Model Checking the Interplay of Trust and Commitments in Multi-Agent Systems and Applications

Narges Baharloo

A Thesis

In

The Concordia Institute for Information and Systems Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy (Information Systems Engineering) at

Concordia University

Montréal, Québec, Canada

March 2024

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Narges Baharloo**

Entitled: **Model Checking the Interplay of Trust and Commitments in Multi-Agent Systems and Applications**

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Information Systems Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____ Chair
Dr. Rodolfo Coutinho

_____ External Examiner
Dr. Muhammad Younas

_____ Examiner
Dr. Juergen Rilling

_____ Examiner
Dr. Rachida Dssouli

_____ Examiner
Dr. Farnoosh Naderkhani

_____ Supervisor
Dr. Jamal Bentahar

Approved by _____
Dr. Jun Yan, Graduate Program Director

April 30, 2024 _____
Dr. Mourad Debbabi, Dean Gina Cody School of Engineering and Computer Science

ABSTRACT

Model Checking the Interplay of Trust and Commitments in Multi-Agent Systems and Applications

Narges Baharloo, Ph.D.

Concordia University, 2024

Effective and secure communication among agents is crucial for developing open multi-agent systems (MASs). In such systems, agents operate autonomously and engage in interactions within unpredictable and dynamic environments, making it essential to contemplate security-driven, social and communicative modalities. The concepts of trust and social commitments have garnered considerable attention within MASs to model flexible and secure communication mechanisms. Numerous attempts have been made to define their semantics separately. Nevertheless, there is still a need for more in-depth exploration of the connection between trust and social commitments.

On the other hand, the combination of IoT and ad hoc networks has introduced novel service models in various multi-agent applications. However, effective communication remains the essential factor in enabling the coordination and interaction among various components within these systems. This cooperation empowers them to work together to address challenges that may exceed the capabilities of individual intelligent elements. One primary concern faced by applications based on IoT-ad hoc networks applications is ensuring the reliability of their components, especially when dealing with entities that may engage in malicious behavior and uncertainty.

Moreover, in many situations, trust falls along a continuum, with various levels of strength and weakness. The level of trust often depends on factors such as past experiences,

evidence of reliability, transparency, and communication. Critical infrastructure systems, like banking and healthcare, rely on strong trust in data security and privacy. Systems with strong trust are often entrusted with significant responsibilities and critical tasks.

This thesis addresses these challenges through three main verification approaches: 1) model checking combined trust and commitments framework; 2) verifying three-valued trust model for IoT-ad hoc Systems; and 3) model checking weak and strong trust over commitments in MASs. In the first approach, we develop a formal and practical framework, TCTLC, for handling trust over social commitments. We extend the Model Checker for Multi-Agent Systems (MCMAS) with the MCMAS-TC tool and analyse its time and space complexity. We also validate the technique through an industrial case study. In the second approach, we propose a 3v trust model using three-valued logic to address trust over social commitments in uncertain IoT-ad hoc settings. We introduce the 3v-TCTLC modeling language to handle uncertainty in trust assessments. Moreover, we enhance the "MACMAS-interactor" tool to enable its interaction with the MCMAS-TC model checker by incorporating new functionalities to handle our 3v-TCTLC logic. This approach is validated through case studies in smart health monitoring and smart home systems, verifying system models against specified criteria in uncertain contexts. The third approach introduces a novel logic, T^{ws}CTLC, designed to capture properties related to both weak and strong trust over commitments. We establish the semantics over the extended interpretation of the original multi-agent systems formalism and provide postulates with accompanying proofs to support our logic. The approach is validated using a scalable case study.

These contributions aim to advance trust and commitment management in MASs and enhance the reliability and effectiveness of IoT in smart environments, providing practical tools and methodologies for uncertain scenarios.

ACKNOWLEDGEMENTS

First and foremost, I offer my deepest gratitude to God for granting me the strength, wisdom, and perseverance to complete this research.

I am sincerely grateful to my supervisor, Dr. Jamal Bentahar, for his dedicated guidance, timely advice, and scientific insights, which played a vital role in the successful completion of my Ph.D. studies.

I extend my heartfelt appreciation to my committee members and external examiner for their continuous support, inspirational suggestions, and kind enthusiasm throughout my research, which greatly contributed to the completion of my thesis.

I would like to express my gratitude to my husband, Dr. Mohammad Maleki, for his constant encouragement during my research journey.

I am deeply thankful to my parents, my only daughter, Fatemeh, who provide unending inspiration, and my fellow research lab colleagues for their unwavering support. Additionally, I would like to acknowledge Concordia University for providing me with the necessary research facilities that have been crucial for conducting this work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

MAS Multi-Agent System

CTL Computation Tree Logic

CTLC Computation Tree Logic with Commitments

CTL^{cc} Computation Tree Logic with conditional commitments

BDD Binary Decision Diagram

OBDD Ordered Binary Decision Diagram

ISPL Interpreted Systems Programming Language

LTL Linear Temporal Logic

MCMAS Model Checker for Multi-Agent Systems

NuSMV New Symbolic Model Verifier

PRISM PRobabilistIc Symbolic Model checker

PSPASE Polynomial Space

SMV Symbolic Model Verifier

SPIN Simple Promela INterpreter

P Polynomial time

PSPACE Polynomial Space

MCMAS-TC Model Checker for Multi-Agent Systems with Trust and Commitments

Chapter 1

Introduction

In this chapter, we explore the research context and articulate the problem statements, leading us to formulate the research questions. Subsequently, we outline the PhD research objectives that we aim to achieve. The chapter concludes by presenting the structure and organization of the thesis.

1.1 Context of Research

Agent communication is a key element towards the creation of open multi-agent systems (MASs). Due to the autonomous nature of agents in these systems, and the fact that these agents must interact in unreliable and dynamic environments, reasoning about trust and commitments is highly desirable. Indeed, several definitions of trust in MASs have been proposed [44, 55, 57, 92]. For instance, in [55] the purpose of trust is to ensure that a specific action is performed within a relationship between agents. On the other hand, social commitments are used to define satisfied contracts between agents but violation and cancellation could be possible [24, 25, 50, 90]. The concepts of trust and commitments involve interactions between multiple agents. There has been sustained interest in the

modeling of these two key concepts of MASs. Previous studies have focused on defining the semantics of trust and social commitments as individual entities. Nonetheless, [92] combined temporal modalities of linear temporal logic (LTL) with modalities for commitments and trust. A model-theoretic semantics along with reasoning postulates are also presented. The presented semantics is interpreted using the universal grammar of Montague [78], where a word is assigned to a power-set of worlds in this system. However, verifying such a semantics using model checking is still challenging. Consequently, reasoning about the relation between trust and commitment in a Kripke-based semantics and interpreted systems is an objective yet to be reached. In fact, in various business applications, agents need to reason on their trust, negotiate, exchange and reason regarding commitments at the same time especially when they communicate with each other. Despite the fact that trust information related to an agent is not publicly verifiable when it is defined using private beliefs, agents are still required to engage in trust reasoning when making commitments to one another. For example, an agent should be able to decide when to trust what the other agent commits about. Thus, trust and social commitments are not independent and they should definitely interact and co-exist in agent-based systems. For instance, customers may trust merchants to deliver items after sending payments. However, mere trust is not sufficient for customers to act accordingly. Customers also require corresponding commitments from the specific merchants. Trust and commitments are orthogonal concepts; they do not necessarily imply each other. Customers can act based on either trust or commitments, but ideally, both should be involved and work together. Commitments cannot be considered superfluous if trust is established, and trust cannot be misplaced if there are commitments attached to it [93]. To motivate our efforts to model and combine trust and commitments into a single framework, we explore an electronic commerce example that arises in practical settings.

Example 1. We consider the interactions between customers and merchants. Specifically, we examine the importance of trust in the transaction process when a customer accepts a price and commits to making a payment. In this scenario, the merchant must trust that the customer will fulfill their commitment, just as the customer must trust that the merchant will deliver the requested items as promised. To provide an illustration, let us examine a scenario in which the customer approaches the merchant with a request to buy certain items. After the payment has been successfully processed, the customer has fulfilled their payment commitment. The merchant, on the other hand, trusts the customer to have made the commitment and allows the request to proceed. At this point, the merchant can initiate the delivery process and promptly inform the customer.

Hence, the need for a tool that expresses the interaction between trust and commitments is confirmed. Model checking has attracted many contributions with industrial implications [6, 26, 41, 44, 72]. Therefore, in this thesis, we show how the relation between trust and commitments can be represented in a logical-based formalism by expanding the Computation Tree Logic (CTL), the classic branching time logic [38,51], with modalities about combined trust and commitment to specify protocol properties for multi-agent interactions.

Another issues that have captured our interest are the Internet of Things (IoT) and ad hoc networks, which have emerged as crucial technologies in a wide range of multi-agent applications deployed within dynamic and uncertain physical environments. In recent years, these technologies have attracted tremendous research efforts and are the subject of extensive ongoing investigations [82, 84, 86]. The integration of IoT with ad hoc networks has introduced innovative service paradigms across a variety of multi-agent applications [96]. These applications span various sectors, including autonomous transportation, healthcare systems, military monitoring tasks, and diverse smart applications [4, 75, 112]. Although the growing number of these applications made them even more appealing

to users and significantly contributed to their economic success, it raises, however, a number of challenges related to their present and future behaviors. Indeed, IoT devices and networks are susceptible to vulnerabilities arising from uncertainty and incomplete information, which undermine the reliability of communication due to nature of its components complexity and heterogeneity. Therefore, there is a pressing need for effective and reliable detection to identify any weaknesses in these systems in order to achieve their full potential.

The Internet of Things (IoT) encompasses the integration of physical objects that can be equipped with electronics, software, embedded sensors, actuators, and network connectivity, enabling them to collect and exchange many data from the environment without human intervention [66]. IoT is not a single technology, but rather a combination of diverse technologies that collaborate harmoniously. On the other hand, ad hoc networks play a crucial role as one of the key technologies in the domain of IoT. They have attracted considerable interest as a novel structure of decentralized wireless communication networks that deviates from traditional centralized communication. However, the current ad hoc networks in IoT still face significant challenges in accurately modeling the behaviors of its components. Particularly, when it comes to fully understanding the behavioral patterns of other network participants.

Nevertheless, communication is the key aspect for facilitating the coordination and interaction of various components within these systems, enabling them to collectively address obstacles that may be beyond the capabilities of individual intelligent components. One of the main issues that encounter IoT-ad hoc network based applications is how to ensure the reliability of their components in the presence of misbehaving entities. Such entities not only create an exception for other entities, but also may obstruct their proper work [4]. In fact, due to weak connectivity, resource limitations, and limited physical protection, security emerges as a significant concern. The fact that these components

operate autonomously and must interact with one another in unreliable environments, makes the need to verify the presence of un-trusted participants highly desired.

Trust over commitments based solutions are widely regarded as a promising approach to mitigate the presence of intentionally or unintentionally generated bogus or poor-quality results. Evaluating the trustworthiness of a component serves as a reliable metric to facilitate the decision-making process. Moreover, commitments offer social and observable meaning that can be applied to capture the dynamic messages in these systems. Modeling social commitments is of great importance for IoT-ad hoc network based applications that might not always lead to the fulfillment of the commitment actions.

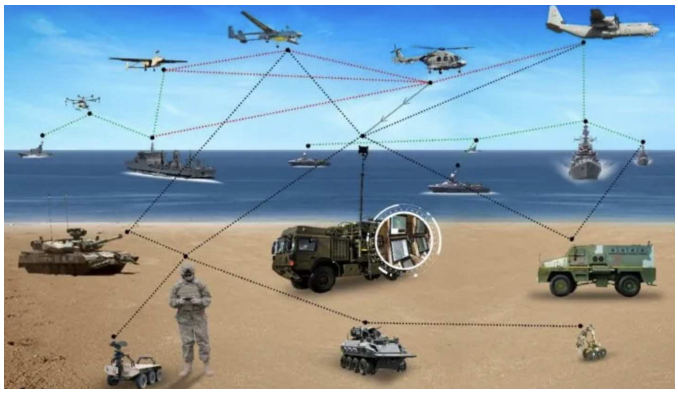

Figure 1.1: Overview of a military force system

Let us consider a scenario wherein a multinational military force is deployed in a war zone. In such a critical situation, it becomes crucial for the diverse components of the force to establish effective and secure communication channels among themselves, ensuring the integrity of information without any risk of compromise. To achieve this, the force can employ secure techniques for data transmissions, enabling real-time coordination, information sharing, and decision-making among its components [52]. Therefore, in these situations, two important considerations come to light: (1) trust emerges as a key factor in making well-informed decisions about communication partners, and (2) successful

partnerships are built on the foundation of social commitments within satisfied contracts. Figure 1.1 shows overview of the considered military force system.

Formal verification techniques stand as a pivotal technique for identifying potential vulnerabilities in IoT-ad hoc network systems. Model-checking [37] is one of these techniques used to ensure that IoT-ad hoc based systems meet a given specification [39, 54, 113]. It helps to identify potential vulnerabilities in these systems and to test different security protocols. The approach consists of three essential elements: a concrete system model represented by a finite state model, as well as formal properties denoted as model-driven specifications expressed in the form of propositional temporal logic. Additionally, there exists a verification process aimed at assessing whether the system fulfills the given specifications. When the outcomes is a "true" response, it signifies that the model fulfills the specification. Otherwise, a counterexample is generated that exposes the sequence of steps where an error occurred. The use of model checking in these systems is particularly significant because these systems may have to operate under different network conditions, with different types of devices, and with different types of data. Thus, it can help to ensure that the system is robust enough to handle the challenges and that it can operate reliably and securely.

Many approaches aiming to provide social modeling for trust and commitments among IoT-ad hoc network based applications can be found in the existing literature [14, 42, 46, 80]. However, very few approaches addressed the uncertainty from a high level abstraction viewpoint. Verifying social trust over commitments in systems that are treated under the assumption of uncertainty is another important aspect to be addressed, especially, when they are developed in critical applications. Classical logical frameworks that handle trust on commitments in IoT-ad hoc systems have traditionally employed a binary approach, where we either fully fulfil or violate the behavior of a component (i.e., (T, F)). However, in many contexts, it is quite difficult to determine, with absolute certainty, whether a property

about the behavior of an entity is true or false. The three-valued logic is an extension of classical logic that includes extra truth values that use a 3-valued lattice of truth values (T, M, F), with M representing undefined or uncertain information.

Thus, Online interactions are marked by uncertainty and, additionally, the anonymity of the interaction participants. Consequently, there is no assurance that this process will be definitively fulfilled in specific applications. Hence, there is a substantial need for formally defining and automatically verifying trust and commitments interactions among autonomous agents under uncertainty.

We have also introduced and discussed the concepts of weak and strong trust, which are novel aspects in the realm of trust, aiming to enhance the depth of our research. As trust is a complex and important element in human relationships, playing a crucial role in shaping how people and groups interact. It can be broadly categorized into two main types: Weak trust and strong trust are like two opposite ends of a spectrum, each having its unique qualities and significant consequences.

Weak trust is often characterized by its superficial and temporary nature, typically based on limited interactions or experiences with a person or entity. This type of trust is fragile and can be easily broken due to misunderstandings, unmet expectations, or changes in circumstances. Weak trust is commonly seen in new relationships, where there is not enough history or shared experiences to form a solid foundation of trust. In situations involving weak trust, individuals tend to be cautious and skeptical, seeking verification and assurances before fully committing to a relationship or agreement. This is often observed in business transactions with new partners, interactions with unfamiliar individuals, or in online interactions. Weak trust requires constant validation and reassurance to be maintained, and even then, it remains vulnerable to being compromised. On the other hand, strong trust is built over time through consistent and positive interactions, shared experiences, and a history of reliability and integrity. This type of trust is robust and

resilient, able to withstand challenges and conflicts that may arise. Strong trust is typically found in committed partnerships, where individuals have had the opportunity to prove their trustworthiness and demonstrate their commitment to the relationship.Strong trust is characterized by a deep sense of security, confidence, and mutual respect, allowing for open communication, vulnerability, and collaboration. In relationships with strong trust, individuals feel comfortable sharing their thoughts, feelings, and experiences, knowing that they will be understood and supported. This level of trust fosters a positive environment where individuals can thrive, grow, and work together towards common goals.

1.2 Research Questions

This research emphasizes the significance of trust and commitments in open multi-agent systems, particularly in the context of the Internet of Things (IoT) and ad hoc networks. Trust is crucial for ensuring reliable interactions among autonomous agents, while commitments establish the foundation for fulfilling agreements between agents. The research proposes using formal verification techniques like model checking to identify vulnerabilities and ensure that systems meet specified requirements, especially in uncertain and dynamic environments.

Furthermore, it suggests extending traditional binary logic with a three-valued logic to handle uncertainty, acknowledging that in certain scenarios, absolute certainty about agent behavior can be challenging to determine.

Overall, current research highlights the importance of trust and commitments in multi-agent systems, presents formal methods for verification, and addresses the complexity of uncertainty in modeling these concepts in practical applications, such as IoT and ad hoc networks. The challenges outlined above prompt several research questions:

Q1: *How can we define a temporal logic that can specify both trust and social commitments*

simultaneously?

In the literature, there is no such work that considers computationally grounded semantic that is capable to verify trust and social commitment relationship simultaneously. Therefore, we investigate the possibility of using the existing conventional temporal logics such as CTLC (an extension of CTL with modalities for reasoning about commitments and their fulfillments) logic to define the new logic. In our research, we extend the current CTLC logic with trust modality and introduce a logical language named Trust Computation Tree Logic Commitment (TCTLC). In fact, the reasons that encouraged us to extend CTLC logic are: (1) this logic is an extension of the CTL logic which has an efficient model checking procedure; (2) it has grounded semantics which means that it can be associated to computational models; and (3) its model is defined over the interpreted system formalism which is very suitable to model MASs and agent communication.

Q2: *How can we specify the capabilities, soundness and completeness of the developed temporal logic?*

Indeed, properties and reasoning postulates define these capabilities since reasoning rules capture common reasoning patterns that apply uniformly to trust and commitments relationship and their actions. In our research, we express a set of postulates along with formal proofs to support the developed logic.

Q3: *How can we define a temporal logic that can specify trust over social commitments properties with the presence of uncertainty?*

To tackle this inquiry, we propose to extend the developed TCTLC logic to multi-valued cases. We use the multi-valued logic represented in [32]. In contrast to the classical TCTLC logic, in multi-valued TCTLC logic, the modalities of trust and commitment are defined over multiple truth degrees which provides us more flexible modeling language. This allows us to express our models in uncertain settings more effectively. To the best of our knowledge, our work is the first attempt to address the

problem of modeling and verifying IoTs with trust over commitment in uncertain scenarios.

Q4: *Which formal methods should be used to verify the proposed temporal logic?*

Two verification techniques to verify the logic have been put forward: 1) Direct method in which new model checker is implemented from scratch or existing tool is extended with new algorithms for the required temporal modalities, and 2) Indirect method which is also called transformation-based method. The idea behind the indirect approach is to apply certain reduction rules to transform the current problem to an existing model checking problem. In this proposal, we explore the direct technique to verify our classical logics TCTLC and $T^{ws}CTLC$ by introducing new BDD-based model checking algorithms and implement these algorithms on top of MCMAS model checker and also we explore the indirect technique to verify our three-valued logic. we extended "MACMAS-interactor" tool to automatically calls MACMAS-TC tool to verify two-valued logic after reducing the the problem of three-valued model checking to classical model checking by adding new functionalities to make the tool enable to handle our 3v-TCTLC logic.

Q5: *How can we evaluate the developed temporal logic?*

An evaluation of both temporal logics has been performed by applying the proposed algorithms to multiple real-world case studies along with experimental results. Moreover, we have computed the time and space complexity of our developed logic for theoretical evaluation.

Q6: *How can we formulate and support a temporal logic that is capable of delineating both weak and strong trust in relation to social commitments?*

Our research incorporates a computationally grounded semantic capable of verifying both weak and strong trust in the context of social commitments. Consequently, our study explores the potential of leveraging conventional temporal logics, such as Computation Tree Logic (CTLC), to formulate a new logic. We expand upon the current CTLC logic, incorporating modalities for both weak and strong trust, and introduced a new logical

language named T^{ws}CTLC. To support the proposed logic we present a series of postulates accompanied by formal proofs as explained in Q2.

1.3 Research Objectives and Contributions

The ultimate goal of this research is to develop a formal and comprehensive framework that can investigate the interaction between trust and social commitment in MASs from the semantics and model checking perspectives in classical and uncertain settings. The main contributions can be summarized as follows:

1. We develop a new logic (TCTLC) capable of simultaneously reasoning about trust, social commitments, and their interactions within MASs.

2. We propose a set of reasoning rules along with formal proofs to support our logic.

3. We develop of a new logic, (3v-TCTLC), to specify trust over social commitments properties with the presence of uncertainty.

4. We consider an efficient model checking technique and implement it to verify the proposed logics.

5. We evaluate the proposed model checking problem of the developed logics in real and scalable case studies.

6. We formulate a new logic, T^{ws}CTLC, that facilitates the analysis of weak and strong trust, social commitments, and their interconnections within MASs.

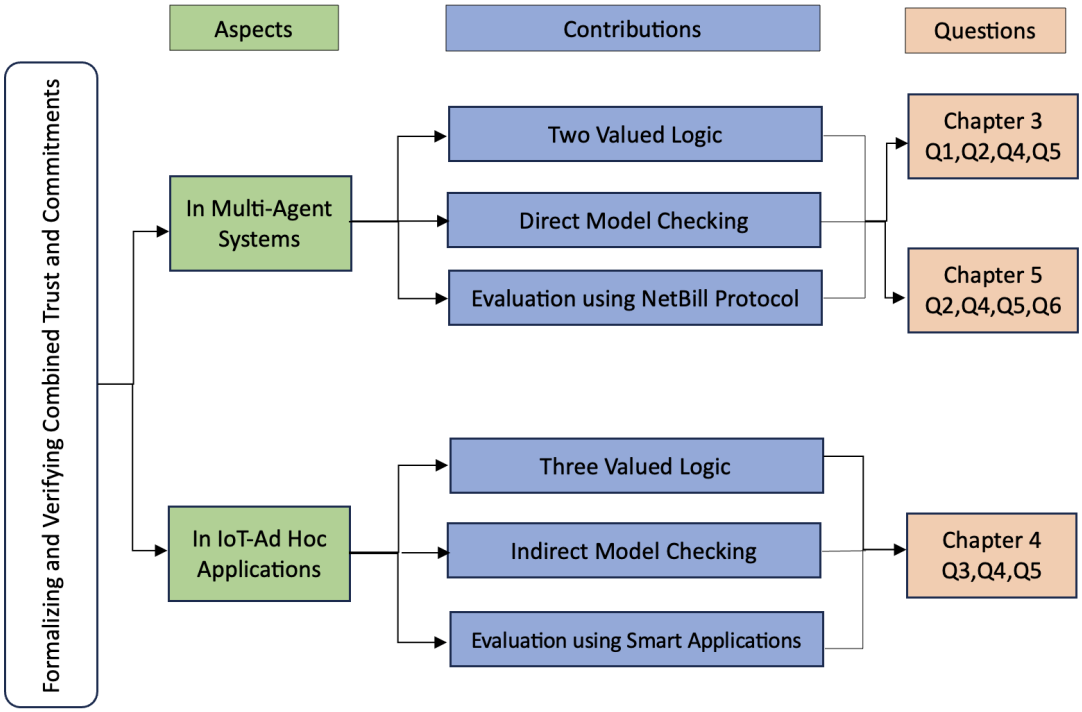

Figure 1.2: The thesis objectives, research questions and corresponding chapters

1.4 Thesis Organization

The rest of the thesis is organized as follows: we provide the necessary background information to comprehend the various concepts integral to our research work in Chapter 2. In Chapter 3, we present TCTLC, a new logic and its semantics that allows us to reason about trust and social commitment consistently. We present a set of reasoning postulates along with formal proofs to support the relationship between trust and commitment. We develop a new BDD symbolic algorithm dedicated to our logic. To check the effectiveness of the proposed algorithm, we implement it on top of the MCMAS model checker. Finally, the experimental results of verifying the NetBill protocol. Chapter 4 presents a novel approach for modeling and verifying IoT-ad hoc systems using a concrete application.

It involves the introduction of a new logic called 3v-TCTLC, which extends the existing 3v-CTL framework to reason about the aspect of trust over commitments considering uncertain environments. To ensure efficient model checking of this logic, we have devised a new algorithm that reduces the three-valued model checking problem to the classical two-valued model. Additionally, we have enhanced the functionality of the "MCMAS-interactor" tool to automate the 3v-TCTLC model checking process. This tool not only generates the two-valued TCTLC model automatically but also applies the new algorithm for model checking and presents the output. We have conducted an evaluation of both the algorithm and the tool. The results clearly demonstrate the effectiveness and efficiency of our proposed 3v-TCTLC logic and algorithm. Chapter 5 introduces a novel logic, T^{ws}CTLC, designed to encapsulate the properties associated with both weak and strong trust. The semantics is defined over an expanded interpretation of the original MASs's formalism. Furthermore, we provided postulates and their corresponding proofs to validate our logic. We introduced a model checking algorithm for T^{ws}CTLC, which extends the CTLC symbolic algorithm, and implemented it to create a new tool named $MCMAS - T^{ws}$. To validate our approach, we conducted experiments in a concrete case study, demonstrating the practical application of our proposed framework. Finally, in Chapter 6, we provide a summary of the results we achieved and outline potential future expansions of this work. An overview of the above defined objectives, research questions and corresponding chapters in which they are addressed are presented in Figure 1.2.

Chapter 2

Research Background

In this chapter, we summarize the relevant background for this thesis. More details about related work are provided in the following chapters where the relevant contributions are presented.

2.1 Agents and Multi-Agent Systems

An agent is an autonomous entity located in an environment which it can meet its designed objectives. Thus, an agent can control its state and behavior without intervention from other agents. An agent generally has the following properties [107]:

- Proactivity: Agents can initiate actions and make decisions based on their defined goals and objectives.
- Reactivity: They have the capacity to sense and respond to changes and events occurring within their environment.
- Social ability: Agents can engage in interactions with other agents by exchanging messages or data, facilitating collaboration and coordination.

- Rational ability: The actions of an agent are consistent with its objectives and goals, exhibiting a rational decision-making process.

Nowadays, the number of Web services is growing rapidly around the world. Human's services such as paying utility bills, buying and selling goods, managing bank account, booking airline and hotel reservations, etc can be implemented with a lower cost and at greater efficiency by software agents. Agents should be intelligent to actively seek ways to support human users. An intelligent agent satisfies the capabilities of proactivity, reactivity, rational and social ability. Wooldridge [106] defined the term agent as follows. "An agent is a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its delegated objectives".

The term multi-agent system (MAS) is defined as a "system in which several interacting, intelligent agents pursue some set of goals or perform some set of tasks" [104]. Such a system can be either compromised of heterogeneous agents, which refers to diversity of the agent goals, behavior and implementation or homogeneous agents, which refers to the identical agents with both design and implementation wise. These properties are what make MASs complicate and difficult to design, but they are also what make them highly effective in modeling and solving real-world problems ranging from industrial to commercial, healthcare, and governmental applications.

2.2 Agent Communication Languages (ACLs)

Effective communication is crucial for autonomous agents within Multi-Agent Systems (MASs) to collaborate on complex problems beyond individual capabilities. Thus, fostering communication among agents is pivotal in developing efficient MASs. In practical scenarios, agents engage in various forms of interaction to fulfill their objectives. These interactions range from cooperation to competition, depending on the agents' goals and

transaction contexts.

Establishing a standardized framework for agent communication is widely acknowledged. Despite numerous efforts in the literature to establish communication standards, there's been no consensus. Agent Communication Languages (ACLs) define semantics either internally, based on agents' beliefs and goals, or externally, based on their social commitments. Approaches utilizing internal semantics are termed mental approaches, focusing on agents' cognition, while those using external semantics are social approaches, considering the social context. Unlike mental approaches, social commitments have proven to be potent representations for agent interactions. They offer a social semantics that transcends agents' internal states, providing meaningful and observable exchanges among agents. In the context of this thesis, our focus lies on communication where message semantics are publicly defined, specifically in terms of social commitments.

2.3 Trust in MASs

As part of their mission, agents interact in MASs, which form uncertain and dynamic environments [107]. In these systems, agents may join or leave their interactions at any time as there is no guarantee about their behaviors. Therefore, using trust to govern agent communication and agents' dynamics is highly significant [7]. Trust is considered a key element that enables the collaboration, communication and interaction among autonomous agents. It has been the focus of many research contexts.

Trust in MASs has been investigated in a number of ways. We can classify them into two main approaches. The first approach enables agents to measure the amount of trust between interacting agents [35, 81, 100]. An agent with high degree of trust would be selected for interaction over multiple interactions. Thus, when there are repeated or ongoing interactions between multiple agents, the agent who is considered trustworthy or reliable is

more likely to be chosen as a partner for these interactions each time. In other words, if one agent has established a strong level of trustworthiness, it is expected to be consistently preferred or prioritized for engagement or collaboration over a series of interactions. In this way, trust models aim to guide an agent to make decision on when, how, and who to interact with. On the other hand, the second approach is obtained through trust models which allows parties to trust each other by reasoning about honesty or reliability of their counterparts. This approach focuses on proposing formal semantics to reason about trust. Several frameworks that provide a semantics for trust have been proposed and different application domains are considered [23, 31, 42, 92, 101]. In this paper, we follow the second approach and model trust as a modal operator with an intuitive semantics.

2.4 Social Commitments in MASs

Agents communicate in MASs and organize their actions and behaviors to achieve their goals [79]. Autonomous and heterogeneous agents use Agent Communication Languages (ACLs) to interact with each other. Therefore, formal semantics of these ACLs [47] are needed. The first approaches to define the semantics were based on agents' mental states which capture beliefs-desires-intentions concepts of communication. The mental approaches pertain to the actions carried out by agents to fulfill their intentions and rely on the underlying assumption that agents have the ability to read each other's minds [88]. In these approaches, semantics verification problem exists, which makes the verification of the agents' actions based on the semantics impossible [106]. Since its quite difficult to verify the compliance of agents with the defined mental semantics, a social approach has been introduced [88]. At the core of the social approach are social commitments which have been successfully used to model multi-agent interactions [9, 46, 90, 111]. Commitments are used as contracts among agents to reach an agreement. Commitments have been

used in various areas including communication protocols, IoT applications, and business contracts [18, 40, 46, 111]. Commitment also manipulated through a set of actions such as creation, fulfillment, release, violation, cancellation, delegation and assignment [89]. In this paper, we are interested in communicative social commitments as specified in [1,24,47,50].

2.5 Interpreted Systems

[53] introduced interpreted systems as a knowledge representation formalism for MASs to capture the concepts of knowledge and time. It is a useful formalism to model various classes of communicating MASs where agents can be heterogeneous and interact within a global system by exploiting various communication mechanisms [24, 47].

The formalism of interpreted systems functions through a set of n agents $A = \{1,\ldots,n\}$ in a set of *local states* L_i for each agent $i \in A$ giving i's complete system information at any given time. *Global states* are instantaneous compositions of all agents at any given time in the system in the sense that they provide a representation or arrangement of all the agents' states or properties at a specific moment in time to describe the overall state of the system. A global state $g \in G$ is a tuple $g = (l_1,\ldots,l_n)$ where $l_i \in L_i$. The standard product $L_1 \times \cdots \times L_n$ of n agents local states produces the set of global states G, i.e., $G = L_1 \times \cdots \times L_n$. The local state of agent i in the global state g is denoted by $l_i(g)$. $I \subseteq G$ is the set of initial global states. The *Local protocol* $\rho_i : L_i \rightarrow 2^{Act_i}$ is defined for each agent $i \in A$ to identify the enabled actions in a given local state. For each agent i, a set of local actions Act_i is assigned in order to address the system's temporal evolution. It is supposed that the action $null \in Act_i$ for every agent i is the *action of silence*.The *Global evolution function* is as follows: $\tau : G \times ACT \rightarrow G$, where $ACT = Act_1 \times \cdots \times Act_n$ and each component $a \in ACT$ is a joint action, which is a tuple of actions. τ_i is a local evolution function that determines the transitions for an individual agent i between his local states

and it is defined as follows: $\tau_i = L_i \times ACT_i \rightarrow L_i$.

[24] and [47] advocated the extended formal framework of interpreted systems to consider the run-time agent communication. A set of local variables Var_i is assigned to each agent $i \in A$ which represent the channels of communication that are used to send and receive messages. *Shared and un-shared variables* are used to establish commitments among interacting agents. A communication channel among the agents i and j exists iff $Var_i \cap Var_j \neq \emptyset$. The variable x value in the set Var_i at local state $l_i(g)$ is represented by $l_i^x(g)$. So, if $l_i(g) = l_i(g')$, then $\forall x \in Var_i$, $l_i^x(g) = l_i^x(g')$. For the shared variable $x, l_i^x(g) = l_j^x(g')$ means that the agent i in $l_i(g)$ and the agent j in $l_j(g')$ have an equal value for this variable.

2.6 The CTLC Logic

The CTLC logic is a combination of branching time CTL [38, 51] with modalities for representing and reasoning about social commitments with their fulfillment [24, 63].

Definition 2.1. [Syntax of CTLC] The syntax of CTLC is defined as follows:

$$\varphi ::= \rho \mid \neg\varphi \mid \varphi \vee \varphi \mid EX\varphi \mid EG\varphi \mid E(\varphi \cup \varphi) \mid C_{i \rightarrow j}\varphi \mid Fu(C_{i \rightarrow j}\varphi)$$

where:

- $\rho \in \Phi_p$ is an atomic proposition;
- E is the existential quantifier on paths;
- The Boolean connectives $\neg$ and $\vee$ are defined in the usual way;
- X, U, and G are CTL path modal connectives standing for "next", "until", and "globally";and

- The modal connectives $C_{i\to j}\varphi$ and Fu stand for "commitment" and "fulfillment of commitment";

In this logic, $C_{i\to j}\varphi$ is read as "agent i commits towards agent j that φ". $Fu(C_{i\to j}\varphi)$ is read as "the commitment $C_{i\to j}\varphi$ is fulfilled".

Definition 2.2. [Semantics of CTLC] Given the model M, the satisfaction of a CTLC formula φ in a global state s, denoted by $(M,s)\models\varphi$ is recursively defined as follows:

- $(M,s)\models p$ iff $p\in V(s)$;
- $(M,s)\models\neg\varphi$ iff $(M,s)\not\models\varphi$;
- $(M,s)\models\varphi\vee\psi$ iff $(M,s)\models\varphi$ or $(M,s)\models\psi$;
- $(M,s)\models EX\varphi$ iff there exists a path π starting at s such that $(M,\pi(1))\models\varphi$
- $(M,s)\models E(\varphi\cup\psi)$ iff there exists a path π starting at s such that for some $k\geq 0,(M,\pi(k))\models\psi$ and $(M,\pi(j))\models\varphi$ for all $0\leq j<k$;
- $(M,s)\models EG\varphi$ iff there exists a path π starting at s such that $(M,\pi(k))\models\varphi$ for all $k\geq 0$;
- $(M,s)\models C_{i\to j}\varphi$ iff for all global states $s'\in S$ such that $s_{\sim_{i\to j}}s'$, we have $(M,s')\models\varphi$;
- $(M,s)\models Fu(C_{i\to j}\varphi)$ iff $\exists s'\in S$ such that $s'_{\sim_{i\to j}}s$, we have $(M,s)\models\neg C_{i\to j}\varphi\wedge\varphi$ and $(M,s')\models C_{i\to j}\varphi$;

The semantics of CTLC state formulae in the model M combines both the semantics of CTL (see for example [38, 51]) and the semantics of the trust, commitment and fulfillment modalities.The state formula $C_{i\to j}\varphi$ is satisfied in the model M at s iff the content φ holds in every accessible states s' obtained by the accessibility relation $\sim_{i\to j}$. The state formula $Fu(C_{i\to j}\varphi)$ is satisfied in the model M at s iff state s satisfies the negation of the

commitment and there exists a state s' satisfying the commitment and s is accessible from s' by the accessibility relation $\sim_{i \to j}$.

2.7 Verification of Trust in MASs

Ensuring that a system aligns with its design requirements is particularly challenging, especially within multi-agent systems where numerous autonomous entities operate. The presence of these entities amplifies complexity and heterogeneity, posing significant hurdles to verification. One of the primary challenges in Multi-Agent Systems (MASs) is maintaining reliable trust relationships among potentially misbehaving entities. Such entities not only disrupt other agents but also impede their functionalities [61]. Given that these agents are autonomous and must interact within unreliable environments, there's a pressing need to reason about trust and identify untrusted computations.

Technically, trust verification mechanisms in MASs can be categorized based on when verification activities occur: runtime and design-time. Runtime approaches commonly employ monitoring, observing the system's evolving executions during operation, and checking if desired properties hold [5, 17, 19]. Runtime verification extracts relevant information from the system to detect undesired behaviors. Conversely, design-time approaches rely on static formal verification, often utilizing model checking to verify various aspects of MASs [24, 71, 103]. Each technique has its advantages and limitations. Runtime verification offers real-time observation and reaction to undesired behaviors but may have limited coverage [68]. In contrast, design-time techniques systematically check all possible states but face the state explosion problem with large systems. Both techniques complement each other in detecting untrustworthy behaviors and improving MASs development.

Despite the dynamic nature of trust, verifying trust properties at design time remains

critical and beneficial. Model checking trust ensures trust behaviors can occur among interacting agents. If model checking reveals no execution satisfies a trust property, the model must be revised as it's unsafe for deployment. Conversely, if all possible executions are trusted, demonstrating equivalence between the implemented and designed models suffices. In cases where some paths are trusted and others aren't, model checking guides dynamic verification to monitor untrusted paths. In this thesis, a design-time approach is adopted to enhance MAS utilization by reducing development time and costs while increasing confidence in system safety, efficiency, and robustness. Formal evaluation using model checking techniques has proven highly effective [37, 38].

2.8 Model Checking

Model checking is a verification that verifies if a given model M satisfies system specification φ or not. Model checkers have three components as follow:

- A method of encoding the state machine which represents the system under verification.
- A specification based on a temporal logic.
- A verification procedure that determine whether the specification is met.

Model checkers terminate with a "true" answer which means model satisfies the specification or provide a counterexample which shows the steps where error was encountered. Figure 2.1 depicts the model checking procedure.

It is not possible to talk about model checking without discussing the state explosion problem which occurs when the number of states grows exponentially with the number of variables in the system. One solution to the state explosion problem is symbolic model checking using BDDs (Binary Decision Diagram) [76]. To represent the state space

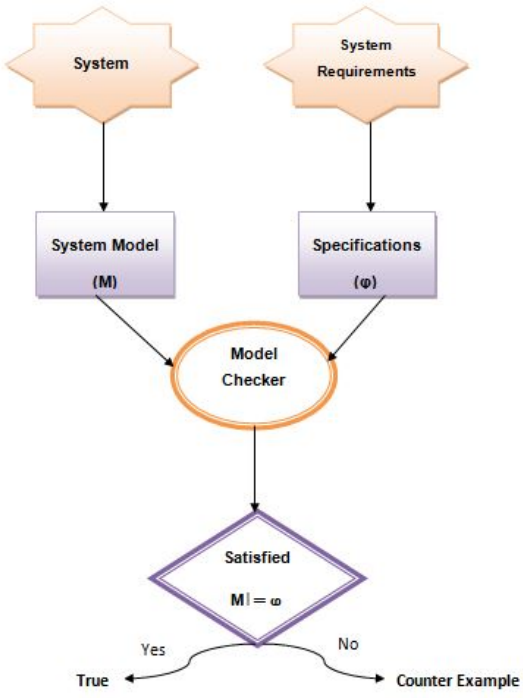

Figure 2.1: A model checker process

symbolically, BDDs data structure is used. As a result, they require fewer states in order to represent the state space. These data structures allow the manipulation of entire sets of states at a time, and allow for more efficient operations. For the verification many tools are developed:

- NuSMV [36] is an extended version of Symbolic Model Verifier (SMV) [79] that is used in Computation Tree Logic (CTL) and LTL to check finite state systems against properties.
- MCMAS [73] is an OBDD-based symbolic model checker developed for multi-agent systems which can verify various properties specified by as CTL, CTLC and CTLK logics.
- The PRISM tool [67] is used for checking probabilistic specifications in probabilistic model. The specifications can be expressed in the Probabilistic Computation Tree

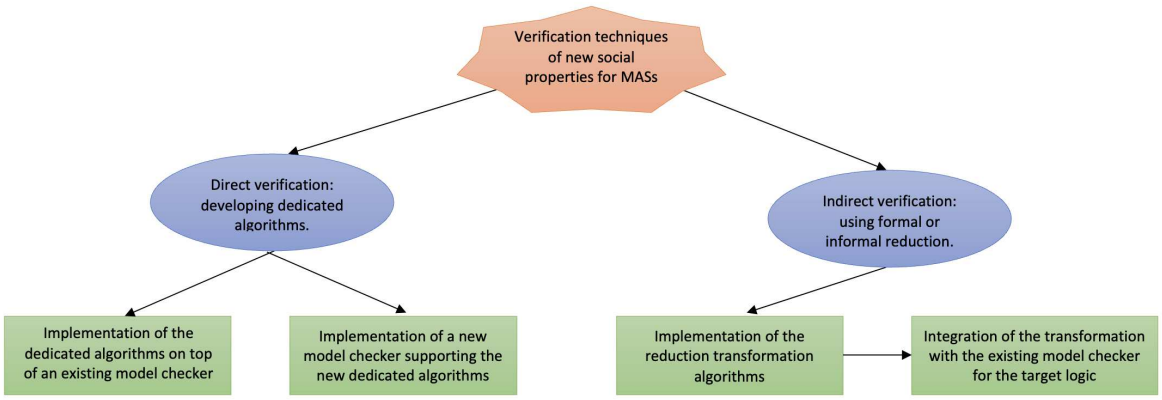

Figure 2.2: A typical verification process for multi-agent systems.

Logic (PCTL).

- SPIN [58] is a model checker for Linear Temporal Logic (LTL) to provide the correctness of process interactions.

To minimize post-development expenses and enhance confidence in the safety, efficiency, and reliability of Multi-Agent Systems (MASs), two model checking approaches have been proposed for reasoning about new social properties. These methods are categorized as direct and indirect verification techniques (refer to Fig 2.2). The direct approach involves the creation of specialized verification algorithms tailored to verify new social modalities. These algorithms are then integrated into an existing model checker or used to develop a new dedicated model checker. Conversely, the indirect approach aims to simplify the verification process by transforming the task of checking the new logic into checking an existing logic, leveraging reduction techniques. This eliminates the need for developing dedicated verification algorithms, focusing instead on automating transformations and integrating them into existing model checkers.

Practically, the indirect approach, known as transformation-based model checking, offers advantages in reducing post-development expenses associated with maintaining the model checker. It requires minor adjustments in cases where the core model checker

undergoes significant changes that affect backward compatibility. Additionally, it allows for the utilization of reliable model checkers with a long history of enhancements and strong community support.

2.9 Ad Hoc Networks and IoT

An ad hoc network is characterized by its self-organizing nature and lack of a predetermined topology. Within this network, nodes fulfill the roles of both hosts and routers, with the freedom for any node to join or leave from the network at any given moment. The communication between nodes occurs directly and wirelessly, provided they are within the radio range of each other. Remarkably, this network operates in a fully decentralized manner, devoid of any fixed infrastructure, such as access points or base stations, that would typically be required for its functioning. Six fundamental features delineate an ad hoc network: distributed functionality, multi-hop routing, autonomous end-points, dynamic topology, shared physical medium, and lightweight end-points. Figure2.3 illustrates the structure of wireless ad hoc networks.

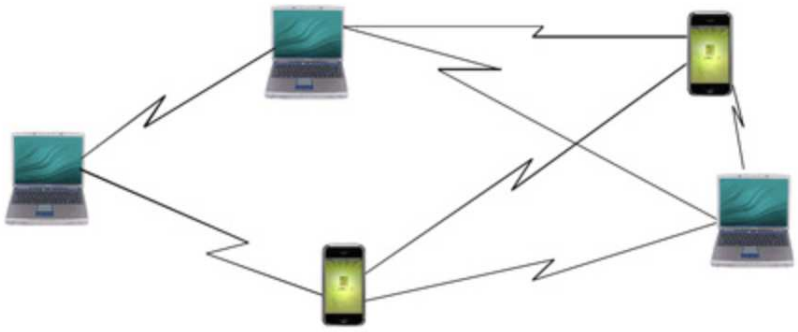

Figure 2.3: Wireless ad hoc network structure

In the contemporary era, individuals find themselves constantly surrounded by a diverse array of objects, spanning a wide range of sizes. This phenomenon has prompted

the development of electronic systems and devices capable of establishing interconnections through digital communication, with the internet serving as the principal conduit for data transmission. Within this networked framework, the human individual assumes the role of a data traffic consumer, while communication, management, and information exchange primarily occur between the interconnected objects and devices. The capacity of physical or virtual entities to be uniquely identified, engage in communication with other entities, and interact with their environment has facilitated the establishment of networks encompassing interconnected devices, end users, and various other entities within the global internet infrastructure. The term "Internet of Things" (IoT) denotes the infrastructure comprising interconnected objects, devices, and other entities, which possess the ability to communicate, exchange information, and execute operations over the internet, leveraging the interplay between communication technologies and networks [29].Figure 2.4 displays various examples of IoT devices.

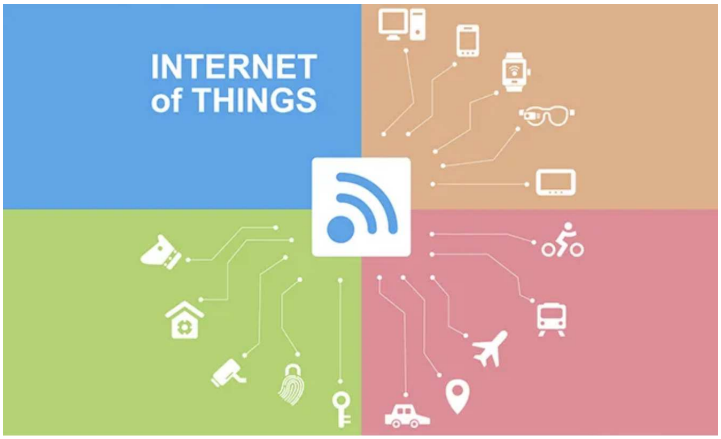

Figure 2.4: Examples of IoT devices

Sensors assume a pivotal role in facilitating the connection between various entities and their corresponding data, allowing for remote access by end users. These sensors undertake the task of collecting pertinent information, which is subsequently converted into a digital format and transmitted to other devices within Internet of Things (IoT) systems,

employing diverse wired or wireless technologies [99]. Given the extensive deployment and rapid proliferation of sensors worldwide, they serve as the primary interface that bridges the gap between entities, communication channels, and end users. Figure 2.5 shows a variety of sensors used to execute the required functionalities. Nevertheless, one of the paramount challenges encountered in IoT systems pertains to the selection of an appropriate medium for data transmission, as well as the routing of data across disparate heterogeneous networks [95]. Ad-hoc wireless networks and wireless technologies emerge as the most efficient and cost-effective means for transmitting data within IoT systems, thereby offering the added advantage of mobility, which effectively caters to human requirements while significantly diminishing installation expenses in comparison to wired technologies.

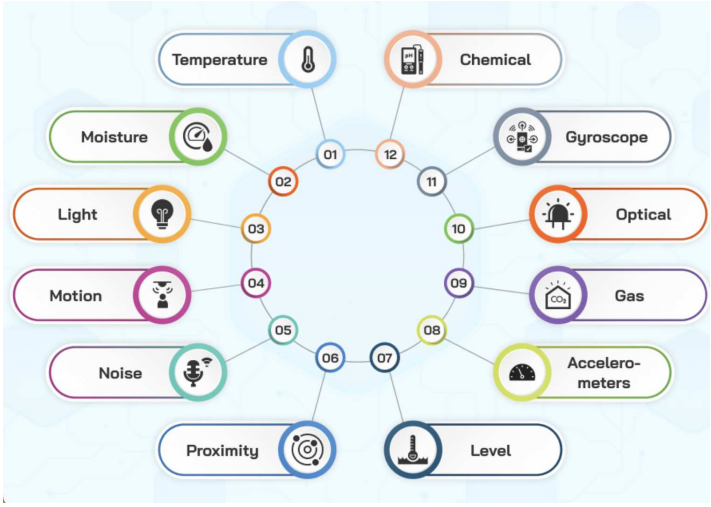

Figure 2.5: IoT sensor types

The Internet of Things (IoT) system encompasses digitally identified and interconnected entities, enabling wireless communication among them and facilitating the formation of ad hoc networks. These entities, commonly referred to as "things," possess both sensing and actuating capabilities, empowering them to interact with their surroundings. However, for IoT systems to operate effectively, they must address various critical factors, including device and thing heterogeneity, energy optimization, data

management, interoperability, and security considerations [28, 77]. The ability of IoT systems to accommodate these factors renders them applicable across diverse domains within smart cities [56], including healthcare, buildings, energy, transportation, and industrial sectors. Key technologies employed in the implementation of IoT systems within these domains comprise wireless sensor networks and ad hoc networks [20, 112].

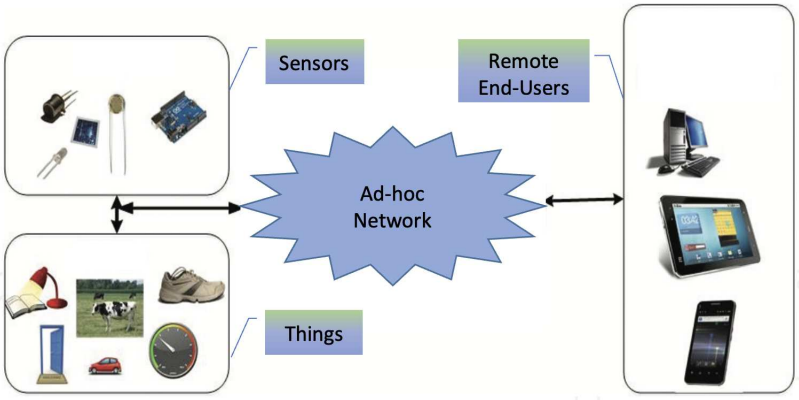

Figure 2.6: IoT-ad hoc network global structure

Figure 2.6 illustrates the global structure of an IoT-ad hoc network, where sensors collect data from a wide range of IoT devices and transmit it across the ad hoc network through various applications, ultimately reaching end-users' smart devices.

2.10 Three-Valued Lattice Logic (3v-Logic)

The application of classical two-valued logic has been prevalent in reasoning processes relying on determinate information. However, with the proliferation of interconnected components within IoT and the complexity nature of their communications, there arises a growing demand for reasoning mechanisms capable of handling uncertain or inconsistent information. In response to this necessity, the development of a logic system that can accommodate multiple degrees of truth, tailored to the requirements of specific

applications, becomes imperative. Such a logic framework utilizes truth values arranged in a lattice structure, thereby belonging to the category of many-valued logic known as lattice-valued logic [109].

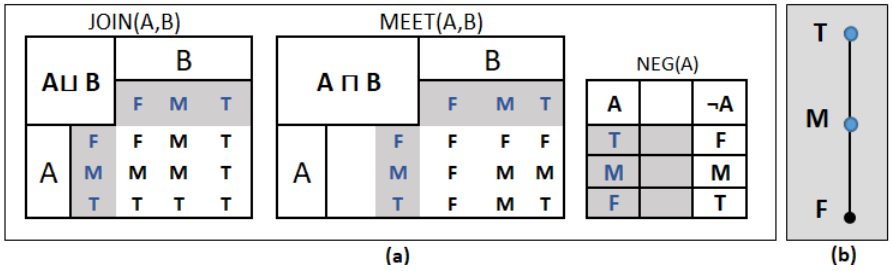

Figure 2.7: (a)The truth table of Kleene's logic;(b) 3v-lattice

In particular, 3-valued logic is a type of logic that utilizes the 3-valued lattice implication. It is an extension of the 2-valued logic and based on the Kleene's logic truth values of true (T), maybe to present uncertain or missed information (M), and false (F). It was first initiated by the logicians Jan Łukasiewicz and Clarence Irving Lewis. The 3-valued lattice [85,109] is represented by a set of objects with a 3v-valued ordering relation (T, M, F) represented in Figure 2.6(b). It is an algebraic structure (L3, $\sqcap, \sqcup$) where every two elements have a meet denoted by (a $\sqcap$ b) and a join denoted by (a $\sqcup$ b). The symbols $\sqcap$ and $\sqcup$ operate as the AND and OR in the Kleene's logic.The truth values of this logic is shown in Figure 2.7(a).

2.11 Quasi-Boolean Algebras and IoT System Reasoning

Quasi-Boolean algebras, also referred to as De Morgan algebras, differ from the latter by not mandating the presence of least and greatest elements. The emergence of these algebras arises from their aptitude in delineating real-world problems, with operators mirroring their classical counterparts. They derive truth values from a finite distributive lattice, adhering

to the truth ordering, and uphold the negation operator in line with De Morgan laws and involution ($\neg\neg x = x$) [109]. Fig2.8 illustrates various scenarios:

1. In Fig2.8(a), a classical logic is depicted within the lattice, where $\neg T = F$ and $\neg F = T$.

2. Fig2.8(b) illustrates a lattice presenting a 3-valued logic, where T signifies true, F denotes false, and M denotes maybe. This logic facilitates reasoning about uncertain and inconsistent information in IoT systems. For instance, complete and consistent sensor readings are designated the value T, while inconsistent or incomplete readings receive the value M, as they may still hold truth. Only definitively incorrect information is assigned the value F.

3. Fig2.8(c) portrays Belnap's 4-valued logic [21], wherein N represents neither true nor false, and B denotes both true and false. This logic aids in reasoning about inconsistent yet preexisting information in IoT systems, arising from conflicting sensor readings.

4. The lattice depicted in Fig2.8(d) represents the product algebra 2×2, encompassing truth values TT, FF, TF, and FT. This logic facilitates the resolution of disagreements between two knowledge sources. For instance, disparate designers modeling a single IoT system may disagree on certain system behaviors.

5. Fig2.8(e) encompasses both disagreement and uncertainty, serving to describe IoT system behaviors that encompass multiple truth levels, such as behaviors that must, must not, should, and should not be implemented in the system [64].

The 3-valued logic, known as Kleene's logic, extends the traditional two-valued logic and represents a subset of multi-valued logic. The designation "M" (for maybe) also implies

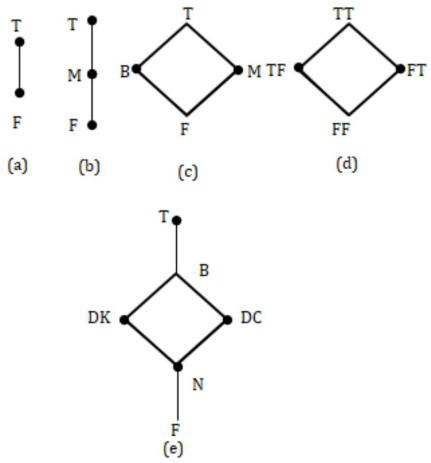

Figure 2.8: Types of lattices

"undefined" or "neither". This logic proves effective in addressing uncertainty arising from various sources:

- Partial IoT models where some behaviors remain unknown.
- Abstracted IoT systems where certain behaviors are omitted to simplify the state space.
- Open IoT systems where certain behaviors are determined by other components or features.
- Uncertain IoT models where behaviors may conflict or provide conflicting information at specific phases of the IoT system life cycle.

2.12 Summary

In this chapter, we have provided the foundational background required to navigate through the thesis. In the upcoming chapter, we will introduce a novel, systematic method for comprehensively capturing the interplay between trust and social commitments within Multi-Agent Systems (MASs).

Chapter 3

Model Checking Combined Trust and Commitments in Multi-Agent Systems

In this chapter, we introduce a comprehensive and systematic approach for the formally reason about trust, communicative social commitments, and how they interrelate within Multi-Agent Systems (MASs). Specifically, we formulate a new and coherent temporal logic named $TCTLC$ to facilitate this process. Section 3.1 peresents an overview of the proposed approach. In Section 3.2, we present the formalism of extended interpreted systems and introduce the new TCTLC logic. In Section 3.3, we introduce a set of reasoning rules in TCTLC to clarify the relation between trust and commitment. Section 3.4 introduces the BDD-based symbolic model checking algorithms of our logic along with the complexity of model checking TCTLC. Finally, we implement the proposed algorithms using a new model checker tool called MCMAS-TC dedicated to TCTLC, verify the NetBill protocol case study, and report our experimental results and comparison with benchmarks

in Section 3.5.

3.1 An Overview of the Proposed Approach

In this chapter, our objective is to explore the connection between trust and social obligations from a semantic point of view. Therefore, we show how the relation between trust and commitments can be represented in a logical-based formalism by expanding the Computation Tree Logic (CTL), the classic branching time logic [38, 51], with modalities about combined trust and commitment to specify protocol properties for multi-agent interactions. To do so, we illustrate our work in four phases outlined in Fig. 3.1. These phases are as follows:

In the first phase: we introduce Trust Computation Tree Logic with Commitments (TCTLC), a new logic that allows us to reason about trust and social commitment in a combined and consist manner. Moreover, we introduce a new semantics of trust based on the underlying commitment's fulfillment. To demonstrate the relationship between trust and commitment, we provide a desiderata of reasoning rules accompanied by formal proofs.

In the second phase: we introduce our formal verification technique based on extending an existing model checker, MCMAS (Model Checker for Multi-Agent Systems) with new algorithms. We develop new Binary Decision Diagram (BDD)-based symbolic algorithms dedicated to the proposed logic.

In the third phase: we compute the complexity of the proposed model checking algorithm for TCTLC in terms of both time and space. The objective is to demonstrate that the model checking algorithm is efficient and still has the same time complexity for explicit models and the same space complexity for concurrent programs as the standard CTL logic.

In the forth phase: to analyze the proposed algorithm's effectiveness, we implement

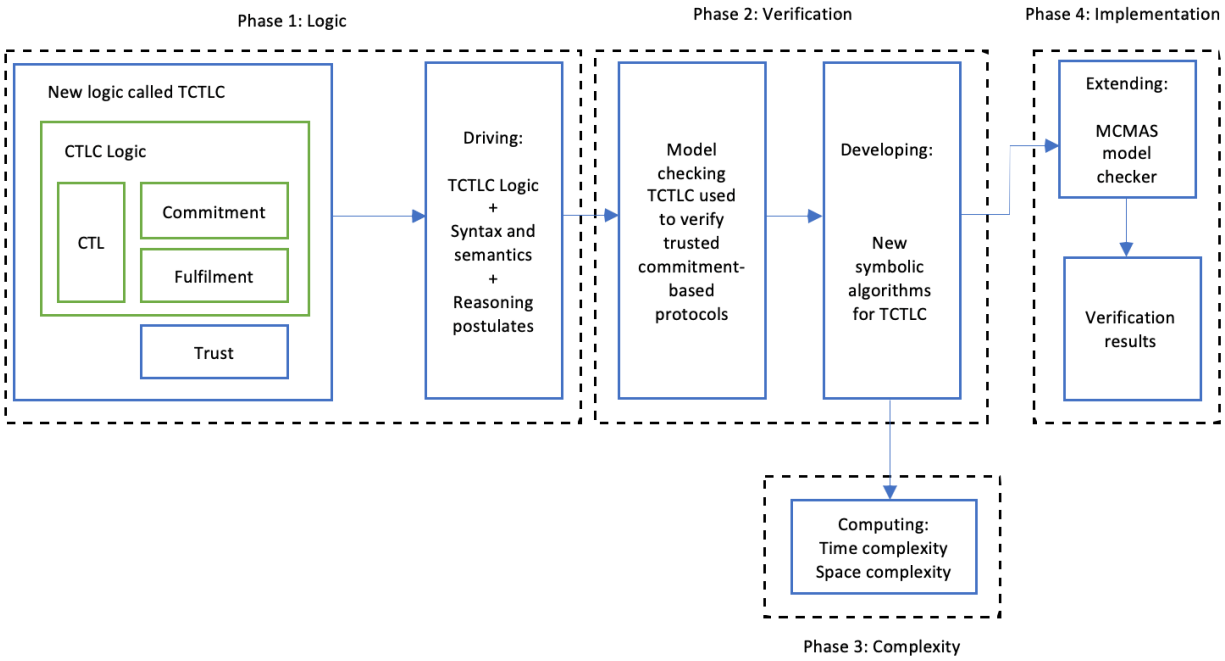

Figure 3.1: The building blocks of our approach

it as libraries attached to the MCMAS model checker after extension of its input language, named Interpreted Systems Programming Language (ISPL), with shared and non-shared variables. Finally, the results of the experiments obtained for the verification of the NetBill protocol as a case study are revealed and compared with relevant benchmarks.

3.2 The TCTLC Logic

TCTLC is a combination of branching time CTLC (Trust Computation Tree Logic with Commitments) [24, 63] with modalities for representing and reasoning about trust.

Definition 1 (TCTLC Model) A formal model $M = (S, I, R_t, \{\sim_{i\to j}: (i,j) \in A^2\}, V)$ of TCTLC is a tupled structure defined as follows:

- S is a set of global states of the system;
- $I \subseteq S$ is a set of initial states;

- $R_t \subseteq S \times S$ is a total temporal relation defined by $(s,s') \in R_t$ iff there is a joint action $a = (a_1,...,a_n) \in ACT$ such that $\tau(s,a_1,...,a_n) = s'$;

- For each pair of autonomous agents $(i,j) \in A^2, \sim_{i \to j} \subseteq S \times S$ is a social accessibility relation defined by $s \sim_{i \to j} s'$ iff (1) $l_i(s) = l_i(s')$; (2) $(s,s') \in R_t$; (3) $Var_i \cap Var_j \neq \emptyset$, and $\forall x \in Var_i \cap Var_j$, we have $l_i^x(s) = l_j^x(s')$, and $\forall y \in Var_i - Var_j$, we have $l_j^y(s) = l_j^y(s')$; and

- $V : S \to 2^{AP}$ is a valuation function over the set AP of atomic variables.

We suppose that every accessible state s' from s is the next state of s using R_t. The intuition behind the definition of the accessibility relation $s \sim_{i \to j} s'$ among two states s and s' is the existence of a communication channel between agents i and j in the system via shared variables that facilitate the exchange of information between the two agents such that i in s sends a content through the channel and j receives in s' the channel's content. Upon completion of the information exchange process between agents i and j, the shared variables associated with the agents will have identical values of x for i in $l_i(s)$ and for j in $l_j(s')$ (i.e. $l_i^x(s) = l_j^x(s')$), reflecting the updated information received through the communication channel. As the agent i initiates the communication and is not learning any new information, s and s' are indistinguishable for agent i ($l_i(s) = l_i(s')$) and as the agent j receives the communication, states s and s' are indistinguishable regarding to the non-shared variables that have not been communicated by i ($\forall y \in Var_i - Var_j, l_j^y(s) = l_j^y(s')$).

Thus, the accessibility relation requires that the immediate successor state does not reveal any information to the sending agent i. it is in fact enough to only enforce the requirement of non-revelation of information to the sender in the immediate successor state (i.e., $(l_i(s) = l_i(s'))$). This is because if a successor of s' (e.g., s'') is accessible from this state (i.e., accessible from $s' : s' \sim_{i \to j} s''$), then we will have $l_i(s') = l_i(s'')$. Consequently, $l_i(s) = l_i(s'')$. This means the requirement is automatically satisfied in the subsequent accessible

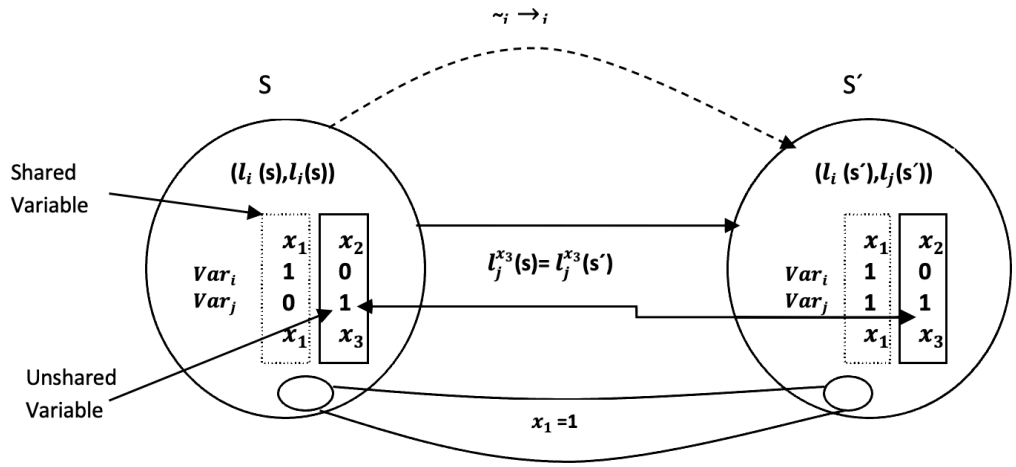

Figure 3.2: Illustration of $\sim_{i\to j}$, the binary accessibility relation for commitments

states. Also, for the shared and non-shared variables, the requirement of non-revelation needs to be enforced only on the immediate successor since these variables are defined independently from the states as they only depend on the agents. Consequently, if the information is not revealed to the immediate successor, then, it cannot be reveled to any other non-immediate successor.

This idea is presented in Fig. 3.2, where agent i and agent j are communicating and shared and un-shared variables of these agents through the channel are included in: Agent $i : Var_i = \{x_1, x_2\}$; Agent $j : Var_j = \{x_1, x_3\}$. The variable x_1 is the shared variable between i and j which present communication channel between both agents. The variables x_2 and x_3 are un-shared variables among them. The value of x_1 for j in s is changed to the value of x_1 for i in s', which illustrates establishment of communication channel and the transmission of a message.

Definition 2 (TCTLC's Syntax). The following grammar provides the TCTLC's syntactic constructs:

$\varphi ::= \rho \mid \neg\varphi \mid \varphi \vee \varphi \mid EX\varphi \mid EG\varphi \mid E(\varphi \cup \varphi) \mid C_{i\to j}\varphi \mid Fu(C_{i\to j}\varphi) \mid T_{i\to j}\varphi$

where:

- $P \in \Phi_p$ is an atomic variable;
- E is the path existential operator;
- The Boolean connectives $\neg$ and $\vee$ are defined in the usual way;
- X, U, and G are CTL path modal connectives standing for "next", "until", and "globally";
- The modal connectives $C_{i \to j}\varphi$ and Fu stand for "commitment" and "fulfillment of commitment"; and
- $T_{i \to j}\varphi$ stands for "trust".

where $\rho, \vee, E, X, G, \cup, C$ and Fu are defined in Definition 2.1. The operator $T_{i \to j}\varphi$ is to indicate that "agent i trusts agent j that φ holds.

A path, denoted by π, starts from a particular state, e.g., $s_0 \in S$, and is formed by an infinite sequence of states and transitions $(s_i, s_{i+1}) \in R_t$ for $i = 0, 1, \ldots$. A finite path of length k is a subset of an infinite path with k states. A state s' is considered reachable from s_0 if there exists a finite path π in the transition system that can reach that state, i.e., $\pi = s_0 \to s_1 \to \cdots \to s_k$ such that $k \geq 0$ and $s_k = s'$ [18]. To simplify the notation, the previous finite path will be denoted by $\pi = \langle s_0, \ldots, s_k \rangle$. The set of all paths in the system is denoted by $\prod$. The following is a definition of the set of states which are in the past of s: $Pas(s) = \{s' \in S \mid (s', s) \in R_t \, or \, \exists \pi \in \prod \text{ such that } \pi = \langle s', \ldots, s \rangle\} \cup \{s\}$.

Definition 3.1. Definition 3 (Semantics of TCTLC) Given the TCTLC model M and the system's global state s, $(M, s) \models \varphi$ is the recursive satisfaction of a TCTLC formula φ and is defined as follows:

- $(M, s) \models p$ iff $p \in V(s)$;
- $(M, s) \models \neg\varphi$ iff $(M, s) \not\models \varphi$;

- $(M,s) \models \varphi \vee \psi$ iff $(M,s) \models \varphi$ or $(M,s) \models \psi$;
- $(M,s) \models EX\varphi$ iff there exists a path π starting at s such that $(M,\pi(1)) \models \varphi$;
- $(M,s) \models E(\varphi \cup \psi)$ iff there exists a path π starting at s such that for some $k \geq 0, (M,\pi(k)) \models \psi$ and $(M,\pi(j)) \models \varphi$ for all $0 \leq j < k$;
- $(M,s) \models EG\varphi$ iff there exists a path π starting at s such that $(M,\pi(k)) \models \varphi$ for all $k \geq 0$;
- $(M,s) \models C_{i\to j}\varphi$ iff for all global states $s' \in S$ such that $s_{\sim_{i\to j}}s'$, we have $(M,s') \models \varphi$;
- $(M,s) \models Fu(C_{i\to j}\varphi)$ iff $\exists s' \in S$ such that $s'_{\sim_{i\to j}}s$ and $(M,s') \models C_{i\to j}\varphi$;
- $(M,s) \models T_{i\to j}\varphi$ iff $(M,s) \models C_{j\to i}\varphi$ and $\exists s'$ such that $s' \in Pas(s)$, we have $(M,s') \models Fu(C_{j\to i}\varphi)$;

The formal semantics of TCTLC state formulae in the model M combines both the branching time semantics [38, 51] and the semantics of the trust, commitment and fulfillment modal operators. The state formula $C_{i\to j}\varphi$ holds in he model M at s iff the commitment subject φ is satisfied in all accessible states s' through the binary relation $\sim_{i\to j}$. The model M at s satisfies the state formula $Fu(C_{i\to j}\varphi)$ iff there exists a state s' where the commitment holds and s is accessible from s' using the binary relation $\sim_{i\to j}$.

The state formula $T_{i\to j}$ holds in the model M at the state s iff s satisfies the commitment $C_{j\to i}\varphi$ and there exists a state s' satisfying the fulfillment of the commitment where s' is in the past of s. The semantics of trust indicates that trust is achieved when we reach a reachable state from the fulfillment state, in which the commitment is active.

3.3 Reasoning Rules

We discuss in this section several relevant reasoning postulates reflecting the properties of the proposed logic. These postulates are highly important in a logical framework. They are used as rules to express the properties of the logic, and to illustrate how trust, commitment, and their fulfilment interact with one another. These postulates allow us to better understand the expressiveness and soundness of the logic [91, 92]. Expressiveness refers to the ability of the logic to represent different concepts and reasoning patterns, while soundness refers to the correctness and validity of the reasoning that the logic allows.

P1: No active trust without commitment.

$\neg C_{i\to j}\varphi \to \neg T_{j\to i}\varphi$

Meaning: When there is no commitment about φ, the trust to bring about φ is discharged.

Proof: The proof is straightforward from the semantics of $T_{i\to j}\varphi$ which indicates that the current state satisfies $C_{j\to i}\varphi$.

P2: Fulfillment of the active trust.

$T_{j\to i}\varphi \to EF(Fu(C_{i\to j}\varphi))$

Meaning: Trust about a content yields fulfillment of the commitment to bring about the same content.

Proof: According to the semantics of $T_{j\to i}\varphi$, we obtain $(M,s) \models C_{i\to j}\varphi$. Moreover, in the semantics of $C_{i\to j}\varphi$, for all global states $s' \in S$ such that $s_{\sim_{i\to j}}s'$, we have $(M,s') \models \varphi$ and then the commitment is fulfilled when its content holds in the reachable accessible states. Consequently, $(M,s) \models EF(Fu(C_{i\to j}\varphi))$, so the postulate holds.

P3: No fulfillment without commitment.

$\neg A(\neg C_{i\to j}\varphi \cup Fu(C_{i\to j}\varphi))$

Meaning: Fulfillment of a commitment requires prior existence of this commitment.

Proof: Using the semantics of $Fu(C_{i\to J}\varphi)$, we obtain $(M,s') \models C_{i\to j}\varphi$ where s' is accessible from the current state s. The postulate follows from the fact that s' is in the past of s.

P4: No trust without fulfillment.

$\neg A(\neg(Fu(C_{i\to j}\varphi)) \cup T_{j\to i}\varphi)$

Meaning: Trust about a content yields prior fulfillment of the commitment about the same content.

Proof: The proof is a direct consequence of the fulfillment past condition in the semantics of trust.

P5: Commitment and fulfillment conditions to trust.

$Fu(C_{i\to j}\varphi) \to \varphi \wedge AF(C_{i\to j}\varphi \to T_{j\to i}\varphi)$

Meaning: If there is fulfilment, then the content holds and any future commitment yields trust.

Proof: For the first part of the conjunction, φ holds in the current fulfillment state because it is accessible from the commitment state. For the second part, assume that $(M,s) \models Fu(C_{i\to j}\varphi)$ and $(M,s) \models AF(C_{i\to j}\varphi)$, then trust is satisfied in exactly the same states where the commitment holds as the two conditions of trust semantics hold, so the proof is completed.

P6: Commitment inference condition.

$C_{i\to j}\varphi_1, \varphi_1 \vdash \varphi_2$ infer $C_{i\to j}\varphi_2$

Meaning: The commitment about φ_2 yields if the agent i commits to bring about the content from which φ_2 derives.

Proof: The proof drives from the satisfaction of φ_2 in all the accessible states. These states

satisfy φ_1 since $\varphi_1 \vdash \varphi_2$, so the postulate holds.

P7: Fulfillment inference condition.

$Fu(C_{i\to j}\varphi_1), \varphi_1 \vdash \varphi_2$ infer $Fu(C_{i\to j}\varphi_2)$

Meaning: The fulfilment of the commitment about φ_2 yields if the agent i fulfills to bring about the content from which φ_2 derives.

Proof: The proof drives from the fact that the state s' from which the current s is accessible and where $C_{i\to j}\varphi_1$ holds satisfies $C_{i\to j}\varphi_2$ (from P6).

P8: Trust inference condition.

$T_{i\to j}\varphi_1, \varphi_1 \vdash \varphi_2$ infer $T_{i\to j}\varphi_2$

Meaning: Trust about a derived content is true if the trust regarding the original content is true as well.

Proof: The proof is a consequence of P6 and P7.

P9: No commitment and trust without a holding content.

$AG\neg\varphi \to \neg C_{i\to j}\varphi \wedge \neg T_{j\to i}\varphi$

Meaning: If the content doesn't globally hold, then the commitment and trust about the content never hold.

Proof: The proof is derived directly from the semantics of the commitment, the trust and the fact that all accessible states are reachable.

P10: From whole to parts.

$C_{i\to j}(\varphi_1 \wedge \varphi_2) \to C_{i\to j}\varphi_1$

$Fu(C_{i\to j}(\varphi_1 \wedge \varphi_2) \to Fu(C_{i\to j}\varphi_1)$

Meaning: Once a commitment (fulfillment, trust) to a conjunction holds, the commitment

(fulfillment, trust) to each part of the conjunction also holds.

Proof: The commitment part follows from the fact that in the formal semantics of commitment, all the accessible states which satisfy the conjunction $\varphi_1 \wedge \varphi_2$ also satisfy φ_1. Then, $C_{i\to j}\varphi_1$ follows. From this part and the semantics of fulfillment, the second part follows, from which the trust part derives. $T_{i\to j}(\varphi_1 \wedge \varphi_2) \to T_{i\to j}\varphi_1$

It is worth mentioning that two fulfillment actions of two distinct commitments cannot be shorten into one fulfillment action because the two underlying commitments could take place in two different states. Consequently, two distinct trusts cannot be shorten into one combined trust. This combination would hold if the two underlying commitments hold in the same state as shown by the next postulate:

P11: From parts to whole.

$C_{i\to j}\varphi_1 \wedge C_{i\to j}\varphi_2 \to EF(\quad Fu(C_{i\to j}\varphi_1) \wedge Fu(C_{i\to j}\varphi_2) \to Fu(C_{i\to j}(\varphi_1 \wedge \varphi_2)))$

$C_{i\to j}\varphi_1 \wedge C_{i\to j}\varphi_2 \to EF(T_{j\to i}\varphi_1 \wedge T_{j\to i}\varphi_2 \to T_{j\to i}(\varphi_1 \wedge \varphi_2))$

Proof: The first part follows from the formal definition of $Fu(C_{i\to j}\varphi)$ and the fact that $C_{i\to j}\varphi_1 \wedge C_{i\to j}\varphi_2 \to C_{i\to j}(\varphi_1 \wedge \varphi_1)$ holds. The second part follows from the first part and the semantics of $T_{j\to i}\varphi$.

P12: No commitment, fulfillment and trust to the false content.

$\neg C_{i\to j}\bot$

$\neg Fu(C_{i\to j}\bot)$

$\neg T_{i\to j}\bot$

Meaning: A commitment (trust) to false cannot hold and similarly, a commitment to false cannot be fulfilled.

Proof: Since there is no accessible state that satisfies $\bot$, the first point follows, from which the second and third point derive, so the postulate.

P13: Commitment consistency.

$C_{i\to j}\varphi \to \neg C_{i\to j}\neg\varphi$

$T_{j\to i}\varphi \to \neg C_{i\to j}\neg\varphi$

Meaning: When the commitment (trust) about φ is satisfied, then committing about the negation of φ cannot take place.

Proof: The first part of the rule is direct from the semantics of $C_{i\to j}\varphi$ as there is no accessible state satisfying $\neg\varphi$. The second part follows since trust yields commitment.

P14: Trust consistency.

$C_{i\to j}\varphi \to \neg T_{j\to i}\neg\varphi$

$T_{i\to j}\varphi \to \neg T_{i\to j}\neg\varphi$

Meaning: When the commitment (trust) holds, then there is no possibility to trust its negation.

Proof: The first part is direct from P13. The second part is from P1 and P13 as follows: $T_{i\to j}\varphi \to C_{j\to i}\varphi \to \neg C_{j\to i}\neg\varphi \to \neg T_{i\to j}\neg\varphi$.

3.4 Symbolic Model-Checking TCTLC

In automata-based model checking, the state explosion often occurs when the number of system's states increases exponentially. Performing model checking in a symbolic way is one of the solutions that mitigate this problem. This technique takes as input φ, a TCTLC formula and computes $[[\varphi]]$, the set of states where φ holds. $[[\varphi]]$ is represented using a special data structure named Ordered Binary Decision Diagram (OBDD) [30] that provides suitable manipulation techniques. This data structure is also used to represent the set of initial states I. The set $[[\varphi]]$ is then compared against the set I. Consequently, we conclude

that a TCTLC model M satisfies the TCTLC formula φ iff I is part of $[[\varphi]]$, i.e., $I \subseteq [[\varphi]]$. In a formal way, this fact can be represented as $(M,I) \models \varphi$ iff $(M,s) \models \varphi \ \forall \ s \in I$. In this work, to model check the TCTLC logic, we expand the symbolic verification algorithm of CTL proposed by [38] by adding procedures that compute the set of states that satisfy the new logic formulae.

Algorithm 3.1 :$SMC(\varphi,M)$: the set $[[\varphi]]$ satisfying the $TCTLC$ formula φ

1: φ is an atomic formula: return $V(\varphi)$;
2: φ is $\neg\varphi_1$: return $S - SMC(\varphi_1,M)$;
3: φ is $\varphi_1 \vee \varphi_2$: return $SMC(\varphi_1,M) \cup SMC(\varphi_2,M)$;
4: φ is $EX\varphi_1$: return $SMC_{EX}(\varphi_1,M)$;
5: φ is $E(\varphi_1 \cup \varphi_2)$: return $SMC_{EU}(\varphi_1,\varphi_2,M)$;
6: φ is $EG\varphi_1$: return $SMC_{EG}(\varphi_1,M)$;
7: φ is $C_{i\to j}\varphi_1$: return $SMC_c(i,j,\varphi_1,M)$;
8: φ is $Fu(C_{i\to j}\varphi_1)$: return $SMC_{fu}(i,j,\varphi_1,M)$;
9: φ is $T_{i\to j}\varphi_1$: return $SMC_t(i,j,\varphi_1,M)$;

First, we present our main algorithm (Algorithm 3.1). To build the set $[[\varphi]]$, the algorithm takes the TCTLC model M and the TCTLC formula φ as inputs and iterates recursively through the structure of the formula φ using sets operations.

The lines 1 through 6 call the CTL standard algorithms. Lines 7 and 8 invoke the commitment and its fulfilment procedures and Line 9 calls our proposed procedures which are defined in Algorithms 3.5 and 3.6 respectively. Algorithm 3.2 illustrates the computations that construct the set $Pre_\sim(s')$ as the pre-images of the s' by means of the binary relation $(\sim_{i\to j})$. Formally: $Pre_\sim(s') = \{s \in S \mid s \sim_{i\to j} s'\}$.

In Fig 3.3, we demonstrate the proposed algorithms through an example. The values of x and y which are the shared and un-shared variables are given at each state. Algorithm 3.2 computation works as follows:

$Pre_\sim(s_0) = \{\}$, $Pre_\sim(s_1) = \{s_0\}$, $Pre_\sim(s_2) = \{s_1,s_4\}$, $Pre_\sim(s_3) = \{s_1\}$, $Pre_\sim(s_4) = \{s_2,s_3\}$, $Pre_\sim(s_5) = \{s_3\}$, $Pre_\sim(s_6) = \{\}$.

Algorithm 3.2 : $SMC_{\sim}(i,j,s',M)$: the set $Pre_{\sim}(s')$

1: $X_1 \leftarrow \{s \in S \mid l_i(s) = l_i(s')$ and $(s,s') \in R_t$ and $Var_i \cap Var_j \neq \emptyset$ and $l_i^z(s) = l_j^z(s')\ \forall z \in Var_i \cap Var_j\}$ and $l_j^u(s) = l_j^u(s')\ \forall u \in Var_i - Var_j$;
2: return X_1;

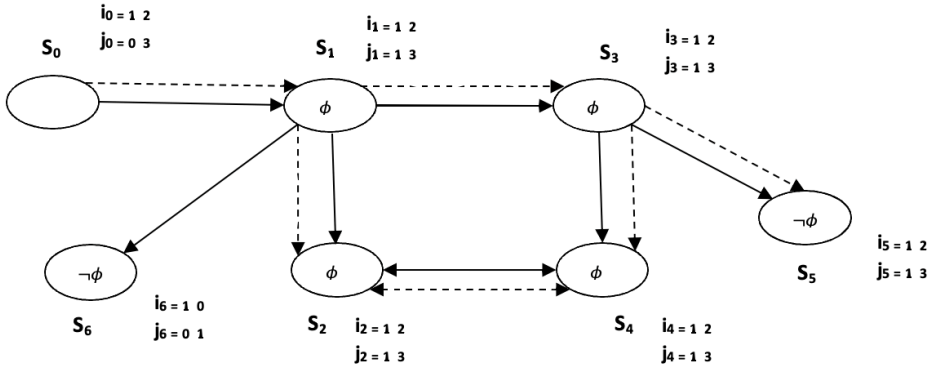

Figure 3.3: An illustrative example of the algorithms

3.4.1 Symbolic Algorithm for the Social Commitment Modality

Algorithm 3.3 addresses the semantics of $C_{i\to j}\varphi$ to return the set $[[C_{i\to j}\varphi]]$ of states where the commitment holds. Initially, the algorithm calculates the set X_1 consisting of states satisfying the negation of the content $\neg\varphi$, then it builds the set X_2 consisting of states that have accessibility to a state s' where $\neg\varphi$ holds (i.e., in X_1) through the binary relation $\sim_{i\to j}$. Finally, the algorithm returns X_2's complement. Fig. 3.3 shows an illustrative example where the calculation is as follows: $X_1 = \{s_0, s_5, s_6\}$ and $X_2 = \{s_3\}$. Then, the algorithm returns $S - X_2 = \{s_0, s_1, s_2, s_4, s_5, s_6\}$. The state s_3 is not labeled by the commitment as it has accessible state s_5 satisfying $\neg\varphi$.

Algorithm 3.3 : $SMC_c(i,j,\varphi,M)$: the set $[[C_{i\to j}\varphi]]$

1: $X_1 \leftarrow SMC_c(\neg\varphi, M)$;
2: $X_2 \leftarrow \{s \in S \mid \exists s' \in X_1\ s.t.\ s \in SMC_{\sim}(i,j,s',M)\}$;
3: return $S - X_2$;

3.4.2 Symbolic Algorithm for the Fulfillment Modality

As the other procedures, the function $SMC_{fu}(i,j,\varphi,M)$ takes as input the formula φ and the underlying model M and begins with constructing X_1, which is equal to the set $[[C_{i\to j}\varphi]]$ of the states where the commitment $C_{i\to j}\varphi$ holds. Then it builds the set X_2 of states that satisfy the content φ and $\neg C_{i\to j}\varphi$. Afterwards, the procedure computes the states in X_2 that have accessibility from a state in X_1 through $\sim_{i\to j}$. In Fig. 3.3, $X_1 = \{s_0,s_1,s_2,s_4,s_5,s_6\}, X_2 = \{s_3\}$ and $X_3 = \{s_3\}$. Algorithm 3.4 returns X_3.

Algorithm 3.4 : $SMC_{fu}(i,j,\varphi,M)$: the set $[[Fu(C_{i\to j}\varphi)]]$

1: $X_1 \leftarrow SMC_c(i,j,\varphi,M)$;
2: $X_2 \leftarrow SMC(\varphi,M) \cap (S - X_1)$;
3: $X_3 \leftarrow \{s \in X_2 \mid \exists s' \in X_1 \cap SMC_{\sim}(i,j,s,M)\}$;
4: return X_3;

3.4.3 Symbolic Algorithm for the Trust Modality

Algorithm 3.5 implements the semantics of $T_{i\to j}\varphi$. It starts by calculating the set X_1 of states satisfying the commitment $C_{j\to i}\varphi$. Then it computes the set X_2 of states satisfying the formula $Fu(C_{j\to i}\varphi)$. Afterward, the procedure builds the set X_3 of states which are in the past of the states satisfying $C_{j\to i}\varphi$. Finally, the algorithm returns those states (i.e., in X_4) that hold $C_{j\to i}\varphi$ (i.e., in X_1) and are reachable from past states satisfying $Fu(C_{j\to i}\varphi)$ (i.e., in $X_2 \cap X_3$).

Indeed, Algorithm 3.5 calls the procedure $Past(X)$ introduced in [50] (Line 4) that constructs the set of past states of X. The Procedure $Past(X)$ is presented in Algorithm 3.6. In this Algorithm, line 1 defines that each state is the past of itself. Then, it iterates using *while... do* to reach the fix-point in the past by calling the procedure $pre_{\exists}(X)$ that is presented in [59]. A set $X \subseteq S$ is an input of the procedure $pre_{\exists}(X)$ which builds the set of states $Y \subseteq S$ such that a transition to a state in X is possible. Formally:

$Y = pre_{\exists}(X) \leftarrow \{s \in S \mid \exists s'\ s.t.\ s' \in X \text{ and } (s,s') \in R_t\}$

In Fig 3.3 the computation of Algorithms 3.5 and 3.6 are as follows: $X_1 = \{s_0, s_1, s_2, s_4, s_5, s_6\}, X_2 = \{s_3\}$ and $X_3 = \{s_0, s_1, s_2.s_3, s_4\}$ and finally $X_4 = \{s_5\}$ is returned.

Algorithm 3.5 : $SMC_t(i, j, \varphi, M)$: the set $[[T_{i \to j}\varphi]]$

1: $X_1 \leftarrow SMC_c(j, i, \varphi, M)$;
2: $X_2 \leftarrow SMC_{fu}(j, i, \varphi, M)$;
3: $X_3 \leftarrow Past(X_1)$;
4: $X_4 \leftarrow \{s \in X_1 \mid \exists s' \in X_2 \cap X_3 \; s.t. \; (s', s) \in R_t\}$
5: return X_4;

Algorithm 3.6 : Past(X): the set of X past states

1: $V \leftarrow pre_{\exists}(X) \cup X$
2: $W \leftarrow \emptyset$
3: while $W \neq V$ do
4: $\quad W' \leftarrow W$
5: $\quad W \leftarrow V$
6: $\quad V \leftarrow V \cup pre_{\exists}(V - W')$
7: end while
8: return V

3.4.4 Time and Space Computational Complexity

In this section, we analyse the time and space computational complexity of the TCTLC model checking problem. Let $|M|$ and $|\varphi|$ be the size of the TCTLC model M and formula φ respectively.

Theorem 3.1. *The model checking algorithm of TCTLC can be solved in time* $O(|M| \times |\varphi|)$.

Proof. It is known from [49] that model checking commitment and fulfillment can be done in time $O(|M| \times |\varphi|)$. For the trust modality, the algorithm calls the model checking algorithms of commitment and fulfilment, and then the procedure $pre_{\exists}(X)$ that looks in the past of the commitment state to see if there is a fulfillment state. This can be done by injecting a fresh and unique atomic proposition χ in the state that satisfies the commitment

and return that state if there is a state that satisfies $\neg(\neg Fu(C_{i\to j}\varphi))\cup\chi$, which can be done in time $O(|M|\times|\varphi|)$, so the result. □

Theorem 3.2. *The problem of model checking TCTLC is P-complete.*

Proof. Membership in P (upper bound) is direct from Theorem 3.1. Since TCTLC extends the commitment logic, the P-hardness (lower bound) follows from the P-completeness of the model checking problem of this logic proved in [49]. □

In practice, the model M is implemented as a synchronized product of n modules or agents called concurrent programs. In this setting, it is suitable to analyze the space complexity of the TCTLC model checking problem for these programs.

Proposition 3.1. *Model checking TCTLC is PSPACE-hard with respect to the size of concurrent programs and the length of the formula.*

Proof. The lower bound in PSPACE (hardness) is direct from the PSPACE-completeness of the Computation Tree Logic of conditional commitments CTL^{cc} [63] since TCTLC subsumes CTL^{cc}. □

Theorem 3.3. *The space complexity of model checking TCTLC is PSPACE-complete with respect to the size of concurrent programs and the length of the formula.*

Proof. Model checking the trust modality in TCTLC can be solved by the algorithm explained in the proof of Theorem 3.1, which calls the model checking problem of the fulfillment modality proved to be PSPACE-complete [49]. The reduction of this modality can be computed in polynomial space with respect to the model (as the same model is used) and the formula as only an atomic proposition of fixed size is being added. Thus, the result follows from Proposition 3.1. □

3.5 Implementation

In this section, we aim to develop an automated verification tool for trust and commitment interactions that enriches the literature of model checking intelligent and autonomous MASs. Also, we intend to model check different properties of MASs based on trust and commitment relationships involving multiple interacting agents. Hence, we have incorporated and extended version of MCMAS, the powerful and extendable model checker adapted to MASs [74], with our proposed algorithms presented in Section 3.4 as additional libraries of the tool. The selection of MCMAS as the model checking toolkit is based on its specific design for the verification of MASs using various agent-based logics. MCMAS is a toolkit that uses OBDDs [30] for the symbolic representation of state spaces, providing an effective solution to the issue of combinatorial explosion encountered in the verification of numerous MAS models. MCMAS is an expandable open-source model checker that has been widely employed in numerous research projects. Its versatility, efficacy, and user-friendly interface render it a valuable instrument for verifying MASs across diverse application domains, such as robotics, cyber-physical systems, and multi-agent decision-making. MASs in MCMAS are modeled using Interpreted Systems Programming Language (ISPL), which exhibits a semantics similar to interpreted systems, a widely used framework in temporal-epistemic logics introduced by [53]. This feature enables easier specification and reasoning about MAS models, and promotes the seamless integration of MCMAS with other tools and frameworks.

In this section, wextended the MCMAS tool by adding our symbolic algorithms of trust, commitments and their fulfillment which are implemented in C++. Our extended tool generates an extended version of the input language ISPL to compute shared variables, which is required for the computation of the accessibility relations. The newly implemented tool is called Model Checker for Multi-Agent Systems with Trust and Commitments

(MCMAS-TC)[1].

3.5.1 NetBill Protocol Properties

The NetBill payment protocol, which is designed for the good trades using software with encryption over the Internet [94], is applied to illustrate our implementation. The NetBill payment is a step-wised protocol: (1) customer and merchant authenticating each other using a public-key certificate, (2) requesting and sending a quote by customer and merchant, respectively, (3) sending the digital information if the merchant's quote accepted by the customer, (4) providing an electronic payment order by customer including the good description, (5) verification of the electronic payment order (or EPO) by merchant, (6) sending it to the Netbill server, (7) checking the bank account of the customer reliability by the Netbill server, and finally, (8) authorizing payment to the merchant account and providing a receipt included the key for decryption the goods. After the reception of the key and receipt, the customer must decrypt the information related to the purchased goods.

We verify the NetBill protocol against the properties related to the interaction between trust, commitment and its fulfilment. Concretely, over this protocol, the following properties are verified:

Reachability property. Expresses that a particular computation sequence from the initial state will be reached. φ_1 states that a path exists in which the merchant will not trust the customer to pay until the customer has fulfilled the underlying commitment.

$\varphi_1 = (\neg Fu(C_{Cus \to Mer} Payment) \cup (Fu(C_{Cus \to Mer} Payment) \wedge T_{Mer \to Cus} Payment))$

Safety property. States that a bad situation will never be reached. For instance, φ_2

[1]The tool is accessible at: `https://www.dropbox.com/sh/9ctdnt3cdqny0wp/AACDxqtubbPbeJHdu_Vzazfxa?dl=0`

expresses that the situation where the customer fulfills the payment commitment, but the merchant never trusts that customer with regard to the payment is not reachable.

$\varphi_2 = AG\neg(Fu_{Cus\rightarrow Mer}\ Payment \wedge \neg T_{Mer\rightarrow Cus}\ Payment)$

Liveness property. States that a good situation will eventually happen. The formula φ_3 shows that if the customer fulfills the payment, then in all future computations, the merchant will trust the customer whenever this customer commits to pay.

$\varphi_3 = AG(Fu(C_{Cus\rightarrow Mer}Payment) \rightarrow AF(C_{Cus\rightarrow Mer}Payment \rightarrow T_{Mer\rightarrow Cus}Payment))$

Table 3.1: The NetBill protocol verification results using MCMAS-TC

Exp.#	Agents#	States#	Memory (MB)	Time (Sec.)
1	2	17	10.67	<0.01
2	4	111	11.28	<0.01
3	6	818	12.46	<0.01
4	8	5836	14.49	0.58
5	10	40528	16.60	1.50
6	12	278030	19.53	2.40
7	14	1.90766e+06	22.43	2.66
8	16	1.37581e+07	31.90	4.33
9	18	2.21354e+08	40.28	11.62
10	20	5.61637e+09	132.94	101.66
11	22	1.3327e+11	133.24	115.50

3.5.2 Verification Results

Our study aimed to assess the efficacy and scalability of the developed algorithms in terms of both model checking processing time and memory usage by BDD. To achieve this, we began by creating a model of the protocol and formulae to be evaluated using the extended ISPL language. Specifically, we encoded the NetBill protocol scenario in terms of the

local states and roles, shared and non-shared variables, and atomic propositions utilized by the interacting agents for communication. Subsequently, we included the initial states, formulae, and atomic propositions in the Evaluation, IntState, and Formulae sections of the model. Finally, we used our MCMAS-TC tool to check the correctness of the encoded protocol against the specified formulae.

Our experiments are performed on Dual-Core CPU T4500 at 2.30 GHz running an operating system with 64 bits. We report 11 experiments[2] for various numbers of agents in Table 3.1. We consider the system size in terms of the number of agents, the model size in terms of the number of states, the BDD memory in use, along with the execution time. In fact, the experiments revealed that all the verified formulae hold in the model. As reported in the table, we observe that with regard to the number of agents, the memory usage increases in a polynomial way but the state space increases rapidly in an exponential way.

[2]The experiments are available at: `https://www.dropbox.com/sh/atrqezkzeypuas3/AABPNgZqLB3bIMq3ij_FF3Qua?dl=0`

Table 3.2: Comparison between proposed and existing techniques

Agents#	Memory (MB)	Time (Sec.)	Agents#	Memory (MB)	Time (Sec.)	Agents#	Memory (MB)	Time (Sec.)
[3]			[47]			Proposed Technique		
2	4.398	<0.001	2	4.241	<0.020	2	10.67	<0.01
3	4.492	<0.001	3	5.507	0.184	4	11.28	<0.01
4	4.698	0.1	4	12.957	2.736	6	12.64	<0.01
5	4.691	1.3	5	15.432	63.687	8	14.49	0.58
6	4.750	12.1	6	83.839	630.914	10	16.60	1.50
7	4.880	168.6	[48]			12	19.53	2.40
8	4.969	1753.5	2		0.190	14	22.43	2.66
9	5.151	27557.4	3		0.746	16	31.90	4.33
[2]			4		6.635	18	40.28	11.62
2	5.112		5		32.680	20	132.94	101.66
3	5.212		6		286.932	22	133.24	115.50
4	6.708		7		1684.409			
5	13.652		8		5123.356			
6	41.888							

One of the motivations of our investigation is to compare our results with relevant benchmarks. Table 3.2 provides an overview of the NetBill verification results in [2, 3, 47, 48] and our technique. It also shows our results for better comparison. As shown in the table, our approach outperforms the benchmarks in terms of scalability and execution time. From the scalability point of view, there is more overhead in terms of memory usage, and the state explosion problem happens earlier in the benchmarks compared to our approach. In [3], although we observe that the memory usage is less than our technique up to 9 agents, the experiments in this approach go to halt faster when the number of agents increases. Using this approach, up to 9 agents can be checked to verify the same scenario. We achieve higher performance in our approach that is more scalable with better results when the system becomes larger. Moreover, the usage of time in our technique is much better than [3]. Figure 3.4 depicts the comparison of the execution time. We considered 12 agents only

as the other techniques support up to 9 agents only. The figure shows that the verification process in our technique is much faster than the benchmarks.

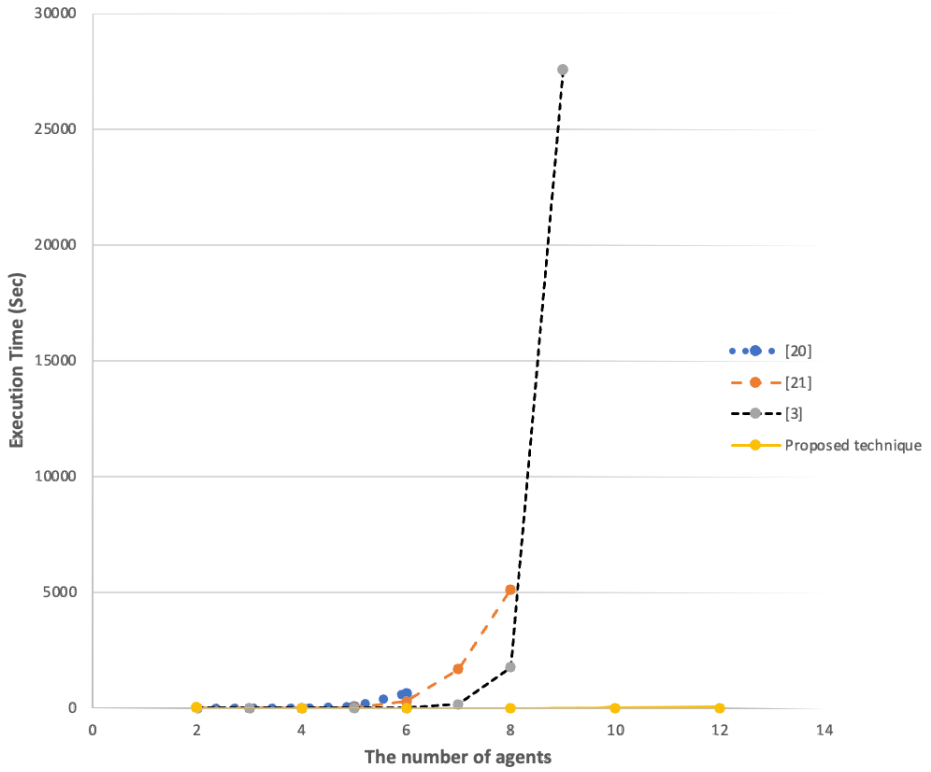

Figure 3.4: Comparison of the models' performance as function of the system size, in terms of the number of agents, and execution time

Furthermore, we conducted a comparison between our proposed technique and the indirect model checking method introduced by [43]. It is crucial to clarify that our trust logic ($TCTLC$) and the trust logic defined by [43] ($TCTL^C$) have distinct semantics. In our proposed trust logic, we leverage the concept of commitments made in the past, requiring an examination of agents' past behaviors and tracking previous commitments and their fulfillment. This, however, comes with computational demands. On the other hand, the semantics proposed in [43] only considers forward accessibility relations.

To ensure a fair and meaningful comparison, we implemented the semantics of [43] ($TCTL^C$) into our tool and compared the results, as shown in Table 3.3. The obtained

results, using the same machine, are very close. However, it is essential to highlight a key consideration. When specifying system properties, the indirect technique employs specific reduction rules to transform the current problem into a standard model checking problem, where both the model and formula undergo transformation. Consequently, the witness examples (if the property is satisfied) and the counterexamples (generated in case a property is not satisfied) are relevant to the transformed model, not the original model. Hence, they cannot directly contribute to understanding satisfaction or identifying issues in the original model.

Table 3.3: Comparison between proposed and existing transformation toolkit

Agents#	Memory (MB)	Time (Sec.)	Agents#	Memory (MB)	Time (Sec.)
[43]			Proposed Technique		
2	9.04	0.04	2	10.65	< 0.1
4	9.26	0.10	4	10.89	< 0.1
6	9.73	0.30	6	11.25	< 0.1
8	9.57	0.45	8	11.65	< 0.1
10	10.12	0.49	10	11.69	< 0.1
12	10.71	1.08	12	12.25	0.66
14	11.50	1.35	14	12.71	0.95
16	12.28	1.73	16	13.93	1.02
18	12.75	2.36	18	13.67	1.52
20	13.14	2.82	20	15.76	2.9
...	...	...	...	...	...
30	15.47	5.34	30	17.10	5.51

3.6 Summary

We introduced TCTLC logic that encapsulates the combination of two crucial concepts in MASs: trust and agents' commitments. Our exploration addressed two primary aspects: the formulation of reasoning postulates and the intricacies of model checking. The semantics

for trust were delineated based on accessibility relations, utilizing extended versions of interpreted systems to articulate the interaction between trust and commitment among the interacting entities. Furthermore, our proposed postulates were substantiated with rigorous proofs. We developed new symbolic algorithms and established that our model checking algorithm mirrors the complexity of model checking CTL in the context of both explicit models and concurrent programs. The proposed algorithms were incorporated into our MCMAS-TC tool, which was then employed to verify the NetBill protocol. The outcomes of these experiments were meticulously documented and reported.

Chapter 4

Verifying Trust over IoT-Ad Hoc Network-based Applications under Uncertainty

In this chapter[1], we propose a novel approach for modeling and verifying IoT-ad hoc systems using a concrete application. Section 4.1 presents an overview of the proposed approach. Section 4.2 provides the previous studies related to the topic. We also point out the shortcomings of the existing research that our work aims to address. Section 4.3 discusses the hypothesis underlying the proposed solution and potential limitations. Section 4.4 introduces a way of coping with uncertainty through the 3v-TCTLC logic. We will discuss how 3v-TCTL can be used to model uncertain systems and how it can be used to reason about the behavior of such systems. Section 4.5 presents our proposed 3-valued model checking technique for uncertain systems. It discusses the details of our proposed algorithm, including how it works and how it can be used to efficiently analyze uncertain

[1]The results of this chapter are published in [15]

systems. Finally, In section 4.6, we will summarize the results of our work and discuss how our proposed technique can be used to further improve the analysis of uncertain systems.

4.1 An Overview of the Proposed Approach

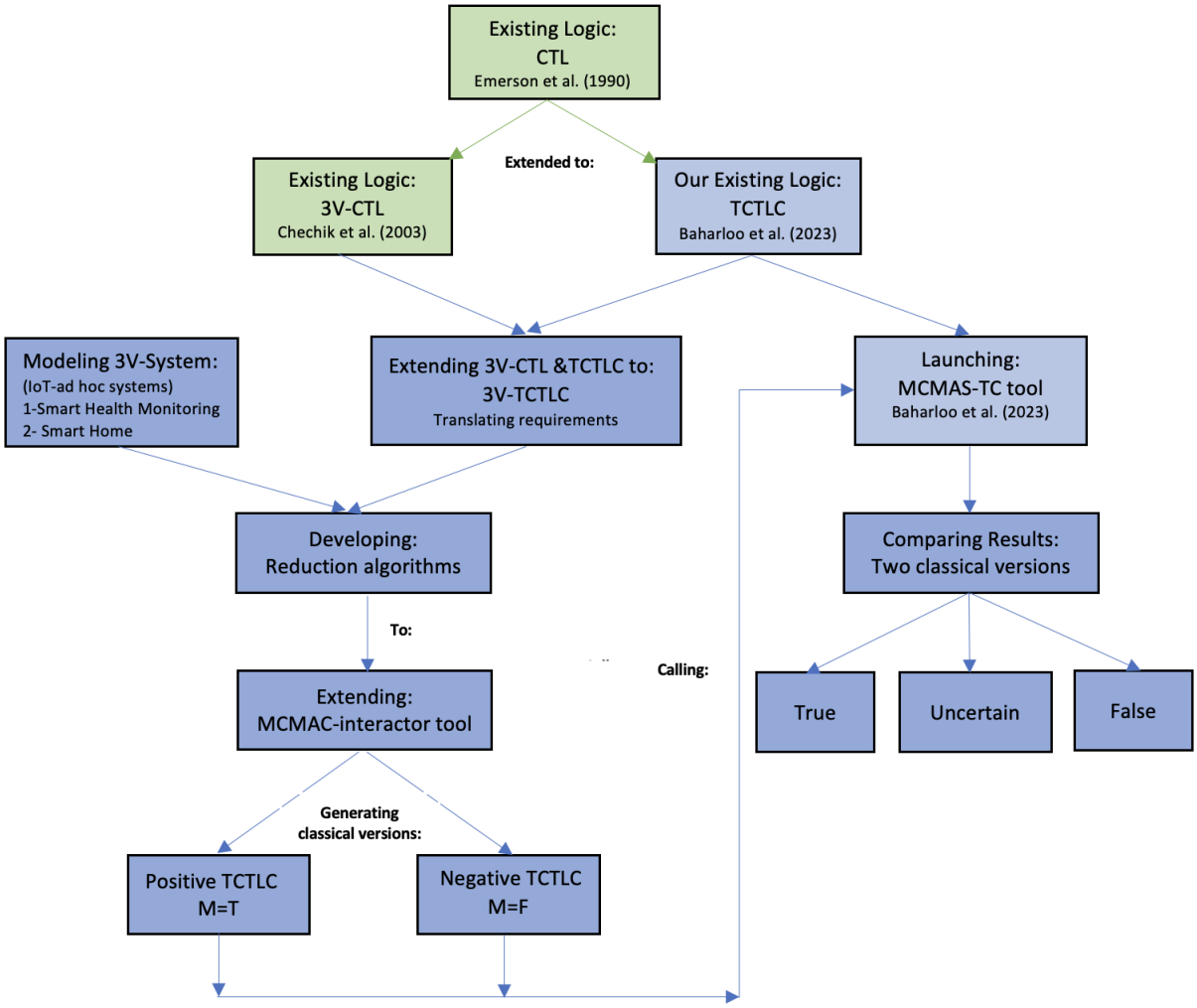

Figure 4.1: The proposed approach outline

Figure 4.1 illustrates overall contributions of this chapter. We introduce several significant contributions in the context of IoT and ad hoc networks as follows:

- Trust-Based Solutions in Uncertain Settings: This research emphasizes the importance of trust-based solutions for mitigating the presence of misbehaving entities within IoT-ad hoc network-based applications which are susceptible to vulnerabilities due to uncertainty and incomplete information. Trust is seen as a key factor in making well-informed decisions about communication partners, and successful partnerships are built on social commitments within satisfied contracts.

- Introduction of 3v-TCTLC Logic: We introduce a novel logic called 3v-TCTLC, which extends the existing 3v-CTL framework to reason about trust over commitments in uncertain environments. This logic is crucial for modeling and verifying IoT-ad hoc systems. It accommodates the difficulty of determining, with absolute certainty, whether a property about entity behavior is true or false, by incorporating three-valued logic (T, M, F), where M represents undefined or uncertain information.

- Formal and Efficient Verification Technique: A formal verification technique through model checking is proposed as a pivotal method for identifying vulnerabilities in IoT-ad hoc network systems. Model checking involves concrete system modeling, formal properties expressed using a temporal logic, and a verification process to assess system fulfillment of specifications. To ensure efficient model checking using 3v-TCTLC, we present a new algorithm that reduces the three-valued model checking problem to the classical two-valued model. This algorithm simplifies the process of verifying trust and commitments in uncertain environments.

- Enhanced MCMAS-Interactor Tool: We enhance the functionality of the “MCMAS-interactor" tool, automating the 3v-TCTLC model checking process. This tool generates the two-valued TCTLC model automatically and applies the new algorithm for model checking, presenting the output.

- Evaluation of Logic and Algorithm: The paper conducts an evaluation of both the 3v-TCTLC logic and the model checking algorithm, demonstrating their effectiveness and efficiency in addressing trust and commitment issues in IoT-ad hoc network systems.

The proposed method shares common goals and challenges with the existing literature in trust management for IoT and ad-hoc networks. However, it distinguishes itself through its unique integration of model checking, the introduction of a novel multi-valued logic (3v-TCTLC), and its emphasis on simultaneous trust and commitment verification in uncertain environments. These differences contribute to the advancement of intelligent systems in uncertain network settings.

4.2 Related Work

Various trust-based protocols have been developed in the context of ad hoc networking. The authors of [102] use a finite state machine model to represent ad hoc on-demand distance vector (AODV) actions that are observed locally in order to develop a statistical description of each peer's behavior. It examines different node mobility patterns which provide further insights. The work of Liu et al. extended AODV routing protocol named B-AODV [70] that incorporates trust mechanisms to enhance end-to-end delivery security. The work in [98] introduced a trust management approach that utilizes a human-based model to establish trust relation among nodes in an ad hoc network. The proposed trust management scheme determines trust by using trust ratings at the local level and has a low message overhead for a large-scale network. [114] presents a formal analysis of a routing algorithm that incorporates group trust and routing algebra and utilizes trust as its basis. The key safety and security concerns related to the Internet of Things (IoT) connected to mobile ad hoc networks is introduced in [83]. A demonstration of ALARM's

ability to achieve privacy, security, anonymity, authentication, and integrity was proposed in [45]. Several other trust-based techniques have been introduced in IoT-ad hoc networks [33, 34, 105, 108]. While the approaches mentioned above have typically overlooked the utilization of model checking for the verification of system operation and compliance with designated requirements, it is noteworthy that our work stands out through its integration of model checking to guarantee that the system functions as intended and adheres to its predefined specifications.

Despite the separate explorations of model checking trust and commitment in various investigations [6, 25, 26, 41, 44, 90], our work distinguishes itself by effectively addressing the challenges of uncertainty in IoT-ad hoc systems while also integrating model checking for trust and commitment. Multi-valued model checking is an approach that has proved to be an efficient way of reasoning about systems with uncertain behaviors. Several studies addressing this topic can be found in the literature. For instance, Chechik et al. [32] introduced the multi-valued version of the Computational Tree Logic (CTL) named χCTL, which is used for reasoning about uncertainty and inconsistency. This approach has been successfully applied to various domains such as robotic systems, control systems, and distributed systems. In [6], the authors applied the technique to a Smart Home application, which is an Internet of Things (IoT) type of application. The mv-ATL*, a multi-valued version of Alternating-time temporal logic, was introduced by the authors in [60] for the purpose of verifying systems with multiple truth values. It is based on a 6-valued lattice, which assigns different truth values to the system's components.

Recently, the multi-valued model checking technique has been effectively used for verifying multi-agent systems with trust and commitment protocols that are designed under uncertain settings. In [7, 8, 10, 11], the authors proposed new logics called 3v-TCTL and 4v-TCTL for reasoning about uncertainty and inconsistency in intelligent systems. These logics are extensions of Temporal Computation Tree Logic (TCTL) which is a

formal language for specifying properties of distributed systems. The new logics are designed to capture multi-valued probability functions which are useful in evaluating the performance of trust and commitment protocols in multi-agent systems. The proposed logics are powerful tools that can help to facilitate the design of intelligent systems with trust and commitment protocols. Inspired by this work, we propose a multi-valued logic that combines trust and commitment to verify IoT-ad hoc systems where trust and commitment are considered the main communication protocols among the system's agents interacting within uncertain environments. However, unlike [7, 8, 10, 11] that focus on the trust and commitment separately, our paper mainly focuses on modeling and verifying trust and commitments simultaneously by considering a different logic.

In summary, the related proposals in trust management for IoT and ad-hoc networks encompass various trust mechanisms, applications, and contributions. These efforts collectively contribute to addressing the security and trust challenges in these domains, and our research distinguishes itself by integrating model checking to ensure system reliability and by introducing a novel multi-valued logic for trust and commitment verification, contributing to the development of practical intelligent systems in uncertain environments.

4.2.1 Comparative Analysis

We conducted a thorough comparative assessment, leveraging findings from previous research studies, to evaluate the performance and notable contributions of the proposed tool in the context of multi-valued model checking. Our comparisons were grounded in various critical criteria, including the contributions of each tool to the field, their relevance to IoT and ad hoc networks, and their applicability to classical and multi-valued model checking. In Table 4.1, the first group provides an overview of trust-based techniques introduced in IoT and ad hoc networks. However, these approaches do not incorporate model checking to verify the system's intended operation and compliance with requirements. Moving to

the second group of the table, we highlighted studies proposing classical model checking tools for trust and commitments. Notably, these tools lack applicability to multi-valued logic. In the third group of the table, we presented tools specifically designed for applying multi-valued model checking to verify systems with multiple truth values, addressing the concepts of trust and commitment within their logics. The concluding part of the table introduces our tool, named MV-model checker. The evaluation criteria demonstrate that our tool bridges the gap identified in the preceding groups. Notably, our work is applicable to both classical and multi-valued model checking techniques. Furthermore, it effectively handles the verification of systems using mv-TCTLC for combined trust and commitment in IoT-ad hoc systems.

4.3 Hypothesis and Limitations

This section provides a clear outline of the driving hypothesis behind our research and the limitations encountered during the development and application of our proposed method.

4.3.1 Hypothesis

In this research, we hypothesize that a practical verification approach, specifically designed for IoT-ad hoc applications in the realms of smart health monitoring and smart home systems, could effectively incorporate trust and commitments mainly under conditions of uncertainty. By introducing the 3v-TCTLC logic, an extension of the multi-valued CTL, we aim to provide a robust framework for reasoning about uncertainty in IoT-ad hoc systems, using flexible patterns captured by trust over commitments.

4.3.2 Critical Analysis

The proposed work has the following limitations:

Table 4.1: Summary of related work

Reference	Key Contributions	IoT	Ad hoc Network	Trust	Comm.	Applicable for MC	Applicable for MV-MC
[102]	Behavioral modeling		*	*			
[70]	Enhanced security		*	*			
[98]	Low message overhead		*	*			
[114]	Trust-based routing		*	*			
[83]	Security concerns	*	*	*			
[45]	Privacy, authentication		*	*			
[62]	IoT security based on trust	*		*			
[97]	QoE estimation and synthesis			*			
[13]	Profile injection attack detection			*			
[110]	Malware analysis in IoT and Android	*		*			
[44]	Model checking trust			*		*	
[26]	Model checking trust			*		*	
[41]	Model checking trust			*		*	
[25]	Model checking comm.				*	*	
[90]	Model checking comm.				*	*	
[32]	CTL extension to MV-CTL					*	*
[60]	MV logic for MASs					*	*
[7]	MV-model checking trust	*		*		*	*
[11]	MV-model checking trust	*		*		*	*
[8]	MV-model checking trust	*		*		*	*
[6]	MV-model checking comm.	*			*	*	*
[10]	MV-model checking comm.	*			*	*	*
Our approach	MV-model checking combined trust and comm.	*	*	*	*	*	*

1. Scope Limitations: The developed method primarily focuses on uncertainty within the states (i.e., in properties) of IoT-ad hoc systems. It does not address uncertainty in both the states and the transitions. Future research is needed to extend the method to cover this broader scope.

2. Lattice Expansion: The semantics of the 3v-TCTLC logic is designed for a three-valued lattice. While this is suitable for many scenarios, it may not fully capture the complexity of IoT-ad hoc systems. Expanding the semantics to encompass arbitrary lattices with more than three values is an area for future development.

3. Nodes Mobility Modeling: Our tool focuses on design-time modeling and verification rather than runtime. At the design-time, nodes mobility can be captured in our approach using discrete time steps. In this case, the underlying principles of trust and commitment under uncertainty can naturally capture scenarios with mobile nodes. However, capturing mobility in a continuous-time setting is an interesting and challenging extension that will be investigated as future work. For this particular case, a continuous time logic should be considered where a formalization of continuous mobility and its model checking procedure should be introduced.

Acknowledging these limitations, the paper outlines areas for future research and development to expand and enhance the method's capabilities, emphasizing the need for further investigation in these directions.

4.4 Modeling Uncertainty in IoT-Ad Hoc Systems with 3v-TCTLC

The accuracy of models for IoT-ad hoc systems can be compromised when there is insufficient or missing information. Various factors contribute to the absence of information, such as system abstraction, unexpected system behaviors influenced by the environment, partitioning of system space, or incomplete knowledge about system properties [27, 65, 73]. Recently, there has been a growing interest in employing 3v-model checking, which utilizes 3v-logic with truth values of T, M, and F. This logic offers greater adaptability and expressiveness compared to traditional 2-valued logic. This approach has demonstrated its efficacy in addressing problems that necessitate reasoning in uncertain scenarios specific to IoT-ad hoc based systems.

4.4.1 3-Valued TCTLC

In this section, we present three-valued TCTLC (3v-TCTLC) logic to reason about uncertainty in IoT-ad hoc systems. We clarify the impact of uncertain or missing information on the satisfaction of TCTLC formulas in IoT-adhoc systems.

Definition 4.1. (3v-TCTLC model) The model is obtained from the classical model of our TCTLC by extending the latter using the three-valued lattice and replacing the valuation function V by the multi-valuation function $\mathbb{V} : S \rightarrow (\mathrm{AP} \rightarrow L_3)$, a total three-valued labeling function which maps every atomic proposition $a \in AP$ in $s \in S$ to an element from L_3.

Definition 4.2. (3v-TCTLC Syntax) The syntax of 3v-TCTLC logic is equivalent to TCTLC logic, but the formulae are evaluated based on the lattice L_3.

Definition 4.3. (3v-TCTLC Semantics) The semantics of this logic is an extension to (χCTL) [32] considering that we deal with L_3. Below, we add our semantics of the 3v-TCTLC. Given a 3v-TCTLC IoT-ad hoc model and a formula, the satisfaction degree of the formula is defined as follows:

- $\| x \| (s) = (\mathbb{V}(s))(x)$ where $x \in AP$ and $\mathbb{V}$ is defined as in Definition 4.1.
- $\| \varphi \vee \psi \| (s) = \| \varphi \| (s) \sqcup \| \psi \| (s)$ denotes the truth degree according to which $\varphi \vee \psi$ holds in state s.
- $\| \varphi \wedge \psi \| (s) = \| \varphi \| (s) \sqcap \| \psi \| (s)$ denotes the truth degree according to which $\varphi \wedge \psi$ holds in state s.
- $\| \neg \varphi \| (s) = \overline{\| \varphi \|}(s)$ denotes the truth degree according to which φ doesn't hold in state s.
- $\|EX\varphi \| (s) = pre^R_{\exists}(\| \varphi \|)(s) = \bigsqcup_{t \in S} \Big(\| \varphi \| (\mathrm{t}) \sqcap \mathbb{R}(\mathrm{s,t}) \Big)$ where $pre^R_{\exists}(\| \varphi \|)(s)$ represents the degree of truth regarding the existence of a path in the system where φ holds in the next state.

- $\| AX\varphi \| (s) = pre^R_\forall(\| \varphi \|)(s) = \bigsqcap_{t \in S} \Big(\| \varphi \|(\text{t}) \sqcup \neg \mathbb{R}(\text{s,t}) \Big)$ where $pre^R_\forall(\| \varphi \|)(s)$ defines the backward image of state s which specifies the value of φ in the next state of all possible paths.

- $\| EG\varphi \| = \nu\mathbb{Z}. \| \varphi \| \cap_L \| EX\mathbb{Z} \|$ where $\nu\mathbb{Z}$ denotes the greatest fixed point of the global operator G. The semantics articulates the degree of truth regarding the existence of a path in the system where φ globally holds.

- $\| E[\varphi \cup \psi] \| = \mu\mathbb{Z}. \| \psi \| \cup_L (\| \varphi \| \cap_L \| EX\mathbb{Z} \|)$ where $\mu\mathbb{Z}$ indicates the smallest fix point of the formula $\varphi \cup \psi$. The semantics indicates in which truth degree there is a path where φ holds until ψ holds using the smallest fix point of $\varphi \cup \psi$.

- $\| C_{i \to j}(\varphi) \|_{\mathscr{M}} (s) = T$ *iff* $\forall s^{'} \in S \;\; s.t. \;\; s \sim_{i \to j} s^{'}$ we have $\| \varphi \|_{\mathscr{M}} (s^{'}) = T$.
  This semantics states that the satisfaction degree of the formula $C_{i \to j}(\varphi)$ in state s of the system $\mathscr{M}$ is T if and only if the truth degree of φ in all the accessible states $s^{'}$ is T.

- $\| C_{i \to j}(\varphi) \|_{\mathscr{M}} (s) = M$ *iff* $\forall s^{'} \in S \;\; s.t. \;\; s \sim_{i \to j} s^{'}$ we have $\| \varphi \|_{\mathscr{M}} (s^{'}) \neq F$ and $\exists s^{'} \in S \;\; s.t. \;\; s \sim_{i \to j} s^{'}$ and $\| \varphi \|_{\mathscr{M}} (s^{'}) = M$.
  This semantics states that the satisfaction degree of the commitment formula is M if and only if the degree of truth of φ in every $s^{'}$ is not equal to F and there is at least one $s^{'}$ where φ holds with M. In other words, φ in all the accessible states must be T or M because $T \sqcap M \; = M$ and must not be F where $F \sqcap M \sqcap T = F$.

- $\| Fu(C_{i \to j}(\varphi)) \|_{\mathscr{M}} (s) = T$ iff $\exists s^{'} \in S \; s.t. \;\; s^{'} \sim_{i \to j} s$ and $\| C_{i \to j}(\varphi) \| (s^{'}) = T and \| C_{i \to j}(\varphi) \| (s^{'}) \neq M$.
  This semantics states that the satisfaction degree of the formula $Fu(C_{i \to j}(\varphi))$ in state s of the system $\mathscr{M}$ is T if and only if the truth degree of the formula $C_{i \to j}(\varphi)$ in state $s^{'}$, which has accessibility to s, is T and is not equal to Mas $T \sqcap M \; = M$.

- $\| Fu(C_{i\to j}(\varphi)) \|_{\mathscr{M}}(s) = M$ iff $\exists s' \in S \; s.t. \; s' \sim_{i\to j} s$ and $\| C_{i\to j}(\varphi) \|(s') = M$.

  This semantics states that the satisfaction degree of the formula $Fu(C_{i\to j}(\varphi))$ in state s of the system $\mathscr{M}$ is M if and only if the truth degree of the formula $C_{i\to j}(\varphi)$ in state s', which has accessibility to s, is M.

- $\| (T_{i\to j}(\varphi)) \|_{\mathscr{M}}(s) = T$ iff $\| C_{j\to i}(\varphi) \|_{\mathscr{M}}(s) = T$ and $\exists s' \in Pas(s)$ we have $\| Fu(C_{j\to i}(\varphi)) \|_{\mathscr{M}}(s') = T$.

  This semantics states that the satisfaction degree of the formula $T_{i\to j}(\varphi)$ in state s of the system $\mathscr{M}$ is T if the truth degree of the formula $C_{i\to j}(\varphi)$ in s is T and the truth degree of formula $Fu(C_{j\to i}(\varphi))$ in a state s', which has reachability to s, is also T.

- $\| (T_{i\to j}(\varphi)) \|_{\mathscr{M}}(s) = M$ iff $\| C_{i\to j}(\varphi) \|_{\mathscr{M}}(s) = M$ and $\exists s' \in Pas(S)$ we have $\| Fu(C_{i\to j}(\varphi)) \|_{\mathscr{M}}(s') = M \vee T$.

  This semantics states that the satisfaction degree of the formula $T_{i\to j}(\varphi)$ in state s of the system $\mathscr{M}$ is M if and only if the truth degree of the formula $C_{i\to j}(\varphi)$ in s is M and the truth degree of the formula $Fu(C_{j\to i}(\varphi))$ in a state s', which has reachability to s, is M or T because $M \sqcap T = M$ and $M \sqcap M = M$.

- $\| (T_{i\to j}(\varphi)) \|_{\mathscr{M}}(s) = M$ iff $\| C_{i\to j}(\varphi) \|_{\mathscr{M}}(s) = T$ and $\exists s' \in Pas(S)$ we have $\| Fu(C_{i\to j}(\varphi)) \|_{\mathscr{M}}(s') = M$.

  This semantics states that the satisfaction degree of the formula $T_{i\to j}(\varphi)$ in state s of the system $\mathscr{M}$ is M if and only if the truth degree of the formula $C_{i\to j}(\varphi)$ in s is T and the truth degree of the formula $Fu(C_{j\to i}(\varphi))$ in a state s', which has reachability to s, is M.

- $\| x \|(s)$ denotes the truth degree according to which x holds in s.

- $\cap_L$ refers to the multi-valued intersection defined within the algebra L.

- $\cup_L$ refers to the multi-valued union defined within the algebra L.

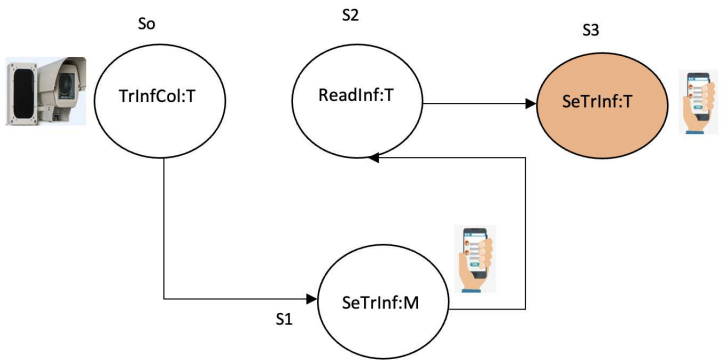

Figure 4.2: A scenario of smart transportation system

Example: Consider the 3v-IoT ad hoc system, which presents a scenario of a "smart transportation system based on IoT devices and an ad hoc network" (Figure 4.2). This system is being implemented in a city with the aim of improving traffic flow and mitigating congestion. It incorporates sensors and cameras installed at critical intersections to collect real-time information on traffic volume and movement. Upon processing this data, the system promptly sends alerts to users through a mobile app. An AI algorithm analyzes the data to optimize traffic signal timings and regulate vehicle flow. Additionally, the system is committed to providing drivers with up-to-date traffic updates and alternative routes via the mobile app, thus establishing trust between the system and its users. Ultimately, the system strives to reduce traffic congestion and enhance overall traffic efficiency within the city.

Three-Valued Sets and Relations

Following the logic of χCTL, our 3v-TCTLC logic depends on the concepts of the multi-valued sets (mv-sets) and the multi-valued relations (mv-relations) where these sets and relations are computed based on the values of the propositions variables taken from the 3v-lattice. Below we provide a simple calculation example of how we get the mv-sets and mv-relations.

3v-relations 3v-sets: The 3v-sets of the variables *SeTrInf* and *ReadInf*, for example, is

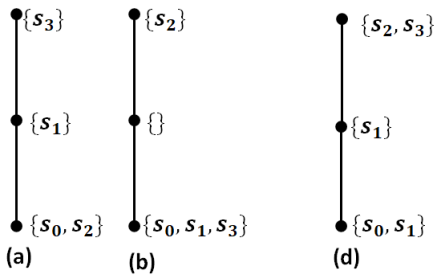

Figure 4.3: (a) 3v-sets of variable SeTrInf; (b) 3v-sets of variable ReadInf; (d) 3v-sets of (‖SeTrInf ‖ ⊓ ‖ ReadInf‖)

denoted by ‖SeTrInf ‖ and ‖ ReadInf ‖ respectively, and the multi-valued sets of the join of these variables formally written as (‖ SeTrInf ‖ ⊓ ‖ ReadInf‖). The 3v-sets from the two variables are shown in Figure 4.3 (a) and (b). In (a), we mapped state s_3 to the T of the lattice L_3 because the variable "SeTrInf" has the T value in this state, and s_0 and s_2 mapped to the value F because the same variable doesn't satisfy in these states. State s_1 is mapped to the value M as "SeTrInf" has the value M in this state. Similarly, we get the 3v-set of the variable "ReadInf" in (b). The join of "SeTrInf" and "ReadInf" is shown in (d) where the states are mapped to the L_3 truth values based on the join operator explained in Figure 2.7.

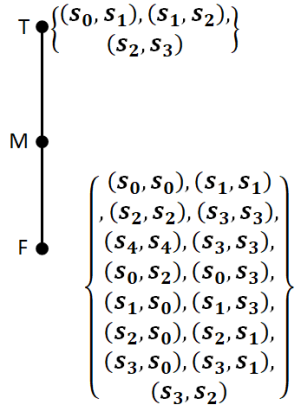

Figure 4.4: 3v-relation of 3v-IoT-ad hoc model

A 3v-relation $\mathbb{V}$ on two sets a and b is a 3v-set on $a \times b$. Figure 4.4 shows the

3v-relations of our model that represent the values of the transition between states. In the figure, the relation (s_0, s_1), (s_1, s_2) and (s_2, s_3) are mapped to T because these relations have the same truth values in the system. The other relations are mapped to the value F, as these relations do not exist in the system. At the same time, an empty set is mapped to the value M as no state relations have this value in the system.

In this model, In state S_0 system collects the information from sensors and camera. It is uncertain in S_1 if the system send the traffic information to the user or not (SeTrInf=M) and in S_3, the system commits to send traffic information after information updating in S_2. We suppose that we need to verify the truth degree of the trust formula "the user trusts to the system commitment regarding to send the traffic information after traffic information updates" in state S_3 between smart system and user which is represented by $\| (T_{user \to sys}(SeTrInf)) \| (s_3)$. The algorithm starts by checking first if the current state (S_3) satisfies $\| (C_{sys \to user}(SeTrInf)) \|$ formula which means if the system commits to send the traffic information updates. Then checks the truth degree of prior fulfilment of the commitment formula $\| (Fu_{sys \to user}(SeTrInf)) \|$, which defines that the system has sent the traffic information before update. The degree of truth of this formula is established based on the truth degrees of fulfilment of atomic proposition (SeTrInf) in the state $(S_1 = T)$ and commitment to this proposition in the state $(S_3 = M)$ that we obtain $T \sqcap M \ = M$, which means it is unknown whether the user trusts the system's commitment to send updated traffic information.

4.5 Three-Valued Model Checking IoT-Ad Hoc Systems with mv-TCTLC

In general, there are two main algorithms used for performing three-valued model checking. The first algorithm, known as the direct algorithm [32, 69, 87], handles the 3v-model

checking problem directly and employs a more expressive syntax and semantics. The second algorithm is indirect or reduction-based approach. Reduction algorithms transform the 3-valued model checking problems into classical ones by considering various scenarios and offer two key advantages: the ability to utilize existing model checking tools and the efficiency in dealing with the state explosion problem.

The purpose of this article is to develop an indirect 3-valued verification technique to reason about uncertainty in IoT-ad hoc systems. This technique will enable the verification of various 3v-properties based on trust and commitment relationships between agents. To achieve this, we need to reduce the three-valued model to the classical two-valued model. To do so, we introduce the reduction algorithm by extending the reduction technique introduced and implemented as software tool named "MCMAS-interactor" in [12]. We extended this tool which automatically interacts with MCMAS-TC, by adding new functionalities to make the tool capable of handling our 3v-TCTLC logic and automatically dealing with our MCMAS-TC tool. In fact, MCMAS is an existing model checker specifically designed for verifying various agent-based logics and is an open-source model checker that is widely used in various research projects due to its efficacy, versatility, and user-friendly interface. The agent-based IoT-ad hoc systems in MCMAS are modeled using ISPL, a programming language with a semantics similar to interpreted systems [53]. This simplifies the specification and reasoning about IoT-ad hoc models and allows MCMAS to integrate with other tools and frameworks. The idea behind this new developed tool [2] is to generate classical TCTLC models from the 3v-TCTLC to apply the model checker MCMAS-TC which we developed for verifying classical TCTLC models in [16].

In this technique, the number of the classical models equals the number of the join-irreducible elements in lattice L_3, which are T and M. One classical model considers the atomic propositions with truth values T as true and the others (F and M) as false. The

[2]The tool and the case studies are accessible online at: `https://www.dropbox.com/scl/fi/whmnaa5datn2kbpeflnzg/NargesTool.zip?rlkey=a4dv7k35qv13yjix525i2qu4e&dl=0`

second one considers the atomic propositions with truth values M and upper (M and T) as true and the others as false. The next step is to verify the same formula over the two classical models by calling MCMAS-TC. Finally, we make the approximation for the final results as follows: if the result of the model with M and T is true, then we consider the join-irreducible M. If not, we consider the answer an empty set. The same applies to the model that considers only T as true. Formally, the final results are obtained as follows: $\emptyset \sqcup M = M, \emptyset \sqcup T = T, \emptyset \sqcup \emptyset = F$.

Algorithm 4.1 Transform 3v-model $\mathbb{M}_m = (S, R_t, \{\sim_{i\to j}: (i,j) \in A^2\}, I, \mathbb{V})$ into $M_t = (S, R, \{\sim_{i\to j}: (i,j) \in A^2\}, I, V)$ and $M_f = (S, R, \{\sim_{i\to j}: (i,j) \in A^2\}, I, V)$

1: **Input** the 3v-model $\mathbb{M}_m$ and comm. trust formula φ
2: **Output** the classical models M_t and M_f
3: $S_t = S_{\mathbb{M}}$ and $S_f = S_{\mathbb{M}}$
4: $I_t = I_{\mathbb{M}}$ and $I_f = I_{\mathbb{M}}$
5: Initialize $R_t = \emptyset$ and $R_f = \emptyset$
6: Initialize $(V_t(S_t))(x) = (\mathbb{V}_m(S_m))(x)$ and $(\mathbb{V}_m(S_m))(x) = M \Rightarrow (V_t(S_t))(x) = T$
7: Initialize $(V_f(S_f))(x) = (\mathbb{V}_m(S_m))(x)$ and $(\mathbb{V}_m(S_m))(x) = M \Rightarrow (V_f(S_f))(x) = F$
8: **Get** the sets t and f
9: **Call procedure TRANSandLINK(t,f,$\mathbb{M}_m$)**
10: **Call MCMAS-TC on M_t and φ**
11: **Call MCMAS-TC on M_f and φ**

Algorithm 4.1 shows the steps of our 3v-TCTLC model transformation to the two classical TCTLC models. Line 1 inputs the 3v-TCTLC model $\mathbb{M}_m$ and the formula. Line 2 outputs the two classical models M_t, which treats M as true and M_f, which treats M as false. Line 3 gives the two classical models the same number of states in $\mathbb{M}_m$. Line 4, gives the two classical models the same initial states in $\mathbb{M}_m$. Line 5 initializes the relations of the states in each classical model by empty sets. Line 6 gives each state in M_t the same label of atomic propositions in $\mathbb{M}_m$ and then changes the value M to T. The same applies in line 7, changing the value M to F. Line 8 gets the sets of the obtained states for each model. Line 9 calls procedure 4.1 to establish the transitions and accessibility relations between states.

Procedure 4.1 TRANSFORM 3V-TCTLC MODEL

1: **procedure** TRANSANDLINK($t, f, \mathbb{M}_m$)
2: **for each** $(s_m, s_m^{'}) \in S_m^2$ **do**
3: **if** $(s_m, s_m^{'}) \in R_m$ **then**
4: $R_t := R_t \cup \{(s_m, s_m^{'})\}$
5: **if** $s_m \sim_{i \to j} s_m^{'}$ **then** $\mathrm{R}_t = \mathrm{R}_t \cup s_m \sim_{i \to j} s_m^{'}$
6: **end if**
7: **end if**
8: **end for**
9: **for each** $(s_m, s_m^{'}) \in S_m^2$ **do**
10: **if** $(s_m, s_m^{'}) \in R_m$ **then**
11: $R_f := R_f \cup \{(s_m, s_m^{'})\}$
12: **if** $s_m \sim_{i \to j} s_m^{'}$ **then** $\mathrm{R}_f = \mathrm{R}_f \cup s_m \sim_{i \to j} s_m^{'}$
13: **end if**
14: **end if**
15: **end for**
16: **Return** (M_t, M_f)
17: **end procedure**

Line 1 in the procedure 4.1 gets the sets of states for each classical model with the complete 3v-model. Lines from 2 to 8 relate the states for the model M_t with the same transitions and accessibility relations found in $\mathbb{M}_m$. The same applies for M_f through the lines from 9 to 15. Then the procedure returns the complete classical models to be verified by MCMAS-TC in lines 10 and 11.

We perform the introduced verification technique in smart health monitoring system and smart home system to evaluate the effectiveness of our proposed approach. Over these systems, we verify the properties related to the trust on commitments concept. These properties are categorized into the following three distinct properties: safety property is related to the absence of any bad situations, reachability property is related to a specific sequence of computations that will lead to the initial state, and liveness property is related to a good situation that will eventually happen. Our verifications are conducted on Dual-Core CPU T4500 at 2.30 GHz processor and 64-bit operating system.

The selection of application fields of smart health monitoring and smart home for our approach was driven by several considerations:

1- Relevance to IoT-Ad Hoc Systems: In modern IoT deployments, ad hoc communication plays a vital role, especially in environments like smart homes and healthcare facilities. These scenarios often involve interconnected devices that need to self-organize and communicate efficiently, making them a suitable testbed for evaluating our approach.

2- Real-World Practicality: We aim to ensure that our approach is applicable to real-world situations. Smart health monitoring and smart home systems are representative of practical IoT use cases, and their evaluation allows us to assess the effectiveness and applicability of our method in situations that closely resemble the environments where IoT-ad hoc systems are deployed.

In summary, smart health monitoring and smart home systems are relevant, practical, and representative of the challenges faced in IoT-ad hoc systems. We believe that evaluating our approach in these application fields enhances the practicality and real-world applicability of our research, and it allows us to demonstrate the adaptability of our method to a wide range of IoT environments.

These scenarios are related to ad hoc systems, depending on the specific requirements and use cases. Ad hoc networks are characterized by their decentralized nature, where devices can communicate directly with each other without relying on a centralized infrastructure like traditional networks. Here is how ad hoc networks can be applied in these case studies:

Smart Health Monitoring:

- Mobile Health Devices: Ad hoc networks can be utilized when mobile health monitoring devices (e.g., wearable sensors, medical IoT devices) need to communicate with each other or with a central monitoring station. For example, in a hospital setting, patient-worn sensors can form an ad hoc network to transmit vital signs to a central monitoring system.

- Emergency Response: In emergency situations or during patient transfers, ad hoc networks can establish quickly to ensure continuous monitoring and data transmission. This flexibility is crucial for providing immediate care and maintaining connectivity.

Smart Home Systems:

- Home Automation: Ad hoc networks can be employed for home automation, where various smart devices (e.g., smart thermostats, lights, security cameras) communicate with each other. These devices can form ad hoc connections to coordinate actions, such as adjusting lighting based on occupancy or triggering security alerts.

- Ad hoc Networking: ad hoc networks are particularly useful in smart homes. Devices can act as both endpoints and routers, creating a self-healing network. If one device moves or loses connectivity, others can reroute data, ensuring robust communication.

4.5.1 Case Study 1: Three-Valued Model Checking a Smart Health Monitoring System with 3v-TCTLC under Uncertainty

We present a formal model depicting a scenario involving a Smart Health Monitoring System that incorporates uncertainty. The system involves a patient equipped with

intelligent wearable devices containing sensors for monitoring blood pressure and glucose levels. Five agents are involved in this system: Patient, Hospital, Glucose Sensor, Blood Pressure Sensor, and Database. Interactions among these agents are predicated on fulfillment, commitment, and trust. The system initiates in state S_0, wherein the glucose and blood pressure sensors record high levels of glucose and pressure in the patient's body. In state S_1, it remains uncertain whether the glucose sensor promptly fulfills the action of alerting the patient through a smartphone application. Similarly, in S_2, the uncertainty lies in whether the blood pressure sensor promptly fulfills the action of alerting the patient via the smartphone application. Concurrently, the hospital receives no readings from the sensors in this state. Upon the sensors updating their readings in S_3, the hospital remains unaware

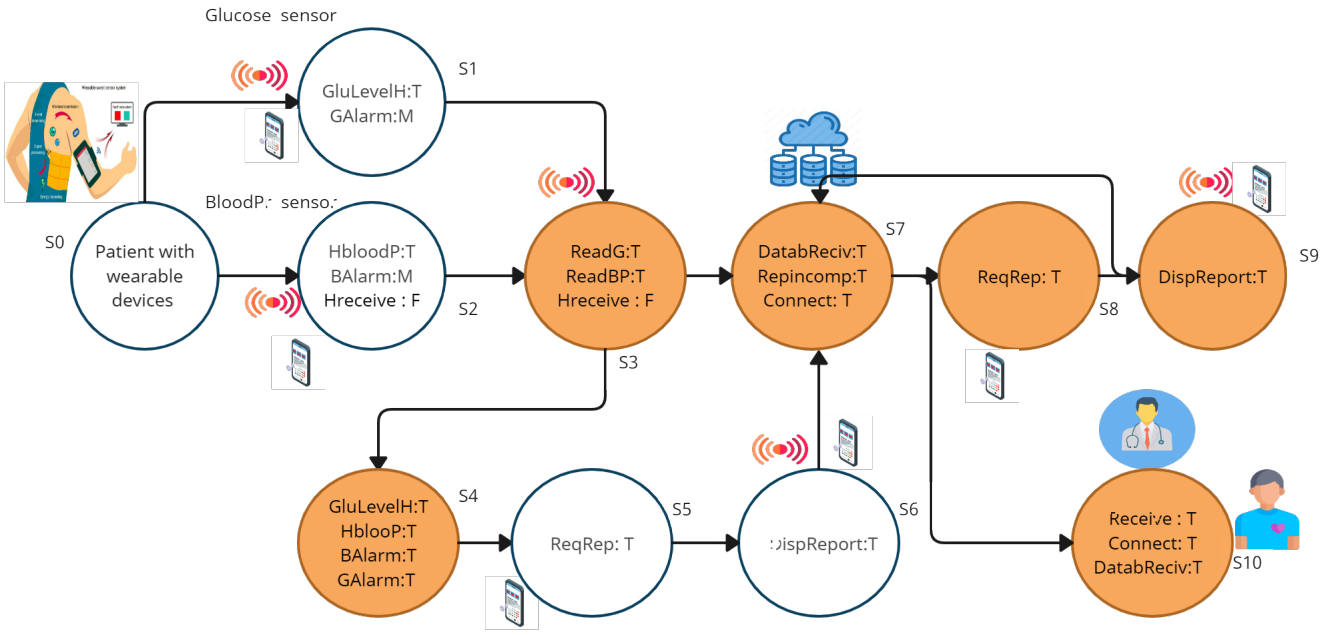

Figure 4.5: Smart health monitoring system

of these updated readings. In state S_4, when the measured levels change, the sensors commit to alerting the user, while the user places trust in the sensors to send alerts. The application fulfills the action of requesting the sensors to provide a comprehensive report in S_5. Subsequently, the sensors fulfill the action of committing to display their reports in S_6,

followed by transmitting the updated information to the database in S_7. In S_8, if the reports are incomplete or missing certain information, the application commits to requesting the reports again from the sensors, and the sensors, in turn, trust the application for the same action. The sensors fulfill the action of displaying the reports, and the application trusts the sensors for the same action in S_9. The application re-sends the reports to the database for processing and classification of the received information. Subsequently, the hospital staff commit to contacting the patient for further treatments, and the patient places trust in the hospital for the same action in state S_{10}.

System Specifications

- Safety properties
  - It is not the case that the sensor fulfills reporting high blood pressure levels, and the system doesn't trust the sensor's blood pressure readings.
    $\varphi_1 = AG\,\neg(Fu(BSen, Sys, Hblood) \wedge \neg T(Sys, BSen, Hblood))$;
  - A bad situation to be avoided is that the system fulfils to request the report and the sensor never trusts the system with regard to requesting report.
    $\varphi_2 = AG\,\neg(Fu(Sys, BSen, ReqRep) \wedge \neg T(BSen, Sys, ReqRep))$;
  - It is not the case that the hospital staff fulfils to connect the patient for further information and the patient does not trust to connect.
    $\varphi_3 = AG\,\neg(Fu(Hospital, Pat, Connect) \wedge \neg T(Pat, Hospital, Connect))$;
- Reachability properties
  - The patient will not trust the glucose level sensor to send an alert until the sensor has fulfilled the commitment to send the alert.

$\varphi_4 = E\ (\neg Fu(GluSen, Pat, GAlarm)\ U(Fu(GluSen, Pat, GAlarm) \wedge$
$T(Pat, GluSen, GAlarm)))$;

– The system will not trust the blood pressure sensor to send an alert until the sensor has fulfilled the commitment to send the alert.
$\varphi_5 = E\ (\neg Fu(BSen, Sys, BAlarm)\ U(Fu(BSen, Sys, BAlarm) \wedge$
$T(Sys, BSen, BAlarm)))$;

– The blood pressure sensor will not trust the system to request the report until the sensor has fulfilled the underlying commitment.
$\varphi_6 = E\ (\neg Fu(Sys, BSen, ReqRep)\ U(Fu(Sys, BSen, ReqRep) \wedge$
$T(BSen, Sys, ReqRep)))$;

• Liveness properties

– The hospital fulfills receiving information, and there exists a future computation that the hospital commits to receive the information and the patient trusts the hospital to receive this information.
$\varphi_7 = EF(Fu(Hospital1, Pat1, Hreceive) \wedge EF(SC(Hospital1, Pat1, Hreceive) \wedge$
$T(Pat1, Hospital1, Hreceive)))$;

– The glucose sensor fulfills the action of alerting, and there exists a future computation that the glucose sensor commits to alert, and the patient trusts.
$\varphi_8 = EF(Fu(GluSen, Pat, GAlarm) \wedge EF(SC(GluSen, Pat, GAlarm) \wedge$
$T(Pat, GluSen, GAlarm)))$;

– The blood pressure sensor fulfills the commitment to displaying the report and

there exist a future computation that the blood pressure sensor commits to report display, and the system trusts.

$\varphi_9 = EF(Fu(BSen, Sys, DispReport) \wedge EF(SC(BSen, Sys, DispReport) \wedge T(Sys, BSen, DispReport)))$;

– The blood pressure sensor fulfills the commitment to send the alert, and there exists a future computation that the sensor commits to sending the alert, and the system trusts.

$\varphi_{10} = EF (Fu(BSen, Sys, BAlarm) \wedge EF(SC(BSen, Sys, BAlarm) \wedge T(Sys, BSen, BAlarm)))$;

Moreover, we checked φ_{11} to verify if the hospital commits to the patient receiving the information, the patient will trust the hospital regarding the receiving information.

$\varphi_{11} = AG\ (SC(Hospital, Pat, Hreceive) \wedge$

$T(Pat, Hospital, Hreceive))$;

System Verification

We verify our system by extending the Java tool named MCMAS-interactor developed in [12]. As shown in Figure 4.6, we added the buttons "Upload 3v-TCTLC", "Generate Negative TCTLC" and "Generate Positive TCTLC" for performing the implementation of our approach. Using the tool, we upload the 3v-TCTLC model (Figure 4.6). Then, the tool generates two classical TCTLC from the 3v-model. Next, we call MCMAS-TC over each model by pressing the "launch MCMAS" button (Figure 4.7) and checks if the model with True value (M.T) yields F then the result is "False"; if not, it checks the model with value False (M.F), if yields T the result returns "True"; otherwise, the result will be "Maybe".

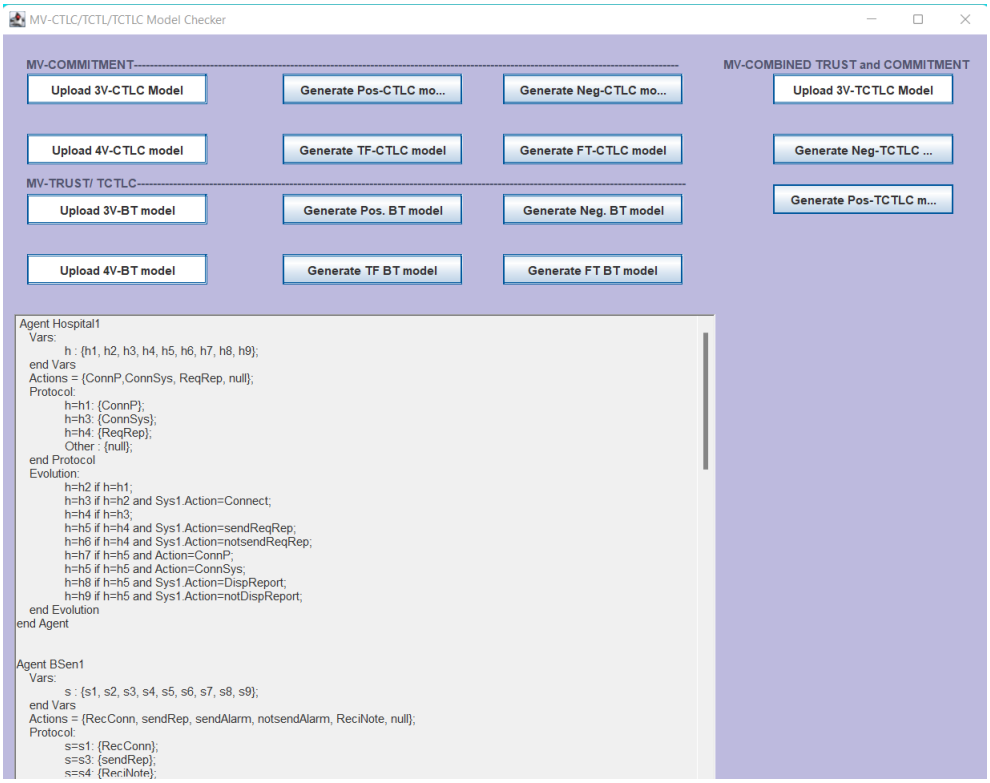

Figure 4.6: Uploading the 3v-TCTLC model

After verifying our smart health monitoring system and considering two classical models ((M.T) and (M.F)), we obtained that in the behaviour of our system exists missing information on the properties φ_4, φ_5, φ_8 and φ_{10}. The missing information in Figure 4.5 has caused uncertainty as to whether these properties are being satisfied by the system. However, the properties φ_1, φ_2, φ_3, φ_6 and φ_9 verified as True and the properties φ_7 and φ_{11} are not satisfied in the system. As indicated by the results presented in Table 4.2, a refinement of our system is necessary regarding the properties having values M and F.

In order to check the effectiveness and scalability of our proposed technique, we performed eight experiments, starting with 5 agents and increasing up to 40 agents. Table 4.3 displays the number of agents (#Age.), the number of reachable states (#St.), and the average of the memory (Ave.(MB)) in use and the execution time (Ave.time(S)) of two classical models (M.T) and (M.F) in seconds. The results in the table indicate that the

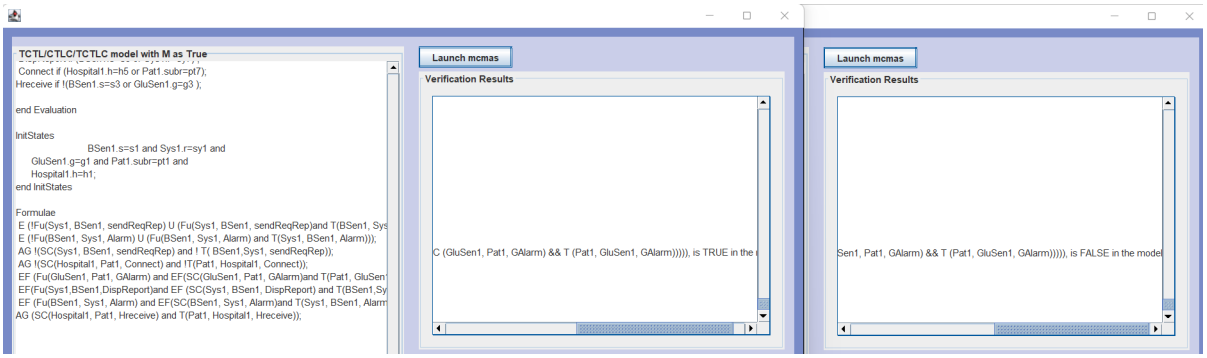

Figure 4.7: The verification process shows the uncertain results

Table 4.2: The verification results of the smart hospital model

Pro.	M.T	M.F	Result
φ_1	T	T	T
φ_2	T	T	T
φ_3	T	T	T
φ_4	T	F	M
φ_5	T	F	M
φ_6	T	T	T
φ_7	F	F	F
φ_8	T	F	M
φ_9	T	T	T
φ_{10}	T	F	M
φ_{11}	F	F	F

state space grows exponentially as the number of agents increases, while the average of the memory usage and the execution time for each model increase in a polynomial manner. These findings confirm the high scalability of our approach.

4.5.2 Case Study 2: Three-Valued Model Checking a Smart Home System with 3v-TCTLC under Uncertainty

For further experimentation, we model and verify our second system, a Smart Home System with fulfillment, trust, and commitment under uncertain settings. The system comprises six

Table 4.3: Scalability results of running the tool 8 times, starting with 5 and ending with 40 agents.

#Age.	#St.	Ave.(MB)	Ave.time(S)
5	28	11	<0.001
10	340	12	<0.01
15	4828	16	0.5
20	72100	24	1.5
25	1.10763e+06	26	3.5
30	1.73087e+08	30.3	4.5
35	2.73218e+08	30.9	7
40	4.33801e+09	34	11

agents: Fire Sensor, Smog Sensor, Motion Sensor, Temperature Sensor, Fire Department, and Smart Application used by the user for device control and connection with the fire department. The functionalities of this system are as follows: In state S_0, when the smog sensor detects smog in the room, it fulfills the action of sending an alert to the user via a smartphone application in S_5. In state S_1, if the smog level increases, the smog sensor commits to re-alerting the user and the fire sensor, and both trust the former to fulfill this commitment in S_2. Starting from S_3, in the event of a fire risk, the system sends a fire alert to the user in S_5. However, if there is no fire risk, the fire sensor notifies the user in S_5 and refrains from connecting with the fire department in S_4.

Furthermore, when the motion sensor detects motion near the door in state S_6, it fulfills the action of sending an alert to the application in S_7. The user then decides whether the application will automatically open the door in S_{13}. Additionally, when the doorbell rings, the motion sensor commits to notifying the application, and the application trusts the motion sensor to fulfill this commitment in S_9. In state S_{10}, there is uncertainty regarding whether the temperature sensor fulfills the action of sending temperature readings to the thermostat. Likewise, in S_{12}, there is uncertainty regarding the commitment of the temperature sensor to send the temperature readings and the trust of the thermostat in the

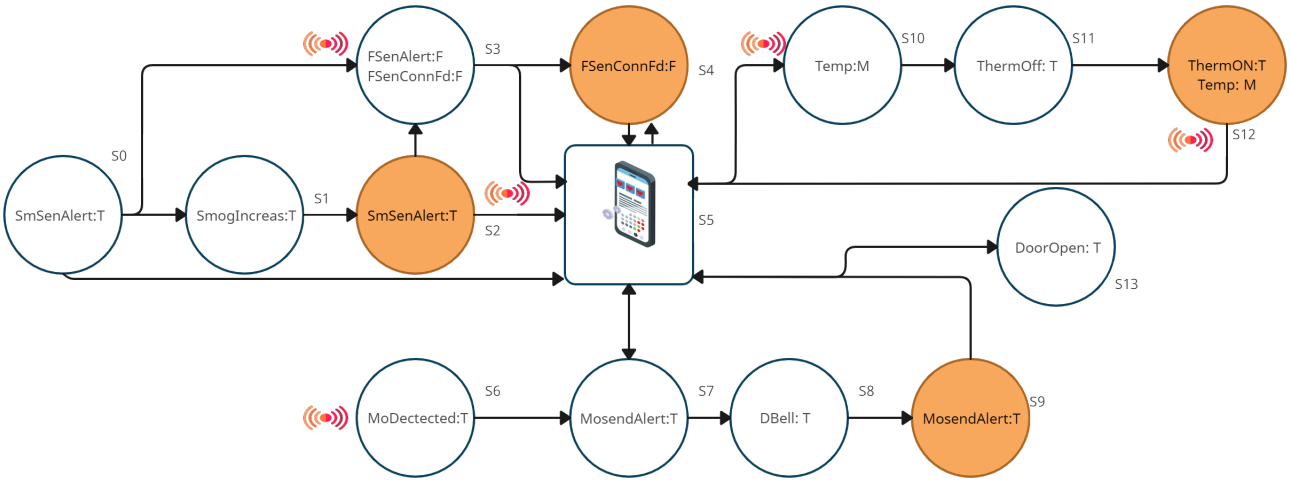

Figure 4.8: Smart home system

sensor's fulfillment of this action.

System Specifications

- Safety properties
  - It is not the case that the motion sensor commits to sending an alert when detecting movement, and application doesn't trust the sensor to send the alert.
    $\varphi_1 = AG\,\neg(SC((MoSen1, App1, MosendAlert)$
    $\wedge\,\neg T(App1, MoSen1, MosendAlert))$;
  - It is not the case that the motion sensor fulfils to sending an alert to the application, and the latter never trusts the motion sensor concerning sending the alert.
    $\varphi_2 = AG\,\neg(Fu(MoSen, App, MosendAlert)$
    $\wedge\,\neg T(App, MoSen1, MosendAlert))$;
- Reachability properties

- The temperature sensor commits to sending the current temperature degree to the thermostat, and the latter trusts the former to provide this temperature reading.

  $\varphi_3 = EF\ (SC(TempSen, Therm, Temp) \wedge T(Therm, TempSen, Temp))$;

- The application will not trust to the smog sensor to send alert until the smog sensor has fulfilled the commitment to send the alert.

  $\varphi_4 = E\ (\neg Fu(SmSen, App, SmSenAlert)\ U$
  $(Fu(SmSen, App, SmSenAlert) \wedge T(App, SmSen, SmSenAlert)))$;

- The thermostat will not trust the temperature sensor to send the temperature readings until the sensor has fulfilled the commitment to send the temperature readings.

  $\varphi_5 = E\ (\neg Fu((TempSen, Therm, Temp)\ U$
  $(Fu(TempSen, Therm, Temp) \wedge T(Therm, TempSen, Temp)))$;

- liveness properties

  - The fire sensor fulfills its commitment to the application by connecting the fire department upon detecting a fire, and the application trusts the sensor to initiate this connection.

    $\varphi_6 = AG\ (Fu(FireSen, App, FirSenConnFd) \wedge$
    $EF(SC(FireSen, App, FirSenConnFd) \wedge T(App, FireSen, FirSenConnFd)))$;

  - The smog sensor fulfills the commitment to send the alert to the fire sensor, and there exists a future computation that the smog sensor commits to sending the alert, and the fire sensor trusts.

$$\varphi_7 = EF\ (Fu(SmSen, FireSen, SmSenAlert)$$
$$\wedge EF(SC(SmSen, FireSen, SmSenAlert) \wedge T(FireSen, SmSen, SmSenAlert)));$$

– The temperature sensor fulfills the commitment to send the temperature readings and there exist a future computation that the temperature sensor commits to sending the temperature, and the thermostat trusts.
$\varphi_8 = EF\ (Fu(TempSen, Therm, Temp) \wedge EF(SC(TempSen, Therm, Temp) \wedge$
$T(Therm, TempSen, Temp)));$

We verified in φ_9 if the fire sensor commits to sending a fire alert to the fire department whenever a smog is detected, and the latter trusts the first to send the alert .

$\varphi_9 = AG\ (SC(FireSen, App, FSenConnFd) \wedge T(App, FireSen, FSenConnFd));$

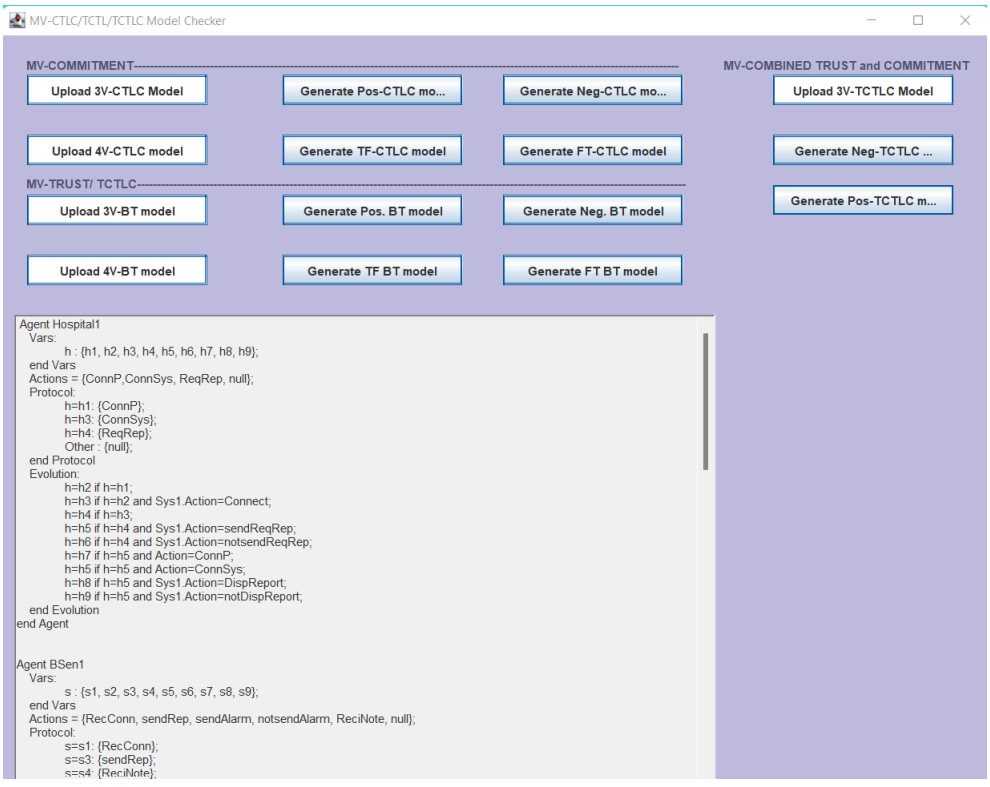

Figure 4.9: Uploading the 3v-TCTLC model

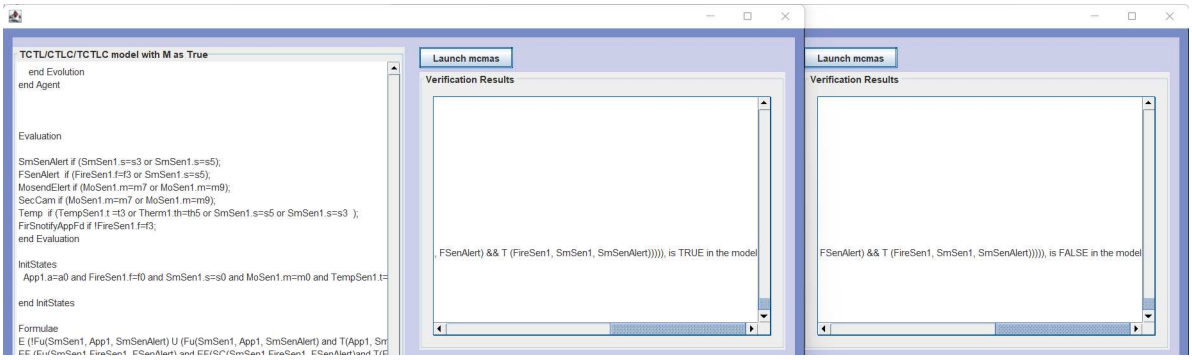

Figure 4.10: The verification process shows the uncertain results

System Verification

Using the MCMAS-interactor tool with our extensions, we verified our second case study. Figure 4.9 shows the uploading of the 3v-smart home system. by pressing the buttons "Generate Pos TCTLC" and "Generate Neg TCTLC", we get the classical models. Then by pressing the "launch mcmas" button we get the verification results of the two systems as shown in Figure 4.10.

Table 4.4: The verification results of the Smart Hospital model

Pro.	**M.T**	**M.F**	**Result**
φ_1	T	T	T
φ_2	T	T	T
φ_3	T	F	M
φ_4	T	T	T
φ_5	T	F	M
φ_6	F	F	F
φ_7	T	T	T
φ_8	T	F	M
φ_9	F	F	F

We evaluated our smart home system and compared the two classical models with True value (M.T) and False value (M.F). This has led to uncertainty regarding whether

properties φ_3, φ_5, and φ_8 are being met by the system as of the missing information represented in Figure 4.10. On the other hand, properties φ_1, φ_2, φ_4 and φ_7 were confirmed to be True, while φ_6 and φ_9 where not satisfied by the system. Our analysis of the results in Table 4.4 revealed that improvement of the system is required for properties φ_3, φ_5, φ_6, φ_8 and φ_9 which have values M and F.

We checked the effectiveness and scalability of our proposed technique, we conducted

Table 4.5: Scalability results of running the tool 8 times, starting with 6 and ending with 42 agents.

#Age.	**#St**	**Ave.(MB)**	**Ave.time(S)**
6	19	10	<0.001
12	167	11	<0.01
18	1615	14	0.5
24	16415	21	1
30	169279	23.3	1.5
36	1.74605e+06	23.6	3
42	1.79356e+07	29	5

eight experiments with a range of agents from 6 to 48. The results in Table 4.5 show that the state space #St increases exponentially as the number of agents (#Age) increases, and the average memory usage (Ave.(MB)) and execution time (Ave.time(S)) increase in a polynomial manner. These outcomes ensure the high scalability of our approach.

4.5.3 Comparative Analysis

Additionally, we considered baseline methods to enhance the comparative study and ensure a robust evaluation of our approach against existing techniques. Because practical tools for system verification with multiple truth values are rather limited, we benchmarked our current findings against an existing classical verification tool that support the notion of combined trust and commitment [22] and against tools which include the notion of trust

Table 4.6: Comparison between proposed and existing techniques

#St.	Memory (MB)	Time (Sec.)	#St.	Memory (MB)	Time (Sec.)
[44]			Proposed Technique (case study 1)		
17	9	0.09	28	11	<0.001
289	15	0.424	340	12	<0.01
4913	28	0.991	4828	16	0.5
85521	42	4.137	72100	24	1.5
1.41986E+06	45	11.971	1.10763e+06	26	3.5
2.41376E+07	39	12.127	1.73087e+08	30.3	4.5
[47]			2.73218e+08	30.9	7
12	4.241	0.020	4.33801e+09	34	11
446	5.507	0.184	[22]		
4224	12.057	2.736	17	10.67	<0.01
33454	15.432	63.687	111	11.28	<0.01
238787	83.839	630.914	818	12.46	<0.01
[50]			5836	14.49	0.58
10	8.6	<0.01	40528	16.60	1.50
43	8.971	<0.01	278030	19.53	2.40
239	9.958	<0.01	1.90766e+06	22.43	2.66
1597	12.056	<0.01	1.37581e+07	31.90	4.33
11545	16.856	1	2.21354e+08	40.28	11.62
88055	36.134	2	5.61637e+09	132.94	101.66
708461	45.592	8	1.3327e+11	133.24	115.50
6.01734e+06	56.28	29			
5.25729e+07	94.36	426			
4.59517e+08	153.008	1128			

or commitment in their logics [44, 47]. Table 4.6 provides a comprehensive comparison of the verification results in [22, 44, 47] and our technique. This allows for a detailed examination of performance metrics, facilitating a more informed assessment. Our results are prominently featured in the table, ensuring a comprehensive evaluation. As observed in the table, our approach excels in scalability and execution time when contrasted with the benchmarks. Specifically, our technique demonstrates superior scalability, handling a larger number of states before encountering the state explosion problem, which is evident in the benchmarks. While there is a slight increase in memory usage and execution time in our approach, the trade-off results in enhanced scalability and efficiency compared to the existing techniques. This reinforces the effectiveness of our proposed approach in

addressing the challenges associated with state explosion and verifying systems with a significant number of states. Figures 4.11 and 4.12 illustrate that our approach utilizes significantly less memory and time as the number of states increases, especially when the number of states is at its maximum. The verification process in our technique outpaces the benchmarks, demonstrating faster performance even with a higher number of states.

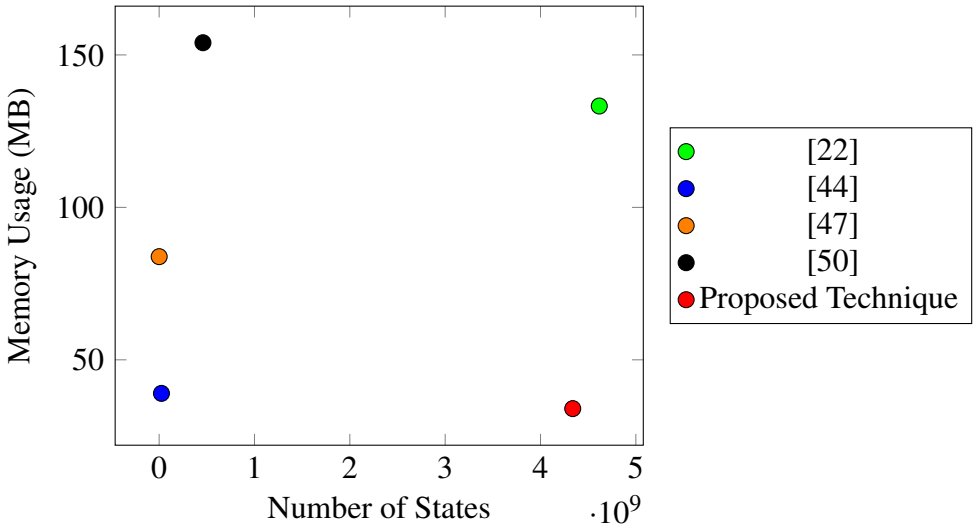

Figure 4.11: Comparison result for memory usage and number of states

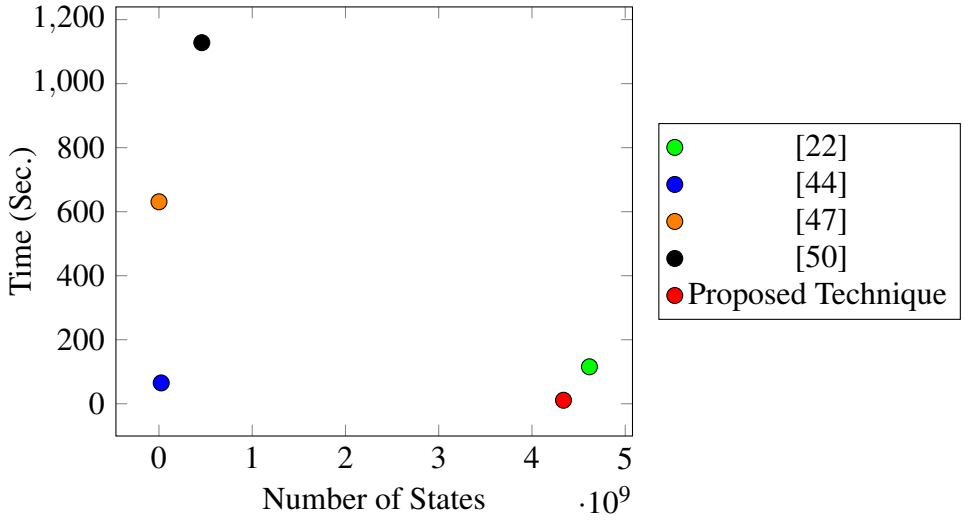

Figure 4.12: Comparison results for time and number of states

4.5.4 In-depth Verification and Scalability Analysis

This section provides a deeper and more detailed understanding of our method's verification and scalability outcomes, along with its strengths and areas that require further enhancement.

Verification:

Our approach, which incorporates the 3v-TCTLC logic, has played a crucial role in the practical verification of IoT-ad hoc applications, particularly in smart health monitoring and smart home systems. The extension of the MACMAS-interactor tool has streamlined the verification process by automatically invoking the MACMAS-TC tool.

Scalability:

We have conducted a thorough evaluation of the reliability and scalability of our systems. When implemented, our framework demonstrated its robustness in addressing safety, liveness, and reachability properties. This highlights the adaptability of our method across various scalable real-world IoT-ad hoc scenarios.

Advantages:

- The 3v-TCTLC logic provides a robust framework for addressing and reasoning about uncertainty in real-world scenarios captured by trust over commitments.
- Our extended tool enhances the verification process, ensuring a more streamlined and efficient approach.

Disadvantages:

- We have identified areas for improvement in properties that were not fully satisfied by the system, signifying potential areas for refinement in our approach.

- Model Checking Constraints: Due to the complexity of developing a direct multi-valued model checking approach and the absence of relevant existing tools, our methodology is constrained within the reduction approaches category. Reduction facilitates the reuse of established model checkers like NuSMV and MCMAS.

4.6 Summary

In this chapter, we presented a practical verification approach to verify IoT-ad hoc applications in the context of smart health monitoring and smart home system. The approach incorporates trust and commitments under uncertainty. For reasoning about uncertainty in IoT-ad hoc systems with trust on commitment, we introduced a new 3-valued logic called 3v-TCTLC, which extends the multi-valued CTL, with new modalities for trust, commitment and fulfilment taken from TCTLC logic. To verify our three-valued logic, we extended "MCMAS-interactor" tool to automatically invoke the MCMAS-TC tool to verify two-valued logic after reducing the problem of three-valued model checking to classical model checking by adding new functionalities to make the tool enabled to handle our 3v-TCTLC logic. We evaluated the reliability and scalability of our systems by implementing our framework and considering safety, liveness, and reachability properties. Our approach using 3v-TCTLC logic was crucial in effectively verifying IoT-ad hoc applications, especially in smart health and home systems. The enhanced MACMAS-interactor tool has made the verification process more efficient. Additionally, our approach demonstrated reliability and scalability supporting very large systems reaching 40 agents with $4.34e+09$ states in the smart health monitoring system and 42 agents with $1.8e+07$ states in the smart home system. This was achieved with fast verification times, taking only a few seconds (11 sec. and 5 sec. respectively). These results underscore the applicability and adaptability of our methods in real-world

IoT-ad hoc scenarios. Our contribution not only enhances the reliability and security of IoT-ad hoc systems but also provides practitioners with practical methods and tools featuring user-friendly interfaces for application in various concrete, real-world scenarios. It represents a significant step forward in ensuring combined trust and commitment within uncertain network environments.

Chapter 5

Formal Verification of Weak and Strong Trust over Commitments in Multi-Agent Systems

In the context of model checking, "weak trust" and "strong trust" are terms often used to describe the level of confidence or assurance we have in the correctness of a system or software model. In Section 5.1, We propose $T^{ws}CTLC$, a new logic able to express properties about weak and strong trust. Our proposed logic is an extended logic built on top of the CTLC logic. This approach employs weak and strong trust modalities to facilitate trust reasoning. Moreover, we introduce the semantics over the extended version of original interpreted systems formalism. Section 5.2 addresses postulates with proofs to reason about our logic. Additionally, we put forth a model checking algorithm for $T^{ws}CTLC$ that extends the CTLC symbolic algorithm, implementing it as a new open-source tool named $MCMAS-T^{ws}$ in Section 5.3. Finally, to illustrate the practical application of our proposed framework, we conduct an evaluation using a real-life case study in the e-commerce domain

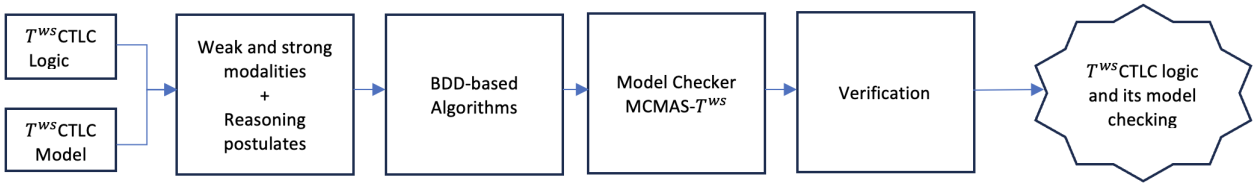

Figure 5.1: Our approach phases

in Section 5.4. Figure 5.1 illustrates our work in several phases.

5.1 T^{ws}CTLC logic

T^{ws}CTLC is a combination of branching time CTLC logic with modalities for representing and reasoning about weak and strong trust.

Definition 5.1. (Model of T^{ws}CTLC). We employ the extended version of original interpreted systems formalism defined in Chapter 3, Section 3.2 for T^{ws}CTLC model.

Definition 5.2. (Syntax of $T^{ws}CTLC$). The syntax of $T^{ws}CTLC$) is defined as follows:

$\varphi ::= \rho \mid \neg\varphi \mid \varphi \vee \varphi \mid EX\varphi \mid EG\varphi \mid E(\varphi \cup \varphi) \mid C_{i\to j}\varphi \mid Fu(C_{i\to j}\varphi) \mid T$

$T ::= T^{w}_{i\to j}\varphi \mid T^{s}_{i\to j}\varphi$

Where:

- $P \in \Phi_p$ is an atomic proposition;

- E is the existential quantifier on paths;

- The Boolean connectives $\neg$ and $\vee$ are defined in the usual way;

- X, U, and G are CTL path modal connectives standing for “next”, “until”, and “globally”;

- The modal connectives $C_{i\to j}\varphi$ and Fu stand for "commitment" and "fulfillment of commitment"; and

- T stands for "trust", respectively.

where $\rho, \vee, E, X, G, \cup, C$ and Fu are defined in Definition 2.1. A weak (strong) trust $T^w_{i\to j}\varphi (T^s_{i\to j}\varphi)$ is read as " agent i weakly (strongly) trusts the agent j that φ.

A path π is an infinite sequence of reachable global states $(s_0, s_1, ...)$ in S such that $\forall i \geq 0, (s_i, s_{i+1}) \in R_t$. A state s' is reachable from a state s iff $\exists \pi$ such that $s = \pi(0)$ and $s' = \pi(k), k \geq 0$. This reachability is defined by $s \to s'$ [18].

Definition 5.3. (Semantics of T^{ws}CTLC) Given the model M, the satisfaction of a T^{ws}CTLC formula φ in a global state s, denoted by $(M,s) \models \varphi$ is recursively defined as follows:

- $(M,s) \models p$ iff $p \in V(s)$;
- $(M,s) \models \neg\varphi$ iff $(M,s) \not\models \varphi$;
- $(M,s) \models \varphi \vee \psi$ iff $(M,s) \models \varphi$ or $(M,s) \models \psi$;
- $(M,s) \models EX\varphi$ iff there exists a path π starting at s such that $(M, \pi(1)) \models \varphi$
- $(M,s) \models E(\varphi \cup \varphi)$ iff there exists a path π starting at s such that for some $k \geq 0, (M, \pi(k)) \models \psi$ and $(M, \pi(j)) \models \varphi$ for all $0 \leq j < k$;
- $(M,s) \models EG\varphi$ iff there exists a path π starting at s such that $(M, \pi(k)) \models \varphi$ for all $k \geq 0$;
- $(M,s) \models C_{i\to j}\varphi$ iff for all global states $s' \in S$ such that $s_{\sim_{i\to j}} s'$, we have $(M,s') \models \varphi$;
- $(M,s) \models Fu(C_{i\to j}\varphi)$ iff $\exists s' \in S$ such that $s'_{\sim_{i\to j}} s$ and $(M,s') \models C_{i\to j}\varphi$;
- $(M,s) \models T^w_{i\to j}\varphi$ iff $(M,s) \models C_{j\to i}\varphi$ and $\exists P_{I\to s}$ such that $\exists s' \in P_{I\to s}$, we have $(M,s') \models Fu(C_{j\to i}\varphi)$;
- $(M,s) \models T^s_{i\to j}\varphi$ iff $(M,s) \models C_{j\to i}\varphi$ and $\forall P_{I\to s}$ such that $\exists s' \in P_{I\to s}$, we have $(M,s') \models Fu(C_{j\to i}\varphi)$;

The semantics of T^{ws}CTLC state formula is defined as semantics of CTLC in Definition 2.2, in addition to the modal weak and strong trust operators. The state formula $T^{w}_{i\to j}\varphi$ is satisfied in the model M at s iff s satisfies the commitment $C_{j\to i}\varphi$ and at least in one path from initial sate to state s, there exist a state s' satisfying the fulfillment of the commitment, and from which s is reachable from initial state.The idea behind this semantics is to say that a weak trust is achieved when we reach a reachable state from the fulfillment state along a path from initial state, in which the commitment is active. The semantics of the strong trust $T^{s}_{i\to j}\varphi$ is similar, but the focus is on checking the all paths from initial state to state s if there exists a state s' satisfying the fulfillment of the commitment.

5.2 Reasoning Postulates

To illustrate the relation between commitments, weak and strong trust in MASs using semantics and model checking, we defined several reasoning postulates in the T^{ws}CTLC logic. NetBill payment protocol, which is designed for the encrypted software goods trades over the Internet [94], can be used as an instance for the reasoning postulate. This protocol is defined in Chapter 3, Section 3.5.1.

The proposed postulates:

We note that when the rule holds for strong trust, it also holds for the weak cases.

P1: Commitment to the trusted content is necessary.

$T^{s}_{i\to j}\varphi \to C_{j\to i}\varphi$

Meaning: When the strong trust holds, a commitment with the same content comes into being.

Proof: the proof is straightforward from the semantics of $T^{s}_{i\to j}\varphi$ which indicates that the current state satisfies $C_{j\to i}\varphi$.

Example: By applying this postulate, $T^s_{Cus \to Mer}\, DeliverGoods \to C_{Mer \to Cus}\, DeliverGoods$, once the customer strongly trusts the merchant to deliver the goods, the merchant is committed to bringing it about.

P2: Fulfillment of the trusted content.

$T^s_{j \to i}\varphi \to EF(Fu(C_{i \to j}\varphi))$

Meaning: Strong trust to a content yields fulfillment of the commitment to that content.

Proof: From the semantics of $T^s_{j \to i}\varphi$, we obtain $(M,s) \models C_{i \to j}\varphi$. Moreover, from the semantics of $C_{i \to j}\varphi$, for all global states $s' \in S$ such that $s_{\sim_{i \to j}} s'$, we have $(M,s') \models \varphi$ and the commitment is fulfilled when its content holds in the accessible state. Consequently, $(M,s') \models Fu(C_{i \to j}\varphi)$, so the postulate.

Example: Assuming that the merchant strongly trusts to the customer to send the the payment, then customer will fulfill his commitment of sending the payment. Formally: $T^s_{Mer \to Cus}\, SendPayment \to EF(Fu(C_{Cus \to Mer}\, SendPayment))$.

P3: The content's fulfillment is derived from the trusted content.

$T^s_{i \to j}\varphi_1, \varphi_1 \vdash \varphi_2$ infer $EF[Fu(C_{j \to i}\varphi_2)]$

Meaning: Commitment about φ_2 is fulfilled if strongly trust to bring about the content from which φ_2 is derived.

Proof: The proof directly follows from the consequence of P2 and $\varphi_1 \vdash \varphi_2$.

Example: Suppose that the customer strongly trusts the merchant for delivering goods, then eventually warranty paperwork will be delivered as well, because delivering good automatically triggers the process of delivering the warranty. Formally expressed as follow: $T^s_{Cus \to Mer}\ DeliverGoods$, $DeliverGoods \vdash DeliverWarranty$ infer $Fu(C_{Mer \to Cus}\, DeliverWarranty)$

P4: Trust to the content that is fulfilled and recommitted previously.

$Fu(C_{i\to j}\varphi) \to EF[(C_{i\to j}\varphi) \to T^{w}_{j\to i}\varphi]$

Meaning: Weak trust to the commitment about φ holds once the commitment about φ is fulfilled and then recommitted in previous.

Proof: Assume that $(M,s) \models Fu(C_{i\to j}\varphi)$ and $(M,s) \models EF(C_{i\to j}\varphi)$ then we obtain $(M,s') \models C_{i\to j}\varphi$. lets s' be the state s_1 which means s' is in the future of s. consequently, $\exists P_{I\to s'}$ such that $\exists s'' = s$ and $s'' \models Fu(C_{i\to j}\varphi)$. Thus, from the semantics, trust will be achieved and we are done.

This postulate is valid only for weak trust. This is because the semantics of $T^{w}_{j\to i}\varphi$ requires the existence of a path that satisfies $Fu(C_{i\to j}\varphi)$, which is not the case for strong trust.

Example: Suppose that the merchant delivers the goods, then in a possible future of the delivery, the costumer will weakly trust the merchant regarding the delivery of the ordered goods if there exists a commitment to the delivery, which means: $Fu(C_{Mer\to Cus}\ DeliverGoods) \to EF[(C_{Mer\to Cus}\ DeliverGoods) \to T^{w}_{Cus\to Mer}\ DeliverGoods]$

P5: No active commitment and trust to the content which is being held.

$\varphi \to \neg C_{i\to j}\varphi \wedge \neg T^{s}_{j\to i}\varphi$

Meaning: When φ holds, the commitment and the trust to bring about φ is discharged.

Proof: The rule comes directly from the semantics of $C_{i\to j}\varphi$ and $T^{s}_{i\to j}\varphi$.

P6: No active trust to the content once fulfilled.

$Fu(C_{i\to j}\varphi) \to \neg T^{s}_{i\to j}\varphi$

Meaning: When a commitment is fulfilled, strong trust is no longer active.

Proof: From the semantics of $Fu(C_{i\to j}\varphi)$, there exists a state $s' \in S$ such that $s'_{\sim_{i\to j}}s$ and $(M,s) \models \varphi$. According to the consequence of P5, we are done

P7: No commitment and trust to the content that does not hold globally.

$AG\neg\varphi \rightarrow \neg C_{i\rightarrow j}\varphi \wedge \neg T^{s}_{j\rightarrow i}\varphi$

Meaning: If the content of the commitment and strong trust does not hold, then the commitment and strong trust never hold.

Proof: The proof is derived directly from the semantics of commitment, strong trust and the fact that all accessible states are reachable $(i.e., (s,s') \in R_t)$.

Example: According to the NetBill protocol, if goods are never delivered, then the merchant cannot commit to delivery, and customer cannot strongly trust to bring that goods. $AG\neg\ DeliverGoods \rightarrow \neg C_{Mer\rightarrow Cus}\ DeliverGoods \wedge \neg T^{s}_{j\rightarrow i}\ DeliverGoods$

P8: Content holds once its commitment is fulfilled.

$Fu(C_{i\rightarrow j}\varphi) \rightarrow \varphi$

Meaning: When a commitment is fulfilled, its content holds.

Proof: the proof is straightforward from semantics of fulfillment.

P9: No longer active commitment once fulfilled.

$Fu(C_{i\rightarrow j}\varphi) \rightarrow \neg C_{i\rightarrow j}\varphi$

Meaning: The commitment to φ is discharged once fulfilled.

Proof: the proof directs from semantics of fulfillment.

P10: The combination of commitment contents once are committed separately.

$C_{j\rightarrow i}\varphi_1 \wedge C_{j\rightarrow i}\varphi_2 \rightarrow C_{j\rightarrow i}(\varphi_1 \wedge \varphi_2)$

Meaning: The conjunction of two commitments yields a combined commitment.

Proof: The proof comes from the semantics of commitment.

P11: The combination of trust contents once are trusted separately.

$T^s_{j\to i}\varphi_1 \wedge T^s_{j\to i}\varphi_2 \to T^s_{j\to i}(\varphi_1 \wedge \varphi_2)$

Meaning: The conjunction of two strong trusts yields a combined trust.

Proof: The rule is derived from the semantics of strong trust.

P12: Commitment to combination of trust contents once are trusted separately.

$T^s_{j\to i}\varphi_1 \wedge T^s_{j\to i}\varphi_2 \to C_{j\to i}(\varphi_1 \wedge \varphi_2)$

Meaning: The conjunction of two trusts yields a combined commitment.

Proof: The rule is direct from the semantics of strong trust.

P13: No commitment to the false content.

$\neg C_{i\to j}\bot$

Meaning: A commitment to false cannot be held.

Proof: The proof is straightforward from the semantics of commitment.

P14: No trust to the false content.

$\neg T^s_{j\to i}\bot$

Meaning: A trust to false cannot be held.

Proof: The proof comes from semantics of strong trust.

P15: No fulfillment for the false content.

$\neg Fu(C_{i\to j}\bot)$

Meaning:A commitment to false cannot be fulfilled.

Proof: The rule follows from semantics of fulfillment since there is no accessible state s that satisfies $\bot$.

P16: No commitment to the negation of a content that is committed.

$C_{i\to j}\varphi \to \neg C_{i\to j}\neg\varphi$

Meaning: When a commitment to φ holds, the commitment to negation of φ cannot be held.

Proof: The rule is directly follows from the fact that in semantics of $C_{i\to j}\varphi$, all accessible sates satisfy φ, and a state satisfy φ cannot satisfy $\neg\varphi$.

P17: No trust to the negation of content that is trusted.

$T^s_{i\to j}\varphi \to \neg T^s_{i\to j}\neg\varphi$

Meaning: When the strong trust holds, then there is no possibility of trusting the negation of the same content.

Proof: The rule is derived from the semantics of strong trust.

P18: No commitment to the negation of content that is trusted.

$T^s_{j\to i}\varphi \to \neg C_{i\to j}\neg\varphi$

Meaning: When the strong trust to a content holds, there is no possibility to commit to the negation of the same content.

Proof: The proof comes from the consequence of P1.

P19: Commitment to one part of the conjunction once all parts are committed.

$C_{i\to j}(\varphi_1 \wedge \varphi_2) \to C_{j\to i}\varphi_1$

Meaning: If commitment to a conjunction holds, then the commitment to each part of the conjunction also holds.

Proof: The rule follows from the fact that in semantics of commitment all accessible sates that satisfy $\varphi_1 \wedge \varphi_2$ also satisfy φ_1 .

P20: Trust to one part of the conjunction once all parts are trusted.

$T^s_{i\to j}(\varphi_1 \wedge \varphi_2) \to T^s_{j\to i}\varphi_1$

Meaning: If trust to a conjunction holds, then the trust to each part of the conjunction also holds.

Proof: The rule straightforwards from the semantics of the strong trust.

P21: Commitment to one part of the conjunction once all parts are trusted.

$T^s_{i\to j}(\varphi_1 \wedge \varphi_2) \to C_{j\to i}\varphi_1$

Meaning: If trust to a conjunction holds, then the commitment to each part of the conjunction also holds.

Proof: The proof is direct from the semantics of $T^s_{i\to j}\varphi$.

5.3 Model Checking T^{ws}CTLC

Model checking involves automatically determining whether a formula is fulfilled within a specified model. This section introduces a streamlined algorithm designed to tackle the T^{ws}CTLC model checking issue efficiently. We start by introducing the primary algorithm (Algorithm 5.1), which expands upon the conventional symbolic model checking algorithm for CTL. Within this algorithm, we directly invoke established procedures for the CTL modalities to compute the set of states that satisfy the respective modalities.

Algorithm 5.1 operates as follows: Initially, it takes the model M and the T^{ws}CTLC formula φ as inputs, and it returns the set $[[\varphi]]$ comprising states that satisfy φ in M. The algorithm navigates through the structure of φ recursively, utilizing a set of Boolean operations applied to sets to construct the set $[[\varphi]]$. Lines 1 to 6 invoke standard procedures used in CTL to determine the set of states satisfying regular CTL formulas. Lines 7 and 8 invoke the commitment and its fulfilment procedures and lines 9 and 10 call our procedures

designed to compute the set of states satisfying weak and strong trust formulas.

Algorithm 5.1 :$SMC(\varphi, M)$: the set $[[\varphi]]$ satisfying the $T^{ws}CTLC$ formula φ

1: φ is an atomic formula: return $V(\varphi)$;
2: φ is $\neg\varphi_1$: return $S - SMC(\varphi_1, M)$;
3: φ is $\varphi_1 \vee \varphi_2$: return $SMC(\varphi_1, M) \cup SMC(\varphi_2, M)$;
4: φ is $EX\varphi_1$: return $SMC_{EX}(\varphi_1, M)$;
5: φ is $E(\varphi_1 \cup \varphi_2)$: return $SMC_{EU}(\varphi_1, \varphi_2, M)$;
6: φ is $EG\varphi_1$: return $SMC_{EG}(\varphi_1, M)$;
7: φ is $C_{i\to j}\varphi_1$: return $SMC_c(i, j, \varphi_1, M)$;
8: φ is $Fu(C_{i\to j}\varphi_1)$: return $SMC_{fu}(i, j, \varphi_1, M)$;
9: φ is $T^w_{i\to j}\varphi_1$: return $SMC_{t^w}(i, j, \varphi_1, M)$;
10: φ is $T^s_{i\to j}\varphi_1$: return $SMC_{t^s}(i, j, \varphi_1, M)$;

5.3.1 BDD-based Algorithms for the Weak and Strong Trust

In this section, we present the model checking algorithms for both the T^w and T^s operators. These algorithms, given a $T^{ws}CTLC$ formula φ and a $T^{ws}CTLC$ model M within a interpreted system, calculate the set of states in M where φ is held. Algorithm 5.2 describes the procedure $SMC_{t^w}(i, j, \varphi, M)$. This procedure returns the set of states in which the weak trust formula holds. It begins by determining the set X_1, comprising states that satisfy the commitment $C_{j\to i}\varphi$. Subsequently, it calculates the set X_2, which contains states satisfying the formula $Fu(C_{j\to i}\varphi)$ using the algorithm 3.3 and algorithm 3.4 in Chapter 3. The procedure then assembles the set X_3, consisting of states preceding those satisfying $C_{j\to i}\varphi$ by calling procedure $Past(X)$ defined in 3.6 . Finally, the algorithm identifies and returns the states in X_4, which in at least one path from initial state, both adhere to $C_{j\to i}\varphi$ (i.e., in X_1) and are reachable from past states satisfying $Fu(C_{j\to i}\varphi)$ (i.e., in $X_2 \cap X_3$). In Algorithm 5.3, to compute the formula T^s, we follow the same steps as in Algorithm 5.2, except lines 4 which checks whether all paths from initial state to state s satisfying $Fu(C_{j\to i}\varphi)$. Indeed, this is based on our proposed semantics of strong trust where the set of global states satisfying the formula T^s in a given model M is computed by verifying all paths to check if there exists a state satisfying the fulfillment of the commitment.

Algorithm 5.2 : $SMC_{t^w}(i,j,\varphi,M)$: the set $[[T^w_{i\to j}\varphi]]$

1: $X_1 \leftarrow SMC_c(j,i,\varphi,M)$;
2: $X_2 \leftarrow SMC_{fu}(j,i,\varphi,M)$;
3: $X_3 \leftarrow Past(X_1)$;
4: $X_4 \leftarrow \{s \in X_1 \mid \exists P_{I\to s}, \exists s' \in X_2 \cap X_3 \; s.t. \; (s',s) \in R_t\}$
5: return X_4;

Algorithm 5.3 : $SMC_{t^s}(i,j,\varphi,M)$: the set $[[T^s_{i\to j}\varphi]]$

1: $X_1 \leftarrow SMC_c(j,i,\varphi,M)$;
2: $X_2 \leftarrow SMC_{fu}(j,i,\varphi,M)$;
3: $X_3 \leftarrow Past(X_1)$;
4: $X_4 \leftarrow \{s \in X_1 \mid \forall P_{I\to s}, \exists s' \in X_2 \cap X_3 \; s.t. \; (s',s) \in R_t\}$
5: return X_4;

5.4 Implementation and Experiments

Our algorithms are integrated with the MCMAS model checker, which yield $MCMAS-T^{ws}$, the model checker for MASs with weak and strong trust. Additionally, we aim to conduct model checking on various properties of MASs, focusing on weak and strong trust relationships that involve multiple interacting agents.

In the following section, we consider the NetBill protocol as a demonstrative application example introduced in Chapter 3, Section 3.5.1. This serves to illustrate how our model checking technique can effectively be employed on a e-commerce platform to verify trust transactions against specific quantified temporal trust conditions.

5.4.1 Assessment of Performance

We use our formal model M associated to the interpreted systems introduced earlier in Chapter 3, Section 3.2 to formally model the NetBill protocol. In this scenario, the weak and strong trust relationships between the participating parties express the system requirements that regulate the interacting agents. Such requirements are specified using our logic of trust $T^{ws}CTLC$. Consequently, the trust relationships are instantiated, aiding prospective agents in determining the extent to which they should trust other agents.

The following protocol properties are expressed within the $T^{ws}CTLC$ logic to verify the accuracy of the process model.

- $\varphi_1 = T^s_{Mer \to Cus}\, SendPayment \to EF(Fu(C_{Cus \to Mer}\, SendPayment))$
- $\varphi_2 = Fu(C_{Mer \to Cus}\ \ DeliverGoods) \to EF[(C_{Mer \to Cus}\ \ DeliverGoods) \to T^s_{Cus \to Mer}\, DeliverGoods]$
- $\varphi_3 = AG\neg\, DeliverGoods \to \neg C_{Mer \to Cus}\, DeliverGoods \wedge \neg T^s_{j \to i}\, DeliverGoods$

These formulae express reachability and liveness properties for both weak and strong trust. For example, the formula φ_1 encodes the fact that there exists a state reachable from the initial state, such that the merchant strongly trusts to the customer to send the the payment, then customer will fulfill his commitment of sending the payment. The formula φ_2 states that the merchant delivers the goods, then in a possible future of the delivery, the costumer will weakly trust the merchant regarding the delivery of the ordered goods if there exists a commitment to delivery. Moreover, in terms of liveness property, φ_3 states that in all computation paths, it is always the case that if goods are never delivered, then the merchant cannot commit to delivery, and customer cannot strongly trust to bring that goods.

5.4.2 Experimental Results

To evaluate the scalability of our technique and implementation, we measured the processing time for model checking and the BDD memory usage required for successful verification on a machine equipped with a Dual-Core CPU T4500 at 2.30 GHz running an operating system with 64 bits. Our experiments involved varying the number of agents from 2 to 22 to explore different scalability levels, aiming to observe significant results in handling complex problems. The experiments demonstrated satisfaction of all tested formulae. Table 5.1 documents the verification outcomes alongside the count of agents

and reachable states in the constructed model. It is evident that the number of reachable states corresponds to the exponential growth of the state space with the increasing number of agents of agents, the memory usage increases polynomially. In fact, we cannot offer a comprehensive comparison of these results with other implementations since, as far as we know, there is no model checker tool available for verifying properties of weak and strong trust, which is the focus of our work.

Table 5.1: The NetBill protocol verification results

Exp.#	Agents#	States#	Memory (MB)	Time (Sec.)
1	2	83	12.3	0.09
2	4	218	13.78	0.92
3	6	6723	15.42	1.17
4	8	327021	19.25	2.87
5	10	1.22982e+07	30.62	4.52
6	12	1.25958e+08	42.54	7.09
7	14	3.70736e+08	75.23	9.2
8	16	2.87521e+09	110.22	45.21
9	18	4.31257e+10	159.29	82.65
10	20	5.51692e+11	179.92	120.01
11	22	4.22543e+12	185.53	152.92

5.5 Summary

To capture the interaction between social commitments, weak and strong trust from the semantics point of view, we introduced T^{ws}CTLC, a new logic for trust (weak and strong), communicative commitments and their interactions. We presented a new semantics of T^{ws}CTLC based on social accessibility relation. We used original interpreted systems and its extended versions to describe interaction among interacting parties. Moreover, we proposed some postulates with proofs to reason about our logic. Moreover, we presented a model checking algorithm for T^{ws}CTLC that extends the CTLC symbolic algorithm and its

implementation that results in a new open source tool called $MCMAS-T^{ws}$. We evaluated our approach by means of a real-life case study in the e-commerce domain in order to explain our proposed framework in a practical setting.

Chapter 6

Conclusion

6.1 Summary and Discussion

The goal of these theses is to build greater trust and commitment in multi-agent systems, while also improving the dependability and efficiency of IoT within intelligent environments. By doing so, we provide practical tools and methodologies that are specifically tailored for situations characterized by uncertainty.The framework consists of three primary components. First, in Chapter 3, we introduced TCTLC, a new logic that captures the combination of two key concepts in MASs, namely trust and agents' commitments. We addressed two main issues: reasoning postulates and model checking. We presented the semantics for trust, based on accessibility relation. We used the original interpreted systems and its extended versions to describe the interaction between trust and commitment among interacting parties. Moreover, we supported our proposed postulates by proofs. We presented new symbolic algorithms and demonstrated that (1) the time complexity of model checking TCTLC in explicit models is P-complete with respect to the size of the model and the length of the formula; (2) the complexity of the same problem for concurrent programs is PSPACE-complete, considering the size of the program's

components. As a result, our model checking algorithm has the same complexity as model checking CTL concerning both explicit models and concurrent programs. We implemented the proposed algorithms as part of our MCMAS-TC tool, verified the NetBill protocol, and reported the results of the experiments. We compared our results with relevant benchmarks. It also shows our results for better comparison.

The second component of our framework in Chapter 4 focused on a practical verification approach to verify IoT-ad hoc applications in the context of smart health monitoring and smart home system. The approach incorporates trust and commitments under uncertainty. For reasoning about uncertainty in IoT-ad hoc systems with trust on commitment, we introduced a new 3-valued logic called 3v-TCTLC, which extends the multi-valued CTL, with new modalities for trust, commitment and fulfilment taken from TCTLC logic. To verify our three-valued logic, we extended "MACMAS-interactor" tool to automatically invokes MACMAS-TC tool to verify two-valued logic after reducing the the problem of three-valued model checking to classical model checking by adding new functionalities to make the tool enable to handle our 3v-TCTLC logic. We evaluated the reliability and scalability of our systems by implementing our framework and considering safety, liveness, and reachability properties.

In the third part of the framework in Chapter 5, we presented a new logic, $T^{ws}CTLC$, specifically crafted to encapsulate properties pertaining to both weak and strong trust. We defined the semantics over an expanded interpretation of the original system's formalism. We offered postulates along with their respective proofs to validate our logic. Additionally, we introduced a model checking algorithm for $T^{ws}CTLC$, expanding upon the CTLC symbolic algorithm, and implemented it to create a new open-source tool named $MCMAS-T^{ws}$. We assessed our approach through a NetBill protocol to illustrate our proposed framework in a practical context.

6.2 Directions for Future Work

The research contributions of this thesis filled out some of the important gaps in the current literature. However, considering the rapid advance of technology, there are still challenging problems to be explored. A summary of future research directions are provided as follow:

- This work opens the door for an extension by investigating group trust and commitments along with their interactions.The logical framework presented in this research models trust using social commitments among two agents. The proposed logic does not support group trust and commitments (one-to-group, group-to-one, and group-to-group). Thus, in our future work, we aim to enrich the area of trust and communicative social commitments from new perspectives. In particular, we aim to introduce a new consistent, formal and computationally grounded semantics to reason about group trust over communicating social commitments.

- The logical framework presented in this thesis models unconditional trust and social commitment. TCTLC logic does not support conditional trust. Expressing conditional trust requires an extension of TCTLC. Thus, in our future work we aim to distinguish the two languages by extending TCTLC. As our main objective in this proposal is the verification of temporal trust and commitment, we will define how this logical relationship will be used to model checking procedure of conditional trust and commitment.

- Since our research solely focused on uncertainty within the states (i.e., in properties), we intend to extend our investigation to verify IoT-ad hoc systems with trust over commitment in the presence of uncertainty in both the states and the transitions.

- We intend to expand our semantics to encompass arbitrary lattices that can contain more than three values. This expansion will enable us to reason about both

uncertainty and inconsistency in IoT-ad hoc systems.

- We also plan to Expand our method from discrete-time to continuous-time modeling, addressing trust and commitment under uncertainty. This entails exploring continuous time logic and formalizing mobility, enhancing our framework's understanding of mobile node behavior.

Bibliography

[1] Faisal Al-Saqqar, Jamal Bentahar, and Khalid Sultan. On the soundness, completeness and applicability of the logic of knowledge and communicative commitments in multi-agent systems. *Expert Systems with Applications*, 43:223–236, 2016.

[2] Faisal Al-Saqqar, Jamal Bentahar, Khalid Sultan, and Mohamed El Menshawy. On the interaction between knowledge and social commitments in multi-agent systems. *Applied Intelligence*, 41(1):235–259, 2014.

[3] Faisal Al-Saqqar, Jamal Bentahar, Khalid Sultan, Wei Wan, and Ehsan Khosrowshahi Asl. Model checking temporal knowledge and commitments in multi-agent systems using reduction. *Simulation Modelling Practice and Theory*, 51:45–68, 2015.

[4] Waleed Alnumay, Uttam Ghosh, and Pushpita Chatterjee. A trust-based predictive model for mobile ad hoc network in internet of things. *Sensors*, 19(6):1467, 2019.

[5] Hind Alotaibi and Hussein Zedan. Runtime verification of safety properties in multi-agents systems. In *2010 10th International Conference on Intelligent Systems Design and Applications*, pages 356–362. IEEE, 2010.

[6] Ghalya Alwhishi, Jamal Bentahar, and Nagat Drawel. Reasoning about uncertainty over IoT systems. In *2022 International Wireless Communications and Mobile*

Computing, IWCMC 2022, Dubrovnik, Croatia, May 30 - June 3, 2022, pages 306–311. IEEE, 2022.

[7] Ghalya Alwhishi, Jamal Bentahar, and Ahmed Elwhishi. Three-valued model checking smart contract systems with trust under uncertainty. In Irfan Awan, Muhammad Younas, Jamal Bentahar, and Salima Benbernou, editors, *The International Conference on Deep Learning, Big Data and Blockchain (DBB 2022), Rome, Italy, 22-24 August 2022*, volume 541 of *Lecture Notes in Networks and Systems*, pages 119–133. Springer, 2022.

[8] Ghalya Alwhishi, Jamal Bentahar, and Ahmed Elwhishi. Three-valued model checking smart contract systems with trust under uncertainty. In *The International Conference on Deep Learning, Big Data and Blockchain (DBB 2022)*, pages 119–133. Springer, 2022.

[9] Ghalya Alwhishi, Jamal Bentahar, and Ahmed Elwhishi. Verifying timed commitment specifications for IoT-cloud systems with uncertainty. In Muhammad Younas, Irfan Awan, and Wenny Rahayu, editors, *9th International Conference on Future Internet of Things and Cloud, FiCloud 2022, Rome, Italy, August 22-24, 2022*, pages 173–180. IEEE, 2022.

[10] Ghalya Alwhishi, Jamal Bentahar, and Ahmed Elwhishi. Verifying timed commitment specifications for iot-cloud systems with uncertainty. In *2022 9th International Conference on Future Internet of Things and Cloud (FiCloud)*, pages 173–180. IEEE, 2022.

[11] Ghalya Alwhishi, Jamal Bentahar, and Ahmed Elwhishi. Multi-valued model checking a smart glucose monitoring system with trust. In *International Wireless*

Communications and Mobile Computing IWCMC, June 19 - 23, Marrakesh, Morocco, pages xx–xx. IEEE, 2023.

[12] Ghalya Alwhishi, Jamal Bentahar, Ahmed Elwhishi, Witold Pedrycz, and Nagat Drawel. Multi-valued model checking IoT and intelligent systems with commitment protocols in multi-source data environments. *Available at SSRN 4421677*, 2023.

[13] Mohammed S Alzaidi, Piyush Kumar Shukla, V Sangeetha, Karuna Nidhi Pandagre, Vinodh Kumar Minchula, Sachin Sharma, Arfat Ahmad Khan, and V Prashanth. Applying machine learning enabled myriad fragment empirical modes in 5g communications to detect profile injection attacks. *Wireless Networks*, pages 1–14, 2023.

[14] Samiha Ayed, Amal Hbaieb, and Lamia Chaari. Blockchain and trust-based clustering scheme for the IoV. *Ad Hoc Networks*, 142:103093, 2023.

[15] Narges Baharloo, Jamal Bentahar, Ghalya Alwhishi, Nagat Drawel, and Witold Pedrycz. Verifying trust over iot-ad hoc network-based applications under uncertainty. *Ad Hoc Networks*, 154:103380, 2024.

[16] Narges Baharloo, Jamal Bentahar, Nagat Drawel, and Witold Pedrycz. Model checking combined trust and commitments in multi-agent systems. *Expert Systems with Applications*, 243:122856, 2024.

[17] Najwa Abu Bakar and Ali Selamat. Runtime verification of multi-agent systems interaction quality. In *Intelligent Information and Database Systems: 5th Asian Conference, ACIIDS 2013, Kuala Lumpur, Malaysia, March 18-20, 2013, Proceedings, Part I 5*, pages 435–444. Springer, 2013.

[18] Matteo Baldoni, Cristina Baroglio, and Elisa Marengo. Behavior-oriented commitment-based protocols. In *ECAI*, volume 215, pages 137–142, 2010.

[19] Andreas Bauer, Martin Leucker, and Christian Schallhart. Runtime verification for ltl and tltl. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 20(4):1–64, 2011.

[20] Paolo Bellavista, Giuseppe Cardone, Antonio Corradi, and Luca Foschini. Convergence of MANET and WSN in IoT urban scenarios. *IEEE Sensors Journal*, 13(10):3558–3567, 2013.

[21] Nuel D Belnap Jr. A useful four-valued logic. In *Modern uses of multiple-valued logic*, pages 5–37. Springer, 1977.

[22] Jamal Bentahar, Narges Baharloo, Nagat Drawel, and Witold Pedrycz. Model checking combined trust and commitments in multi-agent systems. *Available at SSRN 4370421*, 2023.

[23] Jamal Bentahar, Nagat Drawel, and Abdeladim Sadiki. Quantitative group trust: A two-stage verification approach. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, pages 100–108, 2022.

[24] Jamal Bentahar, Mohamed El-Menshawy, Hongyang Qu, and Rachida Dssouli. Communicative commitments: Model checking and complexity analysis. *Knowledge-Based Systems*, 35:21–34, 2012.

[25] Jamal Bentahar, J-J Ch Meyer, and Wie Wan. Model checking agent communication. In *Specification and verification of multi-agent systems*, pages 67–102. Springer, 2010.

[26] Simona Bernardi, Ugo Gentile, Stefano Marrone, José Merseguer, and Roberto Nardone. Security modelling and formal verification of survivability properties: Application to cyber–physical systems. *Journal of Systems and Software*, 171:110746, 2021.

[27] Anna Bernasconi, Claudio Menghi, Paola Spoletini, Lenore D Zuck, and Carlo Ghezzi. From model checking to a temporal proof for partial models. In *Software Engineering and Formal Methods: 15th International Conference, SEFM 2017, Trento, Italy, September 4–8, 2017, Proceedings 15*, pages 54–69. Springer, 2017.

[28] Sami Bettayeb, Mohamed-Lamine Messai, and Sofiane Mounine Hemam. A robust and efficient vector-based key management scheme for IoT networks. *Ad Hoc Networks*, 149:103250, 2023.

[29] Rasa Bruzgiene, Lina Narbutaite, and Tomas Adomkus. MANET network in internet of things system. In Jesus Hamilton Ortiz and Alvaro Pachon de la Cruz, editors, *Ad Hoc Networks*, chapter 5, pages 89–114. IntechOpen, 2017.

[30] Randal E Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, 100(8):677–691, 1986.

[31] Deniz Canturk, Pinar Karagoz, Sang-Wook Kim, and Ismail Hakki Toroslu. Trust-aware location recommendation in location-based social networks: A graph-based approach. *Expert Systems with Applications*, 213:119048, 2023.

[32] Marsha Chechik, Benet Devereux, Steve Easterbrook, and Arie Gurfinkel. Multi-valued symbolic model-checking. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 12(4):371–408, 2003.

[33] Ray Chen and Jia Guo. Hierarchical trust management of community of interest groups in mobile ad hoc networks. *Ad Hoc Networks*, 33:154–167, 2015.

[34] Ray Chen, Jia Guo, Fenye Bao, and Jin-Hee Cho. Trust management in mobile ad hoc networks for bias minimization and application performance maximization. *Ad Hoc Networks*, 19:59–74, 2014.

[35] Zhimin Chen, Yi Jiang, and Yao Zhao. A collaborative filtering recommendation algorithm based on user interest change and trust evaluation. *International Journal of Digital Content Technology and its Applications*, 4(9):106–113, 2010.

[36] Alessandro Cimatti, Edmund Clarke, Fausto Giunchiglia, and Marco Roveri. Nusmv: a new symbolic model checker. *International journal on software tools for technology transfer*, 2(4):410–425, 2000.

[37] Edmund M Clarke, E Allen Emerson, and Joseph Sifakis. Model checking: algorithmic verification and debugging. *Communications of the ACM*, 52(11):74–84, 2009.

[38] EM Clarke, Orna Grumberg, and Doron A Peled. *Model Checking*. The MIT Press, 1999.

[39] F De Renesse and AH Aghvami. Formal verification of ad-hoc routing protocols using spin model checker. In *Proceedings of the 12th IEEE Mediterranean Electrotechnical Conference (IEEE Cat. No. 04CH37521)*, volume 3, pages 1177–1182. IEEE, 2004.

[40] Nirmit Desai, Amit K Chopra, and Munindar P Singh. Amoeba: A methodology for modeling and evolving cross-organizational business processes. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 19(2):1–45, 2009.

[41] Nagat Drawel, Jamal Bentahar, Amine Laarej, and Gaith Rjoub. Formalizing group and propagated trust in multi-agent systems. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 60–66, 2021.

[42] Nagat Drawel, Jamal Bentahar, Amine Laarej, and Gaith Rjoub. Formal verification of group and propagated trust in multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 36(1):19, 2022.

[43] Nagat Drawel, Amine Laarej, Jamal Bentahar, and Mohamed El Menshawy. Transformation-based model checking temporal trust in multi-agent systems. *Journal of Systems and Software*, page 111383, 2022.

[44] Nagat Drawel, Hongyang Qu, Jamal Bentahar, and Elhadi Shakshuki. Specification and automatic verification of trust-based multi-agent systems. *Future Generation Computer Systems*, 107:1047–1060, 2020.

[45] Karim El Defrawy and Gene Tsudik. Alarm: Anonymous location-aided routing in suspicious manets. *IEEE Transactions on Mobile Computing*, 10(9):1345–1358, 2010.

[46] Warda El Kholy, Mohamed El-Menshawy, Jamal Bentahar, Mounia Elqortobi, Amine Laarej, and Rachida Dssouli. Model checking intelligent avionics systems for test cases generation using multi-agent systems. *Expert Systems with Applications*, 156:113458, 2020.

[47] Mohamed El Menshawy, Jamal Bentahar, Warda El Kholy, and Rachida Dssouli. Reducing model checking commitments for agent communication to model checking arctl and gctl*. *Autonomous agents and multi-agent systems*, 27(3):375–418, 2013.

[48] Mohamed El-Menshawy, Jamal Bentahar, Warda El Kholy, and Rachida Dssouli. Verifying conformance of multi-agent commitment-based protocols. *Expert Systems with Applications*, 40(1):122–138, 2013.

[49] Mohamed El-Menshawy, Jamal Bentahar, Warda El Kholy, and Amine Laarej. Model checking real-time conditional commitment logic using transformation. *J. Syst. Softw.*, 138:189–205, 2018.

[50] Mohamed El-Menshawy, Jamal Bentahar, Hongyang Qu, and Rachida Dssouli. On the verification of social commitments and time. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 483–490. International Foundation for Autonomous Agents and Multiagent Systems, 2011.

[51] E Allen Emerson. Temporal and modal logic. In *Formal Models and Semantics*, pages 995–1072. Elsevier, 1990.

[52] Laurent Eschenauer, Virgil D Gligor, and John Baras. On trust establishment in mobile ad-hoc networks. In *Security Protocols: 10th International Workshop, Cambridge, UK, April 17-19, 2002. Revised Papers 10*, pages 47–66. Springer, 2004.

[53] Ronald Fagin, Joseph Y Halpern, Yoram Moses, and Moshe Y Vardi. Reasoning about knowledge, vol. 4, 1995.

[54] Zheng Fang, Hao Fu, Tianbo Gu, Zhiyun Qian, Trent Jaeger, Pengfei Hu, and Prasant Mohapatra. A model checking-based security analysis framework for IoT systems. *High-Confidence Computing*, 1(1):100004, 2021.

[55] Diego Gambetta et al. Can we trust trust. *Trust: Making and breaking cooperative relations*, 13:213–237, 2000.

[56] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7):1645–1660, 2013.

[57] Andreas Herzig, Emiliano Lorini, Jomi F Hübner, and Laurent Vercouter. A logic of trust and reputation. *Logic Journal of the IGPL*, 18(1):214–244, 2010.

[58] Gerard J. Holzmann. The model checker spin. *IEEE Transactions on software engineering*, 23(5):279–295, 1997.

[59] Michael Huth and Mark Ryan. *Logic in Computer Science: Modelling and reasoning about systems*. Cambridge university press, 2004.

[60] Wojciech Jamroga, Beata Konikowska, and Wojciech Penczek. Multi-valued verification of strategic ability. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 1180–1189, 2016.

[61] Özgür Kafalı and Pınar Yolum. Detecting exceptions in commitment protocols: Discovering hidden states. In *International Workshop on Languages, Methodologies and Development Tools for Multi-Agent Systems*, pages 112–127. Springer, 2009.

[62] Athota Kavitha, Vijender Busi Reddy, Ninni Singh, Vinit Kumar Gunjan, Kuruva Lakshmanna, Arfat Ahmad Khan, and Chitapong Wechtaisong. Security in iot mesh networks based on trust similarity. *IEEE Access*, 10:121712–121724, 2022.

[63] Warda El Kholy, Jamal Bentahar, Mohamed EL Menshawy, Hongyang Qu, and Rachida Dssouli. Conditional commitments: Reasoning and model checking. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 24(2):9, 2014.

[64] Beata Konikowska and Wojciech Penczek. Reducing model checking from multi-valued ctl* to ctl. In *International Conference on Concurrency Theory*, pages 226–239. Springer, 2002.

[65] Beata Konikowska and Wojciech Penczek. Model checking for multivalued logic of knowledge and time. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 169–176, 2006.

[66] Sachin Kumar, Prayag Tiwari, and Mikhail Zymbler. Internet of things is a revolutionary approach for future technology enhancement: a review. *Journal of Big data*, 6(1):1–21, 2019.

[67] Marta Kwiatkowska, Gethin Norman, and David Parker. Prism: Probabilistic symbolic model checker. In *International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, pages 200–204. Springer, 2002.

[68] Martin Leucker and Christian Schallhart. A brief account of runtime verification. *The journal of logic and algebraic programming*, 78(5):293–303, 2009.

[69] Yongming Li, Lihui Lei, and Sanjiang Li. Computation tree logic model checking based on multi-valued possibility measures. *Information sciences*, 485:87–113, 2019.

[70] Sheng Liu, Yang Yang, and Weixing Wang. Research of AODV routing protocol for ad hoc networks1. *AASRI Procedia*, 5:21–31, 2013.

[71] Alessio Lomuscio, Charles Pecheur, and Franco Raimondi. Automatic verification of knowledge and time with nusmv. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*, pages 1384–1389. IJCAI/AAAI Press, 2007.

[72] Alessio Lomuscio and Edoardo Pirovano. Parameterised verification of strategic properties in probabilistic multi-agent systems. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, pages 762–770, 2020.

[73] Alessio Lomuscio, Hongyang Qu, and Franco Raimondi. Mcmas: A model checker for the verification of multi-agent systems. In *Computer Aided Verification: 21st International Conference, CAV 2009, Grenoble, France, June 26-July 2, 2009. Proceedings 21*, pages 682–688. Springer, 2009.

[74] Alessio Lomuscio and Franco Raimondi. Mcmas: A model checker for multi-agent systems. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 450–454. Springer, 2006.

[75] Sérgio F Lopes, António D Costa, and Henrique M Dinis Santos. Wireless networks and IoT applications. *Mobile Networks and Applications*, pages 1–3, 2023.

[76] Kenneth L McMillan. Symbolic model checking. In *Symbolic Model Checking*, pages 25–60. Springer, 1993.

[77] Daniele Miorandi, Sabrina Sicari, Francesco De Pellegrini, and Imrich Chlamtac. Internet of things: Vision, applications and research challenges. *Ad Hoc Networks*, 10:1497–1516, 2012.

[78] R. Montague. Universal grammar. *Theoria*, 36(3):373–398, 1970.

[79] Arash Mousavi, M Jan Nordin, and Zulaiha Ali Othman. Ontology-driven coordination model for multiagent-based mobile workforce brokering systems. *Applied Intelligence*, 36(4):768–787, 2012.

[80] Dang Tu Nguyen, Chengyu Song, Zhiyun Qian, Srikanth V Krishnamurthy, Edward JM Colbert, and Patrick McDaniel. IotSan: Fortifying the safety of IoT systems. In *Proceedings of the 14th International Conference on emerging Networking EXperiments and Technologies*, pages 191–203, 2018.

[81] Eugénio Oliveira, Henrique Lopes Cardoso, Maria Joana Urbano, and Ana Paula Rocha. Normative monitoring of agents to build trust in an environment for B2B. In *IFIP International Conference on Artificial Intelligence Applications and Innovations*, pages 172–181. Springer, 2014.

[82] Flauzac Olivier, Gonzalez Carlos, and Nolot Florent. New security architecture for IoT network. *Procedia Computer Science*, 52:1028–1033, 2015.

[83] Mamata Rath and Chhabi Rani Panigrahi. Prioritization of security measures at the junction of MANET and IoT. In *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies*, pages 1–5, 2016.

[84] Daniel G Reina, Sergio L Toral, Federico Barrero, Nik Bessis, and Eleana Asimakopoulou. The role of ad hoc networks in the internet of things: A case scenario for smart environments. *Internet of things and inter-cooperative computational technologies for collective intelligence*, pages 89–113, 2013.

[85] Steven Roman. *Lattices and ordered sets*. Springer Science & Business Media, 2008.

[86] P Satyanarayana, G Diwakar, BV Subbayamma, NV Phani Sai Kumar, M Arun, and S Gopalakrishnan. Comparative analysis of new meta-heuristic-variants for privacy preservation in wireless mobile ad hoc networks for IoT applications. *Computer Communications*, 198:262–281, 2023.

[87] Sharon Shoham and Orna Grumberg. Multi-valued model checking games. In *Automated Technology for Verification and Analysis: Third International Symposium, ATVA 2005, Taipei, Taiwan, October 4-7, 2005. Proceedings 3*, pages 354–369. Springer, 2005.

[88] Munindar P Singh. Agent communication languages: Rethinking the principles. *Computer*, 31(12):40–47, 1998.

[89] Munindar P Singh. An ontology for commitments in multiagent systems. *Artificial intelligence and law*, 7(1):97–113, 1999.

[90] Munindar P Singh. A social semantics for agent communication languages. In *Issues in agent communication*, pages 31–45. Springer, 2000.

[91] Munindar P Singh. Semantical considerations on dialectical and practical commitments. In *AAAI*, volume 8, pages 176–181, 2008.

[92] Munindar P Singh. Trust as dependence: a logical approach. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 863–870. International Foundation for Autonomous Agents and Multiagent Systems, 2011.

[93] Munindar P Singh. Commitments in multiagent systems: Some history, some confusions, some controversies, some prospects. *The Goals of Cognition. Essays in Honor of Cristiano Castelfranchi*, pages 601–626, 2012.

[94] Marvin A Sirbu. Credits and debits on the internet. *IEEE spectrum*, 34(2):23–29, 1997.

[95] Priya Suresh, J Vijay Daniel, Velusamy Parthasarathy, and RH Aswathy. A state of the art review on the internet of things (IoT) history, technology and fields of deployment. In *2014 International conference on science engineering and management research (ICSEMR)*, pages 1–8. IEEE, 2014.

[96] Abeer Tout, Sanaa Sharafeddine, and Nadine Abbas. UAV-assisted multi-tier computing framework for IoT networks. *Ad Hoc Networks*, 142:103119, 2023.

[97] Peerapong Uthansakul, Patikorn Anchuen, Monthippa Uthansakul, and Arfat Ahmad Khan. Estimating and synthesizing qoe based on qos measurement for improving multimedia services on cellular networks using ann method. *IEEE Transactions on Network and Service Management*, 17(1):389–402, 2019.

[98] Pedro B Velloso, Rafael P Laufer, Daniel de O Cunha, Otto Carlos MB Duarte, and Guy Pujolle. Trust management in mobile ad hoc networks using a scalable maturity-based model. *IEEE transactions on network and service management*, 7(3):172–185, 2010.

[99] Ovidiu Vermesan and Peter Friess. *Internet of things: converging technologies for smart environments and integrated ecosystems*. River publishers, 2013.

[100] Omar Abdel Wahab, Jamal Bentahar, Hadi Otrok, and Azzam Mourad. Towards trustworthy multi-cloud services communities: A trust-based hedonic coalitional game. *IEEE Transactions on Services Computing*, 11(1):184–201, 2018.

[101] Shu-Ping Wan, Jia Yan, and Jiu-Ying Dong. Trust and personalized individual semantics based fusion method for heterogeneous multi-criteria group decision making and application to live streaming commerce. *Expert Systems with Applications*, 208:118151, 2022.

[102] Xiaofeng Wang, Ling Liu, and Jinshu Su. Rlm: A general model for trust representation and aggregation. *IEEE Transactions on Services Computing*, 5(1):131–143, 2010.

[103] Matt Webster, Louise Dennis, and Michael Fisher. Model-checking auctions, coalitions and trust. In *Automated Reasoning Workshop 2009 Bridging the Gap between Theory and Practice ARW 2009*, page 43, 2009.

[104] Gerhard Weiss. *Multiagent systems: a modern approach to distributed artificial intelligence*. MIT press, 1999.

[105] Yingkun Wen, Yan Huo, Liran Ma, Tao Jing, and Qinghe Gao. Quantitative models for friendly jammer trustworthiness evaluation in iot networks. *Ad Hoc Networks*, 137:102994, 2022.

[106] Michael Wooldridge. *An introduction to multiagent systems*. John Wiley & Sons, 2009.

[107] Michael Wooldridge and Nicholas R Jennings. Intelligent agents: Theory and practice. *The Knowledge Engineering Review*, 10(2):115–152, 1995.

[108] Hui Xia, Zhiping Jia, Xin Li, Lei Ju, and Edwin H-M Sha. Trust prediction and trust-based source routing in mobile ad hoc networks. *Ad Hoc Networks*, 11(7):2096–2114, 2013.

[109] Yang Xu, Da Ruan, KY Qin, and Jun Liu. Lattice-valued logic. *Studies in fuzziness and soft computing*, 132, 2003.

[110] CS Yadav, J Singh, A Yadav, HS Pattanayak, R Kumar, AA Khan, MA Haq, A Alhussen, and S Alharby. Malware analysis in iot & android systems with defensive mechanism. electronics 2022, 11, 2354, 2022.

[111] Pınar Yolum and Munindar P Singh. Reasoning about commitments in the event calculus: An approach for specifying and executing protocols. *Annals of Mathematics and Artificial Intelligence*, 42(1-3):227–253, 2004.

[112] Gregory S Yovanof and George N Hazapis. An architectural framework and enabling wireless technologies for digital cities & intelligent urban environments. *Wireless personal communications*, 49:445–463, 2009.

[113] Irfan Zakiuddin, Michael Goldsmith, Paul Whittaker, and Paul Gardiner. A methodology for model-checking ad-hoc networks. In *Model Checking Software: 10th International SPIN Workshop Portland, OR, USA, May 9–10, 2003 Proceedings 10*, pages 181–196. Springer, 2003.

[114] Chi Zhang, Xiaoyan Zhu, Yang Song, and Yuguang Fang. A formal study of trust-based routing in wireless ad hoc networks. In *2010 Proceedings IEEE INFOCOM*, pages 1–9, 2010.