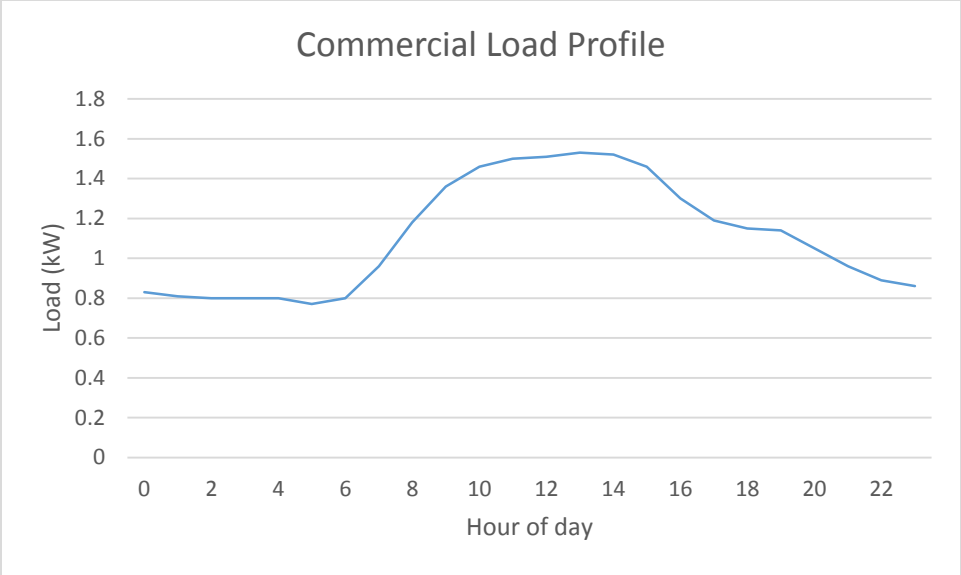


Figure 6.3: Expected commercial load over a day.

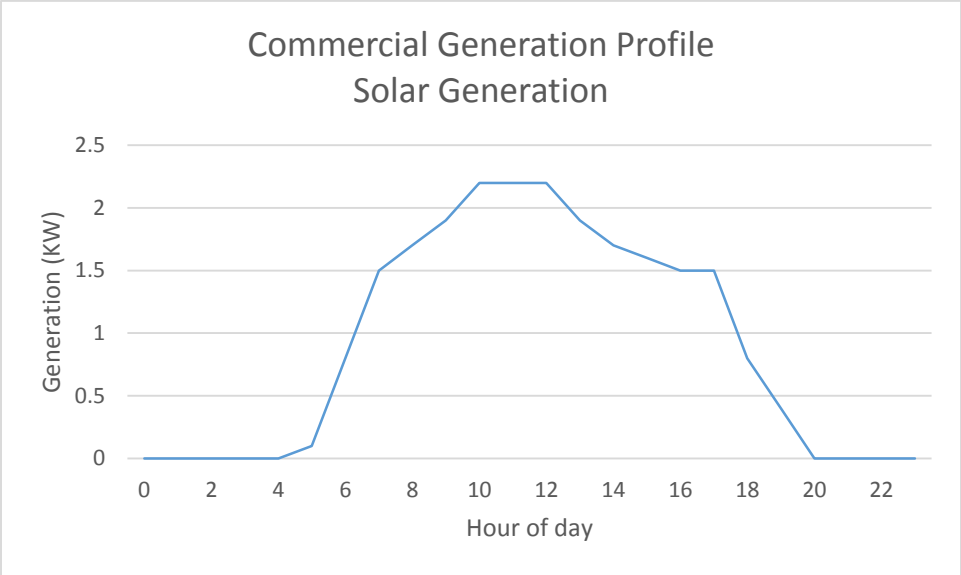


Figure 6.4: The expected commercial generation over a day. The generation profile is based on solar energy.

6.3.3 Industry Agents

The configurations for the industry agents appear in Code Snippet 6.3: Industrial agent configurations. Industrial agents can suspend a large portion of their load to off peak hours. The load profile of industries are shown in the following.

```
<AgentGenerator name="Industrial" population="25">
   <SuspendableModelGenerator type="StaticRandomModel">
      <Mean type="ConstantModel" format="json">
         <Constant value="0.5"/>
      </Mean>
   </SuspendableModelGenerator>
   <LoadModelGenerator name="LoadModel">
      <Mean type="DataModel" format="json">
         [9.99, 9.71, 9.57, 9.68, 10.29, 11.63, 13.53, 15.56,
          17.3, 18.5, 19.23, 19.54, 19.8, 19.8, 19.41, 18.54,
          17.25, 16.26, 15.84, 15.39, 14.23, 12.72, 11.38, 10.5]
      </Mean>
      <StandardDeviation type="Constant">
         <Constant value="2"/>
      </StandardDeviation>
      <Adjuster type="Clamp">
         <Min value="0"/>
      </Adjuster>
   </LoadModelGenerator>
   <ForesightRandomParameter type="Constant">
      <Constant value="24"/>
   </ForesightRandomParameter>
</AgentGenerator>
```

Code Snippet 6.3: Industrial agent configurations.

Figure 6.5 shows the load profile of industry entities. The load profile is similar to that of commercial entities; however, power demand at any given hour is much higher in scale.

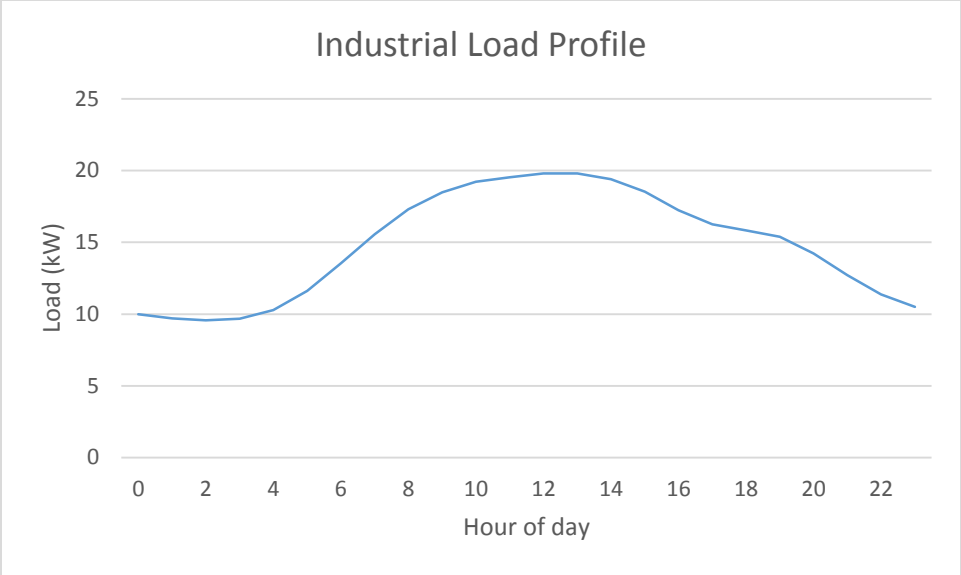


Figure 6.5: The expected power requirements of industrial entities over a day.

6.3.4 General Environment Configurations

As mentioned earlier some general environment configuration parameters directly impact agent performance for particular simulations. Particularly, *agent memory*, *neighborhood connectivity*, and *neighborhood ordering* each impact how well an agent can use information received by a neighbor, how many neighbors are available to collaborate with, and the ability to recognize or distinguish between neighbors respectively. For instance, if there are too few or too many neighbors then neighborhood collaboration may be rendered ineffective. In order to ensure the agent performance in the neighborhood simulation are not restricted by these parameters we first do some rough sensitivity analysis and parameter sweeping.

Note that we are interested in making sure that we select the right simulation configurations so that we are not leaving a test case in some local minimum where its performance is not well represented in comparison with the other simulation cases.

Performance includes equilibrium stability, how well an agent can use additional resources to improve load factor (see Definition 5.1), and load factor values themselves.

In the following we present some initial results on finding the appropriate configurations for our simulation.

6.4 Initial Results of Configurations

Table 6.3 lists the general environment configuration parameters their description and their values as selected from the results of the preliminary analysis. Other parameters exist but are irrelevant.

Table 6.3: General simulation parameters and their values.

Name	Value	Description
GenerationScrooge	False	If true the agent will try to spend all its generation on filling its storage before selling it to the grid; however, by doing so the storage may predictably fill forcing all future excess generation to be sold to the grid. Regardless of the setting ultimately all excess power is sold to the grid, the setting simply controls whether the agent will procrastinate the sale as a result of trying to keep the excess power as long as possible.
ForceDumping	False	If true the agent can choose to waste excess generation instead of selling it to the grid in order to maintain a more steady demand fluctuation.

AgentMemory	1,200	The number of hours into the past that the agent can remember and draw predictions from.
GridBuyBack	∞	A hard limit on the total amount of power the grid will buy back from all agents until the future.
ConnectedGraph	True	Insures a path exists between any two neighbors through the neighborhood graph.
Suspension	False	Indicates where agents are allowed to suspend power. This value is false unless in some simulation it is specified otherwise.
Connectivity	12	The number of neighbors surrounding any agent, this value is a best effort.
OrderNeighbors	True	Indicates whether agents pick neighbors in some order of preference when collaborating or if they are oblivious to the neighbor identities and pick neighbors randomly
SimulateFailures	False	Indicates whether or not to simulate system failures and blackouts.

If false the parameter *OrderNeighbors* causes agents to randomly select a neighbor at every instant the agent wishes to collaborate with others. As such, effectively no agent can observe the identity of any other agent including its neighbors and cannot choose which part of its neighboring environment it can influence. On the other hand if *OrderNeighbors* is true agents may distinguish and

select to influence particular neighbors. The implementation being studied leverages the ability to distinguish neighbor and selectively interact with them by enabling each agent to use its memory to keep track of the amount each of its neighbors satisfied its requests for power; for example, if an agent requests x units of power from a neighbor and the neighbor responded with y units of power then the agent will remember y until memory limits cause the agent to replace y with more relevant up to date information. Using this information, when interested in collaborating, an agent will select to interact with its neighbors in order of their previous reputation. The subtle difference between *OrderNeighbors* being true or false is central to this study since when *OderNeighbors* is true agents are less independent and more entangled with other agents than when *OrderNeighbors* is false. Although it may seem that agents are just as autonomous for either value of *OrderNeighbors* since in both cases agents can opt out of interacting with their neighbors, when *OrderNeighbors* is true agents are actually slightly less autonomous than otherwise since their state is dependent on the information and influence received by neighbors. The difference in autonomy between the two environments becomes even clearer when considering coalition enabled environments where the increased reliability of interactions comes at the cost of externally imposed rules in addition to more agent information and influence entanglement; in such light one can more easily see that the difference in autonomy between the two environments defined by *OrderNeighbors* is but a notch in the greater scale situated between utter absence of information distribution and control between agents to full information distribution and control among agents such

that any agent is equivalent to the entire system of agents. A system of agents where any one agent has as much information and control over the environment as the whole system, is completely centralized. In such a situation the agents are not distinguishable and the whole system appears as one agent or in fact one algorithm. This raises an interesting question: what avenues of information and control over the environment must be carved out of an optimal and centralized system in order to split the system into agents with some degree of autonomy such that their autonomy is minimized while solution approximation is optimized considering the nature of the problem the system targets? Experience suggests that the more the autonomy of the agents, the lower the complexity of the resulting system of agents will be compared to the centralized system.

Preliminary data was collected by running the simulations with the parameter sweeping configurations listed in Table 6.4 and Table 6.5. In order to find the number of simulation hours necessary to capture the *equilibria* of each simulation, all simulations were run until the variance of their 96-hour rolling variance over each of the 96 most recent hours was negligibly close to zero.

Table 6.4: The number of iterations and duration of each simulation.

Name	Value/Values	Description
Runs	20	The number of times each setting was simulated. This allows for anomalies caused by outliers to be averaged out.
Hours	12,000	The number of simulated hours each run simulated. The number of simulated hours determines whether the equilibrium of the simulation is captured.

Table 6.5: Parameters sweeping configurations.

Parameter	Initial Value	Final Value	Step
AgentMemory	120	1,200	60
NeighborhoodConnectivity	4	30	2
OrderNeighbors	False	True	

Figure 6.6 and Figure 6.7 show the results of the preliminary simulations. The 14 groups in the horizontal axis correspond to the neighborhood connectivity values: $4, 6, \ldots, 30$. The horizontal values of each group correspond to the 19 agent memory values: $120, 180, \ldots, 1140$, which is in units of simulation hours.

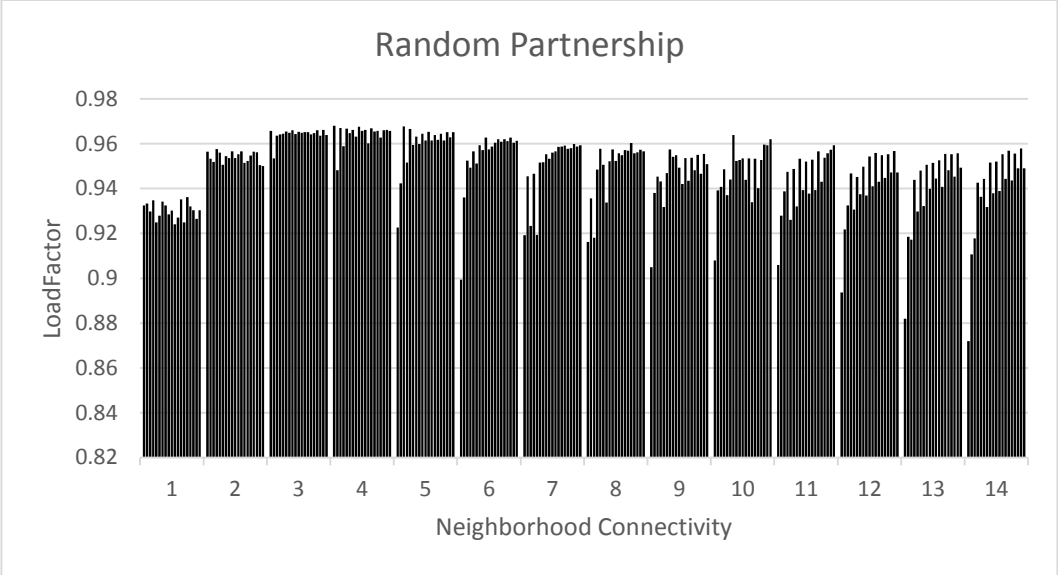


Figure 6.6: The average load factors for each neighborhood connectivity and agent memory setting when OrderNeighbors is set to false. The horizontal axis is made of 14 groups each corresponding in order to one of the neighborhood connectivity values being swept. Each neighborhood connectivity group consists in order of the 18 agent memory values being sampled.

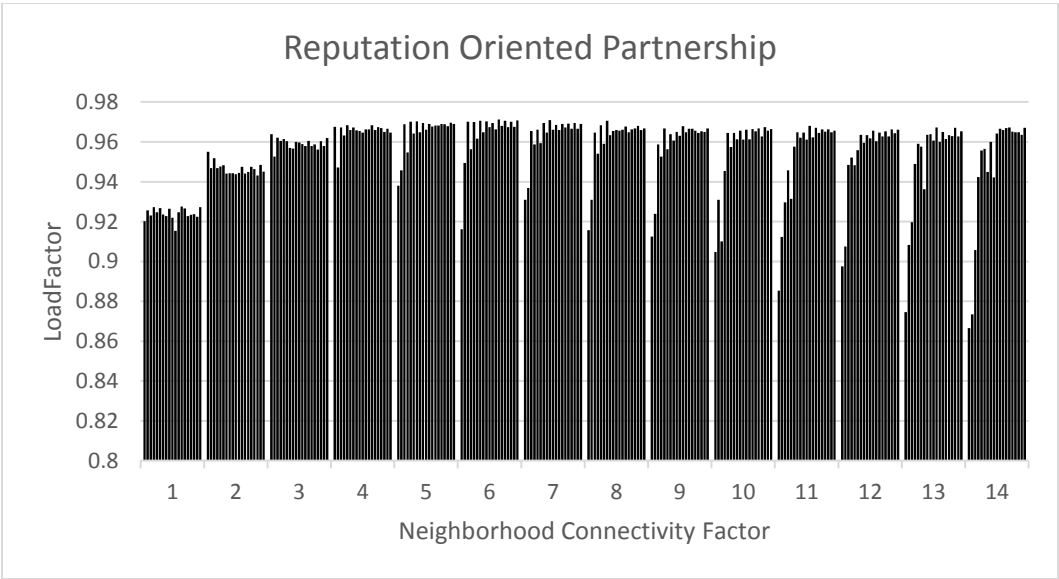


Figure 6.7: The average load factors for each neighborhood connectivity and agent memory setting when OrderNeighbors is set to true. The horizontal axis is made of 14 groups each corresponding in order to one of the neighborhood connectivity values being swept. Each neighborhood connectivity group consists in order of the 18 agent memory values being sampled.

Figure 6.8 and Figure 6.9 show comparable system performance as a function of neighborhood connectivity when, for both agent types, the memory size is held constant at 720 units.

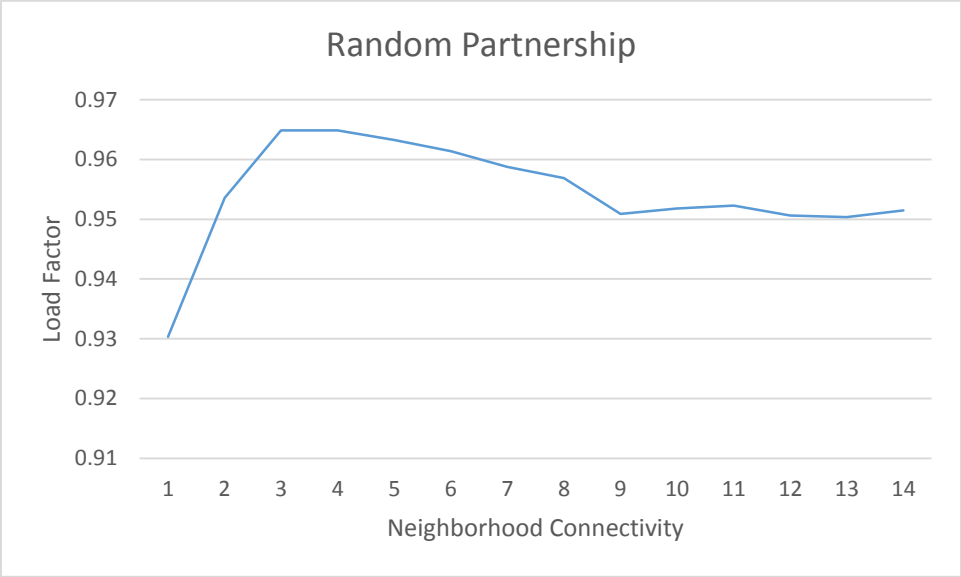


Figure 6.8: The load factor change over neighborhood connectivity corresponding to a memory size of 720 illustrating the typical performance as a function of neighborhood connectivity for agents randomly selecting neighbors when interacting.

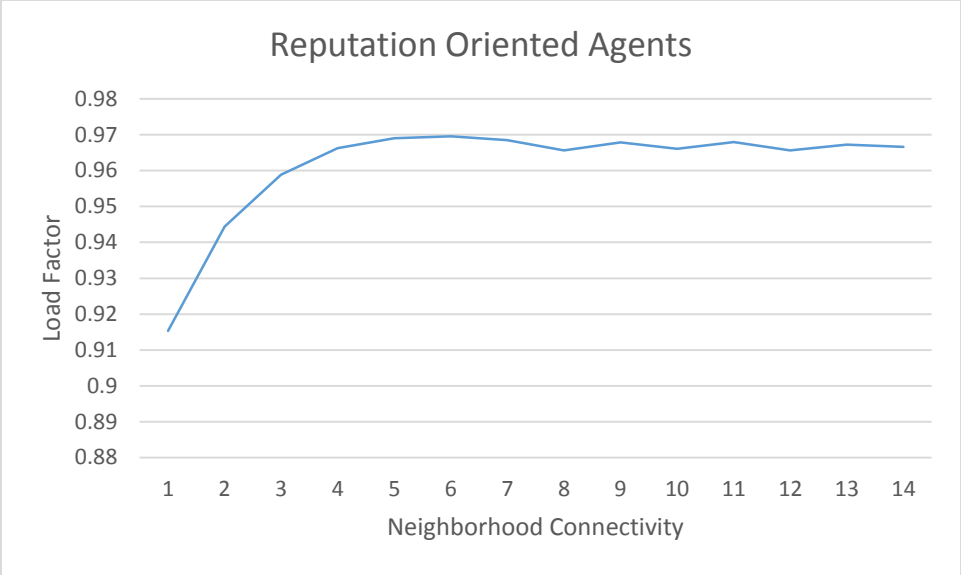


Figure 6.9: The load factor change over neighborhood connectivity corresponding to a memory size of 720 illustrating the typical performance as a function of neighborhood connectivity for agents who exhibit preferences when interacting with neighbors.

Figure 6.6 and Figure 6.8 affords the following key observations:

1. System performance improves dramatically as connectivity increases after which peaking at roughly a neighborhood connectivity of 10 then dropping steadily as connectivity increases further;
2. System performance improvement decreases as agent memory increases;
3. For neighborhood connectivity's less than those associated with local optimal, system performance is not significantly impacted if not reduced as agent memory increases;
4. As neighborhood connectivity increase every other agent memory step shows a separate trends where odd steps consistently perform less than even steps; and

5. The high fluctuation of performance as agent memory increases indicates performance instability and parameter sensitivity.

As neighborhood connectivity increases, so does the number of options each agent has to choose from when needing assistance with its objective. As the number of options increases so do the odds of an agent getting help; however, as the number of neighbors increases so do the chances that any one neighbor prepares resources to provide assistance to others but never is able to do so since other neighbors provided that assistance sooner. Preparing the resources to collaborate with other agents increases demand; however, this demand is short lived since agents who unsuccessfully prepared resources quickly learn their help is not needed and discontinue until they are again randomly picked to collaborate in which case they again increase preparation. The effects of decreased system performance as connectivity increases can be seen particularly well in Figure 6.8. Despite the fact that agents randomly select their neighbors for collaboration, as agent memory increases so does the ability of any neighbor to better predict the amount of resources to prepare. Given the agent settings and their distribution, the amount of impact agent memory comes second to neighborhood connectivity; as such, *agent memory is rendered nearly mute when optimizing agent performance by neighborhood connectivity.*

Since the number of agents requesting to collaborate with a neighbor is less while neighborhood connectivity is low, it is more often the case that the same pair of agents collaborate every time; therefore, *agent memory does not help improve*

predictions by much since the collaboration patterns are fairly consistent. However, *as the number of neighbors increases so does the number of collaborating pairs and so does the combined collaboration pattern complexity making agent memory advantageous*.

The increase in system performance jaggedness as agent memory increases is due to the fact that the step was intentionally selected to be a fraction of the period of the expected agent behavior; in particular, the 60-memory unit step is 2.5 times the length of the expected 24-hour period. The agent memory size corresponding to odd steps of the parameter sweep is fully divisible by the expected agent behavior period; which is not the case for even steps in the sweep. Selecting such a parameter sweep step affords us additional insight into system performance sensitivity and equilibrium stability without complicating the preliminary analysis.

Such insight is afforded as a result of the nature of the problem of scheduling resources. In order to schedule resources an agent must predict its behavior in the future; however, inconsistent prediction quality among different hours in the future limits the reliability of resource scheduling to that of the most poorly predicted hour in the schedule. Sensitivity to consistent unbiased prediction quality among hours being scheduled is not surprising since an error in any efficiently packed schedule will cascade and cannot be recovered from unless the schedule can be made more efficient which contradicts the initial assumption of an efficiently packed schedule. In order to minimize predictive quality bias, the agents are designed to give each hour of their expected behavior period, an equal amount of memory up to what is possible

given the discrete nature of agent memory and simulation time. Consequently, in order for an agent to divide its memory evenly over each period hour, the problem becomes: an agent must predict its behavior period in order to predict its behavior in general. We elected to design the agents such as to constantly predict their behavior period to be the expected agent behavior period of 24 hours (or whatever period is dictated by the agent configurations); this prediction is simple and fairly accurate. As such, it is assumed every agent knows its own expected behavior period, which is not an over expansive assumption, since this information is common knowledge among the agents – in analogy, real world consumers commonly know they have a daily routine.

Although the expected behavior of each agent has a period of 24 hours, this may not hold true for any particular span of time. As such, it is often the case that over any given short period of time and agents behavior does not have a period of 24 hours, in which case, the agent will suffer prediction biases from having a memory size equally divisible over the expected period hours. Consequently, the sensitivity an agent has to locally miss predicting its behavior period can be captured by simply providing agents with a memory length which is unlikely to be divisible by any of its local behavior patterns.

From Figure 6.6, in particular from the jaggedness of system performance as agent memory increases, it can be inferred that an agent who randomly selects neighbors for collaboration, suffers more and more performance and equilibrium instability as a result of its behavior period dynamism.

Figure 6.7 and Figure 6.9, which corresponds to agents having information about the identity of their neighbors and the ability to selectively interact with them, offers the following observations:

1. System performance improves dramatically as connectivity increases after which peaking at roughly a neighborhood connectivity of 12 then dropping negligibly as connectivity increases further.
2. System performance improves drastically as agent memory increase such that it quickly peaks.
3. For neighborhood connectivity less than those associated with the local optimal, system performance is not significantly improved if not slightly reduced as agent memory increases.
4. Performance given changes in agent memory is very stable.

Item 3 and 4 in the previous list is of particular interest since it indicates that agents who are able to selectively interact and identify their neighbors are not only less sensitive to parameter changes, but they also support a more established equilibrium than agents who randomly elect neighbors for collaborations. Furthermore, the average and maximum performance of agents with preferences is higher than those without. These advantages make agents aware of reputation more appropriate for generalizing results since they are less affected by parameters not targeted by the study.

In order to better understand the impact memory size has on the reputation-based system, we took the maximum load factor over all connectivity values for each

memory size and split them into two series based on whether the memory size was biased or unbiased (see Figure 6.10). The results show that in general as memory increases, so does the maximum possible load factor achievable by any sampled network connectivity. Although the maximum load factor corresponding to unbiased memory appears not to be increasing for large memory sizes, after taking noise into account, the given results show the maximum load factor always increases given more memory although ever so slightly.

From the results provided in Figure 6.10 we can also infer that equilibrium stability must also increase as memory increases, which agrees with what we would expect. In order to justify our claim, we must more accurately define the notion of memory size bias introduced earlier in this section and better understand its effects on performance. As mentioned earlier every agent exhibits a periodic behavior of which the agent is itself aware. As prescribed by our simulation configurations, the behavior of each agent has an expected period of 24 hours. Let the *memory bias* of an agent be the *minimum number of hours in the expected agent behavior period having the same share of memory units after memory is divided among each period hour as evenly as possible*. For the sake of clarity the relationship between the integral values of bias, b, and memory size, m, is:

$$b = \left|\left((m + 12)\ mod\ 24\right) - 12\right| \qquad (6.1)$$

As such, the maximum possible bias in our simulation is 12 hours, since any number larger would mean that the memory is not divided across the period hours as evenly as possible. The minimum bias in our simulation is 0 and corresponds to memory

sizes which are divisible by the 24. As discussed earlier, memory bias produces biases in prediction and therefore significantly reduces performance. Although adding any number of memory units less than 24 to an unbiased memory size does provide space for more information for the agent, this information only serves to bias predictions for a particular set of period hours since other period hours would lack such added information. Therefore, keeping in mind that the simulation divides memory as evenly as possible across the expected period hours, for any memory size having the same minimum number of memory units per period hour, the maximum and maximum load factor corresponds to the memory size with the minimum and maximum bias respectively. Therefore, the load factor of the unbiased memory sizes from Figure 6.10 must be the maximum possible for any network connectivity value that was sampled, since the bias of those memory values is 0. Likewise, the load factor of the biased memory size (from Figure 6.10) must be the lowest that can be maximally achieved by any sampled network connectivity value.

As mentioned earlier, over any particular short span of time, the behavior periods of an agent likely does not exhibit the expected behavior period of that agent. When normally a memory size divisible by 24 would be unbiased, since none of the behavior periods in such a time span is 24, the memory size would be biased for the agent behavior periods in that time span. As such, the maximum performance achievable under the constant parameters and all the network connectivity values should be between the unbiased and biased load factors in Figure 7.10. Now, we have observed that the distance between the unbiased and biased performance decreases as

memory size increases, it follows that equilibrium stability—which is inversely correlated to such distance—also increases as memory size increases, given the right network connectivity conditions exist.

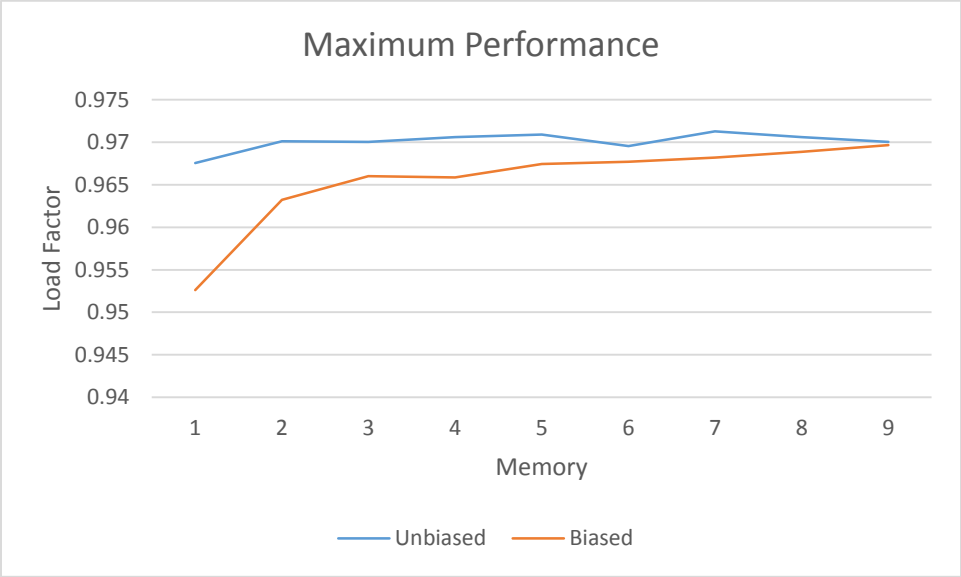


Figure 6.10: The maximum recorded load factor for each of the tested neighborhood connectivity as memory is increased. Each memory bucket from 1 through 9 signifies memory values of 120, 240, …, 1080 for the unbiased sequence and 180, 300, …, 1140 for the biased memory sequence. If we were to measure time in units of 5 periods (similar to say how a week can be viewed as seven 1-day periods) then each memory bucket would have enough space to store information for 1, 2, …, 9 units of time.

This phenomenon is observable regardless of whether agents interact with one another or are completely independent. So by picking the memory to be large enough for all test cases the amount of system underperformance and equilibrium unreliability for any particular test can be made negligible. As such we will conduct tests with the memory size to be set to the largest unbiased tested value which is enough for each agent to store information they observe for 1,200 units of time.

Neighborhood connectivity can only influence the test case involving neighborhood trading. So, in order to avoid any underperformance in that test case

with respect to others, we will perform tests using the empirically optimal neighborhood connectivity of 12, which was concluded from the results in Figure 6.7.

According to the results shown in Figure 6.6 and Figure 6.7 and by the reasoning brought in this section, the peak performance of agents who are able to selectively interact and identify their neighbors is not only higher but is much more stable and consistent over parameter sweeping; therefore, we will consider *OrderNeighbors* to have a value of true in the simulations.

We have justified the values selected for the problematic general parameters; the remainder of the parameters are not particular to any simulation case and have a straightforward impact on the simulations.

6.5 Solo Simulation

In this simulation agents do not interact or even know of one another's existence. The agents are only able to observe and interact with their local environment which only includes adjusting their demand from the grid using their own resources which only includes generation and storage. The results of this simulation will give us a baseline on agent performance.

6.5.1 Demand Management Implementation

Pseudo Code 6.1 details the implementation of the proposed methodology and approach described in Chapter 3 and Chapter 4. The pseudo code explicitly describes how an agent manages its resources in order to advance and postpone its load depending on its available resources and predicted over and under average deficit.

Satisfy as much load possible with generation

if $\overline{Deficit} > 0$

  if $Deficit = \overline{Deficit}$ **then**

    Satisfy load by buying $\overline{Deficit}$ amount of electricity from grid

  else if $Deficit > \overline{Deficit}$ **then**

    Satisfy load by buying $\overline{Deficit}$ amount of electricity from grid

    $StorageShare \leftarrow \frac{StoredPower}{LocalOverAvgDeficit()} Load$

    Satisfy load by retrieving $StorageShare$ amount of electricity from storage

  else

    if $Deficit > 0$ **then**

      Satisfy suspended load using remaining generation

    else

      Satisfy all load by buying electricity from the grid

    $SuspendShare \leftarrow \frac{SuspendedLoad}{LocalUnderAvgDeficit()} \left(\overline{Deficit} - Deficit\right)$

    Satisfy suspended load by buying $SuspendedShare$ electricity from grid and let the satisfied amount be referred to by $Adjustment$

    $Remainder \leftarrow LocalUnderAvgDeficit() - Adjustment$

    if $Remainder > 0$ **then**

      $BatteryShare \leftarrow \frac{AvailableStorageCapacity}{Remainder} \left(\overline{Deficit} - Deficit - Adjustment\right)$

      Energize the battery $BatteryShare$ amount of electricity from the grid

else

  if $Deficit = \overline{Deficit}$ **then**

    Sell $-\overline{Deficit}$ amount of electricity to grid from generation

  else if $Deficit < \overline{Deficit}$ **then**

    Sell $-\overline{Deficit}$ amount of electricity to grid from generation

    $SuspendShare \leftarrow \frac{SuspendedLoad}{LocalUnderAvgDeficit()} \left(\overline{Deficit} - Deficit\right)$

    Satisfy suspended load by spending $SuspendedShare$ electricity from generation and let the satisfied amount be referred to by $Adjustment$

    $Remainder \leftarrow LocalUnderAvgDeficit() - Adjustment$

    if $Remainder > 0$ **then**

      $BatteryShare \leftarrow \frac{AvailableStorageCapacity}{Remainder} \left(\overline{Deficit} - Deficit - Adjustment\right)$

      Energize the battery $BatteryShare$ amount of electricity from generation

```
else Deficit < 0 then
    Sell all generation to the grid
    StorageShare ← StoredPower / LocalOverAvgDeficit() (Deficit − Deficit̄)
    Sell StorageShare amount of stored power to grid from storage
else
    StorageShare ← StoredPower / LocalOverAvgDeficit() (Deficit − Deficit̄)
    Satisfy load by retrieving StorageShare amount of electricity from
storage letting Remainder be any StorageShare left over from satisfying load
    Sell Remainder amount of stored power to grid from storage
```

Pseudo Code 6.1: Simulation implementation of an agent's power management following the approach and methodology described in Chapter 3 and Chapter 4.

The implementation in Pseudo Code 6.1 divides the state of and agent into the following cases:

1. $Deficit = \overline{Deficit} > 0$
2. $Deficit > \overline{Deficit} > 0$
3. $Deficit < \overline{Deficit}\ and\ \overline{Deficit} > 0$
4. $Deficit = \overline{Deficit} \leq 0$
5. $Deficit < \overline{Deficit} \leq 0$
6. $Deficit > \overline{Deficit}\ and\ \overline{Deficit} \leq 0$

The reason for this division of state is not because the proposed approach requires each case to be handled uniquely, but instead because the source and destinations the agent receive and sends power to are different for each case. The approach does not require for such casing; it would have been possible to abstract the power sources and destinations such that all cases would be handled under one general case; however, such an implementation would not be enlightening. In each case, at any moment in time, the agent simply tries to reduce the difference between the average deficit and its current deficit. This is done by postponing or advancing

demand at any given moment by the ratio of resources to total predicted adjustments. The equations in the algorithm describing how much power should be transferred for each adjustment and the expressions describing each of the 6 cases are effectively utility functions which could be abstracted and formalized; however, that level of indirection is not necessary for our interests. Abstracting and formalizing the utility functions would allow for demand adjustments that are more complex than simply reducing demand fluctuation by a constant factor as suggested by the proposed approach. Such an abstraction would be useful for situations where the cost associated with demand fluctuation is itself complex and possibly not monotonically increasing.

After an agent has managed its resources and adjusted its demand or if it did not have any resources to begin with, the agent follows the prescribed steps in Pseudo Code 6.2 to satisfy any remaining load and spend any remaining generation. In other words any remaining demand after adjustments must be applied to the grid or reconciled if the grid is not available. These steps are prescribed since the agent does not have resources or options at this point. Table 6.6 describes the variables and function used in the pseudo code.

```
Satisfy any remaining load by buying power from the grid if possible
Satisfy any remaining load by using available stored power if possible
Use any remaining generation to satisfy any suspended load
if GenerationScrooge is true then
  store any remaining generation if possible
Sell any remaining generation to the grid if possible
// at this point the agent has done anything is could to satisfy it load
and benefit from its generation
Forcefully postpone any remaining load since no resources are available
Dump any remaining generation since it cannot be benefited from
```

Pseudo Code 6.2: An agent not having any resources or an agent with resources who has completed its demand adjustment using Pseudo Code 6.1 satisfies its remaining load and spends its remaining generation according to these prescribed steps. These steps are prescribed because the agent has not options at this point.

Table 6.6: A brief description of the variables introduced in Pseudo Code 6.1.

Variable	**Definition**
$\boldsymbol{Deficit}$	The amount of power the agent will require in addition to its generation to satisfy its needs.
$\overline{\boldsymbol{Deficit}}$	Average deficit as recalled using the agent's memory.
$\boldsymbol{Load}$	The amount of load for the given hour that is remaining and must be satisfied.
$\boldsymbol{LocalOverAvgDeficit}()$ $\boldsymbol{LocalUnderAvgDeficit}()$	The total amount of deficit above (under) the average deficit the agent has or predicts it will have starting from the current moment in time

	until the first moment in the future where the deficit is below (above) average.
$\boldsymbol{StoredPower}$	The maximum amount of power the agent can extract from its storage at that given moment.
$\boldsymbol{AvailableStorageCapacity}$	The maximum amount of power the agent can store for future use at that given moment.
$\boldsymbol{SuspendedLoad}$	The total amount of load the agent has postponed for the future.

6.6 Neighborhood Setup

The neighborhood described until this point is a simple implementation of an ad-hoc coalition. If an agent A is unable to reduce its demand fluctuation any further on its own, then A looks to its neighbors for assistance by requesting its electricity shortage from each neighbor in some order of priority. If neighbor N agrees to assist A then it does so because A shares the benefits of the additional reduction of demand fluctuation. In other words, the collaboration of A and B produces value which they benefit from as a collective. This collective can be arbitrarily large and can form arbitrarily complex networks with cycles because already collaborating agents may also collaborate with other neighboring agents. In order to facilitate sharing of benefits we elected to simulate trading of monetary value and electricity among neighbors particularly such that both collaborating parties receive half the benefits of

collaborating. As described in previous sections the benefits of collaborating (reducing demand fluctuation) consist of local and aggregate benefits. For the sake of completeness, aggregate benefits appear as a reduction in future electricity rates the benefits of which are shared among collaborating agents by default, and local benefits b_l can be expressed as:

$$b_l = d(p_s - p_b) \tag{6.2}$$

where d is the remaining demand adjustment an agent requests assistance with from its neighbor, and p_s and p_b are the current selling and buying electricity rates that an agent must cope with if not trading electricity with a neighbor (in terms of the simulation implementation p_s and p_b are the grid agent's selling and buying price). As such a collaborating pair of neighboring agents each receives $\frac{b_l}{2}$ benefits. In other words, the remaining demand adjustment d than an agent fails to personally handle would have a surplus cost of b_l, which the agent can save if it were to trade with a neighbor.Since at any given moment, by simplicity of the neighborhood design, no agent knows how much excess resources any neighbor N has available or will have available in the future, and furthermore, since an agent only knows the power requirements of a neighbor confidently the moment the neighbor makes the request, agents reserve the option not to partake in collaborations at any moment if the collaboration risks self-interest. In other words, an agent A knows its condition better than it can guess the condition of a neighbor N; therefore, A does not collaborate with N unless it is sure the chance and amount of what it stands to gain is more than the risk and amount it stands to lose.

In particular agents do not know of one another's demand profile or buffering resource availability, as such they are less sure of the benefits they can gain from cooperating with one another than they can be sure of watching out for their immediate self-interest instead. As such, an agent may not prepare in advance to assist a neighbor in favor of being better prepared to handle its own potential yet more predictable problems.

6.6.1 Neighborhood Implementation

Agents enabled to perform neighborhood interactions manage their demand as in Pseudo Code 6.1 with the added steps in Pseudo Code 6.3.The added steps govern how an agent determines its excess resources and how it applies the proposed approach to those resources and the neighborhood request history to reduce demand fluctuation. If an agent's available storage is more than the total predicted local under average deficit, then the agent will have excess free resources since at most the agent would store all the total predicted local under average deficit to supply the predicted future equivalent over average deficit. In this case the agent will store power for a neighbor as specified by the proposed approach only if the predicted neighbor request for that moment is less than average; as such, the agent will effectively be advancing the neighbors future request to the current moment.

An agent sells power to its neighbors upon the neighbors request using the algorithm described in Pseudo Code 6.4. It is important to note that both Pseudo Code 6.3 and Pseudo Code 6.4, which implement neighborhood interactions, comply with the methodology and approach described in Chapter 3 and Chapter 4 respectively.

```
if AvailableStorageCapacity > LocalUnderAvgDeficit()
        and NeighborhoodRequest > PredictedNeighborhoodRequest() then
   underAvgRequest ← NeighborhoodRequest − PredictedNeighborhoodRequest()
   requestRation ← (AvailableStorageCapacity − LocalUnderAvgDeficit()) / LocalUnderAverageRequest() underAvgRequest
   Store as much requestRation from generation and then grid as possible
power ← 0
in order of reputation for each neighbor in neighborhood
   if power < request then
      ask neighbor to sellPower(request − power) and let result be the power
provided by neighbor
      power ← power + result
   else return power
return power
```

Pseudo Code 6.3: The algorithm used by an agent to prepare resources for trading with its neighbors. An agent which can perform neighborhood interactions performs this algorithm as part of its demand management after performing Pseudo Code 6.1.

```
def sellPower(request)
if StoredPower ≤ LocalOverAvgDeficit()
        or NeighborhoodRequest ≥ PredictedNeighborhoodRequest()
        or NeighborhoodRequest ≥ NeighborhoodRequestRation() then return 0
   power ← (StoredPower - LocalOverAvgDeficit()) / LocalOverAvgRequest() · PredictedNeighborhoodRequest()
            −powerSoldToNeighbors
if request > power then return as much of power from storage possible
else return as much of request from storage possible
```

Pseudo Code 6.4: An agent capable of neighborhood interactions will sell power to neighbors based on this algorithm.

An agent who is providing power on request to a neighbor is in fact treating the neighbors request just as it would any other load with the exception of a

neighbor's request having lower priority. The demand associated with satisfying the neighborhood's request is managed in the same way as the agent's personal demand would, however with added restriction of reduced priority and complete absence of information regarding the requesting agent's demand profile aside from that which can be inferred from the requests itself. In this way an agent does not know how to efficiently appropriate resources for a neighbors future request but the agent can predict based on previous requests how much resources and by when they must be appropriated. Table 7.7 documents the variables and their definitions, used in Pseudo Code 7.3 and 7.4.

Table 6.7: Variables and their definitions as introduced in Pseudo Code 6.3 and Pseudo Code 6.4.

Variable	**Definition**
$\boldsymbol{NeighborhoodRequest}$	The amount of power that has been requested by the neighborhood for the current hour
$\overline{\boldsymbol{NeighborhoodRequest}}$	The average amount of power that has been request from by the neighborhood as recalled from the agents memory
$\boldsymbol{powerSoldToNeighbors}$	The amount of power sold to neighbor during the current hour

$\boldsymbol{LocalOverAvgRequest}()$	The total amount of power requested or predicted to be requested by the neighborhood which is in excess of the average neighbor request starting from the current hour until the first hour where neighborhood requests are less than average
$\boldsymbol{PredictedNeighborhoodRequest}()$	The amount of neighborhood request which is predicted for the current hour as extrapolated from the agent's memory

6.7 Unified Agent Implementation

Holding the solution approach and all simulation variables constant an agent can only vary on its performance through varying its degrees of collaboration. Holding all simulation variables constant, each simulation case being studied is in fact a discrete sampling of the agent collaboration space. The unity test case is purposed to sample the maximum end of the agent collaboration spectrum. The agents in the unity test case should be able to optimally collaborate. Agents can only share information and act on behalf of one another either directly or indirectly. Furthermore, an agent can only act on its environment by manipulating its resources and consuming power, all other actions, such as trading with a neighbor, if any, only facilitate resource manipulation and power consumption and therefore can be considered a component

of resource manipulation and power consumption. Therefore at maximal collaborative extreme agents can at best share all their state information and grant full control over resources and power consumption to one another. Since in addition to holding all parameters constant we are holding the solution approach itself constant there is only one very specific way to leverage all shared state information and shared will to act on the environment. Therefore, the system must leverage all information to compute the mean demand, over and under demand, and the ratio of demand to displace as a function of available resources; furthermore, the system must leverage all abilities to act on the environment to use buffering resources and consume power in accordance to what is calculated with the information. Therefore, such a system of agents would appear to function exactly as any one agent would except for the fact that the system of agents would have full information and control over each other. It is not practical to construct or simulate a system where all agents can fully communicate state information and fully command each other's will. However, since under the imposed restrictions the maximally collaborative system of agents is equivalent to one agent having a load and generation profile and resources equal to that of the entire system of agents, it is possible to synthesize such a system by simulating one agent.

Synthesizing the system of maximally collaborative agents, although impractical in the real world, affords us an understanding of what is theoretically achievable if given the proposed methodology, collaboration were maximized. By definition an agent of such a theoretical maximally collaborative MAS would have no autonomy of its own since its information, actions, and ability to achieve a goal

would be entirely dependent on all other agents in the system. This is a key idea we will return to when trying to understand the role autonomy plays in the proposed approach.

6.8 Summary

Over this chapter we have justified the parameters we are holding constant and described in detail each simulation case. We are interested not only in studying the performance of the proposed methodology and approach described in Chapter 3 and Chapter 4 by comparing the performance of various solution implementations each focused on leveraging a key MAS collaboration concept, but we are also interested in understanding the role of collaboration itself in the grander scheme of things. In order to do so we intend to hold the solution approach and all simulation parameters constant. Under such conditions an agent can only vary its performance through varying its degrees of collaboration. The simulation cases are designed such that they each are a discrete sampling of the agent collaboration space. By empirically maximizing the performance of each simulation case through fine tuning the parameters that uniquely influence each of collaborative capacities of the simulation cases, we can compare the performance of each case without any bias up to empirical precision.

CHAPTER 7 SIMULATION RESULTS

We ran the simulations outlined in Table 6.1 with the configurations described in Chapter 6. Table 7.1 recaps our intentions for the simulations and our direction in this chapter.

Table 7.1: Recapping the outline of where we are heading with the simulations in this chapter.

Simulation	Distinguishing Factor	Simulation Objective
Solo Simulation Case	Agents are fully autonomous up to that prescribed by the proposed approach	To understand the dynamics of the proposed approach and provide a lower bounds on expected performance when varying simulations on resources and autonomy.
Neighborhood Simulation Case	Agents are mostly autonomous and have information dependencies only	To understand the dynamics of applying the proposed approach MAS concepts, in particular ad-hoc coalition with partial information sharing.
Unity Simulation Case	Agents are minimally autonomous up to what is prescribed by the proposed approach	To understand an upper bound on performance when varying only autonomy and resources. Provides a means of comparing optimal performance of the proposed approach to a more feasible collaboration based approach.

Storage resources were varied among simulation instances by a multiple of the storage capacity defined by the simulation configurations described in Code Snippet 6.1, Code Snippet 6.2, and Code Snippet 6.3. The storage capacity multipliers were swept over 0 to 9.75 inclusively with a step of 0.25. As such, each simulation case was associated with 40 instances, each instance corresponding to one storage capacity multiplier. Each instance was run 20 rounds, in order to compensate for noise introduced by the stochastic nature of the simulation, and for 12,000 simulation hours, in order to capture the equilibrium. The load factor and price at equilibrium for each simulation instance was estimated by averaging the load factor and price of the last 1,200 simulations hours over the corresponding 20 round of that instance. Figure 7.1 and Figure 7.2 show the resulting load factors and base prices respectively.

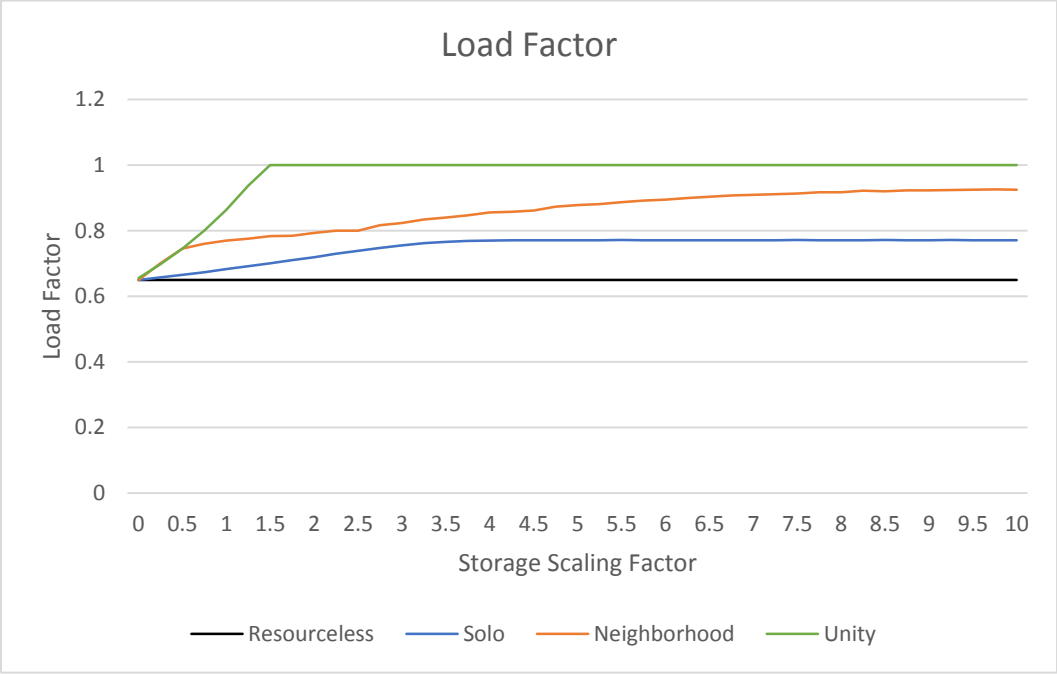


Figure 7.1: The load factor corresponding to each of the simulation cases as storage capacity is scaled by the factors specified on the horizontal axis.

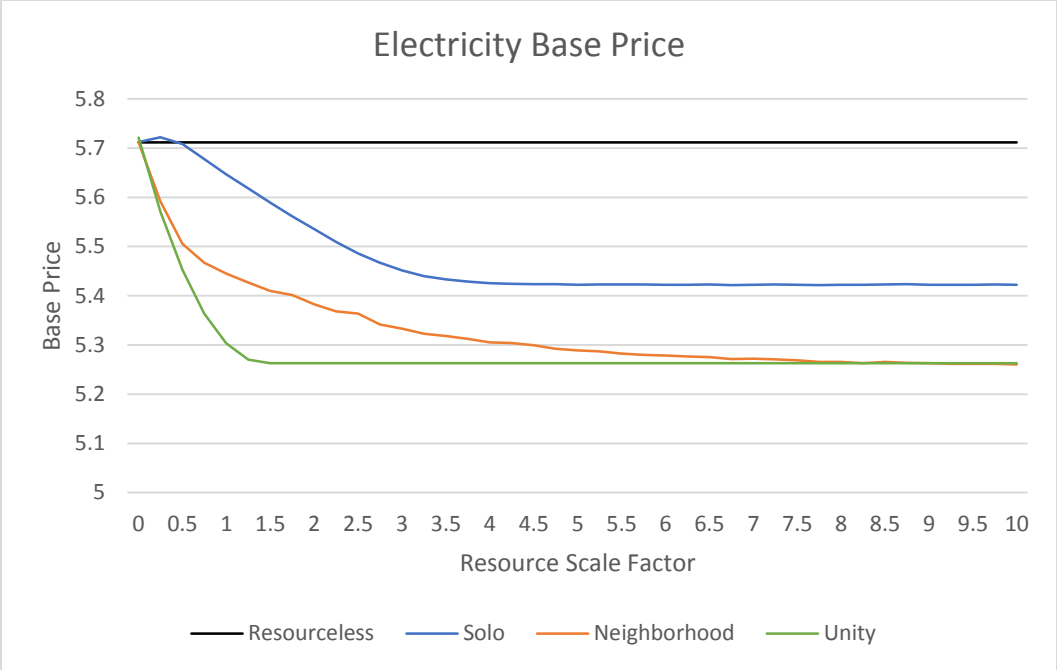


Figure 7.2: The base price corresponding to each of the simulation cases as storage capacity is scaled by the factors specified on the horizontal axis.

7.1 Solo Simulation Case

The solo simulation provides a baseline for the performance achievable given the prescribed approach. In other words, any performance lower than this base line requires the agents not be using the prescribed approach. We summarize the reasoning brought until now for this claim.

For system demand fluctuation to reduce, aggregate demand must be adjusted. We have proposed a solution describing how to compute and apply demand adjustments to reduce demand fluctuation in Chapter 3 and Chapter 4. The solution depends on two items:

1. Agent control over the environment: the ability to act on buffering resources
2. Agent information about the environment: the ability to predict future deficit

In order to substantiate our claim, we will show how the variables for these two items are all either fixed or controlled in the simulation.

Agent control over the environment:

The simulations fix buffering resource sizes and the smart grid problem prescribes that buffering resources be controlled in a distributed manner. Therefore, up to this point, all variables related to the extent at which an agent can control its environment are fixed except for the amount of overlapping—and sometimes interfering—control that agents have on the environment. The simulation cases each control the amount of overlapping control agents have on the environment in order to facilitate studying the effects of this control on system performance.

Agent information about the environment:

The simulations fix the amount of agent memory. Furthermore, as opted by implementation of the proposed approach covered in Chapters Chapter 3, Chapter 4, and Chapter 5, agents predict future deficit on the expected deficit behavior computed from their memory. The amount of information the agents share with each other is controlled by each simulation case. Table 7.2 summarizes the overlaps in information and environment control for each simulation case.

Table 7.2: The amount of information and environment control overlap for agents in each of the simulation cases.

Simulation case	Required resource control overlap	Required information overlap
Solo	None, each agent controls their own resources	Only know about their own deficit history and expected deficit period
Neighborhood	None, each agent controls their own resources	In addition to the information agents have in the solo simulation case, neighboring agents know each other's identity and can share how much deficit over average they are unable to level out and how much power they are willing to trade.
Unity	All agents share control over all their resources	All agents share information about each other's expected deficit period, demand deficit history, and resource states. All agents recognize each other's identity.

Therefore the solo simulation represents the case where agent are only aware of their own state and control their own resources. Since all other parameters are fixed, the solo simulation represent the performance which can be achieved using the proposed approach, with only local control and information. Any additional control and information should only provide a means of improving performance; in the worst case

agent of such a system would have everything necessary to perform at least as well as the solo simulation case.

Figure 7.1 shows that as the resources factor—i.e., storage scaling factor—increases the load factor of the solo simulation case increases at a near linear rate after which the rate of increase in load factor slowly decreases until the load factor reaches a cap. The results suggest that the load factor and resources have a proportional relation until a point where the improvement in performance becomes sub-linear[3].

We collected information on the load factor of agents with resources and agents without access to resources in order to understand the sub-linear increase in load factor and load factor cap; see Figure 7.3.

The results show that the load factor of the proposed approach is not inhibited by anything; the proposed approach can leverage any amount of resources until agents reach the proximity of the maximum load factor. As shown in Figure 7.3, the reason for the load factor cap of the solo simulation seen in Figure 7.1 is because agents without resources do not benefit from additional resources provided by the other agents. Since the demand fluctuation of agents without resources is never improved, they will always introduce some fluctuation in aggregate demand, hence the load factor cap.

[3] A sub-linear relationship of variable y with respect to variable x, indicates that as x increases y generally increase less and less. Similarly a super-linear between the same variables would indicates that as x increases, y generally increases more and more.

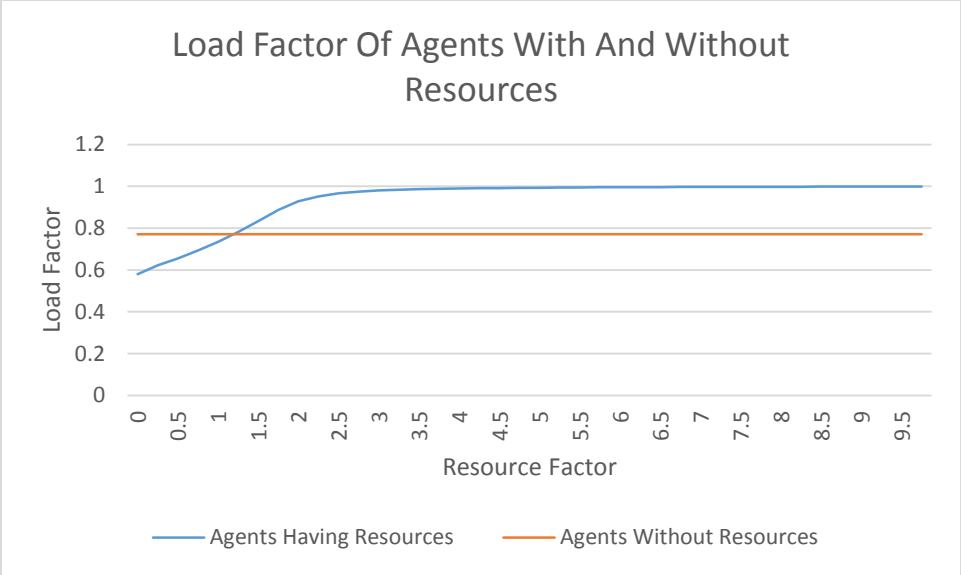


Figure 7.3: The load factor of agents with resources increases until peaking at near maximum which the load factor of the agents without resources remains constant. The reason for the load factor of the agents without resources starting higher than the agents with resources is because the agents without resources include the industry agents whereas the later does not; as such only the load factor trending is comparable.

Figure 7.3 also shows a sub-linear increase in load factor as the resources factor increases. This is caused by agents having resource capacity and deficit profiles that are sampled from a normal distribution. Because of the resource capacity and deficit distribution, as resources increase, some agents reach their load factor cap sooner than others. Furthermore, due to the normal distribution of the deficit of any agent, at some point the increase in capacity can only be of assistance if the deficit is over the expected amount for a given hour. Since the chances of a deficit being higher than the expected amount for a given hour reduces in accordance with a bell curve for higher and higher demands the chances that any agent can use the additional resource capacity decrease for significantly large resources. Therefore, compensating for the

effects of resource and demand distribution, we can conclude a linear relation between load factor and resource capacity up to the maximum load factor.

This is an important finding since if the approach were to produce a sub-linear load factor relation with respect to resource capacity, then the approach would have a decreasing ability to leverage new resources which could arguably suggest performance shortcomings. The same could be said for a super-linear[3] relationship between the load factor and resource capacity since this would arise the question of why the approach is not able to leverage lesser resources the same proficiency as it can higher resources?

Limited to the proposed approach, the agents in the solo simulation case are completely independent of each other. This means that each agent can achieve its goals without any reliance on information or actions from other agents. Although system performance as a whole can be influenced by the demand fluctuation cancelations among multiple agents, the performance of any particular agent is not dependent on another agent. Furthermore, although each agent can glean some information about other agents from observing electricity rates or other environmental properties, under the proposed approach and in the solo simulation case, the agents do not leverage this information when taking action. The agents only minimize the fluctuation of their personal demand as prescribed by the proposed approach. Simply put the solo simulation the minimum base line performance for a completely autonomous implementation of the proposed approach. This gives us a basis in which to compare the performance of other simulation cases in terms of autonomy.

7.2 The Unity Simulation Case

In the unity simulation case, as defined and implemented in Subsection 6.7, each agent has full information about the deficit of all other agents; therefore, each agent can compute the expected aggregate demand fluctuation for the system as a whole. Furthermore, although it may be physically infeasible, theoretically, the agents can coordinate resources such that, despite being managed in a distributed manner, collectively the resources can be treated as a large aggregated logical storage system. One way to do so would be to change and discharge all resources according to their capacity and to route all excess power at one agent through a peer-to-peer network or the distribution network to a sink where the power is stored or used[4]. The purpose of our unity simulation is to provide an upper performance bound for the given approach, since the unity simulation has all the same parameters as the other simulation cases except there are no bounds on information and ability to control the environment as dictated by the proposed approach.

Given the proposed approach, the advantage the unity simulation has over all other simulation cases is that demand adjustments are made to aggregate system demand which takes into account any demand profile fluctuation cancelations; as such, the agents of the unity simulation do no spend resources on reducing local demand fluctuation which would have otherwise been canceled out without such expenses. Furthermore, since all agents have full control over one another resources,

[4] Note that such an infrastructure would be very impractical given today's technology leave agent privacy interests, security problems, regulations and laws which may each alone preempt such a system.

technically there are no agents without resources. The only limitation the system can have is the proposed approach itself. Figure 7.1 shows that as resource factor increases the load factor of the unity simulation increases linearly until suddenly hitting the maximum load factor.

The unity simulation, represents the case where *agent have minimal autonomy*. In fact, the results of the simulation are synthesized by simulating the system as one giant agent in having all the resources, deficit memory, etc. of the entire system as described in Sections 6.7. It is worth noting our distinction between simulate and synthesize. A simulation suggests that all relevant nuances and details are carried out in a virtual manner in order to produce results that reflect those nuances whereas synthesis will directly produces only the results. Synthesis is only possible if there are no nuances and special dynamics to be captured making a generalization and abstraction of the system possible. In the case of the unity simulation, it is not practical to simulate the information communication allowing for total awareness of every agent regarding anything any other agent knows and for all agents to coordinate each other control over one another's resources; however, since all agents know and control everything any other agent knows and controls we use this uniformity and lack of nuance to generalize one agent to synthesize the precise results of the theoretical simulation. In fact it is not difficult to implement a full simulation for the unity case; however, producing the results in a timely manner is not practical. We outline one such implementation in Sections 6.7. Although it may be

possible to simulate the unity case, it is nearly impossible to physically produce a smart grid that can support the power transmission among other physical limitations.

It is difficult to prove that the agents of the unity simulation indeed are minimally autonomous in the absolute sense since the proposed approach may inhibit minimal autonomy by indirectly forcing some level of independence among agents; however, up to what is required by the proposed approach the agents of the unity simulation are minimally autonomous. This is simply because we are applying the proposed approach at the aggregate scale which is the composite of all the system agents. If an agent were independent it could only be that it does not contribute to aggregate demand fluctuation and does not have any resources to share this is the only case where if the agent is removed the for the system or the system is removed from the agent it would not have any impact on performance. We are not interested in such agents since they do not exist in the problem statement where demand fluctuation is required.

7.3 Neighborhood Simulation Case

Figure 7.1 shows that, unlike the load factor of the solo and unity simulation cases, the load factor of the neighborhood simulation, for the most part, improves at a decreasing rate as the resource factor increases. The only exception is were at roughly 2.5 times the configured storage capacity the load factor improvement rate of the neighborhood simulations increases. Figure 7.1 also shows that between the resources factor of 0 and about 0.5 the neighborhood load factor is almost as good as that of the unity simulation case. The reason the non-uniform load factor improvement and loss

of effective resource utilization beyond the resource factor of 0.5 can be understood by observing the load factor behavior of each agent type when it collaborates with other agents.

In order to further understand the collaboration dynamics among agent types, we ran 6 simulations where each simulation corresponded to a pair of agent types one having and the other lacking resources. The configurations in these simulations are identical to those of the neighborhood simulation case with the exception that all agent types other than the agent type pair being examined are excluded from the system. Table 7.3 describes the agent type pairs included in each of the simulation cases.

Table 7.3: Table of agent types associated with each simulation.

#	Agents with resources	Agents without resources
1	100 Home agents	100 Home agents
2		50 Commercial agents
3		25 Industrial agents
4	50 Commercial agents	100 Home agents
5		50 Commercial agents
6		25 Industrial agents

Figure 7.1 and Figure 7.5 show change in load factor relative to resource factor as corresponding to the simulations listed in Table 7.3: Table of agent types associated with each simulation. In particular, Figure 7.1 and Figure 7.5 correspond to the simulations where the resource possessing home and commercial agent type

respectively, support agent types without any resources through neighborhood collaboration.

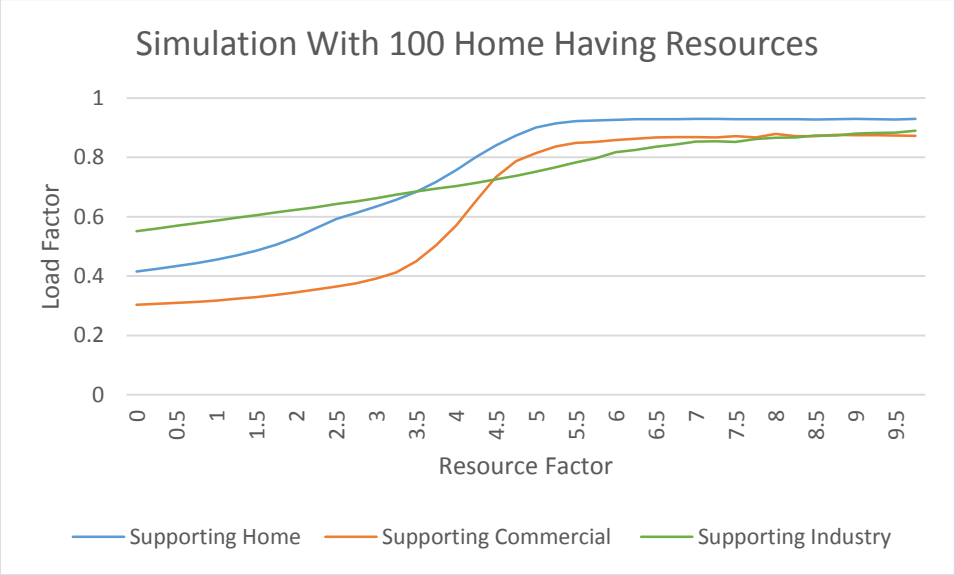


Figure 7.4: The load factor corresponding to systems where 100 homes with resources support homes, commercial agents, and industrial agent each not having any resources.

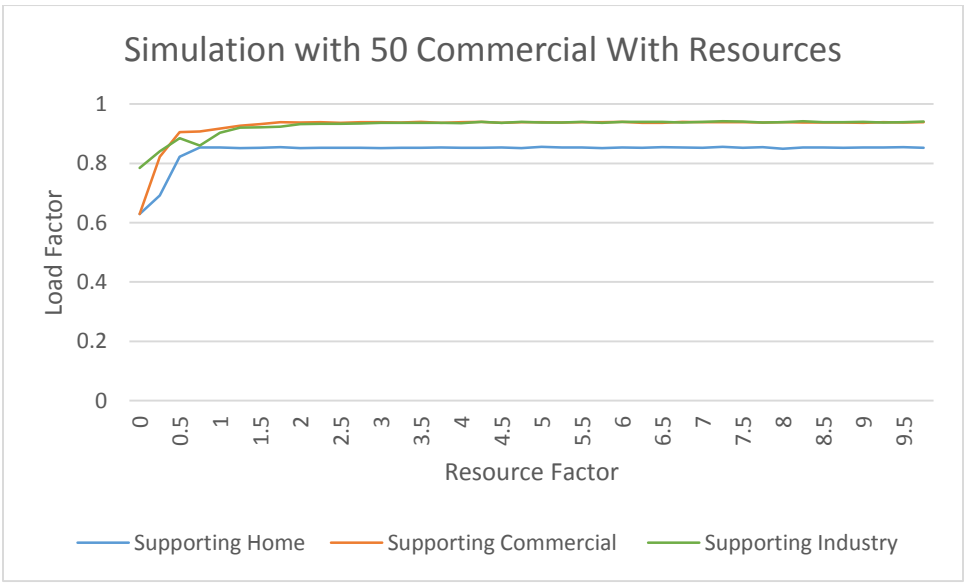


Figure 7.5: The load factor corresponding to systems where 50 commercial agents with resources support homes, commercial agents, and industrial agent each not having any resources.

Figure 7.4 and Figure 7.5 afford the following observations each providing unique insight into the dynamics of the solution approach when applying neighborhood collation:

1. The load factors in Figure 7.4 and Figure 7.5 all increase until reaching maximum after which neither agents nor neighborhood collaboration are able to leverage any additional resources toward improving system load factor.
2. Figure 7.5 shows that the load factor for all simulations where the commercial agent type having resources supports other agent types, drastically increases as the resource factor increase from 0 to 0.5.
3. The load factors of each collaborating agent type pair in Figure 7.4 and Figure 7.5 exhibit an inflection point at different resource factors, respectively. Although the inflection points in the simulation cases where commercial agent types having resources support commercial and industry agent types without resources are not prominent, nonetheless the inflection point exists for even these cases and are demonstrated later in Subsection 7.3.3.
4. Figure 7.5 shows a pronounced dip in load factor for the simulation where commercial agent types having resources support the industry agent type when the resource factor reaches roughly 0.75.

Over the following subsections we will explain the dynamics behind each of these observations and how they manifest in the overall performance of the neighborhood simulation case performance results shown in Figure 7.1.

7.3.1 Understanding the Load Factor Cap

All simulations, except for the unity simulation case, show that agents reach a performance ceiling lower than the maximum possible. That is, at this performance ceiling no amount of additional resources can improve system performance. For performance to improve until reaching the absolute maximum as resources increase, agents who are unable to independently modify their demand must somehow do so entirely with the resources of other agents. The results for the simulations listed in Table 7.3 show that neighborhood collaboration is unable—or at least insufficient—to provide a mechanism for agents without resources to indirectly or directly manipulate the resources of others to the extent that they depend in order to completely remove system demand fluctuation. The bottle neck can only be of either of the following types:

1. resource-possessing agents do not offer enough control over their resources to other agents
2. collaborating agents do not have proper information to coordinate action on one another's behalf correctly

In other words resource-possessing agents are too autonomous with respect to resource-deprived agents to allow maximum demand fluctuation reduction. Since neighborhood collaboration allows for full indirect manipulation of all excess resources of neighboring agents the bottleneck is information.

As specified by the neighborhood design, if a resource-deprived agent requests to purchase precisely the power exceeding its average deficit from any

neighbor having unlimited resources, then that neighbor can respond by providing all the power necessary for the agent to eliminate its demand above average deficit. Given every such agent power request, a helpful neighbor will eventually have all the information needed to predict the demand over average deficit of the requesting agent to the same accuracy as the requesting agent itself. What the neighbor is lacking however, is information regarding the requesting agent's demand under average deficit. Without this information the neighbor can only make an educated guess regarding the demand below average deficit.

Figure 7.7 and Figure 7.10 illustrate how an agent and its neighbor would collaborate in the scenario just described above if the agent's deficit profile was sinusoidal; the blue curve represents the agent's deficit. The agent requests power from its neighbor when the deficit is above average. Using the information gained from the requests it receives, the neighbor is able to prepare resources and collaborate with the requesting agent. The grey curve illustrates how the neighbor would have prepared its resources in order to collaborate with the agent and how the transaction is conducted after the preparation. In particular, notice how the preparation is a flat constant; this is because, the neighbor does not know any better than the expected deficit below average corresponding to the collaborating agent.

Figure 7.7 illustrates the effective demand profile of the agent after its demand is adjusted by collaborations with its neighbor. Figure 7.7 and Figure 7.10 illustrate how the neighbor's lack of information regarding the agent's deficit below average can in general produce a limit on load factor improvement since an agent's deficit

below average is seldom the expected (average) value thereby making it unlikely that the neighbor can properly cancel out the demand fluctuation related to the agent's demand below average deficit.

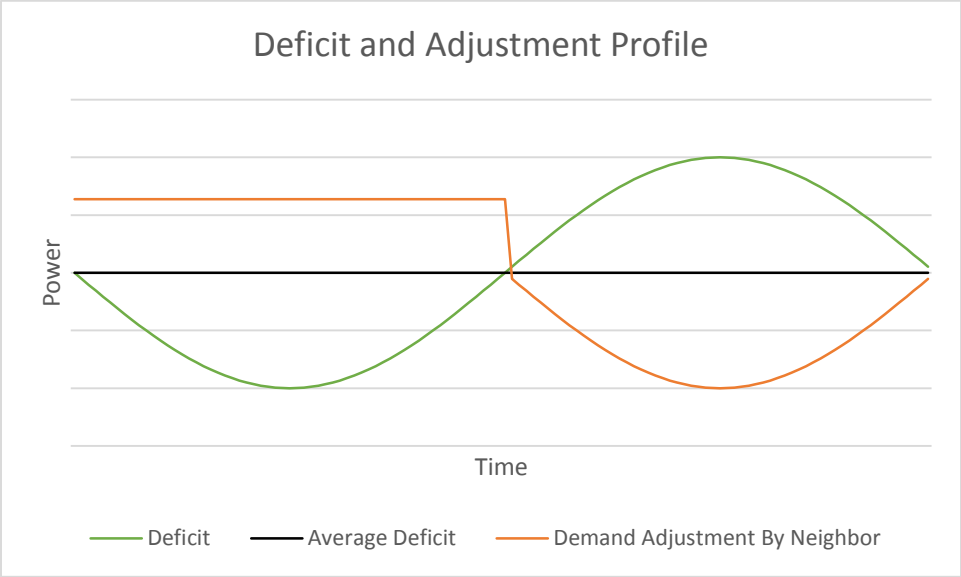


Figure 7.6: An illustration of how an agent's neighbor with a large amount of excess resources would behave given the agent's deficit profile in green.

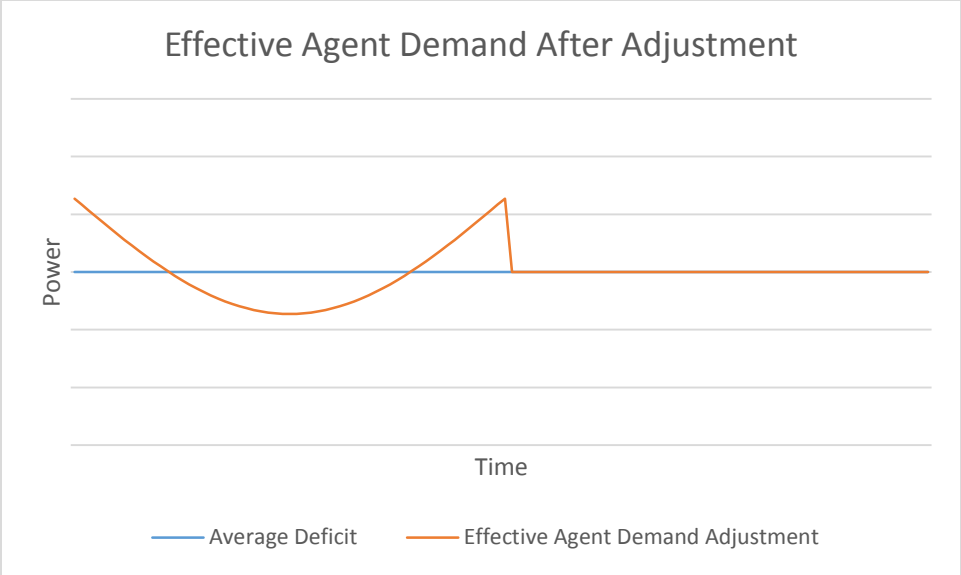


Figure 7.7: The effective demand of the agent after the collaborating neighbor's demand adjustments are completed.

Under such restrictions and the proposed approach, the best a neighbor can do is completely eliminate demand fluctuation for a requesting agent's demand over average deficit and reduce, but not in all likelihood, eliminate the demand fluctuation for any arbitrary demand behavior below average deficit for the same agent. The only way the neighbor can improve on its collaboration effectiveness is through finding the missing information.

Although it is possible to glean some of the missing information from the environment through trial and error, say by observing electricity prices after making a change to collaboration, this little information must be extracted from the noise introduced by the trial an error of other agents and the dynamism of the system both of which quickly becomes impractical without new venues of collaboration as the number of agents increases.

There are two points here to take away:

1. The reason for a load factor cap is because of the lack of information neighbors have regarding the demand behavior below deficit of requesting agents.
2. If the missing information were made available to an agent's neighbors, then the neighbors would be directly or indirectly dependent on the source of the information and as such would have less autonomy.

In essence, given the restriction and proposed approach, improving the system performance would require agents to be less autonomous otherwise any performance improvement would be done without the missing information which results in a contradiction since the only way performance can be improved is if the neighbor demand adjustments cancel out those of the agent requesting power in which case the inverse of the neighborhood demand adjustments would be the missing information.

7.3.2 Understanding the Drastic Increase in Load Factor

Figure 7.5 shows that all agent types benefit dramatically from collaborating with commercial agent types with resources. In fact the rapid increase in load factor between resource factor 0 and 0.5, shown in Figure 7.1, is almost entirely the result of agents collaborating with commercial agent types having resources. The drastic load factor improvements seen in Figure 7.5 are due mainly to the large amount of resources that commercial agents have. Since each step in the resource factor multiplies each agent's resource capacity, the total resources of the system increase drastically given the large resources capacity of commercial agents. The high quantity of commercial agent resources causes the performance cap to be reached much faster

than that of the home agents explaining the knee in the neighborhood load factor at resource factor 0.5 in Figure 7.1.

Since resources among all simulation cases are the same, the large amount of resources corresponding to commercial agents does not explain why the load factor of the neighborhood simulation case below a resource factor of 0.5 is almost as high as the load factor of the unity simulation case (see Figure 7.1). The nearly identical performance of the neighborhood and unit simulation cases for small resources has interesting implications in the field of MAS-based fluctuation reduction solutions, including MAS smart grids, since the results suggest that for practical resource capacities the neighborhood system does perform near optimal under the proposed approach.

Figure 7.1 shows that for small amounts of excess resources neighborhood collaboration provides enough information to participating agents to leverage their excess resources as effectively as in the unity simulation case. This may be in contrast to our expectations since neighbors are missing the information needed regarding requesting agent demand under average in order to properly adjust the demand corresponding to this region as illustrated in Figure 7.6 and Figure 7.7.

The reason why neighbors are so effective despite their lack of information is that small demand adjustments based on the expected demand under average deficit behavior, leads to an adjustment profiles which is similar to that of adjustments based on the missing information. In other words, for small demand adjustments, the adjustment profile of the neighborhood and unity simulation cases are similar for

adjustments to demand under average deficit. The demand over average deficit can be adjusted just as effectively by neighborhood collaboration as it can if the resource-requesting agent managed the excess resources itself. For the sake of clarity, Figure 7.8 and Figure 7.9 illustrate how for small enough demand adjustments, optimally adjusted demand and demand adjusted by the neighborhood are negligibly different.

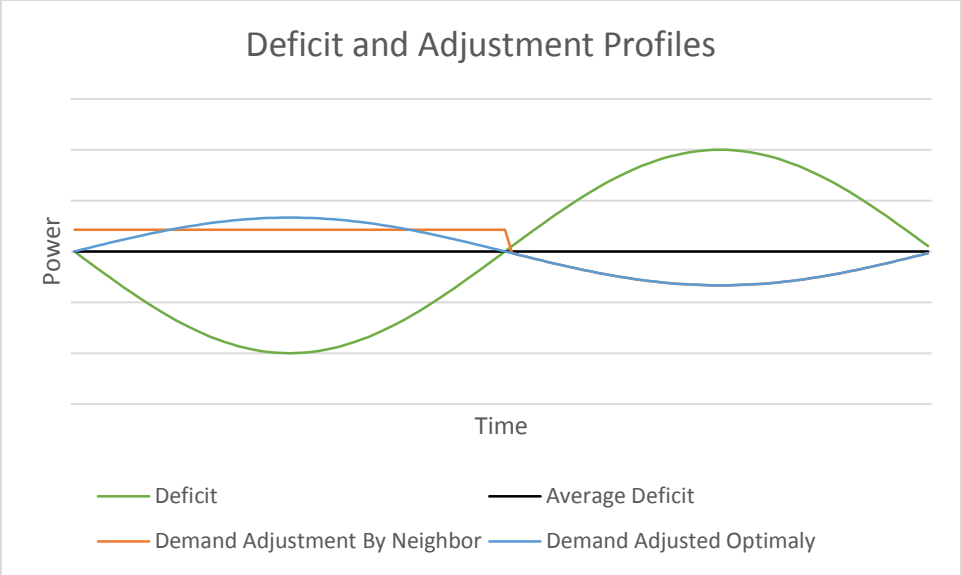


Figure 7.8: An Illustration of how optimal demand adjustments and neighborhood adjustments can be negligibly different for small enough adjustments.

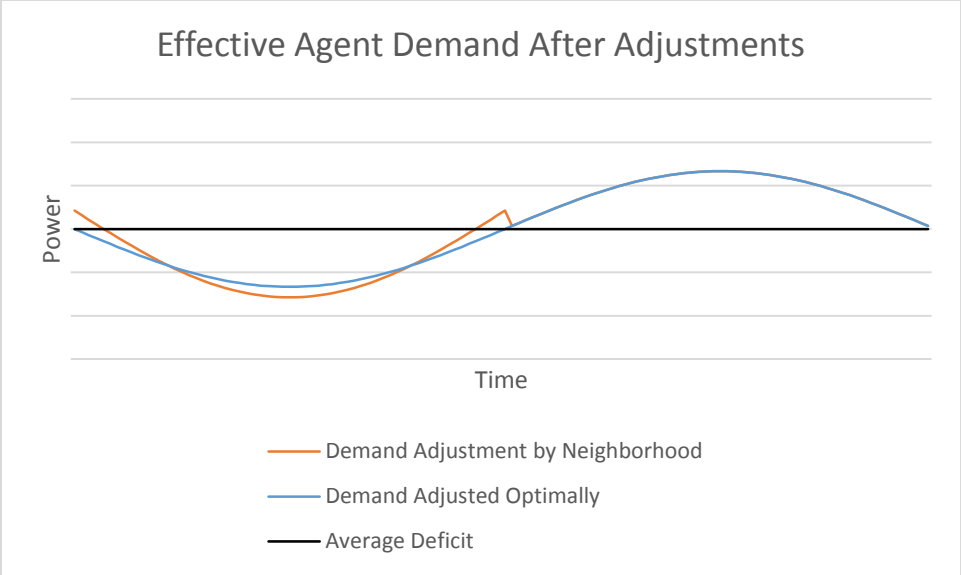


Figure 7.9: An Illustration of how optimally adjusted demand and neighborhood adjusted demand can be negligibly different for small enough adjustments.

Comparing the illustrations in Figure 7.8 and Figure 7.9 to Figure 7.6 and Figure 7.7 outlines how the neighborhood performs better for small demand adjustments than for large demand adjustments.

In essence, from an information theoretic standpoint, for small adjustments, the information requirements are very low since the adjustments being made for each hour under the proposed approach are all very similar and well approximated by their average. In other words, since the adjustments for each hour all have a more equal chance of happening, the information needed to select each hour's demand adjustments is less. Therefore, for small adjustments, even a little information regarding the demand adjustments under average deficit, such as, the total sum of demand adjustments under deficit in the case of neighborhood collaboration, goes a long way.

Therefore, the neighborhood simulation case can sustain low agent information dependence, which translates to higher autonomy, while still performing near optimal given the proposed approach for low resource availability conditions.

7.3.3 Understanding the Inflection Point

The inflection point is caused by neighbors putting personal interest ahead of other agents. Neighbors will only partake in collaboration if they are sure that in the future their excess resources will not be needed to reduce their own unpredicted demand fluctuation. The reason is that a neighbor does not know for a certainty if the currently requesting neighbor will indeed in the future request to collaborate again. In such a case the neighbor puts its own demand fluctuation, even if the extent of fluctuation is improbable, over any external request since the request is not reliable. In other words, the agents are aware of the nature of ad-hoc collaborations not having any guarantees and therefore will only prepare resource for another agent if that perpetration does not conflict with its own.

As resource capacity increases from nil, agents focus their resources on reducing their own demand fluctuation. As resource capacity further increase, these agent will find semi-excess resources beyond their average requirements becoming available; however, due to the stochastic nature of their deficit the agents find they can still leverage these resources to stabilize deficit outliers even if only seldom. Although the agent cannot fully benefit from the semi-excess resources, it will be reluctant to commit the resources to meet external requests since the requests are not guaranteed to repeat. At some point the resources become enough to where an agent

no longer worries that it cannot reduce its own fluctuation more than it is sure it can reduce the fluctuation of a requesting agent. It is at this point in the resources dimension that the neighbor can dedicate excess resources to a requesting agent. As the resources of neighboring agents reach this transition point, they are less and less able to use semi-excess resources in reducing their load factor causing the increase in load factor of the system to diminish. Once the transition point is passed, the agents suddenly start using the excess resources to reduce the load factor of the combined collaborating pair. Since resource capacity and deficit are stochastic, each neighbor may reach this transition point sooner or later than other neighbors causing an inflection point to appear in the load factor as opposed to a sudden jump.

The inflection points in Figure 7.5 are not very pronounced since the load factor cap is reached so fast. In order to compare the amount in which the commercial agents and homes having resources are each able to contribute to the performance of other agent types through neighborhood collaboration, we repeated the simulations shown in Figure 7.5 such that the total system resources capacity increases at the same rate as the simulations shown in Figure 7.4. The results of simulations appear in Figure 7.10.

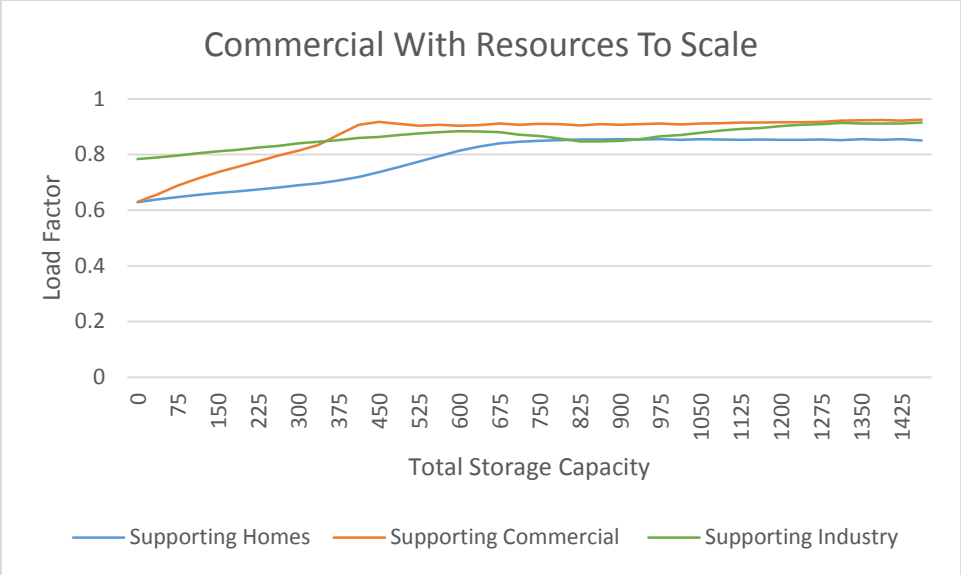


Figure 7.10: The load factor corresponding to the resource factors roughly between 0 and 1.25 from Figure 7.5 where instead of resource factor the horizontal axis is in total storage capacity. The total storage capacity of each horizontal tick corresponds to the total storage capacity of at each resource factor tick in Figure 7.4.

Table 7.4 provides the approximate storage capacity corresponding to each of the load factor inflection points of each of the 6 simulation cases described in Table 7.3. The results were found by observing a change in sign in the rate of change of the load factors of each simulation case; there was an outlier in the home-to-home collaboration case which produced a short lived inflection point for which we compensated.

Table 7.4: The approximate storage capacity observed from Figure 7.4 and Figure 7.10 where the load factor inflection point occurs for each of the simulation cases in Table 7.3.

	Home with resources	**Commercial with resources**
Home without resources	300	565
Commercial without resources	412.5	262.5
Industry without resources	450	412.5

We make the following observations from Table 7.4 and the storage capacity points where the load factor first top off in Figure 7.4 and Figure 7.10:

1. The home-to-home and commercial-to-commercial load factor inflection points happen at a lower storage capacity point than that of the home to commercial and commercial to home simulations.
2. The home to industry and commercial to industry load factors do top off at much higher storage capacities than that of the other collaborating type pairs
3. The home to industry and commercial to industry load factors top off at storage capacities higher than 1200 units but their inflection points happen at nearly $1/3$ the storage capacity

7.3.3.1 Similar Agents Collaborate More Effectively

Since similar agents have similar deficit profiles there is less tension between choosing self-interest over collaboration for agents having resources. The reason is because similar agents can benefit from the same demand adjustment strategy;

therefore, adjustments that an agent would make for itself do not conflict with the adjustments the agent would make in order to collaborate with its neighbor. In such a case an agent will find that excess resources perpetrations, although still at the risk of not ending in collaboration, are likely to be helpful to the agent itself to some extent in the case of outlying deficits. For this reason, home-to-home and commercial-to-commercial collaboration achieve load factor inflection points at lower storage capacities than that of the other collaborations. Consequently, the collaboration performance of similar agents is higher than that of dissimilar agents.

7.3.3.2 Industrial Agents Collaborate More Ineffectively

They key feature of industrial agents is their massive deficit peaks. The reason for collaboration among homes and industry and commercial and industry not improving load factor as fast as that of other collaborating agent types is due slightly to the fact that much more resources are needed in order to reduce the large demand peaks caused by industry. More significantly though is that as larger and larger demand adjustments become possible with the availability of more resources, so does the resource preparations. The larger the resource preparations the larger the demand adjustments made by the neighborhood when the requesting agents demand deficit is below average. As explained in Subsection 7.3.2, for such large adjustments, the difference between the adjustment made by the neighborhood and the adjustment that would have been possible if information about the deficit profile below average were available is very pronounces. As a results, such large adjustments are suboptimal compared to the best adjustments possible given the proposed approach.

Consequently neighborhood collaboration performance when collaborating with industry type agents is very poor.

7.3.3.3 Industrial Agent Collaboration Inflection Points

The reason for the load factor inflection points of collaborating agent type pairs involving industrial agent types being so distant from the point where the load factor tops off is because the inflection point only has bearing on the agent being able to transition from only considering immediate self-interest and interest gained through collaboration. Home and commercial agents are able to satisfy their personal requirement with relatively little resources with respect to industry agents. Therefore the load factor inflection points for home and commercial agents should be relatively the same for any collaboration varying only as a result of tension between differences in deficit profile among the collaborating parties. There is no exception for home or commercial collaborations with industry agents.

7.3.4 Understanding Load factor Dip

A dip in load factor corresponding to the case where commercial agent types with resources collaborate with industrial agent types, is well pronounced in Figure 7.5 and Figure 7.10. The dip in load factor is the result of requesting agents being indecisive about with which of their neighbors to collaborate. This indecisiveness comes about when both some of the neighbors of a requesting agent trade roughly the same portion of the original request and also sometimes fail to trade due their personal interested being at risk. In such a case, the requesting agent's collaboration preferences very around a critical point which when crossed causes the requesting agent to reorder

whom and how much it wishes to collaborate. The constant change in collaboration trends causes spikes in trade requests from neighbors which in turn make it more difficult for the neighbors to predict and rely on the requesting agent's future collaboration.

Figure 7.11 shows the aggregated requests placed by requesting agents from neighboring agents and the corresponding aggregated request predictions made by the neighbors for the cases where commercial agent types having resources collaborate with industrial agent types. Figure 7.11 top and bottom, correspond to the resource factor of 0.5 and 0.75 respectively. One would expect that with more resources the neighbors should be able to collaborate more steadily; however, comparing the two graphs in Figure 7.11 one can see more inconsistent requests and prediction in the case having more resources. The correlation between predictions and requests in the top graph is 0.912 and in the bottom is 0.849; meaning that the request predictions of the case with more resources are more inaccurate. It is important to note that the values shown in Figure 7.11 are aggregates across all agents in the simulation case and therefore the aggregations cancel out the majority of local fluctuations in neighborhood request and request predictions. As such, even a small decrease in correlation between aggregated request and predicted request, suggests a much more pronounced decrease in correlation for a per agent basis. The lower correlation in predictions and requests have more severe consequences in the case of the industry and commercial agents over other collaborations involving one of these agents since:

1. The deficit profiles of the commercial and industrial agent types allow for more tension in the choice the commercial agent faces between self-interest and collaboration.
2. The industry agent's deficits have very high peaks, as a result, in the case that an industry shifts from requesting to collaborate with one commercial agent to another, the change in requests being observed from each of the commercial agents drastically changes.

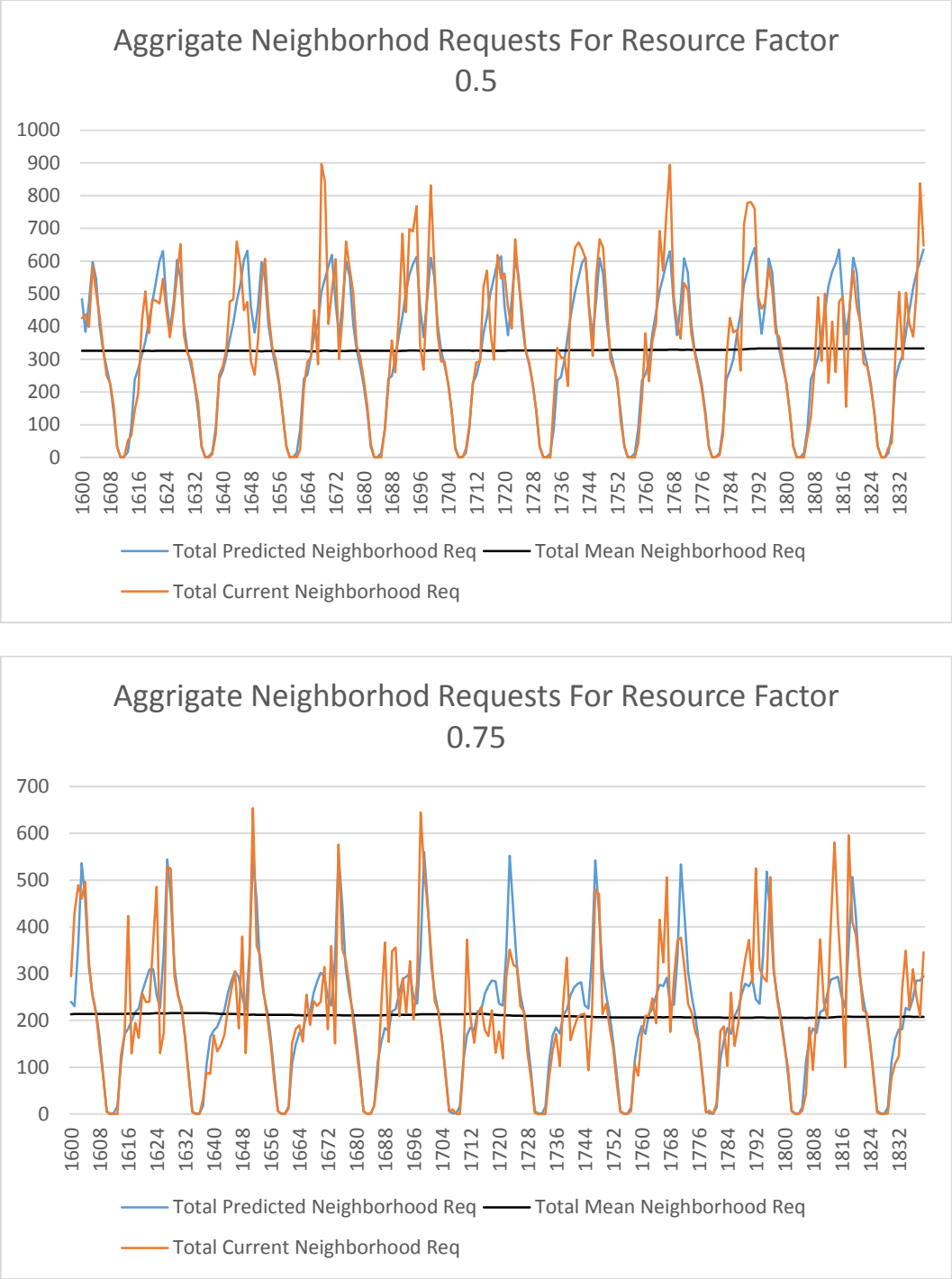


Figure 7.11: The aggregate neighborhood requests and predicted request corresponding to the commercial agent types having resources collaborating with industrial agents having a resource factor of 0.5 (top) and a resource factor of 0.75 (bottom).

7.4 Time to Equilibrium

In extremely dynamic environments it is often more important that the time to equilibrium be low than the actual equilibrium be near optimal. It is often the case for chaotic or extremely dynamic environments that over the time needed for a system to reach equilibrium performance the environment would have already transitioned to a different state. In this section we take a look at how the time to equilibrium of the proposed approach fair for each simulation case.

We measured the load factor standard deviation of a moving window of 96 hours (4 periods) starting from each hour for each simulation case. We approximated the time required to reach equilibrium by averaging the first hour where the ratio of the corresponding load factor stander deviation over that of the next hour was within $\pm 0.5\%$ of 1. In other words, we measured the average simulation time required to reach the first 96 hour period where the load factor was stable. Figure 7.12 shows average time required to reach equilibrium as the resource factor increases. It is important to note that the manner in which we measured the time to equilibrium is very subjective to the moving window length and the selected percentage change in standard deviation. Although this measure allows us to compare how rapidly each simulation case is able to adjust to it environment, the measure does not actually tell use when any particular simulation reaches equilibrium. In practice the state of being in equilibrium or not is often not discrete but rather a fuzzy measure; consequently, measure the time required for a system to reach equilibrium in an absolute manner is often not practical.

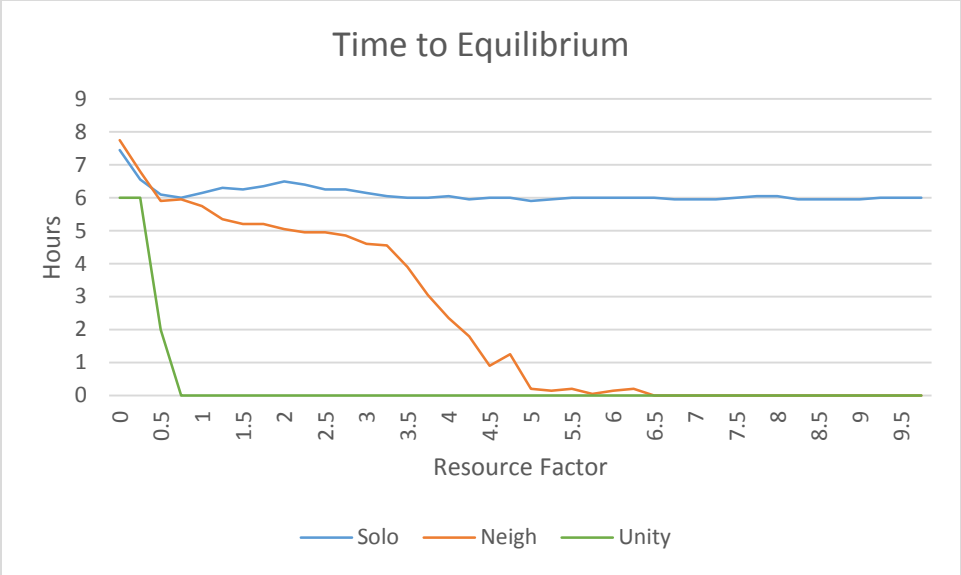


Figure 7.12: The average number of simulation hours it tokes each simulation case to reach equilibrium as measured in the manner described in this section..

The results show that the solo agents roughly take a constant time to reach equilibrium whereas with enough resources neighborhood collaborating agents and unity agents are able to ultimately reach equilibrium in a negligible amount of time.

In order to understand how reliable each of the simulation case systems are in quickly attaining equilibrium we measured the standard deviation of the time to equilibriums of each of the 20 simulation repetitions (see Figure 7.13). A high confidence in time to equilibrium means that the attributed system is expected to reach equilibrium in a reliable manner every time the environment state changes whereas a low confidence in time to equilibrium suggests that the attributed system may not in some instance be able to adapt to sudden changes in environment very reliably.

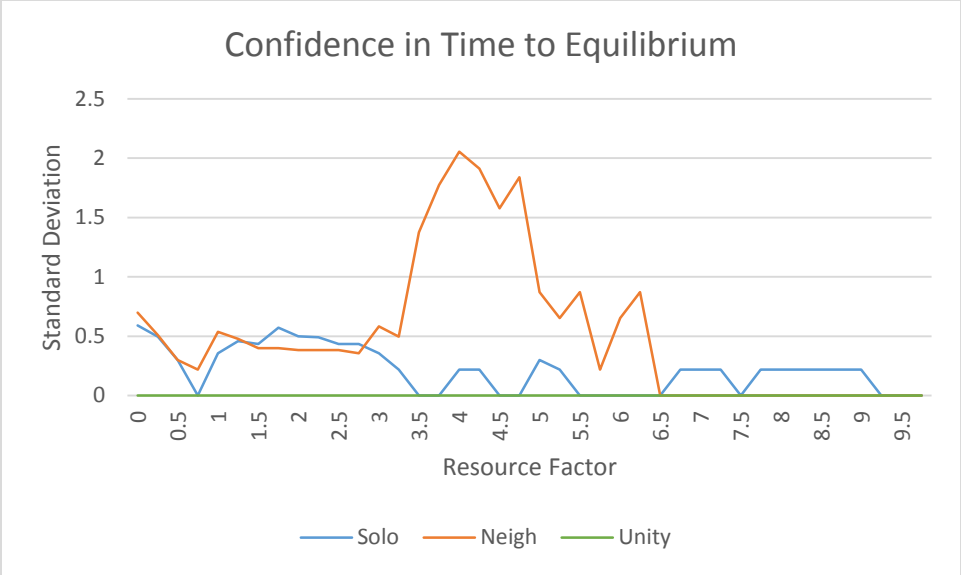


Figure 7.13: The confidence in time to equilibrium for each resource factor and simulation case. The confidence is measured in units of standard deviation of time to equilibrium.

Although, for the most part the results in Figure 7.13 are not conclusive, the results do support that the time to equilibrium of the unity simulation case system has near maximum confidence whereas the neighborhoods confidence progressively reduces until peaking at a resources factor of roughly 4 and subsequently increasing to near maximum. The peak lack of confidence of the neighborhood simulation case corresponds to the inflection point visible in Figure 7.1 occurring at the same resource factor.

These results are expected since:

- The unity simulation case all agents are aware of all information relevant to the proposed approach and therefore are not struck by any surprises when following the approach.

- Neighborhood collaboration is not reliable particularly when the resource factor nears the inflection points where an agent is struck whether to pursue its self-interest or collaboration.

Despite the time to equilibrium confidence of the neighborhood simulation case being generally lower the decreasing average time to equilibrium with respect to that of constant time to equilibrium of the solo agents can make the neighborhood enable system an option for environments where the solo agents are not applicable.

CHAPTER 8 MEASURING AUTONOMY

Over the course of our observations we have identified and measured the deficiencies of solo agents with respect to other more collaborative agents. The short coming of the solo agents are:

1. Solo agents cannot stabilize the system demand fluctuation beyond reducing their local demand fluctuation to virtually its minimum. In other words, solo agents, even with excess resources, cannot actively contribute to reducing the demand fluctuation of other system agents
2. Solo agents have no information about the demand profiles of other agents and as such cannot actively take advantage of demand fluctuation cancelations among multiple agents. In the worst case, it may be that, independently, solo agents suffer extreme demand fluctuations while the system as a whole is completely without demand fluctuation. Under such conditions the solo agent will still consume resources to reduce local demand fluctuations.

As a result of the shortcomings listed above, solo agents have the lowest performance of all the other agents. In other words, solo agents are unable to fully leverage their resources in order to attain high load factors. As explained earlier, in order to overcome these limitations, solo agents must have more information. This information cannot be gained without introducing inter-agent dependencies. Therefore, in order to advance performance, solo agents must sacrifice autonomy by some means or another.

In general reduced autonomy comes at a cost. Under such conditions it is naturally of interest to find the conditions where:

1. the costs of reducing autonomy offers the largest amount of system performance improvement benefits
2. the minimum cost of reducing autonomy is accepted in order to satisfy some system performance constrain

In line with finding these points of interest, over our study we have suggested a means to find system performance bounds in terms of autonomy; namely, to hold a solution approach and all relevant configurations short of degree in which an agent is informed and in control of its external environment, constant. In this chapter we define a means of quantifying the autonomy of direct or indirect collaboration techniques relative to an MAS solution approach.

We start our discussion on measuring autonomy by iterating over some key concepts that have been, in previous chapters, presented in general or used in the context of the smart grid or the proposed demand fluctuation reduction approach. In Section 2.2 we defined autonomy to be the degree in which a set of goals can be independently realized. Over our discussion of the simulations and their results in Chapter 6 and Chapter 7 we refer to the term agent performance as a measure of how well an agent can realize its goals given its dynamic environment and a set of resources. Simply, performance gauges how well an agent uses its resources in order to realize its goals all while under constraints of its environment. Crucially, performance is not a function of a particular state of the environment. For instance, it

is possible an agent effectively uses its resources to achieve its goals at unproblematic moments in the environment while under the regular environment stress, it may fail to properly utilize resources and achieve goals. Such an agent would have poor performance since it is not adept to its environment.

In general, aside from personal information and given resource, an agent can only be dependent in achieving its goals on information and control over the environment external to itself. It is normal for an agent to be dependent on its environment; however, it is desirable that an agent be less dependent on some parts of an environment such as other agents, users, or facilities. When measuring a particular type of agent autonomy, we are measuring how much the performance of that agent is dependent on its information and control over some subset of its external environment. In Section 6.1 we defined the information and control an agent has over its external environment to be the authority of the agent over its environment, or simply authority. We have used the notion of agent authority in designing and explaining the smart grid simulation cases, particularly in the introduction of Chapter 6 and Chapter 7 and in Sections 7.1 and 7.2.

8.1 Relative Enhancement Autonomy

We build off of the work done by Braynov and Hexmoor on quantifying relative autonomy [24] in order to define a means of quantifying the autonomy of an MAS with respect to the maximum potential of the solution approach. We recap the definition introduced in Section 2.5. For the sake of simplicity instead of considering

the autonomy of agent i with respect to a set of agents S, we will consider the particular case where the set S contains only agent j.

Definition 8.1: Relative Agent-Agent Autonomy

For some ratio scale measure of agent performance, let v_j^i denote the performance of agent i when in the presence of agent j. The relative autonomy, A_j^i of agent i with respect to agent j is:

$$A_j^i = \frac{v_j^i}{v_i^i}$$

At the heart of the definition, the relative autonomy A_j^i is the fraction of performance agent i gains when in the presence of agent j, over the performance agent i would have had relying on only itself. It is possible that the presence of agent j disturbs the performance of agent i by imposing some direct or indirect limitations on system control or visibility, or perhaps due to constructive collaboration the presence of agent j brings about performance improvements for agent i. At any rate, the performance changes are solely attributed to the presence of agent j.

It is worth clarifying that the performance of an agent, and consequently the relative agent-agent autonomy, is dependent on an often non-empty set of variables. An agent or system uses a set of resources and in order to accomplish its goals. The effectiveness of the agent or system in utilizing those resources to achieving some extent of their goals is the performance of that agent or system. Consequently; performance is a function of resources. Again, the measure of an agent's performance

captures the agent's ability to achieve its goals in its environment. As such, performance captures an agent's ability to succeed regardless of environment dynamism and chaotic events.

It is noteworthy that changes in agent-agent autonomy measures do not directly map to the natural quality measure of the autonomy concept; meaning, an increase or decrease in an agent-agent autonomy measure, respectively, does not directly mean that the agent is more or less autonomous. For instance, it is possible $A_j^i > A_j^k$ for agents i, j and k, which, contrary to expectation, depending on application, can be interpreted as agent i is less autonomous than is agent k with respect to agent j.

We extend Definition 8.1 to allow for the relative autonomy associated with a solution approach enhancement to be measured. Let S denote the MAS for a given solution approach and let $G(S)$ denote the set of goals of the agents of S such that, for any $g_1, g_2 \in G(S)$, it is not possible for either g_1 or g_2 to fully contain the other unless $g_1 = g_2$. In other words let $G(S)$ be the set of generalized goals of S where each goal has a unique component which is not addressed by any other goals. Furthermore, let S_k denote the MAS following the prescribed approach having enhancement k where an enhancement is defined as any agent adaptation where the final agents have more authority and $G(S) = G(S_k)$ such that the added authority results in improved goal realization. Put simply, an enhancement must improve agent performance through increasing agent authority all while leaving the goals defined by

the approach unchanged. By the definition of enhancement, an enhanced agent must be less autonomous than the original agent.

We clarify that an enhancement to an approach is a different means of achieving the goals defined by the approach itself. The goals defined by an approach may or may not be identical to the goals defined by the optimal solution. Therefore, an approach enhancement is limited by the goals of the approach itself and consequently can only improve the system to the extent of which optimally achieving the approach goals are possible. In many cases the approach goal may indeed be exactly that of the optimal solution. But this does not need to be the case since often times the approach suggests an approximation of the optimal solution. It is worth clarifying that an agent or environment property change does not necessarily constitute an enhancement to an approach if access to information or control over the external environment with respect to any agent remains the same. By definition, changes which do not impact the information or control of any agent do not introduce any changes in autonomy relative to themselves and an external entity since otherwise the enhancement would be performing better without any additional authority which would require that the enhancement not follow the approach. Examples of an enhancement could be:

- various learning, since an agent produces a model which can extract information from observation data or allow for a new point of control to be discovered

- various agent collaborations, negotiations, voting, auctioning, etc., since each of these mechanisms may introduce new means of control or state information to the agent they did not already have

We are interested in such a definition of enhancement because we are interested in describing a means of measuring the autonomy changes introduced by generalized MAS mechanism since these enhancements are fundamental techniques in MAS research to improve existing solution approaches.

Definition 8.2: Relative Enhancement Autonomy

For some ratio scale performance measure, let v_S and v_{S_k} be the performance of S and S_k for some approach and approach enhancement k with goals $G(S) = G(S_k)$ respectively. The autonomy, $A_S^{S_k}$, of S_k with respect to S is:

$$A_S^{S_k} = \frac{v_{S_k}}{v_S}$$

Since the objective of approach S and enhanced approach S_k are the same the performance measurements of v_S and v_{S_k} are comparable. If the objectives of S and S_k were not the same, then it would likely be the case that a measurement of how effective S is at achieving $G(S)$ cannot be compared with a similar measurement of how effective S_k is at achieving $G(S_k)$. Under such conditions the measurements would likely not be comparable since, either the measure would not capture the effectiveness of achieving both goal sets simultaneously or if so the scale of the

measures would not be the same. Since $G(S) = G(S_k)$ it must be the cause that the performance measures of S and S_k are comparable. Similar to Definition 8.1 the autonomy measure defined in Definition 8.2 is generally a function of several variables.

By definition, the aggregated effects of autonomy internal to S is a base-line dictated by the solution approach; therefore, up to what is required by the solution approach the internal autonomy of S is maximal. By definition, the difference in performance of S_k is solely dependent on the reduced autonomy introduce by the enhancement to the approach. Last but not least, the definition requires that the goals of the two systems being compared be the same. Therefore, Definition 8.2 is based on the same principles as Definition 8.1. Table 8.1 summarizes how Definition 8.2 is extended from Definition 8.1.

Table 8.1: An outline of how Definition 8.2 extends from Definition 8.1.

	Definition 8.1	**Definition 8.2**
Objectives	The objective of agent i is constant regardless of whether j is present or not, therefore the performance measurements v_j^i and v_i^i are comparable	By definition, the objective of approach S and enhanced approach S_k are the same, therefore the performance measurements v_S and v_{S_k} are comparable

Measure in the numerator	Performance of agent i when directly or indirectly interacting with agent j beyond the interactions it would have had otherwise; in other words, performance of agent i when affected through any channel of autonomy triggered by the existence of j	System performance of agents directly or indirectly interacting with each other beyond that which would have been possible otherwise; in other words, the system performance of S when effected through the channels of autonomy introduced by enhancement k
Measure in the denominator	Baseline performance agent i has in the presence of only itself	System baseline performance of the approach S only

Essentially, Definition 8.2, measures the fraction of performance gain that is solely produced by direct or indirect collaboration introduced by the enhancement, to that of the base-line approach which only allows for collaboration up to what is required by the approach. In other words, Definition 8.2, captures the percent performance improvement brought about by increased collaboration. Again this collaboration may be constructive or destructive. At any rate, any change in the measure value of Definition 8.2 is directly the result of changes in autonomy brought about by the enhancement, and therefore, provides a measure of the autonomy of the enhancement with respect to the approach. As such, Definition 8.2 allows one to determine the extent of any introduced dependencies agents have in achieving their

goals after incorporating various common MAS techniques, such as learning, coalitions, etc. into a previous solution approach.

Similar to agent-agent autonomy, relative enhancement autonomy does not directly map to the qualities associated with the notion of autonomy. In fact, relative enhancement autonomy has an inverse relation with the notion of autonomy. This means the higher the measured relative enhancement autonomy the lower the autonomy and visa-versa.

Relative enhancement autonomy measures the autonomy of the enhancement free of the maximum autonomy dynamics inherent to the approach. As such, relative enhancement autonomy captures variations in autonomy introduced by the enhancement alone. For instance, let us assume the maximum autonomy inherent to an approach fluctuates as more resources are made available. Relative enhancement autonomy would allow for the changes in autonomy introduced by the enhancement to be measured without the measures being influenced the approach in which the enhancement is based. As such, the measures are clear of the fluctuating dynamics of the maximum autonomy limits inherent to the approach.

One of the weaknesses of relative autonomy is that the scale of the measure is not uniform for different resources. This is because, for different sets of resources it is possible that the corresponding minimum autonomy attainable by the approach be different as well. Although relative enhancement autonomy captures only the changes in autonomy resulting from the enhancement, those changes are measured relative to a scale which varies depending on the minimum autonomy of the approach. Although

for any given set of resources, using relative enhancement autonomy, it is possible to contrast the autonomy of various enhancements clear of the dynamics of the maximum autonomy of the approach the enhancements share, it is often not very meaningful to compare measures corresponding to different sets of input resources since each is subject to a different scale.

8.2 Smart Grid Performance Measure

Until now we have been using load factor in order to directly measure the demand fluctuation of a smart grid agent or system. Although, in our discussion of the smart grid we have used the term performance, we did not quantify any measure other than to suggest the performance of an agent or system is not only related to its load factor but to the amount of resources it requires to achieve any such load factor.

In essence, the suggested approach spends storage space and load suspension tolerance in order to reduce demand fluctuation. Therefore, the performance of an agent or system based on this approach is dependent on how effectively the agent or system can use any given resources to achieve its goal. Consequently, smart grid performance is a function of these resources. Performance is also dependent on other variable as well; however, we hold these variable constant in our simulation instances.

There are many ways to define a measurement for this performance definition. For simplicity we elect the following performance definition. Let the performance of a smart grid agent or system be the load factor per unit of resources the agent or system is able to achieve for a given amount of resources in its expected environment.

Particularly, in the case of our simulation tests where agents never had any tolerance for load suspension, the smart grid performance measure is simply the agent or system load factor to unit of storage capacity for a given amount of storage. Let $\lambda_S(r)$ signify the load factor of smart grid system S and storage capacity r then the system performance $v_S(r)$ is:

$$v_S(r) = \frac{\lambda_S(r)}{r} \tag{8.1}$$

8.3 Smart Grid Autonomy Measure

Given the relative enhancement autonomy from Definition 8.2 and the performance equation (8.1) the smart grid autonomy measure is:

$$A_S^{S_k}(r) = \frac{v_{S_k}(r)}{v_S(r)} = \frac{\frac{\lambda_{S_k}(r)}{r}}{\frac{\lambda_S(r)}{r}} = \frac{\lambda_{S_k}(r)}{\lambda_S(r)} \tag{8.2}$$

We extend the definition of $A_S^{S_k}(r)$ for the case where $r = 0$ to be:

$$\lim_{r \to 0^+} A_S^{S_k}(r) = \lim_{r \to 0^+} \frac{\lambda_{S_k}(r)}{\lambda_S(r)} \tag{8.3}$$

Figure 8.1 shows the results of measuring the relative enhancement autonomy for each of the simulation cases.

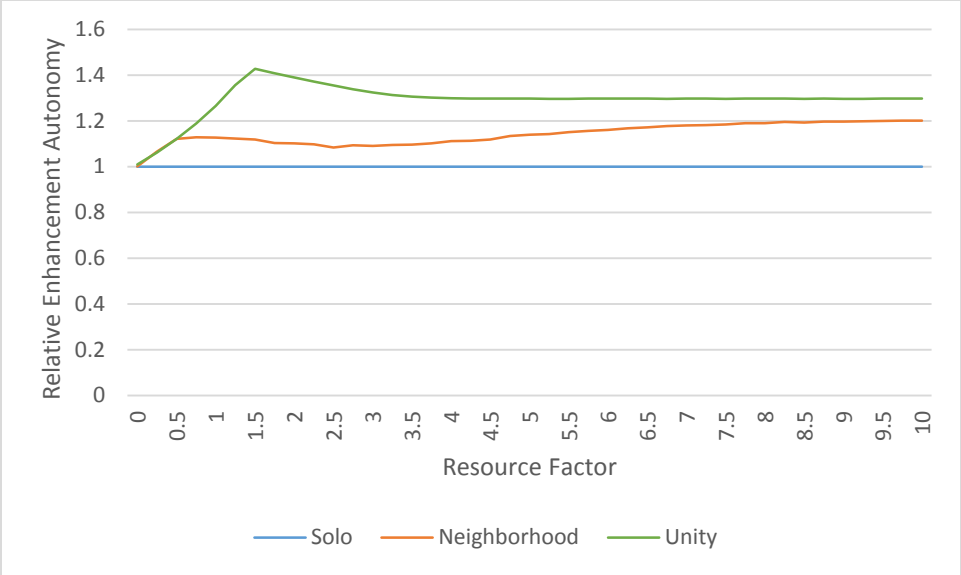


Figure 8.1: The change in autonomy beyond that of the solo simulation case as introduced by collaboration in the unity and neighborhood simulation cases.

The fraction of performance each of the unity and neighborhood simulations have over the solo simulation measure the extent at which the goal of each simulation is realized as a sole consequence of the agent collaboration introduced by each simulation case. In other words, the measure indicates what extent of goal realization is dependent on the agent collaboration enhancements of the unity and neighborhood simulations. As such, the values in Figure 8.1 capture the autonomy inherent to the collaboration tactics of the unity and neighborhood simulations.

We are only interested in presenting the relative enhancement autonomy measurement and motivating its value and short comings, so we will not delve into analyzing the results of the relative enhancement autonomy measure of the smart grid simulation cases in fine detail.

From Figure 8.1 we can conclude that as the resource factor of 2.5 is approached the agents of the neighborhood simulation case become less and less

reliant on the neighborhood to achieve their goals with respect to their independent strategy. As such, at a resource factor of 2.5, the agents of the neighborhood simulation case are less reliant on neighborhood collaboration itself in order to achieve their goals than that of the same agents at any resource factor greater than or equal to 0.5.

Given the ability to measure relative enhancement autonomy one can for instance know from measurement shown in Figure 8.1 that at a resource factor of 2.5 an 8.38% percent performance improvement is entirely the result of neighborhood collaboration enhancement made to the base-line approach of the solo simulation case.

More interesting perhaps is the peak and subsequent dip of relative enhancement autonomy of the unity simulation case. At the peak relative enhancement autonomy achieved at resource factor 1.5, the unity simulation system performance shows a 42.75% improvement over the baseline solo simulation system which is solely attributable to the unrestricted collaboration of the unity simulation. At this point the approach reaches its maximal dependence on collaboration. In other words if at the resource factor of 1.5 all collaboration in the unity simulation was lost the system would lose the ability to achieve an equivalent of 42.75% of the goal achievable without collaboration. Such information not only enables the reliability of the system to failing collaboration, memory, learning, or some other enhancement affecting autonomy to be gauged and fine-tuned but also allow the dependencies inherent to an approach enhancement itself to be gauged and fine-tuned to meet

computational complexity requirements. For instance, given relative autonomy measures, the resources of a neighborhood trading enabled smart grid can being tuned so that the agents not only perform well but are also as independent to neighborhood collaboration failures as possible or perhaps the infrastructure supporting the neighborhood collaboration could be budgeted to reflect the dependency of the MAS.

The dip in relative autonomy of the unity simulation case as the resource factor increases beyond 1.5 is entirely due to the increased relative performance of the base-line solo simulation case for the same resource factors. Since the agents of the unity simulation case are defined to be minimally autonomous up to what the approach allows, the corresponding relativity enhancement autonomy measures are maximal for any implementation of the approach. Similarly since the agents of the solo simulation case are defined to be maximally autonomous up to what is required by the approach, the corresponding relative enhancement autonomy measures are minimal for all implementation of the approach. Therefore all enhancements to the approach must have a relative enhancement autonomy measure between that of the unity and solo simulation cases. By definition the relative enhancement autonomy measure of the base-line approach is always the constant 1. Since the upper bound of the relative enhancement autonomy can scale freely with respect to the base-line for various MAS input parameters in a similar manner to that of the smart grid simulation, then all the values confined between the upper and lower bound are skewed by the relative nature of the measurement. As such it is hard to identify the trends in autonomy natural to an enhancement since the relative enhancement

autonomy for the enhancement will be influenced by the autonomy trends of the base-line approach. Therefore, relative enhancement autonomy is valuable when we are interested relational measurements and not when we are interested in measuring the absolute autonomy of an enhancement alone.

8.4 Absolution Enhancement Autonomy

Occasionally, given a set of input resources, it is possible to find the maximum performance of an MAS following a prescribed approach over the range of applicable authority options. In many cases, the ability to find a maximally performing MAS is possible simply because the approach itself dictates the scope of applicable collaboration and learning by describing what information and control is useful to the approach. In other cases, it may be impractical to find the maximum performing enhancement to an MAS or to even simulate such a system; however, it may still be possible to find the maximum performance of an approach by other means. This is possible since in some cases the approach, the fixed input resources, and the complete knowledge of the problem statement and environment allow for the maximum performance to be calculated, modeled, or synthesized. In the worst case, it is possible to impose reasonable assumptions allowing for an upper or lower bound on the maximum performance to be found.

It is often a much simpler task to find the minimum expected performance of a solution approach for a given set of resources when varying only authority. The reasoning is that if an approach and problem statement and their simulation environment, and finally the set of approach resources, are known then the

information and control each agent is minimally prescribed to have by the approach is already given.

For such approaches, where the minimum and maximum performance over the range authority prescribed by the approach for a given set of input resources is available, we define the absolute autonomy of an enhancement.

Definition 8.3: Absolute Enhancement Autonomy

For some ratio scale performance measure, let v_S, v_{S_k}, v_{S_M} be the performance of the approach S, and performance of the approach enhanced by k and M respectively such that $G(S) = G(S_k) = G(S_M)$ and M is the maximum performing enhancement of S for any set of input resource. The absolute autonomy A_{S_k} of S_k is:

$$A_{S_k} = \frac{v_{S_k} - v_S}{v_{S_M} - v_S}$$

The absolute autonomy A_S of approach S is defined to be 0.

It is important to notice that as a consequence of the definition of an enhancement, $v_S < v_{S_k} \leq v_{S_M}$. Furthermore, by definition of an enhancement it is not possible for $S = S_k$; hence, the explicit declaration of $A_S = 0$ in Definition 8.3. Finally it is worth noting by Definition 8.3, it must be that $0 \leq A_{S_k} \leq 1$ for any S, k and set of input resources. Similar to relative enhancement autonomy, absolute enhancement autonomy has an inverse relationship with the quantitative notion of autonomy.

The key benefit of absolute enhancement autonomy is that its measure is scaled to match the minimum and maximum possible autonomy of the approach. As such, the measure is not only reflective solely of the autonomy changes introduced by the enhancement, but the measure follows a uniform scale for across possible resources sets. This feature makes for the differences among autonomy changes introduced by different enhancements to be not only compared but also uniformly measured. Where it is not very meaningful to compare the difference of relative enhancement autonomy of two separate enhancements corresponding to two different resource sets it is meaningful to do so in terms of absolute enhancement autonomy. The key short coming of the absolute enhancement autonomy is that it requires a means of measuring the maximum possible performance of the approach. This is often not feasible since in the worst case one must solve the original problem whose infeasibility motivated the MAS approximation in the first place, and in the best case, a mathematical modeling, synthesizing (computational modeling) or estimating the maximum performance of the approach is required which is a difficult task in itself.

We close our discussion by considering the absolute enhancement autonomy of the smart grid simulations shown in Figure 8.2 as an example.

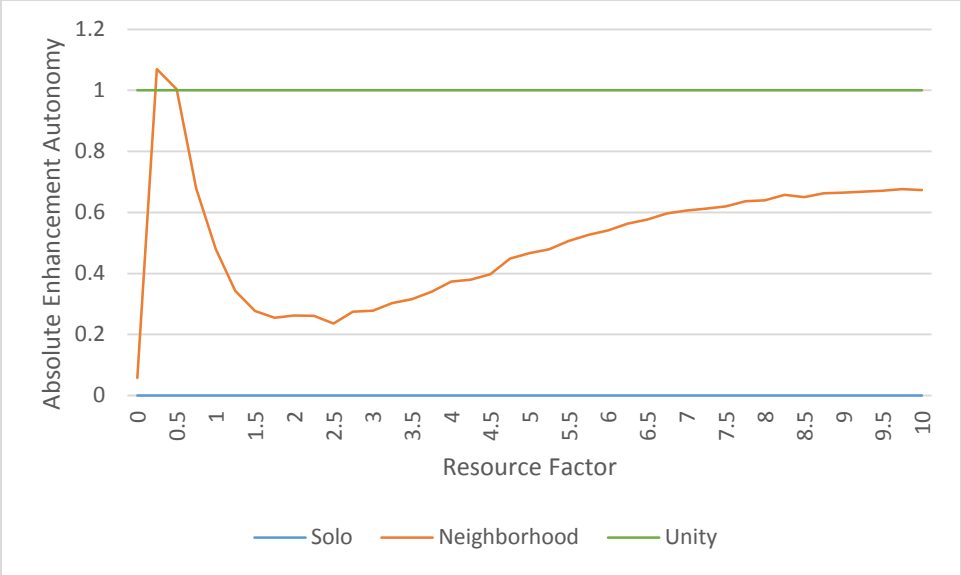


Figure 8.2: The absolute enhancement autonomy of the solo, neighborhood, and unity simulation cases.

As observable from Figure 8.2, due to the stochastic nature of the simulations, the measured results for the neighborhood simulation case are not exactly what is theoretically expected. Namely, the expected measurements for the resources factors 0, 0.25, and 0.5 are exactly 0, at most 1, and at most 1 respectively, which differ from the measures results. Nonetheless, the outlying measurements miss the expected values with an average error rate of 4.3%, which is well within tolerance for outlying samples. The error could have been reduced by running each simulation more than 20 times and taking the median performance instead of the average, or completely eliminated if we were to modify the simulation infrastructure to replay the same random events for each simulation case. At a resource level of 0 all simulation cases are reduced to the same plain grid and no longer have interesting smart grid properties. Figure 8.3 show the results after cleaning the data, applying polynomial

regression to reduce noise, and excluding the sample related to the uninteresting 0 resource factor.

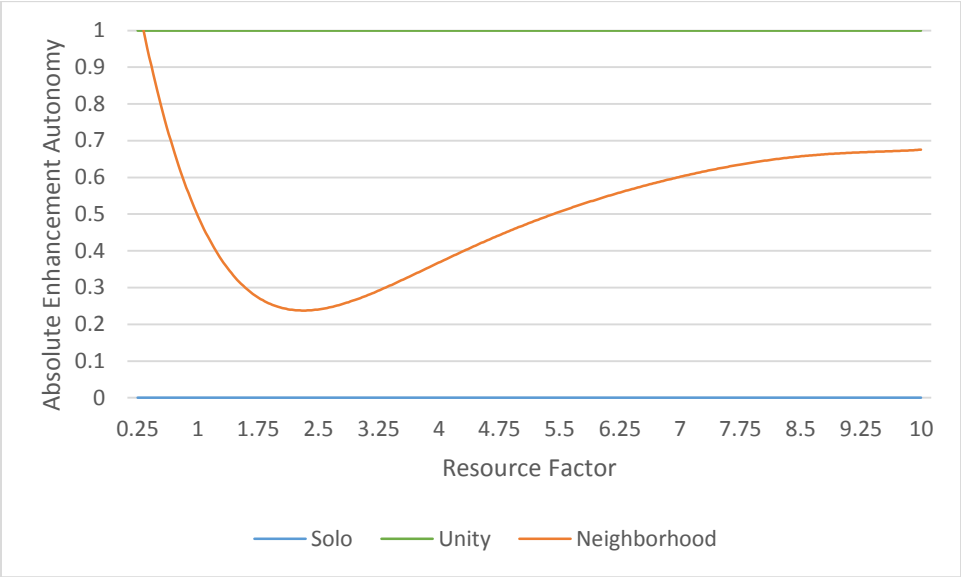


Figure 8.3: Absolute approach enhancement autonomy of the simulation cases after cleaning the data and using polynomial regression.

Results from Figure 8.3 show the autonomy of the neighborhood enhancement with respect to the maximal (Unity) and minimal (Solo) autonomy enhancements applicable to the proposed solution approach. Since the autonomy introduced to the approach by the neighborhood enhancement is measured and scaled against the minimal and maximal autonomy of any possible enhancement to the approach, the results express the autonomy of the neighborhood enhancement in terms of the autonomy space of all possible approach enhancements. Therefore, the resulting measure is absolute over the domain of possible enhancement.

Absolute enhancement autonomy affords us clear insight into the dependencies an enhancements introduces in achieving the goals of an approach. The afforded insight is clear since all relative autonomy information corresponding to the

approach under consideration in addition to that of any other potential approaches have been discarded. In essence the resulting the autonomy measurements are from a scale which corresponding only to the possible enhancements of an approach.

Returning to our example, given the absolute neighborhood autonomy measure from Figure 8.3 one can immediately see the observations we made in Section 7.3 from Figure 7.1, in particular the near optimal performance of the neighborhood enhancement at low resource factors, the skepticism of agents in collaborating with neighbors at resource factor 2.5, and the inflection point in load factor. The inflection point in load factor for the neighborhood simulation case is not easy to find in Figure 7.1 or even Figure 8.4, but it is well pronounced at a resource factor of 4.25 from Figure 8.3.

The absolute enhancement autonomy trend shows only the changes in autonomy that the enhancement introduces. This allows for the dynamics and in particular the short comings of the enhancement to be identified. One important enhancement shortcoming that the absolute enhancement autonomy exposes is the addition of artificial or costly dependencies in the system. In the example of the neighborhood simulation case, the notion of self-interest implemented in the enhancement is suboptimal since it results in a dip in absolute enhancement autonomy at resource factor 2.5, whereas the same approach at higher and lower resources is able to sustain higher absolute enhancement autonomy. The results suggest that a similar enhancement with a better implementation of the notion of self-interest can produce more consistent results. In short many of the results we found in Section 7.3

could have been found much more readily using absolute enhancement autonomy allowing for the enhancement itself to be gauge more appropriately.

CHAPTER 9 CONCLUSIONS AND FUTURE WORK

We have provided two full MAS solution implementations for the smart grid problem, namely that based on the solo and the neighborhood agents. In order to do so, we first generalized the smart grid problem to an instance of the stabilization problem class. We then devised a methodology for solving instances of the stabilization problem class based on the core defining factor of the class, namely the monotonic increase in cost with respect to fluctuations in a set of properties – see Section 1.2. In particular the proposed solution methodology adjusts the fluctuating properties of any stabilization problem such that the property values approach their average by manipulating a set of given buffering resources which can advance or postpose the fluctuating properties. The proposed solution methodology has significant consequences in that it applies to a large number of seemingly unrelated problems some of which are very high profile MAS problems such as intelligent traffic management systems and stock market trading systems to name a few – see Subsections 9.1.2.1 and 9.1.2.2. Many problems which are considered subjectively in the MAS community are in fact at heart an instance of the stabilization problem.

We adapted the proposed solution methodology into a solution approach for the smart grid problem where we treated the problem as a representative of the stabilization problems. Since the approach is based on the solution methodology for the stabilization class, even though the approach is customized for the smart grid

problem, still by reconfiguring and relabeling parameters it is readily possible to solve many other instances of the stabilization class; for instance any demand driven market problem is likely to be solvable with the same approach as the smart grid. The differences between various custom approaches grounded in the proposed methodology mainly derive from how an agent can predict the fluctuating properties, the rules/infrastructure governing agent collaborations, and the limitations imposed on manipulating buffering systems.

The results of the solo simulation case, corresponding to the base-line smart grid MAS having no agent collaboration abilities, showed that not only does the methodology allow a maximum demand stability to be achieved but also does so with a linear relation to resource capacity which suggests that the methodology can equally utilize resources in order to stability objectives – see Section 7.1 and Figure 7.3.

The results of the neighborhood simulation case, corresponding to the smart grid MAS having ad-hoc coalition abilities in terms of neighborhood collaboration, showed sustained performance improvements over the solo simulation case. In particular the performance improvements for small resource capacities where near maximal with respect to what the approach allows – see Subsection 7.3.2 and Figure 7.1. As such, the results support the theoretical ability for the solution methodology to completely stabilize properties – see Chapter 3.

We investigated the time to equilibrium of both our smart grid MAS solutions with respect to the maximal value the approach allowed given the imposed restrictions. The results showed that the base-line smart grid MAS having no agent

collaboration have a nearly constant time to equilibrium which made the system less effective under highly dynamic conditions. On the other hand the smart grid MAS solution with ad-hoc collaboration did show improvements in time to equilibrium up to a point where with enough resources time to equilibrium was negligible. These improvements did have inconsistent confidence levels suggesting that for some resource capacities the reliability of time to equilibrium of the neighborhood simulation case is substantially lower than other resource capacities levels and even that of the solo simulation case. These results suggested for large enough resources the neighborhood is much more adaptive and versatile against highly dynamic systems than the solo simulation case. However, the neighborhood simulation case was far from the optimal achievable by the approach suggesting the effects of noisy local observations versus aggregate observations.

The common theme over the course of our investigation was the effects of autonomy on the solution approach. By generalizing our investigation process we defined two autonomy measures which capture the autonomy introduced by enhancement to an approach; namely, relative enhancement autonomy and absolute enhancement autonomy. Using these measures we were able to readily show the neighborhood enhancement dynamics exposed over Chapter 7 particularly those in Subsections 7.3 and 7.4. The absolute enhancement autonomy measure clearly exposed the main shortcoming of the neighborhood simulation; namely, the inappropriately defined notion of self-interest indeed produced outcomes which were not actually in the self-interest of the neighborhood agents. One of the main values

that the autonomy measures provide, is the ability to find anomalies in performance dependencies of an enhancement. Finding such anomalies allow for the enhancement to be improved much more readily.

MAS affords us approximate solutions to problems which are otherwise intractable. Autonomy is one, if not the key, mechanism of MASs in which difficult centralized problems with optimal solutions are transformed into a simpler distributed problem with an approximate solution. Intuitively, the range between full autonomy and full unification form a continuous scale such that as a solution tends from autonomy to unification a low computational complexity MAS approximation tends to an optimal yet high computational complexity solution. The ability to measure the autonomy of enhancements to a base-line approach allows for the approximation and complexity of an MAS to be fine-tuned allowing the best mix of reliability, accuracy, and computational complexity to be pinpointed by an enhancement.

9.1 Future Work

The research presented in this thesis covers three main areas:

1. Smart Grid MAS Solutions
2. Stabilization Problem Solutions
3. Enhancement Autonomy

We divide the future work into these areas.

9.1.1 Smart Grid MAS Solutions

Our investigations into the smart grid cover only the basics. The following is a non-exhaustive list of details which we have not covered in this thesis yet which are essential to understanding smart grid MAS solutions:

1. Dynamically changing pricing function such as auctions and market depth
2. Investigations into system reliability and resilience to failure
3. The application of learning and reliable coalition
4. Investigations into the dynamics of agent load suspension tolerance

Although the simulation infrastructure used in this thesis and variants of the infrastructure allow the simulation of stochastically modeled failures, the simulation of auctioneers, buyers and sellers, and the ability to postpone demand by relying on agent tolerance to load suspension, neither this thesis nor our separate investigations have significantly covered the items listed above. In particular, investigations into various applications of learning and reliable collaboration to smart grid MAS solutions are a very promising area of research. A major short coming of our research is the absence of a long term reliable collaboration mechanism to contrast against the short term ad-hoc collaboration of the neighborhood. One of the collaborative strategies which captured our interest is based on the idea of agents leasing their excess resources to one another. Such a strategy could potentially reduce the issues and shortcomings that naturally arise from the complicated definition of self-interest necessary in short lived ad-hoc collaborations – see Subsections 7.3.3 and 7.3.4. Leasing excess resources is essentially a contractual guarantee of collaboration which

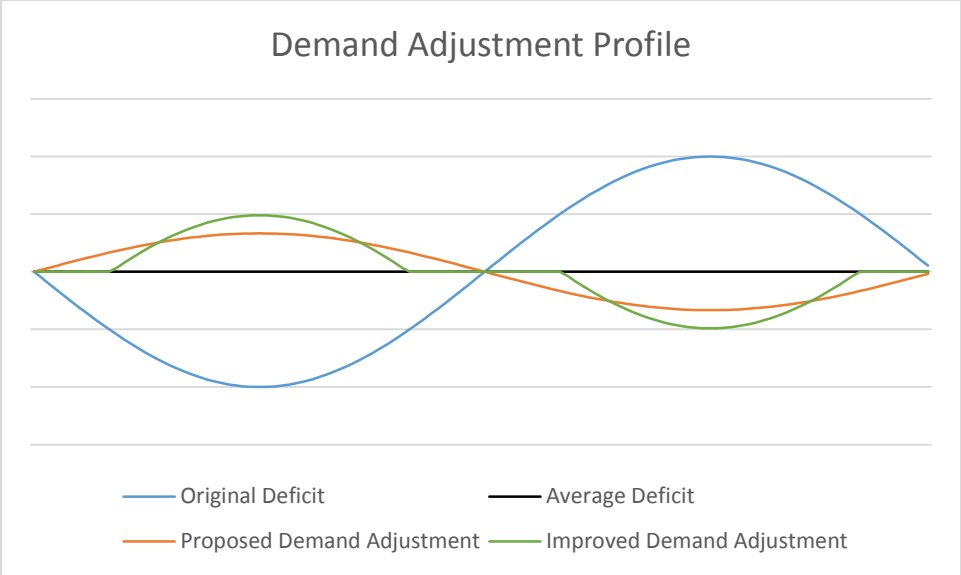
simplifies the problem of defining self-interest due to the reliable nature of such collaboration. More importantly perhaps, leasing excess resources has the potential of addressing the information deficiency faced by neighborhood agents when trying to prepare resources – see Subsection 7.3.2. By leasing, agents exchange the control over excess resources where as in the neighborhood model agents exchange partial deficit profile information. As such leasing allows the control and full information to be places in the hands of resource deprived agents whereas neighborhood collaboration only allows partial deficit profile information and control to be available to agents who own excess resources. Investigating the autonomy and performance dynamics of reliable collaborations based on leasing in comparison to ad-hoc neighbor collaborations not only promises an effective MAS-based smart grid solutions but also offers deeper insight into solutions to the stabilization problems class in general.

9.1.2 Stabilization Problem Solutions

We are aware of several limitations to the proposed methodology from Chapter 3 which each can be addressed in future investigations. One of the main shortcomings of the approach is the lack of consideration of the effects of the pricing function in the demand adjustment profile. In ideal conditions where predictions about the future are at least consistent as time passes if not accurate, the methodology suggests producing a demand adjustment profile which is a constant fraction of the negated predicted deficit profile – see Section 3.3 particularly Eq. (3.8). As an example let us consider the scenario where for some problem, the cost function

associated with fluctuation is such that bounding the range of a demand profiles optimizes profit. Let us consider the problem of optimally adjusting a sinusoidal deficit profile with 0 average deficit given the aforementioned cost function. Figure 9.1 top, illustrates a more appropriate adjustment profile (green) than the adjustment profile proposed in this thesis (orange). Figure 9.1 bottom, compares the demand profiles resulting from applying the more appropriate demand adjustment (green) and demand adjustment proposed by this thesis (orange). The range of the more appropriate demand adjustment profile is substantially lower than that of the demand adjustment proposed by this thesis. Consequently, given the particular cost function considered in this example the more appropriate demand adjustment will result in higher profits compared with the adjustment proposed in this thesis.

As suggested by the results in Section 7.1, the methodology can produce optimal stabilization for all instances of the stabilization problem class; however, given an arbitrary cost function the methodology is likely to have sub optimal reductions in costs. As such, much work remains in creating a cost optimal generalized solutions to the stabilization problem class.



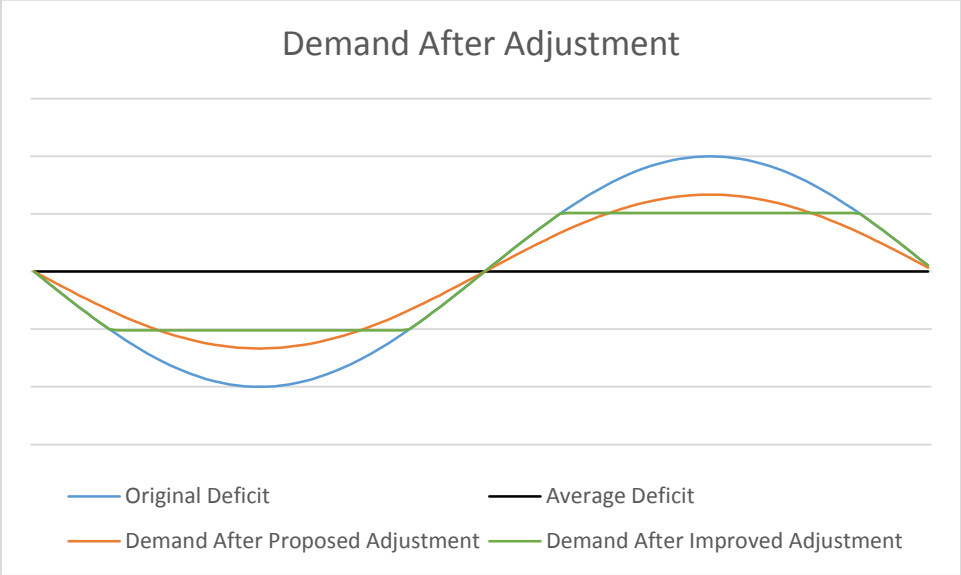


Figure 9.1: The top graphs illustrates what the proposed demand adjustment profile would be for the corresponding original deficit profile in comparison with a more appropriate demand adjustment profile which considers a particular cost function that dictates: bounding minimum and maximum of the original demand is most profitable. The bottom graph shows the resulting demand profiles after the adjustments are made.

More important remaining investigations may perhaps be simply further understanding the scope of the stabilization problem class itself. For instance it is not unlikely that some MAS-based stock market related problems, or even MAS-based

intelligent traffic management problems are indeed instances of the stabilization problem class if not at least their equivalent approximation problems are instances of the stabilization class. Providing a solution approach to each of these problems are promising investigations in their own right. The following subsections provides evidence as to why the problem of stock trading or intelligent traffic systems are likely instances of the stabilization problem class despite seeming unrelated on the surface.

9.1.2.1 The Stock Trading Problem

In the case of the stock trading problem, an agent is motivated to maximize profit by strategically buying and selling stocks. We observe profiting in stock trades is made possible by fluctuations in stock price over time, yet at the same time for a trader to make optimal benefits they must buy and sell at local minimum and maximum prices respectively. In doing so such optimal trades have the effect of increasing and reducing the minimum and maximum price respectively. In the same line, if an agent over sells or over buys, it will cause a local minimum or local maximum in prices respectively which another agent could use to its advantage to profit at the expense of the later. Put simply, an agent profits by bounding the local minimum and local maximum in price, or in other words by stabilizing the price, where as an agent opens itself to the risk of losing capital by introducing instability in the price. As such, agents are interested in reducing any price fluctuations caused by changes in market demand. Finally the available funds and stocks of a trader each act as buffering resources which can be used to conduct transaction which in turn lead to price

adjustments and potentially price stabilization. As such, the problem of optimally trading stocks appears to have all the key properties of a stabilization problem introduced in Section 1.2.

9.1.2.2 The Intelligent Traffic Management Problem

Let us consider the problem of equipping land vehicles with an agent device which given a source and destination can without user interference drives the vehicle safely to the destination. It is likely infeasible to reliably control such an agent in a centralized manner due mostly to wireless communication limitations during bad weather, in tunnels, between tall buildings, and during traffic congestions (where bandwidth limitations may cause problems).

Let us assume that a path between two points contains only one lane but can have many points where other paths merge. Such an assumption is not far from realistic since a multi-lane freeway can be simulated by a path for each lane and in the case of adjacent lanes enough merging points between the two paths could simulate adjacency and ability to swap lanes with any desired accuracy.

As a vehicle merges into a new path it will slow down and subsequently accelerate causing the traffic behind the vehicle in the previous and new path to slow down in order to avoid collisions. The agents are interested in minimizing the amount of fuel costs and wear on the vehicle while giving a smooth and safe ride to the passengers. The agent can accomplish this by minimizing the amount and number of start and stops. In other words, the vehicle saves costs and gains value by reducing unnecessary fluctuations in speed – for safety reasons when merging to a new path

speed must fluctuate and as such reducing this fluctuation is not of interest; however, reducing other forms of speed fluctuation reduces risk of collisions. Finally a vehicle has some space in front and behind itself between other vehicles. An agent can postpone reducing its speed by cruising the distance it has in front of itself until the next vehicle or it can advance reducing speed by using the space it has behind itself until the previous vehicle. An agent can advance and postpone acceleration in the same manner using the distance before and after itself. As such, an agent has buffering resources in which is can use to stabilize its speed in a dynamic environment. Finally we observe the problem just posed satisfies all the key properties of the stabilization problem described in Section 1.2.

REFERENCES

[1] J. W. Forrester, Industrial Dynamics, MIT Press, 1961.

[2] T. Moyaux, B. Chaib-draa and S. D'Amours, "Multi-Agent coordination based on tokens: reduction of the bullwhip effect in a forest supply chain," *AAMAS*, pp. 670-677, 2003.

[3] V. Jaglan, V. Dhankhar, S.Srinivasan and M. Kumar, "A Multi-Agent Based System For Reduction Of Bullwhip Effect In Supply Chain Management," *Asian Journal Of Computer Science And Information Technology*, vol. 2, no. 4, pp. 82-88, 2012.

[4] S. O. Kimbrough, D. Wu and F. Zhong, "Computers play the beer game: can artificial agents manage supply chains?," *Decision Support Systems*, pp. 323 - 333, 2002.

[5] W.-Y. Liang and C.-C. Huang, "Agent-based demand forecast in multi-echelon supply chain," *Decision Support Systems*, pp. 390 - 407, 2006.

[6] S. Yung and C. Yang, "A new approach to solve supply chain management problem by integrating multi-agent technology and constraint network," in *HICSS*, 1999.

[7] M. F. Zarandi, M. Pourakbar and I. Turksen, "A Fuzzy agent-based model for reduction of bullwhip effect in supply chain systems," *Expert Systems with Applications*, pp. 1680 - 1691, 2008.

[8] U.S. Energy Information Administration, "Negative prices in wholesale electricity markets indicate supply inflexibilities," http://www.eia.gov/, 23 2 2013. [Online]. Available: http://www.eia.gov/todayinenergy/detail.cfm?id=5110#. [Accessed 14 7 2014].

[9] N. Bowden and J. E. Payne, "Short term forecasting of electricity prices for MISO hubs: Evidence from ARIMA-EGARCH models," *Energy Economics: Technological Change and the Environment*, vol. 30, no. 6, p. 3186–3197, 2008.

[10] M. S. Fox, M. Barbuceanu and R. Teigen, "Agent-Oriented Supply-Chain Management," *Information-Based Manufacturing*, pp. 81-104, 2001.

[11] S. Ghosn, P. Ranganathan, S. Salem, J. Tang, D. Loegering and K. Nygard, "Agent-Oriented Designs for a Self Healing Smart Grid," 2010.

[12] M. Pipattanasomporn, H. Feroze and S. Rahman, "Multi-agent systems in a distributed smart grid: Design and implementation," Seattle, 2009.

[13] S. D. Ramchurn, P. Vytelingum, A. Rogers and N. Jennings, "Agent-based control for decentralised demand side management in the smart grid," 2011.

[14] R. Roche, B. Blunier, A. Miraoui, V. Hilaire and A. Koukam, "Multi-agent systems for grid energy management: A short review," 2010.

[15] S. Kahrobaee, R. Rajabzadeh, L.-K. Soh and S. Asgarpoor, "A Multiagent Modeling and Investigation of Smart Homes With Power Generation, Storage, and Trading Features," *Smart Grid, IEEE Transactions on,* vol. 4, no. 2, pp. 659-668, 2013.

[16] M. Deindl, C. Block, R. Vahidov and D. Neumann, "Load shifting agents for automated demand side management in micro energy grids," in *IIEEE Intl. Conf. on Self-Adaptive and Self-Organizing Systems*, Venice, 2008.

[17] M. M. V. Prashant P. Reddy, "Factored Models for Multiscale Decision-Making in Smart Grid Customers," in *AAAI Conference on Artificial Intelligence*, Toronto, 2012.

[18] P. Vytelingum, T. D. Voice, S. D. Ramchurn, A. Rogers and N. R. Jennings, "Agent-based Micro-Storage Management for the Smart Grid," Toronto, 2010.

[19] A.-H. Mohsenian-Rad, V. W. Wong, J. Jatskevich, R. Schober and A. Leon-Garcia, "Autonomous Demand Side Management Based on Game-Theoretic Energy Consumption Scheduling for the Future Smart Grid," *Smart Grid, IEEE Transactions,* vol. 1, no. 3, pp. 320 - 331, 2010.

[20] H. K. Nguyen, K. Hee, J. Song and Z. Han, "Demand side management to reduce Peak-to-Average Ratio using game theory in smart grid," *Computer Communications Workshops (INFOCOM WKSHPS), IEEE Conference on,* pp. 91 - 96, 2012.

[21] Z. M. Fadlullah, M. Q. Duong, N. Kato and I. Stojmenovic, "A Novel Game-based Demand Side Management Scheme for Smart Grid," *Wireless Communications and Networking Conference (WCNC),* pp. 4677 - 4682, 2013.

[22] K. S. Barber and C. E. Martin, "The Motivation for Dynamic Decision-Making Frameworks in Multi-Agent Systems," *Autonomous Agents and Multi-Agent Systems,* vol. 13, no. 3, 2006.

[23] S. Barber, A. Goel and C. E. Martin, "Dynamic adaptive autonomy in multi-agent systems," vol. 2, no. 12.

[24] S. Braynov and H. Hexmoor, "Quantifying Relative Autonomy in Multiagent," *Agent Autonomy,* 2003.

[25] S. A. Mostafa, M. S. Ahmad, M. Annamalai, A. Ahmad and G. S. Basheer, "A Layered Adjustable Autonomy Approach for Dynamic Autonomy Distribution," *Frontiers in Artificial Intelligence and Applications,* vol. 252, 2013.

[26] G. Dorais, R. P. Bonasso, D. Kortenkamp, B. Pell and D. Schreckenghost, "Adjustable autonomy for human-centered autonomous systems," 1999.

[27] J. M. Bradshaw, M. Sierhuis, A. Acquisti, P. Feltovich, R. Hoffman, R. Jeffers, D. Prescott, N. Suri, A. Uszok and R. V. Hoof, "Adjustable Autonomy and Human-Agent Teamwork in Practice: An Interim Report on Space Applications," *Agent Autonomy* , vol. 7, 2003.

[28] M. Nowostawski and M. Purvis, "The Concept of Autonomy in Distributed Computation and Multi-agent Systems," in *Intelligent Agent Technology, IEEE/WIC/ACM International Conference on*, Fremont, CA, 2007.

[29] J. W. Tweedale, "Using Multi-agent Systems to Pursue Autonomy with Automated Components," in *17th International Conference in Knowledge Based and Intelligent Information and Engineering Systems*, 2013.

[30] C. Tessier, L. Chaudron and H.-J. Müller, "Conflicting Agents: Conflict Management in Multi-Agent Systems," *Multiagent Systems, Artificial Societies, and Simulated Organizations*, vol. 1, no. 16, p. 335, 2001.

[31] J. Liu, X. Jin and K. C. Tsui, Autonomy Oriented Computing: From Problem Solving to Complex Systems Modeling, Boston: Kluwer Academic Publishers, 2005.

[32] S. A. Mostafa, M. S. Ahmad, A. Y. C. Tang, A. Ahmad, M. Annamalai and A. Mustapha, "Agent's Autonomy Adjustment via Situation Awareness," *Studies in Computational Intelligence*, 2014.

[33] S. A. Mostafa, M. S. Ahmad, A. Ahmad, M. Annamalai and A. Mustapha, "A Dynamic Measurement of Agent Autonomy in the Layered Adjustable Autonomy Model," *Recent Developments in Computational Collective Intelligence* , 2014.

[34] S. A. Mostafa, M. S. Ahmad, M. Annamalai and S. S. G. Azhana Ahmad, "A Conceptual Model of Layered Adjustable Autonomy," *Advances in Information Systems and Technologies* , vol. 206, 2013.

[35] S. A. Mostafa, M. S. Ahmad, M. Annamalai and S. S. G. Azhana Ahmad, "A Dynamically Adjustable Autonomic Agent Framework," *Advances in Information Systems and Technologies*, vol. 206, 2013.

[36] *Repast [Online]. Available: http://repast.sourceforge.net/.*

[37] P. R. v. Oel, M. S. Krol and A. Y. Hoekstra, "Application of multi-agent simulation to evaluate the influence of reservoir operation strategies on the distribution of water availability in the semi-arid Jaguaribe basin, Brazil," *Physics and Chemistry of the Earth*, Vols. 47-48, pp. 173-181, 2011.

[38] N. Bowden and J. E. Payne, "Short term forecasting of electricity prices for MISO hubs: Evidence from ARIMA-EGARCH models," *Energy Economics: Technological Change and the Environment*, vol. 30, no. 6, pp. 3186-2197, 2008.