Robust and Cheating-Resilient Power Auctioning on Resource Constrained Smart Micro-grids

DOCTOR OF PHILOSOPHY THESIS

Submitted in fulfilment of the requirements of the PhD Degree in Computer Science at the

UNIVERSITY OF CAPE TOWN

By

Mufudzi Anesu Chapman MARUFU
MRFMUF001

December 2017

Declaration of Authorship

I, Mufudzi Anesu Chapman MARUFU, declare that this Doctor of Philosophy Thesis titled, 'Robust and Cheating-Resilient Power Auctioning on Resource Constrained Smart Micro-grids' and the work presented in it is my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.
- Where co-authorships are involved, my co-authors have agreed that I may include the publication(s).

Signed: Signed by candidate

Date: 21/05/2018

Certificate

I, Prof. Thomas MEYER, as the candidate's administrative supervisor, have approved this thesis for submission.

Signed: _____

Date: _____

""Half the job is in the discovery; the other half is having the courage to present the findings.""

GALILEO

UNIVERSITY OF CAPE TOWN

Abstract

Doctor of Philosophy in Computer Science

Robust and Cheating-Resilient Power Auctioning on Resource Constrained Smart Micro-grids

by Mufudzi Anesu Chapman MARUFU

The principle of Continuous Double Auctioning (CDA) is known to provide an efficient way of matching supply and demand among distributed selfish participants with limited information. However, the literature indicates that the classic CDA algorithms developed for grid-like applications are centralised and insensitive to the processing resources capacity, which poses a hindrance for their application on resource constrained, smart micro-grids (RCSMG). A RCSMG loosely describes a micro-grid with distributed generators and demand controlled by selfish participants with limited information, power storage capacity and low literacy, communicate over an unreliable infrastructure burdened by limited bandwidth and low computational power of devices. In this thesis, we design and evaluate a CDA algorithm for power allocation in a RCSMG. Specifically, we offer the following contributions towards power auctioning on RCSMGs.

First, we extend the original CDA scheme to enable decentralised auctioning. We do this by integrating a token-based, mutual-exclusion (MUTEX) distributive primitive, that ensures the CDA operates at a reasonably efficient time and message complexity of $\mathcal{O}(N)$ and $\mathcal{O}(logN)$ respectively, per critical section invocation (auction market execution). Our CDA algorithm scales better and avoids the single point of failure problem associated with centralised CDAs (which could be used to adversarially provoke a break-down of the grid marketing mechanism). In addition, the decentralised approach in our algorithm can help eliminate privacy and security concerns associated with centralised CDAs.

Second, to handle CDA performance issues due to malfunctioning devices on an unreliable network (such as a lossy network), we extend our proposed CDA scheme to ensure robustness to failure. Using node redundancy, we modify the MUTEX protocol supporting our CDA algorithm to handle fail-stop and some Byzantine type faults of sites. This yields a time complexity of $\mathcal{O}(N)$, where N is number of cluster-head nodes; and message complexity of $\mathcal{O}((logN)+W)$ time, where W is the number of check-pointing messages. These results indicate

that it is possible to add fault tolerance to a decentralised CDA, which guarantees continued participation in the auction while retaining reasonable performance overheads. In addition, we propose a decentralised consumption scheduling scheme that complements the auctioning scheme in guaranteeing successful power allocation within the RCSMG.

Third, since grid participants are self-interested we must consider the issue of power theft that is provoked when participants cheat. We propose threat models centred on cheating attacks aimed at foiling the extended CDA scheme. More specifically, we focus on the *Victim Strategy Downgrade*; *Collusion by Dynamic Strategy Change*, *Profiling with Market Prediction*; and *Strategy Manipulation* cheating attacks, which are carried out by internal adversaries (auction participants). Internal adversaries are participants who want to get more benefits but have no interest in provoking a breakdown of the grid. However, their behaviour is dangerous because it could result in a breakdown of the grid.

Fourth, to mitigate these cheating attacks, we propose an exception handling (EH) scheme, where sentinel agents use allocative efficiency and message overheads to detect and mitigate cheating forms. Sentinel agents are tasked to monitor trading agents to detect cheating and reprimand the misbehaving participant. Overall, message complexity expected in light demand is $\mathcal{O}(nLogN)$. The detection and resolution algorithm is expected to run in linear time complexity $\mathcal{O}(M)$.

Overall, the main aim of our study is achieved by designing a resilient and cheating-free CDA algorithm that is scalable and performs well on resource constrained micro-grids. With the growing popularity of the CDA and its resource allocation applications, specifically to low resourced micro-grids, this thesis highlights further avenues for future research. First, we intend to extend the decentralised CDA algorithm to allow for participants' mobile phones to connect (reconnect) at different shared smart meters. Such mobility should guarantee the desired CDA properties, the reliability and adequate security. Secondly, we seek to develop a simulation of the decentralised CDA based on the formal proofs presented in this thesis. Such a simulation platform can be used for future studies that involve decentralised CDAs. Third, we seek to find an optimal and efficient way in which the decentralised CDA and the scheduling algorithm can be integrated and deployed in a low resourced, smart micro-grid. Such an integration is important for system developers interested in exploiting the benefits of the two schemes while maintaining system efficiency. Forth, we aim to improve on the cheating detection and mitigation mechanism by developing an intrusion tolerance protocol. Such a scheme will allow continued auctioning in the presence of cheating attacks while incurring low performance overheads for applicability in a RCSMG.

Acknowledgements

I would first like to express my sincere thanks and gratitude to my supervisor Dr Anne V.D.M Kayem for her encouragement, insightful advice and discussions, and for always encouraging me to look at the bigger picture. I also appreciate her impressive editorial ability and incisive comments which made our publications and this thesis clearer and concise. During this PhD journey, I will always reflect on her patient supervision which has undoubtedly helped me develop my research skills.

I would also like to take the opportunity to acknowledge the people who have assisted me during the course of my PhD. Professor Stephen D. Wolthusen who has acted as a co-supervisor over the duration of my research, with some stimulating and insightful comments. Our collaboration resulted in a number of co-authored papers and an opportunity to travel, experience and exchange ideas with colleagues at Gjøvik University College at the Norwegian Information Security Laboratory at the Faculty of Computer Science and Media Technology. Professor Thomas Meyer, who has acted as the main supervisor on submission of the thesis, for his help in steering this ship for the better part of my final year.

I would also like to thank my colleagues Pacome L Ambassa and Goitom Weldehawaryat, with whom I co-authored papers, and, Hendrick Strauss, Ronke Sakpere, Ammar Abuelgasim, who, were part of the Information Security group. Furthermore, I would also like to thank the computer science department, the staff members especially Ms Eve Gill for all the travel arrangements in my various research trips. Lastly, I thank my family and friends for the continued support and prayers throughout this journey. This one is for us.

This thesis was done in-part to a larger work under the SANCOOP Project (SMARtPowerNet). The SmartPowerNet project jointly undertaken by Gjøvik University College at the Norwegian Information Security Laboratory at the Faculty of Computer Science and Media Technology and the Information Security Group, at the Department of Computer Science by University of Cape Town, jointly funded by the Norwegian Research Council (Forskningsrådet project no. 237817) and the South African National Research Foundation under the Forskningsradet SANCOOP programme. I also thank the HPI school and UCT postgraduate funding office for the financial support.

Contents

List of Figures

List of Tables

List of Algorithms

"To my mother, THANK YOU!"

Chapter 1

Introduction

> Sometimes a scream is better than a thesis.
>
> RALPH WALDO EMERSON

1.1 Overview

The shift in emphasis from centralised to decentralised systems to meet the increasingly demanding requirements of complex systems has been observed in the past decades. Autonomy and flexibility in such systems also led to the emphasis of the agent-oriented approach as an appropriate computational model. In such systems, decentralised control is guaranteed by interaction of agents capable of local decision-making based on incomplete and imperfect knowledge about the system. Such decentralised control is more challenging when the *computational ecology* is comprised of low computational and communication components. Against this background, this thesis seeks to develop techniques for decentralised control in such computationally constrained setups.

Decentralised control can be achieved by the allocation of scarce resources in a decentralised way [1]. The subject on decentralised resource allocation has long been studied in economics. This thesis is specifically concerned with using economic tools to achieve decentralised power allocation in resource constrained smart micro-grids (RCSMG). A RCSMG loosely describes a micro-grid where communications are handled by an infrastructure characterised by limited bandwidth, low computational power and energy. The constraints we are thus concerned with are the ones imposed by the processing limitations of the underlying communications network. Such constraints can be counterbalanced by designing power allocation algorithms that are able to run efficiently even under limited processing power.

In general, decentralised control can be non-price based or price based. A non-price based approach does not involve price for the resources and includes techniques such as game theory models. A price based approach is market oriented which uses price as the main economic motivator. We will focus on the market based approach in this thesis due to its ability to facilitate resource allocation based on very limited information, its flexibility, its reliance on local decision-making by selfish individuals and ability to adapt to changes. According to Smith [2], the interaction of selfish, profit motivated agents in a free market can result in a close to optimal allocation of resources. Efficient resource allocation is an emergent behaviour from the interaction. Although such market mechanisms exist in a multitude of forms, we are interested in a power marketing mechanism that is based on the principle of auctioning[1]. Auctions can be single sided, or double sided [4]. Our focus is on the double-sided variety which address the decentralised power allocation problem with multiple buyers and sellers.

The most prominent and ideal mechanism is the Continuous Double Auctioning (CDA) that aligns well with our focus and objectives. Thus, we focus on the CDA as the economically inspired mechanism for decentralised control. The CDA, is a good way of encouraging users to compete for power resources during both low and peak periods. For instance, during periods of scarcity, we can fetch attractive prices for power suppliers thereby encouraging grid participants to generate and share power. At the same time, we are also able to handle power surpluses by dropping prices to encourage purchases without having to deal with issues such as power storage. However, due to the distributed management of the power generation resources we must handle elasticity in demand in addition to ensuring algorithmic efficiency even under low processing conditions. Furthermore, robustness to failure and fair resource allocation is an important consideration in maintaining user participation on the grid. By not providing users with firm guarantees of reliable access to power, a breakdown of the grid is likely to occur due to users withdrawing from grid participation. We counter the occurrence of such situations by extending our auctioning algorithm to provide fault tolerance as well as robustness to cheating attacks towards power theft.

1.2 Context and Motivation

Rural and/or remote areas in developing world countries are sometimes completely disconnected, or intermittently connected to the national grid. The result is that these regions suffer from power shortages that are economically detrimental. In recent years, individually owned generators have become a popular approach for addressing this problem [5]. However, this solution is individualistic and is not environmentally friendly. A reasonable and cost effective

[1] A mechanism that establishes prices based on participant's offers to buy or sell resources [3].

approach to addressing this issue is to design a distributed cyber-physical-system based on low-cost information technology, to coördinate and manage available power resources distributed over the region [6], [7]. We term such a micro-grid a “resource constrained" micro-grid because all communications are handled over a lossy network composed of low-cost, low-processing, and low-energy devices. The limitations of the network, such as low-bandwidth, intermittent wireless connectivity, low processing power and unreliable battery-life imply that existing algorithms must be remodelled to operate efficiently and reliably under these conditions, to guarantee grid sustainability. In focusing on power auctioning we consider specifically the continuous double auctioning scheme which is popular for power marketing on standard smart micro-grids. We show that standard CDA schemes operate inefficiently on RCSMGs because they consider a centralised architecture which results in a single point of failure and therefore need to be re-adapted to handle bottlenecks, scalability, privacy and concerns associated with centralisation. We offer a breakdown of the drawbacks of applying the standard CDA scheme to RCSMGs in Section 3.3.1.

1.3 Power Auctioning - Problem Statement

Unlike common alternatives like First-Come-First-Served allocation, reservation mechanisms or priority queues market-based resource allocation alleviates the power allocation challenges more naturally. Market-based allocation mechanisms are preferable as they create a competitive environment that mutually balances conflicts of interest between parties (Section 3.2). As such, we use a market-based approach to manage the unbalanced power in a typical resource constrained micro-grid while providing fault tolerance and security. Specifically, we propose a Continuous Double Auctioning (CDA) algorithm as an ideal way to control and minimise the differences between the current energy demand and the actual energy production. The differences between the actual demand and generated power are caused by “elasticity" in both demand and generator capacity. Users have partially predictable electricity demand which must be supported by uncontrolled, partially predictable, and constrained power generators. In addition, energy cannot be stored efficiently, which implies that supply and demand of power has to be balanced at all times thereby adding complexity to power trading operations. Thus, there is a need of a power allocation scheme that addresses the allocation problem. The classic CDA algorithms have desirable properties that can help address this challenge significantly. These include:

- simple and fairly robust, achieving high market efficiency in a wide range of market conditions;
- local decision-making by users who have incomplete and imperfect information [2];

- continuous matching which makes it flexible and fulfils the requirement of immediate allocation [4], [8], [9];
- fairness of resource allocation.

Given such desirable properties, our research needs to contribute to the efficient power allocation within a low resourced platform, to the robustness and the security aspects of the CDA. We will now discuss the research aims of this thesis that deal with a number of the aforementioned issues.

1.3.1 Problem #1: Decentralised Efficient Allocation

The application of auction market mechanisms [10], specifically CDA algorithms [11]–[13] has been at the heart of major road-maps for resource allocation in micro-grid structures. However, existing CDA algorithms are mostly centralised and assume that computational resources and bandwidth are abundant (no resource constraints). This is not realistic on lossy networks where low cost, low bandwidth, and processing poor devices are used to support the cyber-system of a micro-grid. Thus, the first research aim of our thesis is **to modify the structure of the CDA so that it can solve power allocation in a RCSMG architecture in a decentralised way**. The improvement should be scalable, maintain the desired properties associated with CDA algorithms (such as high market efficiency), and minimise the communication and computational overheads.

1.3.2 Problem #2: Robustness to Failure

In the considered constrained grid platform, problems ranging from signal loss and distortion to component failures emerge [7], [14] which usually results in system failure. System failures and inadequate fault tolerance result in poor performance, system shut-downs, and security vulnerabilities [15]; therefore compromising grid stability and disrupting user participation (since users buy into the concept of resource sharing with the expectation of a reliable process). For example, regular node/component failures may require frequent restarts of the power allocation algorithm, which consumes a significant amount of system resources and culminates in poor system performance. In such cases, inadequate fault tolerance can create an avenue for weak attacks such as Denial of Service (DoS) attacks which capitalise on node failures. A system is fault tolerant if it is capable of providing correct service despite the occurrence of one or more faults, thereby masking the presence of faults [15]. The second research aim is **to integrate some fault tolerance into the power allocation scheme to ensure uninterrupted power allocation in the presence of faults**. We believe that adding some level of fault tolerance improves the CDA algorithms robustness and therefore participants buy in, which is desirable in maintaining

grid stability.

However, in the event that fault tolerance has failed to guarantee continued market access for traders, participants may fail to secure power during the trading period. This means that the variations in generation and demand that make demand management (balancing demand and supply) a challenge will still exist. Matching supply and demand is vital to grid stability [16]–[18]. It is understood that power consumption scheduling (demand management) can smoothen the demand profiles out over time to avoid overloading during peak times, thus, encouraging grid participants to shift their heavy consumption to off-peak periods. Similar to auctioning, a myriad of centralised power scheduling approaches [19], [20] may not be ideal in the RCSMGs context. The third research aim is **to design a decentralised power consumption scheduling scheme that complements the power auctioning algorithm within the RCSMG**. This is important as it minimises the total power consumption while maximising on the social benefit [18] of power distribution on the grid.

1.3.3 Problem #3: Detecting and Mitigating Cheating

Given the economic significance of the CDA in resource allocation, it seems reasonable to expect a well-established body of research towards security. Surprisingly, this is not the case. To the best of our knowledge, the number of scientific studies towards securing distributed forms of CDA is staggeringly small [21]–[24]. Given the background that cheating is prevalent in auctions [25], participants can engage in undesirable and fraudulent behaviour in trying to gain an unfair advantage, which can disrupt grid stability. Furthermore, it is understood that not only is cheating auction mechanisms specific, but so are the cheating attacks and plausible mitigation measures [21]. This means existing classic forms of cheating are not as useful in equipping the system developers with tools to secure the CDA algorithm. In addition, system defenders have limited understanding of the types of attacks there should defend against. There is no systematic framework available for designing such attacks to allow for an in-depth study. The fourth research aim of this thesis is **to develop a framework that allows for the design and study of cheating attacks on a decentralised CDA**. This is motivated by a need to better understand cheating attacks that can manifest in a decentralised CDA. Studying cheating attacks is important from the power theft perspective, because power theft can cause a breakdown of the grid which is undesirable.

Cheating within CDAs can be resolved in one of three ways: adding cryptography scheme [22], [24]; modifying the auction protocol [26]; or adding a distinct proactive, detection and mitigation protocol [21], [27]. It is prudent to study plausible mitigation solutions on these CDA schemes. Cryptography and some standard security solutions are not ideal in mitigating

the automated cheating forms, due to their computational demand and inapplicability [21]. Modifying the auction protocol may change the core fundamental principles (which alters the desirable properties of an auction), emphasised when building the auction mechanism. Thus, the fifth research aim of this thesis is **to develop a mitigation scheme that addresses automated cheating attacks designed for decentralised CDAs**. The mitigation solution should prove the generality and utility of the scheme in a given resource constrained smart micro-grid environment.

1.4 Contributions

The thesis is organised to separate the three avenues of research it contributes to. Given the research aims outlined in 1.3, we now highlight the specific contributions to the state of the art made by this thesis:

Contribution #1: State-of-the Art

In Chapter 2 we present the background and context of our research work. Our work is built on an amorphous distributed model proposed by Kayem *et al.* [7] which captures a remote community-based setup, where community members agree to coöperate to coördinate grid activities with the main aim of guaranteeing a reliable, efficient and fair access to power. We extend Kayem *et al.*'s model by specifying assumptions and parameters governing our contributions. Further models and concepts valuable to our study are described.

In Chapter 3 we review the literature on CDA algorithms developed for grid-like platforms. Special attention is taken towards identifying the CDA algorithms developed for allocating power within grid-like platforms or constrained environments. Thus, we discuss the strengths and weaknesses of each algorithm; the overheads incurred; reliability; and security considerations made for each algorithm.

Contribution #2: Decentralised Efficient Allocation

In Chapter 4, we propose a token-based CDA algorithm which maintains the desirable standard CDA properties (such as high market efficiency, low communication overheads, fairness of profit distribution among traders and robustness), while efficiently matching loads and generator capacity. We show theoretically that our CDA algorithm satisfies the mutual exclusion properties, while yielding an acceptable time and message complexity of $\mathcal{O}(N)$ and $\mathcal{O}(logN)$

respectively. Our decentralised CDA algorithm scales better to similar CDA algorithms designed for grid-like platforms. Our algorithm should generally be compatible to micro-grids supported by a hierarchical network topology where households form cluster nodes around a single shared smart meter-cluster head (a setup similar to the one discussed Section 2.2. Some results in this chapter are based on the following publication:

- **Anesu M.C. Marufu**, Anne V.D.M. Kayem, and Stephen D. Wolthusen (2015) "A Distributed Continuous Double Auction Framework for Resource Constrained Microgrids", In Proceedings, 10th International Conference on Critical Information Infrastructures Security (CRITIS 2015), October 5-7, 2015, Berlin, Germany; Vol. 9578, Lecture Notes in Computer Science, Springer-Verlag, pp. 183-196 (DOI: 10.1007/978-3-319-33331-1 15) [28]

Contribution #3: Robustness to Failure

In Chapter 5, we integrate fault tolerance capabilities to our CDA algorithm. Fault tolerance allows the CDA algorithm to run with some measure of reliably which significantly contributes to the overall grid robustness and stability. We mainly focus on handling some crash-fail and Byzantine faults. To handle crash-fail faults we propose a simple effective protocol based on redundancy of the functional shared smart meter nodes. Contrary to fault tolerance approaches proposed to handle failure in a MUTEX oriented protocol in the literature which bypass faulty nodes through a network reconfiguration process, our approach masks crash failures of cluster head nodes through redundancy. Masking failure of the shared smart meter nodes ensures the dependent mobile phone cluster nodes hosting trading agents ($\mathcal{TA}$ s) are not isolated from auctioning. A redundant shared smart meter component acts as a backup which takes over if the primary shared smart meter components fails, allowing for some fault tolerance and a graceful degradation of the network. Extension of crash-fail fault tolerance ability to our CDA algorithm yields upper bounds in: time complexity of *O(N)* (where N is number of cluster nodes); and message complexity of$\mathcal{O}((logN) + W)$ time (W is the number of check-pointing messages; N is the number messages exchanged per critical section execution). To handle selected few Byzantine faults we extend the CDA algorithm, which incurs no other costs. Results in this chapter are based on the following publication:

- **Anesu M.C. Marufu**, Anne V.D.M. Kayem, and Stephen D. Wolthusen (2016) "Fault-Tolerant Distributed Continuous Double Auctioning on Computationally Constrained Microgrids", In Proceedings of the 2nd International Conference on Information Systems Security and Privacy (ICISSP 2016), Rome, Italy —Feb. 19-21, 2016, SCITEPRESS pp. 448-456 [14]

In Chapter 6, in a collaborative effort, we propose a decentralised power consumption scheduling algorithm that can be used to complement the decentralised CDA in guaranteeing power allocation in a resource constrained smart micro-grid. The scheduling algorithm employs the alternating direction method of multipliers (ADMM) to decompose the scheduling problem into smaller sub problems that are solved in parallel over local computation devices, which yields an optimal solution. The ADMM can be used to model a scheduling solution that handles both semi-decentralised and fully decentralised providing another layer of power management that compliments the decentralised CDA layer. The reliability and security aspects of the scheduling algorithm are beyond the scope of this thesis. This chapter is based on our results from the following application:

- Goitom K.Weldehawaryat, Pacome L. Ambassa, **Anesu M.C. Marufu**, Stephen D. Wolthusen and Anne V.D.M. Kayem (2016), "Secure and Decentralised Power Consumption Scheduling in Constrained Micro-Grids", In Proceedings, 2nd Workshop on the Security of Industrial Control Systems and Cyber-Physical Systems (CyberICPS 2016), September 26-30, 2016, Heraklion, Greece (in press) [16]

In this conjoint work my main role was mainly designing and analysing the decentralised and fully distributed ADMM algorithms; providing the correctness and complexity analysis. Modelling, problem formulation and security aspects of the paper were heavily contributed by the co-authors.

Contribution #4: Detecting and Mitigating Cheating

In Chapter 7, we propose the Automated-Cheating-Attacks *ACA* framework that can be used to model cheating attacks on decentralised Continuous Double Auction (CDA) algorithms. The framework is a process consisting of eight stages, labelled 1 through 8, to derive attacks. We then use the ACA framework to design novel automated cheating attacks for a decentralised CDA. The generalisability of the framework is demonstrated by mapping existing attack models to the models proposed in this chapter. The cheating attacks designed from the *ACA* framework are then classified with respect to their design. A study of a selected few attacks is done to inform the design of a plausible mitigation solution.

We propose, in Chapter 8, a citizen approach based exception handling (EH) mechanism where sentinel agents use allocative efficiency and message overheads to detect and mitigate cheating forms. The automated cheating attacks give rise to exceptions; situations which fall outside the normal operating conditions expected of the $\mathcal{TA}$ s. One way to deal with exceptions is employing exception handling by distinct domain-independent agents. Sentinel agents are

tasked to monitor changes in message overheads and decrease in market efficiency. Colluding $\mathcal{TA}$ s will gain higher surplus with a trade-off in the allocative efficiency. The exception handling mechanism use these parameters to detect and mitigate cheating. The second exception measure takes care of the number of messages passed among $\mathcal{TA}$ s in the auction. This gives a computational measure of efficiency, that is, how many resources the auction consumes in a run. A slight and sudden increase in the number of messages will give off a red flag for possible cheating. Our exception handling (EH) solution yields a new overall complexity of $\mathcal{O}(nlogN)$ in light demand and runs at a linear time complexity of *O(W)* (where W is the number of participating TAs) and a reasonable ease of implementation.

The results in Chapter 7 and 8 are based on our following publications:

- **Anesu M.C. Marufu**, Anne V.D.M. Kayem, and Stephen D. Wolthusen (2016), "Power Auctioning in Resource Constrained Micro-Grids: Cases of Cheating", In Proceedings, 11th International Conference on Critical Information Infrastructures Security (CRITIS 2016), October 10-12, 2016, Paris, France, vol. 10242 of Lecture Notes in Computer Science, pp. 137–149 Springer-Verlag DOI: 10.1007/978-3-319-71368-7_12 [27]
- **Anesu M.C. Marufu**, Anne V. D. M. Kayem, and Stephen D. Wolthusen (2016), "Circumventing Cheating on Power Auctioning in Resource Constrained Micro-Grids ". In Proceedings, IEEE 14th International Conference on Smart City (HPCC/SmartCity/DSS), Sydney, NSW, 2016, pp. 1380–1387 doi:10.1109/HPCC-SmartCity-DSS.2016.0195 (IEEE Press [21]
- **Anesu M.C. Marufu**, Anne V. D. M. Kayem, and Stephen D. Wolthusen (2018), Secure Power Marketing in Resource Constrained Smart Micro-Grids: Adversarial Cases and Counter Measures, Book Chapter (reviewed and accepted), Submission to: Smart Micro-Grid Systems Security and Privacy, Springer.

Finally, in Chapter 9, we summarise the contributions of this work and end by highlighting the new areas of interest for future work.

1.5 Future work

With the growing popularity of the CDA and its resource allocation applications, specifically to low resourced micro-grids, this thesis highlights further avenues for future research. Additional work can be made towards the algorithms efficiency, reliability and security aspects. First, we intend to introduce mobility of the agent-hosting mobile phones, by allowing connectivity at other neighbouring shared smart meters. Such mobility should not disrupt the desired CDA properties, or greatly expose the CDA algorithm to critical reliability and security issues. In

addition, we foresee more work towards in ensuring security and reliability in the enhanced decentralised CDA, after the introduction of mobility. Such an approach, will allow a progressive evolution of our decentralised CDA. Secondly, we seek to develop a simulation of the decentralised CDA based on the formal proofs presented in this thesis. Such a simulation platform can be used for future studies that involve decentralised CDAs. Third, we seek to find an optimal and efficient way in which the decentralised CDA and the scheduling algorithm can be integrated and deployed in a low resourced smart micro-grid. Such an integration is important for system developers interested in exploiting the benefits of the two schemes while maintaining system efficiency. Forth, we aim to improve on the cheating detection and mitigation mechanism by developing a intrusion tolerance protocol. Such a scheme will allow continued auctioning in the presence of cheating attacks while incurring low performance overheads for applicability in a RCSMG.

Chapter 2

Background

All theories are legitimate, no matter.
What matters is what you do with them.

JORGE LUIS BORGES.

2.1 Overview

Kayem *et al.* [7] provide a RCSMG architecture that incorporates power flow, communication, and control network structures that encapsulates the constraints we are interested in. The conceptual distributed model captures well a remote community-based smart micro-grid, where community members agree to coöperate to coördinate grid activities with the main goal of guaranteeing a reliable, efficient and fair access to power. Such a model, although it is conceptual, it is novel and has allowed for some results to be drawn from questions asked on scheduling [16], power consumption monitoring [29], [30], power auctioning [14], [28], and security in power auctioning [21], [27]. The models and results we outline in this thesis were based mostly on the RCSMG architecture described by Kayem *et al.*. We make additional assumptions and consider more constraints (these are indicated clearly in the different chapters). The outline of the chapter is as follows. In section 2.2, we describe the RCSMG architecture upon which the rest of thesis was modelled on. Specifically, we present the notation, the power network, the communication network and the control/power management layer. In section 2.2.4.1 we focus on continuous double auctioning (CDA), where we give a brief overview of the CDA market institution and useful terminologies. In section 2.3 we describe the RCSMG as a distributed system, outlining the system model, the distributive primitives, and fault models that are considered for later chapters of this thesis.

2.2 Resource Constrained Smart Micro-grid

2.2.1 Notation

We assume an energy sharing agreement exists among the community members P and a grid coördinator GC (utility provider), to provision trading of power. The GC can be an individual or business with the generation capacity to power the entire, micro-grid. The GC can be a participant with a significantly higher proportion of power than the other users in the RCSMG. All participants generating extra power and willing to sell can do so to participants requesting power through the GC who administrates the auction market. The GC can start the auction trade day and can carry out administrative duties. Furthermore, any extra power generated in the RCSMG, can be bought and sold by the GC to other linked RMGs. This is however beyond the scope of the model we consider. Stanczak *et al.* [13], proposed a similar model that considers an auctioning scheme that allows internal traders to buy and sell with external traders.

Participant: P denotes a set of all authorised participants that reside in a dispersed area, who can either be producers (sellers) $P_{\tilde{S}}$ or consumers (buyers) $P_{\tilde{B}}$ depending on whether they are generating and willing to sell, or consuming energy. Hence, $P_{\tilde{S}} \subset P$ and $P_{\tilde{B}} \subset P$. We consider the existence of a symmetric relationship between participants i.e a member can behave as a seller in some instances (if he/she is generating more energy than he/she requires) or as a consumer (buying energy from other members willing to sell). We consider that a user has access to their electric energy consumption as well as generation profile which enables significant user participation (as supported in [31]).

Household: Each element of P owns a single household/ small business (H) within the given community (number of P = number of H) which can have generation and storage capacity (see 2.2.2). Each household has appliances connected to sensors which collect consumption data (see 2.2.2, B). Ambassa *et al.* [29], [30] propose an algorithm to allow secure household consumption data collection. A member of P is responsible for reporting his/her household consumption, and allowing trade (buying or selling) of power for his household. We assume that automation of trade can be assigned to agents ($\mathcal{TA}$) for such tasks (see 2.2.4). Each member of H's power demand and/or generation, reports are first aggregated at the M_{mp}, where a set of U users is authorised to report consumption or interact with the $\mathcal{TA}$. A request to participate by the $\mathcal{TA}$ is transmitted from the mobile device M_{mp} to the smart meter M_{sm}.

Aggregation Device: Each member of H has a single aggregation device M_{mp}. In our case we assume this is a mobile phone. As advised by Kayem *et al.* [7], a single M_{mp} and a set of users U

controlling the M_{mp} is declared for each H . Each member of $M_{mp:j}$ hosts a single trading agent $\mathcal{TA}$ that participates in the auction market autonomously on behalf of a member of P. Each member of H's power demand and/or generation reports are first aggregated at the M_{mp} and they inform $\mathcal{TA}$ s on the bid or ask to offer in the auction market. A request to participate by the TA is transmitted from the mobile devices M_{mp} to the smart meter M_{sm}.

Shared Smart Meter: A shared smart meter is denoted by M_{sm}. Sharing smart meters in economically challenged areas presents a cost-effective solution to reduce costs [7]. Each M_{sm} has a maximum nodal degree of d indicating the maximum number of households that a single M_{sm} can handle efficiently. Each M_{sm} is endowed with protocols that enable distribution of the token integrated order-book to all requesting cluster M_{mp} s.

Household Clusters: Each member of H is grouped in sets of ς clusters. This set is denoted by $\varsigma = c_1, ..., c_j, ..., c_N$ where the subscript j represents the j^{th} cluster and N is the maximum permitted number of clusters. Each member of H belongs to only one cluster c_j that is associated with a single shared smart meter $M_{sm:j}$.

Authorised Users: A mobile phone M_{mp} can only report consumption or generation data of its associated household, and for a period Δt only one user can make reports. All authorisation changes must be explicit. U is a set of authorised users, where $U \subset P$.

Adversary (Attacker): For simplicity and to narrow the scope of our thesis we shall only consider internal attackers (participants who have some motive to cheat in the system). Adversaries, denoted by P_A, describe a subset of P members, that act maliciously to disrupt or cheat in the system. Thus, $P_A \subset P$, while $P_A \neq P$. Without losing generality, this set of adversaries can comprise buyers and/ or sellers, $P_A = \{P_\beta; P\}$

2.2.2 Power Network

We assume a power network of nearby households equipped with renewable generators and/or distributed energy storage and a set of distribution lines that connect household and represents the power line 2.1. Thus, each household $h \in H$ can have a set of electric appliances consuming energy, a distributed generator, and/ or an energy storage component.

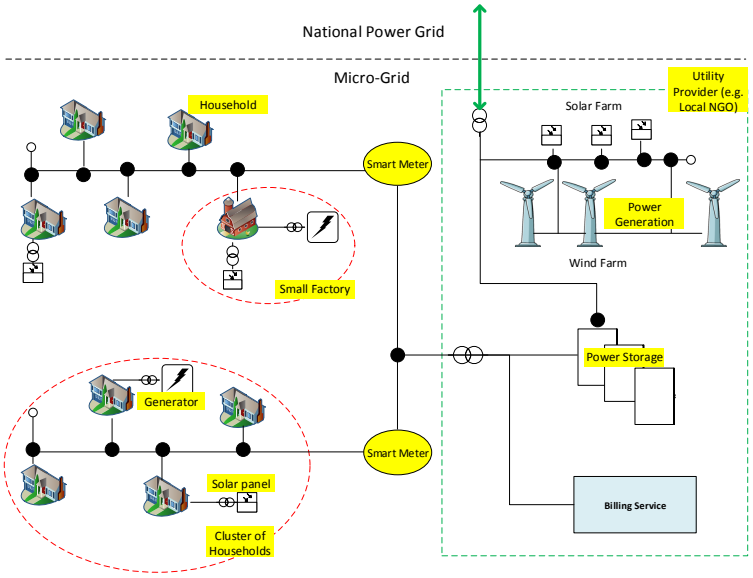

FIGURE 2.1: The power network, adopted from Kayem *et al.* [7, p. 3]

A. Power Generation and Storage

We denote a subset of households with generators to be h_g, the subset of households with small storage as h_s, and the subset of users without generators and/or storage devices as h_r. A household can belong both to h_g and h_s, but users in h_r cannot be in either one of, or both h_g and h_s. If $h_i \in h_r$ then $h_i \notin h_g \cup h_s$.

The utility does the following:

- generate part of the electricity in the RCSMG;
- facilitate efficient exchange of energy in the community through an auction market;
- participate in a short-term wholesale market.

B. Household Appliances

We consider that each household has electrical appliances equipped with a sensor and connected directly to the power source. If an electrical appliance is not sensor-enabled, consumption is reported manually. But, for simplicity, we focus on sensor supported reporting. The set of electric appliances in a particular household denoted $\mathcal{A}_h = \{a_{\{h,1\}}, ..., a_{h,A}\}$ contains four types of loads as proposed by Ambassa *et al.* [30]: resistive load, inductive load, non-linear load and composite load (potentially interruptible), non-interruptible and deferrable. An appliance has a low cost, often inaccurate, sensor which measures consumption of the appliance and transmits it to a local controller *LC* (in our context this can be the M_{mp}). This transmission

occurs via an unreliable communication channel which means the measurements are likely to be untrustworthy. As a result, household demand may not correspond to the exact load requirements. Such data inaccuracies can be mitigated by assuming that the appliances' power consumption is bounded by a minimum and a maximum value representing the lower bound and upper bound. Specifically for the scheduling problem (in Chapter 6) we consider the upper bound value.

2.2.3 Communication Network

We assume a hierarchically clustered network with three, distinct interdependent networks namely: the Home Area Network (Level 1); the Neighbourhood Area Network (Level 2); and the Micro-grid Network (Level 3) with two sets of entities: many mobile phones M_{mp} and relatively fewer, fixed shared smart meters M_{sm}, hence $M_{mp} >> M_{sm}$ (Figure 2.2). We consider M_{sm} are cluster head nodes (Level 3), while different M_{mp} are child nodes (Level 2). The clustering process repeats recursively cluster heads level; M_{sm} may be clustered about neighbourhoods.

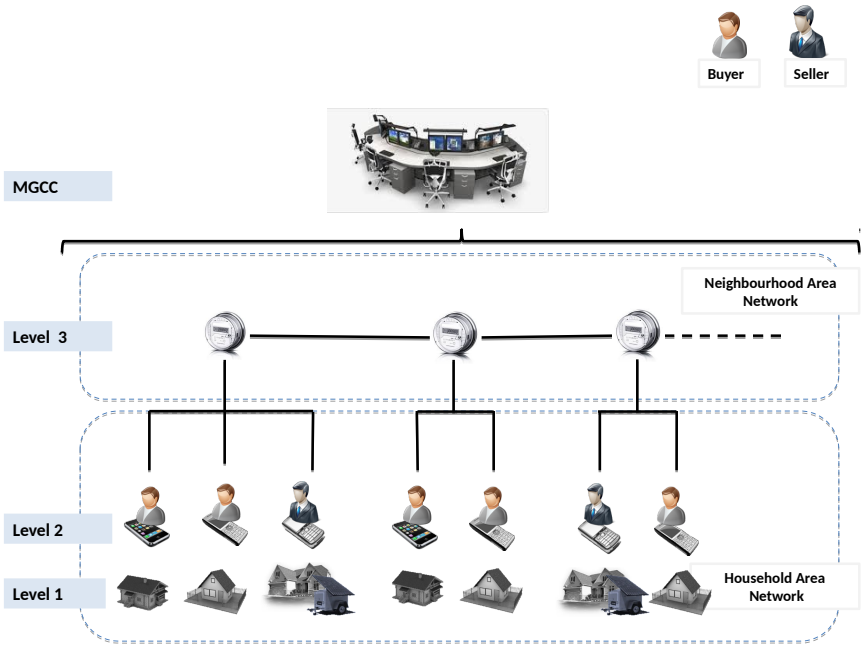

FIGURE 2.2: The communication network

Level 1 describes a HAN of sensors connected to appliances, storage and generator devices. This allows coördination through wireless sensor network protocols such as Bluetooth, ZigBee, and WiFi. Generation and consumption data is communicated through the single M_{mp} within the HAN. Just outside Level 1, we have Level 2 - a NAN consisting of households within the same cluster that share a single M_{sm}.

Level 2 is characterised by frequent and unpredictable topology changes and highly variable message delay, as users move constantly within and outside the household clusters. M_{mp}s are carried by the authorised participants, $\mathcal{U}$, who move independently in any direction. Transmission of messages is variable, since node proximity changes. We consider the M_{mp} nodes can only connect to a single M_{sm} (their proxy cluster head node), to which there are securely authenticated before joining the network. Each mobile phone is subject to frequent disconnections and doze mode functions(optionally disconnecting to save battery life). Further, M_{mp} only connect to M_{sm} nodes in the same cluster when connection to the cluster head was lost during its execution in the market. For simplicity, we consider same cluster M_{mp} form a mesh network with the M_{sm}, supported by wireless communication such as ZigBee and WiFi. We assume, each M_{sm} is capable of supporting $|d|$ maximum number of M_{mp} within a considerable radius (range of 10-100m) to its location.

Level 3 is the MGN, where all information from the other sub-networks (HAN and NAN) is aggregated. Level 3 is where the micro-grid control centre (utility provider) operates from. We assume, communication protocols applicable for this level include WiMAX, Cognitive Radio, together with wireless mesh topology to support data transmission.

2.2.4 Power Management and Control Layer

2.2.4.1 Auction-based Management

To understand how the control network handles trader bidding and double auctioning information to ensure causality in operations between the distributed network components, we first discuss the concepts of continuous double auctioning and automated trading before describing the system model.

A. Continuous Double Auctioning: To better understand continuous double auctioning, we consider the terminology in Smith's [2] seminal work on double auctions. Buyers and sellers are known to be traders who interact to exchange goods. The process permits traders to alter the allocation of commodities, without changing their total quantity. This occurs when traders swap indivisible goods for some divisible money. The amount a trader places on goods is known as the trader's private value or *limit price*. This value is only known by a trader and when a buyer (seller) pays (accepts) more(less) than this limit price, a loss is recorded. A buyer is a trader willing to buy a commodity while a seller is a trader willing to sell their commodity at a given price. A market institution defines how this exchange occurs. In specific terms it defines the rules on what traders can do and the allocation of the commodity with respect to traders actions. Allocation of commodities in response to trader action is known as market clearing. Friedman

[4] defines an auction as a market institution, where messages from traders include some price information which gives priority to higher bids and lower asks. An ask defines the price offer submitted by a seller to sell a unit of goods while a bid is the request offer submitted by a buyer to buy a unit of goods. Sellers and buyers can submit their $asks$ and $bids$ at anytime during a CDA trading day. The current lowest ask in the market, is called the $outstandingask$ (*oa*). The current highest bid in the market, is called the $outstandingbid$ (*ob*). Permitting multiple buyers and sellers to trade is what is loosely called as 'double auctioning'. NYSE spread-improvement requires that a submitted bid or *ask* 'improves' on the *ob* or *oa*. The *no-order queuing rule* specifies that offers are single unit, therefore are not queued in the system, but are erased when a better offer is submitted as bid-spread (the difference between *ob* and *oa*) decreases.

B. Automated Trading Agents: In market-based control, software agents[1] [2] offer truly automated and distributed control systems, rather than relying on a central auctioneer [34] or human intervention. Agent technology can contribute to different aspects of consumer buying —deciding what to buy, whom to buy it from, how much to pay, and the actual trade of goods for money [35]. The level of sophistication of agent automation benefits CDAs. To achieve this, software agents acting on behalf of human users (as delegates) are required to fulfil the user requirements and expectations [1], [36] . Such an agent must show the following properties: autonomy; adaptivity, pro-activeness; reactivity; prediction; social ability; ability to learn; and sometimes mobility [36].

Structure According to Klusch [33], agents can be differentiated between communication, knowledge, collaboration and rather low-level task skills as depicted in Figure 2.3. In the Figure, the corresponding, key enabling technologies are listed below each skill type. In our case, the *Task level skills* would include information retrieval and bidding. The *communication skills* imply the agent accesses information systems and databases or processes input from humans or other agents. An Agent Naming Service (ANS) and an Agent Communication Language (ACL) enable communication between intelligent agents on different levels. Similar to Klush, it is considered that the ACL sits on top of, middleware platforms like OMG's CORBA and Sun's Java RMI, or specific APIs like JDBC (Java Database Connectivity), OKBC (Open Knowledge Base Connectivity) or ODBC (Open Database Connectivity). *Knowledge Skills* entails the techniques that allow the information agent to acquire and maintain knowledge about itself and its user, the network and market environment. *Collaboration skills* of the information agent with other agents relies on negotiation, conversation, whereas collaboration with the human user relies on human computer interaction techniques.

[1] Referred to as a bargaining agents by Priest and Tol [32]
[2] Referred to as information agents by Klusch [33]

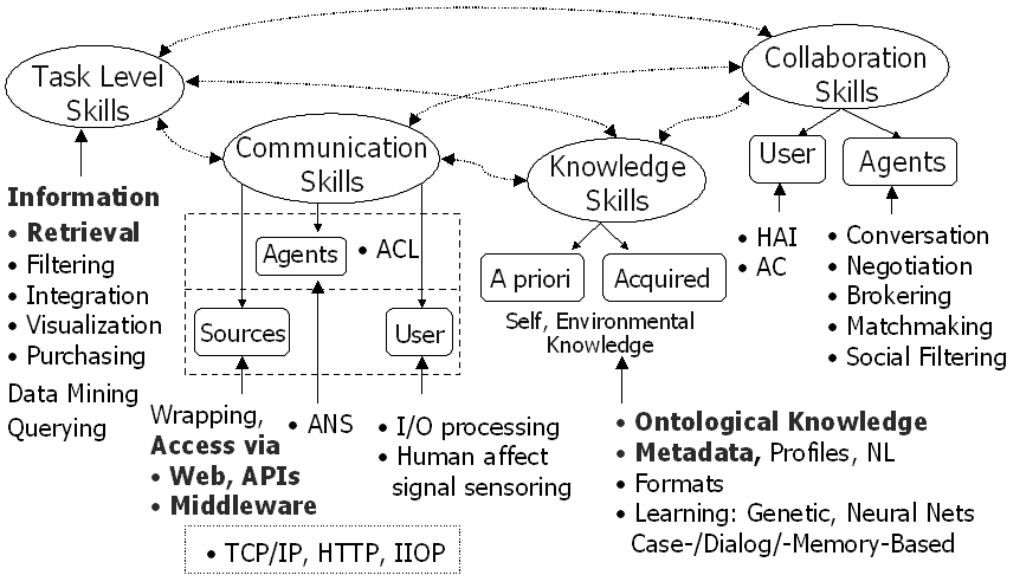

FIGURE 2.3: Automated Agent Structure and Skills,
(Adopted from Guerin and Pitt [7, p. 4])

Agent Communication Framework: Our framework builds on the frameworks presented by Wooldridge [37] and, Guerin and Pitt [38]; we attempt to make the framework more general to allow ACLs with social semantics to be accommodated. We consider the agent communication framework to be a 4-tuple:

$$X_{ACF} =< \upsilon, \varrho, \mathcal{ACL}, \phi, >$$

where:

- υ is the agent's unique version identifier to identify a new agent at the beginning of a trade day or in the event of token regeneration.
- ϱ designates the agent program that is executed at every $\mathcal{TA}$ visited by the token.
- $\mathcal{ACL} =< L_C, L_F, L_T >$ is an ACL including mental L_T and social components L_F.
- ϕ is the initial assertion for social states.

Negotiation Strategy Thus, the agent requires a strategy for negotiation which is at least as effective as a qualified human being in the same situation [32]. Multi-agent based distributed energy resource management has been studied extensively by [39]–[41]. In this thesis, we consider trades in an auction occur when trading agents ($\mathcal{TA}$ s) interact with one another (i.e., they buy and sell goods or services). Each $\mathcal{TA}$ knows information about itself and can collect information made public from auction market. The $\mathcal{TA}$ s employ some heuristics in their 'strategy' to handle incomplete information and the dynamic market environment [36].

The strategy's input includes private information (limit price, eagerness to trade, etc.) and public information (such as outstanding ask/bid, last transaction price, etc.). The strategy's out is the offer to be submitted. Since the Santa Fe Double Auction Tournament (SFDAT) [42] was conducted, several bidding strategies have been developed to determine the most efficient strategy best for the CDA market. These include: Gode and Sunder's [43] Zero-Intelligence (ZI) strategy; Cliff's [44], [45] Zero-Intelligence Plus (ZIP) strategy ; Preist and Tol's [32] CP strategy; Gjerstad and Dickhaut's [46] Gjerstad-Dickhaut (GD) strategy; Tesauro and Das' [47] Modified Gjerstad-Dickhaut (MGD) strategy; Tesauro and Bredin's [48] Extended Gjerstad-Dickhaut (GDX) strategy; He *et al.*'s [49] Fuzzy Logic (FL) strategy; Vytelingum *et al.* [50] Risk-Based (RB) strategy; Cliff's [51] ZIP60 strategy; Vytelingum's [1].

C. Distributed System Model: Consider the RCSMG to be a distributed system consisting of a finite set of $\mathcal{N}$ sites, (i.e. the M_{sm} nodes) $\Upsilon = \{S_1, S_2, ..., S_N\}$, that are spread-out forming a (NAN) network. The nodes communicate only by sending and receiving messages. Message delivery is guaranteed if nodes and links are up. Solving both link and node failure simultaneously can be a challenge, thus, we assume, every node pair is connected by means of a reliable communication channel. Messages can be delivered in a different order to the one they were sent. We consider a semi-synchronous fully-connected* network where process speeds and message transmission times are bounded. The maximum latency for sending a message between two sites is denoted by $\mathcal{T}_{lat}$. A critical section execution takes an average of $\mathcal{T}_{cs}$. We shall consider two fault models (in section 2.3.2) where: (i) nodes can fail by crashing only, and this crash is permanent, or (ii) nodes can fail by displaying Byzantine type of faults. Initially, assume that nodes are fairly reliable (in Chapter 4), then in Chapter 5 we tighten the assumption to consider cases of node and link failure.

2.2.4.2 Demand Response Management

We now discuss how the control network handles power consumption, generation, and state estimation information to ensure causality in operations between the distributed network components. The control network facilitates demand response automation of the micro-grid. As proposed by Kayem *et al*, we model this as an acyclic graph $G = (M_{sm}, E)$, where M_{sm} is a finite set of shared smart meters and E is a finite set of communication paths between the M_{sm}. Each household can be represented on the tree schema by M_{mp}. Figure 3 illustrates the schema for grid connectivity, from the control network perspective.

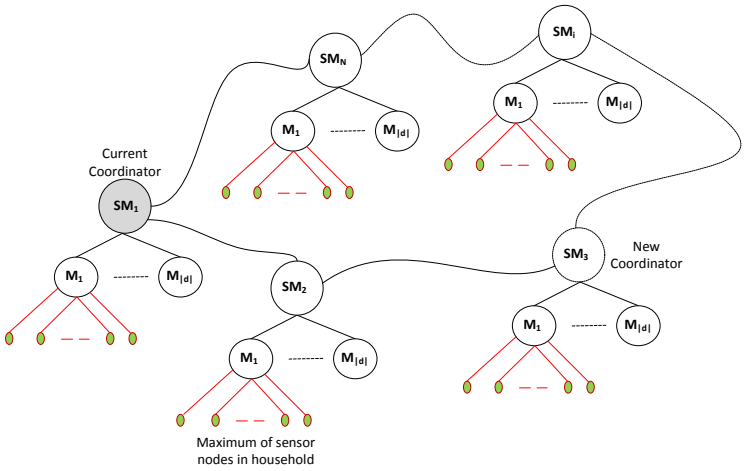

FIGURE 2.4: Smart Micro-grid Schema,
(Adopted from Kayem et al [7, p. 4])

2.3 Distributed Algorithms

A distributed system is a collection of individual computing devices (processors) that can communicate with each other. As mentioned earlier in this chapter, at any single time within a CDA only one TA is allowed to submit an offer into the market. Thus, the challenge of serialisation of concurrent accesses to the auction market, results in a mutual exclusion problem with emphasis on low communication overheads.

2.3.1 Mutual Exclusion Primitives

A Mutual exclusion problem occurs when there is a collection of asynchronous processes, each alternately executing a critical and a non-critical section, that must be synchronised so that no two processes ever execute their critical sections concurrently [52]. Our choice for an appropriate MUTEX algorithm was governed by the following:

- The characteristics of our problem —$\mathcal{TA}$ s need to mutual exclusively submit an offer into the order book (participation in the market)
- The characteristics of the system the algorithm will run on —we consider a constrained hierarchical architecture (see Section 2.2)
- The type of interprocessor communication —we consider the nodes communicate by message passing
- The level of timing synchronisation between separate processes —we consider a loosely asynchronous communication

Many practical solutions crafted to address the mutual exclusion problem rely heavily on a central coördinator that manages access to the critical sections [53]. We disregard centralised solutions using coördinators, and consider a decentralised algorithm, where a subset of the nodes M_{sm} have a slightly added responsibility over M_{mp} nodes. We propose a token-based approach inspired by Raymond's work [54] to address the mutual exclusion problem. Our choice was inspired by low message traffic generally associated with token-based algorithms over other approaches. [53][54] [55] [56].

A token is a *permit* for entry into the auction market, passed around among requesting $\mathcal{TA}$ s. A $\mathcal{TA}$ requiring participation in the auction should acquire the token. Thus, at any moment, only one $\mathcal{TA}$ can exclusively hold the token. We propose an enhancement to this phenomenon, to support the CDA market protocol. We consider that the token contains a copy of the order-book, where the trading agent submits its offer once it enters the auction market.

The token is a mobile agent, that is, a *program code* that migrates from one process to another. Unlike an ordinary message that is passive, the mobile agent is an active entity that can be compared with a messenger. The agent code (the token) is executed at the M_{mp}, uses its resources, allows $\mathcal{TA}$ s to submit a bid and completes the auctioning process. We consider this to be weak mobility. With weak mobility, the control state is not transferred, so at each host M_{mp}, the code executes without this information. The token with weak mobility can be modelled as follows:

Definition 2.1 (Token Descriptor)**.** The token is a 4-tuple

$$X_{Token} =< I, \upsilon, \varrho, \beta, TokenCounter >$$

where:

- I denotes the initiator (grid coördinator) who initiates the auction and sends the token to the M_{mp} nodes requesting it first. At the end of a trade day the token is sent back to I.
- υ is the agent's version identifier and uniquely identifies a new agent at the beginning of a trade day or in the event of token regeneration.
- ϱ designates the agent program that is executed at every $\mathcal{TA}$ visited by the token.
- β denotes a 'briefcase' component encapsulating the token's data variables, such as $Obook$, ob, oa, R.
- $TokenCounter$ variable stores the number of auction market rounds. The variable is auto-incremented every time a $\mathcal{TA}$ receives it. When the predefined number of rounds is reached trading is terminated.

The logical tree-based (token-based) algorithms are very sensitive to node failure. If a node having the token fails or the token is lost in transit, a complex process of token regeneration

and recovery has to be started. Chapter 4 describes in detail the decentralised CDA algorithm that employs the described MUTEX model. To prove correctness we propose that the MUTEX protocol must satisfy the following properties:

- **ME1**: [Mutual exclusion] At most, only one $\mathcal{TA}$ can stay in the auction market (critical section) at anytime. This is a safety property. Safety invariant can be written as $N_{cs} \leq 1$ where $N_{cs}$is the number of processes in the critical section at anytime.
- **ME2**: [Freedom from deadlock] When the auction market is free, but no requesting $\mathcal{TA}$ s can enter the auction market, making further progress impossible. This is a liveness property. Assuming a configuration satisfies the precondition P and is expected to satisfy the postcondition Q at termination. Let GG be the disjunction of all guards of all the processes. Then the desired safety property is expressed by the invariant $Q \vee GG$
- **ME3**: [Freedom from Starvation] A $\mathcal{TA}$ should not be forced to wait indefinitely to execute in the auction market, while other trading agents are repeatedly executing their requests. This is a liveness property.
- **ME4**: [Fairness] requests must be executed in the order in which they are made. Fairness is also a liveness property, as the $\mathcal{TA}$ should execute in a finite time.
- **ME5**: [Termination] Starting from the initial state, every feasible behaviour leads to a system configuration in which all the guards are false and the terminal configuration is reached. While partial correctness ensures that the desired postcondition holds when all guards are false termination ensures whether the terminal state is reachable via all admissible behaviours. This is a liveness property

2.3.2 Classification of Faults

Faults can be classified into one of the three categories according to their persistence, namely transient faults, intermittent faults and permanent faults [53]. Transient faults occur once and then disappear. Intermittent faults can be characterised by a fault continuously occurring and disappearing many times. Permanent faults are persistent and continues to exist until the faulty component is repaired or replaced. Any one of these faults may either be a fail-silent failure (also known as fail-stop) or a Byzantine failure [53]. Jalote [15] classify faults in a distributed system based on how the faulty component behaves when it fails. Such a classification assigns faults to one of the four categories: crash faults, omission faults, timing faults, and Byzantine faults. Crash faults causes the component to completely stop and lose its internal state. Omission faults causes a component not respond to some inputs or fail to send some messages. Timing faults cause a component to respond either too late or early. Byzantine fault causes the component to behave in a totally arbitrary manner during failure.

Our focus is on faults with direct and immediate impact on the CDA algorithms' execution. We are aware that such faults may emanate from the routing protocol, hardware failures, etc.. But, we narrow our scope to faults that disrupt the distributive protocol serialising market access.

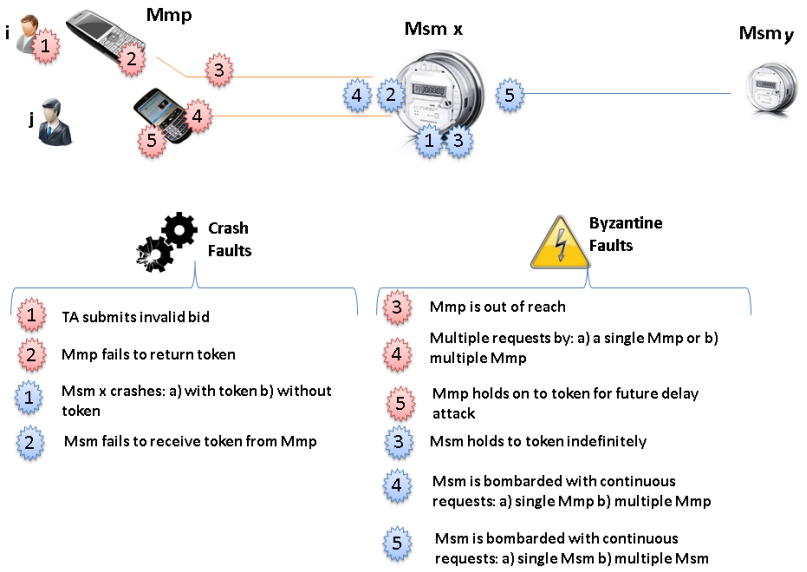

FIGURE 2.5: CDA Fault Scenarios

As an initial step to building a fault-tolerant system, the literature [57] [15] [58] [59] suggests defining the fault model (the number and classes of faults that need to be tolerated). A fault model includes a set of failure scenarios along with frequency, duration and impact of each scenario [57]. Inclusion of fault scenarios in our fault model is based on the frequency, impact on the system and feasibility or the cost of providing protection. We understand there are variations in the literature about definitions of several of the well-known failure classification schemes. We consider van Steen et al.'s [58] classification of faults into: crash failures, omission failures, timing failures, response failures and arbitrary failures.

For simplicity and as a first step towards tolerating failure of some cluster nodes, we consider first crash failure because this set of failures provides a simple abstraction meant for simplifying the design of fault-tolerant algorithms (Section 5.3.1). If a system fails to tolerate crash failures, it cannot tolerate the other classes of failure [53] [58]. We go further and look at more challenging Byzantine faults resulting from nodes behaving in an arbitrary manner (Section 5.3.2). We separate failure scenarios into the following cases:

- a single fault;
- multiple separate single faults;
- multiple concurrent faults

Figure 2.5 shows a visual representation of the fault scenarios we consider affecting the token distribution within the CDA algorithm. The scenarios presented are not exhaustive but they help inform on fault tolerance measures that should be developed, thereby ensuring the CDA algorithm executes successfully. In Section 4.6 our initial CDA algorithm considers token handling measures which mitigates fault scenarios 1-3 under crash faults and scenario 2 under Byzantine faults. An in-depth discussion and mitigation measures of the other fault scenarios (and their respective fault models) is presented in Chapter 5.

Chapter 3

Literature Review

> To raise new questions, new possibilities, to regard old problems from a new angle, requires creative imagination and marks real advance in science.
>
> ALBERT EINSTEIN.

3.1 Overview

Overall, the main objective of power allocation or resource management schemes is to establish a mutual agreement between power generators and consumers while providing a seamless way to allow exchange of power. In this chapter we discuss the state-of-the-art in the area of economic models for resource allocation in grid-like platforms. In Section 3.2 we provide a taxonomy of the resource allocation models in literature, and build a case on why Continuous Double Auction (CDA) algorithms are appropriate for constrained grid-like environments. In Section 3.3 we then review the main closely related CDA algorithms designed for resource allocation in grid-like platforms. Further, we review the main related work on robustness (in Section 3.4) and cheating resolution (in Section 3.5) aspects within decentralised CDAs.

3.2 Economic Models for Resource Allocation

Economic resource management models can be separated into 3 main classes: Conventional models, Non-price based models and Price-based models.

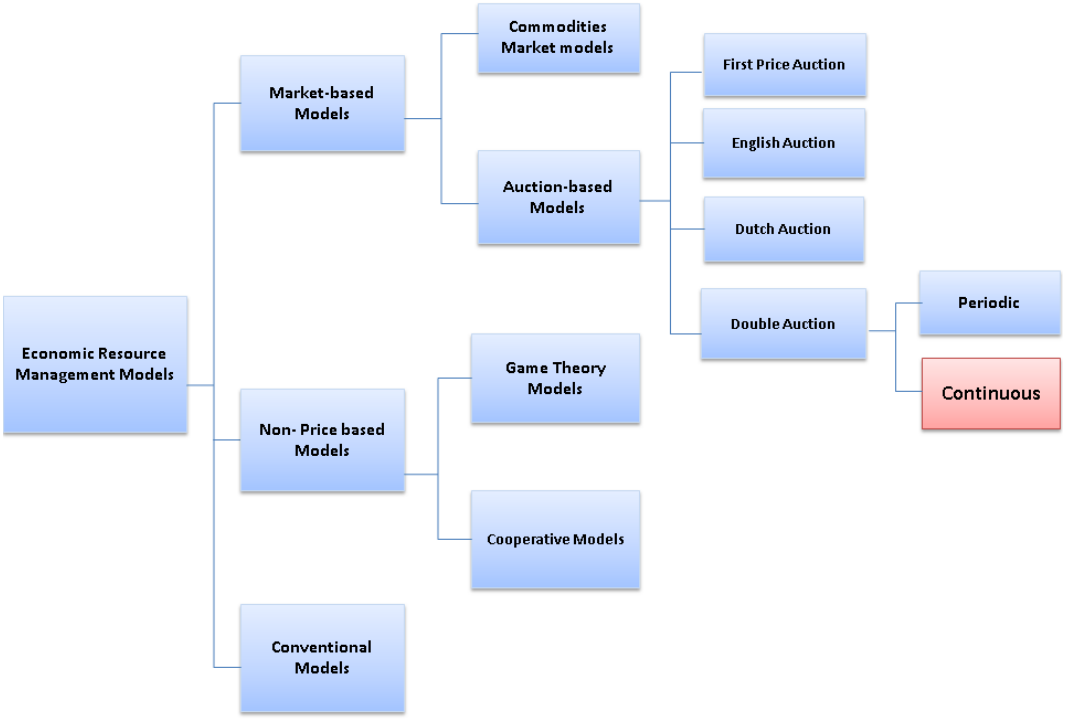

FIGURE 3.1: Taxonomy of Resource Allocation Models

Conventional resource management schemes employ relatively static models with a centralised controller that manages resource generated and usage. Such conventional management strategies can work well where resources are known in advance, but, fail to work in heterogeneous and dynamic systems where jobs need to be executed by computing resources whose availability is difficult to predict [60]. Furthermore, conventional methods are not suitable as they assume complete control over resources and requests; a steady supply of energy from more reliable sources; and finite resources for the energy distribution and control mechanism.

Non-price based resource management schemes employ more dynamic systems but the price is not considered. Pourebrahimi *et al.* [60] categorises non-pricing approaches into 2 subclasses: game-theory based and coöperative mechanisms. The coöperative mechanisms ensure all entities in the distributed system have a global utility function. In these approaches, each participant (usually an agent) is initially endowed with some resources. They exchange them until the marginal rate of substitution of the resources is the same for all the agents and there is no further incentive for coöperation. Kurose and Rahul's [61] use decentralised algorithms to allocate resources (such as files or file fragments) coöperatively. Meanwhile, game-theory based mechanisms are anchored on principles that include selfish optimization and individual utility functions [31].

Market-based approaches (Price-based approaches) use money and pricing as the main technique

for coördination between resource producers and consumers [60]. Market-based allocation mechanisms eliminate the need for a centralised control (which emanates into single point of failure) and suit decentralised nature of the micro-grid, implicitly enhancing robustness [62]. In addition, market-based methods create a competitive environment that mutually balances conflicts of interest between parties; thereby balancing demand and supply. Moreover, pricing signals can be adjusted to achieve secondary objectives. Such allocation provides an environment that facilitates complex combinatorial resource requests, i.e. users can freely acquire energy in a market at their convenient time, provided they pay sufficiently high prices. Market-based approaches are classified into two categories of economic models: Commodity Markets and Auctions.

Commodity market models allow providers to specify their resource price and then charge the consumers according to the amount of resources they consume, while in the auction market each provider and consumer acts independently, and they agree privately on the selling price. However, the commodity market model is not incentive-compatible [4], lacks flexibility [9], since all types of resource have to be predefined. The use of a commodity market model for obtaining global equilibrium prices for resources in a Grid context has already been proposed in [63], [64]. Stuer [65] enhances this approach further and extend it to allow for trading and pricing of substitutable goods. An overview can be found in [66].

When well-designed [67], the auction market models can achieve desired economic outcomes in RCSMG (described in Section 2.2); as they require little global information, support decentralisation, offer high allocative efficiency whilst being easy to implement in grid-like settings [68], [31], [22], [69], [13] [28]. In addition, auctions are used for products that have no standard values and the prices are affected by supply and demand at specific time. Auctions can be classified into four basic types based on the interactions that exist between consumers and providers: the ascending auction (English auction), the descending auction (Dutch auction), the first-price and second-price sealed auction, and the double auction [68]. There is a range of different variants of the double auctions. Parsons [70] reviews some double auctions variations, with particular attention to the differences between them. Parsons also describes some theoretical and experimental work that has been carried out on double auctions.

Among the plenitude of auction formats available, the CDA is the most appropriate market model that can facilitate power allocation. This claim can be supported by Smith's seminal work [2] on competitive market behaviour, which significantly advanced research on CDAs. He demonstrated that the market efficiency achieved in CDAs populated by a relatively small number of selfish human traders, in a decentralised environment, where no single agent has complete and perfect information about the system, was close to one. He further proved that transactional prices converge to the market's theoretical competitive equilibrium price. These results were novel as they showed that markets governed by a decentralised mechanism, such

as the CDA, do not need to be large to be efficient, as had previously been assumed. Many subsequent research endeavours in this area have been heavily influenced by this work

De Assuncao and Buyya [71], study the communication requirements of First-Price sealed, English, Dutch, and CDAs for resource allocation in Grid computing environments. Their results show that the English auction incurs higher communication overheads, while the CDA incurs the least. These findings are valuable especially if we consider the RCSMG platform. Grosu and Das [72], investigate three types of auction allocation protocols: First-Price Auction, Vickrey Auction and Double Auction, with the find the best suited for the grid environment from users' resources' perspective. The results show that when a mix of risk-averse and risk-neutral users is considered, First-Price Auction favours resources while Vickrey Auction favours users. On the other hand the Double Auction favours both users and resources. Similarly, Kant and Grosu [73] proposed a double auction allocation model for grids they evaluated with other double auction protocols for resource allocation: Preston-McAfee Double Auction, Threshold Price Double Auction and CDA. Kant and Grosu showed that the CDA protocol is preferable, from a resource's and user's perspective providing high resource use. The CDA offers continuous matching and clearance, which makes it flexible and fulfils the need for immediate allocation. Resource allocation is an emergent behaviour of the individual trading agents' complex interactions; the transactions correspond to allocations [74].

3.3 Decentralised Efficient Power Allocation

Application of CDAs has been at the heart of major road-maps for micro-grid structures [75], [76], [9] [68], [13]. A myriad of these CDA schemes have deferring constructs architectures and attributes. The two main architectures are the centralised and distributed types. Both architectures have certain properties associated with the communication model necessary to apply them. A decentralisation architecture is more desirable over a centralised architecture as it eliminates bottleneck and scalability issues [77]. In addition, a centralised architecture suffer from a single point of failure which raises robustness/resilience and security flaws. Although centralised approaches show some clear limitations, a completely decentralised approach also raises specific problems. Despite the chosen approach, the main challenge is to make sure they find the resources needed by users. The next subsection presents a review of CDAs designed for resource allocation in grid-like platforms.

3.3.1 CDAs for Resource Allocation

Koutsopoulos and Iosifidis [77], note that decentralised multi-lateral node interactions and the double node role as resource provider and consumer amidst resource constraints cannot be

captured by single-sided auctions, and even more so, by mechanisms that rely on a central controller.

Pourebrahimi *et al.* [60] proposed a CDA algorithm that allocates computational resources (CPU time) with a single centralised market in a local grid. Their CDA showed comparable resource efficiency, in comparison to a non-market based model. The system works by allowing every node that has idle CPU time and a node that has tasks waiting for free CPU, to send (receive) some requests (offers) to (from) the central auctioneer. Pourebrahimi's CDA scheme employs a centralised architecture with three main participants: *Consumers* and *producers* agents and a centralised *Auctioneer*. Similar to our work, *consumers* and *producers* are autonomous agents capable of decision-making governed by capability and local knowledge. The *Auctioneer* is mediating agent that matches the offers using a double auction protocol; implicitly using a periodic clearing mechanism. It is not clear how many and how often messages are exchanged between the Auctioneer, Consumer and Provider agents. The CDA formulation does not seem to consider reliability and security aspects that may hinder auctioning.

Vytelingum's work [1] was aimed towards CDA structure and behaviour. In the structural aspect, Vytelingum looked at how the market protocol of the can be modified to meet desirable properties such as: a high market efficiency, fairness of profit distribution and market stability. Next the modified protocol is used to solve a complex decentralised task allocation problem with limited-capacity suppliers and consumers with inelastic demand. Vytelingum shows that the structure of the CDA (decentralised over centralised) affects the efficiency of the CDA. The main focus of his contribution is towards market efficiency, which is a slight departure from our work which investigates computational efficiency. In the behavioural aspects, Vytelingum developed a multi-layered framework for designing strategies that autonomous agents can use for trading in various types of market mechanisms. The framework is further used to design a novel Adaptive —Aggressiveness AA strategy for the CDA. The AA strategy outperformed the benchmarks in terms of market efficiency in a dynamic market. These contributions are valuable to our work as study of the multi-layered framework and the AA strategy allowed us to discover novel attack paths an attacker could exploit in order to cheat in the CDA. Further, despite the valuable insights Vytelingum's work provides, it does not show how a decentralised CDA algorithm would look like (communication-wise) or how it would perform on low computation resource platform(in message complexity). No insights can be obtained on the robustness and potentially security of the CDA to failure or attacks.

Tan and Gurd [9] proposed a stable continuous double auction (SCDA[1]), based on the more conventional CDA. The SCDA reduces the unnecessary price volatility caused by impatient as well as insensitive behaviour of some market participants by a novel fuzzy logic and heuristic based price adjustment mechanism. This is a slight departure from our work which does

[1]This work is part of Tan's thesis [78]

not focus on providing economic efficiency, but rather, focusses on providing computational efficiency to a CDA. The SCDA entails interactions between three types of software agents namely: provider agent, user agent and auctioneer agent. The auctioneer agent coördinates and manages the trading process by communicating with all its participating user and provider agents. In addition, the auctioneer agent also monitors the system's efficiency and calculates the theoretical market efficiency for evaluation purposes. Evidently the CDA herein adopts a centralised architecture and its associated challenges described earlier in this section. User and provider agents communicate solely with the auctioneer agent; there is no direct communication between them. It is not clear how many messages are passed between the agents, and how often. Instead of a broadcast by the centralised agent, the agents send a query to the auctioneer. Tan's CDA may fail to offer efficient resource allocation in a resource constrained platform since it was built without consideration of the RCSMG constraints.

Izakian *et al.* [68] continuous double auction method for grid resource allocation. Users have one or more independent, computational-intensive jobs for execution and will pay for it. Also, resource owners have computational resources and will rent them for profit. Agents employing decision-making methods (opposed to known strategies) act on behalf of their human owners who are either providers or consumers. Provider agents determine their bid value based on workload while consumer agents determine their bid value based on remaining time for bidding and remaining resources for bidding (inspired by [79]). The communication and computational overheads incurred in this CDA are not clear in the paper. In each time unit, consumer agents and provider agents find their bid and request values and send them to the auctioneer. It is not clear how many messages (offers to the auctioneer, or trade confirmation) are passed between consumer agents, provider agents and the auctioneer. Furthermore, no reliability or security aspects were considered in the formulation of this CDA mechanism.

Teymouri and Rahmani [80] proposed a CDA where users and providers connect with some auctioneers randomly. Each user or provider can connect to several auctioneers to participate in different auctions to increase its chance. Market information is decentralised among a set of auctioneers and each auctioneer keeps the information of local users and providers. The CDA algorithm engages less with the users, as updating of bids is done by the auctioneer itself. Intuitively, although the single point of failure is eliminated by such an approach the other issues remain; scalability may be of concern, security and privacy concerns in case of a rogue/malicious auctioneer(s). Work by Teymouri and Rahmani is still valuable, as the researchers show that their protocol increases the success rate of jobs and providers benefits while decreasing the users' cost.

Majumder *et al.* [81] proposes an incentive-compatible double auction mechanism, for energy trade between buying and selling agents, with the aim of solving the social maximization problem. The authors implicitly assume, the controller is trustworthy, which can be unrealistic

in a scheme that has selfish participants. Further, the auctioneer presents a single point of failure.

Stańczak *et al.* [13] present an example of balancing a micro-grid with its own energy production sources and connected to a higher voltage distribution grid, by introduction of the continuous double auction market with agents using the Adaptive-Aggressive Strategy. Consider a centralised CDA architecture were each trader decides how much energy has to be bought or sold and sends it as an offer to the market with a proposed price per unit. The CDA algorithm continuously collects offers and matches the new ones with the already submitted bids or asks (persistent bids and asks). If the deal balancing for the energy is not accomplished in the current session, a balancing mechanism will check the true usage/production of energy and clears the unsettled offers by exchanging them with the external grid.

In summary, we note that, CDA algorithms designed for grid-like environments, subject to the literature we reviewed, use a centralised approach to guarantee market participation. Each trader submits their bids to a centralised auctioneer, which we ascertain is not ideal for the considered RCSMG. The knowledge gap is in the lack of a CDA deployable in constrained grid-like platforms, with traders coördinated without relying on a centralised auctioneer. We contribute to this knowledge gap by designing such a decentralised CDA algorithm that we believe contributes to the state of the art and is useful within RCSMGs. The motivation herein, is to design a CDA algorithm that not only eliminates the shortfalls associated with centralised CDA algorithms, but is also sensitive to computational resources.

3.4 Robustness to Failure

3.4.1 Fault tolerance in MUTEX protocols

In Chapter 4 we propose a token-based MUTEX protocol to support the decentralised CDA. This implies the proposed CDA formulation inherits properties of the distributive primitives. Token-based MUTEX algorithms serialise access to the critical section (CS), through the maintenance of a single token, which cannot simultaneously be accessed at more than one node in the system. Requests to enter the CS are directed to whichever node is the current token holder.

The token-based MUTEX approach being proposed for our CDA scheme (Section 4.3) was inspired by Raymond's algorithm [54]. In Raymond's algorithm, requests are sent over a static spanning tree of the network, toward the token holder. Furthermore, Raymond's algorithm is resilient to non-adjacent node crashes and recoveries, but not node/link failures [54]. Chang *et al.* in [82] extend Raymond's' algorithm by imposing a logical direction on a number of edges to induce a token-oriented *directed acyclic graph* (DAG), where there exists a directed

path originating from n token-requesting nodes while terminating at the token holding node. Resilience to link and site failures is achieved by allowing request messages to be sent over all edges of the DAG. Chang *et al.*'s solution incurs high message overheads and fails to consider link recovery. Dhamdhere *et al.* [83] show that Chang *et al.*'s algorithm will suffer from deadlock then go on to propose a dynamically changing sequence number to each node to form a total ordering of the nodes. Since the token holding node possesses the highest sequence number, a DAG is maintained if the links are defined to point to a node with higher sequence number. If a node has no outgoing links to the token holder (due to link failure), it will flood the network with messages to build a spanning tree. Once the token is part of this spanning tree, a token will be granted to the particular requesting node, bypassing other earlier requests. Persistent link failures will inevitably lead to starvation since priority is given to the nodes that lose a path to the token holding node. Similar to Chang *et al.*'s algorithm, flooding of messages incurs overhead not suitable for a typical resource constrained setup. Walter [84] presents a reversal of the technique, where a destination-oriented DAG is maintained if it its a dynamic destination. Ordering of nodes is similar to that of Dhamdhere [83], but the lowest node is assigned the token. The aforementioned solutions mainly address the issue of link failure, with little effort on addressing node failure, explicitly assuming that nodes do not fail and network partitioning does not occur. Another body of work towards solving node failure is evident. Revannaswamy *et al.* [85] propose a solution to handle failures of nodes and links in a network with definitions carried over from [54]. Their attempt at fault tolerance involves eliminating the failed nodes and obtaining a different tree structure utilising chords (edges of the tree yet unused for message exchanges) from the network. A reconfiguration process attempts to connect parts of the tree separated due to failures. Since reconfiguration eliminates the failed components, if the static cluster heads (which act as proxies to the mobile nodes) fail, the trading agents residing on the mobile nodes will not be able to take part in the auction market. Thus, we argue that this approach is not an efficient solution to ensure reliability and availability in the given context of computationally constrained, micro-grids.

3.4.2 Decentralised Consumption Scheduling

Demand Side Management (DSM) facilitates power demand profile smoothing across time by avoiding peak power periods [86]. These solutions minimise the power costs while guaranteeing user satisfaction; this can be achieved through optimisation methods [17], [18]. Standard approaches to addressing such optimisation problems on smart grid networks include dual decomposition, and augmented Lagrangian methods [87]. However, dual decomposition methods are not robust, requiring many technical conditions, such as strict convexity and finiteness of all local cost functions. Augmented Lagrangian methods can be used to bring

robustness to the gradient method, and in particular, to yield convergence without assumptions such as strict convexity or finiteness of the objective function [88]. Nevertheless, this method has the disadvantage of not being separable across the devices in the network. ADMM can be used to achieve both separability and robustness for distributed optimisation [88]. Kraning *et al.* [89] studied an energy management model for a large-scale electrical power network using ADMM. The problem is solved in a distributed manner by alternating between the parallel optimisation of single device objective functions and computing average power imbalances in the nets which the devices belong to. Wei *et al.* [90] proposed an asynchronous ADMM algorithm with a convergence rate of $O(1/k)$, where k is the number of iterations. Although their method applies to a general network utility maximization problem, the results can be adapted to solve the power consumption scheduling problem. This work employs a decentralised optimisation approach based on the ADMM method to solve the power consumption scheduling problem in a distributed manner.

3.5 Detecting and Mitigating Cheating

3.5.1 Understanding Cheating Attacks

Faults and Failures are not the only problems that one should seek to address. CDA algorithm design and deployment should take into account the fact that actors in the auction market are self-interested [12], implying they may misrepresent their preferences (e.g. amount of electricity required, the capacity they can supply and prices they would accept) or even change agent's bidding strategy to maximise their profit [27]. Cheating is arguably the most significant group of auction frauds forming the bulk of all internet frauds [23], [25]. Cheating unlike other fraud categories leaves no direct evidence of its occurrence, while financial loss resulting from such cheating behaviour cannot be precisely measured. Cheating is encouraged by: cheap user pseudonyms; greater information asymmetry; lack of personal contact between participants; and the tolerance of bidders [91].

The tendency for participants to cheat or employ strategies to gain some economic advantage, disrupts services and hinders trust, which is necessary for incentivising energy sharing among the members of such a community. Therefore, it is common for system defenders to study common cheating attacks with the intention of understanding and developing robust defence mechanisms. Most widely studied classic cheating attacks occur in single-sided auctions and to a less extent centralised CDAs. These classic forms include: *Multiple bidding*[26]; *Bid shading* [92]; *Rings* [91]; *Shill bidding* [91], [93], [94]; *Misrepresented/ non-existent items*; and *Bid Snipping*. Trevathan and Read in [91] note that cheating is auction mechanism specific, evident in the decentralised CDA being fairly comprehensive in discouraging some standard cheating forms

from occurring. This means, system defenders cannot rely on analysing only classic attack forms to equip themselves with information for deployment of effective defence solutions. We observed this notion in our work [27], indicating how exploitation of agent-to-agent strategy dynamics in the market opens new avenues for automated cheating attacks.

In summary, we learn that some $\mathcal{TA}$ strategies are superior, with the ability to gain more surplus from trade than their inferior counterparts. Experimental evidence indicates that the Adaptive Aggressive strategy is one such superior strategy, while the Zero Intelligence is the most inferior [95], [1], [74]. This could be attributed to AA agents being adaptive to different combinations of competitors; and to different supply and demand relationships. ZI agents low performance could be attributed to their failure to analyse and adapt to their environment and the competition. The AA obtains huge profit margins in comparison to ZI strategy [36]. In addition, we also learn that some TA strategies are only superior within given population ratios [96], which implies that changing agent population ratios can lead to fairly predictable surplus gain for specific strategies over others.

Automation in auctioning can pave way to an interesting set of automated cheating forms which to the best of our knowledge had not been documented prior to our published work in [27]. This means system defenders face a huge challenge of designing a robust security solution that can capture such novel attacks. One way of addressing such a challenge suggested by thesis is creating a more encompassing systematic approach that allows the design and study of cheating attacks. To the best of our knowledge, there is no systematic framework available, that allows the design and study of such cheating attacks on CDA platforms. As affirmed by [97] [98] [99], attack models and exploitation scenarios provide a valuable source of information, which inspires development of effective mitigation solutions (e.g. Intrusion Detection Systems (IDS)). In the next subsection we review the attack modelling and evaluation literature.

3.5.2 Attack Modelling

One way of approaching the issue of security in a decentralised agent-based CDA is by modelling random cheating attacks, which can be helpful in understanding potential vulnerabilities and plausible exploits. However, such an approach yields specific attacks that may be difficult to generalise (since there are specific to the auction algorithm they target) and does not give enough coverage of cheating attacks that can help defenders to develop more elegant and effective solutions. To provide such a broader coverage of attacks, one envisages the use of security quantification approach.

According to Sedaghatbaf *et al.* [100], security quantification can take one of three forms: analysis of large amount of logged operational data; use of simulation techniques and tools; and construction of analytic models. Analysis of logged data is straightforward, but less

desirable. Such an analysis is practical, only useful after an incident of a security breach occurs. Additionally, it can be an expensive approach, as it requires building a real system, taking measurements and analysing the data statistically. Thus, simulation techniques and tools can provide an alternative, but they also suffer from lack of proper techniques and tools specific to security quantification. Constructing analytical models is preferred, since it can be performed in an a priori manner with less costs. In this thesis, we carry out a model-based security evaluation approach to provide coverage of attacks in a decentralised CDA. Such an approach is novel within the CDA security research community.

The existing model-based, security evaluation approaches can be categorised from the attacker behaviour viewpoint into behavioural [101] [102] [100] and non-behavioural/ technical approaches [103][104] [105] [106]. As observed in Niitso [102] and Sedaghatbaf [100], attack modelling approaches proposed in the literature have focussed mostly on the technical aspects and finding possible attack vectors. Technical approaches are based upon simplistic assumptions about the factors that may affect attacker's decisions without considering factors such as the costs of bribing people that may affect an attacker's decisions on whether to perform the attack, and how to perform it. In this thesis we propose an *ACA* framework (Section 7.2), that provides more coverage of attacks on a decentralised CDA by considering both, technical and behavioural/non-technical aspects of attack modelling. The framework is inspired by the work of Adepu and Mathur [103]. Adepu and Mathur developed a framework that enables researchers to design a variety of cyber and physical attacks for the assessment of attack detection methods and tools. Their framework defines attackers intent but fails to explicitly consider human behaviour aspects (e.g. attacker's rational and attacker's ability) which our framework considers.

3.5.3 Cheating Mitigation

Cheating is specific to the auction mechanism, which implies that cheating forms and the related countermeasures are dependent on the auction mechanism. To the best of our knowledge, research in [22] and [24], is the only closest work that specifically addresses CDA security. Wang and Leung in [22], describe an anonymous and secure CDA protocol for electronic marketplaces, which is strategically equivalent to the traditional CDA protocol. Trevathan *et al.* in [24], demonstrated that, Wang and Leung's scheme [22], allows bidder identity to be revealed immediately after his/her first bid. Furthermore, it allows profiles to be created about a bidder's trading behaviour, as bids are linkable. Hence, in their scheme they propose incorporating a group signature scheme to anonymise traders and secure the trading process. Wang and Leung's scheme is given in the context of Internet retail markets while Trevathan *et al.*'s scheme was designed specifically for share market applications. Although the schemes described in these works could protect the customer's privacy including affording anonymity,

robustness and non-repudiation, but they are not suitable for deterring automated cheating in decentralised CDAs such as the one we describe in this thesis. This is due to the following reasons:

- *Presence of a centralised auctioneer:* the CDA mechanisms discussed in the aforementioned works [22], [24], rely on a centralised architecture, where bids are relayed through a central component (Auctioneer), which determines the winner according to the auction rules. The problem arises when an Auctioneer influences the auction proceedings in a manner inconsistent with the auction rules. For example, the Auctioneer may block bids, insert fake bids, steal payments, profile bidders, prematurely open sealed bids, artificially inflate/deflate prices or award the item to someone other than the legitimate winner. Protecting a bidder's identity and bidding information is crucial since each bidder/seller's private information can be inferred at the central Auctioneer. Furthermore, the Auctioneer presents a single point of failure, is open to biases and can be easily be manipulated to obtain favourable trades or reveal traders' reserved information. In a decentralised CDA [28], the auctioneering duties are carried out by a mobile-token distributed among the participants following a MUTEX protocol. Franklin and Reiter [107] suggest distributing the role of Auctioneer amongst s servers. The auction can be considered secure/fair unless a threshold t, $1 \leq t \leq s$ of the servers collude. However, these types of schemes have a high communication overhead and cannot be trusted when the same company owns all the servers.
- *Different auction clearing mechanisms:* work by Wang and Leung [22] and Trevathan *et al.* [24] considers a double auction mechanism where the market clears periodically. Such a market mechanism allows bidders and sellers to submit bids for a period where the auctioneer then clears the matching bids. The CDA mechanism we consider (see Section 3.3) lasts for a fixed period, known as the trading period (at the end of which the market closes and no more offers are accepted). The traders will continuously submit offers at anytime during the trade period while the market continuously clears matching *bids* and *asks* (i.e., whenever a new transaction is possible between an acceptable bid and ask). In addition, the trading parties mutually and exclusively submit an order into a mobile *order-book* to negotiate a deal. These differences in clearing mechanisms means the auction schemes are susceptible to different attacks; plausible mitigation methods that can be employed also differ. These works are valuable to our work, as they provide a form of benchmark for our security considerations towards a cheating attack.
- *Inadequate and resource intensive solutions:* Cryptography is an effective approach in deterring cheating, resulting from manipulations via communication channels or those emanating from a corrupt centralised auctioneer [25]. While cryptography helps in bid authentication and privacy [25], [108], [94], the literature lacks solutions towards preventing automated forms and other classic forms of cheating [91]. These include bid

sniping, shill bidding, bid siphoning and misrepresented goods [108]. Furthermore, many cryptographic operations are computationally intensive, and are thus not favourable in RC environments (such as one described in Section 2.2).

3.5.4 Automated Security Preserving Mechanisms

While auction developers may tinker with new auction formats and policies, many of the aforementioned problems remain unresolved and threaten the future of online auctioning [21]. Further, such modifications of the auction protocol alters desired core properties offered by an auction mechanism. Modification of the CDA can for example result in a double auction protocol similar to the Threshold Price Double auction (TPD) protocol [26], which is dominant strategy, incentive compatible and can be used to make trading agents declare their real evaluation value. In this particular case, such a modification significantly increases the number of message overheads. In addition, CDAs are known to perform efficiently over TPDs [73]. Intuitively, a comprehensive solution is one that incorporates a cryptographic model (to solve the main security and privacy problems), proactive programs (to detect cheating), and an auction format that discourages cheating. In this thesis we contribute to this cause by proposing a proactive protocol to detect and resolve automated forms of cheating. To the best of our knowledge, adding a distinct, proactive detection and mitigation protocol has never been done to address cheating in CDAs. We consider cheating attacks (discussed later in Chapter 7) to be exceptions[2] that can be resolved by crafting an Exception Handling (EH) mechanism that conforms to one of two approaches: the Citizenship / Organisational view and the Survivalist / Agent view approach.

Citizenship approach: This involves an entity/entities situated outside the system agents [70], [109], [110], [111], which monitor by listening to agent's internal events and messages passed to detect exceptional situations. There is considerable support of this [112]. Related work subsumed by the citizenship approach lies into two further sub-categories; centralised (a single entity is responsible for handling exceptions) [110], [113] and decentralised views (EH is a shared responsibility by more than one entity) [109], [111]. For instance, Tripathi and Miller [110] assigned an agent, named a guardian, in a MAS that manages EH centrally for global exceptions. Klein and Dellarocas [113] proposed a shared EH service to the whole MAS. At first glance this approach seems decentralised, but, sentinel agents responsible for exception detection are the only entities distributed. Centralised view yields higher overheads, undesirable for computationally constrained, environments. Klein and Dellarocas improve their approach by proposing a more decentralised approach which assigns the responsibility of EH to sentinel agents [109]. Every agent has a sentinel agent that controls agent communication

[2]Exceptions arise in situations where an agents' behaviour falls outside the normal, system operating conditions.

and have the ability of using a centralised reliability database shared by all sentinels. Similarly, Haegg [111] proposes special agents (also called sentinels) which are assigned to each system agent and controls the agent's communication for error detection and recovery.

The citizenship approach works in our scenario (see Section 2.2) as it is based on the same assumption that no-one has access to the trading agents' internal state. The main criticism of the citizen approach is that, when the number of participant agents is huge, EH requires more resources for the specialised agents to handle exceptions of the whole system. Adopting such an approach should consider this drawback. We are aware that it is essential to reduce the number of messages passed among the participating agents and this is discussed in our solution (in Chapter 7).

Survivalist approach: This is where individual system agents are elaborately designed to cope with all the exceptions that they might face [114], [115], [116]. Souchon *et al.*, [114] proposes a layered handling approach depending on Java call stack structure- SaGe Framework. Mallya and Singh [115] proposed commitment protocols, where an exception is identified by violation of a protocol. Exceptions are handled by definitions of the recovery plan for the exceptional situation. Cakirlar, Ekinci and Dikenelli [116] classify multi-agent exceptions and implement these levels with their approach on *SEAGENT* goal-oriented multi-agent development framework. Coöperation and internal intents of agents are modelled with goals. The mechanism to handle exceptions is embedded inside the agent.

Designing survivalist agents greatly increases the burden on agent developers, as they have to anticipate and correctly prepare for all exceptions an agent may face in the environment it may have to operate. In addition, agents become difficult to maintain, understand and reuse as a potentially large body of EH code may obscure the relatively simple normative behaviour of an agent. Due to the aforementioned con's of the survivalist approach in comparison to pro's the citizen approach the citizen approach can provide an effective approach.

Overall, use of EH is not unknown in double auction schemes, as shown by Parsons and Klein's [70]. The authors investigate how EH may address exceptions resulting from unreliable infrastructure in double auctioning applications- a significant departure from our work. Employing the citizen approach, the authors following the approach in [109], define an EH infrastructure that associates a sentinel to every agent. Sentinels provide EH service to agents there are associated with. Work in [70] is valuable to our work as it inspired the design of our cheating EH protocol. The novelty of our *automated security preserving solution* comes from crafting an EH mechanism that deters automated forms of cheating (Non-compliant agents exceptions). This is a significant departure from Parsons and Klein's work [70], whose EH mechanism addresses communication disturbances (Unreliable infrastructure exceptions).

Chapter 4

A Decentralised Continuous Double Auctioning Algorithm

All models are wrong but some are useful

GEORGE BOX.

4.1 Overview

This chapter details our proposed decentralised CDA algorithm for RCSMG, that provides an additional abstract power management layer, which is crafted to allow automated and seamless allocation of power, while eliminating issues of technology literacy, promoting low communication overheads, and handling token loss and some link failures. Our CDA algorithm can leverage autonomous intelligent trading agents that participate on behalf of the human participants to negotiate power trade. As a prerequisite to this chapter, the reader is advised to familiarise themselves with the smart RMG model, CDA model and MUTEX model described in Chapter 2. We make further assumptions to the aforementioned models (in Section 4.2). Section 4.3 then details assumptions and the overarching fundamental principles that guided the development of our proposed decentralised CDA algorithm. In Section 4.3.4 we discuss the data structures and control messages of the algorithm and then in Section 4.3.5 we present the algorithm's procedures. Our token-based CDA algorithm is theoretically shown to satisfy the *MUTEX* properties, while yielding a time complexity of $\mathcal{O}(N)$ and a message complexity of $\mathcal{O}(\log N)$ (Section 4.5). Performance is of paramount importance as the CDA scheme is to be deployed in a resource-constrained environment. Furthermore, we study the effects of faults, particularly token loss and link failure. Extensive fault tolerance measures are beyond the scope

of this chapter and will be addressed in Chapter 5. Section 4.7 concludes this chapter by relating the chapter contributions to the overall thesis problem.

4.2 Assumptions

For simplicity, we consider the following additional assumptions to the constraints and assumptions made in Section 2.2:

- A correct routing protocol exist for successful communication;
- There is a fairly stable connection between the shared smart meters (M_{sm} — M_{sm} links) as cluster heads;
- Messages are delivered within a finite time;
- There is no message loss during transmission and the receiving node (M_{sm}/ M_{mp}) can receive messages without any error or distortion;
- We do not anticipate M_{sm} failure and thus assume the M_{sm} as cluster heads are never disconnected.

4.3 Algorithm Principle

While there exist many variants of CDAs, we structure our CDA around the market protocol initially proposed by Smith [2]. We consider the smart micro-grid setup described in Chapter 2 and we refer the reader to Section 2.2.4.1 for a description of fundamental Continuous Double Auction concepts. We consider that our CDA protocol includes the New York Stork Exchange (NYSE) spread-improvement and the *no-order queuing rules*. The agent traders take part in a single CDA market, trading single-type, homogeneous goods —electric power. The descriptor of a CDA is a septuple [49]

$$X_{CDA} = \{\rho, \tilde{B}, \tilde{S}, V_b, C_s, \Delta_{price}, t_{round}\} \tag{4.1}$$

where:

- ρ is the power in single units to be auctioned.
- $\tilde{B} = b_1, ..., b_n$ is the finite set of identifiers of buyer $\mathcal{TA}$ s, where n is the number of buyer $\mathcal{TA}$ s.
- $\tilde{S} = s_1, ..., s_m$ is the finite set of identifiers of buyer $\mathcal{TA}$ s, where m is the number of seller $\mathcal{TA}$ s.

- $V_b = (V*_1, ..., V*_n)$, where $V *_i (v_{i1}, v_{i2}, ..., v_{in_i})$ is a vector of unit valuations of $\mathcal{TA}$ b_i. Here, n_i is the number of units of p that b_i requires, and v_{ij} is the valuation value for the j^{th} unit acquired.
- $C_s = (C*_1, ..., C*_m)$, where $C *_i (c_{i1}, ..., c_{im_i})$ is a vector of unit costs of $\mathcal{TA}$ s_i. Here, m_i is the number of units that s_i wants to sell, and c_{ij} is the cost of the j^{th} unit.
- Δ_{price} is the minimum price step required in the auction. That is, a buyer (seller) $\mathcal{TA}$ must increase (decrease) its bid (ask) at $nx\Delta_{price}$, where n is a non-negative integer.
- t_{round} is used for defining the condition for terminating the CDA; that is, if there are no new asks or bids during a time period t_{round}, or the maximum threshold of rounds per day R is reached, the CDA terminates.

Formally, we consider $\tilde{S}$ and $\tilde{B}$ to participate in the auction. Each or β abides to an acceptable price range $[Pll, Pul]$, where Pll is the acceptable lower limit price while Pul is the acceptable upper limit price, formed on the basis of the $\mathcal{TA}$'s experience and the trading history of the market. Each $\mathcal{TA}$ (buyer/seller) has a secret reservation value λ_i, where $(i = 1, 2, ...N)$, for a single unit allocation of energy. Without loss of generality, we assume $\lambda_i \in [0, 1]$. For a seller or buyer $\mathcal{TA}$, each unit of goods has a reservation price (limit price), which is secret to each trader. If a seller submits an ask lower than the λ, he will lose profit. If a buyer submits a bid higher than λ, he will also lose profit (no surplus profit). At any single time, only one trader has the exclusive right to submit an offer. The participant can send a new offer when their earlier offer has been cleared (if it was matched and the deal was accomplished).

A round r in the CDA is the time period between two successive transactions or the period from the beginning of the CDA to the time when the first transaction occurs. If a round is the $q^{th}(q \in IN^+)$ round of the CDA, then q is called the round number. A CDA usually consists of multiple rounds which form the *trade period*, R. Thus, $R = r_1, ..., r_n$ is the finite set of rounds in a single run or trade day. For a CDA that has lasted r rounds, where $(r > 0)$, let $p_i(1 \leq i \leq r)$ denote the price of the i^{th} transaction. A history H_l in a CDA is the set of transaction prices during the last l rounds,

$$H_l = \{p_{r-l+1}, \cdots, p_i, \cdots, p_r\},$$

where $p_i(r - l + 1 \leq i \leq r)$ is the transaction price of round i, and $l(l \leq r)$ is called the history length. We define *supply* to be the total number of units of power that all the sellers can sell in a trade period /run, R. We define *demand* as the total number of units of energy that all the buyers desire in R.

4.3.1 Auctioning Activities

Activities considered in our decentralised CDA scheme can be summarised into:

1. Registration: In order to participate in the auction, agents must first register (with a registration manager). This is a once-off procedure. When an agent is registered, it can participate in any number of auctions rounds. At the beginning of each R, $\mathcal{TA}$ s enter a "mock marketplace", to determine the equilibrium price and overall surplus distribution.

2. Initialisation: The token (a mobile object) is initialised, such that $r = 0$. After this initialisation, when the equilibrium price is found, all trades will commence at this price.

3. Participation Request: A new round of the CDA starts, $r = r + 1$, $oa = \infty$, and $ob = 0$. Any agent willing to submit an offer in the market will request for the token. A MUTEX protocol is used to serialise a fair market access to the participants.

4. Bid Formation: An agent that receives the token will compute their offer (bid/ask) using the AA trading strategy and submit it into the token (see Definition 2.1).

5. Transacting: Considering the offer is not invalid, if a matching bid/ask is found the transaction is instantly concluded and a trade occurs. Otherwise, the offer is put in the order-book as an outstanding ask/bid. Thus, several situations might arise during a round:

   (a) When a seller $\mathcal{TA}$ submits an $aska$,
   - if $a \geq oa$ then a is an invalid ask;
   - if $ob < a < oa$, then oa is updated to a;
   - if $a \leq ob$, then this seller $\mathcal{TA}$ makes a deal at ob; go to 3.

   (b) When a buyer $\mathcal{TA}$ submits a bid of b,
   - if $b \leq ob$, then b is an invalid bid;
   - $if ob < b < oa$, then ob is updated to b;
   - if $b \geq oa$, then this buyer $\mathcal{TA}$ makes a deal at oa; go to 3.

   (c) This process repeats until no new bids (asks) are submitted during a time period t_{round}.

   The trade information and outstanding offers are made public and visible to other agents. The number of trades and wins for each $\mathcal{TA}$s are recorded and kept in the token. We consider 'truthfulness' similar to [9], where both buyers and sellers full-fill their contractual obligations once bids are matched.

6. Termination: The order-book is mutual exclusively distributed until the end of a trade day.

For the consumers (prosumers) to satisfy their need for energy, while producers (prosumers) gain as much profit as possible, from selling the energy, an effective adaptive agent strategy is expected.

4.3.2 Agent Strategy

We understand that heterogeneous $\mathcal{TA}$ (with different strategies) can operate in the same CDA market but for our study we consider a homogeneous population of agents employing the Adaptive-Aggressive Strategy[1] (AA). Our option was inspired by the AA strategy's performance superiority against other benchmark strategies, and its ability to adapt to dynamic environments, where classic strategies fail. All $\mathcal{TA}$ can participate in the auction market, with exclusive offer submission.

4.3.3 Serialization of Market Access

As a recap, the proposed distributed CDA protocol results in a mutual exclusion problem (see Section 2.3.1). Since at most, one $\mathcal{TA}$ can submit an offer in the auction market at a single instance of time, we propose serialization of market access and fair opportunities to submit an offer. In traditional CDA algorithms when a successful bid or transaction or bid/ ask is made a broadcast is sent to all participants. This means the number of messages may be considered undesirable within a constrained environment. Thus, a workaround we adopt in our CDA algorithm, is a limit on the number of broadcasts (or total elimination). If no broadcast occurs, the market information is made available when the token arrives at a $\mathcal{TA}$. This assumption can be relaxed, by allowing a minimum number of broadcast messages to be sent during the trade day. The argument is that, even in this relaxed condition, the expected number of messages would be greatly reduced to favour deployment in a constrained environment. In such a case, the $\mathcal{TA}$ s participation is driven more by factors such as the households' extended need for energy or need to sell, than the transaction price broadcast. In the following subsection we shall discuss the considered data structures and control messages.

4.3.4 Data Structures & Control Messages

Two types of request messages are passed by nodes requesting to take part in the auction market: Req_{sm} (global token request) and Req_{mp} (local token request). An incoming Req_{sm} is enqueued in a FIFO $RQ1$ queue while Req_{mp} is enqueued in the $RQ2$ FIFO queues at the M_{sm}. At the M_{sm} nodes is a *POINTER* variable which stores the location of an M_{sm} in possession of the token, or

[1] Developed by Vytelingum in his PhD thesis [1] and then presented in [74]

next intermediate M_{sm} pointing to that token holding node (see [54]) where Req_{sm} is sent. When a Req_{sm} is sent, the $TokenAsked$ boolean variable is set to *TRUE* avoiding continuous request messages to be send for the token by the same node. On arrival of the token to M_{sm} and M_{mp} the boolean variables $FlagM_{sm}$ and $FlagM_{mp}$ are set to *TRUE* indicating possession of the token. The *GQ* FIFO queue at M_{sm}, stores a copy of requests submitted and "locked-in" at token arrival on the cluster head. *TokenOB* is an online copy of the CDA order-book carried in the token. *LocalOB* is a local copy of the CDA order-book updated each time a trading agent participates in the auction market. Each M_{mp} has a *ClusterDir* that contains a directory of neighbouring M_{mp} nodes. A *TokenCounter* keeps a record of the number of auction market rounds. When the predefined number of rounds is reached trading is terminated and an end-of-trading-day (t_{round}) message may be communicated to the rest of the participating nodes. This message includes trading-day market information. To get a clearer understanding of how these data structures and control messages function we shall discuss the routines executed by the nodes.

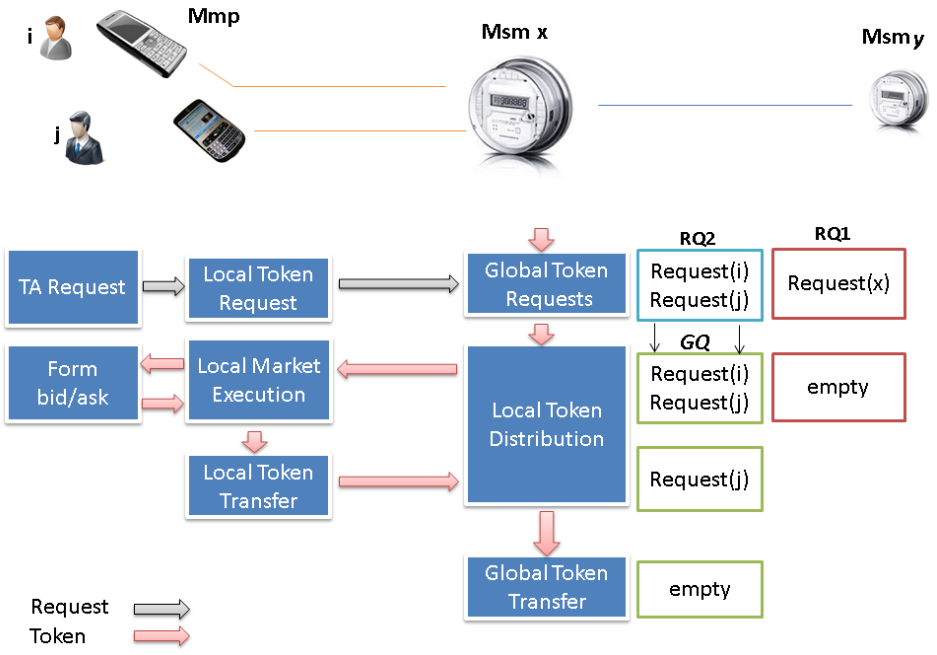

FIGURE 4.1: Decentralised CDA Procedures

4.3.5 Procedures & Routines

The $\mathcal{TA}$ s will execute the following routines:

- *$\mathcal{TA}$Request;*
- *FormOffer*

At the M_{sm} the following routines are executed:

- *LocalTokenDistribution*;
- *GlobalTokenRequest*;
- *GlobalTokenTransfer*

At the M_{mp} the following routines are executed:

- *LocalTokenRequest*;
- *LocalMarketExecution*;
- *LocalTokenTransfer*

Local Token Request: When a $\mathcal{TA}$ needs to trade in the auction market, the $\mathcal{TA}$ triggers the hosting M_{mp} node to send a Req_{mp} to the M_{sm} provided that battery is above the critical condition and it does not already possess the token. When a Req_{mp} is sent to M_{sm}, the sleep mode is deactivated to ensure the M_{mp} node remains online to receive the token. Algorithm 4.1 presents the $LocalTokenRequest$ Procedure.

Algorithm 4.1: $LocalTokenRequest$ Procedure

Input : $Req_{TA}, BatteryLife, CriticalCondition$
Output: $TokenRecieved$

```
1  FlagM_mp ← FALSE ;
2  TokenReceived ← FALSE;
3  for each M_mp ∈ M_mp,a[.] do
4      while FlagM_mp = FALSE do
5          if Req_TA = TRUE then
6              if BatteryLife > CriticalCondition AND FlagM_mp = FALSE then
7                  Send ReqM_mp to Local M_sm ;
8                  Disable doze mode ;
9                  Wait for TOKEN ;
10             else if BatteryLife ≤ CriticalCondition AND FlagM_mp = FALSE then
11                 Do not send ReqM_mp ;
12                 Wait till BatteryLife > CriticalCondition ;   // Battery life determines
                       participation
13         else
14             Do nothing ;                    // TA has not requested for participation
15     TokenReceived ← TRUE;
16     return TokenReceived ;
```

Local Market Execution: This procedure is executed on receipt of the token at M_{mp}. The $TokenCounter$ variable is incremented and the $FlagM_{mp}$ is set to $TRUE$. The $\mathcal{TA}$ forms an offer (bid/ask) which is submitted into *TokenOB* and *TokenCounter* is incremented. If the predefined number of rounds is reached, the trading day is terminated. Success or failure of

an $\mathcal{TA}$ s offer to result in a trade does not affect passing-on of the token. The Algorithm 4.2 presents the $LocalMarketExecution$ Procedure.

Algorithm 4.2: $LocalMarketExecution$ Procedure

Input : $TokenRecieved, TokenCounter++, R, TA_a[.]$.
Output: R, $TokenCounter$ $TokenOB$.

```
1  R ← 1000;
2  TokenCounter ← 0 ;
3  TA_i ← ∅ ;                                          // Where i = 0, 1, 2...  N
4  for each M_mp ∈ M_mp,a[.] do
5      if TokenRecieved = TRUE AND FlagM_mp = TRUE then
6          TokenCounter++ ;                          // The Token Counter is incremented
7          Set FlagM_mp to TRUE ;
8          Ask TA_i to FormOffer() ;                  // A TA submits its offer
9          if tradeID = buyer then
10             bid ← offer ;
11             if bid ≤ ob OR out of [P_ll, P_ul] range then
12                 bid is invalid ;
13             else
14                 ob ← bid ;
15             if ob ≥ oa then
16                 P_t ← oa ;
17                 Trade and TokenOB update ;
18         else if tradeID = seller then
19             ask ← offer ;
20             if ask ≥ oa OR out of [P_ll, P_ul] range then
21                 ask is invalid ;
22             else
23                 oa ← ask ;
24             if ob ≥ oa then
25                 P_t ← ob ;
26                 Trade and TokenOB update ;
27         else
28             no new oa or ob in a pre-specified time period ;
29             Round ended with no transaction ;
30         Update LocalOB from TokenOB ;          // Local Orderbook copy is updated
31         R−− ;                                  // Auto-decrementing remaining trade rounds
32         return TokenOB, R, TokenCounter ;
33     else
34         wait for TOKEN ;
```

Local Token Transfer: On completing of the Local Market execution the token is returned to the cluster's M_{sm}. When direct connection to an M_{sm} is unavailable (due to various reasons), the token possessing M_{mp} will try send the token via a close neighbouring M_{mp} in its *ClusterDir* with the best connectivity to the cluster M_{sm}. The Algorithm 4.3 presents the $LocalTokenTransfer$ Procedure.

Algorithm 4.3: $LocalTokenTransfer$ Procedure

Input : $LocalOB$, $M_{mp}a[.]$.
Output: $TOKEN$

```
for each M_mp ∈ M_mp,a[.] do
    if Connection to M_sm = ALIVE then
        Return Token to M_sm ;
        else if Connection to M_mp via M_sm = ALIVE then
            Send TOKEN to M_sm ;
        else
            Destroy the TOKEN ;
            Revert back changes in the LocalOB ;
    Set Flag M_mp to FALSE
return TOKEN ;
```

Global Token Request: An M_{sm} node uses this procedure to send a request for the token. A Req_{sm} is sent to the neighbour node holding the token or in path to the token and $TokenAsked$ is set to *TRUE*. This ensures that similar request messages are not sent to the same token holder. $EnqueueRequest$ routine is executed while M_{sm} is waiting for the token. $EnqueueRequest$ enqueues the Req_{sm} and Req_{mp} requests into *RQ1* and *RQ2* respectively. The Algorithm 4.4 presents the $GlobalTokenRequest$ Procedure.

Algorithm 4.4: $GlobalTokenRequest$ Procedure

Input : $TokenRecieved$, $ReqM_{sm}$, $ReqM_{mp}$, $TOKEN$, $M_{sm,i}[.]$.
Output: $FlagM_{mp}$, R, $TokenCounter$ $TokenOB$.

```
for each M_sm ∈ M_sm,i[.] do
    if FlagM_sm = FALSE AND RQ1 ≠ 0 AND TokenAsked = 0 then
        Send ReqM_sm to M_sm in POINTER ;
        Set TokenAsked to TRUE ;
        Assign TokenReceived = FALSE ;
        while TokenReceived = FALSE do
            EnqueueRequest(ReqM_sm, ReqM_mp) ;  // Requests are enqueued in RQ1 and
                                                   RQ2
    else if FlagM_sm = FALSE AND RQ1 ≠ 0 AND TokenAsked = 0 then
        while TokenReceived = FALSE do
            EnqueueRequest(ReqM_sm, ReqM_mp) ;
```

Local Token Distribution: Once the token is received the algorithm moves to the *LocalTokenDistribution* routine. Receipt of the token by the requesting M_{sm} node sets the boolean variable $FlagM_{sm}$ to *TRUE* indicating possession of the token. *GQ* is a FIFO queue that stores a copy of requests submitted and "locked in" at the arrival of the token to the cluster head. Once the token is received, all requests in *RQ2* are transferred to *GQ*. While there are still requests in *GQ*, the algorithm will run the $EnqueueRequest$ and $BackupNode$ subroutines before sending the token to an M_{mp} node with a request at the head of *GQ*. After token is allocated to an M_{mp} node,

the nodes corresponding Req_{mp} entry is removed from *GQ*. The M_{sm} node expects the return of the token within a predetermined time. To ensure that M_{sm} does not wait for the token for an undefined time, the routine will consider a time bound on the wait period. Failure of the M_{mp} nodes to return the token within the predefined time results in the token degradation (at M_{mp}) and regeneration (at M_{sm}) from the last known backup. The Algorithm 4.5 presents the $LocalTokenDistribution$ Procedure.

Algorithm 4.5: $LocalTokenDistribution$ Procedure

Input : $TokenRecieved$, $ReqM_{sm}$, $ReqM_{mp}$, $TOKEN$, $M_{sm,i}[.]$.
Output: $FlagM_{mp}$, R, $TokenCounter$ $TokenOB$.

```
1  for each M_sm ∈ M_sm,i[.] do
2      if TokenRecieved = TRUE AND self ReqM_sm is at head then
3          Set FlagM_sm to TRUE ;
4          GQ ← RQ2 ;
5          n ← number of GQ entries ;
6          repeat
7              if M_mp at head ≠ disconnected then
8                  DequeueRequest(ReqM_mp) ;                    // Remove request from GQ
9                  Assign token to M_mp ;
10                 if TokenReturn = TRUE then
11                     Move to next GQ entry ;
12                 else
13                     TokenReturn = timed-out ;      // When a token return has timed out
14                     DequeueRequest(ReqM_mp) ;
15                     TokenRegenerate(BackupID) ;
16             else
17                 Cancel request entry in GQ ;
18                 Move to next GQ entry ;
19             EnqueueRequest(ReqM_sm, ReqM_mp) ;
20             n − − ;
21         until n < 1;
22     else
23         Wait for TOKEN ;
24         EnqueueRequest(ReqM_sm, ReqM_mp) ;
```

Global Token Transfer: A token possessing M_{sm} node will send the token if it has a non-empty *RQ1* (where Req_{sm} at head of the *RQ1* is not its request). The boolean $FlagM_{sm}$ is then set to $FALSE$ and the token is sent to the respective M_{sm} node with a Req_{sm} at the head of *RQ1*. The Algorithm 4.6 presents the $GlobalTokenTransfer$ Procedure.

Algorithm 4.6: $GlobalTokenTransfer$ Procedure

Input : $TokenRecieved$, $ReqM_{sm}$, $ReqM_{mp}$, $TOKEN$, $M_{sm,i}[.]$.
Output: $FlagM_{sm}$, R, $TOKEN$.

1 **for** *each* $M_{sm} \in M_{sm,i}[.]$ **do**
2 **if** $FlagMsm = TRUE\ AND\ RQ1 \neq 0\ AND\ ReqM_{sm}! =' self'$ **then**
3 Set $FlagM_{sm}$ to $FALSE$;
4 Send $TOKEN$ to $ReqM_{sm}$ at $RQ1$ head ;
5 Return $TOKEN$;

4.4 Algorithm Correctness

In this section we show how our decentralised CDA algorithm guarantees the MUTEX properties:

4.4.1 ME1- Mutual Exclusion

Our CDA scheme guarantees that only one node exclusively holds the token that at any instant of time. Each time a node receives a token in Algorithm 4.4 and 4.5, it becomes exclusively privileged: (line 15) and (line 3) respectively. Similarly, each time a node sends the token in Algorithm 4.3 and 4.6, it becomes unprivileged (line 9) and (line 3). Between the instants one node becomes unprivileged and another node becomes privileged, there is no privileged node. Thus, there is at most one privileged node at any point of time in the network.

Theorem 4.1 (Mutual Exclusion)**.** *There is at most one privileged node at any point of time in the network.*

Proof. Assume a $ReqM_{sm}$ in transit on a link, then the link is not directed toward any of the two entities connected by it. More precisely, we denote by transit ($M_{sm;u}$, $M_{sm;v}$) the set of messages in transit from u to neighbour v at time t.

- **Property 1**: If $ReqM_{sm} \in$ transit ($M_{sm;u}$, $M_{sm;v}$)[t], then last ($M_{sm;u}$)[t] $\neq M_{sm;v}$ and last ($M_{sm;v}$)[t] $\neq M_{sm;u}$. The property trivially holds as there are no requests in transit.

If an M_{mp}, fails to return the token to its head M_{mp} within predefined time due to link-failure, the nodes token session is cancelled and the token is considered lost. The respective cluster head will regenerate the token with all details prior its submission to the failed node. This guarantees mutual exclusiveness. □

4.4.2 ME2- Deadlock

A deadloack may occur when the token is free, and one or more $\mathcal{TA}$ s want to enter the auction market but are not able to do so. This happens when: the token cannot be transferred to a node because no node holds the privilege; the node in possession of the token is unaware that there are other nodes requiring the privilege; or the token does not reach the requesting unprivileged node. The logical pattern established using $POINTER$ variables ensures a node that needs the token sends $ReqM_{sm}$ either to $M_{sm;v}$ holding the token or to $M_{sm;u}$ that has a path to the node holding the token [28]. In addition, M_{mp} nodes send $ReqM_{mp}$ to their M_{sm}. Thus, this can never occur. Lets consider the orientation of tree links formed by the M_{sm} nodes, say L at time *t* the resulting in a directed graph. In all cases, there are no directed cycles.

- **Property 2:** $L[t]$ is acyclic.
- **Property 3:** In $L[t]$, from any non-terminal node there is a directed path to exactly one terminal entity.

Theorem 4.2 (Deadlock Free at M_{sm})**.** *From properties 1-3, in $L[t]$ any terminal path leads to the entity holding the token.*

Proof. Within finite time, we consider every message will stop travelling at a M_{sm}. Let us call target $(M_{sm;v})[t]$ the terminal node at the end of the terminal path of $M_{sm}v$ at time t; if a $ReqM_{sm}$ is travelling from $M_{sm;u}$ to $M_{sm;v}$ at time t, then the target of the token message is target $M_{sm;v}$ $[t]$. □

Theorem 4.3 (Deadlock Free at M_{mp})**.** *The M_{sm} entity holding the token will service only M_{mp} entities with $ReqM_{mp}$ at the time of token arrival.*

Proof. Each token receipt by $M_{sm;v}$ results in the *RQ* entries to be transferred to the *GQ*. This mechanism ensures all $ReqM_{mp}$ from $M_{sm;v}$ after the token is received will be considered for next token round. The token is then distributed to the specific M_{mp} in the *GQ* FIFO list. When the *GQ* is empty, the token is passed on to another $M_{sm;w}$. Now, using proof by assertions on the responsible loop (Algorithm 4.5,line 6) we show it terminates.

- *Precondition:* n $\leftarrow$ number of $ReqM_{sm}$ in *GQ*
- *Invariant:* $n \neq 0$
- *Postcondition:* all n requests receive the token after $n+1$ iterations. This is $\mathcal{O}(n)$
- *Initialization:* $True$ in the first iteration of the loop $n = 0$ $P(n) = n{+}1$; then $P(0) = 0{+}1 = 1$ where $n \neq 0$ is $True$
- *Maintenance:* Assume $P(k)$ is True for some k
  $P(k) = k + 1$; then by induction we prove $P(k + 1)$ is $True$

$P(k+1) = (k+1)+1$

$P(k+1) = k+2$ where (k + 2) is $True$

- *Termination:* occurs when *GQ* is empty (no requests in GQ)

Claim: *From above proof every request will be delivered to a target M_{sm} and M_{mp}.*

□

4.4.3 ME3- Starvation

If $M_{sm;u}$ holds the token and another node $M_{sm}v$ requests for the token, the identity of $M_{sm}v$ or of proxy nodes for $M_{sm;v}$ will be present in the *RQ1*s of various M_{sm}nodes in the path connecting the requesting node to the currently token-holding node. Thus depending on the position of the node $M_{sm;v}$ requests in those $RQ1$ s, $M_{sm;v}$ will sooner or later receive the privilege. In addition, enqueuing $ReqM_{mp}$ requests in $RQ2$ ensures that a token gets released to the next M_{sm} at the head of the $RQ1$ once the GQ is empty, no single cluster of M_{mp} nodes continues to trade while other nodes are starved.

4.4.4 ME4- Fairness

Each $\mathcal{TA}$ is endowed with the same opportunity to submit an offer in the auction market. First, the *FIFO* queues *RQ1* and *RQ2* are serviced in the order of arrival of the requests from nodes. A new request from an agent that acquired the token in the recent past is enqueued behind the remaining pending requests. This condition holds given the assumption that latency does not affect delivery of requests from neighbouring M_{sm} nodes.

4.5 Algorithm Performance

Due to the complex nature of MUTEX algorithms, it is very difficult to mathematically analyse their performance [117]. Gravey and Dupis in [118] analyse the performance of two mutual exclusion algorithms for a distributed system consisting of only two sites. From their observations there argue that when number of sites is more than two, analytic performance study of mutual exclusion algorithms becomes intractable, due to the rapid growth of state space of the underlying Markov chain. However, in limiting cases (“low traffic" or “heavy traffic" of CS requests), it is possible to analyse the performance of mutual exclusion algorithms [119]. In this section, we analyse the performance of the proposed algorithm for limiting cases. We employ two metrics that are generally used for measuring performance of mutual exclusion algorithms namely *time complexity* and *message complexity*.

4.5.1 Message complexity:

Assuming the routing, spanning tree and cluster forming algorithms are handled by some protocols separate from our CDA algorithm, their complexities are not considered in our analysis. We employed a token based algorithm with cluster heads forming a logical spanning tree overlaying a physical hierarchically clustered network. Since the tree structure is logically imposed upon underlying network, the pathological cases (e.g. chain topology) where the diameter is not $\mathcal{O}(\log N)$ can be avoided in favour of trees which approximate a radiating star topology (see [28]. Thus, under light demand, our algorithm exchanges $\mathcal{O}(\log N)$ messages per market auction execution. In the *Worst Case*, the token requesting M_{sm} and token holding M_{sm} will be on the farthest sides in the worst situation. In such a case, the message complexity for a request is $(N-1)+(N-1)$ = $\mathcal{O}(2N-2)$, resulting in $\mathcal{O}(N)$. Under heavy load, we expect that as the number of nodes requesting for the privilege increases, the number of messages exchanged per access of token decreases substantially (refer to [54]). Thus, in heavy load the algorithm requires the exchange of only four messages per market auction execution.

4.5.2 Time Complexity:

We consider the input to be the number of agents participating in the CDA. Operation in *LocalMarketExecution* is dominant and will be executed *n* times. We expect our CDA to run in linear time complexity $\mathcal{O}(N)$.

4.6 Token Handling

We employ a protocol using the notion of logical time to detect and recover from token loss —*Token handling*. Unlike other approaches, our approach results in the integration of token handling within the CDA algorithm itself on conception. Thus, we eliminate the need for expensive election protocols that are generally used in token-based algorithms to regenerate lost tokens. Such an approach has since been proposed by Agrawal and Elabbadi [120]. To avoid drowning the contributions of this chapter we only discuss token loss due to M_{mp} link and node failure. In Chapter 5, an in-depth investigation of M_{sm} node and M_{sm}-M_{sm} link failures is done as part of the fault cases presented in Section 2.3.2. The token-handling specifications of our initial algorithm address the following cases:

A link breaks between M_{sm} —$M_{mp}i$, while $M_{mp}i$ is in possession of the token: Frequent link failure is expected between the M_{sm} and the ever mobile M_{mp} nodes. The duration of the disconnections is varied and within a system that performs close to real-time, node reconnection

and message resubmission may not be an ideal solution to address the problem. Such network link failures result in reduced performance of the overall system. The $M_{mp}i$ node temporarily relaying the token through a close-by same-cluster $M_{mp}i$ node with a better connectivity to M_{sm}. This suggestion aims to address transient and to some extent intermittent faults due to link failure, thereby reducing chances of token loss.

An $M_{mp}i$ node completely fails to pass the token back to the proxy M_{sm}: i.e node is completely disconnected or takes long to complete its execution in the auction. It possible that the M_{mpi} may disconnect as a result of battery depletion, sleep (doze) mode activation, or being switched off by user. Loss of a single M_{mpi} has negligible impact on the overall functionality of CDA, although it may disadvantage the $\mathcal{TA}$ residing on the node from participating in the auction. The M_{mp} nodes' failure results in reduced performance as system resources are channelled towards token regeneration. In order to prevent the occurrence of node disconnections the algorithm in [28] prompts an M_{mp} requesting the token to disable the sleep mode functionality until it has executed in the auction market. In addition, timing checks are used to ensure that an M_{mpi} node does not hold on to the token indefinitely as a result of faults or disconnections. Hence, the token is destroyed at the M_{mp} node and a copy is regenerated at M_{sm} once the expected time of execution (considering network latency) has elapsed.

4.7 Summary

This chapter presented a decentralised CDA scheme for an energy distribution within resource limited, smart micro-grids. This CDA captures i) implementation simplicity, ii) robustness through token handling and iii) efficiency in minimising messages exchanged. These attributes are vital in maintaining stability in critical information systems for energy distribution. Trading agents require mutually exclusive permission to submit an offer in the auction market. We cast the CDA protocol as a mutual exclusion problem in a hierarchical clustered network model. The algorithm's correctness and efficiency were presented. We found the time complexity for a single agent to trade in the auction market is $\mathcal{O}(N)$ and message complexity is $\mathcal{O}(\log N)$. These are reasonable upper bounds if we assume node or link do not fail within the smart meter tier of the hierarchical network. These results indicate the scalability of our solution while addressing, in part, the concerns of centralisation. We are aware that, the single token is equally vulnerable to loss due to component malfunctions and probable user misbehaviour. To address this concern we integrate a preliminary token handling protocol to allow resiliency of our solution. Our proposed decentralised CDA algorithm is deployable in islanded, resource-constrained micro-grids operating with no centralised controller. In general, the algorithm should run on micro-grids supported by a hierarchical network topology, where households

form cluster nodes around a single smart meter-cluster head. Importantly, our decentralised CDA algorithm does not capture power transmission losses explicitly. The assumption herein, is that when the bid is submitted into the market, the clearing rule will cater for this phenomenon. An interesting direction for future work may entail relaxing the assumption on M_{mp} mobility; where each M_{mp} is allowed to rejoin the network in another cluster. An interesting challenge is catering for mobility, while effectively minimising communication and performance overheads. In the next chapter, we attempt to make the CDA scheme resilient to link and node failure. These are crucial attributes to consider when aiming to achieve stability of the micro-grid.

Chapter 5

Resilient, Fault Tolerant Continuous Double Auctioning

> Left to themselves, things tend to go from bad to worse.
>
> MURHPY'S COROLLARY.

5.1 Overview

One major drawback of the token-based approach we employ to support our decentralised CDA algorithm for serialising market access (Section 4.3.3), is resilience to failure [82] [83] [84]. Due to this realisation, we include some fault prevention and to a lesser extent fault tolerance as part of the token-handling specification (in Section 4.6) of our initially proposed CDA algorithm. However, the decentralised CDA algorithm fails to guarantee continuous trading agents participation in the event of crash-fail or Byzantine faults (classified in Section 2.3.2). In this chapter, we address the aforementioned challenges that disrupt auctioning and inevitably leads to users buying out. In Section 5.2 we consider additional assumptions, and then discuss the fault models chosen from fault scenarios described in section 2.3.2. While a myriad of faults can manifest on the proposed decentralised CDA, we narrow the scope of our work to crash-fail (5.3.1) and Byzantine (5.3.2) faults. First, we look at providing fault tolerance to crash-fail faults in Section 5.4. In Section 5.4.1 describes data structures and control messages the algorithm extensions use, while Section 5.4.2 details the procedures and routines. We present the correctness (Section 5.4.3.1) and efficiency (Section 5.4.3.2) analyses of the fault tolerant algorithm extensions to handle crash-fail faults. Second we look at fault tolerance to

Byzantine faults (Section 5.5). Section 5.7 summaries the contributions of this chapter in relation to the overall thesis.

5.2 Assumptions

In section 5.3 we specify the fault classes that guide the formulation of our fault models in this chapter. In addition to the assumptions made in sections 2.2.3, 2.2.4.1 and 5.3, we continue working on the notion that each node continues to communicate with its direct neighbours only, similar to Raymond's MUTEX [54].

5.3 Fault Models

In this section, we discuss the Crash-fail and Byzantine Faults introduced in Section 2.3.2.

5.3.1 Crash-Fail Faults

A crash failure occurs when a process prematurely halts, but was working correctly until it stopped. We consider a typical crash failure of the M_{sm} nodes, where once the node has stopped, nothing is heard from it anymore. Communication through M_{sm} is blocked; hence, an auction algorithm does not progress but instead will be held up until the node is recovered. We consider two cases that can arise when an M_{sm} node crashes: (i) in possession of the token or (ii) not in possession.

Case A: If a token possessing M_{sm} node fails

If the CDA algorithm is to be successful it needs to ensure recovery from failure of the M_{sm} site holding the token and all its neighbours. Failure of the token possessing M_{sm} node means, the dependent child M_{mp} nodes connected to it will not access the auction market. In addition, M_{sm} failure can partition the network, which undermines the MUTEX properties and disrupts the functioning of the CDA. We consider the following cases:

- M_{sm} fails with the token in M_{mp}: Under this condition, the token will be trapped in the M_{mp} as it waits indefinitely to be delivered to the M_{sm}. Deadlock and starvation occurs, where other participating M_{mp} nodes wait indefinitely for an inaccessible token.

- M_{sm} fails with the token in its possession: Assuming an M_{sm} node fails and is bypassed by a protocol which prompts a neighbouring M_{sm} node to regenerate the token, resumption of the faulty M_{sm} node may reintroduce the old token, which was kept alive. More than one token is in the network. Furthermore, if an M_{sm} node fails and is bypassed, it loses the opportunity to distribute the token to the rest of its child nodes, and if the failure is prolonged or permanent, starvation of the child M_{mp} nodes will occur.
- The token is in transit between M_{mp} and M_{sm}: In this thesis, we assume that, all communication occurs with no message loss. Hence, we consider that a sent token will be received.
- A token requesting M_{sm} fails: A token requesting M_{sm} is an M_{sm} node waiting on the token or is in the token request path. In this case, if the faulty node were to be used for forwarding the token request, the path would have to be re-established. Failure of such a node results in partitioning of the network, leading to similar impacts stated in the previous case.

Case B: If a token requesting M_{sm} node fails

Failure of a requesting M_{sm} might have less serious, but crucial repercussions compared to those in token possessing M_{sm} failures.

5.3.2 Byzantine Faults

We consider general Byzantine faults where a faulty node(s) exhibit arbitrary behaviour. In particular we consider that a faulty node may corrupt its local state and send arbitrary messages, including specific messages aimed at subverting the system. We investigate Byzantine faults resulting from M_{mp} nodes passing arbitrary request messages to the M_{sm} nodes. The following subsections detail two Byzantine fault cases we aim to provide tolerance against.

Case A: A single M_{mp} sends arbitrary, multiple continuous token requests

The current protocol allows cluster M_{mp} nodes to send a request to the cluster-head M_{sm} node when ever a $\mathcal{TA}$ wants to participate in the auction. The cluster head M_{sm} will in turn send a request to the token-holding or neighbouring M_{sm} which is in the path to the token holding M_{sm}. Thereafter, all Req_{mp} requests from the cluster M_{mp} nodes will be queued in the *GQ* of the cluster head M_{sm} till the token is received. However, say a faulty node(s) was to send continuous multiple requests to the cluster node within an arbitrary time frame, the current mechanism would allow these requests to be enqueued in *GQ*. This assumes the multiple

requests do not in anyway overwhelm the cluster head M_{sm}, leading to a crash. On arrival of the token the *GQ* entries are "locked in" and copied into the *RQ2*. Every M_{mp} request in the *RQ2* is serviced with the token in *FIFO* order. This implies, all the multiple requests belonging to a single, faulty M_{mp} will receive the token multiple times, with respect to the number of its requests present in *RQ2*. If the number of the multiple requests in *RQ2* is significantly large starvation maybe realised. The token will be trapped within a loop servicing the same nodes within one cluster for an arbitrarily long time. The main impacts include:

- The M_{sm} nodes that have requested for the token may wait a finitely longer time than is favourable to the system as there is no mechanism to ensure such starvation or delay is avoided. Even if we assume the system has backup $M2_{sm}$ nodes; starvation or delay is not avoided, despite the main component being active. Fair trade is compromised as trading agents residing outside the cluster (where the token is "trapped") are deprived of a fair opportunity to trade. Market efficiency is expected to be significantly reduced, as a fault agent (at times, unwilling to participate), can constantly submit invalid offers. This phenomenon, portrays a denial of service attack, if the adversary makes the token inaccessible for agents, in other clusters. The token is trapped in a continuous loop at a single M_{sm} node. User trust and therefore participation will be disrupted as services are rendered unavailable.
- However, these measures do not guarantee fair token access to all the cluster nodes. Say, a single, malicious M_{mp} cluster node sends multiple requests to the M_{sm} without giving other M_{mp} nodes opportunity to submit a request before the limit of requests is reached. This makes the token inaccessible to the victim traders, as there are disallowed from submitting a request.

Case B: Multiple M_{mp} send arbitrary, multiple continuous token requests:

In this case, we assume, that more than one M_{mp} is faulty. The faulty nodes are not under any form of coördination, but may simultaneously send multiple requests continuously to their M_{sm} cluster head node. Assuming the arbitrary requests do not overwhelm the head cluster M_{sm} node leading to a crash; a significant number of requests will be enqueued within the GQ before arrival of the token. As with the previous scenario, when multiple nodes send multiple requests delay and therefore starvation is inevitable for a long while.

5.4 Crash-Fail Fault Tolerance Approach

We adopt redundancy, to ensure preservation of M_{sm} nodes functionality with $M2_{sm}$, which take over in the event of a crash-fail faults. This ensures, $\mathcal{TA}$s hosted by M_{mp} nodes are

guaranteed to continue transacting even in the even of the primary M_{sm} failure. Employing a primary backup handles Byzantine failures poorly, because there is usually no check routine to make sure the primary is functioning correctly; which is an obvious downside. More so, the backup system must always be in sync with the main system to guarantee a successful take over of the main systems' tasks. Recovering from a node failure is time consuming and complex.

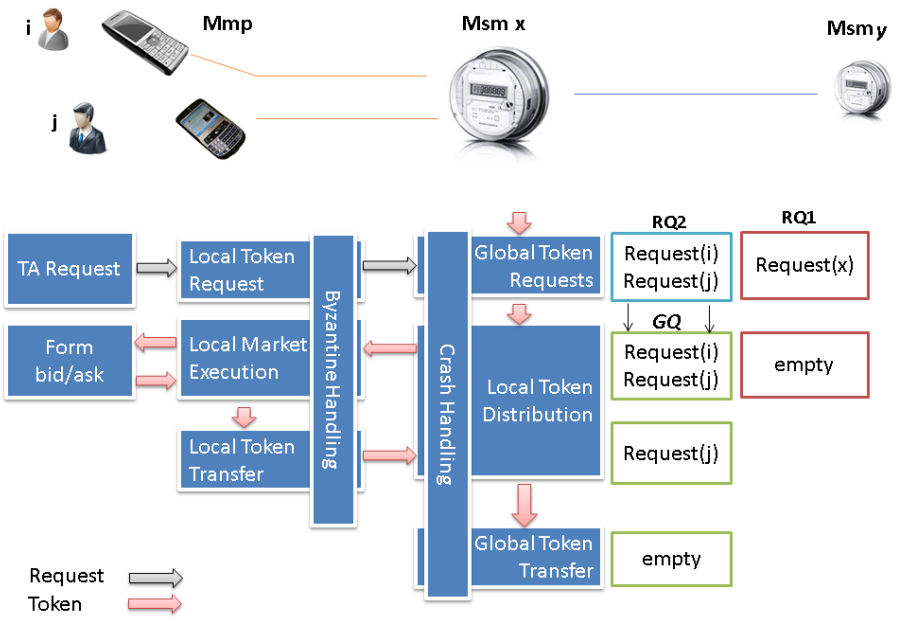

FIGURE 5.1: Crash-Fail and Byzantine Fault Tolerance CDA Procedures

5.4.1 Data Structures & Control Messages

The fault tolerance mechanism is an extension to the initial proposed CDA scheme. Thus, the algorithm continues to use the same control messages and data structures described in 4.3.4.

5.4.2 Procedures and Routines

The M_{sm} and $M2_{sm}$ nodes execute the following routines:

- *GlobalTokenRequest*;
- *LocalTokenDistribution*;
- *GlobalTokenTransfer*

The $M2_{sm}$ nodes also execute an additional routine called

- *CrashFailureHandling*

Similarly, M_{mp} nodes execute the following routines:

- *LocalTokenRequest*;
- *LocalMarketExecution*;
- *LocalTokenTransfer*

The routines executed by the M_{mp} node, do not change after introduction of the fault tolerance protocol, thus we shall only discuss the procedures executed by the M_{sm} and $M2_{sm}$ nodes. We refer the reader to Section 4.3.5 for an understanding of M_{mp} procedure description.

Global Token Request: If we consider an abstraction of the CDA model, the *GlobalTokenRequest()* routine will be the initial procedure that allows an M_{sm} node to send a request to participate in the token exchange. Once a Req_{sm} is sent to the neighbour node holding the token or in path to the token, $TokenAsked$ which is a boolean variable is set to *TRUE*. This avoids forwarding of similar request messages to the same token holder. Two subroutines or methods namely $EnqueueRequest()$ and $Nodebackup()$ are executed while M_{sm} is waiting for the token. $EnqueueRequest()$ enqueues the Req_{sm} and Req_{mp} requests into *RQ1* and *RQ2* respectively, while $Nodebackup()$ creates a checkpoint by sending updates to the $M2_{sm}$ node.

Algorithm 5.1: $GlobalTokenRequest$ Procedure

Input : $TokenRecieved$, $ReqM_{sm}$, $ReqM_{mp}$, $TOKEN$, $M_{sm,i}[.]$.
Output: $FlagM_{mp}$, $TokenAsked$.

```
1  for each M_sm ∈ M_sm,i[.] do
2      if FlagM_sm = FALSE AND RQ1 ≠ 0 AND TokenAsked = 0 then
3          Send ReqM_sm to M_sm in POINTER ;
4          Set TokenAsked to TRUE ;
5          Assign TokenReceived = FALSE ;
6          repeat
7              EnqueueRequest(ReqM_sm, ReqM_mp) ;   // Only 𝒟 ReqM_mp are enqueued RQ2
                during time Q
8              NodeBackup() ;                       // Function runs a backup M_sm,i
9          until TokenReceived = FALSE;
10     else
11         EnqueueRequest(ReqM_sm, ReqM_mp) ;
12         NodeBackup() ;
```

Local Token Distribution: Once the token is received the algorithm moves to the *LocalTokenDistribution* routine. Receipt of the token by the requesting M_{sm} node sets the boolean variable $FlagM_{sm}$ to *TRUE* indicating possession of the token. *GQ* is a FIFO queue that stores a copy of requests submitted and "locked in" at the arrival of the token to the cluster head. Once a token is received all requests in *RQ2* are transferred to *GQ2*. While there are still requests in *GQ*,

the algorithm will run the $EnqueueRequest()$ and $BackupNode()$ subroutines before sending them to an M_{mp} node with a request at the head of the *GQ*. After token is allocated to an M_{mp} node, the nodes corresponding Req_{mp} entry is removed from *GQ*. The M_{sm} node then waits for a predefined time for the return of the token. To ensure that M_{sm} does not wait for the token for an undefined time, the routine will consider a time bound on the wait period. Failure of the M_{mp} nodes to return the token within the predefined time results in the token degradation (an M_{mp}) and regenerated (at M_{sm}) from the last known backup.

Algorithm 5.2: $LocalTokenDistribution$ Procedure

Input : $TokenRecieved$, $ReqM_{sm}$, $ReqM_{mp}$, $TOKEN$, $M_{sm,i}[.]$.
Output: $FlagM_{mp}$, R, $TokenCounter$ $TokenOB$.

```
1  for each M_sm ∈ M_sm,i[.] do
2      if TokenRecieved = TRUE AND self ReqM_sm is at head then
3          Set FlagM_sm to TRUE ;
4          GQ ← RQ2 ;
5          n ← number of GQ entries ;
6          repeat
7              NodeBackup() ;                          // Function runs a backup of M_sm,i
8              if M_mp at head ≠ disconnected then
9                  Remove request entry in GQ ;
10                 Assign token to M_mp ;
11                 if TokenReturn = TRUE then
12                     Move to next GQ entry ;
13                 else
14                     TokenReturn = timed-out ;        // When TOKEN return times-out
15                     Remove entry in GQ ;
16                     TokenRegenerate(BackupID) ;
17             else
18                 Cancel request entry in GQ ;
19                 Move to next GQ entry ;
20             TokenRequest(ReqM_sm, ReqM_mp) ;  // Function enqueues requests into RQ1
                and RQ2
21             n − − ;
22         until n < 1;
23     else
24         Wait for TOKEN ;
25         TokenRequest(ReqM_sm, ReqM_mp) ;
```

Global Token Transfer: A token possessing M_{sm} node, will send the token if it has a non-empty *RQ1*, where Req_{sm} at head of the *RQ1* is not its request. The boolean $FlagM_{sm}$, is set to $FALSE$ and the token is sent to the respective M_{sm}s with a Req_{sm} at the head of *RQ1*.

Crash Failure Handling: This routine is executed at the $M2_{sm}$ node. There are four general activities defined by [15], that systems employing fault tolerance have to perform: error detection,

Algorithm 5.3: $GlobalTokenTransfer$ Procedure

Input : $TokenRecieved$, $ReqM_{sm}$, $ReqM_{mp}$, $TOKEN$, $M_{sm,i}[.]$.
Output: $FlagM_{sm}$, R, $TOKEN$.

for *each* $M_{sm} \in M_{sm,i}[.]$ **do**
  if $FlagMsm = TRUE\ AND\ RQ1 \neq 0\ AND\ ReqM_{sm}! =' self'$ **then**
    Set $FlagM_{sm}$ to $FALSE$;
    Send $TOKEN$ to $ReqM_{sm}$ at $RQ1$ head ;
Return $TOKEN$;

damage confinement, error recovery and fault treatment, and continued system service. The considerations have been factored into the new fault tolerant protocol as shown in Algorithm 5.4.

Algorithm 5.4: $CrashFailureHandling$ Procedure

Input : $TokenRecieved$, $ReqM_{sm}$, $ReqM_{mp}$, $TOKEN$, $M_{sm,i}[.]$.
Output: $FlagM_{sm}$, R, $TOKEN$.

for *each* $M2_{sm} \in M2_{sm,i}[.]$ **do**
  if $Timeout = TRUE\ OR\ N_Query = TRUE$ **then**
    Ping M_{sm} with Enquiry() msg ;
    if M_{sm} *respond* = *TRUE* **then**
      M_{sm} is alive ;
      Wait for $NodeBackup()$;
    else
      FailureContainment() ;
      Resume M_{sm} operations ;
  Return $TOKEN$;

- *Error Detection:* This phase deduces the presence of faults by detecting an error in the state of some subsystem. It is from the errors that failures and faults are deduced. Unless errors are detected successfully by a fault model, the fault model is somewhat useless. We consider that detection of M_{sm} node failure is done by use of timing checks. Timing checks typically use a timer with a value predetermined from the specifications of the component. If a node does not respond within the pre-specified time period, its behaviour is declared erroneous and the node is assumed to have failed. In our context if the standby node $M2_{sm}$ does not receive a backup message from the M_{sm} node with respect to the timer clock, it will send a “query" message to the M_{sm}. If no response is received, it is assumed the M_{sm} has failed which triggers $M2_{sm}$ to enter the recovery phase.
  We consider that, each $M2_{sm}$ carries a failure detector, to try to detect crashed M_{sm}. A failure detector is called strongly accurate if only crashed processes are ever suspected [121]. We understand, in bounded delay networks, a strongly accurate (and complete) failure detector is implemented as follows: Suppose l_{max} is a known upper bound on network latency from M_{sm} to $M2_{sm}$. Each M_{sm} thus broadcasts an “alive" message every

ν time units. Each M_{sm} from which no message is received for $\nu + l_{max}$ time units is suspected to have crashed. An inquiry message is then send to confirm this suspicion (Algorithm 5.4 line 3).

- *Damage Confinement:* The system design incorporates mechanisms to limit the spreading of errors in the system, thereby confining the damage to predetermined boundaries (Algorithm 5.4 line 8). In order to prevent the spread of error, the faulty node is temporarily suspended, while the standby component takes over. We assume, the standby component has dissimilar properties to the primary component, and susceptibility to error.

- *Error Recovery:* The two general techniques for error recovery include backward and forward recovery [15]. We chose backward recovery, where check-pointing is done often on the standby $M2_{sm}$ node, and in the event of a failure a system roll-back occurs. One main drawback of the backward technique is on the incurred overheads. Assuming the $M2_{sm}$ nodes have a fairly stable storage, frequent check-pointing is required, which affects normal system operations, despite failure absence. Despite the high overheads, we opt for backward recovery due to its simplicity. Backward recovery is independent of the fault or failure [15], thus, recovery from arbitrary faults, whether transient or permanent us possible. We assume, check-pointing in our case invokes the $NodeBackup()$ procedure everytime a token is returned by M_{mp} to M_{sm} (Algorithm 5.2 line 7), or when a $ReqM_{mp}$ or a $ReqM_{sm}$ is received (Algorithm 5.1 line 8, 11). To reduce the overhead, one message (that includes M_{sm} current data structures) is sent to the $M2_{sm}$ if M_{sm} is in possession of the token and is check-pointing. We assume a roll-back procedure is executed at the $M2_{sm}$ node with regards to the last checkpoint.

- *Fault Treatment and Continued Service*: The fault or faulty component has to be identified, and its use is suspended through a fault containment method (line 108). By assuming the main component's role the backup $M2_{sm}$ node masks primary M_{sm} node failure, isolating the faulty component but maintaining system availability and reliability. The $M2_{sm}$ re-establishes connection with M_{mp} nodes the resumes communication with the neighbouring M_{sm} nodes (Algorithm 5.4 line 9).

5.4.3 Algorithm Analysis

Fault scenarios are used to investigate if the modified fault-tolerance algorithm maintains the crucial mutual exclusion properties despite the occurrence of faults. Furthermore, we discuss the message and time complexity of the extended algorithm.

5.4.3.1 Correctness of the Algorithm

We analyse how the fault tolerant CDA algorithm guarantees the following properties in the presence of potential fault, thus, failure:

- **ME1- Mutual Exclusion**:
  If *token handling* is diligently implemented, only one token will be in the network at any particular time, despite the presence of faults. The proposed algorithm ensures that at any instant of time, not more than one node holds the token. In the *LocalTokenDistribution* procedure 5.2 whenever an M_{sm} receives a token, it becomes exclusively privileged: (line 3) by setting the $FlagM_{sm}$ to TRUE. Similarly, in the *GlobalTokenTransfer* procedure 4.6 when a node sends the token, it becomes unprivileged (line 3) by setting the $FlagM_{sm}$ to FALSE.
  Between the instants, when one node becomes unprivileged and another node becomes privileged, there is no privileged node. Thus, there is one privileged node at any time in the network. If M_{sm} granting the token fails then a privileged cluster node M_{mp} will not return the token within a predefined time. The token session is cancelled and the token is considered to be lost. The same M_{mp} node, will revert changes to a time before the token is received, and the timed-out token is destroyed, before a token regeneration procedure in the $M2_{sm}$ node. If a privileged M_{sm} node, finds itself without outgoing links to the neighbouring M_{sm} nodes, and time has elapsed, it will destroy the copy of the token, before failure on reboot or resumption of services. The respective $M2_{sm}$ backup node, will regenerate the token with all details with respect to the last checkpoint. Thus, only one token is in the system, guaranteeing mutual exclusiveness.

- **ME2- Deadlock** : When the token is free, and one or more $\mathcal{TA}$ s wants to enter the auction market, unsuccessfully, a deadlock can occur. This happens due to:
  - the token not being able to be transferred to a node because no node holds the privilege,
  - the node in possession of the token being unaware that there are other nodes requiring the privilege, or
  - the token failing to reach a requesting unprivileged node.

  We understand, the logical pattern established using POINTER variables, ensures a node that needs the token, sends $ReqM_{sm}$ either to $M_{sm}v$ holding the token or to $M_{sm}u$ a neighbouring node in the token holder path. M_{mp} nodes send $ReqM_{mp}$ to their cluster head M_{sm} node. If we consider the orientation of the tree links formed by the M_{sm} nodes, say L at time t the resulting directed graph. In all cases, there are no directed cycles, making $L[t]$ acyclic, where from any non-terminal node there is a directed path to exactly one terminal entity. Within finite time, we consider every message will stop travelling at a

M_{sm}. Thus, in the absence of failure, the conditions (i-iii) will not occur. If a privileged node fails our algorithm ensures that the copy in the failed node is deleted while a copy is regenerated from the last checkpoint. By establishing connection with the neighbouring M_{sm} nodes and M_{mp} nodes, previously connected to the primary component, the token path is complete and the $M2_{sm}$ will pass on the token.

- **ME3- Starvation** If $M_{sm}u$ holds the token and another node $M_{sm}v$ requests for the token, the identity of $M_{sm}v$, or of proxy nodes to $M_{sm}v$ will be present in the *RQ1*s of M_{sm}nodes in the path to the token-holding node. Thus, depending on $M_{sm}v$s requests' position in the *RQ1*s, $M_{sm}v$ will sooner or later receive the privilege. In addition, enqueuing $ReqM_{mp}$ requests in $RQ2$, ensures that a token is released to the next M_{sm} at the head of the $RQ1$, once the GQ is empty (line 22), no single cluster of M_{mp} nodes continues to trade while other nodes are starved. In the presence of failure the static spanning tree may be partitioned, which means, a sub-tree that has the token, will benefit, while the other sub-tree(s) is starved. Our algorithm ensures that partitioning does not occur with a backup node taking the place of the failed node.

- **ME4- Fairness**: Every trading agent has the same opportunity to enter the auction market. The *FIFO* queues, *RQ1* and *RQ2*, are serviced according to request arrival order. A new request from an agent that acquired the token in the recent past is enqueued behind the remaining pending requests. If a privileged M_{sm} node fails, while granting the local cluster M_{mp} nodes a opportunity to participate in the auction, our algorithm ensures the remaining M_{mp} nodes get a opportunity to receive the token in request arrival order. Allowing the backup $M2_{sm}$ to resume the roles of the failed M_{sm}, instead of total elimination, guarantees continued, fair token distribution.

5.4.3.2 Performance Analysis

Two metrics used for measuring performance of mutual exclusion algorithms are time complexity and message complexity.

Runtime of the Algorithm: It usually suffices to identify a dominant operation and estimate the number of times it is executed. In the CDA Algorithm, we consider $LocalTokenDistribution$ procedure 5.2, executed on the M_{sm}, as the most costly operation. As a dominant operation, we regard the while loop in line 6. The run time depends linearly on the input size n.

Time Complexity: An analysis of the run time of our fault tolerant CDA algorithm, is not sufficient, as our interests are on investigating the running time increase, when the input

size increases. Thus, we consider the *order of growth* measure, which is estimated by taking the dominant term of the running time expression. The dominant operation in Algorithm 1 is influenced by N which is the number of cluster nodes. As N grows the time complexity increases linearly at an order of $\mathcal{O}(N)$.

Message complexity: Assuming $V = \sum v_1 + v_2 ... + v_n$ is the number of Req_{sm} received by a non-token possessing node $M_{sm}i$ from its neighbours because it is in line to a token holding node. While $U = \sum u_1 + u_2 + ... + u_n$ is the number Req_{mp} received by $M_{sm}i$. Each time a $BackupNode()$ is executed, W, a summation of u and v, is saved in the stable $M2_{sm}$ storage. In a worst case scenario, all the n cluster nodes will send a Rep_{mp} at random times while all child M_{sm} nodes in a K-ary tree. The result is a message complexity of order $\mathcal{O}((\log N) + W)$.

5.5 Byzantine Fault Tolerance Approach

The algorithm continues to use the same control messages and data structures described in 4.3.4 and procedures (routines) described in 5.4.2

5.5.1 Procedures and Routines

The routines executed by the M_{sm} node do change after introduction of the Byzantine fault tolerance protocol. Since the algorithm should tolerate multiple requests coming from a faulty M_{mp} our focus lies mainly on the *Global Token Request* procedure executed by the M_{sm} nodes and $M2_{sm}$ nodes. As before, we refer the reader to Section 4.3.5 and Section 5.4.2 for an understanding of the initial and the fault tolerance enabled procedures respectively.

Global Token Request: A subroutine (method) named $EnqueueRequest(Req_{sm}, Req_{mp})$, is executed when M_{sm} is waiting for the token. This method will enqueue the Req_{sm} and Req_{mp} requests into *RQ1* and *RQ2* respectively, while $Nodebackup()$ creates a checkpoint of updates to the $M2_{sm}$ node. In the event that a faulty M_mp starts sending continuous Req_{mp}'s, an extension allowing only one Req_{mp} from a number of $\mathcal{J}$ M_mp's to be enqueued into $RQ2$, within a time period of Q, while $TokenRecieved$ is FALSE.

Byzantine Fault Handling: To recall, [15] defines four general activities that systems employing fault tolerance have to perform. These include: error detection, damage confinement, error recovery and fault treatment and continued system service.

Algorithm 5.5: $GlobalTokenRequest$ Procedure

Input : $TokenRecieved$, $ReqM_{sm}$, $ReqM_{mp}$, $TOKEN$, $M_{sm,i}[.]$.
Output: $FlagM_{mp}$, $TokenAsked$.

```
for each M_sm ∈ M_sm,i[.] do
    if FlagM_sm = FALSE AND RQ1 ≠ 0 AND TokenAsked = 0 then
        Send ReqM_sm to M_sm in POINTER ;
        Set TokenAsked to TRUE ;
        Assign TokenReceived = FALSE ;
        repeat
            EnqueueRequest(ReqM_sm, ReqM_mp) ;   // Only 𝒟 ReqM_mp are enqueued RQ2
             during time Q
            NodeBackup() ;                       // Function runs a backup M_sm,i
        until TokenReceived = FALSE;
    else
        EnqueueRequest(ReqM_sm, ReqM_mp) ;
        NodeBackup() ;
```

- *Error Detection:* If a node sends more than 1 request, during period Q, it is marked as a faulty node. Every M_{mp} is expected to send a single $ReqM_{mp}$. We consider that each M_{mp}'s identification is known, to the M_{sm} which can then uniquely identify the faulty node and inform the human participant to reset the M_{mp} or associated $\mathcal{TA}$.
- *Damage Confinement:* The responsible M_{sm} will drop the additional requests from any cluster M_{mp} or extra requests beyond $\mathcal{J}$. This ensures, only one request from a single M_{mp} is serviced and the requests have an upper bound, $\mathcal{J}$, of the maximum requests that are acceptable when M_{sm} is waiting on the token.
- *Error Recovery:* After reset, the M_{mp} is expected to resume normal operations. If similar faults are detected again by the M_{sm}, the M_{mp} is suspended from further participation until the issue is addressed by a trusted system specialist.
- *Fault Treatment and Continued Service:* The CDA algorithm can proceed despite the occurrence of the aforementioned fault. The faulty M_{mp}, can receive the token only once after a time period Q, when its proxy M_{sm} has the token.

5.5.2 Performance Analysis

Time Complexity: An increase in the number of cluster head nodes is unaffected by the Byzantine fault tolerance. The dominant operation is still influenced by N which is the number of cluster nodes. As N grows the time complexity increases linearly at an order of $\mathcal{O}(N)$.

Message complexity: We consider the Byzantine fault tolerance extension to be an additional layer on top of the crash-fail protocol, thus, no additional messages are exchanged, apart from those already sent (see Section 5.4.3.2).

5.6 Overall Overheads Analysis

Overall Time Complexity The two fault tolerance protocols described, rely on the dominant operation, which is the number of cluster nodes. As N grows the time complexity increases linearly at an order of $\mathcal{O}(N)$.

Overall Message Complexity The crash-fail fault tolerance protocol has a message complexity of order $\mathcal{O}((\log N) + W)$, and no additional messages are expected from the Byzantine fault tolerance extension, since it runs on top of the crash-fail protocol. The overall message complexity remains as $\mathcal{O}((\log N) + W)$.

5.7 Summary

In this chapter, we integrate a fault tolerance mechanism to the decentralised CDA algorithm for computationally limited micro-grids. First we craft a fault tolerance protocol that addresses *crash-fail* faults of cluster-head nodes. Instead of by-passing the failed node(s) and reconnecting the remaining spanning tree segments as proposed in the literature, our approach ensures the algorithm masks such failure by employing a redundant node to each cluster head node. The fault tolerant CDA, affords trading agents, mutually exclusive permission to participate in the auction market, despite the presence of crash failures. We present the correctness and efficiency of the fault tolerant algorithm. The runtime of the algorithm is linearly dependent on the input size N. The time complexity for a single agent to trade is $\mathcal{O}(N)$, where N is number of cluster nodes. The message complexity is $\mathcal{O}((\log N) + W)$, where W is the number of check-pointing messages; and N is the number messages exchanged per critical section execution. These are reasonable upper bounds for a crash-fail, fault tolerant, supported CDA algorithm employing redundancy and check-pointing. The Byzantine fault tolerance extension we propose, does not incur additional messages. Instead, it makes the underlying fault tolerance protocol more efficient, by eliminating the additional faulty requests. However, note that the proposed mechanisms for token handling and fault tolerance, up till this point, fail to guarantee trader participation, if the agent's hosting M_{mp} device fails to connect to M_{sm}. In the next chapter, we propose a complementary consumption scheduling mechanism, that can run on a RCSMG, while ensuring users get fair power allocation.

Chapter 6

Scheduling Power Consumption

> Electricity is really just organised lightning.
>
> GEORGE CARLIN.

6.1 Overview

Fault tolerance alone can fail to guarantee the continued participation of the automated traders in the auction. This raises the problem of participants buying out of the power-sharing agreement due to unsatisfied power allocation. In this chapter, we consider a complimentary, decentralised consumption scheduling scheme, that provides an extra layer of power management within RCSMGs. Power consumption scheduling smoothens the demand profiles over time to avoid overloading. Thus, a scheduling algorithm can be used to distribute power cost-effectively, encouraging users to shift heavy consumption activities to off-peak periods. For example, a user could opt to use heavy power consuming appliances (e.g. a geyser) at off-peak instead of on-peak periods; with the added benefit of paying less per kilowatt consumed. In a case where auctioning fails, the consumption scheduling algorithm can offer a more enforcing power management option.

To recap, the literature indicates how a myriad of centralised demand management approaches in the literature may not be ideal in the RCSMGs context, as there are: computationally intensive; could raise serious privacy concerns due to centralisation [17], [18]; and there assume availability of security and a reliable network. The latter is unrealistic in computationally limited, micro-grids, where data is transmitted over insecure and unreliable networks. We propose a decentralised scheduling algorithm to address this problem. In Section 6.3 we formulate the

problem of scheduling power distribution on the RCSMG as a convex optimisation problem [19], [20]; with the goal of minimising the total power consumption while maximising on the social benefit [18] of power distribution on the grid. In Section 6.4, we study a decentralised electricity consumption scheduling algorithm based on alternating direction method of multipliers (ADMM) [88], which has been shown [89], [122], to be robust in solving optimisation problems in smart grid communication networks. The ADMM algorithm is presented in a decentralised (Section 6.5.1) and fully distributed way (Section 6.5.2). The ADMM algorithm enables users to report demands as aggregated and not single values, which addresses the privacy concern; while the distribution of computations across network devices, significantly reduces the computational strain. Section 6.6 concludes the chapter discussing integration possibilities of the scheduling scheme to the auctioning algorithm.

6.2 Additional System Model Assumptions

The reader is referred to section 3.4.2 for an overview of related work on power consumption scheduling and section 2.2 for an understanding of the RCSMG model. Besides the RCSMG architecture (inspired by [7]), we make more assumptions, to formulate the power consumption scheduling problem as a convex optimisation problem. The RCSMG architecture describes a combination of a power and communication network, interconnecting households and a utility (see 2.2).

6.2.1 Power Network Model

To recap, the utility is capable of: generating part of the electricity in the RCSMG; facilitating efficient exchange of energy in the community through an auction market; participating in a short-term wholesale market. In this particular chapter, the utility provider buys electricity from competing generators in the MG and then retails it to consumers (households).

6.2.2 Communication Network Model

The communication network can be represented by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ are sets of nodes and edges, respectively. An edge is denoted by $\{i, j\} = \{j, i\}$, and $\{i, j\} \in \mathcal{E}$ implying that nodes i and j communicate directly and thus can exchange their estimates. The nodes represent local controllers[1] (LC)s, at the household and the central central controller (CC) at the utility. At a given interval of time, the LC will collect a snapshot of energy consumption [29];

[1]Mobile phones M_{mp} can be used as the *LC*s (i.e. schedulers) that automatically control the distributed generations (DGs), and loads locally in the households

load demand; and/or generation of the household it is associated with and sends it to the utility. The utility analyses the data considering weather forecast of wind speed and solar radiation. This has been observed by [123] to be helpful in predicting the household load demand based on power consumption patterns.

We consider a continuous discrete-time model with a finite horizon, $\mathcal{T} = \{1, 2, ..., T\}$, where T is finite and divided into T equal intervals of Δt size. Further, an asynchronous ADMM is considered, with possibilities of information loss or delay during communications. The asynchronous ADMM uses the value from the previous transmission to substitute for missing information due to loss and/or delay.

6.2.3 Household Appliance Model

Assume the consumption estimate is provided by the household, and based on a power data model [30], the grid coordinator computes a consumption estimate for billing. Let $p_{a,h} = (p^1_{a,h}, ..., p^T_{a,h})$ be the power consumption scheduling vector of each appliance $a \in \mathcal{A}_h$ over $\mathcal{T}$, $p_{a,h} \in \mathbb{R}^T$, where $p^t_{a,h}$ the power consumption ; t the time slot for appliance $a \in \mathcal{A}_h$ in household h. The set of energy consumption schedules for all appliances in a household $h \in H$ at time horizon $\mathcal{T}$, is denoted as P_h where $P_h = p_{a,h}, \forall a \in \mathcal{A}_h$ and can be represented with a matrix of dimension $|\mathcal{A}_h| \times T$. The total load of a particular household $h \in H$ denoted P_h is the sum of four types of loads: resistive, inductive, non-linear and composite loads [30]. Formally, we let the total load of a household h at T time slot be denoted by $l_h = (l^1_h, ..., l^T_h)$; the total load L_h across all households at each time slot $t \in \mathcal{T}$ can be computed as: $L_t = \sum_{h \in H} l^t_h$

In cases of load scheduling, each category of appliances is studied according to the level of priority, the interruptibility during operation and the energy consumption. The latter represents consumption patterns over a fixed time interval.

6.3 Cost models and Problem Formulation

6.3.1 Cost functions

Assuming intermittent renewable energy can be predicated using a short-term prediction, the total generation capacity of the set of households $h \in H$ is represented by G_h. The household power generation over T time slots is given by $g_h = g^t_h + \epsilon^t_{g,h}, \forall t \in \mathcal{T}$, and it is constrained by ' $0 \leq g^t_h \leq G_h, \forall t \in \mathcal{T}, \forall h \in H_g$, where g^t_h denotes the distributed generation of the household $h \in H_g$ and $\epsilon^t_{g,h}$ is the prediction error considered with distribution $\mathcal{N}(0, \sigma^2_g)$

The power demand projected (Q_{MG}) for the power consumption scheduling horizon $\mathcal{T}$ is defined by

$$Q^t_{MG} = Q^t_{pg} + Q^t_G + Q^t_s, \tag{6.1}$$

where $Q^t_{pg} = \sum_{h \in H_g} g^t_h$ is the predicated renewable generation for the set of households $h \in H_g$, Q^t_G is the predicated utility generation and Q^t_s is the energy available on battery at the scheduling horizon.

Say the predicated generation of household $h \in H_g$ at time $t \in \mathcal{T}$ is g^t_h . When $g^t_h < l^t_h$, the generation of the household does not convert demand and h purchases electricity from the utility. Otherwise, $g^t_h > l^t_h$, and the household sells back extra generation to the utility. The utility generation at time horizon $\mathcal{T}$ is given by

$$Q^t_G = \sum_{h \in H_r} l^t_h + \sum_{h \in H_s} l^t_h + \sum_{h \in H_g} \left(l^t_h - g^t_h\right) \forall t \in \mathcal{T}. \tag{6.2}$$

As the generation from both household and utility are different from the conventional power generation, renewable energy generation does not consume fuel sources. For simplicity, we assume a zero generation cost. Thus, the MG available energy Q^t_{MG} cannot be greater than the MG capacity,i.e, $0 \leq Q^t_{MG} \leq Q^{\max}_G, \forall t \in \mathcal{T}$

When the utility generation is not enough to meet the demand, utility buys excess generation from a set of households $h \in H_g$. The total cost $C_u(q_h)$ for supply generally consists of the generation cost and the energy purchase cost from the household $h \in H_g$. However, as the generation cost is equal to zero, the supplier cost is only the cost of purchasing energy. Let p^t denote the electricity price set by the utility at time t, and let $q_h = q^t_h, \forall t \in \mathcal{T}$ denote the amount of power purchased from the household h. The total utility cost for electricity, $C_u(q_h)$ is given by:

$C_u(q_h) = \sum_{t=1}^{T} p^t \sum_{h \in H_g} \left(g^t_h - l^t_h\right), \forall g^t_h > l^t_h$

Since the utility needs to satisfy the power demand, the total cost from the utility end, taking into account user satisfaction cost is given by

$$C_u = C_u(q_h) + \sum_{t=1}^{T} \sum_{h \in H} s^t_h \tag{6.3}$$

At the household side, the power consumption scheduling problem is usually solved by finding the optimal loads scheduling that minimises the household cost, and this in turn flattens the aggregated load curve and reduce the cost to the utility [124]. Thus, we let $C_{U,h}$ denote the total cost function associated with each household $h \in H$. $C_{U,h}$ encompasses the cost related to

the use of energy; the satisfaction cost induced by the operating mode mismatch; the cost of operating the battery; and the penalty cost.

Penalty Cost Function: A penalty cost function is associated to a duration of an interruption $C(P_{en}^t)$. Similar to the function described in [125], we consider a piecewise linear convex function of the form $C(x) = k_i x + b_i$ to approximate the penalty cost function $C(P_{en}^t)$.

Energy Cost Function: The energy cost (the cost of purchasing energy from the utility) for the household $h \in H$ at time $t \in \mathcal{T}$ is given by

$$C_{a,h}^t = \sum_{t \in \mathcal{T}} \sum_{a \in \mathcal{A}_h} p^t l_h^t, \tag{6.4}$$

where p^t is a dynamic price provided by utility.

Satisfaction Cost Function: Each household chooses a list of appliances and a preferred time for operation. Furthermore, since different households may have diverse preferences, it is not trivial to characterise them with a precise mathematical model. However, according to O'Neill *et al.* [126] the utility function is an abstract method used to model household preference. Similarly, we follow the same approach in this paper and assume that households would prefer to have their appliances operate sooner than later. This preference can be expressed as a strictly concave utility function that represents satisfaction of the user regarding the schedule. But we rather choose to work with the negation of this function, namely a dissatisfaction function that captures the dissatisfaction of the consumers (due to delaying or advancing the operation of an appliance). We denote $\bar{U}\left(p_{a,h}^t\right)$ as dissatisfaction of consumer when running appliance $a \in \mathcal{A}_h$. Depending on the priority level and the interruptibility, the dissatisfaction function may take different forms [127].

- Interruptible loads, $\bar{U}\left(p_{a,h}\right)$ can be defined as

$$\bar{U}\left(p_{a,h}\right) = \textstyle\sum_{t \in \mathcal{T}} \bar{U}\left(p_{a,h}^t\right)$$

- Deferrable loads such as a composite load,$\bar{U}\left(p_{a,h}\right)$ can be defined as

$$\bar{U}\left(p_{a,h}\right) = \bar{U}\left(\textstyle\sum_{t \in \mathcal{T}} p_{a,h}^t \Delta t\right)$$

Battery Cost Function: The battery is an energy storage device that flattens the power load by storing energy during low-cost (high-production) periods for use during high-cost periods. We assume that a subset of households H_s each have a battery storage device. At each interval, one

can either recharge or discharge the battery, but not both at the same time. Each battery has a total capacity $B_s^{\max}$, and let $Q_{c,h}^t$, $Q_{d,h}^t$ and $Q_{s,h}^t$ denote the energy charged, discharged and stored at time $t \in \mathcal{T}$ respectively. The charging and discharging power levels at each time t are bounded, and satisfy the following constraints:

$$0 \leq Q_{c,h}^t \leq Q_{c,h}^{\max}, \forall t \in \mathcal{T}, \forall h \in H_s$$
$$0 \leq Q_{d,h}^t \leq Q_{d,h}^{\max}, \forall t \in \mathcal{T}, \forall h \in H_s,$$

Where $Q_{c,h}^{\max}$ and $Q_{d,h}^{\max}$ denote the maximum charging and discharging rates, respectively. The energy stored in the battery should be non-negative and not greater than the battery capacity, and thus satisfies the following constraint $0 \leq Q_{s,h}^t \leq B_s^{\max}, \forall t \in \mathcal{T}, \forall h \in H_s$

The life-time of energy storage is usually characterised by the number of charging/discharging cycles that the battery storage can sustain, and repeated charging and discharging cause ageing of the battery devices. Therefore, let c_s denote the unit cost of charging and discharging, and the total cost of operating a battery storage is modelled after [128], [129] as follows,

$$C\left(b_s\right) = c_s \left(\sum_{t \in \mathcal{T}} Q_{c,h}^t + \sum_{t \in \mathcal{T}} Q_{d,h}^t \right), \tag{6.5}$$

Where b_s denotes the vector of charging and discharging amount over the scheduling horizon $\mathcal{T}$, respectively.

Therefore, the total cost of the household is given as follows:

$$C_{U,h}^t = \begin{cases} C(P_{en}^t) + \sum_{t \in \mathcal{T}} \sum_{a \in \mathcal{A}_h} \bar{U}\left(p_{a,h}\right) + C_{a,h} & \text{if} \quad h \in H_r \\ C(P_{en}^t) + \sum_{t \in \mathcal{T}} \sum_{a \in \mathcal{A}_h} \bar{U}\left(p_{a,h}\right) + C_{a,h} + C\left(b_s\right) & \text{if} \quad h \in H_s \\ C(P_{en}^t) + \sum_{t \in \mathcal{T}} \sum_{a \in \mathcal{A}_h} \bar{U}\left(p_{a,h}\right) + C_{h,buy} & \text{if} \quad h \in H_g \\ C(P_{en}^t) + \sum_{t \in \mathcal{T}} \sum_{a \in \mathcal{A}_h} \bar{U}\left(p_{a,h}\right) + C_{h,buy} + C\left(b_s\right) & \text{if} \quad h \in H_g, h \in H_s \end{cases} \tag{6.6}$$

For a household with generation $h \in H_g$, the energy cost also includes the profit they make by selling electricity to the utility. The cost function $C_{h,buy}$ is given by $C_{h,buy} = C_{a,h} - \sum_{t=1}^{T} p^t \left(g_h^t - l_h^t\right)$

6.3.2 Problem formalisation

We consider, the optimal power consumption scheduling problem (OPCSP) in a RCSMG to be a constraint based optimisation problem. The global objective is the sum of objective functions (the local objective functions at $h \in H$, and the global objective functions of N household loads). The local objective functions are used to find optimal schedule by minimising the households' cost functions while the global objective function is used to minimise the utility's costs (of operating the MG) while balancing the total power consumption/generation. The power consumption scheduling problem can be formulated as follows:

$$\min_{q_h, p_h} \sum_{t=1}^{\mathcal{T}} \left(C_u(q_h(t)) + \sum_{a \in A_h} \sum_{h \in H} C_{U,h}(p_h(t)) \right) \tag{6.7a}$$

$$S.t. \qquad q_h(t) = p_h(t), \forall_t \in T \tag{6.7b}$$

$$p_{a,h}(t)^{\min} \leq p_{a,h}(t) \leq p_{a,h}(t)^{\max}, \forall t \in \mathcal{T}, E_h^{\min} \leq \sum_{a \in \mathcal{A}_h} l_h^t \leq E_h^{\max}, \forall t \in \mathcal{T} \tag{6.7c}$$

$$0 \leq Q_{c,h}(t) \leq Q_{c,h}^{\max}, \forall t \in \mathcal{T}, \forall h \in H_s \tag{6.7d}$$

$$0 \leq Q_{d,h}(t) \leq Q_{d,h}^{\max}, \forall t \in \mathcal{T}, \forall h \in H_s, 0 \leq Q_{s,h}(t) \leq B_s^{\max}, \forall t \in \mathcal{T}, \forall h \in H_s \tag{6.7e}$$

$$0 \leq Q_{MG}^t \leq Q_G^{\max}, \forall t \in \mathcal{T}, L_t = \sum_{h \in H} \sum_{a \in \mathcal{A}_h} E_{a,h} \tag{6.7f}$$

Our objective function (6.7a), $C_{U,h}$, represents the household cost function as described by the household cost model, while C_u represents the utility cost function. The constraint (6.7b) represents the power supply-demand balance for each time slot t; and (6.7c) is the appliances operational constraints. (6.7d)-(6.7e) describe the battery storage operational constraints, and 6.7f is the MG generation constraint. The objective function (6.7a) is considered as a convex function, and its minimisation can lead to an optimal solution. Intuitively, such problem (6.7a) could be solved by the *CC* in a centralised manner; but for reasons explained in the overview of this chapter, such an approach raises a number of issues. Thus, to solve (6.7a) a distributed convex optimisation algorithms such as ADMM can be employed. In the next section we briefly describe the ADMM approach.

6.4 The ADMM Approach

The ADMM approach is a fairly dated concept which was introduced in the mid-seventies [130], and was recently reviewed [88]. An ADMM algorithm solves convex optimisation problems by decomposing them into smaller optimisation problems, which are simpler to solve individually in a distributed manner. The ADMM is known to solve problems of the form:

$$\min_{x_1,x_2} \quad f_1(x_1) + f_2(x_2) \quad \text{subject to} \quad A_1x_1 + A_2(x_2) = c \tag{6.8}$$

with variables $x_i \in \mathbb{R}^{n_i}$, where $f_i : \mathbb{R}^{n_i} \to \mathbb{R}$ are closed, proper, convex functions; $A_i \in \mathbb{R}^{mxn_i}$ are given matrices; and $c \in \mathbb{R}^m$ is a given vector. *Augmented Lagrangian* methods yield convergence without assumptions like strict convexity or finiteness of f_i [88]. Thus, the augmented Lagrangian for (6.8) is defined as follows:

$$\mathcal{L}_\rho(x_1, x_2, \lambda) = f_1(x_1) + f_2(x_2) - \lambda^T(A_1x_1 + A_2x_2 - c) + \frac{\rho}{2}||A_1x_1 + A_2x_2 - c||_2^2$$

where $\lambda \in \mathbb{R}^m$ is the Lagrange multiplier and $\rho > 0$ is the penalty parameter. First, the augmented Lagrangian is minimised with respect to the first variable x_1; next, using the new value for x_1, $\mathcal{L}_\rho$ is minimised with respect to x_2. Finally, the dual variable λ is updated as follows:

$$\begin{aligned} x_1^{k+1} &:= \operatorname*{argmin}_x L_\rho(x_1, x_2^k, \lambda^k) \\ x_2^{k+1} &:= \operatorname*{argmin}_y L_\rho(x_1^{k+1}, x_2, \lambda^k) \\ \lambda^{k+1} &:= \lambda^k + \rho(A_1x_1^{k+1} + A_2x_2^{k+1} - c) \end{aligned} \tag{6.9}$$

Thus, x_1 and x_2 are updated in an alternating fashion, which accounts for the term *alternating direction*. Separating the minimisation over x_1 and x_2 is precisely what allows for decomposition when $f_1(x_1)$ or $f_1(x_2)$ are separable, which will be useful in our algorithm's design. The drawback of the standard ADMM method is that it partitions the problem into only two sub-problems and thus cannot be implemented efficiently in a centralised way for a larger network. One way is to simply replace the two-block alternating minimisation scheme sequentially (the Gauss-Seidel update fashion), i.e., update x_i for $\{i = 1, 2, ..., N\}$. This approach updates the blocks one after another, which is not suitable for parallelization. To overcome this disadvantage, the Jacobi-type scheme updates all the N blocks in parallel [131]. In the proximal Jacobian Multi-block ADMM, the update of x_i is

$$x_i^{k+1} = \operatorname{argmin} \mathcal{L}_\rho(x_i, \{x_j^k\}_{j\neq i}, \lambda^k) + \frac{1}{2}||x_i - x_i^k||_{\mathbf{P_i}}^2 \quad \text{where} \quad ||x_i||_{\mathbf{P_i}}^2 = x_i^T\mathbf{P_i}x_i \tag{6.10}$$

for some symmetric and positive semi-definite matrix $\mathbf{P_i} \succeq 0$. When the x_i- sub-problem is not strictly convex, adding the proximal term [132] can make the sub-problem of x_i strictly or strongly convex, and make the problem more stable. The update of the Lagrangian multiplier in the proximal Jacobian ADMM is:

$$\lambda^{k+1} = \lambda^k - \gamma\rho(\sum_{i=1}^{N} A_i x_i^{k+1} - c), \tag{6.11}$$

Where $\gamma > 0$ is the damping parameter. The resulting optimisation problem is solved with the ADMM, where convergence is guaranteed if the following requirements are satisfied [88]:

1. The functions f_i are closed, proper, and convex
2. The Lagrange function $\mathcal{L}_\rho$ has a saddle point $(x_1^*, x_2^*, \lambda^*) \in \mathcal{R}$ such that $\mathcal{L}_\rho(x_1^*, x_2^*, \lambda) \leq \mathcal{L}_\rho(x_1^*, x_2^*, \lambda^*) \leq \mathcal{L}_\rho(x_1, x_2, \lambda^*)$

Wei *et al.* [131] proved the global convergence of Jacobian ADMM for appropriately chosen regularization matrices $\mathbf{P_i}$. Moreover, they showed that Jacobian ADMM has a convergence rate of $o(1/k)$. In section 6.5.1 and 6.5.2, we employ the multi-block ADMM to solve the power consumption scheduling problem in a decentralised and a fully-distributed manner. Liu *et al.* [133] discuss the use of Multi-block ADMM for smart-grid applications.

6.5 The ADMM Algorithms

6.5.1 Decentralised Power Consumption Optimisation

In this section, we study a decentralised power consumption scheduling algorithm using ADMM to solve the optimisation problem (6.7a). Procedures in 6.1 and 6.2 shows the resulting power consumption, decentralised asynchronous ADMM scheduling algorithm. Each *LC* is responsible for updating its own (p_h^{k+1},λ^{k+1}) using the most recent q_h value (denoted as $\tilde{q_h}$) received from *CC*. As such, the *CC* updates its q_h after receiving ($p_h^{k_h}$ and $\lambda_h^{k_h}$) from the *LC*s. Analogous to equation (6.9), the value of p_h^{k+1} and ${\lambda_h}^{k+1}$ can be updated as follows:

$$p_h(t)^{k+1} = \underset{p_h(t)}{\operatorname{argmin}}\, C_h(p_h(t)^k) + \langle \lambda_h^{k_h}, p_h(t)^k \rangle + \frac{\rho}{2}||p_h(t)^k - \tilde{q_h}||^2 \tag{6.12a}$$

$$\lambda_h^{k+1} = \lambda_h^{k_h} + \rho(p_h(t)^{k_h+1} - \tilde{q_h}) \tag{6.12b}$$

We consider q_h as the load which is suggested by the central controller to minimise the fluctuation in the power generation and consumption, and p_h as the load according to the consumers' own benefit. In a synchronization communication model, the *CC* must wait for the $p_h^{k_i+1}$ updates from all the N *LC*s. One draw back in this approach is that the *CC* has to wait for all updates

(p_h) from all the LC before updating λ resulting in the straggler problem[2]. A similar approach is discussed in [134]. We consider an asynchronous algorithm, where instead of a full synchronization on all LCs' reports in each ADMM iteration, a partial synchronization is only required. Updates from the more swift LCs are incorporated more frequently by the *CC*, while those from the slower *LC*s are not allowed to be older than a certain maximum delay. We consider the *CC* node keeps a clock k, which is incremented by 1 from zero after each λ^{k+1} update. Likewise, each *LC* also has a clock k_i, which is also incremented by 1 from zero after each λ_i update. All the clocks k and $(k_i)_{i=1}^{N}$ are run independently. We let, $p_h^{k_i}$ (where $i \in \{1, 2, ...N\}$) be the values of p_h when a *LC* i's clock is at k_i; and λ^k be the value of $\lambda(k)$ when the central controller's clock is at k.

To alleviate the straggler problem, a *partial barrier* can be employed [135]. The *CC* only needs to wait for a minimum of W updates, (where $W \geq 1$ and $W < N$). Reliance on this partial barrier with a small W means updates from slower *LC*s will be incorporated into computations less frequently than faster controllers. To ensure sufficient "freshness" of all the updates, we enforce a *bounded delay* condition: update from every *LC* has to be serviced by the central controller at least once every T iterations. T is a user-defined parameter ($T \geq 1$), where updates from each *LC* i can at most be T clock cycles old (with respect to the grid controllers clock). When both the minimum W updates and bounded delay conditions are met, the controlling node will proceed with the q_h update. We let Φ^k be the set of *LC*s with ($p_h^{k_i}$) updates that are received by controlling node (at iteration k). When the central controller updates, and sends ${q_h}^{k+1}$ to the *LC*s in Φ^k and its clock k is incremented by 1. Analogous to 6.9, the controller updates q_h as:

$$ {q_h}^{k+1} = \underset{q_h}{\operatorname{argmin}} \sum_{h=1}^{N} \langle -\tilde{\lambda_h}, q_h \rangle + \frac{\rho}{2} ||\tilde{p_h} - q_h||^2 = \frac{1}{N} \sum_{h=1}^{N} (\tilde{p_h} + \frac{1}{\rho} \tilde{\lambda_h}) \tag{6.13} $$

where $\tilde{p_h}$ and $\tilde{\lambda}_h$ are the most recent p_h and λ_h received from *LC* by the *CC*.

[2]The straggler problem allows the system to move forward only at the pace of the slowest *LC*

Algorithm 6.1: Local Controller Procedure

Initialize: $k = 0$,$\tilde{p_h}^{k_h+1} = 0$ $\tilde{\lambda_h} = 0$, $h = 1, 2, ..., N$
$\tilde{p_h}^{k_h+1}$ *being the most recent updates*
repeat
  repeat
    wait;
  until *receive W LC updates* ***and*** $max(T_1T_2, ..., T_N) \leq T$;
  for *LC* $h \in \phi^k$ **do**
    $T_h \leftarrow 1$;
    $p_h \leftarrow$ newly received p_h from local controller h;
    $\lambda_h \leftarrow$ newly received λ_h from local controller h;
  for *LC* $h \notin \phi^k$ **do**
    $T_h \leftarrow T_h + 1$;
  Update q_h^{k+1} by (6.13);
  Send q_h^{k+1} to all LC in ϕ^k;
until *termination*;
$k \leftarrow k + 1$;
until termination
output q_h^k

Algorithm 6.2: Central Controller Procedure

Initialize: $\lambda_h^0 = 0, k_h = 0$
repeat
  update $p_h^{k_h+1}$ by (6.12a);
  send $p_h^{k_h+1}$ to the grid controller;
  repeat
    wait;
  until ${q_h}^{k+1'}$ *is received from central controller*;
  Update $\lambda_h^{k_h+1}$ by (5b) ;
  $k_h \leftarrow k_h + 1$;
until *termination*;

Correctness Analysis

- *Partial correctness:* We claim that the loop *invariant* always hold at the loop test: $k \leq T$ ($k_h \leq T_h$) and $1 \leq W \leq N$ where $T \geq 1$.

- *Base case*: Assuming the loop invariant holds and the loop test passes. Say in first iteration, k = 1 then $k \leq T$ ($k_h \leq T_h$) is satisfied where $T \geq 1$ (considering that T has a considerable number of cycles). An update will occur in both algorithm fragments.
- *Inductive case*: Assume that the loop invariant holds at the loop test, and also that the loop test passes. New values of k, $\tilde{p_h}$ and $\tilde{\lambda_h}$ (λ_h and k_h) will also hold given $h = 1, 2, ..., N$
- *Termination*: The loop always terminates in the presence of at least one LC update ($1 \geq W \leq N$) and at most T clock cycles when $k = T$ ($k_h = T_h$).

Message complexity

Considering the decentralised algorithm, every *LC* sends a message in one clock cycle, k to receive an update. The *CC* will broadcast w messages to a set of *LC*s with p_h updates that were received (at iteration k). Each time the *CC* will require at least Z messages for a single successful update of the price signal, where Z is a partial barrier of the minimum updates that the *CC* requires to be updated. Overall the *CC* would require Z^{T_N} where in T is user-defined.

6.5.2 Fully Decentralised Power Consumption Optimisation

In subsection 6.5.1 a decentralised approach for solving the power consumption scheduling problem is presented. However, it is envisaged that as micro-grids become increasingly interconnected, in the future, a fully distributed power consumption scheduling approach will present a valuable alternative to the decentralised approach. In this section, we study a fully distributed algorithm to solve the optimisation problem (6.7a), where the devices make and coordinate their schedules through local communication with their neighbouring nodes. The key feature of this algorithm is that the computation is localised at the nodal level of the micro-grid network, which does not require any form of central coordination.

We reformulate the OPCSP (6.7a) using the ADMM formulation made in section 6.4 to get:

$$\min_{\{q_h, p_h\}} \sum_{t=1}^{T} \left(\sum_{n=1}^{N} C_u(q_h(t)) + \sum_{m=1}^{M} C_{U,h}(p_h(t)) \right) \tag{6.14}$$

Constraint (6.7b) is the power supply-demand balance equation that ensures the total demand is satisfied by the power generation for each time slot t. It couples variables across different DGs and loads. Constraint (6.7c)-(6.7f) are local constraints that ensure the loads, batteries and generators do not violate operative limits. Let $\lambda := [\lambda^1, , , \lambda^T]$ denote Lagrange multiplier vector associated with the coupling equality constraint. The augmented Lagrangian for equation 6.14

can be given as follows:

$$\mathcal{L}_\rho(q_h, p_h, \lambda) = \sum_{t=1}^{T}\sum_{n=1}^{N} C_u(q_h(t)) + \sum_{t=1}^{T}\sum_{m=1}^{M} C_h(p_h(t)) - \sum_{t=1}^{T}\lambda(t)\left(\sum_{n=1}^{N} C_u(q_h(t)) - \sum_{m=1}^{M} C_h(p_h(t))\right) + \frac{\rho}{2}||\sum_{t=1}^{T}\left(\sum_{n=1}^{N} C_u(q_h(t)) - \sum_{m=1}^{M} C_h(p_h(t))\right)||_2^2, \tag{6.15}$$

where λ and $\rho/2$ are the penalty coefficients for the first and second order terms of disagreement.

The OPCSP is solved across the *LC*s of the DERs and loads. That is, at each step k, each *LC* of DG and load solves the primal problem of ensuring that the local constraints hold, then communicates the generation and consumption schedules to their neighbouring nodes. The update of the *LC*s can be performed concurrently according to the proximal Jacobian multi-block ADMM. Algorithm 6.3 presents the resulting fully decentralised power consumption scheduling solution. Initially set $k \leftarrow 0$. The *LC*s of the DGs and loads set their initial schedules and communicate them to their neighbour nodes. One of the *LC*s sets the initial $\lambda(k)$, and broadcasts to its neighbour nodes. Then,

- *The LC of each DG unit solves the following problem (analogous to equation 6.10): OPCSP-LC(DG)*

  $q_h(t)^{k+1} = \text{argmin}_{q_h}\, C_u(q_h(t)) + \rho/2 \sum_{t=1}^{T} ||q_h(t) - p_h(t)^k - \frac{\lambda^k}{\rho}||_2^2 + \frac{1}{2}||q_h(t) - q_h(t)^k||_{\mathbf{P_i}}^2$
  The $q_h(t)$ is computed and broadcast to the neighbouring nodes while the utility function is kept private.
- *The LC of each load solves the following problem (analogous to equation 6.10): OPCSP-LC(Load)*

  $p_h(t)^{k+1} = \text{argmin}_{p_h(t)}\, C_h(p_h)^t + \rho/2 \sum_{t=1}^{T} ||(p_h(t)) - (q_h(t))^k - \frac{\lambda^k}{\rho}||_2^2 + \frac{1}{2}||p_h(t) - p_h(t)^k||_{\mathbf{P_i}}^2$
  The $p_h(t)$ is computed and broadcast to the neighbouring nodes while the information of cost function is kept private.
- The dual updating step λ can be computed by any one of the *LC*s and broadcast for all neighbouring nodes. That is, after receiving schedules from the neighbouring *LC*s of DGs and loads, one of the LCs perform a simple update on the dual variable.

This ensures the privacy of the consumer's preferences and constraints (the load) and the production costs and constraints of DGs is preserved by the power consumption, since control

signals and schedules is the only data exchanged between *LC*s.

Algorithm 6.3: Distributed Power Consumption Scheduling using ADMM

Initialize: $k \leftarrow 0, \rho > 0, \gamma > 0, \lambda^0 \leftarrow 0$

repeat

  /* The LC at each DG and each load computes a schedule */

  /* by solving the corresponding OPCSP-LC */

  $q_h(t)^{k+1} = \text{argmin}_{q_h}\, C_u(q_h(t)) + \rho/2 \sum_{t=1}^{T} ||q_h(t) - p_h(t)^k - \frac{\lambda^k}{\rho}||_2^2 + \frac{1}{2}||q_h(t) - q_h(t)^k||_{\mathbf{P_i}}^2$

  $p_h(t)^{k+1} =$
  $\text{argmin}_{p_h(t)}\, C_h(p_h(t)) + \rho/2 \sum_{t=1}^{T} ||(p_h(t)) - (q_h(t))^k - \frac{\lambda^k}{\rho}||_2^2 + \frac{1}{2}||p_h(t) - p_h(t)^k||_{\mathbf{P_i}}^2$

  /* Update λ using one of the LCs in each period */

  $\lambda(t)^{k+1} = \lambda(t)^k - \gamma\rho \left(\sum_{n=1}^{N} q_h(t+1) - \sum_{m=1}^{M} p_h(t+1) \right)$

  $k \leftarrow k + 1$;

until *convergence*;

6.6 Summary

We presented a robust enforcing consumption scheduling algorithm which operates close to real-time that can be integrated into the CDA (dedicated on future provisioning of power). In standard grid architectures, demand management is handled via scheduling protocols that are centrally coordinated. Centralised approaches are however computationally intensive, as such, there are not well suited for distributed grid architectures with limited computational power. We address this problem with a decentralised scheduling algorithm. In our scheduling algorithm, the alternating direction method of multipliers (ADMM) is used to decompose the scheduling problem into smaller sub problems that are solved in parallel over local computation devices, which yields an optimal solution. We show that ADMM can be used to model a scheduling solution that handles both decentralised and fully decentralised cases.

Importantly, it is reasonable, to expect power scheduling algorithms to have significant attention from a security perspective [16]. In [16], we investigate false data injection attacks that can be provoked by compromising parts of the communication infrastructure or a set of computing devices. Such attacks result in the decentralised consumption scheduling algorithm failing to converge to the optimal solution or allows it to converge toward a value that benefits the attacker. Such an approach although valuable, fails to provide a more encompassing and systematic way in which system defenders can better understand vulnerabilities and attacks in developing security solutions. As future work, it is worthwhile developing a framework to formalise and design attacks on the decentralised ADMM types of algorithms.

In Chapters 7 and 8 we shift our focus to cheating detection and mitigation aspects of the decentralised CDA algorithm.

Chapter 7

The Design and Classification of Cheating Attacks in CDAs

> Good fences make good neighbours.
>
> UNKNOWN.

7.1 Overview

Despite the evident benefits and hype, there is a sinister reality quickly associated with continuous double auctioning (CDA). The algorithm is susceptible to fraudulent behaviour and malicious attacks in the form of *Cheating*. Participants can engage in undesirable and fraudulent behaviour in an attempt to gain an unfair advantage or additional energy in the auction. The literature we reviewed in Section 3.5, shows that cheating attacks and their inherent mitigation measures are auction mechanism specific. This observation, coupled with the realisation that known classic cheating attacks are unlikely to manifest on a decentralised CDA (Section 4), leaves system defenders limited understanding of the attacks to defend against and the ideal mitigation measures to deploy. The design of cheating attacks is important, because it provides insights into new attacks; can inspire design of novel mitigation solutions; maintains grid stability and reliability. To the best of our knowledge, no systematic framework exists, to guide the design and analysis of cheating attacks. The purpose of this chapter, is to advance the state-of-the-art, by providing such a framework for decentralised CDA for RCSMGs. To model cheating attacks on decentralised CDA algorithms we propose an ACA framework in Section 7.2. The *ACA* framework consists of a *Domain model* 7.2.1, an *Attacker model* 7.2.4, and an *Attack model* 7.2.5. In Section 7.3, we classify the cheating attacks then in Section 7.4 further analyse a few selected cheating attacks, outlining the attacks' feasibility, procedures, performance overheads

and drawbacks. The summary in Section 7.5 concludes this chapter. The plausible detection and mitigation solutions are discussed in the next chapter.

7.2 Framework Design

In this chapter we propose a framework that allow designing of a variety of cheating attack models for CDA. Our framework, which we shall refer to as the Automated-Cheating-Attacks (*ACA*) Framework is inspired by the work of Adepu and Mathur [103]. We base our framework on Adepu and Mathur's work as it is simple, allows easy design of attacks and to the best of our knowledge is the closest framework that describes designing of attacks with more coverage. By leveraging some aspects of Adepu and Mathur's framework and extending it to capture the CDA and behavioural aspects, we are able to develop an arguably effective framework for designing cheating attacks for the given context. Figure 7.1 depicts the *ACA* Framework. We envisage that our framework allows researchers to design a variety of cheating attacks on CDAs in general, and on decentralised CDAs specifically, for the assessment of attack detection methods and tools.

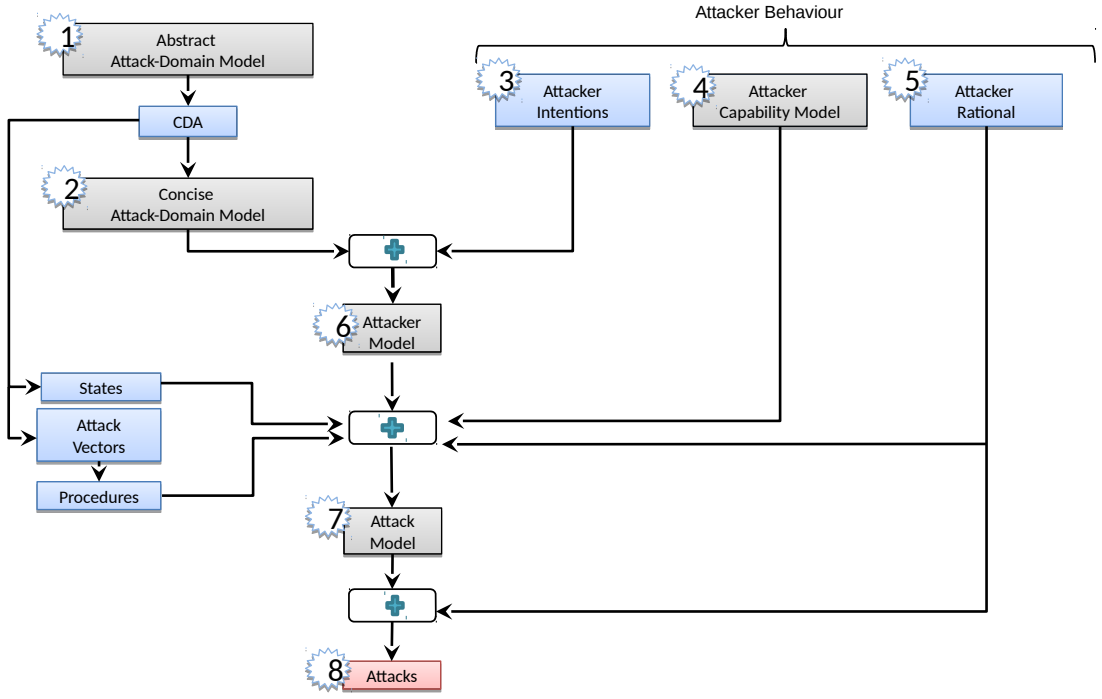

FIGURE 7.1: *ACA* Framework for deriving attacks

For simplicity and ease of conceptualising, the ACA framework has the following main components or models, namely: *Attack-Domain* model, *Attacker* model, and *Attack* model, as these directly guide the design of attacks. The *Attack-Domain* model captures the CDA and supporting

(infrastructure) elements that could serve as the target of an attacker. The *Attacker* model captures adversary's intentions as functions on the attack-domain model. This identifies the target and intent of an attacker. The *Attack* model captures the relevant encompassing elements of an attack which include: attack domain, attack vectors, attack procedures and the start/end states of the attack. The *ACA* Framework summarised in Figure 7.1 is a process consisting of eight stages, labelled 1 through 8, to derive attacks. Similar to [103] at Stage 1 is an abstract[1] domain model. Three dimensions can be used to describe the attack-domain space: components *Cm*, process properties *Pr*, and system performance metrics *Pe*. For example, *Cm* has terminal nodes, communication nodes, etc.; while *Pr* has properties such as number of trades, surplus(profit margins) etc.; and *Pe* has metrics such as allocative efficiency, number of messages, time to trade, etc.; but no further details of these elements (Figure 7.2). Stage 2, is a mapping of the abstract attack-domain maps to a concise attack-domain model for a CDA as explained in subsection 7.2.1. Stage 3, 4, 5 are aimed at defining the attacker's behaviour aspects. Stage 3, defines the *Attacker Intents*, which are combined with the concrete domain model in stage 6 to generate an attacker model. Stage 4 and 5 defines the *Attacker's Capability* model and *Attackers Rational* respectively. In Stage 7, the *Attacker* model is combined with the states, Attacker capability, Attack Vectors, and Attack Procedures to generate an *Attack* model. The left-hand side inputs are mostly technical aspects, while the right-hand side inputs are human behaviour aspects. Attacks are derived from the attack model and the *Attackers Rational* in stage 8. The following subsections outline the 8-stage process citing specific examples of a decentralised CDA [28] designed for a RCSMG context.

7.2.1 Attack Domain Model

The attack domain consists of three finite sets namely the component set (Cm), property set (Pr), and performance set (Pe). As shown in Figure 2, each of these is part of a three-dimensional attack space. Formally, an attack-domain model ADM of a CDA is a 3-tuple of (Cm, Pr, Pe), where Cm, Pr, and Pe denote system components, system properties, and system performance metrics, respectively. Thus, ADM defines a finite attack space an attacker can explore and enables specification of the attacker intent (Figure 7.2).

Cm usually includes elements that may be physical, cyber, and logic, but in this thesis we narrow Cm to define cyber components (eg. wireless network) on top of which a CDA algorithm is run. A CDA is supported by components that are usually networked together to ensure multiple buyers and sellers participate in the auction. Cm defines the cyber and logical components that supports the auctioning algorithm and the participants such as terminals for participants; networking components; software agents trading on behalf of the participants; etc. There

[1]The domain is considered abstract, as its elements do not have the specifics required for the modelling and analysis of a CDA.

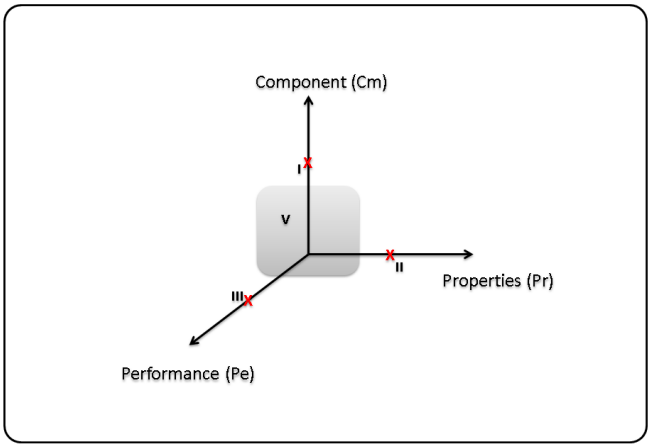

FIGURE 7.2: Attack Domains

certainly are other categories of components in a CDA; thus the examples given herein are not claimed to be complete in any sense. In Adepu *et al.* [103], each element of Cm is also referred to as an attack point, or simply as a point. An attack point serves to define an attack vector.

The Pr dimension includes measurable properties of the products being produced or controlled by the CDA such as market surplus, number of trades, the market equilibrium price. It is important to note that the same property is likely to be measured at different physical/logical locations depending on the architecture of the CDA.

The Pe dimension refers to one or more performance characteristics of a CDA such allocative efficiency, convergence rate to equilibrium price, communication overheads, time of a single trade, etc. While the Pe metrics may appear similar to Pr properties; it is arguably better to consider them as separate dimensions of a CDAs. Consider an element in Pe to refer to a measurement taken at a specific point in the progression of a CDA to ascertain how a component or the whole system is performing. As such elements of the set Pr show how well the whole CDA or its components are performing. Pe is measured at different physical/logical locations (measurement points) depending on the architecture of the CDA.

7.2.2 Concise Attack Model

In order to come up with a *concise Attack-Model* there is need to understand the CDA component in the framework. Section 4.3, formalises the descriptor of the decentralised CDA as a septuple. To recap, the main activities —which define valid states, S (see section 7.2.6) in a decentralised CDA are: Registration, Initialisation, Participation Request, Bid Formation, Transacting, Termination. The decentralised CDA algorithm has been partitioned into procedures, which are

executed at the different components within the auction infrastructure. In this chapter, we focus on the Local Market Execution procedure (Algorithm 4.2) as it is more informative on the types of attacks that can occur. For further understanding of the Continuous Double Auction algorithms and their application in grid-like platforms, the reader is also referred to Chapter 3 and the literature [9], [136], [77], [68], [13], [73], [1].

7.2.3 Attacker Behaviour Model

The ability to model 'attacker behaviour' can provide attack scenarios that will probably happen, which gives precise risk assessments and damage predictions. Representing attacker behaviour in terms of attack effects, instead of the attack itself, allows indirect evaluation of the system security, by identifying families of attacks and not each instantiation [137]. Attacker behaviour can be modelled as a strategic decision-making process that accounts for the following factors affecting the attacker's decisions:

Attacker Intent: Intent-based approach is evident in earlier design attacks [138]. As social criminals, in cyberspace, attacks are typically not random, and an attacker launches an attack to achieve some malicious goals [139]. From Figure 7.2, point **I** implies an attacker designs an attack with the possible intention of damaging a specific component in its domain without, in the short term, affecting any system property or performance. Point **II** requires an attack to alter some system property such as profit margins (cheating). Point **III** requires an attack for reducing some system output such as convergence to an equilibrium price within an auction trade day. Such an attacker model captures the mapping of attacker intentions to the attack-domain.

Attacker Capability Model: Formally, the attacker capability model *ATC* of a CDA is a 3-tuple (Sk, De, Dn), where Sk, De, and Dn denotes sets of system knowledge, disclosure resources, and disruption resources. Thus, *ATC* defines a finite attack space an attacker can explore in relation to attacker capability/ constraints (see Figure 7.3).

Similar to previous approaches [140] [141], we propose three dimensions for the attack space: the adversary's a priori system model knowledge Sk; his disclosure resources De; and disruption resources Dn. An attacker with *a priori* system knowledge (including knowledge of elements on ADM) can construct more complex attacks, possibly more difficult to detect and with more severe consequences. This depends on the extent and depth of the knowledge such as knowledge of terminals used by auction participants; the networking components; the software components etc.. Similarly, the disclosure resources De enable the attacker to obtain sensitive information about the system, or its components during the attack by violating the data confidentiality. For example the attacker can obtain private information secret to other auctioneers (trading agents).

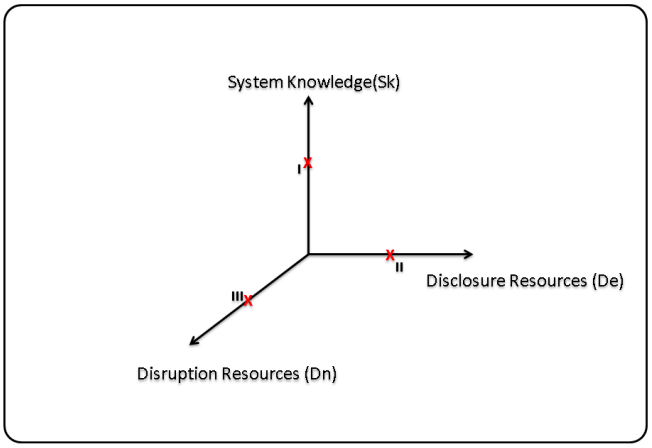

FIGURE 7.3: Attacker's Capability Space

We consider that disclosure resources cannot be used to disrupt the system operation. On the other hand, disruption resources can be used to affect the system operation, which happens for instance when data integrity or availability properties are violated

Attacker Rational: As emphasised by Sedaghatbaf and Azgomi [100], it is necessary to find out how probable an attack is, from its potential costs and benefits, instead of relying only on the technical aspects of the system. The supporting notion is that attacks are considered unlikely if their cost is not worth their benefits. Thus, if an attack is unlikely to occur, it is assumed the system is secure towards that attack, despite its potential technical vulnerability, because it is irrational for an attacker to perform an unprofitable attack. Formally, the Attacker's Rational $\mathcal{R}$ is a 3-tuple (Cb, Sd, Ac), where Cb is a cost-benefit-analysis function, Sd is the behaviour set dependent on the system's defence, and Ac is the number of attackers and their coordination. This model is helpful in building a taxonomy of attacks within the *ACA* Framework.

- *Cost-Benefit-Analysis:* An attacker's decision to execute an attack is made by a cost-benefit model, such as those defined by Amenaza [142] and Buldas *et al.* [143]. Thus, we consider an attacker attacks only if the overall attack is profitable. The attacker chooses the most beneficial strategy in each stage of the attack. This assumption serves as an upper bound in the modelling context, as any irrational decision hinders the result of the attacker himself.
- *System Defence Interdependency:* The behaviour of an attacker and the system's defence mechanism are interdependent, and security reactions can change the possible attack steps for the attacker [139]. Petri net-based models [144] or partially observable Markov decision process [137] can be used to capture the dynamics between the interaction between attacker and defender (i.e. the system's defence mechanism).

- *Attacker Coordination:* The type of plausible attack need to factor in the potential of more than one individual attacking the system concurrently. Such attackers can be individualistic or can collaborate and coordinate to deliver some attack that benefits the parties.

7.2.4 Attacker Model

The *Attacker Model* Ä is defined as a 2-tuple (Ĩ , ADM) where Ĩ is a finite set of attackers' intents and ADM is an attack-domain model. In this chapter, the attacker's goal or an objective is treated as an intent. Given its inherently personal and social nature, Adepu and Mathur [103], note it is difficult to precisely specify the intent, set Ĩ . However, we consider intents to include: damage to a component (Cm), disturbing a system property (Pr), or altering system performance (Pe). An intent is applied to an element or a region of the domain as in Figure 7.2. Thus, an intent function is used by an attacker to one or more elements of an *Attack Domain* that defines a CDA. The *Attacker Model* does not describe the actual attack itself. As noted by Adepu and Mathur, attacks are designed to achieve an intention goal.

7.2.5 Attack Model

For a CDA, say X we consider an attacker model $\ddot{\mathcal{A}}_{\mathcal{X}} = (\tilde{\text{I}}, ADM)$, where $ADM = (Cm, Pr, Pe)$ and Ĩ is a function. The attack model $\breve{\mathcal{A}}$ is a sextuple

$$\breve{\mathcal{A}}_X = \{\dot{\mathcal{A}}, \breve{\mathcal{C}}, \tilde{\mathcal{I}}, \mathcal{D}, \mathcal{P}, \mathcal{S}_0, \mathcal{S}_e, \mathcal{R}\} \tag{7.1}$$

where, $\dot{\mathcal{A}}$ is potentially infinite set of procedures to launch attacks; $\breve{\mathcal{C}}$ is an attacker capability model for the attacks derived from *ATC*; $\tilde{\mathcal{I}}$ is a finite set of attacker intents; $\mathcal{D}$ is the domain model for the attacks derived from the attack-domain model *ADM* of X; $\mathcal{P} \subseteq Cm$ is a finite set of attack points; and $\mathcal{S}_0$ and $\mathcal{S}_e$ are possibly infinite sets of states of X, that denote, possible start and end states of interest to the attacker; $\mathcal{R}$ is the attacker rational that determines the feasibility of launching the attack. An attack point in X could be a physical element or an entry point through the communications network connecting CDA and the underlying power system.

We are aware that some attacks could be modelled to leverage on faults, but that phenomenon is beyond the scope of attacks we cover in the *ACA* Framework. Implicitly, our attack model herein assumes no faults occur (thus, fault-related attacks are excluded). We note that several formalisms exist for modelling attack procedures, $\dot{\mathcal{A}}$, by employing graphical methods, among others [145], [144], and [146] as cited in [103]. Modelling of $\dot{\mathcal{A}}$ are imposed on the attack methods. The set Å is similar to the attacker payload [138].

From the above *Attack Model* definition several attack models can be built for a given CDA. The different attack models on the same CDA are derived by altering/changing the constraints imposed on Å and selecting different subsets of the Attack-Domain variables (Cm, Pr, Pe); and Attacker Behaviour which encompasses the Attacker Capabilities (Sk, De, Dn); the Attacker Rational (Cb, Sd, Ac). The context in which an attack model is used will determine its appropriateness and effectiveness. We expect the attack model presented herein, to be useful in designing, among other attacks, cheating attacks to study the resilience of a decentralised CDA.

7.2.6 Attacks

For a CDA, say X, we consider an attack model

$$\breve{\mathcal{A}}_X = \{\dot{\mathcal{A}}, \breve{\mathcal{C}}, \tilde{I}, \mathcal{D}, \mathcal{P}, \mathcal{S}_0, \mathcal{S}_e, \mathcal{R}\} \tag{7.2}$$

An attack Æ in X is defined as a terminating or a non-terminating procedure $\dot{a} \in \dot{\mathcal{A}}$ designed to realise a finite set of intents $\tilde{i} \subset \tilde{\mathcal{I}}$, aimed at domain $d \subset \mathcal{D}$, requiring a finite set of capabilities $\breve{c} \subset \breve{\mathcal{C}}$, launched through a finite set of points $p \subset \mathcal{P}$ when X is in state $s_0 \in \mathcal{S}_0$ and possibly removed when X is in state $s_e \in \mathcal{S}_e$.

Attack Success: We consider an attack Æ to be successful if all intents in ĩ are realised in a finite time. Part successful attacks occur when a subset of intents in ĩ are realised, while unsuccessful attacks occur when none of the intents in ĩ is realised and there is no intended or unintended side effect of applying an å to CDA_X. The attack procedure å may or may not terminate after all intents in ĩ are realised.

Attack Vector: An attack vector is a path in a CDA that starts at an attack point p and allows the exploitation of a vulnerability. In other words, an attack is a parameterised procedure that exploits such a path. Identification of attack vectors is beyond the scope of this book chapter, but it is possible to identify attack vectors for the design of attack procedures using the attack domain model and knowledge of its operation.

Attack Procedure: One key factor to a successful attack, is the design of an effective attack procedure å. This requires the attacker to be familiar with at least the targeted components and CDA system C. In cases where multiple points are attacked, and deception is needed to avoid detection, å might actually involve computation of data values being sent in real time. For example, an attacker may be interested in knowing the evaluation and reservation prices of other victim participants in real time, so as to inform his decision in the CDA. Doing so requires

a deeper knowledge of the properties of the CDA and its components. It is assumed that, such knowledge is used in the design of å for a successful attack. Thus, in the attacker and attack models, the knowledge of the system: partial or complete (as specified in Attack Capability model), is encapsulated in the attack procedure.

Attack Start States: Say an attacker, is able to install a piece of malware into a component(s) when the CDA system is in some state s. Then consider the malware will remain dormant until the CDA system reaches another state $s^{'}$ when it actually executes the intended attack. The start state s_0 in this case can one of who cases:

- The attack is a sequence of two separate attacks, with malware injection attack followed by the malware's payload execution. The start state for malware injection is s and for the subsequent payload execution launched by the malware is $s^{'}$.
- The entire attack is one and we treat the start state as s.

The best approach between the aforementioned two depends on the intent of the attack. If the state in which malware is inserted is key to the attack being not detected, while the payload execution can be launched in any state, then s should to be considered the start state while $s^{'}$ can be ignored. However, if malware must launch the subsequent attack in a specific state for intent realization, then $s^{'}$ should be considered. Examples of attacks and their types are given in Section 7.4.

Valid State Sequences: Valid sequences are those sequences that appear at least once during the normal operation of CDA_C (see Section 4.3). Given how complex a CDA can be, the space of all valid sequences of any arbitrary length is huge and difficult, if not impossible, to enumerate. Under attack, or in case of some form of failure, CDA_X might enter an invalid sequence, i.e., a sequence that would never occur under normal operation constrained by rules of the CDA.

7.3 Design and Classification of Cheating Attacks

The *ACA* Framework described allows a variety of attacks which are beyond only cheating attacks to be designed. Overall, the set of attacks that can be designed at a time is governed by the intent of an attacker. For instance, cheating intent can be considered as a function that is applied by an attacker to the property dimension Pr Dimension of the Attack-Domain in order to influence profit distribution. In Figure 7.2 this set of attacks can be mapped on Point **II** domain. The RCSMG platform warrants that such attacks be resource aware in order to be successful and to avoid easy detection. As such, a concise *Attack-Domain Model* will define the upper and lower bounds of the system resources in which plausible attacks can be executed.

Consider the decentralised CDA as the appropriate auction mechanism for power allocation in a smart grid. Such a decentralised CDA will determine the states, attack vectors and tentative procedures on which an attack can be build on. In turn, this implicitly narrows the set of plausible attacks that are likely to manifest such a CDA scheme. It is due to this reasoning that traditional cheating attacks can be eliminated as there are unlikely to manifest on the given CDA states and attack vectors. For example in a centralised auction the auctioneer is a vector for the attack. This is not necessarily the case in a decentralised scheme considered herein (as proposed in [28]). The attack model in turn specifies the attacker model according to a CDA's states and attack procedures, which are guided by the attacker's capability and rational. Cheating attacks can be classified with respect to the attacker's capabilities and attacker's rationality, in that order.

TABLE 7.1: Adversary types and capabilities

Adversary Capabilities			
Attacker Types	Knowledge	Disclosure	Disruption
Limited Attacker	High	-	-
Advanced Attacker A	High	Yes	-
Advanced Attacker B	High	-	Yes
Expert Attacker	High	Yes	Yes

Attacks are categorised according to attacker capabilities in step 7 in Figure 7.1: Limited, Advanced (A and B) and Expert (shown in Table 7.1). For the purpose of this chapter we will sideline Attacker B type and focus on Attacker A as the most likely capable attacker types to bring forth automated forms of cheating. At step 8 of the framework, the attackers can be further categorised using the Rational Model (Section 7.2.3) according to the number of adversaries and their interaction as follows: a single attacker; multiple non-cooperating attackers; and multiple cooperating attackers (Figure 7.4). The single attacks can be carried out even by multiple adversaries, with an additional complexity on the impacts and the respective attack model that can be designed.

7.4 Cheating Attack Cases

In this section we use the *ACA* framework as a basis for modelling the cheating attacks towards a decentralised CDA. The cheating attacks we consider herein are not exhaustive but crucial in informing on the mitigation measures that can be put in place ensuring the CDA algorithm executes successfully. The *Victim Strategy Downgrade* and *Collusion by Dynamic Strategy Change* have been described in [27]. We expand on the aforementioned attacks with a more formalised approach demonstrating how the proposed framework can be used in designing such attacks.

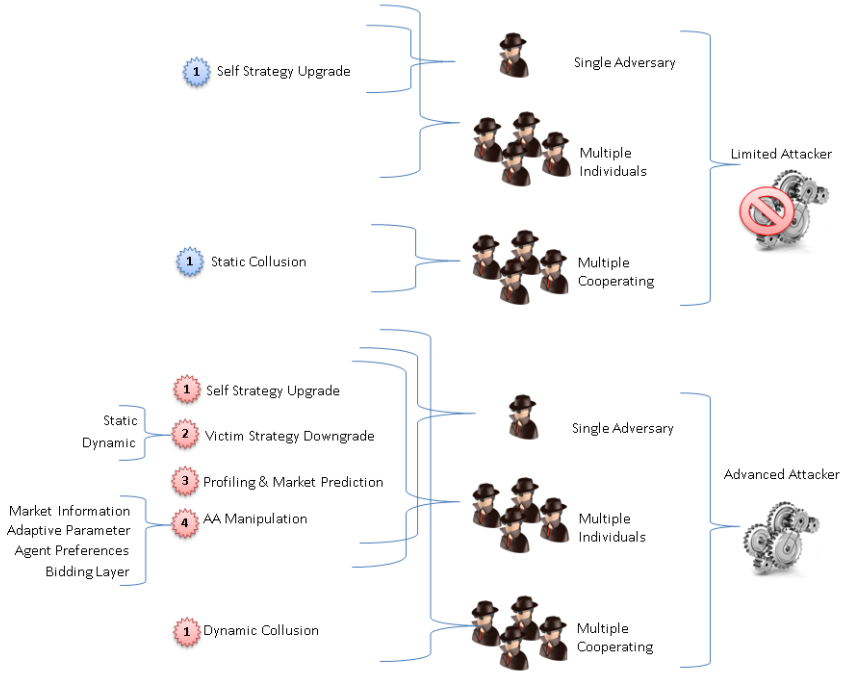

FIGURE 7.4: A classification of cheating attacks on decentralised CDAs

We additionally consider *Profiling and Market Prediction Attack*; and *AA Manipulation Attack*.Our choice of attacks was guided by the following reasons:

- can be generalised and can provide a wider coverage of attacks to consider for the system defenders;
- can provide more insight into the system's behaviour under different attack procedures and parameters;
- can be used to learn and test the resilience of the system.

7.4.1 Victim Strategy Downgrade

An attacker, P_{Ai}, where ($i = 1$, $P_A \subset P$, and P is a set of all participants) employs an automated tool (e.g. malware) to gain control of other trading agents $\mathcal{TA}$ strategies. The attacker can 'downgrade' victim *TA*s to trading strategy that is inferior to gain them additional advantage. We are aware the same attack can be perpetrated by an adversary simply changing their agent strategy to adopt a different but superior bidding strategy which affords higher surplus. We will focus on the former, as it presents an exciting problem opposed to a simple P_{Ai} upgrading to a superior strategy. The Victim Strategy Downgrade has been described in detail in the literature [27] and [21]. The term attacker and adversary are used interchangeably. We shall consider the following assumption:

- the adversary P_A has control of $P - 1$ victim $\mathcal{TA}$ s;

- despite $P-1$ victim $\mathcal{TA}$ s having their strategy downgraded, to say, *ZI Strategy*, the $\mathcal{TA}$ s will seamlessly continue participating in the auction as normal;
- the adversary has computational capacity to launch the attack and a reliable connection exists with the victim $\mathcal{TA}$s.

Motivation: A number of trading agent strategies that have been developed for the CDA over the years. We note experimental evidence that supports the phenomenon of some strategies being superior; with the ability to gain more surplus from trade than their inferior counterparts [1], [74], [95], [36]. The Adaptive Aggressive strategy is one such superior strategy, while the Zero Intelligence is the most inferior [95], [1], [74]. Ma and Leung [36], demonstrate that AA agents are adaptive to different combinations of competitors; and to different supply and demand relationships. ZI agents behaved worse since they do not analyse their environment and the other agents whom they are competing with. The AA obtains huge profit margins in comparison to ZI strategy [36].

Attack: Assume an attacker distributes a malicious code to attach itself onto other participants *TA*s (the attack vector). The malicious code (or malware describe in [97]) carries a payload capable of incorporating an inferior *ZI* strategy to the victim $\mathcal{TA}$ s' bid forming mechanism. Formally, a strategy downgrade attack Æ (sda) in X is defined as a terminating procedure å $\in$ Å designed to realise a finite set of intents ĩ $\subset$ Ĩ (where ĩ includes gaining control of other $\mathcal{TA}$ s strategy component, changing their strategy, influencing the market to gain additional surplus), aimed at domain $d \subset D$ (where d is a set of all M_{mp} components where $\mathcal{TA}$ s are hosted), requiring a finite set of capabilities č $\subset$ Č (č includes knowledge of participants, ability to launch a malicious code[2], access and communication with victim $\mathcal{TA}$ s), launched through a finite set of attack points $p \subset P$ (p is a set of all victim $\mathcal{TA}$ s), when X is in state $s_0 \in S_0$ (where s_0 is a set of all valid CDA states where an attack can be launched) and possibly removed when X is in state $s_e \in S_e$ (where s_e is a set of all valid states the attack can be stopped which is controlled by a clock trigger).

Attack States: The attack is a sequence of two separate phases: malware injection phase followed by the malware's payload execution phase. The start state for malware injection is s and for the subsequent payload execution launched by the malware is $s^{'}$. Malware injection occurs in any of the valid states of the CDA, while payload execution is triggered by a clock. It is important that victim $\mathcal{TA}$ s are all in $s^{'}$ concurrently through a synchronisation protocol. The aim of this attack is not to put the overall CDA into an invalid state, but to ensure despite the attack the CDA is always in its valid states.

[2]These are described as disclosure resources

Attack Variants This attack can take two forms: static or dynamic downgrade. In static downgrade, an agent will instantly change its strategy on infection (once adversary payload is delivered). This change is somehow permanent. If no further coordination occurs between the infected agents and the adversary, it would be difficult to use communication overheads to infer occurrence of such cheating. However, such attack can easily be detected by analysis of market efficiency as *ZI* agent population yields fairly lower market efficiency than a homogeneous *AA* population. Thus, an advanced attacker would employ a dynamic downgrade to victims *TA* strategies allowing victim agents to revert to the *AA* strategy based on a clock-based trigger. We consider the payload incorporates the inferior *ZI* strategy to the victim $\mathcal{TA}$ s bid forming mechanism. Since the attack is dynamic, the payload will ensure the victim toggles between *AA* and *ZI* strategy in strict response to a clock trigger or messages from the adversary. Algorithms 7.1 and 7.2 present the dynamic attack from an attacker and the victims' perspective.

Attacker Rational In order to evade easy detection an attacker can send $P-1$ 'revert' messages to the victims using a private back-channel directly to victims. Only the trading outcome is affected as the CDA remains fairly the same. The Adversary is required to be computationally upt, and to establish a reliable connection with the victims in order to execute such an attack. In a resource constrained setup, this might not be feasible. To maximise on the surplus to be gained the adversary can only participate when the clock trigger occurs. If a considerable number of victim $\mathcal{TA}$ s have requested for the token, the adversary might not gain as high a surplus as early requesting $\mathcal{TA}$ s will have traded against each other. This can be addressed by making the clock trigger to encompass the full duration of a single or multiple trade days in a random manner.

Attack Analyses: The adversary P_A is required to be computationally apt, and to establish a reliable connection with the victims in order to execute such an attack. In a resource constrained setup, this might not be feasible. To maximise on the surplus to be gained the adversary can only participate when the clock trigger occurs. If a considerable number of victim $\mathcal{TA}$ s have requested for the token, the adversary might not gain as high a surplus as early requesting $\mathcal{TA}$ s will have traded against each other. Thus, the clock trigger can be used to encompass the full duration of a single or multiple trade days in a random manner.

Algorithm 7.1: Dynamic Strategy Downgrade Attack - Adversary

Input : $WillingToTrade$, $Clock$

Output: $Revert_{msg}$, $ReqM_{mp}$

```
1  Initialisation: R = 1000, Clock = FALSE
2  /* Adversary sends a payload triggered by a clock                    */
3  repeat
4      repeat
5          Wait;
6      until Clock = TRUE;
7      if WillingtoTrade = TRUE then
8          Request TOKEN ;
9      else
10         Send Revert_msg to P − 1 victims ;
11 until termination;
```

Algorithm 7.2: Dynamic Strategy Downgrade Attack - Victim

Input : $Revert_{msg}$, $Clock$

Output: $strategy$

```
1  Initialise: R = 1000, Clock = FALSE
2  /* For each infected TA                                              */
3  for each TA ∈ TA_i[.] do
4      repeat
5          repeat
6              Wait;
7          until Clock = TRUE;
8          repeat
9              Use ZI strategy;
10         until Revert_msg = TRUE or Clock = FALSE;
11     until termination;
```

Sketch Proof: Algorithm 1.2 outputs $Revert_{msg}$ messages to the victim *TA*s if the adversary is not willing to trade and the clock triggers a strategy downgrade. The sub-algorithm terminates at the end of a trade day when $R = t_{round}$. The loop will always terminate when $Clock$ is TRUE (since the attack is clock triggered). If we consider the inner conditional statement (line 7), say S, **if c then** A_1 **else** A_2 **endif**. If c is well defined (meaning it can be evaluated), $P \wedge c \overset{A}{\rightarrow} Q$ and $P \wedge \bar{c} \overset{A}{\rightarrow} Q$, then $P \overset{A}{\rightarrow} Q$. This suggests that we verify the correctness of the branch (both when c is true and when c is false). The sub-algorithm's preconditions are $WillingTotrade = \{0, 1\}$, $clock = \{0, 1\}$ and the postcondition is a $Revert_{msg}$ or $ReqM_{mp}$. If

the condition c is $WillingToTrade = TRUE$ then a request is made for the TOKEN. In the other case, $WillingToTrade = FALSE$ holds and a $Revert_{msg}$ is sent to the victim *TA*s. In Algorithm 1.3 each victim will change strategy when the clock trigger is TRUE. This repeats until $Revert_{msg}$ is received or the clock trigger is off (FALSE).

7.4.2 Dynamic Collusion Attack

Assume adversary participants P_{Ai}, form a coalition and coordinate among themselves using an automated tool to gain additional surplus over the rest of victim participants, P_ν. We consider, $i = \{1...\eta...\kappa\}$, where η is the bound on maximum colluders that guarantee added profit on strategy switch, κ is the number of all the colluders and $P_A \subset P$ and $P_\nu \subset P$. The aim of the tool is to coordinate population ratio of P_A's to P_ν's by maintaining η. The P_ν will continue using the default strategy (in this case the AA strategy). For this *Collusion Attack* we further consider the following:

- a symmetrical relationship of buyers and sellers;
- all $\mathcal{TA}$ s get the same amount of units to trade;
- only the P_A colluders can change their strategy;
- despite random and numerous strategy changes, P_A s can seamlessly continue participating in the auction;
- the P_A colluders have computational capacity and reliable channels to coordinate the attack.

Motivation: Vach and Maršál experimental results in [96] indicate that additional surplus can be gained from changing population ratios of $\mathcal{TA}$'s from *AA* to *GDX*[3] strategy with the minority population dominating in average profit.

Attack: κ of P_A install a tool (piece of software or script) that enables only η to autonomously change agent strategy from *AA* to *GDX*. Colluders' TA is the attack points where the CDA rule allowing TAs to participate regardless of the bidding strategy they use becomes the vector of this type of attack. The colluding $\mathcal{TA}$ s use a separate channel to communicate among themselves. On receipt of the beginning of trading day signal the strategy changing automated tool will allow a number of adversary agents to shift their strategy to the *GDX*. In considering experimental evidence in literature [96], the adversary agent population ratio to the truthful agent population can be either 2 to 4 or 1 to m to ensure a high surplus on adversary population. To ensure such coordination the strategy changing automated tool can use some group MUTEX

[3]Strategy developed by Tesauro and Bredin [48] as a modification of the Gjerstad-Dickhaut (GD) strategy that uses dynamic programming to price orders.

protocol to select η number of colluders allowed to change their strategy at one particular time. The η-MUTEX algorithm will allow at most η colluders at a time to change their strategy (enter critical section). The protocol is token based and η tokens are used. Thus, a colluding $\mathcal{TA}$ can only change its strategy to GDX when it is in possession of the token. As an input the algorithm gets P; the number of all participants in the market. For instance, Chaudhuri and Edward [147] proposed one such protocol which performed on a worst case message complexity of $O(\sqrt{n})$. The single trade-day-signal that is used to trigger the selection of colluding $\mathcal{TA}$ s can be altered to a number of signals (therefore trading days) to allow the selected colluders more number of rounds to benefit before the change. Tolerance to node and link failure ensures robustness of the η-MUTEX protocol. We assert that as long as the colluders are coordinated in such a manner this attack will not deviate, with high probability; κ colluders will continuously take turns and cheat.

Formally, a dynamic collusion attack Æ (dca) in X is defined as a non-terminating procedure å ∈ Å designed to realise a finite set of intents ĩ ⊂ Ĩ (where ĩ includes coordination with other colluders $\mathcal{TA}$ s strategy component, using a κ-MUTEX to switch between strategies), aimed at domain $d \subset D$ (where d is a set of all M_mp components where colluding $\mathcal{TA}$ s are hosted), requiring a finite set of capabilities č ⊂ Č (č includes knowledge of all participating $\mathcal{TA}$ s, ability to launch a coordinating tool, connection and communication with colluding $\mathcal{TA}$ s), launched through a finite set of enabling points $p \subset P$ (p is a set of all colluding $\mathcal{TA}$ s), when X is in state $s_0 \in S_0$ (where s_0 is the initial state CDA at the start of an auction where an attack is launched) and possibly removed when X is in state $s_e \in S_e$ (where s_e is a set of all valid states the attack can be stopped or other colluders are selected).

Attacker Rational: Intuitively, a group of attackers can opt collude motivated by the significant surplus there will obtain in the CDA. In taking turns to cheat, this attack can be difficult for the system defenders to detect. Since at any time, κ colluders will cheat and obtain surplus, the system defender can be suspicious of such behaviour. A possible workaround would be to randomly allow such cheating in different periods of the auction market.

Attack Analysis: Additional computational overheads are incurred by the κ adversaries, as they need to establish a reliable connection among themselves and change their strategy in turns. No additional messages are introduced on the CDA communication channel in this attack, assuming that adversaries strictly communicate in a back channel. Using the same experimental findings that can be used to support occurrence of this collusion attack [96], we observe that if all users do not use *AA* strategy the market allocative efficiency is significantly reduced. Additionally, messages need to be exchanged among the κ P_A s. For this attack to be a success, at least η adversaries ($\kappa = \eta$) should agree to collude, otherwise additional surplus will

not be realised. This is because the ratios between AA (of victim $\mathcal{TA}$ s) and GDX (of adversary $\mathcal{TA}$ s) is such that *GDX* can not gain additional surplus.

7.4.3 Evasive Agent Attack

We consider a single adversary that employs an evasive strategy that leverages on other traders secret information such as reservation price to make better predictions of bids/asks to submit. Assuming the auction allows trade of single indivisible electrical power/energy unit. Each bidder and seller associates two values with a unit of energy —a reservation price and a bid/ask. Reservation price (limit price) is the maximum (minimum) price a bidder (seller) is willing to pay (be paid) for energy based on personal valuation and preferences. This information is private to each trading agent. An offer (bid/ask) on the other hand is the publicly declared price that a bidder (seller) is willing to pay (sell). We further consider the following assumptions:

- a symmetrical relationship of buyers and sellers;
- all $\mathcal{TA}$ s get the same amount of units to trade;
- only the P_A can change its strategy;
- despite random and/ or numerous strategy changes, the attacker P_A can seamlessly continue participating in the auction;
- the attacker P_A and the victims P_V have computational capacity and reliable channels to send their reservation price to the attacker P_A.

Motivation: Assume each agent's reservation value can be independently drawn from a cumulative distribution function (CDF) F over $[0, 1]$, where $F(0) = 0$ and $F(1) = 1$.Similar to [148] We assume $F(.)$ is strictly increasing and differentiable in the interval $[0, 1]$. The derivative of CDF, $f(\lambda)$ is then the probability density function (PDF). The adversarial node knows his reservation value and the distribution *F* of other agents. The adversarial agent tries to maximise his utility and quits the auction if the auction price goes beyond its reservation value. The private data is then used in formulating a trading pattern to obtain a trading advantage over other traders. This phenomenon is very similar to insider trading described by Kyle in [149]. Kyle uses a dynamic model of insider trading to examine the value of private information to an insider.

Attack: Similar to the aforementioned attacks, an adversary employs a tool (e.g. a piece of malware [97]) whose payload elicits victim $\mathcal{TA}$ s' private data (e.g. reservation price) and sends it to the attacker. The vector of the attack is other participants $\mathcal{TA}$ s. At the beginning of each *trading day* or a predetermined time specified by a timer, the victim $\mathcal{TA}$ s will automatically send their private information to the adversary agent. We assume the adversarial node employs

fuzzy logic where the illegally elicited private information forms part of a fuzzy set to inform some fuzzy rules. Knowledge gained from the fuzzy sets can be combined using rules to make decisions based on this information. One such strategy is proposed by He *et al.* [49]. This approach is supported by the notion that a $\mathcal{TA}$'s decision-making about bidding involves uncertainty, multiple factors, and non-determinism that are affected by the attitudes toward risk of its opponents, the nature of the market supply (demand), and the preferences of the other bidders. Formally, an evasive attack Æ(ea) in X is defined as a non-terminating procedure å ∈ Å designed to realise a finite set of intents ĩ ⊂ Ĩ (where ĩ includes gaining control of other $\mathcal{TA}$ s strategy component, eliciting their reservation price, using the data to make informed offers in the market), aimed at domain $d \subset D$ (where d is a set of all M_mp components where $\mathcal{TA}$ s are hosted), requiring a finite set of capabilities č ⊂ Č (č includes knowledge of all participating $\mathcal{TA}$ s, ability to launch a malicious code, connection and communication with victim $\mathcal{TA}$ s), launched through a finite set of attack points $p \subset P$ (p is a set of all victim $\mathcal{TA}$ s), when X is in state $s_0 \in S_0$ (where s_0 is a set of all valid CDA states where an attack can be launched) and possibly removed when X is in state $s_e \in S_e$ (where s_e is a set of all valid states the attack can be stopped).

Attack States: The attack is a sequence of two separate phases: malware injection phase followed by the malware's payload execution phase. The start state for malware injection is s and for the subsequent payload execution launched by the malware is $s^{'}$. Malware injection occurs in any of the valid states of the CDA, while payload execution is triggered by a clock. It is important that victim $\mathcal{TA}$ s are all in $s^{'}$ concurrently through a synchronisation protocol. The evasive attack does not put the system in an invalid state, which arguably makes it difficult to detect through simple defensive measures that observe occurrence of invalid states. Algorithms in 7.3 and 7.4 present the Evasive Attack from the victims' and attacker's perspective.

Attack variants In our description of the evasive attack, we only considered a single adversary. However, a variant of such an attack can have multiple non-cooperating attackers using the same attack simultaneously. Intuitively, the steps of the attack are the same but additional aspects should be considered. For instance, the re-infection of victim TAs by other attackers; attackers being victims to attacks from other attackers; the increase in computational and communication overheads from the victims' perspective; increase in complexity of concurrent attacks.

Attack Analysis: This attack results in frequent messages of victim trader's reservation price being sent to the attacker. Additional computational resources are consumed within each victim node, while additional storage and computational resources are required of the adversary.

Assuming the victim $\mathcal{TA}$ s communicate with the adversary through a back-channel, the initial message complexity of the CDA algorithm is not affected. The attack results in frequent messages being passed between the attacker and the victim $\mathcal{TA}$ s. Such activity can be observed and used to inform some irregular agent behaviour.

Algorithm 7.3: Evasive Attack - Adversary

Input : $WillingToTrade$, $Clock$

Output: R, $Conceal_{msg}$, $ReqM_{mp}$

1 Initialisation: $R = 1000$, $Clock = FALSE$
2 /* Adversary sends a payload triggered by a clock */
3 **repeat**
4 **repeat**
5 Wait;
6 **until** $Clock = TRUE$;
7 **if** $WillingtoTrade = TRUE$ **then**
8 Change to FL Strategy;
9 Request TOKEN ;
10 **else**
11 Send $Conceal_{msg}$ to $P - 1$ victims ;
12 **until** *termination*;

Algorithm 7.4: Evasive Attack - Victim

Input : $ClockInput$, $Conceal_{msg}$

Output: $Private_{msg}$

1 Initialise: $R = 1000$, $Clock = FALSE$
2 /* For each infected TA */
3 **for** *each* $TA \in TA_i[.]$ **do**
4 **repeat**
5 **repeat**
6 Wait;
7 **until** $Clock = TRUE$;
8 **repeat**
9 Send $Private_{msg}$;
10 **until** $Conceal_{msg} = TRUE$ ***or*** $Clock = FALSE$;
11 **until** *termination*;

Sketch Proof: The Algorithm 7.3 outputs $Conceal_{msg}$ messages to the victim *TA*s if the adversary is not willing to trade while the clock has triggered a strategy downgrade. The outer loop (line 3), guarantees the sub-algorithm terminates at the end of a trade day when $R = t_{round}$.

The inner loop (line 4) will always terminate when $Clock$ is TRUE (since the attack is clock triggered). If we consider the inner conditional statement (line 7), say S, **if c then** A_1 **else** A_2 **endif**. If c is well defined (meaning it can be evaluated), $P \wedge c \xrightarrow{A} Q$ and $P \wedge \bar{c} \xrightarrow{A} Q$, then $P \xrightarrow{A} Q$. We verify the correctness of the branches (both when c is true and when c is false). The sub-algorithm's preconditions are $WillingTotrade = \{0, 1\}$, $Clock = \{0, 1\}$ and the postcondition is a $Revert_{msg}$ or $ReqM_{mp}$. If the condition c is $WillingToTrade = TRUE$ then a request is made for the TOKEN. In the other case, $WillingToTrade = FALSE$ holds and a $Conceal_{msg}$ is sent to the victim *TA*s. In Algorithm 7.4 each victim will send $Private_{msg}$ when the clock trigger is TRUE. The outer loop (line 4), shows the sub-algorithm terminates at the end of a trade day, t_{round}. The first inner loop (line 5) will always terminate when $Clock$ is TRUE. This warrants victim TAs to sent until Clock is FALSE or a $Conceal_{msg}$ from the adversary is recorded.

7.4.4 Adaptive Aggressive Strategy Manipulation

Assuming an attacker has access and control of victim TA's through an automated tool (a malware, [97]) he/she can manipulate a number of components, parameters and variables involved in the bid-formulation process of a population of homogeneous victim $\mathcal{TA}$ s. The end-goal in this scenario is to influence victim $\mathcal{TA}$s behaviour to get favourable offers in the auction. For instance, victim $\mathcal{TA}$ s will be manipulated to sell (buy) energy at very low (high) offers favourable to a buying (selling) attacker. Figure 7.5 shows the components (presented in [1]) that an adversary could manipulate in order to tip the auction balance to their favour. Overall, the AA Strategy manipulation attack can take one of many forms, which include: market information attack, adaptive parameters attack, agent preferences attack. These variations are not exhaustive, but, can be helpful in understanding *AA Strategy Manipulation Attack*. For this chapter we shall consider only two randomly chosen variants of the *AA Manipulation Attack*, namely: *Market information Attack* and *Adaptive-Parameters Attack*

A. Market-Information Attack

An adversarial payload can alter the victim TA's market information input to trigger aggressive agent behaviour in the auction. Figure 7.5 shows the AA strategy's 3 main components of 'logic and reasoning'(the Equilibrium Estimator, the Bidding Layer and the Adaptive Layer) rely on market information (the current outstanding bid/ask, the equilibrium price, etc.) as input to their computations. The payload will give misrepresented market information as input to the victim $\mathcal{TA}$ s causing the victim $\mathcal{TA}$ s to make false (non-profitable or non-competitive) offers. For instance, assume the payload's input of the market conditions is a false high outstanding bid to a buyer $\mathcal{TA}$. This means the buyer $\mathcal{TA}$ s are forced to adapt their bids to the false outstanding bid. The result is a set non-competitive bids which can be profitable for the adversary. This

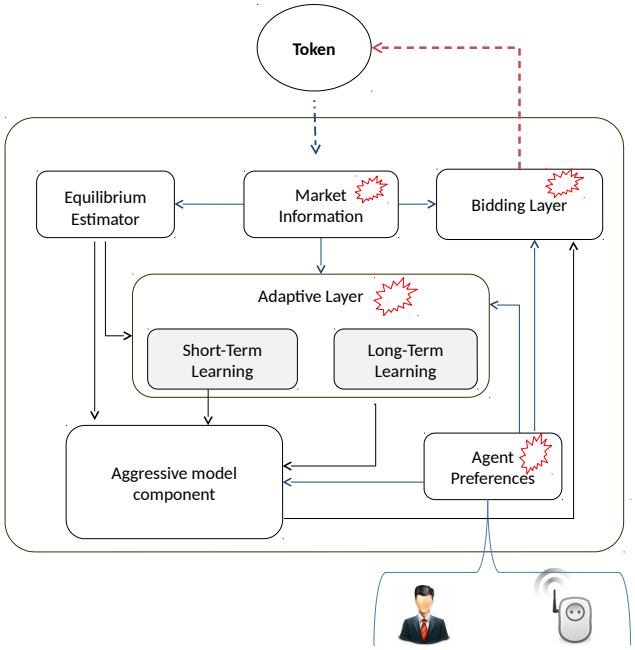

FIGURE 7.5: The AA Strategy Manipulation attack points (Adapted from Vytelingum [1])

ultimately prolongs converge of the auction market to a stable equilibrium price. Further, market efficiency is expected to be greatly reduced as a result.

B. Adaptive-Parameters Attack

An aggressive agent will submit better offers than what it believes the competitive equilibrium price to be in an attempt to improve its chances of successfully transacting. In turn, such an agent will compromise its profit margins for a chance to trade [1]. Thus, similar to the previously discussed variant of the attack, if an adversary was to employ a payload that alters short term and long term parameters of victim $\mathcal{TA}$ s, the adversary will gain an added advantage and surplus in the market. For instance, by changing the value of θ (volatility parameter) the adversary forces trading agents under his control to adapt a more aggressive bidding behaviour[4], in long term, while his own agent obtains better profitable deal. Intuitively, by failing to effectively adapt to the ever-changing market the victim agents will obtain less surplus while adversarial agent(s) benefit from this condition.

Attack: Assume a single attacker delivers a malicious code to victim $\mathcal{TA}$ s whose payload is aimed at one of the AA manipulation attack variants (Market information attack or Adaptive parameters attack). The payload can incorporate a malicious code to the victim $\mathcal{TA}$ s that

[4](Aggressive behaving agents focus on successfully bidding while trading off their profitability)

corrupts the input required for normal AA strategy functioning. A decentralised CDA ensures that market access is granted on the basis of a token which is distributed by a MUTEX protocol. Arrival of the Token (which carries the order-book and ensures a single TA mutually exclusively participates in the auction), can be a trigger that the payload will use to effect changes on the specific AA strategy component. An AA strategy attack, Æ(aasma) in X is defined as a non-terminating procedure å ∈ Å designed to realise a finite set of intents ĩ ⊂ Ĩ (where ĩ includes gaining control of other $\mathcal{TA}$ s strategy component, altering inputs of internal components of TAs, provoke bad offer creation), aimed at domain $d \subset D$ (where d is a set of all M_mp components where $\mathcal{TA}$ s are hosted), requiring a finite set of capabilities č ⊂ Č (č includes knowledge of all participating $\mathcal{TA}$ s, ability to launch a malicious code, connection and communication with victim $\mathcal{TA}$ s), launched through a finite set of attack points $p \subset P$ (p is a set of all victim $\mathcal{TA}$ s AA strategy component), when X is in state $s_0 \in S_0$ (where s_0 is a set of all valid CDA states where an attack can be launched) and possibly removed when X is in state $s_e \in S_e$ (where s_e is a set of all valid states the attack can be stopped).

Attack States: The start state for malware injection is s and for the subsequent payload execution launched by the malware is $s^{'}$. Malware injection occurs in any of the valid states of the CDA, while payload execution is triggered by the arrival of a token at the victim TA. The AA strategy manipulation attack does not put the system in an invalid state, arguably making it challenging for the system defender to detect through the occurrence of invalid states. The attack only affects the trade outcome as the CDA states remain the same. Algorithms 7.1 and 7.2 present the dynamic attack from an attacker and the victims' perspective.

Attacker Rational Similar to the strategy downgrade attack, an attacker can send $P-1$ 'revert' messages to the victims using a private back-channel directly to victims in order to evade easy detection. For this to occur, the Adversary is required to be computationally capable of establishing a reliable connection with the victims and coordinating the poisoning of their strategy component. In a resource constrained setup, this can be challenging. This attack is similar to the strategy downgrade attack in that it ensures victim $\mathcal{TA}$ s perform inefficiently in

the market, while the attacker with a better efficient strategy capitalises on this phenomenon.

Algorithm 7.5: AA Strategy Manipulation - Adversary

Input : $TokenReceived$

Output: $Relieve_{msg}$, $ReqM_{mp}$

Initialise: $R \leftarrow 1000$, $TokenReceived \leftarrow FALSE$
/* Adversary sends a payload triggered by a arroval of the TOKEN */
repeat
  repeat
    Wait;
  until $TokenReceived = TRUE$;
  if $WillingtoTrade = TRUE$ **then**
    Request TOKEN ;
  else
    Send $Relieve_{msg}$ to $P - 1$ victims ;
until *termination*;

Algorithm 7.6: AA Strategy Manipulation - Victim

Input : $TokenReceived$, $Relieve_{msg}$

Output: $offer$

Initialise: $R = 1000$, $TokenReceived = FALSE$
/* For each infected TA */
for *each* $TA \in TA_i[.]$ **do**
  repeat
    repeat
      Wait;
    until $Tokenreceived = TRUE$;
    repeat
      Manipulate inputs; // Dependent on attack variant
    until $Relieve_{msg} = TRUE$ ***or*** $TokenReceived = FALSE$;
  until *termination*;

Sketch Proof: Algorithm 1.6 and 1.7 share similar construct and logic with 1.2 and 1.3 respectively. For instance, Algorithm 1.6 has the $Relieve_{msg}$ send to P-1 victim *TA*s when the adversary is not willing to trade, as opposed to the $Revert_{msg}$ in Algorithm 1.2. Similarly, the Algorithm 1.7 creates a condition for inputs to be manipulated instead of reverting the strategy as shown in Algorithm 1.3. Correctness proof is similar to the one given in 3.1.

7.5 Summary

In this chapter we looked at a general *ACA* framework for designing cheating attacks; a framework that is simple enough to be applicable in a broad range of market mechanisms, but modular enough to be used in the design of complex attacks on such platforms. We believe such a model is important for the CDA system designer because it provides a principled approach towards the systematic engineering of such attacks which, in turn, can foster more reliable and robust attack detection and mitigation strategies. The results are important specifically for this thesis as there help us in formulating the proposed novel exception handling inspired mitigation solution.

The broad applicability and utility of the framework is demonstrated by designing cheating attacks on decentralised CDAs for constrained smart micro-grids. The examples considered are definitely not exhaustive. We do not formally analyse the impact of each attack. This is a tentatively future work direction that can be undertaken to improve on the results of this chapter. More attacks that can be constructed based on different attacker intents, for example of an attacker learning and experimenting with disruptive resources in his arsenal can also be interesting path for future work. We classify the cheating attacks in a decentralised CDA according to the attacker's capabilities (limited and advanced attackers) and the number of attackers a particular attack can have (single, multiple coordinating, and multiple non-coordinating).

The next chapter 8 seeks to use the results and findings presented in this chapter in order to formulate and develop countermeasures aimed at the proposed cheating attacks.

Chapter 8

Detection and Mitigation of Cheating Attacks

> I suppose it is tempting, if the only tool you have is a hammer, to treat everything as if it were a nail.
>
> ABRAHAM H. MASLOW

8.1 Overview

To the best of our understanding, cheating within CDAs can be resolved in one of three ways: adding cryptography scheme; modifying the auction protocol; or adding a distinct, proactive, detection and mitigation protocol. Cryptographic operations can be computationally intensive and may fail to detect undesirable behaviour such as cheating. The review in Section 3.5.3, indicates that a comprehensive solution is one based on a cryptographic model, in conjunction with proactive programs to detect cheating, and an auction format that discourages cheating. The main contribution of this chapter, is the development of a proactive protocol to detect and resolve cheating. We propose a novel exception handling (EH) inspired mitigation mechanism that employs sentinels to detect and resolve a subset of the automated cheating attacks; while yielding a reasonable performance overheads. In Section 8.2 we describe the EH approach that our mitigation solution employs. Furthermore, we modify the initial CDA algorithm's Market Execution Procedure in Section 8.2.1, then present additional procedures executed by the EH sentinels in Sections 8.2.2 and 8.2.3. In Section 8.3, we present mitigation measures to attacks analysed in Chapter 7. In Section 8.4, message and time complexity measures are used to evaluate the performance of our solution. A discussion on applicability of the EH solution within the RCSMG in Section 8.5 and a summary in Section 8.6 concludes this chapter.

8.2 Exception Handling Approach

In [27] [21], cheating attacks are shown to give rise to exceptions; situations which fall outside the normal operating conditions expected of the CDA and its components. As such cheating attacks' exceptions are resolved through *exception handling* by distinct domain-independent agents (Figure 8.1) using the citizen approach [147]. In [27] [21] we proposed two properties of the CDA that can be use the exception handlers for positive detection and mitigation. The first property is allocative efficiency — a measure of how well the market runs [70]. This measure, is given as, a ratio of the profit made during the auction to the profit that could be made if the agents traded in the most efficient way (if each offered at its private value, and the traders were matched to maximise the profits obtained). This provides an economic measure of the effectiveness of the market. For instance, the experimental results motivating the collusion cheating attack (Section 7.4), show that colluding $\mathcal{TA}$ s may gain higher *surplus*, while a decrease in the allocative efficiency will be observed. Intuitively, an exception handling mechanism should use such information, to positively detect a cheating attack and identify the culprit. The second property is the number of messages exchanged by an individual $\mathcal{TA}$. Any slight and sudden increase in the number of messages, a TA exchanges, will raise a red flag incident. In this chapter, we employ the EH approach to mitigate more cheating attacks, designed in the previous chapter. To integrate EH, the Local Market Procedure of the initial CDA algorithm 4.2 are modified and additional procedures that the sentinels will use to detect and mitigate cheating attacks are proposed.

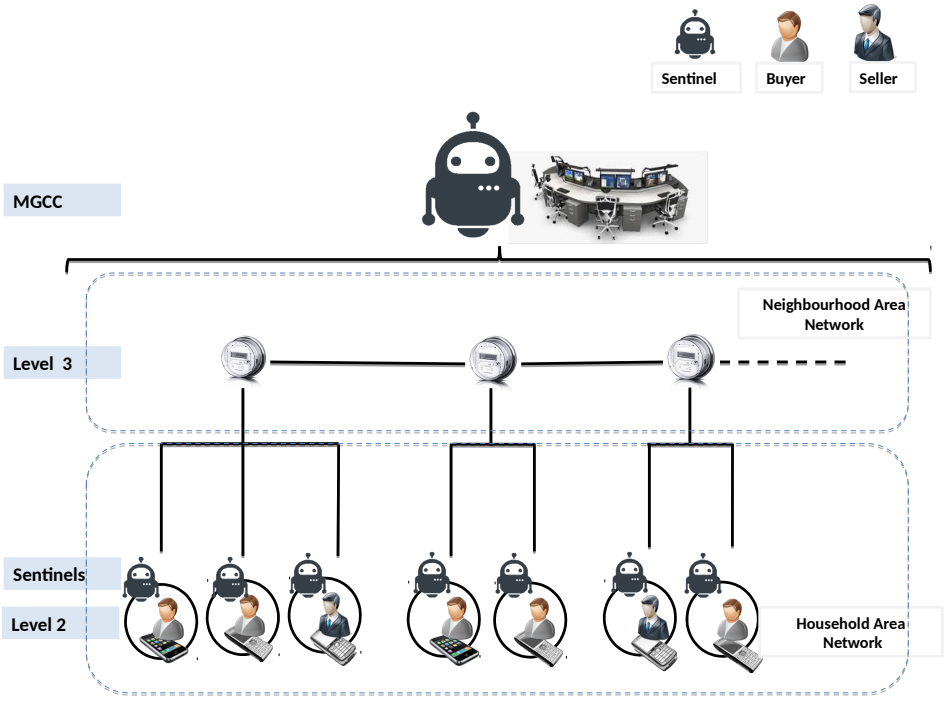

FIGURE 8.1: Exception Handling mechanism in a RCSMG

8.2.1 Local Market Procedure Extension

The effectiveness of the decentralised CDA is based on its arguably efficient token-based MUTEX protocol. EH addition does not affect the MUTEX properties since it does not interfere with the token distribution process. The token-handling protocol is not altered except for additional time. $\mathcal{TA}$ s now require additional time to be considered in the token handling timers. The initial constraints from our initial CDA algorithm are maintained. The algorithm procedure gets inputs from the central sentinel in the variables: TA_iPen (a positively identified adversary TA to be penalised), $TA_i\nu$ (a positively identified victim TA). $TA_a[.]$ represents an array of all positively identified adversary TAs. When an TA_i receives a token (line 8), and a certain TA_is have been identified as adversaries (line 9), the *TA*s in question will be penalised (line 10) and the human participant alerted (line 11). Otherwise, the 'normal' procedure described in 4.2 is executed.

Algorithm 8.1: $LocalMarketExecution$ Extension

Input : $blacklist$, $TokenReceived$, $TA_a[.]$
Output: $FlagM_{mp}$, R, $TokenCounter$, $TokenOB$, $LocalOB$

1 Initialisation: $FlagM_{sm} \leftarrow FALSE$, $TokenReceived \leftarrow FALSE$, $R \leftarrow 1000$, $TokenCounter \leftarrow 0$, $TA_i - Sentinel \leftarrow \emptyset$; `// (where i = 0, 1, 2... N)`
2 **for** *each* $TA \in TA_i[.]$ **do**
3 **if** $TokenRecieved$ = *TRUE* **then**
4 **if** $TA_i \in blacklist$ ***and*** $TA_i \in TA_a[]$ **then**
5 $Penalise[TA_i]$; `// penalised for cheating`
6 Alert Participant! ;
7 **else**
8 $TokenCounter++$; `// the market execution continues as normal`
9 Set $FlagM_{mp}$ to TRUE ;
10 Ask TA_i-Sentinel for report ;
11 Ask TA_i to $FormOffer()$; `// TAs submits its offer`
12 Execute Trade() ; `// TAs offer is accepted or rejected`
13 Update $LocalOB$ from $TokenOB$;
14 **return** $TokenOB$;
15 **else**
16 wait for $TOKEN$;

Sketch Proof: This sub-algorithm is similar to 4.2, with slight changes in lines 8-10 which capture the conditional statement allowing for the penalisation and notification of a cheating agent. If a *TA* is blacklisted and some *TA*s have been identified as victims, the blacklisted *TA* is penalised for cheating. Say, S **if c then** A_1 **else** A_2 **endif**, where c is well defined (meaning it can be evaluated), $P \wedge c \xrightarrow{A} Q$ and $P \wedge \bar{c} \xrightarrow{A} Q$, then $P \xrightarrow{A} Q$. We can verify the correctness of the branch (both when c is true and when c is false). Given the algorithm's precondition, P, is $TokenReceived = \{0, 1\}, blacklist = \{0, 1\}$ and the postcondition, Q, is a $Penalise[TA_i]$ or $TokenOB$ update, when the condition, c, that is $TA_i \in blacklist$ and $TA_i \in TA_a[.]$ is satisfied,

the $Penalise[TA_i]$ variable is set to TRUE and the participant is alerted. In the other case, the auctioning process goes as in Algorithm 1.1. This holds for any number of $TokenReceived$ or blacklisted *TA*s observed.

8.2.2 TA-Sentinel Execution

We consider that $\mathcal{TA}$-sentinels have no access to the internal state of the $\mathcal{TA}$ s there are associated with as this could open up the system to a myriad array of adversaries through sentinel compromise. We assume $\mathcal{TA}$-sentinels and the associated $\mathcal{TA}$ reside on the same M_{mp} (see Figure 8.1). Each $\mathcal{TA}$ -sentinel is capable of monitoring all messages exchanged by the associated $\mathcal{TA}$ s. Algorithm 8.2 presents a pseudocode of the TA-Sentinel procedure. Each $\mathcal{TA}$ -sentinel knows the expected maximum number of messages, u, and some irregular messages, v, that can be exchanged by the associated $\mathcal{TA}$ (line 5). Thus, if a sentinel, $TA - Sentinel_1$, monitoring say TA_i, observes an anomalous message count of $(u + v)$ messages it will red-flag TA_i (line 7) . Red-flagging is the first step in detecting an exception, allowing the $\mathcal{TA}$-sentinel to keep records of this incident for future reference (Algorithm 8.2). On arrival of the token, when $FlagM_{mp}$ is set to TRUE, a TA-sentinel will pass the report to the T-sentinel.

Algorithm 8.2: $TA - SentinelExecution$ Procedure

Input : $FlagM_{mp}$
Output: $RedFlag[TA_i]$

```
1  Initialise: u = 0, RedFlag = FALSE
2  for each TA-Sentinel ∈ TA_i-Sentinel[.] do
3      repeat
4          repeat
5              Listen for messages ;
6              if messages > u then
7                  set RedFlag = TRUE ;          // when additional messages are observed
8              else
9                  set RedFlag = FALSE ;
10         until FlagM_mp = TRUE;
11     until Termination;
```

Sketch Proof: The Algorithm 1.9 allows the TA–Sentinel to observe the messages exchanged by the *TA* it is monitoring. For each TA (line 2), the outer loop (line 3), guarantees the sub-algorithm terminates at the end of a trade day t_{round}. The inner loop (line 4) will always terminate when $FlagM_{mp}$ is TRUE (on the receipt of the TOKEN at the mobile phone M_{mp} hosting the TA). The loop ensures the TA–Sentinel listens to messages based on the condition that additional messages are observed. Say, S **if c then** A_1 **else** A_2 **endif**, where c is well defined (meaning it can be evaluated), $P \wedge c \overset{A}{\rightarrow} Q$ and $P \wedge \bar{c} \overset{A}{\rightarrow} Q$, then $P \overset{A}{\rightarrow} Q$. We can verify the correctness of the branch (both when c is true and when c is false). Given the algorithm's precondition, P,

is $FlagMmp = \{0, 1\}$ and the postcondition, Q, is a $RedFlag[TA_i]$, when the condition, c, is $messages > u$ is satisfied, the $RedFlag$ variable is set to TRUE. In the other case, $Redflag$ is set to FALSE. This holds for any number of messages observed.

8.2.3 T-Sentinel Execution

We consider a T-sentinel that is embedded into the mobile token. The token carries the auction order-book and respective records of the market, which include the current allocative efficiency and each $\mathcal{TA}$'s *surplus*. We assume, the T-sentinel has access to this information, to analyse trade history to confirm cheating of $\mathcal{TA}$ s. Algorithm 8.3 presents the pseudocode that is executed by the T-sentinel. The token is accessible to all $\mathcal{TA}$ s by virtue of the MUTEX protocol. On arrival at, say TA_i, the TA_i-sentinel will be prompted to submit a report to the T-sentinel on condition that a red-flag was recorded prior to token arrival. On reception of the report (line 6), the T-sentinel will blacklist TA_i for further enquiry at the end of the trade day (line 7). The reason is cheating resolution can only be comprehensive if adequate market data has been collected and analysed. By incorporating the T-sentinel into the order-book, our approach ensures: no additional messages are incurred. At the end of the trade day (line 5) the T-sentinel will be tasked with ascertaining the cheating cases presented.

Algorithm 8.3: $T - SentinelExecution$ Procedure

Input : $RedFlag[TA_i]$, $ExpectedSurplus$, $ExpectedEfficiency$
Output: TA_iPen, TA_iVic

```
1  Initialise: RedFlag = FALSE, blacklist = ∅, TA_iPen = ∅, TA_iVic = ∅
2  repeat
3      Receive of RedFlag ;
4      blacklist ← RedFlag[TA_i] ;
5  until termination;
6  Calculate Allocative Efficiency ;
7  if blacklist ≠  and AllocativeEfficiency < ExpectedEfficiency then
8      Cheating Confirmed! ;
9      if TA_i ∈ blacklist and ActualSurplus > ExpectedSurplus then
10         TA_i is an adversary! ;
11         TA_iPen ← TA_i ;                    // The TA is identified as an adversary
12     else if TA_i ∈ blacklisted and ActualSurplus < ExpectedSurplus then
13         TA_i is a victim! ;
14         TA_iVic ← TA_i ;                      // The TA is identified as a victim
15     else
16         False flag!;
17 else
18     No Cheating! ;
```

Sketch Proof: The Algorithm 1.10 executed by the T–Sentinel ensures receipt of *Redflag*s (which leads to blacklisting of *TA*s) and the identification of the victims and adversary by considering

the allocative efficiency, actual *surplus* and expected *surplus*. The first loop (line 2) ensures the red-flagged TAs that are identified are blacklisted until end of trading day t_{round}. Assume S, **if** c_1 **then** A_1 **elseif** c_2 **then** A_2 **else** A_3 **endif**, where c_1 and c_2 are well defined (meaning these can be evaluated), $P \wedge c_1 \overset{A}{\rightarrow} Q$ or $P \wedge c_2 \overset{A}{\rightarrow} Q$, then $P \overset{A}{\rightarrow} Q$. We can verify the correctness of the branches (when both c_1 and c_2 are true and when these are false). Given the algorithm's precondition, P, is the $RedFlag$ and the *AllocativeEfficency*, while the postcondition, Q, is a TA_iPen, TA_iVic confirmation or false flag indication. If the condition, c_1, is $TA_i \in blacklist$ **and** *ActualSurplus* $>$ *ExpectedSurplus* while condition c_2 is $TA_i \in blacklisted$ **and** *ActualSurplus* $<$ *ExpectedSurplus* are satisfied, the TA_iPen, TA_iVic can simply be confirmed and S verified. In the other case, A false flag is reported due to the condition in line 7.

8.3 Cheating Attacks Mitigation

Attacks are designed to achieve a goal inherent to cheat. This implies some financial benefit is sort by the attacker, usually at the cost of system performance. Intuitively, one way in which a defender of a decentralised CDA system can address such forms of attacks is by detecting inconsistent behaviour such as profit distribution and system performance properties. Specific $\mathcal{TA}$ *surplus* margins form the core requirement in distinguishing the cheating nodes from well behaving $\mathcal{TA}$ s. Sudden increase in *surplus* is the goal of adversarial nodes, thus making it the best parameter to use in detecting cheating.

Strategy-Downgrade Cheating If a red-flagged incident, of say, TA_i coincides with other $\mathcal{TA}$ s red-flag incidents, agent manipulation or collusion can be re-affirmed with greater probability. Intuitively, it is implied that extra messages are being exchanged by $\mathcal{TA}$ s, indicating plausible $\mathcal{TA}$ manipulation. Thus, *Strategy-Downgrade Cheating* is confirmed by:

- a sudden decrease in the market allocative efficiency;
- evidence of more than 1 blacklisted agent in previous trade rounds;
- a sudden decrease in the number of wins by red-flagged $\mathcal{TA}$ s;
- identification of an individual $\mathcal{TA}$ with a constantly higher *surplus*, while the other $\mathcal{TA}$ s have distinctly low *surplus*.

Dynamic Collusion Attack Similar to the Strategy-Downgrade Attack, adversary $\mathcal{TA}$ s can positively be identified by the extra *surplus* they gain. One evident problem of this notion is, the scheme would have to store the *surplus* margins of individual $\mathcal{TA}$ s for further inquiries. A possible workaround is, ensuring accurate *surplus* records are made and kept by the T-sentinel as soon as a successful trade is made. *Collusion Attack* is confirmed by:

- a sudden decrease in market allocative efficiency;
- a number of blacklisted agents in previous trade rounds;
- sudden increase in the number of wins by red-flagged $\mathcal{TA}$ s;
- identification of a subset of η $\mathcal{TA}$ s constantly obtaining a higher *surplus* in as many rounds.

Evasive Attack The Evasive adversary can positively be identified by a significant gain in extra *surplus* and an unusual rise in a number of messages passed by the adversary agent. *Evasive Attack* is confirmed by:

- blacklisting of 1 or more agents in previous trade rounds;
- sudden increase in the number of wins by red-flagged $\mathcal{TA}$ s;

AA Strategy Manipulation Attack Detection can follow the same notion described for the strategy downgrade attack. Thus, *AA Strategy Manipulation Attack* is confirmed by:

- a sudden decrease in the market allocative efficiency;
- blacklisting of 1 or more $\mathcal{TA}$ s in previous trade rounds;
- a sudden decrease in the number of wins by red-flagged $\mathcal{TA}$ s;
- identification of an individual $\mathcal{TA}$ with a constantly higher *surplus*, while the other $\mathcal{TA}$ s have distinctly low *surplus*.

Resolution will follow positive identification and confirmation of cheating. If a blacklisted TA_i is considered as an adversary TA_a, it will be penalised in the next trade day. If a sufficiently large amount of penalty is imposed on a discovered cheating $\mathcal{TA}$, cheating will not be profitable for the adversary. There are several methods to impose such a penalty. For example, a form of a security deposit similar to one described in [26] can be utilised. If a $\mathcal{TA}$ does not cheat, the security deposit would be returned and when caught cheating it would be confiscated. Similarly, the cheating $\mathcal{TA}$ can be disallowed from trading in the current round or subsequent rounds. Further, if a $\mathcal{TA}$ is penalised the human participant is alerted of their penalisation. In cases where the human contests the penalty, further investigation can be done by the system administrators for instance. However, this is beyond the scope our work, but can be explored as future work.

8.4 Performance Analysis

We use message and time complexity analysis to give a sketch evaluation of our solution.

Message Complexity Overall, in constructing the *EH* scheme the emphasis in on ensuring that no significant additional message overheads are incurred. Thus, a slight deviation in the message complexity from the initial CDA algorithm was expected. At the M_{sm} level the *EH* solution involves an exchange of $O(logN)$ messages to pass the token (orderbook) per market auction execution under light demand, per critical section execution. At the M_{mp} level, apart from $4n$ messages passed by $\mathcal{TA}$s to execute, the only additional messages are those between the T- and $\mathcal{TA}$ -sentinels: T-sentinel prompting $\mathcal{TA}$ to report = 1; $\mathcal{TA}$ -sentinel reporting = 1; T-Sentinel issuing a penalty = 1; resulting in $3n$ additional messages where n is the number of $\mathcal{TA}$ s. Total messages exchanged at the M_{mp} level would be $7n$. Overall, message complexity expected in light demand is therefore $\mathcal{O}(nlogN)$.

Time Complexity We consider the input of the overall algorithm to be M, that is the number of $\mathcal{TA}$ s participating in the CDA. Since operation in *T-SentinelExecution Procedure* is dominant of the two detection and resolution procedures, it is executed M times by the T-sentinel. We expect the detection and resolution algorithms to run in linear time complexity $\mathcal{O}(M)$.

8.5 Discussion

In this chapter, allocative efficiency and message complexity are employed to analyse our proposed solution. Allocative efficiency is simply a measure of how well the market runs. It is usually measured as, a ratio of the profit made during the auction, to the profit that could be made if the agents traded in the most efficient way (if each offered at its private value, and the traders were matched to maximise the profits obtained). As indicated in [27], any occurrence of cheating would significantly result in allocative efficiency decrease. Such a sudden decrease in allocative efficiency is the used as an indication of cheating. Intuitively, resolving cheating will restore the auctions high allocative efficiency. Message complexity —the number of messages sent during the auction is the second measure. As such, any decrease in efficiency, would give probable suspicion that malicious or malfunctions are consuming extra resources in a run. Thus, by combining allocative efficiency and messages transmitted within an exception handling protocol we can to offer reasonable identification of automated cheating forms. Our proposed EH scheme incurs considerably lower overheads attributed to the token-based MUTEX protocol that is used in the CDA scheme yielding a low message complexity. Due to this reason our solution may be modified to handle other exceptions manifesting from infrastructure issues like the unreliability of communication or death of agents [70]. Although our EH solution is yet to be evaluated experimentally, its integration into a CDA scheme might not significant message overheads being incurred. we base these claims from similar work by

Parsons and Klein [70] indicating in a similar double auction[1] resolution handlers performed with no significant computational overhead. This phenomenon is important if our solution is to perform in a RCSMG setup. However, one significant concern is the potential increase in the message size of the token. This is related to added functionality of the token due to the incorporation of the T-Sentinel. Thus, crafting an elegant solution that addresses this challenge and ensures performance is maintained a potential direction to explore. Furthermore, our solution guarantees identification of an adversary through *surplus* recorded after a successful trade. If *surplus* margins are manipulated an adversary may evade identification and false identification.

8.6 Summary

In this chapter, an exception handling (EH) was proposed as a tool for detection and resolution of some cheating attacks, allowing the identification and reprimand of the culprit trader. The cheating attacks give rise to exceptions, which are situations which fall outside the normal operating conditions expected of the $\mathcal{TA}$ s. We adopt a similar approach to one in [70] of employing *exception handling* by distinct domain-independent agents. The exception handling mechanism makes use of allocative efficiency and message overheads to detect and mitigate cheating forms described herein. We argue that EH could deter automated cheating attacks, while yielding low message overheads. The *EH* solution does not offer intrusion tolerance, which is, the ability of the system and its components to perform their intended function in spite of partially successful attacks. This can be an interesting direction that can be pursed as future work. We envisage that continued work in designing different types of attacks can provide a vast information for system defenders to ensure CDA algorithms realise their fullest potential within resource constrained smart micro-grids. As future work, additional validation and evaluation of the EH protocol through some theoretical and experimental evidence can be carried out. Due to the significance of CDAs in resource allocation, we believe that exploring and mitigating cheating will be interesting to micro-grid research community.

[1] The double auction has more communication overheads than the one we propose in this thesis

Chapter 9

Conclusion

> The plain fact is that there are no conclusions.
>
> JAMES JEANS, PHYSICIST.

9.1 Overview

This thesis has looked at a decentralised Continuous Double Auctioning algorithm for guaranteeing power allocation within a resource constrained smart micro-grid (RCSMG). We show that such an approach ensures scalability, efficient performance and sensitivity to constrained computational resources of the platform. A decentralised CDA can provide a vital market mechanism, with applications ranging from market-based control, decentralised resource allocation, to financial markets. With such valuable applications, understanding and improvement of efficiency, fault tolerance and security aspects of the CDA algorithms is essential. To this end, this thesis has looked at the efficiency (Chapter 4), robust (Chapter 5 and 6) and security aspects (Chapter 7 and 8) of the CDA and present research contributions towards them. In the following subsection we re-cap these contributions and match them against our original research aims in Section 1.3. Thereafter, we outline directions for future research in this area in 9.3.

9.2 Research Achievements

We began with a focus on a CDA that is decentralised and sensitive to operate in a constrained environment (see Chapter 4). It is known, the classic CDA scheme allows for efficient allocative market efficiency. First, we extend the original CDA scheme to enable decentralised auctioning.

This ensures, support of the desirable properties associated with CDA, while improving its adaptability on a RCSMG. We integrated a token-based, mutual-exclusion (MUTEX) distributive primitive, that allow a CDA to operate at a reasonably efficient time and message complexity of $\mathcal{O}(N)$ and $\mathcal{O}(logN)$ respectively, per critical section invocation (auction market execution). Such a decentralised CDA can be adopted, instead of the centralised CDAs, when the desirable properties of a decentralised mechanism are required; scalability is a major concern; and sensitivity to platform processing constraints is necessary. In addition, such a decentralised CDA scheme presents an ideal alternative in scenarios when privacy and security associated with centralised CDAs is a concern.

Next, we considered reliability and robustness of our CDA, since we envisage its use within a RCSMG, characterised by malfunctioning devices on an unreliable network (such as a lossy network). We looked at two significant fault models encompassing different fault scenarios, namely Fail-Stop and Byzantine faults (see Chapter 5). First, we modified the MUTEX protocol supporting our CDA algorithm to handle fail-stop and some Byzantine type faults of sites. We do this by using node redundancy of important cluster head nodes. The resulting algorithm yields a time complexity of $\mathcal{O}(N)$, where N is number of cluster-head nodes; and message complexity of $\mathcal{O}((logN) + W)$ time, where W is the number of check-pointing messages. In general, our extended fault tolerant CDA can allow for continued power allocation despite component and device failures, while maintaining reasonable performance overheads. This is important especially for applications in a constrained environment plagued by component failure and performance limitations. Secondly, we realised that, fault tolerance does not guarantee power allocation if the auction algorithm fails. Thus, we have proposed a decentralised consumption scheduling scheme that compliments the auctioning scheme in guaranteeing successful power allocation within the RCSMG. This work is reported in Chapter 6. We formulated the problem of scheduling power distribution on the smart micro-grid as a convex optimisation problem; with the goal of minimising the total power consumption while maximising on the social benefit of power distribution on the grid. We use the alternating direction method of multipliers (ADMM) to decompose the scheduling problem into smaller, sub problems, solved in parallel over local computation devices, yielding an optimal solution. Our scheduling solution can handle both decentralised and fully decentralised cases within the same RCSMG.

We went on to consider the issue of cheating, provoked when participants misbehave to get favourable outcomes from the auction. Such behaviour can compromise grid stability and may lead to participants pulling out. Thus, in line with our fourth research question, we look at the design and study of cheating attacks. We developed a multi-layered framework (ACA) for the systematic design of cheating attacks for our CDA and other variants, to help system defenders get an insight into developing effective mitigation solutions (see Chapter 7).

Furthermore, we developed a novel mitigation scheme to address the cheating attacks we designed towards our decentralised CDA scheme in line with our fifth research aim (see Chapter 8). We used the exception handling (EH) approach with sentinel agents using allocative efficiency and message overheads to detect and mitigate cheating forms. This result is important in showing how proactive detection and mitigation can effectively be used to solve cheating. In general, such exception handling can offer an extra security layer to deter cheating, on top of some standard security solutions like cryptography.

When taken together, the outlined contributions present an important step towards improving the structure of the CDA, while highlighting that decentralised CDA can offer valuable power allocation within constrained micro-grids. The aforementioned advances, show how we successfully managed to address the research aims we set out at the beginning of our thesis. Importantly, our work highlights new avenues that require more investigations. In the next section, we discuss the future work and potential points of departure.

9.3 Future Work

While this thesis addresses efficiency, reliability and security issues on a decentralised CDA, there still exist other areas where substantial work is required. Some of the most interesting and vital of these include the following:

- The main advancement offered by the decentralised CDA we propose, is an integration of a token-based, MUTEX for serialization of market access. Our protocol works on the assumption that, $\mathcal{TA}$ hosting mobile phones only connect to a single, specific, shared meter. In reality, one would expect, each user being allowed the flexibility to rejoin the network at another cluster. The interesting challenge is not on the mobility, but the ability to cater for mobility while effectively minimising communication and performance overheads. Thus, we intend to extend the CDA protocol to allow such mobility and flexibility without compromising the desired CDA properties, reliability and security. The aim of this extension would allow us to observe the evolution of the decentralised CDA from a reliability and security perspective.
- Furthermore, we intend to simulate the decentralised CDA, first, without mobility then later with mobility. This can be done by employing a multi-agent modelling platform such as Netlogo, due to its simplicity and suitability for modelling large collections of independent agents developing over time. In carrying out this inquiry, we seek to establish if integrating a MUTEX protocol on a standard CDA would alter auction market efficiency. Secondly, we seek to establish if mobility would significantly affect the market efficiency and possibly the computational complexity. The results from this simulation can provide a benchmark to future studies that involve a decentralised CDA such as the one we

propose. For instance, one can set out to enquire and validate the cheating attacks we proposed in Chapter 7, through changing the agent bidding strategies. Instead of using Zero Intelligence (ZI) for the agent strategies, other agent strategies such as Adaptive Aggressive strategy can be tested. The suspicion is the allocative efficiency would improve with an efficient strategy supporting why attackers would want to use such a strategy.

- We proposed a decentralised scheduling algorithm in Chapter 6. Two important issues were raised towards security and integration. We believe integrating the scheduling scheme to the auctioning scheme is a vital and interesting challenge where first the framework for integration has to be developed. One approach is to use both auctioning and scheduling alternatively to handle different power capacities, or having auctioning take care of power allocation a day before while scheduling works close to real time. We intend to find an optimal and efficient way in which the two resource allocation applications can be deployed within a constrained grid platform. From a security perspective, a framework should be developed to formalise and design some attacks on the decentralised ADMM types of algorithms. The aim of this investigation would be to find out a more encompassing and systematic way in which system defenders can better understand vulnerabilities and attacks in developing security solutions.
- In Chapter 8, we proposed an EH mechanism to detect and mitigate, automated cheating. The *EH* solution does not offer intrusion tolerance, which is, the ability of the system and its components to perform their intended function, despite partly successful attacks. We aim to further our studies in designing an intrusion tolerance scheme, that allows our decentralised CDA to run continuously, in the event cheating detection and mitigation fails. One challenge and important contribution of exploring this problem, is the development of a cheating tolerance scheme within a constrained platform. The results from such an enquiry will certainly draw attention from the auction, security community.

9.4 Last Remarks

We know about the growing popularity of the CDA and its applications as resource allocation tools, and we expect more research will continue on this technology. From such an understanding, this thesis illuminates the different areas of research on CDA for constrained environments and advances the state-of-the-art on efficiency, reliability and security.

Bibliography

[1] P. Vytelingum, "The structure and behaviour of the continuous double auction," PhD thesis, University of Southampton, UK, 2006. [Online]. Available: http://eprints.soton.ac.uk/263234/.

[2] V. L. Smith, "An experimental study of competitive market behavior," *The Journal of Political Economy*, pp. 111–137, 1962.

[3] P. R. Wurman, "Guest editor's introduction: Dynamic pricing in the virtual marketplace," *IEEE Internet Computing*, vol. 5, no. 2, pp. 36–42, 2001. DOI: 10.1109/4236.914646. [Online]. Available: https://doi.org/10.1109/4236.914646.

[4] D. Friedman, "The double auction market institution: A survey," *The double auction market: Institutions, theories, and evidence*, vol. 14, pp. 3–25, 1993.

[5] G. Platt, A. Berry, and D. Cornforth, "What role for microgrids," *Smart Grid: Integrating Renewable, Distributed & Efficient Energy*, vol. 2012, pp. 185–207, 2012.

[6] N. Hatziargyriou, N. Jenkins, G. Strbac, J. P. Lopes, J Ruela, A Engler, J. Oyarzabal, G. Kariniotakis, and A Amorim, "Microgrids–large scale integration of microgeneration to low voltage grids," *CIGRE C6-309*, pp. 1–8, 2006.

[7] A. V.D. M. Kayem, C. Meinel, and S. D. Wolthusen, "A smart micro-grid architecture for resource constrained environments," in *In proceedings of the 2017 IEEE 31st International Conference on Advanced Information Networking and Applications, AINA*, Taipeh, Taiwan: IEEE, Mar. 2017, pp. 857–864. DOI: 10.1109/AINA.2017.36.

[8] R. Das, J. E. Hanson, J. O. Kephart, and G. Tesauro, "Agent-human interactions in the continuous double auction," in *International Joint Conference on Artificial Intelligence*, LAWRENCE ERLBAUM ASSOCIATES LTD, vol. 17, 2001, pp. 1169–1178.

[9] Z. Tan and J. R. Gurd, "Market-based grid resource allocation using a stable continuous double auction," in *Proceedings of the 8th IEEE/ACM International Conference on Grid Computing*, ser. GRID '07, Washington, DC, USA: IEEE Computer Society, 2007, pp. 283–290, ISBN: 978-1-4244-1559-5. DOI: 10.1109/GRID.2007.4354144. [Online]. Available: http://dx.doi.org/10.1109/GRID.2007.4354144.

[10] Schöne, Stefan, *Auctions in the electricity market: bidding when production capacity is constrained*. Springer Science & Business Media, 2009, vol. 617.

[11] P. Vytelingum, S. Ramchurn, T. Voice, A. Rogers, and N. Jennings, "Agent-based modeling of smart-grid market operations," in *Power and Energy Society General Meeting, 2011 IEEE*, IEEE, 2011, pp. 1–8.

[12] P. Vytelingum, S. D. Ramchurn, T. D. Voice, A. Rogers, and N. R. Jennings, "Trading agents for the smart electricity grid," in *9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010), Toronto, Canada, May 10-14, 2010, Volume 1-3*, International Foundation for Autonomous Agents and Multiagent Systems, 2010, pp. 897–904. DOI: 10.1145/1838206.1838326. [Online]. Available: http://doi.acm.org/10.1145/1838206.1838326.

[13] J. Stańczak, W. Radziszewska, and Z. Nahorski, "Dynamic pricing and balancing mechanism for a microgrid electricity market," in *Intelligent Systems'2014 - Proceedings of the 7th IEEE International Conference Intelligent Systems IS'2014, September 24-26, 2014, Warsaw, Poland, Volume 2: Tools, Architectures, Systems, Applications*, Springer, 2015, pp. 793–806. DOI: 10.1007/978-3-319-11310-4_69. [Online]. Available: https://doi.org/10.1007/978-3-319-11310-4_69.

[14] A. M. C. Marufu, A. V.D. M. Kayem, and S. D. Wolthusen, "Fault-tolerant distributed continuous double auctioning on computationally constrained microgrids," in *Proceedings of the 2nd International Conference on Information Systems Security and Privacy, ICISSP 2016, Rome, Italy, February 19-21, 2016.*, SCITEPRESS, 2016, pp. 448–456, ISBN: 978-989-758-167-0. DOI: 10.5220/0005744304480456. [Online]. Available: https://doi.org/10.5220/0005744304480456.

[15] P. Jalote, *Fault Tolerance in Distributed Systems*. Prentice-Hall Inc., 1994, ISBN: 978-0-13-301367-2.

[16] G. K. Weldehawaryat, P. L. Ambassa, A. M. C. Marufu, S. D. Wolthusen, and A. V.D. M. Kayem, "Decentralised scheduling of power consumption in micro-grids: Optimisation and security," in *Security of Industrial Control Systems and Cyber-Physical Systems - Second International Workshop, CyberICPS 2016, Heraklion, Crete, Greece, September 26-30, 2016, Revised Selected Papers*, ser. Lecture Notes in Computer Science, vol. 10166, Heraklion, Greece: Springer, Sep. 2016, pp. 69–86. DOI: 10.1007/978-3-319-61437-3_5. [Online]. Available: https://doi.org/10.1007/978-3-319-61437-3_5.

[17] I. Koutsopoulos and L. Tassiulas, "Optimal control policies for power demand scheduling in the smart grid," *Selected Areas in Communications, IEEE Journal on*, vol. 30, no. 6, pp. 1049–1060, Jul. 2012, ISSN: 0733-8716. DOI: 10.1109/JSAC.2012.120704.

[18] J. Vardakas, N. Zorba, and C. Verikoukis, "A survey on demand response programs in smart grids: Pricing methods and optimization algorithms," *Communications Surveys Tutorials, IEEE*, vol. 17, no. 1, pp. 152–178, 2015, ISSN: 1553-877X. DOI: 10.1109/COMST.2014.2341586.

[19] W. Shi, X. Xie, C.-C. Chu, and R. Gadh, "Distributed optimal energy management in microgrids," *Smart Grid, IEEE Transactions on*, vol. 6, no. 3, pp. 1137–1146, May 2015, ISSN: 1949-3053. DOI: 10.1109/TSG.2014.2373150.

[20] W. Shi, N. Li, C. C. Chu, and R. Gadh, "Real-time energy management in microgrids," *IEEE Transactions on Smart Grid*, vol. PP, no. 99, pp. 1–11, 2015, ISSN: 1949-3053. DOI: 10.1109/TSG.2015.2462294.

[21] A. M. C. Marufu, A. V.D. M. Kayem, and S. D. Wolthusen, "Circumventing cheating on power auctioning in resource constrained micro-grids," in *2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, IEEE, Dec. 2016, pp. 1380–1387, ISBN: 978-1-5090-4297-5. DOI: 10.1109/HPCC-SmartCity-DSS.2016.0195.

[22] C. Wang and H.-f. Leung, "Anonymity and security in continuous double auctions for internet retails market," in *System Sciences, 2004. Proceedings of the 37th Annual Hawaii International Conference on*, IEEE, Jan. 2004, p. 10. DOI: 10.1109/HICSS.2004.1265431. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1265431.

[23] J. Trevathan, "Security, anonymity and trust in electronic auctions," *Crossroads*, vol. 11, no. 3, pp. 2–2, May 2005, ISSN: 1528-4972. DOI: 10.1145/1144396.1144398. [Online]. Available: http://doi.acm.org/10.1145/1144396.1144398.

[24] J. Trevathan, H. Ghodosi, and W. Read, "An anonymous and secure continuous double auction scheme," in *System Sciences, 2006. HICSS'06. Proceedings of the 39th Annual Hawaii International Conference on*, IEEE, vol. 6, 2006, p. 125. DOI: 10.1109/hicss.2006.45. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1579542&isnumber=33366.

[25] J. Trevathan, "Privacy and security in online auctions," PhD thesis, James Cook University, 2007.

[26] M. Yokoo, Y. Sakurai, and S. Matsubara, "The effect of false-name bids in combinatorial auctions: New fraud in internet auctions," *Games and Economic Behavior*, vol. 46, no. 1, pp. 174–188, 2004.

[27] A. M. C. Marufu, A. V.D. M. Kayem, and S. D. Wolthusen, "Power auctioning in resource constrained micro-grids: Cases of cheating," in *11th International Conference on Critical Information Infrastructures Security*, SPRINGER, 2016.

[28] ——, "A distributed continuous double auction framework for resource constrained microgrids," in *ritical Information Infrastructures Security - 10th International Conference, CRITIS 2015, Berlin, Germany, October 5-7, 2015, Revised Selected Papers*, ser. Lecture Notes in Computer Science, vol. 9578, Springer, 2015, pp. 183–198, ISBN: 978-3-319-33330-4. DOI: 10.1007/978-3-319-33331-1_15. [Online]. Available: https://doi.org/10.1007/978-3-319-33331-1_15.

[29] P. L. Ambassa, A. Kayem, S. Wolthusen, and C. Meinel, "Secure and reliable power consumption monitoring in untrustworthy micro-grids," English, in *Future Network Systems and Security*, ser. Communications in Computer and Information Science, R. Doss, S. Piramuthu, and W. ZHOU, Eds., vol. 523, Springer International Publishing, 2015, pp. 166–180, ISBN: 978-3-319-19209-3. DOI: 10.1007/978-3-319-19210-9_12. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-19210-9_12.

[30] P. L. Ambassa, S. D. Wolthusen, A. V. Kayem, and C. Meinel, "Robust snapshot algorithm for power consumption monitoring in computationally constrained micro-grids," in *Innovative Smart Grid Technologies-Asia (ISGT ASIA), 2015 IEEE*, IEEE, Nov. 2015, pp. 1–6. DOI: 10.1109/ISGT-Asia.2015.7387160.

[31] A. Sobe and W. Elmenreich, "Smart microgrids: Overview and outlook," *CoRR*, vol. abs/1304.3944, 2013. [Online]. Available: http://arxiv.org/abs/1304.3944.

[32] C. Preist and M. van Tol, "Adaptive agents in a persistent shout double auction," in *Proceedings of the first international conference on Information and computation economies*, ACM, 1998, pp. 11–18.

[33] M. Klusch, "Agent-mediated trading: Intelligent agents and e-business," *Journal on Data and Knowledge Engineering*, vol. 36, no. 3, pp. 59–76, 2001.

[34] S. H. Clearwater, R. Costanza, M. Dixon, and B. Schroeder, "Saving energy using market-based control," *Market-Based Control: A Paradigm for Distributed Resource Allocation*, pp. 253–273, 1996.

[35] R. H. Guttman, P. Maes, A. Chavez, and D. Dreilinger, "Results from a multi-agent electronic marketplace experiment," in *Poster Proceedings of the Eighth European Workshop on Modeling Autonomous Agents in a Multi-Agent World (MAAMAW'97)*, Citeseer, 1997, pp. 86–98.

[36] H. Ma and H.-F. Leung, "An adaptive attitude bidding strategy for agents in continuous double auctions," *Electronic Commerce Research and Applications*, vol. 6, no. 4, pp. 383–398, 2008. DOI: 10.1016/j.elerap.2006.12.003. [Online]. Available: https://doi.org/10.1016/j.elerap.2006.12.003.

[37] M. Wooldridge, "Intelligent agents," *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, pp. 3–44, 1999.

[38] F. Guerin and J. Pitt, "Verification and compliance testing," in *Communication in Multiagent Systems*, Springer, 2003, pp. 98–112.

[39] H. K. Nunna and S. Doolla, "Multiagent-based distributed-energy-resource management for intelligent microgrids," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 4, pp. 1678–1687, 2013. DOI: 10.1109/TIE.2012.2193857. [Online]. Available: https://doi.org/10.1109/TIE.2012.2193857.

[40] ——, "Energy management in microgrids using demand response and distributed storage- a multiagent approach," *IEEE Transactions on Power Delivery*, vol. 28, no. 2, pp. 939–947, 2013.

[41] H. K. Nunna, A. M. Saklani, A. Sesetti, S. Battula, S. Doolla, and D. Srinivasan, "Multiagent based demand response management system for combined operation of smart microgrids," *Sustainable Energy, Grids and Networks*, vol. 6, pp. 25–34, 2016.

[42] J. Rust, J. Miller, and R. Palmer, "Behavior of trading automata in a computerized double auction market," *The double auction market: Institutions, theories, and evidence*, pp. 155–198, 1993.

[43] D. K. Gode and S. Sunder, "Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality," *Journal of political economy*, vol. 101, no. 1, pp. 119–137, 1993.

[44] D. Cliff, "Minimal-intelligence agents for bargaining behaviors in market-based environments," *Hewlett-Packard Labs Technical Reports*, 1997.

[45] ——, "Evolutionary optimization of parameter sets for adaptive software-agent traders in continuous double auction markets," *HP LABORATORIES TECHNICAL REPORT HPL*, no. 99, 2001.

[46] S. Gjerstad and J. Dickhaut, "Price formation in double auctions," in *E-Commerce Agents, Marketplace Solutions, Security Issues, and Supply and Demand*, vol. 22, Elsevier, 2001, pp. 106–134. DOI: 10.1007/3-540-45370-9_7. [Online]. Available: https://doi.org/10.1007/3-540-45370-9_7.

[47] G. Tesauro and R. Das, "High-performance bidding agents for the continuous double auction," in *Proceedings 3rd ACM Conference on Electronic Commerce (EC-2001), Tampa, Florida, USA, October 14-17, 2001*, ACM, 2001, pp. 206–209. DOI: 10.1145/501158.501183. [Online]. Available: http://doi.acm.org/10.1145/501158.501183.

[48] G. Tesauro and J. L. Bredin, "Strategic sequential bidding in auctions using dynamic programming," in *The First International Joint Conference on Autonomous Agents & Multiagent Systems, AAMAS 2002, July 15-19, 2002, Bologna, Italy, Proceedings*, ACM, ACM, 2002, pp. 591–598. DOI: 10.1145/544862.544885. [Online]. Available: http://doi.acm.org/10.1145/544862.544885.

[49] M. He, Leung, Ho-fung, and N. R. Jennings, "A fuzzy-logic based bidding strategy for autonomous agents in continuous double auctions," *IEEE Transactions on Knowledge and data Engineering*, vol. 15, no. 6, pp. 1345–1363, 2003. DOI: 10.1109/TKDE.2003.1245277. [Online]. Available: https://doi.org/10.1109/TKDE.2003.1245277.

[50] P. Vytelingum, R. K. Dash, E. David, and N. R. Jennings, "A risk-based bidding strategy for continuous double auctions," in *Proceedings of the 16th Eureopean Conference on Artificial Intelligence, ECAI'2004, including Prestigious Applicants of Intelligent Systems, PAIS 2004, Valencia, Spain, August 22-27, 2004*, vol. 16, IOS Press, 2004, pp. 79–83, ISBN: 1-58603-452-9.

[51] D. Cliff, "Zip60: An enhanced variant of the zip trading algorithm," in *The 3rd IEEE International Conference on E-Commerce Technology. The 8th IEEE International Conference on and Enterprise Computing, E-Commerce, and E-Services*, IEEE, 2006, pp. 15–15. DOI: 10.1109/CEC-EEE.2006.99. [Online]. Available: https://doi.org/10.1109/CEC-EEE.2006.99.

[52] G. L. Peterson, "Myths about the mutual exclusion problem," *Information Processing Letters*, vol. 12, no. 3, pp. 115–116, 1981. DOI: 10.1016/0020-0190(81)90106-X. [Online]. Available: https://doi.org/10.1016/0020-0190(81)90106-X.

[53] S. Ghosh, *Distributed systems: an algorithmic approach*. CRC press, 2014.

[54] K. Raymond, "A tree-based algorithm for distributed mutual exclusion," *ACM Transactions on Computer Systems (TOCS)*, vol. 7, pp. 61–77, 1989. DOI: 10.1145/58564.59295. [Online]. Available: http://doi.acm.org/10.1145/58564.59295.

[55] A. D. Kshemkalyani and M. Singhal, *Distributed computing: principles, algorithms, and systems*. Cambridge University Press, 2008.

[56] V. K. Garg, *Principles of distributed systems*. Springer Publishing Company, Incorporated, 2011.

[57] M. Médard and S. S. Lumetta, "Network reliability and fault tolerance," *Encyclopedia of Telecommunications*, 2003.

[58] van Steen, Maarten and A Tanenbaum, "Distributed systems principles and paradigms," *Vrije Universiteit Amsterdam, Holland*, pp. 1–2, 2001.

[59] A. S. Tanenbaum and van Steen, Maarten, *Distributed Systems*. Prentice-Hall, 2007.

[60] B. Pourebrahimi, K. Bertels, G. Kandru, and S. Vassiliadis, "Market-based resource allocation in grids.," in *Second International Conference on e-Science and Grid Technologies (e-Science 2006), 4-6 December 2006, Amsterdam, The Netherlands*, 2006, p. 80. DOI: 10.1109/E-SCIENCE.2006.100. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/E-SCIENCE.2006.100.

[61] J. F. Kurose and R. Simha, "A microeconomic approach to optimal resource allocation in distributed computer systems," *IEEE Transactions on computers*, vol. 38, no. 5, pp. 705–717, 1989. DOI: 10.1109/12.24272. [Online]. Available: https://doi.org/10.1109/12.24272.

[62] S. H. Clearwater, *Market-based control: A paradigm for distributed resource allocation*. World Scientific, 1996.

[63] R. Wolski, J. Brevik, J. S. Plank, and T. Bryan, "Grid resource allocation and control using computational economies," *Grid computing: making the global infrastructure a reality*, vol. 772, 2003.

[64] R. Wolski, J. S. Plank, J. Brevik, and T. Bryan, "Analyzing market-based resource allocation strategies for the computational grid," *The International Journal of High Performance Computing Applications*, vol. 15, no. 3, pp. 258–281, 2001.

[65] G. Stuer, K. Vanmechelen, and J. Broeckhove, "A commodity market algorithm for pricing substitutable grid resources," *Future Generation Computer Systems*, vol. 23, no. 5, pp. 688–701, 2007. DOI: 10.1016/j.future.2006.11.004. [Online]. Available: https://doi.org/10.1016/j.future.2006.11.004.

[66] R. Buyya, D. Abramson, and S. Venugopal, "The grid economy," *Proceedings of the IEEE*, vol. 93, no. 3, pp. 698–714, 2005.

[67] P. Klemperer, "How (not) to run auctions: The european 3g telecom auctions," *European Economic Review*, vol. 46, no. 4, pp. 829–845, 2002.

[68] H. Izakian, A. Abraham, and B. T. Ladani, "An auction method for resource allocation in computational grids," *Future Generation Computer Systems*, vol. 26, no. 2, pp. 228–235, 2010. DOI: 10.1016/j.future.2009.08.010. [Online]. Available: https://doi.org/10.1016/j.future.2009.08.010.

[69] P. Paĺka, W. Radziszewska, and Z. Nahorski, "Balancing electric power in a microgrid via programmable agents auctions," *Control and Cybernetics*, vol. 41, 2012.

[70] S. Parsons and M. Klein, "Towards robust multi-agent systems: Handling communication exceptions in double auctions," in *3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004), 19-23 August 2004, New York, NY, USA*, IEEE Computer Society, vol. 3, 2004, pp. 1482–1483. DOI: 10.1109/AAMAS.2004.10224. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/AAMAS.2004.10224.

[71] M. D. De Assuncao and R. Buyya, "An evaluation of communication demand of auction protocols in grid environments," in *Proceedings of the 3rd International Workshop on Grid Economics & Business (GECON 2006)*, vol. 16, 2006.

[72] D. Grosu and A. Das, "Auction-based resource allocation protocols in grids," in *Proceedings of the 16th IASTED international conference on parallel and distributed computing and systems*, 2004, pp. 20–27.

[73] U. Kant and D. Grosu, "Double auction protocols for resource allocation in grids," in *International Symposium on Information Technology: Coding and Computing (ITCC 2005), Volume 1, 4-6 April 2005, Las Vegas, Nevada, USA*, 2005, pp. 366–371. DOI: 10.1109/ITCC.2005.135. [Online]. Available: https://doi.org/10.1109/ITCC.2005.135.

[74] P. Vytelingum, D. Cliff, and N. R. Jennings, "Strategic bidding in continuous double auctions," *Artificial Intelligence*, vol. 172, no. 14, pp. 1700–1729, 2008. DOI: 10.1016/j.artint.2008.06.001. [Online]. Available: https://doi.org/10.1016/j.artint.2008.06.001.

[75] T. Cui, Y. Wang, S. Nazarian, and M. Pedram, "An electricity trade model for microgrid communities in smart grid," in *Innovative Smart Grid Technologies Conference (ISGT)*, IEEE, 2014, pp. 1–5, ISBN: 978-1-4799-3652-6. DOI: 10.1109/ISGT.2014.6816496. [Online]. Available: https://doi.org/10.1109/ISGT.2014.6816496.

[76] S. Borenstein, M. Jaske, and A. Rosenfeld, "Dynamic pricing, advanced metering, and demand response in electricity markets," *Center for the Study of Energy Markets*, 2002.

[77] I. Koutsopoulos and G. Iosifidis, "Auction mechanisms for network resource allocation," in *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), 2010 Proceedings of the 8th International Symposium on*, IEEE, 2010, pp. 554–563.

[78] Z. Tan, "Market-based grid resource allocation using a stable continuous double auction," PhD thesis, The University of Manchester, 2007, pp. 1–173. [Online]. Available: https://pdfs.semanticscholar.org/e696/19c2fa2cd4450104cd3267338971d2a2091e.pdf.

[79] P. Anthony and N. R. Jennings, "Developing a bidding agent for multiple heterogeneous auctions," *ACM Transactions on Internet Technology (TOIT)*, vol. 3, no. 3, pp. 185–217, 2003.

[80] S. Teymouri and A. M. Rahmani, "A continues double auction method for resource allocation in economic grids," *International Journal of Computer Application*, vol. 43, pp. 7–12, 2012.

[81] B. P. Majumder, M. N. Faqiry, S. Das, and A. Pahwa, "An efficient iterative double auction for energy trading in microgrids," in *Computational Intelligence Applications in Smart Grid (CIASG), 2014 IEEE Symposium on*, IEEE, 2014, pp. 1–7.

[82] Y.-I. Chang, M. Singhal, and M. T. Liu, "A fault tolerant algorithm for distributed mutual exclusion," in *Ninth Symposium on Reliable Distributed Systems, SRDS 1990, Huntsville, Alabama, USA, October 9-11, 1990, Proceedings*, IEEE, 1990, pp. 146–154. DOI: 10.1109/RELDIS.1990.93960. [Online]. Available: https://doi.org/10.1109/RELDIS.1990.93960.

[83] D. M. Dhamdhere and S. S. Kulkarni, "A token based k-resilient mutual exclusion algorithm for distributed systems," *Information Processing Letters*, vol. 50, no. 3, pp. 151–157, 1994. DOI: 10.1016/0020-0190(94)00019-0. [Online]. Available: https://doi.org/10.1016/0020-0190(94)00019-0.

[84] J. E. Walter, J. L. Welch, and N. H. Vaidya, "A mutual exclusion algorithm for ad hoc mobile networks," *Wireless Networks*, vol. 7, no. 6, pp. 585–600, 2001. DOI: 10.1023/A:1012363200403. [Online]. Available: https://doi.org/10.1023/A:1012363200403.

[85] V. Revannaswamy and P. Bhatt, "A fault tolerant protocol as an extension to a distributed mutual exclusion algorithm," in *1997 International Conference on Parallel and Distributed Systems (ICPADS '97), 11-13 December 1997, Seoul, Korea, Proceedings*, IEEE, 1997, pp. 730–735, ISBN: 0-8186-8227-2. DOI: 10.1109/ICPADS.1997.652623. [Online]. Available: https://doi.org/10.1109/ICPADS.1997.652623.

[86] C. Gellings and J. Chamberlin, *Demand-side management: Concepts and methods*, Second. PennWell Corporation, 1993, p. 465, ISBN: 9780878146307.

[87] Y. Zhang, N. Gatsis, G. Giannakis, Y. Zhang, N. Gatsis, and G. B. Giannakis, "Robust energy management for microgrids with high-penetration renewables," *IEEE Transactions on Sustainable Energy*, vol. 4, no. 4, pp. 944–953, Oct. 2013, ISSN: 1949-3029. DOI: 10.1109/TSTE.2013.2255135.

[88] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, Jan. 2011, ISSN: 1935-8237. DOI: 10.1561/2200000016.

[89] M. Kraning, E. Chu, J. Lavaei, and S. Boyd, "Message passing for dynamic network energy management," *CoRR*, pp. 1–30, Apr. 2012. [Online]. Available: http://arxiv.org/abs/1204.1106.

[90] E. Wei and A. Ozdaglar, "On the O(1/k) convergence of asynchronous distributed alternating direction method of multipliers," *ArXiv e-prints*, Jul. 2013. arXiv: 1307.8254 [math.OC].

[91] J. Trevathan and W. Read, "Undesirable and fraudulent behaviour in online auctions.," in *SECRYPT 2006, Proceedings of the International Conference on Security and Cryptography, Setúbal, Portugal, August 7-10, 2006, SECRYPT is part of ICETE - The International Joint Conference on e-Business and Telecommunications*, vol. 6, INSTICC Press, 2006, pp. 450–458, ISBN: 972-8865-63-5.

[92] R. Porter and Y. Shoham, "On cheating in sealed-bid auctions," *Decision Support Systems*, vol. 39, no. 1, pp. 41–54, 2005. DOI: 10.1016/j.dss.2004.08.006. [Online]. Available: https://doi.org/10.1016/j.dss.2004.08.006.

[93] I. Chakraborty and G. Kosmopoulou, "Auctions with shill bidding," *Economic Theory*, vol. 24, no. 2, pp. 271–287, 2004.

[94] J. Trevathan and W. Read, "Detecting shill bidding in online english auctions," *Handbook of research on social and organizational liabilities in information security*, vol. 446, 2008.

[95] M. De Luca and D. Cliff, "Human-agent auction interactions: Adaptive-aggressive agents dominate," in *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22*, Citeseer, vol. 22, 2011, p. 178. DOI: 10.5591/978-1-57735-516-8/IJCAI11-041. [Online]. Available: https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-041.

[96] D. Vach and M. A. Marsnales, "Comparison of double auction bidding strategies for automated trading agents," Master's thesis, Charles University in Prague, 2015.

[97] M. H. R. Khouzani and S. Sarkar, "Dynamic malware attack in energy-constrained mobile wireless networks," in *Information Theory and Applications Workshop, ITA 2010, San Diego, California, USA, January 31 - February 5, 2010*, 2010, pp. 408–418. DOI: 10.1109/ITA.2010.5454093. [Online]. Available: https://doi.org/10.1109/ITA.2010.5454093.

[98] E Markatos and D Balzarotti, *The red book: The syssec roadmap for systems security research, the syssec consortium, 2013*, 2013.

[99] D. Grochocki, J. H. Huh, R. Berthier, R. Bobba, W. H. Sanders, A. A. Cárdenas, and J. G. Jetcheva, "Ami threats, intrusion detection requirements and deployment recommendations," in *Smart Grid Communications (SmartGridComm), 2012 IEEE Third International Conference on*, IEEE, 2012, pp. 395–400. DOI: 10.1109/SmartGridComm.2012.6486016. [Online]. Available: https://doi.org/10.1109/SmartGridComm.2012.6486016.

[100] A. Sedaghatbaf and M. Abdollahi Azgomi, "Attack modelling and security evaluation based on stochastic activity networks," *Security and Communication Networks*, vol. 7, no. 4, pp. 714–737, 2014.

[101] K. Sallhammar, S. J. Knapskog, and B. E. Helvik, "Using stochastic game theory to compute the expected behavior of attackers," in *2005 IEEE/IPSJ International Symposium on Applications and the Internet Workshops (SAINT 2005 Workshops), 31 January - 4 February 2005, Trento, Italy*, IEEE, 2005, pp. 102–105. DOI: 10.1109/SAINTW.2005.1619988. [Online]. Available: https://doi.org/10.1109/SAINTW.2005.1619988.

[102] M. Niitsoo, "Optimal adversary behavior for the serial model of financial attack trees.," in *Advances in Information and Computer Security - 5th International Workshop on Security, IWSEC 2010, Kobe, Japan, November 22-24, 2010. Proceedings*, Springer, 2010, pp. 354–370. DOI: 10.1007/978-3-642-16825-3_24. [Online]. Available: https://doi.org/10.1007/978-3-642-16825-3_24.

[103] S. Adepu and A. Mathur, "Generalized attacker and attack models for cyber physical systems," in *40th IEEE Annual Computer Software and Applications Conference, COMPSAC 2016, Atlanta, GA, USA, June 10-14, 2016*, IEEE, vol. 1, 2016, pp. 283–292. DOI: 10.1109/COMPSAC.2016.122. [Online]. Available: https://doi.org/10.1109/COMPSAC.2016.122.

[104] B. B. Madan and K. S. Trivedi, "Security modeling and quantification of intrusion tolerant systems using attack-response graph," *Journal of High Speed Networks*, vol. 13, no. 4, pp. 297–308, 2004.

[105] O. M. Dahl and S. D. Wolthusen, "Modeling and execution of complex attack scenarios using interval timed colored petri nets," in *Proceedings of the 4th IEEE International Workshop on Information Assurance (IWIA 2006), 13-14 April 2006, Egham, Surrey, UK*, IEEE, 2006, p. 12. DOI: 10.1109/IWIA.2006.17. [Online]. Available: https://doi.org/10.1109/IWIA.2006.17.

[106] M. Kiviharju, T. Venäläinen, and S. Kinnunen, "Towards modelling information security with key-challenge petri nets," in *Nordic Conference on Secure IT Systems*, Springer, 2009, pp. 190–206.

[107] M. K. Franklin and M. K. Reiter, *Secure auction systems*, US Patent 6,055,518, Apr. 2000.

[108] J. Trevathan and W. Read, "Cryptographic online auction schemes," in *Proceedings of IASK International Conference E-Activity and Leading Technologies*, Madrid, Spain: IASK, 2008, pp. 193–203.

[109] M. Klein, J. A. Rodriguez Aguilar, and C. Dellarocas, "Using domain-independent exception handling services to enable robust open multi-agent systems: The case of agent death," *Autonomous Agents and Multi-Agent Systems*, vol. 7, no. 1–2, pp. 179–189, 2003. DOI: 10.1023/A:1024145408578. [Online]. Available: https://doi.org/10.1023/A:1024145408578.

[110] A. Tripathi and R. Miller, "Exception handling in agent-oriented systems," in *Advances in Exception Handling Techniques (the book grow out of a ECOOP 2000 workshop)*, ser. Lecture Notes in Computer Science, Springer, 2000, pp. 128–146, ISBN: 3-540-41952-7. DOI: 10.1007/3-540-45407-1_8. [Online]. Available: https://doi.org/10.1007/3-540-45407-1_8.

[111] S. Haegg, "A sentinel approach to fault handling in multi-agent systems," in *Australian Workshop on Distributed Artificial Intelligence*, Springer, 1996, pp. 181–195. DOI: 10.1007/BFb0030090. [Online]. Available: https://doi.org/10.1007/BFb0030090.

[112] M. Klein, "Exception handling in process enactment systems," *MIT Center for Coordination Science: Cambridge MA*, 1997.

[113] M. Klein and C. Dellarocas, "Exception handling in agent systems," in *Proceedings of the third annual conference on Autonomous Agents*, ACM, 1999, pp. 62–68. DOI: 10.1145/301136.301164. [Online]. Available: http://doi.acm.org/10.1145/301136.301164.

[114] F. Souchon, C. Dony, C. Urtado, and S. Vauttier, "Improving exception handling in multi-agent systems," in *Software Engineering for Multi-Agent Systems II, Research Issues and Practical Applications [the book is a result of SELMAS 2003]*, Springer, 2003, pp. 167–188. DOI: 10.1007/978-3-540-24625-1_10. [Online]. Available: https://doi.org/10.1007/978-3-540-24625-1_10.

[115] A. U. Mallya and M. P. Singh, "Modeling exceptions via commitment protocols," in *4th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005), July 25-29, 2005, Utrecht, The Netherlands*, ACM, 2005, pp. 122–129. DOI: 10.1145/1082473.1082492. [Online]. Available: http://doi.acm.org/10.1145/1082473.1082492.

[116] I. Cakirlar, E. E. Ekinci, and O. Dikenelli, "Exception handling in goal-oriented multi-agent systems," in *Engineering Societies in the Agents World IX, 9th International Workshop, ESAW 2008, Saint-Etienne, France, September 24-26, 2008, Revised Selected Papers*, Springer, 2008, pp. 121–136, ISBN: 978-3-642-02561-7. DOI: 10.1007/978-3-642-02562-4_7. [Online]. Available: https://doi.org/10.1007/978-3-642-02562-4_7.

[117] M. Singhal, "A dynamic information-structure mutual exclusion algorithm for distributed systems," in *9th International Conference on Distributed Computing Systems, ICDCS 1989, Newport Beach, CA, USA, June 5-9, 1989*, IEEE, 1989, pp. 70–78. [Online]. Available: https://doi.org/10.1109/ICDCS.1989.37932.

[118] A. Gravey and A. Dupuis, "Performance evaluation of two mutual exclusion distributed protocols via markovian modeling," in *Proceedings of the Sixth IFIP Workshop on Protocol Specification, Testing, and Verification*, 1987, pp. 335–346.

[119] A. Dupuis, G. Hebuterne, and J.-M. Pitie, "A comparison of two mutual-exclusion algorithms for computer networks," *Journal of Systems and Software*, vol. 6, no. 1-2, pp. 137–145, 1986.

[120] D. Agrawal and A. Elabbadi, "A token-based fault-tolerant distributed mutual exclusion algorithm," *Journal of Parallel and Distributed Computing*, vol. 24, no. 2, pp. 164–176, 1995. DOI: 10.1006/jpdc.1995.1016. [Online]. Available: https://doi.org/10.1006/jpdc.1995.1016.

[121] W. Fokkink, *Distributed Algorithms: An Intuitive Approach*. MIT Press, 2013.

[122] V. Kekatos and G. Giannakis, "Distributed robust power system state estimation," *IEEE Transactions on Power Systems*, vol. 28, no. 2, pp. 1617–1626, May 2013, ISSN: 0885-8950. DOI: 10.1109/TPWRS.2012.2219629.

[123] G. Weldehawaryat and S. Wolthusen, "Secure distributed demand projection in microgrids," in *Global Information Infrastructure and Networking Symposium (GIIS)*, Oct. 2015, pp. 1–6. DOI: 10.1109/GIIS.2015.7347177.

[124] P. Yang, P. Chavali, E. Gilboa, and A. Nehorai, "Parallel load schedule optimization with renewable distributed generators in smart grids," *Smart Grid, IEEE Transactions on*, vol. 4, no. 3, pp. 1431–1441, Sep. 2013, ISSN: 1949-3053. DOI: 10.1109/TSG.2013.2264728.

[125] I. Koutsopoulos and L. Tassiulas, "Control and optimization meet the smart power grid: Scheduling of power demands for optimal energy management," in *Proceedings of the 2Nd International Conference on Energy-Efficient Computing and Networking*, ser. e-Energy '11, New York, New York, USA: ACM, 2011, pp. 41–50, ISBN: 978-1-4503-1313-1. DOI: 10.1145/2318716.2318723.

[126] D. O'Neill, M. Levorato, A. Goldsmith, and U. Mitra, "Residential demand response using reinforcement learning," in *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*, Oct. 2010, pp. 409–414. DOI: 10.1109/SMARTGRID.2010.5622078.

[127] N. Li, L. Chen, and S. Low, "Optimal demand response based on utility maximization in power networks," in *Power and Energy Society General Meeting, 2011 IEEE*, Jul. 2011, pp. 1–8. DOI: 10.1109/PES.2011.6039082.

[128] R. Urgaonkar, B. Urgaonkar, M. J. Neely, and A. Sivasubramaniam, "Optimal power cost management using stored energy in data centers," in *Proceedings of the ACM SIGMETRICS Joint International Conference on Measurement and Modeling of Computer Systems*, ser. SIGMETRICS 11, San Jose, California, USA: ACM, 2011, pp. 221–232, ISBN: 978-1-4503-0814-4. DOI: 10.1145/1993744.1993766. [Online]. Available: http://doi.acm.org/10.1145/1993744.1993766.

[129] H. Wang and J. Huang, "Bargaining-based energy trading market for interconnected microgrids," in *Communications (ICC), 2015 IEEE International Conference on*, Jun. 2015, pp. 776–781. DOI: 10.1109/ICC.2015.7248416.

[130] D. Gabay and B. Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite element approximation," *Computers & Mathematics with Applications*, vol. 2, no. 1, pp. 17–40, 1976, ISSN: 0898-1221. DOI: http://dx.doi.org/10.1016/0898-1221(76)90003-1. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0898122176900031.

[131] W. Deng, M.-J. Lai, Z. Peng, and W. Yin, "Parallel multi-block ADMM with o(1/k) convergence," *Journal of Scientific Computing*, pp. 1–25, 2016, ISSN: 1573-7691. DOI: 10.1007/s10915-016-0318-2.

[132] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in Optimization*, vol. 1, no. 3, pp. 127–239, Jan. 2014, ISSN: 2167-3888. DOI: 10.1561/2400000003.

[133] L. Liu and Z. Han, "Multi-block ADMM for big data optimization in smart grid," in *2015 International Conference on Computing, Networking and Communications (ICNC)*, Feb. 2015, pp. 556–561. DOI: 10.1109/ICCNC.2015.7069405.

[134] R. Zhang and J. Kwok, "Asynchronous distributed ADMM for consensus optimization," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014, pp. 1701–1709.

[135] J. R. Albrecht, C. Tuttle, A. C. Snoeren, and A. Vahdat, "Loose synchronization for large-scale networked systems.," in *USENIX Annual Technical Conference, General Track*, 2006, pp. 301–314.

[136] A. Haque, S. M. Alhashmi, and R. Parthiban, "Continuous double auction in grid computing: An agent based approach to maximize profit for providers," in *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, IAT 2010, Toronto, Canada, August 31 - September 3, 2010*, IEEE, vol. 2, 2010, pp. 347–351. DOI: 10.1109/WI-IAT.2010.105. [Online]. Available: https://doi.org/10.1109/WI-IAT.2010.105.

[137] Z. Zhang, F. Nait-Abdesselam, and P.-H. Ho, "Boosting markov reward models for probabilistic security evaluation by characterizing behaviors of attacker and defender," in *Availability, Reliability and Security, 2008. ARES 08. Third International Conference on*, IEEE, 2008, pp. 352–359.

[138] D. Gollmann, P. Gurikov, A. Isakov, M. Krotofil, J. Larsen, and A. Winnicki, "Cyber-physical systems security: Experimental analysis of a vinyl acetate monomer plant," in *Proceedings of the 1st ACM Workshop on Cyber-Physical System Security,CPSS 2015, Singapore, Republic of Singapore, April 14 - March 14, 2015*, ACM, 2015, pp. 1–12. DOI: 10.1145/2732198.2732208. [Online]. Available: http://doi.acm.org/10.1145/2732198.2732208.

[139] P. Liu, W. Zang, and M. Yu, "Incentive-based modeling and inference of attacker intent, objectives, and strategies," *ACM Transactions on Information and System Security (TISSEC)*, vol. 8, no. 1, pp. 78–118, 2005. DOI: 10.1145/1053283.1053288. [Online]. Available: http://doi.acm.org/10.1145/1053283.1053288.

[140] A. Teixeira, D. Pérez, H. Sandberg, and K. H. Johansson, "Attack models and scenarios for networked control systems," in *Proceedings of the 1st International Conference on High Confidence Networked Systems*, ser. HiCoNS '12, Beijing, China: ACM, 2012, pp. 55–64, ISBN: 978-1-4503-1263-9. DOI: 10.1145/2185505.2185515. [Online]. Available: http://doi.acm.org/10.1145/2185505.2185515.

[141] A. Teixeira, I. Shames, H. Sandberg, and K. H. Johansson, "A secure control framework for resource-limited adversaries," *Automatica*, vol. 51, pp. 135–148, 2015. DOI: 10.1016/j.automatica.2014.10.067. [Online]. Available: https://doi.org/10.1016/j.automatica.2014.10.067.

[142] T. Amenaza, "Fundamentals of capabilities-based attack tree analysis," *Calgary, Canada, November*, 2005.

[143] A. Buldas, P. Laud, J. Priisalu, M. Saarepera, and J. Willemson, "Rational choice of security measures via multi-parameter attack trees," *Critical Information Infrastructures Security*, pp. 235–248, 2006.

[144] T. M. Chen, J. C. Sanchez-Aarnoutse, and J. Buford, "Petri net modeling of cyber-physical attacks on smart grid," *IEEE Transactions on Smart Grid*, vol. 2, no. 4, pp. 741–749, 2011.

[145] B. Chen, Z. Kalbarczyk, D. M. Nicol, W. H. Sanders, R. Tan, W. G. Temple, N. O. Tippenhauer, A. H. Vu, and D. K. Yau, "Go with the flow: Toward workflow-oriented security assessment," in *Proceedings of the 2013 workshop on New security paradigms workshop*, ACM, 2013, pp. 65–76.

[146] S. Jajodia and S. Noel, "Advanced cyber attack modeling analysis and visualization," DTIC Document, Tech. Rep., 2010.

[147] P. Chaudhuri and T. Edward, "An algorithm for k-mutual exclusion in decentralized systems," *Computer Communications*, vol. 31, no. 14, pp. 3223–3235, 2008. DOI: 10.1016/j.comcom.2008.05.009. [Online]. Available: https://doi.org/10.1016/j.comcom.2008.05.009.

[148] M. Jenamani, Y. Zhong, and B. Bhargava, "Cheating in online auction–towards explaining the popularity of english auction," *Electronic Commerce Research and Applications*, vol. 6, no. 1, pp. 53–62, 2007. DOI: 10.1016/j.elerap.2005.12.002. [Online]. Available: https://doi.org/10.1016/j.elerap.2005.12.002.

[149] A. S. Kyle, "Continuous auctions and insider trading," *Econometrica: Journal of the Econometric Society*, pp. 1315–1335, 1985.