Verifying Temporal Trust Logic using CTL Model Checking

Nagat Drawel CIISE, Concordia University, Canada n_drawe@encs.concordia.ca Mohamed El Menshawy CS, Menoufia University, Egypt moh_marzok75@yahoo.com Jamal Bentahar CIISE, Concordia University, Canada bentahar@ciise.concordia.ca Amine Laarej CIISE, Concordia University, Canada laarej.amine@gmail.com

Abstract

Several formal trust frameworks have been introduced in the area of Multi-Agent Systems (MASs). However, the problem of model checking trust logics is still a challenging research topic that has not been sufficiently investigated yet. In this paper, we address this challenge by proposing a formal and fully automatic model checking technique for a temporal logic of trust. From the logical perspective, the starting point of our proposal is TCTL, a Computation Tree Logic of preconditional Trust that has been recently proposed. We extend this logic by introducing a new modality for conditional trust and describe the logical relationship between preconditional and conditional trust. From the formal verification perspective, we develop transformation-based algorithms fully implemented in a Java toolkit that automatically interacts with the NuSMV model checker. Our verification approach automatically transforms the problem of model checking TCTL into the problem of model checking CTL. We also develop a model checking algorithm for the conditional trust. We provide proofs of the soundness and completeness of our transformation algorithms. Finally, experiments conducted on a standard industrial case study of auto-insurance claim processing demonstrate the efficiency and scalability of our approach in verifying TCTL and conditional trust formulae.

1 Introduction

Trust is regarded as being one of the key aspects behind the success and growth of applications based on Multi-Agent Systems (MASs). It has been the focus of many research projects, both theoretical and practical, in the recent years, particularly in domains where open multi-agent technologies are applied (e.g., Internet-based markets, information retrieval, etc.). The importance of trust in such domains arises mainly because it provides a social control that regulates the relationships and interactions among agents. However, despite the growing number of various multi-agent applications, they still encounter many challenges in the verification of agents' behaviors. The existence of many autonomous entities in such systems makes this verification difficult due to the increase in their complexity and heterogeneity. The main challenge that faces MASs is how to ensure the reliability of the trust relationships in the presence of misbehaving entities. Such entities not only create an exception for other agents, but also may obstruct their proper work [26]. The fact that such systems

Copyright © by the paper's authors. Copying permitted only for private and academic purposes.

In: R. Cohen, M. Sensoy, and T. J. Norman (eds.): Proceedings of the 20th International Workshop on Trust in Agent Societies, Stockholm, July 2018, published at http://ceur-ws.org

usually operate in open and uncertain environments makes reasoning about trust and checking the existence of untrusted computations highly desired.

Many formalisms and approaches that facilitate the specifications of trust in MASs can be found in the literature. However, few approaches addressed trust from a high level abstraction viewpoint [12, 13, 37]. Modal logic approaches provide powerful mechanisms that can be effectively used for trust reasoning. Such approaches yield a formal semantics to reason about trust properties in various applications such as security protocols, information sources, and e-markets. For instance, in [10, 23, 31], the authors proposed several logical frameworks for the concept of trust. Trust in such logics is mostly expressed as a combination of different modalities based on the logic of action and time [22] and the BDI logic [8]. In [28], a modal logic for reasoning about the interaction between belief, evidence and trust is presented. Other approaches are interested in analyzing trust in information sources [3, 10, 11, 27]. Moreover, some proposals have addressed trust in the context of computer security [21, 31]. Most of these approaches focus on the cognitive side of trust (i.e., trusted agents are capable of exhibiting beliefs, desires, and intentions properties). Hence, the trust is considered as a belief of an agent (the truster) involving ability and willingness of the trustee to perform some actions for the truster. Although these approaches are highly appropriate to reason about trust, their verification faces a fundamental limitation due to their reliance on the internal structure of the interacting agents. In fact, the distributed and open nature of MASs makes the capability of handling and verifying the trust interaction issues of such approaches arduous. That is, it is very difficult for one agent to completely trust others by making assumptions about their internal structure.

As in [12, 13, 37], in this paper, we are considering a cognitive-independent view of trust where trust ingredients are seen from a non-epistemic angle. Trust in our work is defined from a high-level abstraction perspective without having to depend on individual agents' internal states. We represent trust as a direct relation from one agent, the truster, toward another agent, the trustee, where such a relation presupposes specific preconditions with respect to a particular content. Specifically, the truster considers the trustee as a trustworthy agent with regard to a specific content when the behavior of the trustee with respect to this content is as the truster expects, where this expectation is shared by the two participants. For example, the buyer trusts that the seller will deliver the ordered items upon the buyer's payment, where the payment consists the precondition and the order delivery is the expectation commonly shared. From the modeling and specification perspectives, we are considering the Trust Computation Tree Logic (TCTL) [13], an extension of the CTL logic [19]. Equipped with reasoning postulates, TCTL does not only provide a formal basis for reasoning about trust states with preconditions, but it can also be seen as a formal modeling of the social trust interactions among agents.

As mentioned earlier, the fact that agents are autonomous and have to interact with each other within unreliable social environments entails that deciding whether to trust another agent (for instance to perform some actions) or not is a challenging task. For instance, agents may not be able to comply with their obligations (e.g., an agent may not send the agreed payment for goods received). This raises the need for developing efficient methodologies to handle their present and future behaviors in order to ensure the fulfillment of the system requirements. Currently, the technique of model checking [6, 7] has attracted several contributions with a significant industrial implication. Although these contributions addressed a number of multi-agent aspects such as social commitments [4, 14] and knowledge [30, 41], model checking trust in multi-agent settings has not been sufficiently investigated yet. From this view, we aim in this work to contribute in the modeling and verification of trust systems. To do so, we show how trust is formally defined and present a formal and fully automatic model checking technique to verify trust properties. We consider two types of trust, analyze the relationship between them and develop transformation-based model checking techniques for verifying the properties that an agent requires to be achieved by the trusted agent. The two types are preconditional, where trust requires first the satisfaction of the precondition, and conditional, where trust is expressed with antecedents and consequences. The following example shows a typical situation in the context of electronic commerce where trust is a highly desired property.

Example 1. Let us consider the buyers-sellers relationships. The seller has to trust the buyer that the payment for the ordered goods will be sent, and the buyer has to trust the seller to ship the goods. For instance, the buyer requests to purchase one or more items from the seller (i.e., the trust relationship is established between the two parties). Once the former selects an item, the seller trusts that buyer with regard to the payment in order for the request to take place. When the requested items are paid, the seller confirms the order and starts the delivery process. Finally, the requested items are shipped and the buyer is notified.

Obviously, on-line interactions are characterized by uncertainty and, moreover, the anonymity of the interaction partners. Thus, there is no guarantee that this process will be surely satisfied in concrete applications. Therefore, the need for a logical language that can provide a certain level of abstraction with the ability to express the trust properties explicitly and a verification procedure to verify the trust interactions are of great significance. Figure 1 illustrates the overall approach of model checking TCTL, which consists of three different but integrated phases. In the first phase, we recall the logic TCTL and its formal model defined using the formalism of vector-based interpreted systems [13]. In the second phase, we

introduce our formal verification technique based on transforming the problem of model checking TCTL into the problem of model checking CTL. In the third phase, we implement our transformation technique in a Java toolkit that automatically interacts with the NuSMV model checker and report the verification results using a case study. Moreover, the paper also introduces conditional trust which is not expressed in TCTL, analyzes the logical relationship between preconditional and conditional trust and exploits this relationship to design a new algorithm to verify properties with conditional trust.



Figure 1: A schematic view of our TCTL model checking approach

The present article is organized as follows. We discuss the related work in Section 2. In Section 3, we recall the syntax and semantics of the TCTL logic framework, and introduce the conditional trust along with its syntax, semantics, and logical relationship with preconditional trust. We present our formal transformation algorithms to model check TCTL and conditional trust in Section 4 along with the soundness and completeness proofs. In Section 5, we present the case study and experimental results obtained using the tool that implements our transformation algorithms. We conclude and identify future research directions in Section 6.

2 Literature Review

While the number of proposals on trust modeling is significant, they differ, however, in the topics they addressed and the systems they implemented. In this paper, we are primarily concerned with the issues of reasoning about and verifying trust in the context of MASs using the model checking approach, which has not been deeply investigated yet for trust systems. Various approaches on trust modeling have developed modal logics that capture the important elements of the cognitive theory of trust as proposed in [5]. In these approaches, trust is considered as a mental attitude based on a specific set of goals and expressed in terms of certain beliefs. In this respect, Herzig et al. [23] proposed a formal logical framework to formally reason about occurrent and dispositional trust in a multi-agent environment. The authors defined a logic that combines temporal, dynamic, belief and choice logics. Moreover, the proposed logic is extended with operators to allow reasoning about reputation in the scope of collective beliefs. In another work [24], Herzig and his colleagues simplified the previous logic presented in [23] by considering a very simple kind of actions based on the concepts of propositional assignment. That is, the truth values of a propositional variable is assigned to either true or false by the corresponding agents' actions. The new logic provides a simple framework, and it is expressive enough to account for the cognitive theory of trust. In [28], Liu and Lorini presented a new dynamic logic called DL-BET for reasoning about the interaction between belief, evidence and trust. The authors introduced three modal operators where each of these concepts are respectively represented. In this logic, the trust operator semantics is interpreted using neighborhood semantics [32], which maps each world into a set of subsets of worlds. The authors provided a complete axiomatization for both the static component of the proposed logic and its dynamic extension. Other approaches have focused on trust in information sources. In this line, an early work presented BIT, a modal logic that extends the traditional doxastic logic with modalities for representing belief, information acquisition, and trust [27]. In the BIT formalism, the trust operator is interpreted using neighborhood semantics [32]. The logic is provided with a rigorous semantics to precisely characterize 1) the relationships among beliefs

and information acquisition, and 2) how different trust properties are represented by considering various axioms of the logic. In a related work, Demolombe and Lorini [11, 31] have focused on analyzing trust in various features of information sources. That is, a certain agent is trusting another agent if the former believes that the information transmitted to them is reliable. In particular, in [31], the authors formalized some security properties and their relationships with trust such as integrity, availability and privacy by proposing a modal operator of obligation. Fuchs and colleagues [21] also addressed trust in the context of computer security. More recent proposals have made the link between trust and argument-based frameworks [3, 34, 39]. However, the above mentioned models are not suitable for verification mechanisms using model checking techniques because of their reliance on the internal structure of the interacting agents. Practically, the fact that MASs are deployed in open environments means that these agents are managed by different providers using different types of platforms. Thus, it is very difficult for one agent to completely trust others or to make assumptions about their internal states. Moreover, model checking neighborhood semantics-based modal logics is yet to be solved [17, 33].

The closest approach to our work is the one presented by Singh in [37] where the social perspective of trust has been put forward. Specifically, the author provided a formal semantics for trust with various logical constraints used to reason about trust from an architectural point of view. This logical model combines temporal modalities of linear temporal logic (LTL) [35], modality *C* for commitments [36] and modality *T* for trust. Yet, unlike our approach that is based on interpreted systems structure with a grounded semantics, Singh's logic is interpreted using neighborhood semantics [32] whose model checking is still an open problem [17].

From the model checking point of view, some authors adopt a direct method, which can be performed by either developing a proper model checker from scratch or by extending existing tools with new algorithms for the needed temporal modalities. For instance, in our previous work [13], we proposed a new logic-based framework for specifying and model checking preconditional trust in MASs. We introduced TCTL, a branching temporal logic of preconditional trust interpreted in a new vector-based version of interpreted systems that captures the trust relationship between the interacting parties. Reasoning postulates and new symbolic model checking algorithms are presented to formally specify and automatically verify the system under consideration against some desirable properties expressed in TCTL. Thus, a new model checker extending MCMAS, called MCMAS-T, dedicated to TCTL, along with its new input language VISPL (Vectorextended ISPL) have been created. However, TCTL is restricted to preconditional trust and does not support conditional trust, and the symbolic model checking algorithms, although efficient with flat models, showed limited efficiency when the verified models are not flat (i.e., include non-self loops). In this paper, in addition to the introduction of conditional trust, we investigate a different verification mechanism that does not suffer from the non-flat-models limitation. This has been achieved thanks to the high efficiency of CTL model checking to which the model checking of TCTL and conditional trust is transformed.

Another relevant work is presented by Aldini in [2], where a formal framework to evaluate the effectiveness and robustness of trust-based models in order to detect and then isolate different kinds of attacks has been introduced. Indeed, the author integrates trust modeling with distributed systems. In this work, the system properties are expressed using a trust temporal logic (TTL) which combines CTL [19] and its action-based extension (ACTL) [9]. Moreover, the trust system model is based on an instance of both labeled transition systems and Kripke structures. The verification of temporal logic properties expressed in TTL has been performed through a mapping to an existing model checking technique. However, the model mapping between the two logics has not been specified and TTL can only specify a single agent model, and it is not adapted to autonomous MASs. El-Qurna et al. presented a model checking framework to verify service trust behaviors model against regular and non-regular properties [18]. The authors introduced an algorithm to generate a configuration graph of a deterministic pushdown automata (PDA), where the trust behaviors are captured through the observations sequences related to certain interactions between the services and users. The trust behavior properties are specified using Fixed point Logic with Chop (FLC). From the model checking prospective, a symbolic *FLC* model checking algorithm is applied in order to verify service trust behaviors with respect to trust properties. However, this approach lacks formal semantics for trust because trust formulae are inferred from the context free grammar of trust pattern languages. Moreover, the approach is not designed to formalize and verify trust in the context of MASs.

On the other hand, model checking trust can also be achieved by indirect techniques, also called transformation-based methods. The idea is to apply certain reduction rules in order to transform the problem at hand to an existing model checking problem. In fact, transformation has been acknowledged as an alternative mechanism for verifying various MASs aspects. The main advantage of this technique is that it enables the designers of MASs applications to get benefit from powerful and already tested model checkers. This technique has been applied for model checking commitments [15], knowledge [29], and the interaction between knowledge and commitments [1, 38]. To the best of our knowledge, our work is the first attempt that introduces and implements a full transformation technique for verifying trust specifications in MASs.

3 Trust and Temporal Logic - TCTL

3.1 Syntax and Semantics

In [13], we introduced TCTL, a temporal logic of trust that extends the Computation Tree Logic (CTL) [19] to enable reasoning about trust and time. The syntax and semantics of TCTL are as follows:

Definition 1. Syntax of TCTL

The syntax of TCTL is defined recursively as follows:

$$\varphi ::= true |\rho| \neg \varphi | \varphi \lor \varphi | EX\varphi | EG\varphi | E(\varphi \cup \varphi) | T_p(i, j, \varphi, \varphi)$$

where $\rho \in AP$ is an atomic proposition from the set of atomic propositions AP, E is the existential quantifier over paths, the formula $EX\varphi$ stands for " φ holds in the next state in at least one path"; $EG\varphi$ stands for "there exists a path in which φ holds globally", and the formula $E(\varphi \cup \psi)$ holds at the current state if there is some future moment for which ψ holds and φ holds at all moments until that future moment. $EF\varphi$ is the abbreviation of $E(true \cup \varphi)$. A, the universal quantifier over paths, can be defined in terms of the above as usual: $AX\varphi = \neg EX \neg \varphi$; $AG\varphi = \neg EF \neg \varphi$; and $A(\varphi \cup \psi) =$ $\neg (E(\neg \psi \cup (\neg \varphi \land \neg \psi)) \lor EG \neg \psi)$. The modality $T_p(i, j, \psi, \varphi)$ stands for "*Preconditional Trust*" and is read as "the truster *i* trusts the trustee *j* to bring about φ given that the precondition ψ holds". That is, we have the trust over the content given that the precondition is satisfied.

TCTL formula is interpreted over vector-based interpreted system formalism. [12] extended the original interpreted systems introduced by [20] to explicitly capture the trust relationship that is established between agents engaged in an interaction. The vector-based interpreted system is composed of:

- A set $Agt = \{1, \dots, n\}$ of *n* agents in which each agent $i \in Agt$ is described by:
 - A non-empty set of local states L_i , which represents the complete information that the agent can access at a particular time;
 - A set of local actions Act_i to model the temporal evolution of the system;
 - A vector v of size n, i.e., $v_{1\times n}^i$, where n is the number of agents in the system at a given time, is associated with each local state $l_i \in L_i$. (v(i), v(j), ..., v(k) are the components of the vector v^i at local state $l_i(s)$). The vector v will be used later to define the trust accessibility relation (Definition 2). Indeed, the vector v^i represents the vision of agent *i* with regard to the trust of other agents.
 - A local protocol $\rho_i : L_i \to 2^{Act_i}$ assigning a list of enabled actions that may be performed by agent *i* in a given local state L_i ;
 - A local evolution function τ_i is defined as: $\tau_i = L_i \times Act_i \rightarrow L_i$, which determines the transitions for an individual agent *i* between local states;
- A set of global states $s \in S$ that represent a snapshot of all agents in the system at a given time. A global state *s* is a tuple $s = (l_1 \dots l_n)$. The notation $L_i(s)$ is used to represent the local state of agent *i* in the global state *s*;
- $I \subseteq S$ is a set of initial global states for the system;
- The global evolution function of the system is defined as follows: $\tau : G \times ACT \longrightarrow G$, where $ACT = Act_1 \times ... \times Act_n$ and each component $a \in ACT$ is called a joint action, which is a tuple of actions;
- As in [20], a special agent *e* is used to model the environment in which the agents operate. *e* is modeled using a set of local states L_e , a set of actions Act_e , a protocol P_e , and an evolution function τ_e .

Definition 2. Model of TCTL

A model of trust generated from vector-based interpreted systems is a tuple $M_t = (S_t, R_t, I_t, \{ \rightsquigarrow_{i \to j} | (i, j) \in Agt^2 \}, V_t)$, where:

- S_t is a non-empty set of reachable global states for the system;
- $R_t \subseteq S_t \times S_t$ is the transition relation;
- $I_t \subseteq S_t$ is a set of initial global states for the system;

- $\rightsquigarrow_{i \to j} \subseteq S_t \times S_t$ is the direct trust accessibility relation for each truster-trustee pair of agents $(i, j) \in Agt^2$ defined by $\rightsquigarrow_{i \to j} iff$:
 - $l_i(s_t)(\mathbf{v}^i(j)) = l_i(s'_t)(\mathbf{v}^i(j));$
 - s'_t is reachable from s_t using transitions from the transition relation R_t ;
- $V_t: S_t \to 2^{AP_t}$ is a labeling function, where AP_t is a set of atomic propositions.

As in [13], the intuition behind the relation $\rightsquigarrow_{i \to j}$ is, for agent *i* to gain trust in agent *j*, the former identifies the states that are compatible with their trust vision with regard to the latter, i.e., where agent *i* is expecting that agent *j* is trustful. Specifically, this is obtained by comparing the element $v^i(j)$ in the local state l_i at the global state s_t (denoted by $l_i(s_t)(v^i(j))$) with $v^i(j)$ in the local state l_i at the global state s'_t (denoted by $l_i(s'_t)(v^i(j))$). Thus, the trust accessibility of agent *i* towards agent *j* (i.e., $\rightsquigarrow_{i \to j}$) does exist only if the element value that we have for agent *j* in the vector of the local states of agent *i* for both global states is the same, i.e., $l_i(s_t)(v^i(j)) = l_i(s'_t)(v^i(j))$. Finally, infinite sequences of states linked by transitions define paths. If π is a path, then $\pi(i)$ is the (i + 1)th state in π .

Definition 3. Semantics of TCTL

Given the model M_t , the satisfaction for a TCTL formula φ in a global state s_t , denoted as $(M_t, s_t) \models \varphi$, is recursively defined as follows:

 $-(M_t, s_t) \models \rho \text{ iff } \rho \in V_t(s_t);$ $-(M_t, s_t) \models \neg \varphi \text{ iff } (M_t, s_t) \nvDash \varphi;$

 $-(M_t,s_t)\models \varphi_1\vee\varphi_2$ iff $(M_t,s_t)\models \varphi_1$ or $(M_t,s_t)\models \varphi_2$;

 $-(M_t, s_t) \models EX\varphi$ iff there exists a path π starting at s_t such that $(M_t, \pi(1)) \models \varphi$;

 $-(M_t, s_t) \models E(\varphi_1 \cup \varphi_2)$ iff there exists a path π starting at s_t such that for some $k \ge 0$, $(M_t, \pi(k)) \models \varphi_2$ and $\forall 0 \le i < k$, $(M_t, \pi(i)) \models \varphi_1$;

 $-(M_t, s_t) \models EG\varphi$ iff there exists a path π starting at s_t such that $(M_t, \pi(k)) \models \varphi, \forall k \ge 0$;

 $-(M_t, s_t) \models T_p(i, j, \psi, \varphi)$ iff $(M_t, s_t) \models \psi \land \neg \varphi$ and $\exists s'_t \neq s_t$ such that $s_{t_{\sim_i \rightarrow j}} s'_t$, and $\forall s'_t \neq s_t$ such that $s_{t_{\sim_i \rightarrow j}} s'_t$, we have $(M, s'_t) \models \varphi$.

Excluding the trust modality, the semantics of TCTL state formulae is defined as usual (semantics of CTL, since the main component of TCTL is CTL). The state formula $T_p(i, j, \psi, \varphi)$ is satisfied in the model M_t at s_t iff (1) there exists a state s'_t such that $s'_t \neq s_t$ and $s_{t_{\sim_{i\to j}}}s'_t$, and (2) all the trust accessible states s'_t that are different from the current state s_t satisfy the content of trust φ .

3.2 TCTL and Conditional Trust

In [37], Singh propounds that trust must be conditional, meaning that trust should be expressed using antecedents and consequents. For example, a customer may trust a merchant as follows: "if I pay, then I trust the merchant will deliver the goods" [37]. Such a statement expresses the customer's expectation and the effect of this expectation on their future plans. Our preconditional trust modality that assumes the prior satisfaction of the precondition is different from conditional trust. Expressing conditional trust requires an extension of TCTL, and to distinguish the two languages, the extended one is called TCTL'. However, there is a logical relationship between preconditional and conditional trust. In fact, as our main objective in the paper is the verification of temporal trust, we will show how this logical relationship will be exploited to inaugurate a model checking procedure for conditional trust (see Section 4.3). The idea we aim to convey is that it is possible to decide if a given state, and thus a given model, satisfies a conditional trust formula by calling the model checking of TCTL. To show this, let us first introduce the syntax and semantics of conditional trust. From the syntax perspective, $T_c(i, j, \psi, \phi)$ is read as "agent i trusts agent j about the consequent ϕ when the antecedent ψ holds". It is worth noticing that in the case of precondition trust, for the trust to take place between the interacting agents i and j, the condition $\psi \wedge \neg \phi$ must be satisfied in the current state s_t to ensure that the precondition ψ holds before the trust content φ is brought about, while conditional trust requires the existence of at least one accessible state satisfying the antecedent ψ . This condition captures the intuition that the satisfaction of the antecedent is possible in some future. The semantics of $T_c(i, j, \psi, \varphi)$ is as follows:

 $-(M_t, s_t) \models T_c(i, j, \psi, \varphi)$ iff $s_t \models \neg \varphi$ and $\exists s'_t \neq s_t$ such that $s_{t_{\neg_i \rightarrow j}} s'_t$ and $s'_t \models \psi$, and $\forall s'_t \neq s_t$ such that $s_{t_{\neg_i \rightarrow j}} s'_t$ and $(M_t, s'_t) \models \psi$, we have $(M_t, s'_t) \models \varphi$.

The non satisfaction of the consequent φ complies with the first postulate in [37] stating that when the consequent holds, the trust in this consequent is "completed and is, therefore, no longer active". The following proposition shows the logical link between conditional and preconditional trust:

Proposition 1. $T_c(i, j, \psi, \phi) \land \psi \equiv T_p(i, j, \top, \psi \to \phi) \land \neg T_p(i, j, \top, \neg \psi).$

The proof of this proposition is direct from the semantics and omitted for the reason of limited space.

Furthermore, it is worth mentioning that conditional trust $T_c(i, j, \psi, \phi)$ is conceptually and semantically different from trust about conditions, which can be represented by $T_c(i, j, \top, \psi \rightarrow \phi)$. An example of the former is "if the customer *i* pays the merchant *j*, then *i* trusts *j* will deliver the goods", while for the latter the example is: "the customer *i* trusts the merchant *j* about the fact that, if *i* pays, then *j* will deliver the goods".

4 Formal Transformation to Model Check TCTL Logic and Conditional Trust

In this section, we first introduce a transformation-based approach to address the problem of model checking TCTL. In a nutshell, given a model M_t representing a trust based MAS and a TCTL formula φ that describes the property that the model M_t has to satisfy, the problem of model checking TCTL can be defined as verifying whether or not φ holds in M_t , which is formally denoted by $M_t \models \varphi$. In particular, we apply specific reduction rules to formally transform the problem of model checking CTL [19]. This provides a way to perform our implementation on NuSMV. Technically, our transformation method encompasses two stages. First, we apply a set of formal rules to transform vector-based transition systems into Kripke structures. Then, we transform TCTL formulae to CTL ones based on certain rules developed specifically for this purpose. Such a transformation is performed by developing two formal methods that provide accurate alignments between source and target models, and at the same time preserve TCTL semantics without losing the validity of the original model properties. This transformation is then extended to check conditional trust.

4.1 Transformation of TCTL Model

In this section, we start by recalling the definition of the CTL model needed for the transformation algorithm.

Definition 4. Model of CTL

A CTL formula is interpreted over a Kripke Structure $M_c = (S_c, R_c, I_c, V_c)$, where:

- S_c is a non-empty set of states for the system;
- $R_c \subseteq S_c \times S_c$ is the transition relation;
- $I_c \subseteq S_c$ is a set of possible initial global states for the system;
- $V_c: S_c \rightarrow 2^{AP_c}$ is a labeling function that maps each state to the set of propositional variables AP_c that hold in it.

Having presented the CTL model, the next step is to establish our transformation technique. Given a TCTL model $M_t = (S_t, R_t, I_t, \{ \rightsquigarrow_{i \to j} | (i, j) \in Agt^2 \}, V_t)$, Algorithm 1 shows how this model is transformed into a CTL model $M_c = (S_c, R_c, I_c, V_c)$. The algorithm takes as input a model M_t (line 1) and outputs the transformed model M_c (line 2). First, the corresponding model M_c has the same set of system states and initial states (i.e., $S_c = S_t; I_c = I_t$). Thereafter, the algorithm initializes the set R_c , and then the set $V_c(s)$ to be equal to the set $V_t(s)$ (i.e., at the beginning, states are labeled with the same atomic propositions). We define a new set of atomic propositions needed to represent the trust accessibility relation to capture the semantics of trust as follows $X = \{\alpha^i \alpha^j | (i, j) \in Agt^2\}$. Thus, the set AP_c is as follows: $AP_c = X \cup AP_t$. Moreover, the algorithm proceeds to transform transition and trust accessibility relations to constitute the transition relations in R_c based on two conditions. The first condition checks if the states s_t and s'_t have a transition relation in R_t , then this relation becomes a transition relation in R_c (lines 8 & 9). For the second condition, the algorithm iterates using for all ... do to go through each truster-trustee pair of agents and checks if the current state s_t has an accessible state s'_t using the accessibility relation $\rightsquigarrow_{i \to j}$ and this state is different from the state itself, then the algorithm: (1) adds a new transition from the corresponding s_c to the corresponding s'_c in R_c if such a transition is not already in R_c , and (2) a new atomic proposition is added into the label of s'_c for the interacting agents *i* and *j* in order to mark the accessible state (lines 12 & 13). Finally, the algorithm returns the transformed model M_c after iterating over all the transitions.

Algorithm 1: Transform $M_t = (S_t, R_t, I_t, \{ \rightsquigarrow_{i \to j} | (i, j) \in Agt^2 \}, V_t)$ into $M_c = (S_c, I_c, R_c, V_c)$

1: **Input**: the model M_t 2: **Output**: the model M_c 3: $S_c := S_t;$ 4: $I_c := I_t;$ 5: Initialize $R_c := \emptyset$; 6: Initialize $V_c(s_c) := V_t(s_t)$ for each $s_c \in S_c$ and $s_t \in S_t$ such that $s_c = s_t$; for each $(s_t, s'_t) \in S_t^2$ do 7: if $(s_t, s'_t) \in R_t$ then 8: $R_c := R_c \cup \{(s_c, s'_c)\}$ where $s_c = s_t$ and $s'_c = s'_t$; 9: for all $(i, j) \in Agt^2$ do 10: 11: if $s_{t \rightsquigarrow_{i \to j}} s'_t$ and $s'_t \neq s_t$ then (1) if $(s_c, s'_c) \notin R_c$ then $R_c := R_c \cup \{(s_c, s'_c)\}$ where $s_c = s_t$ and $s'_c = s'_t$; 12: (2) $V_c(s'_c) := V_c(s'_c) \cup \{\alpha^i \alpha^j\}$ where $s'_c = s'_t$; 13: 14: end if end for 15: 16: end for 17: return M_c ;

4.2 Transformation of TCTL Formulae

This section presents our method to formally transform any TCTL formula φ to a CTL formula $f(\varphi)$ using a recursive transformation function f. The details of this method are illustrated in Algorithm 2. The transformation of the CTL fragment of TCTL is straightforward (lines 1-6). For the trust modality (line 7), the trust formula is transformed inductively into CTL according to the defined semantics as follows: the transformation of the formula $\psi \land \neg \varphi$ should hold in the current state, there exists a path where next state satisfies the added atomic proposition $\alpha^i \alpha^j$, which captures the existence of an accessible state, and along all paths, the next states that satisfy the atomic proposition $\alpha^i \alpha^j$ also satisfy the transformation of the trust content φ .

Algorithm 2: Transform TCTL formula φ into CTL formula $f(\varphi)$

1: f(p) = p if p is an atomic proposition; 2: $f(\neg \varphi) = \neg f(\varphi)$; 3: $f(\varphi \lor \psi) = f(\varphi) \lor f(\psi)$; 4: $f(EX\varphi) = EXf(\varphi)$; 5: $f(E(\varphi \cup \psi)) = E(f(\varphi) \cup f(\psi))$; 6: $f(EG\varphi) = EGf(\varphi)$; 7: $f(T_p(i, j, \psi, \varphi)) = f(\psi) \land f(\neg \varphi) \land EX(\alpha^i \alpha^j) \land AX(\alpha^i \alpha^j \to f(\varphi))$;



Figure 2: Example of the transformation methods

Figure 2 depicts an example illustrating the transformation of a TCTL model and some formulae. On the left side of the figure (part a), the model M_t consists of five global states s_0 , s_1 , s_2 , s_3 and s_4 . The states s_1 , s_2 and s_4 are accessible from s_0 , while s_3 is not. Furthermore, the trust formula $(T_p(i, j, \psi, \varphi))$ holds in s_0 (i.e., $(M_t, s_0) \models T_p(i, j, \psi, \varphi)$). According

to the semantics, we obtain $(M_t, s_0) \models \psi \land \neg \varphi$, and there exists a state s' such that $s' \neq s$ and $s_{\neg i \rightarrow j} s'$, and all the trust accessible states s' such that $s \neq s'$ satisfy φ (i.e., $(M_t, s_1) \models \varphi$), $(M_t, s_2) \models \varphi$, and $(M_t, s_4) \models \varphi$)). Using the proposed transformation technique, the model M_t is transformed into the CTL model M_c of the right side (part b) as follows: the transition and accessibility relations in M_t are transformed into transition relations in M_c , and the atomic propositions $\alpha^i \alpha^j$ are added to represent the accessibility relations. Moreover, each state formula in TCTL is transformed into a CTL formula using the transformation function f. Thus, the formulae $T_p(i, j, \psi, \varphi)$ and $\psi \land \neg \varphi$ are transformed into $f(T_p(i, j, \psi, \varphi))$ and $f(\psi) \land f(\neg \varphi)$ in state s_0 , and for every path, the next state that satisfies the added atomic proposition (i.e., s_1, s_2 , and s_4) also satisfies the transformation of the trust content φ .

Theorem 1 (Soundness and Completeness of the Transformation). Let M_t and φ be respectively a TCTL model and formula and let M_c and $f(\varphi)$ be the corresponding model and formula in CTL. We have $(M_t, s_t) \models \varphi$ iff $(M_c, s_c) \models f(\varphi)$, where s_c is the corresponding state of s_t in M_c .

Proof. We prove this theorem by induction on the structure of the formula φ . All the formulae are straightforward, except the trust formula: $T_p(i, j, \psi, \varphi)$. The first and second parts: $(f(\psi) \land f(\neg \varphi))$ capture the first condition of the semantics where the current state should satisfy $\psi \land \neg \varphi$. The third part $(EX(\alpha^i \alpha^j))$ captures the second condition, i.e., the existence of an accessible state different from the current state since $\alpha^i \alpha^j$ holds only in such accessible states (line 13 of Algorithm 1). Finally, the fourth part $(AX(\alpha^i \alpha^j \to f(\varphi)))$ captures the last condition in the semantics of the trust formula where all accessible states (those satisfying $\alpha^i \alpha^j$ in M_c) should satisfy φ .

4.3 Model Checking Conditional Trust

A similar approach to model checking TCTL can be used to model check conditional trust by transforming the conditional trust formula of TCTL' to a CTL formula as follows: $f(T_c(i, j, \psi, \varphi)) = f(\neg \varphi) \land EX(\alpha^i \alpha^j \land f(\psi)) \land AX((\alpha^i \alpha^j \land f(\psi)) \rightarrow f(\varphi))$. In this section, we introduce an alternative solution that uses the developed model checking algorithm of TCTL. The algorithm of model checking conditional trust (Algorithm 3) capitalizes on the equivalence shown in Proposition 1 (line 3). If $T_p(i, j, \top, \psi \rightarrow \varphi) \land \neg T_p(i, j, \top, \neg \psi)$ does not hold, then a transformation will be needed. Such a transformation works in all cases, but to be more efficient, the algorithm uses it only if the direct condition on line 3 fails. The transformation, which also exploits Proposition 1, introduces two fresh atomic propositions: μ and κ . μ holds in the current state (line 4) and in every state where ψ holds. (line 7). Thus, if $\neg T_p(i, j, \top, \neg \mu)$ holds (condition 1), then there is an accessible state different from the current state where ψ holds. (condition 2), then $\neg \varphi$ holds and all accessible states different from the current state satisfy $\psi \rightarrow \varphi$. The algorithm returns true if the two conditions 1 and 2 hold (line 9), false otherwise (line 10). The following theorem is direct from the semantics of conditional trust and Proposition 1.

Theorem 2 (Soundness and Completeness of Algorithm 3). Algorithm 3 returns true iff $(M_t, s_t) \models T_c(i, j, \psi, \varphi)$.

Algorithm 3: Model Check $T_c(i, j, \psi, \varphi)$

1: Input: M_t , s_t , i, j, ψ , φ 2: Output: true if $(M_t, s_t) \models T_c(i, j, \psi, \varphi)$; false otherwise 3: if $(M_t, s_t) \models T_p(i, j, \top, \psi \to \varphi) \land \neg T_p(i, j, \top, \neg \psi)$ then return true; 4: $V_t(s_t) := V_t(s_t) \cup \{\mu\}$; 5: for all $s'_t \neq s_t$ 6: if $(M_t, s'_t) \models \psi \to \varphi$ then $V_t(s'_t) := V_t(s'_t) \cup \{\kappa\}$; 7: if $(M_t, s'_t) \models \psi$ then $V_t(s'_t) := V_t(s'_t) \cup \{\mu\}$; 8: end for 9: if $(M_t, s_t) \models T_p(i, j, \neg \varphi, \kappa) \land \neg T_p(i, j, \top, \neg \mu)$ then return true; 10: return false;

5 Experimental Results

5.1 Insurance Claim Processing: A Case Study

To explain our approach, we use a standard industrial case study [40] that outlines the process by which auto insurance claims are handled by an insurance company, AGFIL. There are multiple parties involved in the AGFIL cooperation process: *AGFIL, Policyholder, Europ Assist, Lee Consulting Services, Garage*, and *Assessor*. The participating parties work

together to provide a trusted service which facilitates efficient claim settlement. The process starts when the policyholder phones the call center Europ Assist to notify a new claim. Thereafter, Europ Assist registers the information and assigns an appropriate garage to provide the repair service to the policyholder. It then notifies the insurance company AGFIL which checks whether the policy is valid or not, and it confirms the claim coverage. AGFIL then sends the claim details to Lee Consulting Services (Lee CS) which is responsible for managing the operation of this service. Lee CS normally appoints an assessor to conduct a physical inspection of damaged vehicle and checks vehicle repair estimates with the garage. When repairs are completed, the garage will issue an invoice to Lee CS which will check the invoice against the original estimate. Lee CS sends all invoices to AGFIL, which in turn finalizes the payment processes.

5.2 Properties

In the above scenario, the participating parties have to ensure that the trust relationships are established on one another to perform their task accordingly. Such relationships are formalized using our model of trust $M_t = (S_t, R_t, I_t, \{ \rightsquigarrow_{i \to j} | (i.j) \in Agt^2 \}, V_t)$. To verify the correctness of the AGFIL scenario at design time, we used the safety (something bad will never happen) and liveness (something good will eventually occur) properties expressed using our TCTL' logic. Such important properties have been widely investigated in different context; for instance, by [13, 16, 25]. Formally, the safety property φ_1 expresses the negation of the bad situation where the insurance company validates the policyholder claim, but the latter never establishes their trust towards the garage with regard to the vehicle repair.

$$\varphi_1 = AG \neg (validClaim \land \neg T_p(policyholder, Repairer, validClaim, carRepair)).$$

The liveness property φ_2 states that in all paths globally, it is always the case that if the policy holder reports an accident and their claim is valid, then eventually in all future computation paths, their trust towards the insurance company with regard to the claim payment will take place.

$$\varphi_2 = AG(ReportAccident \land ValidClaim \rightarrow AF(T_p(policyholder, AGFIL, validClaim, insuranceClaimPayment)).$$

We also checked a liveness property, given by φ_3 , expressed as a conditional trust. The formula expresses the existence of a computation path where the repairer trusts AGFIL to pay for the repairs once AGFIL accepts the proposed estimate from the garage.

$\varphi_3 = EG(T_c(Repairer, AGFIL, agreeEstimate, fulfillRepairPayment)).$

We have fully implemented the presented model checking algorithms in a Java toolkit¹ that interacts automatically with the NuSMV model checker. We have verified the three properties in a parametric way in different models having different numbers of agents ranging from 6 to 30^2 . The results will be presented and discussed in the next section.

5.3 Experimental Results

We encoded our model M_t by considering the participating agents, the set of their local states and actions, the local protocol, the local evolution function, and the initial states for each agent. We also considered the accessibility relations between agents by encoding the vector variables, which give a particular agent the possibility to establish the trust towards other agents. We used the VISPL input language of the MCMAS_T model checker introduced in [13]. Thereafter, we used our transformation tool to transform the VISPL model and formulae into the SMV model in order to be able to start the verification process using NuSMV. The experiments are conducted on AMD FX-8350 - 8 Cores - 4GHZ per core with 32 GB memory. To test the scalability of our algorithms, we report five experiments in Table 1, where we consider the number of agents (Agents#), the number of reachable states (States#), the transformation times of both models and formulae in milliseconds, and the average total time calculated based on the transformation and verification times. The experiments revealed that all the tested formulae are satisfied in our models. As shown in the table, the number of reachable states reflects the fact that the state space increases exponentially when the number of agent increases. Yet, it is clear that the transformation times of both the models and formulae increase only logarithmically with regard to the number of states. It is also worth noticing that the average total time increases polynomially with respect to the number of states.

¹The toolkit is available at: https://users.encs.concordia.ca/~bentahar/Trust.jar

²The experiments are available at: https://users.encs.concordia.ca/~bentahar/TCTLCaseStudyFiles.rar

Table 1: Verification results

Agents#	States#	Time of model	Time of formula transformation			Average total
		transformation (ms)	(ϕ_1, ϕ_2, ϕ_3) (ms)		time (ms)	
6	26	0.735313	0.015955	0.015038	0.014661	1.660572
12	130	1.322853	0.022917	0.021596	0.023090	35.166989
18	566	1.847349	0.028163	0.027855	0.028726	172.008436
24	2410	2.879735	0.032878	0.032996	0.039967	938.0084367
30	17537	4.317968	0.049961	0.049811	0.050127	1271.962603

6 Conclusion

In this article, we proposed a new model checking framework for the TCTL logic of preconditional trust that is extended to design a new algorithm to model check conditional trust in MASs. We designed transformation-based algorithms that are fully implemented in a new Java toolkit that automatically interacts with the NuSMV model checker of the CTL logic. Our proposed technique is able to automatically transform the problem of model checking TCTL into the problem of model checking CTL. We also discussed the logical relationship between preconditional and conditional trust, which led to the model checking procedure of conditional trust. The proof of the soundness and completeness of our transformation algorithms is provided. Experiments conducted on a standard industrial case study demonstrated the efficiency and scalability of the technique. As future work, we plan to analyze the interaction between social commitments and trust from both, specification and model checking standpoints. Studying trust dynamics in real time MASs is another direction for further investigation.

References

- F. Al-Saqqar, J. Bentahar, K. Sultan, W. Wan, and E. Khosrowshahi Asl. Model checking temporal knowledge and commitments in multi-agent systems using reduction. *Simulation Modelling Practice and Theory*, 51:45–68, 2015.
- [2] A. Aldini. A formal framework for modeling trust and reputation in collective adaptive systems. In *Proceedings of the Workshop on FORmal methods for the quantitative Evaluation of Collective Adaptive SysTems, FORECAST@STAF, Vienna, Austria*, pages 19–30, 2016.
- [3] L. Amgoud and R. Demolombe. An argumentation-based approach for reasoning about trust in information sources. *Argument & Computation*, 5(2-3):191–215, 2014.
- [4] J. Bentahar, M. El-Menshawy, H. Qu, and R. Dssouli. Communicative commitments: Model checking and complexity analysis. *Knowledge-Based Systems*, 35:21–34, 2012.
- [5] C. Castelfranchi and R. Falcone. Principles of trust for MAS: cognitive anatomy, social importance, and quantification. In *Proceedings of the Third International Conference on Multiagent Systems, ICMAS*, pages 72–79, 1998.
- [6] E. M. Clarke, E. A. Emerson, and J. Sifakis. Model checking: algorithmic verification and debugging. *Communica*tions of the ACM, 52(11):74–84, 2009.
- [7] E. M. Clarke, O. Grumberg, and D. Peled. Model checking. MIT press, 1999.
- [8] P. R. Cohen and H. J. Levesque. Intention is choice with commitment. Artificial Intelligence, 42(2-3):213–261, 1990.
- [9] R. De Nicola and F. Vaandrager. Action versus state based logics for transition systems. In *Semantics of Systems of Concurrent Processes*, pages 407–419. Springer, 1990.
- [10] R. Demolombe. To trust information sources: A proposal for a modal logical framework. In *Trust and deception in virtual societies*, pages 111–124. Springer, 2001.
- [11] R. Demolombe and E. Lorini. A logical account of trust in information sources. In *Proceedings of the 11th International Workshop on Trust in Agent Societies*, 2008.

- [12] N. Drawel, J. Bentahar, and E. M. Shakshuki. Reasoning about trust and time in a system of agents. In *The International Conference on Ambient Systems, Networks and Technologies (ANT)*, volume 109 of *Procedia Computer Science*, pages 632–639, 2017.
- [13] N. Drawel, H. Qu, J. Bentahar, and E. Shakshuki. Specification and automatic verification of trust-based multi-agent systems. *Future Generation Computer Systems*, 2018.
- [14] W. El Kholy, J. Bentahar, M. El-Menshawy, H. Qu, and R. Dssouli. Conditional commitments: Reasoning and model checking. ACM Transactions on Software Engineering and Methodology (TOSEM), 24(2):9, 2014.
- [15] M. El-Menshawy, J. Bentahar, and R. Dssouli. Symbolic model checking commitment protocols using reduction. In International Workshop on Declarative Agent Languages and Technologies, pages 185–203. Springer, 2010.
- [16] M. El Menshawy, J. Bentahar, W. El Kholy, and A. Laarej. Model checking real-time conditional commitment logic using transformation. *Journal of Systems and Software*, 2018.
- [17] M. El-Menshawy, J. Bentahar, W. El Kholy, P. Yolum, and R. Dssouli. Computational logics and verification techniques of multi-agent commitments: survey. *The Knowledge Engineering Review*, 30(5):564–606, 2015.
- [18] J. El-Qurna, H. Yahyaoui, and M. Almulla. A new framework for the verification of service trust behaviors. *Knowledge-Based Systems*, 2017.
- [19] E. A. Emerson. Temporal and modal logic. In Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics, pages 995–1072. MIT Press, 1990.
- [20] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. Reasoning about Knowledge. MIT Press, 1995.
- [21] A. Fuchs, S. Gürgens, and C. Rudolph. A formal notion of trust–enabling reasoning about security properties. In *IFIP International Conference on Trust Management*, pages 200–215. Springer, 2010.
- [22] D. Harel, D. Kozen, and J. Tiuryn. Dynamic Logic. MIT press, 2000.
- [23] A. Herzig, E. Lorini, J. F. Hübner, and L. Vercouter. A logic of trust and reputation. *Logic Journal of IGPL*, 18(1):214–244, 2010.
- [24] A. Herzig, E. Lorini, and F. Moisan. A simple logic of trust based on propositional assignments. In F. Paglieri, L. Tummolini, and R. Falcone, editors, *The Goals of Cognition. Essays in Honour of Cristiano Castelfranchi*, Tributes, pages 407–419. College Publications, 2012.
- [25] O. Kafali, N. Ajmeri, and M. P. Singh. Kont: Computing tradeoffs in normative multiagent systems. In Proceedings of the 31st Conference on Artificial Intelligence (AAAI), pages 3006–3012, 2017.
- [26] Ö. Kafalı and P. Yolum. Detecting exceptions in commitment protocols: Discovering hidden states. In *International Workshop on Languages, Methodologies and Development Tools for Multi-Agent Systems*, volume 6039 of LNCS, pages 112–127, 2009.
- [27] C. Liau. Belief, information acquisition, and trust in multi-agent systems-a modal logic formulation. *Artif. Intell.*, 149(1):31-60, 2003.
- [28] F. Liu and E. Lorini. Reasoning about belief, evidence and trust in a multi-agent setting. In *International Conference* on *Principles and Practice of Multi-Agent Systems*, pages 71–89. Springer, 2017.
- [29] A. Lomuscio, C. Pecheur, and F. Raimondi. Automatic verification of knowledge and time with NuSMV. In Proceedings of the 20th International Joint Conference on Artificial Intelligence, pages 1384–1389, 2007.
- [30] A. Lomuscio, H. Qu, and F. Raimondi. MCMAS: an open-source model checker for the verification of multi-agent systems. STTT, 19(1):9–30, 2017.
- [31] E. Lorini and R. Demolombe. Trust and norms in the context of computer security: A logical formalization. In *International Conference on Deontic Logic in Computer Science*, pages 50–64. Springer, 2008.
- [32] R. Montague. Universal grammar. Theoria, 36(3):373-398, 1970.

- [33] E. Pacuit. Neighborhood Semantics for Modal Logic. Springer, 2017.
- [34] S. Parsons, K. Atkinson, Z. Li, P. McBurney, E. Sklar, M. Singh, K. Haigh, K. Levitt, and J. Rowe. Argument schemes for reasoning about trust. *Argument & Computation*, 5(2-3):160–190, 2014.
- [35] A. Pnueli. The temporal logic of programs. In 18th Annual Symposium on Foundations of Computer Science, pages 46–57, 1977.
- [36] M. P. Singh. Semantical considerations on dialectical and practical commitments. In Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI, Chicago, Illinois, USA, pages 176–181, 2008.
- [37] M. P. Singh. Trust as dependence: a logical approach. In *The 10th International Conference on Autonomous Agents and Multiagent Systems*, pages 863–870, 2011.
- [38] K. Sultan, J. Bentahar, W. Wan, and F. Al-Saqqar. Modeling and verifying probabilistic multi-agent systems using knowledge and social commitments. *Expert Systems with Applications*, 41(14):6291–6304, 2014.
- [39] Y. Tang, K. Cai, P. McBurney, E. Sklar, and S. Parsons. Using argumentation to reason about trust and belief. *Journal of Logic and Computation*, 22(5):979–1018, 2012.
- [40] P. R. Telang and M. P. Singh. Enhancing tropos with commitments. In *Conceptual Modeling: Foundations and Applications*, pages 417–435. Springer, 2009.
- [41] R. van der Meyden and K. Su. Symbolic model checking the knowledge of the dining cryptographers. In *17th IEEE Computer Security Foundations Workshop*, pages 280–291, 2004.