Knowledge-Based Systems xxx (2012) xxx–xxx

Contents lists available at SciVerse ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Communicative commitments: Model checking and complexity analysis

Jamal Bentahar [a,*], Mohamed El-Menshawy [a], Hongyang Qu [b], Rachida Dssouli [a]

[a] Concordia University, Faculty of Engineering and Computer Science, Concordia Institute for Information Systems Engineering, EV Building, 1515 Ste Catherine Street West, Montreal, Canada
[b] Oxford University, Department of Computer Science, Wolfson Building, Parks Road, Oxford, UK

ARTICLE INFO

Article history:
Received 26 August 2011
Received in revised form 2 March 2012
Accepted 5 April 2012
Available online xxxx

Keywords:
Multi-agent systems
Social commitments
Agent communication
Model checking
Complexity

ABSTRACT

We refine CTLC, a temporal logic of social commitments that extends CTL to allow reasoning about commitments agents create when communicating and their fulfillment. We present axioms of commitments and their fulfillment and provide the associated BDD-based model checking algorithms. We also analyze the time complexity of CTLC model checking in explicit models (i.e., Kripke-like structures) and its space complexity for concurrent programs, which provide compact representations. We prove that although CTLC extends CTL, their model checking algorithms still have the same time complexity for explicit models, which is P-complete with regard to the size of the model and length of the formula, and the same complexity for concurrent programs, which is PSPACE-complete with regard to the size of the components of these programs. We fully implemented the proposed algorithms on top of MCMAS, a model checker for the verification of multi-agent systems, and provide in this paper simulation results of an industrial case study.

1. Introduction

1.1. Context

Social approaches are increasingly used to define a formal semantics for agent communication languages (ACLs) [46,1,60,16]. These approaches particularly aim to overcome the shortcomings of ACLs semantics defined using mental approaches where the semantics is expressed in terms of agents' internal states such as beliefs, desires and intensions [57]. Commitments are employed in some of these social approaches, which successfully provide a powerful representation for modeling multi-agent interactions [46,13,14,5]. In broad terms, commitments are social, public and help represent the state of affairs in the course of multi-agent interactions. Each social commitment is made by one agent, the debtor, and directed towards another agent, the creditor, to bring about a certain fact [45]. The following is an example of commitment:

Example 1. A merchant *Mer* commits to deliver the item *a* to a customer *Cus*.

The strong point in social approaches under the consideration of commitments is the capability to manipulate commitments via a set of actions called commitment actions (operations) such as creation, discharge, violation, cancelation, release, delegation and assignment [45]. In the last decade, social commitments have been extensively and effectively used in a variety of areas ranging from modeling business processes [14], developing artificial institutions [20], defining programming languages [55], developing Web-based applications [52] to specifying and modeling multi-agent interaction protocols, called commitment-based protocols [60,13,33,2].

Conventionally, the social semantics of ACL messages should satisfy four crucial criteria introduced in [46]: (1) formal (based on some temporal logics); (2) declarative (focuses on what the message means, instead of how the message is exchanged); (3) verifiable (we can check if the agents are acting according to the semantics); and (4) meaningful (the focus is on the content of messages, not on their representation as tokens).

1.2. Motivations

Previous approaches have considered defining the semantics of commitments by means of temporal logics, which we call "logics of commitments". We classify these approaches into two classes. The first class extends temporal logics with predicates to represent and reason about commitments and associated actions, such as [58,59,32,34,39]. The second class enriches temporal logics with modalities for commitments and their actions, such as [46,6,7,5,47,16,17].

On the one hand, the main problem of using predicates is that they fail in representing the notions that are *referentially opaque*. However, the notion of commitments has the property of being referentially opaque, which has an *opaque context*. Simply put, this

* Corresponding author. Tel.: +1 514 848 2424x5382; fax: +1 514 848 3171.
E-mail addresses: bentahar@ciise.concordia.ca (J. Bentahar), m_elme@encs.concordia.ca (M. El-Menshawy), Hongyang.Qu@cs.ox.ac.uk (H. Qu), dssouli@ece.concordia.ca (R. Dssouli).

0950-7051/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.knosys.2012.04.010

means substitution rules usually used in predicate logic cannot be applied. For instance, in Example 1, even if item *a* is equivalent to item *b* committing to deliver item *a* is not like committing to deliver item *b* although items *a* and *b* are equal terms in the sense of predicate logic (i.e., they are co-referential terms). In other words, "substituting equivalents into opaque contexts is not going to preserve meaning" [56]. To solve the problem of using predicates, Verdicchio and Colombetti [53,54] used *many sorted first order logic* to represent and reason about commitments. However, they do not address the problem of commitment verification.

On the other hand, while approaches in the second class have taken a good step towards both developing modal logics for commitments and achieving the four Singh's criteria, they are not sufficiently general and clear in their assumptions. Singh [46] extended computational tree logic (CTL) [11] with modalities for social commitments, beliefs and intentions to specify and reason about the interactions of autonomous and heterogeneous agents within multi-agent systems (MASs). He particularly presented three accessibility relations to capture the meaning of these modalities, for example the semantics of commitments is defined as the content is true along every accessible path starting from commitment state. However, the work does not clarify the intuition an accessible path captures and how accessible paths are computed, which limits the applicability of this approach for verification purposes (we refer to this problem as lack of intuition and computation). Furthermore, this work does not consider the semantics of commitment actions and the approach is not purely social as it refers to a mental component by claiming that communication should be sincere.

Bentahar et al. [6,7] developed "Commitment and Argument Network (CAN)", which is a unified framework for pragmatic and semantic issues. It uses both a temporal logic CTL^{*CA}, which extends CTL^{*} with modalities for commitments and their actions, argument modality, and a dynamic logic (DL) to define a logical semantics for agent communication. They introduced an accessibility relation to define the semantics of commitments. However, the intuition this accessibility relation is capturing and how it could be computationally implemented were not clarified. The semantics of the *Satisfy* (discharge) formula (i.e., satisfaction action) is defined in terms of whether a commitment has been created in the past and still active and its content holds, and a commitment is active if it is not satisfied and violated yet. This semantics is defined in a recursive manner, which makes its verification extremely hard. In addition, the satisfaction semantics focuses on checking the truth condition of the content of commitment, which is a part of the semantics of commitment itself. This problem is also present in Mallya et al.'s framework introduced in [34].

Singh [47] delineated the model-theoretic semantics for social commitments by postulating some rules that can be used as ways to reason about commitments and solely discharge action. This model combines temporal modalities of linear temporal logic (LTL) [11] and modalities for two kinds of commitments: practical commitments, which are about what is to be done, and dialectical ones, which are about what holds. This semantics is interpreted without using an accessibility relation but using Segerberg's idea, which maps "each world into a set of worlds". For instance, to define the semantics of dialectical commitments, the author introduced a function $\mathbb{D}$ that produces a set of propositions for each moment, proposition, and a pair of two agents. This function yields a set where each member is a consequent proposition, which captures what the debtor would be committed to if the antecedent is met. Thus, the semantics is defined by computing the set of moments where the commitment content holds and testing if those moments are among the moments computed by $\mathbb{D}$ on the commitment antecedent. However, how in practice the function $\mathbb{D}$ is computed is not specified, which makes the possibility of implementing and applying such an approach vague. It is also hard to verify the proposed logic using model checking as we need first to find an equivalent semantics using Kripke structures or interpreted systems.

Bentahar et al. [4,5] and El-Menshawy et al. [16] showed how social commitments can be mapped into underlying logic-based formalisms by extending CTL^{*} with modalities to reason about social commitments and action formulae as well as state formulae in order to formally specify multi-agent interaction protocols and some desirable protocol properties. However, the intuition behind the accessibility relation used to define the semantics of commitments and how it can be implemented were not discussed in depth. We can underline that the definition of a clear and verifiable semantics of commitments and related concepts is an objective yet to be reached.

In addition to the above motivation, the present paper aims at applying formal verification techniques using model checking to directly and automatically verify commitments and their fulfillment and violation in practical and industrial scenarios. Examples of such scenarios and applications include business models, commitment-based protocols, agent-based web services and their communities [3]—especially when there is possibility that interacting agents may not behave as they are committed to. In e-business, e-negotiation, community of web services management, B2B settings, and any autonomous agent-based application where different components are developed by different vendors with competing interests, it is unrealistic to assume that all agents will behave according to a given protocol. Concretely, formal verification by means of model checking is beneficial to help model and protocol designers detect and then eliminate errors so that such models and protocols comply with specifications at design time. They also reduce the cost of development process and increase confidence on the safety, efficiency and robustness of models and protocols.

The rest of the paper is organized as follows. Section 2 presents a literature review highlighting problems in existing approaches and our contributions. In Section 3, we briefly summarize the formalism of interpreted systems [19] introduced to model MASs and define the syntax and semantics of CTLC along with the axiomatization of commitments and their fulfillment. The problem of CTLC model checking and the development of a new BDD-based algorithms to perform model checking for the proposed logic are presented in Section 4. We analyze the complexity of CTLC model checking in Section 5. In Section 6, we show how the proposed algorithms are implemented on top of MCMAS to automatically verify a widely discussed case study in agent interactions, the insurance claim processing problem. We discuss relevant work and conclude the paper in Sections 7 and 8, respectively.

2. Literature review and contributions

Several approaches have been put forward to address the challenge of model checking commitments. Telang and Singh [50] used the NuSMV[1] symbolic model checker to verify whether or not the operational model defined in the UML sequence diagrams correctly supports the business model that is aggregated from a set of business patterns. These patterns are defined in a highly abstract level based on the notion of commitments and informally translated into CTL specifications. By informally, we mean the translation rules are not defined in a systematic and formal way so the equivalence between the patterns and CTL specifications can be proved. The commitment is simply defined as an isolated SMV module, which can be instantiated as a domain variable in the main module.

[1] http://nusmv.fbk.eu/NuSMV/download/getting-v2.html.

Please cite this article in press as: J. Bentahar et al., Communicative commitments: Model checking and complexity analysis, Knowl. Based Syst. (2012), http://dx.doi.org/10.1016/j.knosys.2012.04.010

Bentahar et al. [5] presented a new logic called ACTL* (an extension of CTL* with action formulae) to specify protocols and some desirable properties. Their verification method is based on an informal translation of ACTL* formulae and protocol into Alternating Büchi Tableau Automata (ABTA) in order to directly use the CWB-NC[2] automata-based model checker where the commitment states are simply defined as variables and actions as atomic propositions using CCS (the input language of CWB-NC).

El-Menshawy et al. [16] introduced a new branching time logic, CTL^{*sc}, which extends CTL* with commitments and associated actions to drive a new specification language of commitment-based protocols. The symbolic verification technique used in this work is based on reducing CTL^{*sc} into LTL^{sc} and CTL^{sc} frames and then defining the participating agents in the protocol as simple SMV modules using SMV (the input language of NuSMV) and agent sections using ISPL (the input language of the MCMAS[3] symbolic model checker). In this approach, the commitment states and commitment actions are only defined as local state variables.

Gerard and Singh [21] used CTL and MCMAS to verify the refinement of commitment-based protocols by developing a preprocessor tool that reads protocols and specifications from files and then informally translates them into the ISPL language.

Desai et al. [13] developed the idea of supporting the verification of properties geared toward the composition of commitment-based protocols. These properties are specified in LTL and the approach depends on informally translating the protocol into the PROMELA language and analyzing the resulting model with the SPIN automata-based model checker. In this technique, the commitments are represented as simple SPIN processes. Another technique has been proposed by Yolum and Singh [60] using the event calculus where the semantics of protocol messages is described by predicates on events and fluents. A static verification method via an event calculus planner to check if the agent behaviors at run time comply with the given protocol specifications has been defined. Based on this, Chesani et al. [9] developed a run time commitment verification procedure to track the status of commitments.

Although the above approaches have made significant progress, there is only one work [18] on model checking commitment logics, where commitments are not simply translated into domain variables, but verified based on their semantics. In fact, translation-based approaches have the problem of preventing verifying the real and concert semantics of commitments and related concepts as defined in the underlying logics. These approaches only provide partial solution to the problem of model checking commitments [5,16] as they reduce commitment modalities into domain variables, which–as in predicate logics–stop distinguishing among various "modes" of truth such as necessarily true and true in the future. Simply put, translation-based approaches use variables that do not account for the meaning of commitments, so commitments are dealt with as tokens. Furthermore, translation-based approaches may not be straightforward and prone to errors [13,21,50], particularly in the context of complex systems as there are no implemented tools to perform the translation process, and the lack of a full and dedicated model-checking algorithm adds overhead over the actual verification process to perform the translation preprocessing step. Finally, analyzing the time and space complexity of model checking commitments has not been addressed yet.

The present paper is an improvement and continuation of our previous work published in [18] in which we developed a new symbolic algorithm for model checking an extended logic of CTL [11] with modalities for commitments and their fulfillment (discharge) and violation. The algorithm has been fully implemented on top of the MCMAS model checker [29] to directly verify commitments and their fulfillment and violation. Specifically, here we first refine this logic by redefining the social accessibility relation used to define the semantics of commitments. The objective of this new definition is to account for the intuition that commitments are conveyed through communication between interacting agents. Thus, we have a more intuitive definition of commitment than the one defined in the previous work. In fact, communication between agents is considered as the first-class entity in our new approach, which means commitments are made through communication and for communication purposes and not via blackboard or implicitly. So, social commitments in our approach can be called Communicative Commitments. Second, we: (i) alleviate the semantics of fulfilling a commitment by removing the future condition, which is unnecessary condition for the meaning of fulfilling communicative commitments (see the previous semantics of fulfilling a commitment in Section 7); (ii) consider the relationship between commitments and their fulfillment, which is still missing in all the existing approaches; and (iii) remove the violation modality; instead we show how to express violation as property in the refined logic (see Section 3.2 for details).

Furthermore, in this paper we proceed to present the axiomatization and complexity analysis of the refined logic and its model checking, which have not been considered in our previous work. We prove that (i) the time complexity of CTLC model checking in explicit models (i.e., Kripke-like structures) is P-complete with regard to the size of the model and length of the formula; and (ii) the complexity of the same problem for concurrent programs is PSPACE-complete with respect to the size of the program's components. Consequently, our model checking algorithm has the same complexity as model checking CTL with regard to both explicit models [44] and concurrent programs [28]. The motivation behind considering the complexity of model checking for concurrent programs is that in practice, existing model checkers such as MCMAS, NuSMV, and CWB-NC operate on concurrent programs, which are composed of n concurrent processes P_i where each process is described by a transition system (the formal definition is given later in Section 5). Those processes are implemented as agents, modules, or protocols depending on the model checker. More precisely, the Kripke structures that are used to model the systems' configurations are in fact obtained by making several components interact, leading to a combinatorial explosion of the number of possible configurations, so the need for compact representations. In such representations, states and transitions are not listed explicitly as global states and transitions for the system, but only local states and transitions of each component are represented, so that the actual system can still be represented by combining local states and transitions to build reachable states. In general, the relation between explicit models (i.e., Kripke-like structures), which are very large and concurrent programs, which provide compact representations of the systems to be checked, is as stated in [28]: "the Kripke structures to which model checking is applied are often obtained by constructing the *reachability graph of concurrent programs*". In other terms, the explicit models are obtained as the product of the components P_i of concurrent programs. The size of explicit models is thus exponential in the size of processes P_i as the system's evolution results from joint actions of the components [23].

Moreover, we develop and fully implement a new BDD-based model checking algorithm for communicative commitments and their fulfillment on top of CTL and MCMAS. Finally, to check the effectiveness of our approach, we report on the experimental results of verifying a concrete industrial case study, called *Insurance Claim Processing*, which is widely used in the literature. In fact, our

[2] http://www.cs.sunysb.edu/cwb/.
[3] http://www-lai.doc.ic.ac.uk/mcmas/download.html.

Please cite this article in press as: J. Bentahar et al., Communicative commitments: Model checking and complexity analysis, Knowl. Based Syst. (2012), http://dx.doi.org/10.1016/j.knosys.2012.04.010

logic provides balance between expressiveness (expressing commitments, their fulfillment and connection between commitments and fulfillment, which cannot be expressed in CTL only) and verification efficiency. CTLC also addresses the aforementioned shortcomings in existing commitment logics [46,47,6,7,5,16]. A dedicated logic and its model checking for commitments play the same role as CTLK (a combination of CTL with the logic of knowledge) [38] and MCMAS do for knowledge. Our approach can complement the static verification method introduced by Yolum and Singh [60].

3. Interpreted systems and CTLC

3.1. Interpreted systems

The formalism of interpreted systems was potentially investigated by Fagin et al. [19]. This formalism provides a general framework for reasoning and systematically exploring fundamental classes of MASs such as synchronous and asynchronous. We advocate this formalism for many reasons summarized as follows:

1. It allows us to abstract from the details of the components and focus only on modeling key characteristics of the agents and the evolution of their social commitments, which is missing in existing agent communication models.
2. It is a useful tool for ascribing autonomous and social behaviors of interacting agents within real MASs.
3. It supports the interoperability between global (system) and local (agent) models.

Interpreted systems can be defined as follows [19]. Consider a set of n agents $\mathcal{A} = \{1, \ldots, n\}$ in which at any given time each agent in the system is in a particular local state. Intuitively, each local state of an agent represents the complete information about the system that the agent has at his disposal at a given moment. We associate a non-empty and countable set L_i of local states for each agent $i \in \mathcal{A}$.

As in [19], we represent the instantaneous configuration of all agents in the system at a given time via the notion of global state. The set of all global states is denoted by G and a global state $g \in G$ is a tuple $g = (l_1, \ldots, l_n)$ where each element $l_i \in L_i$ represents a local state of agent i. Thus, the set of all global states $G \subseteq L_1 \times \ldots \times L_n$ is a non-empty subset of the Cartesian product of all local states of n agents. We use the notation $l_i(g)$ to represent the local state of agent i in the global state g. $I \subseteq G$ is a set of initial global states for the system.

To account for the temporal evolution of the system, the formalism of interpreted systems associates with each agent i the set Act_i of local actions. It is assumed that $null \in Act_i$ for each agent i, where *null* refers to the silence action (i.e., the fact of doing nothing). The action selection mechanism is given by the notion of local protocol $\mathcal{P}_i : L_i \to 2^{Act_i}$ for each $i \in \mathcal{A}$. That is $\mathcal{P}_i$ is a function giving the set of enabled actions that may be performed by i in a given local state.

As in [19], an interpreted system is a synchronous model. So, we can define the global evolution (or transition) function as follows: $\tau: G \times ACT \to G$, 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 (one for each agent). In addition, an evolution function τ_i that determines the transitions for an individual agent i between his local states is defined as follows: $\tau_i: L_i \times Act_i \to L_i$, where if $\tau_i(l_i(g), null) = l_i(g')$ for two given global states g and g', then $l_i(g) = l_i(g')$.

In this paper, we extend the interpreted system formalism to account for communication that occurs during the execution of MAS. This extension allows us to provide an intuitive semantics for social commitments that are established through communication between interacting agents. Thus, associated with each agent $i \in \mathcal{A}$ is a set Var_i of n local boolean variables, i.e., $|Var_i| = n$, which are used to represent communication channels (each agent has one communication channel with each agent) through which messages are sent and received. Each local state $l_i \in L_i$ of agent i is associated with different values obtained from different assignments to variables in Var_i. We denote the value of a variable x in the set Var_i at local state $l_i(g)$ by $l_i^x(g)$. Thus, if $l_i(g) = l_i(g')$, then $l_i^x(g) = l_i^x(g')$ for all $x \in Var_i$. The idea is that, as in distributed systems, for two agents i and j to communicate, they should share a communication channel, which is represented by a shared variable between i and j. In this perspective, a communication channel between i and j does exist iff $\exists! x \in Var_i \cap Var_j$, which means $|Var_i \cap Var_j| = 1$. For the boolean variable $x \in Var_i \cap Var_j$, $l_i^x(g) = l_j^x(g')$ means the values of x in $l_i(g)$ for i and in $l_j(g')$ for j are the same. This intuitively represents the existence of a communication channel between i (in g) and j (in g') through which the boolean variable x has been sent by one of the two agents to the other, and as a consequence of this communication, i and j will have the same value for this variable.

As in [19], this paper moves away from Kripke models while still benefiting from most of its technical apparatus. In fact, the semantics of our modal language is interpreted using a model generated from the interpreted system formalism.

Definition 1 (*Models*). A model $\mathcal{M}_C = (S, I, R_t, \{\sim_{i\to j}\}_{(i,j)\in\mathcal{A}^2}, \mathcal{V})$ that belongs to the set of all models $\mathbb{M}$ is a tuple, where:

- $S \subseteq L_1 \times \cdots \times L_n$ is a set of reachable[4] global states for the system.
- $I \subseteq S$ is a set of initial global states for the system.
- $R_t \subseteq S \times S$ is the transition relation defined by $(s, s') \in R_t$ iff there exists a joint action $(a_1, \ldots, a_n) \in ACT$ such that $\tau(s, a_1, \ldots, a_n) = s'$.
- For each pair $(i,j) \in \mathcal{A}^2$, $\sim_{i\to j} \subseteq S \times S$ is the social accessibility relation defined by $s \sim_{i\to j} s'$ iff $l_i(s) = l_i(s')$ and $\exists! x \in Var_i \cap Var_j$ s.t. $l_i^x(s) = l_j^x(s')$ and $\forall y \in Var_j - Var_i$ we have $l_j^y(s) = l_j^y(s')$ and s' is reachable from s using transitions from the transition relation R_t.
- $\mathcal{V} : S \to 2^{\Phi_p}$ is a valuation function, where Φ_p is the set of atomic propositions.

The social accessibility relation $\sim_{i\to j}$ from a global state s to another global state $s'(s \sim_{i\to j} s')$ captures the intuition that there is a communication channel between i and $j(\exists! x \in Var_i \cap Var_j)$ and s' is a resulting state from this communication initiated by i at s. The channel is thus filled in by i in s, and in s' j receives the information (i.e., the channel's content), which makes the value of the shared variable the same for i and $j\left(l_i^x(s) = l_j^x(s')\right)$. As i is the agent who intentionally initiates the communication, the source and target (or resulting) states s and s' are indistinguishable for i $(l_i(s) = l_i(s'))$ as i is not learning any new information. And as j is the agent who receives the communication, s and s' are indistinguishable with regard to the variables that have not been communicated by i, i.e., unshared variables $\left(l_j^y(s) = l_j^y(s') \forall y \in Var_j - Var_i\right)$. We illustrate this idea in Fig. 1. It is worth to mention that shared variables are used solely to model communication channels to define the social accessibility relation, which in turn will be used to define the semantics of commitment, meaning that shared variables are completely independent or unrelated to commitment content.

The need for modeling complex and open systems such as MASs using the formalism of interpreted systems is typically conducted by using logic-based formalisms as formal tools for expressing

[4] S contains only states that are reachable from the set of initial states I by means of transition relation R_t.

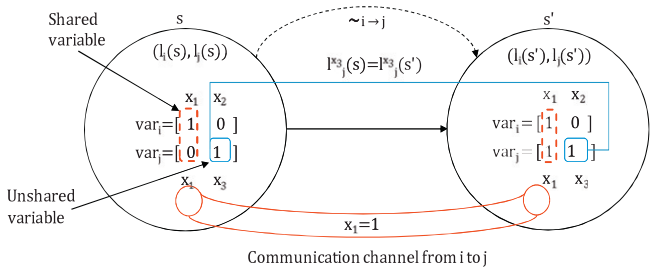

Fig. 1. An example of social accessibility relation $\sim_{i\to j}$. In this example, two agents are communicating and the shared and unshared variables for those agents are as follows. Agent i: $Var_i = \{x_1, x_2\}$. Agent j: $Var_j = \{x_1, x_3\}$. In the figure x_1 is the shared variable (i.e., it represents the communication channel between i and j), and x_3 is a j's variable unshared with i. Notice that the value of the variable x_1 for j in the state s' is changed to be equal to the value of this variable for agent i, which illustrates the message passing through the channel. However, the value of the unshared variable x_3 is unchanged.

properties of real systems. In the following, we introduce a branching time logic for commitments, called CTLC.

3.2. Computation tree logic for commitments (CTLCs)

There are three issues to be addressed when developing logical formalisms: (i) *well-formed formulae* (*wff*); (ii) *proof-theory*; and (iii) *model-theory*. The *wff* of the logic are the statements that can be made in it, proof-theory includes the axioms and rules of inference, which state entailment relationships among *wff*, and model-theory gives the formal meaning of the *wff*. The language and proof-theory are called the syntax whilst the model-theory is called the semantics.

The syntax of CTLC, which is a combination of branching time CTL [11] with social commitments is defined as follows:

Definition 2. (Syntax of CTLC)

$$\varphi ::= p|\neg\varphi|\varphi \vee \varphi|EX\varphi|E(\varphi U\varphi)|EG\varphi|C_{i\to j}\varphi|Fu(C_{i\to j}\varphi)$$

where $p \in \Phi_p$ is an atomic proposition, X, G and U are CTL path temporal connectives standing for "next", "globally" and "until" respectively, and E is the existential quantifier on paths. The modal connectives $C_{i\to j}$ and Fu stand for "commitment" and "fulfillment of commitment", respectively. $C_{i\to j}\varphi$ is read as "agent i commits towards agent j that φ", or equivalently from communication perspective as "i is conveying information φ to j", or simply as "φ is committed to" when i and j are understood from the context. $Fu(C_{i\to j}\varphi)$ is read as "the commitment $C_{i\to j}\varphi$ is fulfilled". Other temporal modalities, e.g., F (future), and the universal path quantifier A can be defined in terms of the above as usual (see for example [11]). We use customary abbreviations—"possibility of committing" ($\widehat{C}_{i\to j}$: the modality dual to $C_{i\to j}$), "implies" ($\supset$), "false" constant ($\bot$) and "true" constant ($\to p$)—as follows:

- $\widehat{C}_{i\to j}\varphi \triangleq \neg C_{i\to j}\neg\varphi$,
- $\varphi \supset \psi \triangleq \neg\varphi \vee \psi$,
- $\bot \triangleq p \wedge \neg p$ and
- $\to p \triangleq \neg\bot$.

$\widehat{C}_{i\to j}\varphi$ is read as "agent i considers it possible to commit towards agent j that φ" or also "φ is compatible with what i commitments to towards j".

A path (or computation) $\pi = (s_0, s_1, \ldots)$ is an infinite sequence of reachable global states in S such that for all $i \geqslant 0$, $(s_i, s_{i+1}) \in R_t$. $\pi(k)$ is the kth global state of the path π. A state s' is reachable from a state s iff there exists a path π such that $s = \pi(0)$ and $s' = \pi(k)$, $k \geqslant 0$. We denote this reachability by $s \to s'$. The set of all paths is denoted by Π, whilst Π^{s_i} is the set of all paths starting at the given state ($s_i \in S$).

Definition 3 (*Satisfaction*). Given the model $\mathcal{M}_C$, the satisfaction of a CTLC formula φ in a global state s, denoted by $(\mathcal{M}_C, s) \models \varphi$ is recursively defined as follows:

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

Excluding the commitment and its fulfillment, the semantics of CTLC state formulae is defined in the model $\mathcal{M}_C$ as usual (semantics of CTL) (see for example [11]). The state formula $C_{i\to j}\varphi$ is satisfied in the model $\mathcal{M}_C$ at s iff the content φ holds in every accessible state obtained by the accessibility relation $\sim_{i\to j}$. When i commits towards j that φ, $\sim_{i\to j}$ captures the intuition that for a commitment to take place, a communication channel should exist between the communicating agents (shared variable), and the accessible state s' (where φ holds) is indistinguishable from the current state s for i (which reflects the persistence of i towards his commitment) as i is the agent who is committing; however, for j who is receiving the commitment, the two states are different as new information is obtained from i through the communication channel and this is why in the accessible state, j has the same value as i has for the shared variable (i.e., the content of the communication channel). Furthermore, the accessible state is not completely different from the current state for j as some information is still the same, and this is why for the unshared variables, the current and accessible states for j are indistinguishable (see Fig. 1). As mentioned earlier, there is no relation between the content φ and the local boolean variables (shared and unshared), which are only used to define the accessibility relation.

The state formula $Fu(C_{i\to j}\varphi)$ is satisfied in the model $\mathcal{M}_C$ at s iff there exists a state s' satisfying the commitment (called here the *commitment state*) from which the current state (i.e., s) is "seen" via the accessibility relation $\sim_{i\to j}$. The idea behind this semantics is to say that a commitment is fulfilled when we reach an accessible state from the commitment state. The commitment is fulfilled because its content holds in this accessible state. Unlike the semantics proposed in [5,16,18,17] in which the state s should not only be accessible but also reachable from the commitment state s', in our semantics, the reachability condition is already included in the accessibility relation, which alleviates the semantic conditions. Furthermore, our logic does not include an additional operator for violation as in [53,54,34,5,16,18]; instead violation can be expressed as follows:

$$\neg AG(C_{i\to j}\varphi \supset AF(Fu(C_{i\to j}\varphi))) \equiv EF(C_{i\to j}\varphi \wedge EG(\neg Fu(C_{i\to j}\varphi)))$$

Violation takes place when there is a computation so that in its future a commitment is established but from the moment where the

Please cite this article in press as: J. Bentahar et al., Communicative commitments: Model checking and complexity analysis, Knowl. Based Syst. (2012), http://dx.doi.org/10.1016/j.knosys.2012.04.010

commitment is active there is a possible computation where globally the fulfillment never happens.

Fig. 2 shows an example of commitment made by a merchant (*Mer*) towards a customer (*Cus*) about φ, for example about delivering a given item (see Example 1). At the commitment state s_1, there are three choices available to the merchant, two of them satisfy its commitment and the other does not. Specifically, when the merchant reaches one of the accessible states s_2 or s_3 from the commitment state s_1, the commitment is fulfilled (the commitment content holds at s_2 and s_3), which means two different ways to fulfill the commitment are possible. However, the commitment is not fulfilled at s_4 where the merchant fails to deliver the item. In fact, the computation (s_1, s_4^*), where * means infinite repetition, corresponds to a violation of the commitment as in the future of this path, fulfillment will never take place.

Definition 4 (*Validity*). A formula φ is valid (we write $\models\varphi$) iff $(\mathcal{M}_C, s) \models \varphi$ for all models $\mathcal{M}_C$ of $\mathbb{M}$ and for all s of S.

3.3. Axiomatization

In this section, we investigate the logical properties that the models $\mathcal{M}_C$ of $\mathbb{M}$ inherit from the axiomatic point of view. An immediate consideration comes from the following.

Lemma 1. *$\sim_{i\to j}$ is serial and transitive.*

Proof. $\sim_{i\to j}$ is serial: this follows from the assumption that for any pair $i,j \in \mathcal{A}$, we have that for any $s \in S$, $\sim_{i\to j}(s) \neq \emptyset$ where the notation $\sim_{i\to j}(s)$ denotes the set of accessible states from s, i.e., $\sim_{i\to j}(s) = \{s' \in S | s \sim_{i\to j} s'\}$.

$\sim_{i\to j}$ is transitive: assume $s \sim_{i\to j} s'$ and $s' \sim_{i\to j} s''$, for any pair $i,j \in \mathcal{A}$, according to the definition of $\sim_{i\to j}$, it is the case that $s \sim_{i\to j} s''$ as $l_i(s) = l_i(s') = l_i(s'')$, $\exists! x \in Var_i \cap Var_j$ s.t. $l_i^x(s) = l_i^x(s') = l_j^x(s'')$, $l_j^y(s) = l_j^y(s') = l_j^y(s'') \forall y \in Var_j - Var_i$, and s'' is reachable from s. □

According to this observation, we can immediately conclude that the logic of commitments that deals with agent communication via social commitments is at least as strong as *KD4n* (where n is the number of agents) which is to be expected. This logic is different from the one introduced in our previous work [18], which is *KB5n* (for more details about the standard systems of modal logic, see for example [22]). As our objective in this paper is to investigate the practical problem of model checking commitments for communicating agents, the completeness issue will not be considered, so that the paper is more focused on the algorithmic and implementation aspects of the verification problem (Section 4) and its computational complexity (Section 5). In the following, we only study the individual validities of the proposed modalities with respect to Definition 4. These validities, which we will refer to as the "axioms" of commitments, are:

Proposition 1. (Axioms of commitments).

1. $\models C_{i\to j}(\varphi \supset \psi) \supset (C_{i\to j}\varphi \supset C_{i\to j}\psi)$ *(axiom K)*
2. $\models C_{i\to j}\varphi \supset \widehat{C}_{i\to j}\varphi$ *(axiom D)*
3. $\models C_{i\to j}\varphi \supset C_{i\to j}(C_{i\to j}\varphi)$ *(axiom 4)*

Using the definition of the accessibility relation $\sim_{i\to j}$ and the semantics of $C_{i\to j}\varphi$ and $\widehat{C}_{i\to j}\varphi$, it is easy to check that the above validities hold. In light of the literature in this area (see for instance [35]), the proposed logic should be seen as supporting a *bird's eye view* of the properties of MASs. Therefore, the first validity of axiom *K* should seem relevant and reasonable. It means the commitment is closed under implication, that is if both the implication $\varphi \supset \psi$ and antecedent φ are committed to, then also the conclusion ψ is committed to.

The second validity, which corresponds to serial frames, expresses that if agent i commits towards agent j that φ, then φ is compatible with what i commits to towards j. Consequently, $\models \neg C_{i\to j}\bot$, which means that one cannot honestly, truthfully and justifiably state to commit to something that is false, which guarantees that the commitments of each agent are consistent.

The third axiom expresses a form of introspection and corresponds to transitive frames. It is perhaps not as strong as a first reading might suggest. It should be read and understood that when an agent commits to φ, he should consider the action of committing as a commitment itself. In fact, it would be unreasonable to allow

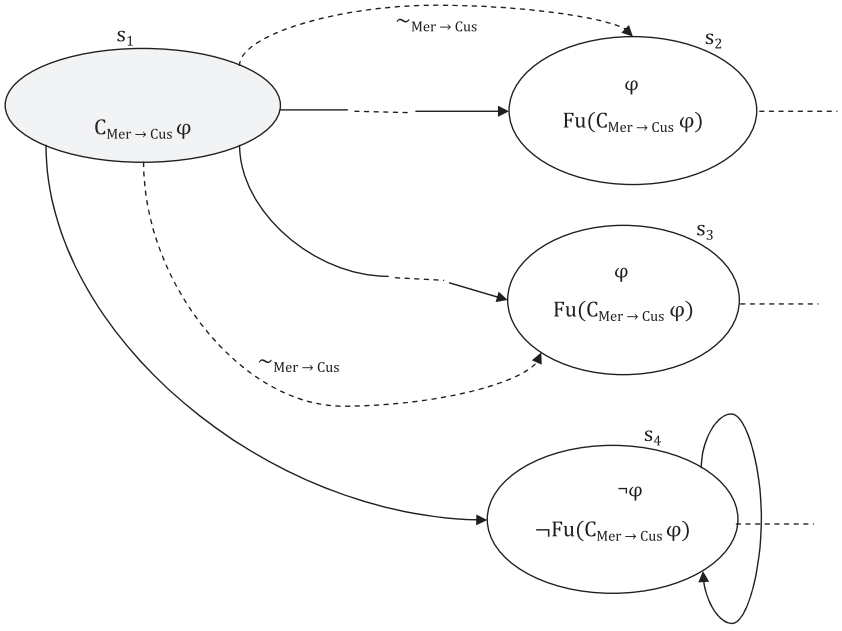

Fig. 2. Illustrative example of commitment, its fulfillment, and its violation.

Please cite this article in press as: J. Bentahar et al., Communicative commitments: Model checking and complexity analysis, Knowl. Based Syst. (2012), http://dx.doi.org/10.1016/j.knosys.2012.04.010

an agent to commit to φ without indicating that he is committing. Let us consider the example of an agent i conveying an information φ to another agent j, it is reasonable to say that i is conveying the information that he is conveying φ. More importantly, this will allow to account for the intentional commitment as the agent is committing that he is committing to φ. The commitment about commitment can be considered as a meta data (or meta information) prescribing a fact of committing as a commitment. It is worth pointing out that *axiom* 4 is desirable and relevant in this context as it encapsulates the fact that agents are aware of their commitments and have full control on what they do commit to.

Proposition 2. (*Axioms of fulfillment*)

1. $\models Fu(C_{i\to j}\varphi) \supset \varphi$ *(axiom T)*
2. $\models C_{i\to j}\varphi \supset C_{i\to j}(Fu(C_{i\to j}\varphi))$
3. $\models C_{i\to j}(Fu(C_{i\to j}\varphi)) \supset C_{i\to j}\varphi$
4. $\models Fu(C_{i\to j}\varphi) \supset C_{i\to j}\varphi$

It is easy to check that the above validities hold using the definition of the accessibility relation $\sim_{i\to j}$ and semantics of $C_{i\to j}\varphi$ and $Fu(C_{i\to j}\varphi)$. The first validity, which corresponds to *axiom T*, expresses that if the commitment is fulfilled, then its content holds, which is very reasonable. The second validity says that if the agent commits to then he commits to fulfill his commitment, which is also a reasonable axiom. The third validity says that if the agent commits to fulfill a commitment about φ, then he is committing about φ, which, in other words, means the commitment is active when the agent commits to satisfy it. This is very reasonable and desirable as an agent cannot commit to satisfy a nonactive (nonexisting) commitment. In fact, putting the second and third validities together shows that committing to something is equivalent to committing to satisfy this commitment, which is very logical. From the semantics of $Fu(C_{i\to j}\varphi)$ and *axiom* 4 of commitments (Proposition 1), we can entail the fourth validity. This validity is important for the logic of communicative commitments as it should be understood as "the commitment should be active when it comes time to its fulfillment", so fulfilling a nonexisting commitment is impossible. Thus, the following proposition follows:

Proposition 3. *The following validity holds:*

$\models AG(\neg E(\neg C_{i\to j}\varphi U \neg C_{i\to j}\varphi \wedge Fu(C_{i\to j}\varphi)))$

Proof. Let us assume the opposite is true, which means:

$EF(E(\neg C_{i\to j}\varphi U \neg C_{i\to j}\varphi \wedge Fu(C_{i\to j}\varphi)))$. According to the semantics of until, $\neg C_{i\to j}\varphi U \neg C_{i\to j}\varphi \wedge Fu(C_{i\to j}\varphi)$ is satisfied in a path if it has $\neg C_{i\to j}\varphi \wedge Fu(C_{i\to j}\varphi)$ in its future. However, from the fourth validity of Proposition 2, this cannot be the case; so the proposition. □

It is important to mention that the commitment $C_{i\to j}\varphi$ in the fourth validity is not a new commitment that is being activated in the current state s but an existing commitment that has been activated in another state s' from which the state s emanates through $\sim_{i\to j}$. So this axiom means the commitment is fulfilled while it is active (see Fig. 3).

In some approaches, for instance [47], instead of the fourth validity, the following axiom (called here *DA*) holds: $Fu(C_{i\to j}\varphi) \supset \neg C_{i\to j}\varphi$. However, *DA* has two problems. The first one is that it forces the fulfillment of commitments to be in the absolute future, so no commitment can be fulfilled in the same state at the moment of its activation. However, for communicative commitments such as those we are considering in this paper, they can be fulfilled in the present, for example when an agent i conveys an information that already holds to an agent j. Thus, with this axiom,

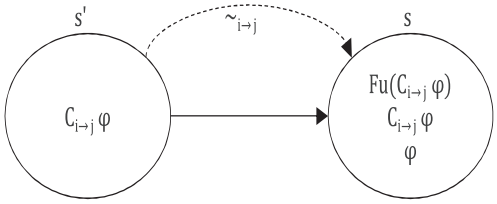

Fig. 3. Link between commitment and its fulfillment where s' is the state of activating the commitment (the state of commitment) and s is the state of fulfilling the commitment (the current state). Notice that the commitment $C_{i\to j}\varphi$ activated in s' is still active in s as the accessibility relation is transitive.

committing about tautologies, atomic propositions and formulae without temporal operators is not supported. The second problem is that commitments can be fulfilled without being activated, which means another axiom (probably with past operators) is needed to prevent fulfilling a commitment which has not been activated or does not exist, or forcing the model to have the following property saying that in all states of every computation, it cannot be the case that a path can have fulfillment in its future without previously having the underlying commitment:

$AG(\neg E(\neg C_{i\to j}\varphi U \neg C_{i\to j}\varphi \wedge Fu(C_{i\to j}\varphi)))$

To summarize, interesting properties are then captured with our logic in terms of connection between commitments and their fulfillment, which are not addressed in the existing commitment logics and cannot be captured with a less strong logic, for instance a classical modal logic. Moreover, CTLC is semantically different from CTL as commitment and fulfillment modalities cannot be expressed with any CTL connector. $C_{i\to j}\varphi$ cannot be expressed by $X\varphi$ because the accessible state is not always the next state; it cannot be expressed by $F\varphi$ either because $C_{i\to j}\varphi \supset F\varphi$, but the reverse is not always true. Thus any formula containing $C_{i\to j}\varphi$ cannot be expressed in CTL.

4. Model checking CTLC formulae

In a nutshell, given a MAS represented as an interpreted system and a formula φ in CTLC describing a property, the model checking problem can be defined as establishing whether or not $\mathcal{M}_C \models \varphi$, i.e., $\forall s \in I : (\mathcal{M}_C, s) \models \varphi$. In this paper, we use symbolic model checking as it alleviates the "state explosion" problem compared to explicit model checking. Symbolic approaches to model checking address this problem by computing the set of states satisfying φ in the model $\mathcal{M}_C$ (denoted by $[\![\varphi]\!]$), which is represented in ordered binary decision diagrams (OBDDs) and then comparing it against the set of initial states I in $\mathcal{M}_C$ that is also represented in OBDD. If $I \subseteq [\![\varphi]\!]$, then the model $\mathcal{M}_C$ satisfies the formula; otherwise a counterexample is generated.

The standard symbolic procedure $SMC(\varphi, \mathcal{M}_C)$ for computing the set of states in $\mathcal{M}_C$ satisfying the formula φ in CTL is introduced in [11]. Given that, in the following we only report the BDD-based algorithms of the new modalities in our logic that extend this procedure.

4.1. BDD-based algorithm of commitment

Algorithm 1 reports the procedure for the social commitment modality, which returns the set of states satisfying $C_{i\to j}\varphi$, i.e., $[\![C_{i\to j}\varphi]\!]$. First, the algorithm computes the set X of states satisfying $\neg\varphi$; then constructs the set Y as a subset of X satisfying the fact that all the shared variables between i and j have the same values; then builds the set Z of states that are reachable and indistinguishable for i from the states in Y and indistinguishable for j from the

Please cite this article in press as: J. Bentahar et al., Communicative commitments: Model checking and complexity analysis, Knowl. Based Syst. (2012), http://dx.doi.org/10.1016/j.knosys.2012.04.010

same states in Y with regard to the unshared variables with i (recall that $s \rightarrow s'$ means that s' is reachable from s). In a nutshell, the set Z includes all the states that can "see" by means of the social accessibility relation $\sim_{i\rightarrow j}$ a state satisfying $\neg\varphi$. Finally, the procedure returns the complement of the set Z. This algorithm is entirely different from Algorithm 8 in our previous work [18], which does not consider shared and unshared variables.

Algorithm 1. $SMC_c(i,j,\varphi,\mathcal{M}_C)$: the set $[\![C_{i\rightarrow j}\varphi]\!]$

1: $X \leftarrow SMC(i,j,\neg\varphi,\mathcal{M}_C)$
2: $Y \leftarrow \{s \in X | l_i^v(s) = l_j^v(s) \forall v \in Var_i \cap Var_j\}$
3: $Z \leftarrow \{s \in S | \exists s' \in Y$ such that $s \rightarrow s'$ and $l_i(s) = l_i(s')$ and $l_j^w(s) = l_j^w(s') \forall w \in Var_j - Var_i\}$
4: return $S - Z$

We can calculate the states satisfying $\widehat{C}_{i\rightarrow j}\varphi$ by calculating the complement of the set of states satisfying $C_{i\rightarrow j}(\neg\varphi)$ using Algorithm 1.

4.2. BDD-based algorithm of fulfillment

The procedure $SMC_{Fu}(i,j,\varphi,\mathcal{M}_C)$ (see Algorithm 2) starts by computing the set X of states satisfying the commitment $C_{i\rightarrow j}\varphi$. It then constructs the set Y of states that are reachable and indistinguishable for i, and indistinguishable for j with regard to unshared variables with i, from the states in X. Afterwards, the algorithm builds the set Z as a subset of the set Y including only states where the shared variables between i and j are the same. Finally, the procedure returns the set Z. Simply put, the algorithm computes and returns the set Z of accessible states that are "seen" by means of the social accessibility relation $\sim_{i\rightarrow j}$ from a state in X.

Algorithm 2. $SMC_{Fu}(i,j,\varphi,\mathcal{M}_C)$: the set $[\![Fu(C_{i\rightarrow j}\varphi)]\!]$

1: $X \leftarrow SMC_c(i,j,\varphi,\mathcal{M}_C)$
2: $Y \leftarrow \{s \in S | \exists s' \in X$ such that $s' \rightarrow s$ and $l_i(s) = l_i(s')$ and $l_j^w(s) = l_j^w(s') \forall w \in Var_j - Var_i\}$
3: $Z \leftarrow \{s \in Y | l_i^v(s) = l_j^v(s) \forall v \in Var_i \cap Var_j\}$
4: return Z

5. Complexity analysis

In this section, we will first analyze the time complexity of model checking CTLC with regard to the size of the explicit model $\mathcal{M}_C$ and length of the formula to be checked. Thereafter, we will analyze the space complexity of model checking CTLC for concurrent programs with respect to the size of the components of these programs and length of the formula.

5.1. Time complexity

In this subsection, we will prove that model checking CTLC is P-complete, so it can be done in polynomial running time in the size of the model and length of the formula.

Theorem 1. *The model checking problem for CTLC can be solved in time $O(|\mathcal{M}_C| \times |\psi|)$ where $|\mathcal{M}_C|$ and $|\psi|$ are the size of the model and length of the formula, respectively.*

Proof. CTLC extends CTL, and it is known from [10] that the model checking problem for CTL is linear in the size of the model and length of the formula. We just need to analyze the time complexity of Algorithms 1 and 2. Steps 2–4 in these two algorithms are simple and it is easy to see that they can be done in linear running time in the size of the model as they are simply constructing sets by performing comparison operations on states. For the reachability test between two states $s \rightarrow s'$, it is also known that the problem can be solved in linear time [48]. Step 1 in Algorithm 1 calls the model checking procedure recursively on the subformula φ of the formula $\psi = C_{i\rightarrow j}\varphi$. The algorithm is recursively called till a CTL subformula is encountered. Thus, the depth of the recursion is bounded by the length of the formula ψ (i.e., linear in the length $|\psi|$). As again model checking CTL is linear in both the size of the model and length of the formula, we conclude that this algorithm has the same complexity. As Algorithm 2 is simply calling Algorithm 1, the result follows. □

Theorem 2. *The model checking problem for CTLC is P-complete.*

Proof. Membership in P (i.e., upper bound) follows from Theorem 1. Hardness in P (i.e., lower bound) follows by a reduction from model checking CTL proved to be P-complete in [44]. □

5.2. Space complexity

In this subsection, we will prove that the complexity of CTLC model checking for concurrent programs is PSPACE-complete. Having discussed the motivation behind the consideration of space complexity for concurrent programs in Section 2, hereafter, we formally introduce these programs and give the proof idea.

Concurrent programs. As introduced in Kupferman et al. [28], a concurrent program Pr is composed of n concurrent processes. Each process P_i is described by a transition system D_i defined as follows: $D_i = (AP_i, AC_i, S_i, \Delta_i, s_i^0, H_i)$ where AP_i is a set of local atomic propositions, AC_i is a local action alphabet, S_i is a finite set of local states, $\Delta_i \subseteq S_i \times AC_i \times S_i$ is a local transition relation, $s_i^0 \in S_i$ is an initial state, and $H_i : S_i \rightarrow 2^{AP_i}$ is a local state labeling function. A concurrent behavior of these processes is obtained by the product of the processes and transition actions that appear in several processes are synchronized by common actions. The joint behavior of the processes P_i can be described using a global transition system D, which is computed by constructing the reachable states of the product of the processes P_i and synchronization is obtained using common action names. Let $AP = \bigcup_{i=1}^n AP_i$, $AC = \bigcup_{i=1}^n AC_i$, $S = \Pi_{i=1}^n S_i$, $s^0 = (s_1^0, s_2^0, \ldots, s_n^0)$, $H(s) = \bigcup_{i=1}^n H_i(s[i])$ for every $s \in S$, and $s[i]$ be the ith component of s. Thus, $D = (AP, AC, S, \Delta, s^0, H)$ where $(s, a, s') \in \Delta$ iff $(s[i], a, s'[i]) \in \Delta_i$ or $s[i] = s'[i]$ for all $1 \leqslant i \leqslant n$.

Proof idea. We will use Generalized CTL* (GCTL*) [8], a logic that extends CTL* by allowing formulae to constrain actions as well as states. First, We prove that model checking GCTL* is PSPACE-complete. Afterwards, by polynomial reduction to GCTL*, we prove that model checking CTLC$^-$, the fragment obtained by removing the fulfillment modality from CTLC, is PSPACE-complete. Then, we provide an algorithm showing that model checking this modality is also a PSPACE-complete problem using the previous result, which will complete the proof.

In what follows, the polynomial reduction is denoted by $\leqslant_p$. We also use disjoint sets Φ_p and Φ_{act} for atomic state and action propositions, respectively.

Definition 5 (*Model of GCTL**). A model $\mathcal{M}_G = (S_G, Ac, l_S, l_{Ac}, \rightarrow, I_G)$ is a tuple where S_G is a set of states; Ac is a set of actions; $l_S : S_G \rightarrow 2^{\Phi_p}$ is the state labeling function; $l_{Ac} : Ac \rightarrow 2^{\Phi_{act}}$ is the action labeling function; $\rightarrow \subseteq S_G \times Ac \times S_G$ is the transition relation; and $I_G \subseteq S_G$ is a set of starting states.

Please cite this article in press as: J. Bentahar et al., Communicative commitments: Model checking and complexity analysis, Knowl. Based Syst. (2012), http://dx.doi.org/10.1016/j.knosys.2012.04.010

Definition 6. (*Syntax of GCTL**)

$\mathcal{S} ::= p|\neg\mathcal{S}|\mathcal{S}\vee\mathcal{S}|E\mathcal{P}$

$\mathcal{P} ::= \theta|\neg\mathcal{P}|\mathcal{S}|\mathcal{P}\vee\mathcal{P}|X\mathcal{P}|\mathcal{P}U\mathcal{P}$

The formulae generated by $\mathcal{S}$ are state formulae, while those generated by $\mathcal{P}$ are path formulae. The state formulae constitute the formulae of GCTL*. The semantics of this logic is given in [8]. Here we only explain the semantics of θ that we will use later in the proof of Theorem 5. A path satisfies θ iff the label of the first transition on the path satisfies θ.

Theorem 3. *Let $|\psi|$ be the size of the state formula ψ and $|\mathcal{M}_G|$ be the size of the model. The model checking problem for GCTL* can be solved in space $O(|\psi|(\log|\mathcal{M}_G|+|\psi|)^2)$.*

Proof. The automata-based model checking procedure for GCTL* proposed in [8] works as follows:

- Step 1: Translating ψ into a variant of alternating tree automaton: and-restricted alternating büchi tableau automaton (arABTA). The size of the resulting automaton is $O(2^{|\psi|})$.
- Step 2: The model checking algorithm works on the product graph of $\mathcal{M}_G$ and arABTA of ψ. The model checking problem is then reduced to the non-emptiness problem of arABTA, which runs in time linear in the size of the product graph, i.e., $O(|\mathcal{M}_G|\times 2^{|\psi|})$.

To complete the proof, we just need to analyze the space complexity of the non-emptiness problem of arABTA. The routine uses two on-the-fly depth-first-searches. The first one recursively searches for true and false leaves which are labeled with Boolean formulas and the second determines if the node given by the first is reachable from itself. On the one hand, it is known from [28] that the first search when it is done in a special type of alternating automata called 1-letter Hesitant Alternating Automata (1-HAA), which are defined over a singleton alphabet and having limited alternation, needs deterministic space $O(|\psi|\log^2 n)$, where n is the size of the automaton 1-HAA. Bhat et al. [8] have proved that arABTA plays the same role as 1-HAA thanks to the and-restricted property according to which the strongly connected components of some states can contain at most one of their children and some states are guaranteed to belong to a different strongly connected component than their children. On the other hand, as shown by Jones [25], the second search (graph accessibility problem) can be done nondeterministically in space $O(\log n)$, or by Savitch's theorem [43] deterministically in space $O(\log^2 n)$. Consequently, the whole routine can be done deterministically in space $O(|\psi|\log^2 n)$. Finally, the model checking problem can be then solved in space $O(|\psi|\log^2(|\mathcal{M}_G|\times 2^{|\psi|}))$, i.e., in space $O(|\psi|(\log|\mathcal{M}_G|+|\psi|)^2)$. □

Theorem 4. *The space complexity of GCTL* model checking for concurrent programs is PSPACE-complete.*

Proof. Membership: As shown in Theorem 3, the model checking problem of GCTL* can be solved in space polynomial in the length of the formula $|\psi|$ and poly-logarithmic in the size of the model $|\mathcal{M}_G|$. Since the size of a concurrent program is independent from the size of the formula, the problem can still be solved in space polynomial in the length $|\psi|$. On the other hand, since the size of a concurrent program is at most exponential with the size of the components, the problem turns out, as shown in [51], to be polynomial in the size of the components. The problem is then in PSPACE.

Hardness: It follows directly from the fact that CTL$^*\leqslant_p$ GCTL* and model checking CTL* is PSPACE-complete for concurrent programs as proved in [28]. □

In what follows, if the underlying language for the satisfaction $\models$ is not clear from the context, we use this language as an index (e.g., $\models_{CTLC^-}$).

Theorem 5. *The complexity of CTLC$^-$ model checking for concurrent programs is PSPACE-complete.*

Proof. Hardness: The hardness follows directly from the fact that CTL$\leqslant_p$ CTLC$^-$ and the problem of model checking CTL is PSPACE-complete for concurrent programs [28].

Membership: We prove that CTLC$^-\leqslant_p$ GCTL*. We note that CTLC and CTLC$^-$ are interpreted in the same model $\mathcal{M}_C$ (Definition 1). We define a transformation f such that: $(\mathcal{M}_C,s)\models_{CTLC^-}\varphi$ *iff* $(f(\mathcal{M}),f(s))\models_{GCTL^*}f(\varphi)$, where φ is a CTLC$^-$ formula and $f(\varphi)$ is the corresponding GCTL* state formula. Let us first define $f(\mathcal{M})$. $f(\mathcal{M})=\mathcal{M}_G=(S_G,Ac,l_S,l_{Ac},\rightarrow,I_G)$ where:

- $S_G=S$; $I_G=I$; $l_S=\mathcal{V}$.
- To define Ac and l_{Ac}, let us first define the set of atomic action propositions $\Phi_{act}=\{\epsilon,\alpha_{1\rightarrow 1},\alpha_{1\rightarrow 2},\ldots,\alpha_{n\rightarrow n}\}$, then $Ac=\{\alpha^o,\alpha^{11},\alpha^{12},\ldots,\alpha^{nn}\}$ such that $l_{Ac}(\alpha^o)=\{\epsilon\}$ and $l_{Ac}(\alpha^{ij})=\{\alpha_{i\rightarrow j}\}$ for $1\leqslant i\leqslant n$ and $1\leqslant j\leqslant n$. α^o is the action labeling temporal transitions defined from the transitions in R_t, and ϵ is the atomic action proposition forming α^o. α^{ij} is the action labeling the transitions defined from the accessibility relation $\alpha_{i\rightarrow j}$ and $\alpha_{i\rightarrow j}$ is the only atomic action proposition forming α^{ij}.
- $(s,\alpha^0,s')\in\rightarrow$ *iff* $(s,s')\in R_t$ and $(s,\alpha^{ij},s')\in\rightarrow$ *iff* $s\sim_{i\rightarrow j}s'$. From this definition of $f(\mathcal{M})$, it is clear that $f(S)=S$.

Let us now define $f(\varphi)$ by induction on the form of φ.

- $f(p)=p$, and $f(\neg\varphi)=\neg f(\varphi)$,
- $f(\varphi\vee\psi)=f(\varphi)\vee f(\psi)$ and $f(EX\varphi)=EXf(\varphi)$,
- $f(E(\varphi U\psi))=E(f(\varphi)U\,f(\psi))$, and $f(EG\varphi)=EGf(\varphi)$,
- $f(C_{i\rightarrow j}\varphi)=A(\alpha_{i\rightarrow j}\rightarrow Xf(\varphi))$
- $f(\widehat{C}_{i\rightarrow j}\varphi)=E(\alpha_{i\rightarrow j}\rightarrow Xf(\varphi))$

The proof of the transformation soundness is by induction on the structure of φ. All the cases are straightforward; we only discuss the case of $C_{i\rightarrow j}\varphi$ (the proof for the case $\widehat{C}_{i\rightarrow j}\varphi$ directly follows from this case). We have $(\mathcal{M}_C,s)\models_{CTLC^-}C_{i\rightarrow j}\varphi$ *iff* $(\mathcal{M}_C,s')\models_{CTLC^-}\varphi$ for every $s'\in S$ such that $s\sim_{i\rightarrow j}s'$. Consequently, $(\mathcal{M}_C,s)\models_{CTLC^-}C_{i\rightarrow j}\varphi$ *iff* $(\mathcal{M}_G,s')\models_{GCTL^*}f(\varphi)$ for every $s'\in S_G$ such that $(s,\alpha^{ij},s')\in\rightarrow$. By semantics of A, X, and θ, we obtain $(\mathcal{M}_G,s)\models_{GCTL^*}A(\alpha_{i\rightarrow j}\rightarrow Xf(\varphi))$. Since it is easy to see that f can be computed in polynomial time, we are done. □

Theorem 6. *The space complexity of CTLC model checking for concurrent programs is PSPACE-complete.*

Proof. Hardness: CTLC is an extension of CTLC$^-$. Hence, CTLC$^-\leqslant_p$ CTLC. So, from Theorem 5 we are done.

Membership: To check if $(\mathcal{M}_C,s)\models Fu(C_{i\rightarrow j}\varphi)$ we use Algorithm 3 that takes as input $\mathcal{M}_C$, s, and φ. The algorithm is a straightforward translation of the semantics of $Fu(C_{i\rightarrow j}\varphi)$ (Definition 3), and it is easy to check that $(\mathcal{M}_C,s)\models Fu(C_{i\rightarrow j}\varphi)$ *iff* the algorithm return YES. The algorithm uses an oracle to check if $s'\models C_{i\rightarrow j}\varphi$, which is PSPACE-complete as proved in Theorem 5. Consequently, the space complexity of this algorithm has the form CM$^{\text{PC}}$ where CM is the complexity of checking if $s'\sim_{i\rightarrow j}s$ and PC is a PSPACE-complete problem [24]. To check if $s'\sim_{i\rightarrow j}s$, only three states are remem-

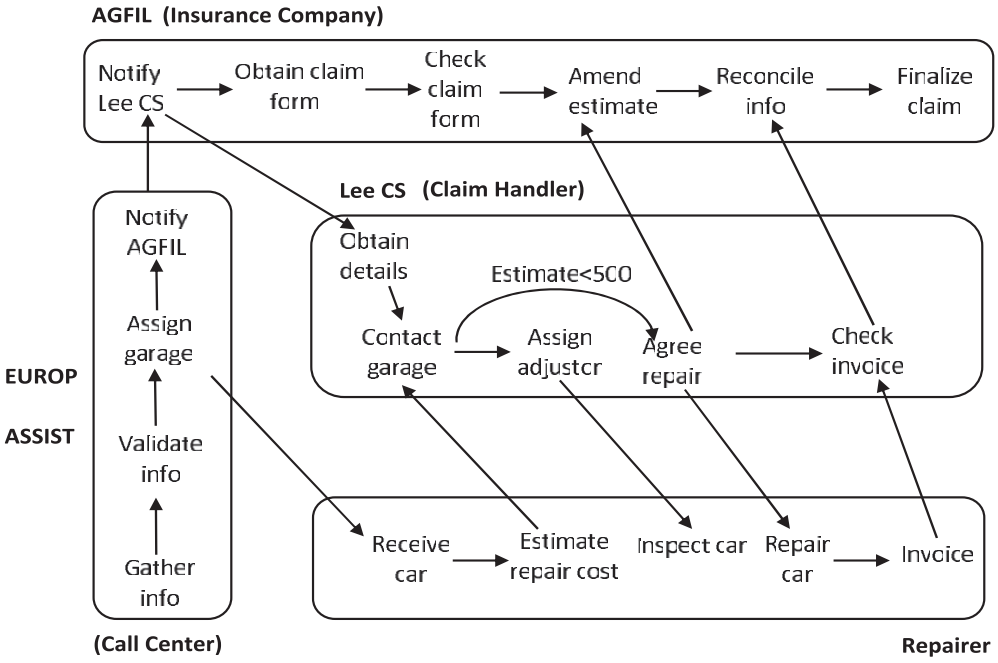

Fig. 4. Insurance claim processing [49].

bered (because accessibility includes reachability, which needs three states to be remembered [51]). Therefore, this verification can be done nondeterministically in logarithmic space, that is NLOGSPACE. Replacing MC by NLOGSPACE, the complexity of the algorithm is NLOGSPACE^{PC}. Finally, the membership in PSPACE follows from the following facts:

1. NLOGSPACE$^{PC} \subseteq$ P^{PC} [24]; and
2. P^{PC} = PSPACE because:
   - P$^{PC} \subseteq$ PSPACE because a PSPACE machine can easily simulate a P machine, and can simulate a PSPACE-complete oracle [24]; and
   - PSPACE $\subseteq$ P^{PC} by using a Cook reduction [12] from any language L $\in$ PSPACE to the PSPACE-complete problem PC. □

Algorithm 3. Checking if $(\mathcal{M}_c, s) \models Fu(C_{i\to j}\varphi)$

```
Repeat
{
  Nondeterministically guess a state s'
    if s' ~_{i→j} s then
      if s' ⊨ C_{i→j}φ then
        return YES
}
Until no state can be guessed
Return NO
```

6. Implementation

One of the main objectives of this paper is to implement the model checking algorithm of social commitments. Moreover, we aim to examine various social properties of multi-agent systems when agents commit towards each other and to determine which of these properties are retained and which are compromised when agents violate their commitments.

We implemented[5] the model checking technique presented in Section 4 on top of the MCMAS symbolic model checker. Furthermore, we extended the implementation of interpreted systems to include shared and unshared variables and the social accessibility relation. We chose MCMAS as it supports the semantics of interpreted systems and CTL and performs OBDD operations by means of the efficient CUDD library.

One example for which we have been able to carry out the above objectives is the *Insurance Claim Processing* (an industrial case study), already applied to show how commitments can specify protocols in business settings [49].

6.1. Case study: insurance claim processing

This industrial case study outlines the manner motor damage claims of policy holders are handled by AGFIL, a private insurance company in Ireland (see Fig. 4). To deal with a motor damage claim, the AGFIL company is required to provide claim reception and car repair to its policy holders. It also needs to assess (or adjust) claims to protect itself against fraud. AGFIL depends on its associates, Europ Assist, Lee Consulting Service (Lee CS), and various repairers when executing these tasks. Europ Assist through the call center offers a 24-h emergency call answering service to policy holders for reporting claims and provides them with the name of an approved repairer facility. Lee CS coordinates with AGFIL and deals with repairers and assessors to handle the claims. A network of approved repairers provide the repair services. AGFIL holds ultimate control in deciding if a given claim is valid and if payment will be made to the repairer.

A typical business scenario is as follows: The policy holder phones the call center to notify the company of a new claim. The call center will gather the information using an incident form, validate the information with its database, then assign the nearest garage and give some suggestions about approved repairs to the policy holder. The incident form is then sent to AGFIL and Lee CS replicates the AGFIL database to receive notification details. The

[5] http://users.encs.concordia.ca/bentahar/MCMAS-C/mcmas-sc.zip and http://users.encs.concordia.ca/bentahar/MCMAS-C/Case-Study.zip.

Please cite this article in press as: J. Bentahar et al., Communicative commitments: Model checking and complexity analysis, Knowl. Based Syst. (2012), http://dx.doi.org/10.1016/j.knosys.2012.04.010

AGFIL claim handler will check to confirm the cover. In the case of the claim being invalid, Lee CS will be contacted and the process will be stopped. Lee CS will agree upon repair figures if an adjuster is not required or will otherwise either appoint an assessor or conduct the assessment itself. When the repairs are completed, the repairer will issue an invoice to the adjuster who will check the invoice against the original estimate. Lee CS sends a monthly bordereaux to AGFIL listing all the repairs for that month. More details about the case study can be found in [49].

6.1.1. Specifications

To verify the correctness of the above scenario at design time, various properties can be formalized using CTLC w.r.t the model $\mathcal{M}_C$.

`Reachability property`. A particular situation can be reached from the initial state via some computation sequences. The following formula means that there exists a path where the call center will not commit to the policy holder for gathering claim information until it receives the accident report from the policy holder.

$$E(\neg\ reportAccident\ U\ (reportAccident \wedge C_{CallCenter \to PolicyHoldr}\ gatherInfo))$$

`Safety property`. This property means "something bad never happens". For example, a bad situation is: the policy holder's claim is validated by the call center, but the repairer never commits to repair the vehicle:

$$AG\neg(validClaim \wedge \neg C_{Repairer \to PolicyHolder}\ carRepair)$$

`Liveness property`. This property states "something good will eventually happen". For example, in all paths globally if the policy holder reports an accident and his claim is valid, then in all future computations, the call center will commit to assign the garage:

$$AG(reportAccident \wedge validClaim \to AF(C_{CallCenter \to PolicyHolder}\ assignGarage))$$

Moreover, the following formulae are some examples of commitments (1.a, 2.a, 3.a, and 4.a), their fulfillment (1.b, 2.b, 3.b, and 4.b), and violations (1.c, 2.c, 3.c, and 4.c) and (1.d, 2.d, 3.d, and 4.d):

1. Commitment 1
   (a) $EF(C_{PolicyHolder \to AGFIL}\ insurancePayment)$
   (b) $EF(Fu(C_{PolicyHolder \to AGFIL}\ insurancePayment))$
   (c) $\neg AG(C_{PolicyHolder \to AGFIL}\ insurancePayment \supset AF(Fu(C_{PolicyHolder \to AGFIL}\ insurancePayment)))$
   (d) $\neg AG(C_{PolicyHolder \to AGFIL}\ insurancePayment \supset EF(Fu(C_{PolicyHolder \to AGFIL}\ insurancePayment)))$
2. Commitment 2
   (a) $EF(C_{AGFIL \to CallCenter}\ receptionPayment)$
   (b) $EF(Fu(C_{AGFIL \to CallCenter}\ receptionPayment))$
   (c) $\neg AG(C_{AGFIL \to CallCenter}\ receptionPayment \supset AF(Fu(C_{AGFIL \to CallCenter}\ receptionPayment)))$
   (d) $\neg AG(C_{AGFIL \to CallCenter}\ receptionPayment \supset EF(Fu(C_{AGFIL \to CallCenter}\ receptionPayment)))$
3. Commitment 3
   (a) $EF(C_{CallCenter \to PolicyHolder}\ assignGarage)$
   (b) $EF(Fu(C_{CallCenter \to PolicyHolder}\ assignGarage))$
   (c) $\neg AG(C_{CallCenter \to PolicyHolder}\ assignGarage \supset AF(Fu(C_{CallCenter \to PolicyHolder}\ assignGarage)))$
   (d) $\neg AG(C_{CallCenter \to PolicyHolder}\ assignGarage \supset EF(Fu(C_{CallCenter \to PolicyHolder}\ assignGarage)))$
4. Commitment 4
   (a) $EF(C_{Repairer \to PolicyHolder}\ carRepair)$
   (b) $EF(Fu(C_{Repairer \to PolicyHolder}\ carRepair))$
   (c) $\neg AG(C_{Repairer \to PolicyHolder}\ carRepair \supset AF(Fu(C_{Repairer \to PolicyHolder}\ carRepair)))$
   (d) $\neg AG(C_{Repairer \to PolicyHolder}\ carRepair \supset EF(Fu(C_{Repairer \to PolicyHolder}\ carRepair)))$

The first formula (1.a) expresses the existence of a computation so that in its future the policy holder commits towards the insurance company AGFIL to pay the insurance and the second formula (1.b) states that such a commitment will eventually be fulfilled, while the third (1.c) and fourth (1.d) formulae are checking violations of this commitment. The formula (1.c) says that there is a computation so that in its future the commitment explained in (1.a) is established but from the moment where the commitment is active there is a possible computation where globally the fulfillment never happens, that is the insurance never get paid. The formula (1.d) expresses that after having the commitment from the policy holder towards the insurance company, the fulfillment does not occur in all states of every possible computation, which is always false because accessible states are reachable.

The other formulae in Commitments 2–4 can be explained in a similar way. Commitment 2 is from the insurance company AGFIL that commits towards the call center to receive the payment (as compensation for the call service it offers). Commitment 3 is from the call center that commits towards the policy holder that a garage will be assigned. In Commitment 4, the repairer commits to the policy holder to repair his car.

6.2. Experimental results

According to the above scenario, we have six agents: AGFIL, PolicyHolder, CallCenter, Repairer, Assessor and Inspector. We encoded our model $\mathcal{M}_C$ by considering these agents, their set of local states and actions, the set of shared and unshared variables, local protocols, local evolution functions, the valuation function and finally the set of initial states. For example, the PolicyHolder agent has 29 local states, 3 boolean variables (communication channels), 21 local actions and four initial states. The complete encoding is available for download (see Footnote 5).

Our experiments were performed on an AMD Phenom (tm) 9600 B Quad-Core Processor with 8 GB memory running Fedora 12 x86_64 Linux. We found that 28 tested formulae (commitments and their fulfillment) hold, 28 tested formulae return false (violation) and 3 tested formulae hold (reachability, safety and liveness). To check the effectiveness of the proposed symbolic algorithm, we reported 8 experiments in Table 1 where the number of agents (#Agent), the number of reachable states (#States), the execution time (Time) in seconds (s), and the memory in use (Memory) in MB are given. From Table 1, we found that the number of reachable states (which reflects state space) increases exponentially when the number of agents increases as we expected. However, the memory usage increases only polynomially, which confirms the theoretical results (i.e., the PSPACE-completeness). Regarding the execution time, the increase is not exponential, but faster than the polynomial, which also confirms the time complexity in concurrent programs, which is APTIME as APTIME = PSPACE

Table 1
Verification results.

# Agents	# States	Time (s)	Memory (MB)
6	41	1	11
12	1669	11	22
18	67,529	26	45
24	2.71806e + 06	49	47
30	1.08909e + 08	278	47
36	4.34657e + 09	458	44
42	1.72867e + 11	5219	139
48	6.85373e + 12	14,477	132

Please cite this article in press as: J. Bentahar et al., Communicative commitments: Model checking and complexity analysis, Knowl. Based Syst. (2012), http://dx.doi.org/10.1016/j.knosys.2012.04.010

(Johnson1990), where APTIME is the class of languages accepted by polynomial-time alternating Turing machines.

Because our model is verified successfully, we consider an imaginary condition under which the commitment violation can occur to further demonstrate our approach. For example, when the policy holder fails to send the agreed amount of insurance payment to the AGFIL company, his commitment is violated. Thus, MCMAS counterexample shows an execution in which there is no way to reach the insurance payment from the policy holder's commitment state.

7. Discussion

7.1. Commitment formalisms and interpreted systems

In the literature about commitment representation, excluding [18,17], abstract models are used to model the interacting agents from high-level perspective without considering agents' states that constitute the life cycle of each agent, agents's actions and how agents can select their choices at each state. In this paper, all the above concepts are explored and the interacting agents are modeled using the formalism of interpreted systems. Moreover, the work presented here is the second one that directly and automatically verifies commitments and their fulfillment by developing and implementing a new model checking algorithm on top of MCMAS. In this section, we show the main differences between this work and our previous work presented in [18] especially in the definition of the accessibility relation and semantics of fulfillment. In fact, while [18] shows that the model checking of commitment logic is feasible, in this paper we focused on efficiency considerations and complexity results. The social accessibility defined in [18] is as follows:

$$s \sim_{i \to j} s' \text{ iff there exists } s'' \neq s \text{ such that } : (1) l_i(s) = l_i(s'') = l_i(s'); \text{ and } (2) l_j(s'') = l_j(s').$$

which means i is uncertain about the current state so that he looks for an intermediate state different from the current state (i.e., $s'' \neq s$) in which there is no difference for i among being in s, s'' and s'; however, for j there is no difference between being in s'' and s' (i.e., accessible state). It is obvious that with this definition, there is no communication established between interacting agents, which is the main concept we are capturing in this paper through communicative commitments. In addition, introducing new intermediate states makes the computation of accessible states more complex. The semantics of fulfillment in [18] is defined as follows:

$$(\mathcal{M}_C, s) \models Fu(C_{i \to j}\varphi) \text{ iff there exists } s' \text{ such that } : (\mathrm{i})(\mathcal{M}_C, s') \models C_{i \to j}\varphi; \text{ and } (\mathrm{ii})\, s \in \mathcal{F}ut(s'); \text{and } (\mathrm{iii}) s' \sim_{i \to j} s.$$

The intuition behind *Fu*'s semantics is to ensure that the current state s is reachable in terms of transitions ($\mathcal{F}ut(s')$) and accessible in terms of the social accessibility relation from the state s' where the commitment holds. As we mentioned in Section 3 the reachability condition is already included in the definition of accessibility relation, which makes the semantics simpler as we do not need the function $\mathcal{F}ut(s')$ that gives the future states of s'.

In the rest of this section, we compare our approach with the relevant work related to (1) verification of commitment-based protocols; (2) recent extension of interpreted systems formalism; and (3) some practical and industrial applications of knowledge-based systems.

El-Menshawy et al. [17] presented an extension of CTL with commitment modality to formally specify commitment-based protocols. In order to automatically verify these protocols, they reduced the problem of model checking this logic into the problem of model checking either CTLK (an extension of CTL with knowledge modality) [38] to directly use the MCMAS tool, or ARCTL (an extension of CTL with action formulae) [38] to use the extended version of NuSMV. Because this approach is translation-driven, it differs from the model checking technique presented in this paper. The differences with the other approaches can also be discussed at the level of the paper motivation.

Lomuscio and Sergot [30] extended the formalism of interpreted systems to model the "correct and incorrect functioning behaviors of the agents" by dividing global states into "green" and "red" states. The resulting logic of deontic interpreted systems is stronger than the standard deontic logic *KD*. They also presented the axioms of this logic and proved theoretically its soundness and completeness. The semantics of correct functioning behavior is defined using a new accessibility relation R_i for each agent i on the global states as follows: for any $i \in \mathcal{A}, (l_1, \ldots, l_n) R_i (l'_1, \ldots, l'_n)$ if $l'_i \in G_i$ (the set of green states for i). This deontic accessibility relation defines accessible states (green states) independently from the current state. From this deontic perspective, there are some similarities between this logic and the logic of commitments. However, the deontic accessibility relation is different from our accessibility relation, which considers both current and accessible states. Furthermore, our accessibility relation accounts for communication between the participating agents by considering the communication channels. It is also unclear how in the deontic interpreted systems the green and red states are determined for each agent. The authors also defined a modality $\widehat{K}_i^j \varphi$, called "doubly-indexed operator", which can be read as "agent i knows φ under the assumption that agent j is functioning correctly". This modality attempts to combine deontic and epistemic modalities to specify what an agent is permitted to know. It indirectly captures the interaction between agents, but it differs from our commitment modality $C_{i \to j}\varphi$ as i does not assume anything about j and he has the possibility of fulfilling his commitment (i.e., correct functioning behavior) or violating it (i.e., incorrect functioning behavior), but in both cases, the commitment's content holds in all accessible states, and the commitment is fulfilled if the agent can be in one of these states, otherwise it is violated. Thus, reasoning about social commitments is different from reasoning about doubly-indexed operator.

7.2. Practical and industrial settings

We turn towards elaborating on the contribution of our work in practical and industrial settings. One of the relevant applications is Intelligent Tutoring Systems (ITSs), which depend on domain experts that provide relevant domain knowledge to the tutors to be able to guide and help learners during problem-solving. The authors in [37] propose a framework that combines two knowledge discovery techniques: sequential patterns mining and association rules discovery. The framework then uses such techniques to discover new knowledge and rules from the recorded usage of experts and new users. The resulting knowledge base can be used as a problem space or task model instead of building a complete task description to avoid the "ill-defined" problem resulting from the complexity of some domains. The knowledge base allows the intelligent tutor to track learners' actions. Our approach develops an effective model checking that can be used to verify domains having a large state space to detect errors and generate counter-examples that guide designers to modify the problem-solving task model. In fact, this approach can be used to check if the used problem-solving task model satisfies some domain-specific properties and if the existing and learned rules satisfy consistency properties.

The authors in [31] propose a new knowledge representation approach based on states where domain-specific knowledge is expressed by combinations of the relevant objects' states. Such an approach may vary when states of those objects change with time as in emergency management and warning systems. Based on this

Please cite this article in press as: J. Bentahar et al., Communicative commitments: Model checking and complexity analysis, Knowl. Based Syst. (2012), http://dx.doi.org/10.1016/j.knosys.2012.04.010

approach, two levels of logical inconsistencies are conducted: syntactic level and semantic level. On the syntactic level, since each piece of domain knowledge is represented by some state combinations of objects, a logical relationship between knowledge is used for detecting the strict and partial logical consistency of the domain knowledge base. On the semantic level, the "information logical inconsistency detection (ILID)" method is used to detect the logical inconsistency in the stored knowledge base. Unlike the logical consistency at the syntactic and semantic levels used in this paper, our logical consistency deduction method is mainly based on verifying the system model against properties since two inconsistent properties (i.e., facts or knowledge) cannot be both satisfied.

In [41], the authors introduce a diagnostic framework to study the consequences of using inaccurate values in "Model-Based Diagnosis (MBD)". The MBD specifies the normal and abnormal behaviors of the system to be diagnosed. Technically, the system is described in first order logic in which the probability of a proposition regarding to a set of interpretations satisfying the proposition is defined and then consistency-based reasoning is used to identify possible "maximal-confirmation diagnoses" in a polynomial time. Our work can contribute to the area of MBD as the simulations produced by our model checker can be seen as a diagnostic approach since it helps identify problematic computations and abnormal behaviors.

The authors in [27] define four strategies for measuring level of trust based on a trust propagation model in social networks. They evaluate the prediction accuracy of those strategies in terms of the length of reliable trust paths and different aggregation methods in order to discover the best strategy having maximum predication accuracy. Their strategies are implemented by using a "Reinforcement Learning" algorithm. In [42], the importance of trust in competitive settings is stressed and a trust model in such settings is proposed by combining reliability and reputation. The author determines some criteria to help choose and dynamically adapt the weights that should be assigned to reliability and reputation. In [26], the authors develop a multi-factor trust framework using a number of measurements to evaluate the trust of interacting agents. The framework considers direct interactions among agents, called online trust estimation, and post evaluation of the actual performance of an agent, called off-line process. The post evaluation is then compared against the information provided by some consulting agents to both adjust the credibility of the contributing agents in trust evaluation and improve the system trust evaluation by minimizing the trust estimation error. Our work can contribute to the area of trust by analyzing the trust behavior of agents in terms of the level of satisfaction and violation of their commitments, which provides a measurement of this trust. This can be done by discovering the trust path from a commitment state to a fulfillment state using, for example, the reachability property.

In [36], the authors present a general model for controlling the behavior of autonomous robots/agents that can engage in a conversational dialog with humans. The core components of this model are called "experts", which are responsible for understanding humans' requests (in general, humans' utterances), providing useful information, and deciding on robots actions. Simple methods are incorporated in the interface of each expert to determine its correctness with the received user utterances. Our approach can complement this model by formalizing the communication between robots and humans and automatically verifying users and robots actions in accordance with given properties by using our model checking technique.

The authors in [15] propose an approach to coordinate the course of agents' actions operating in the same environment in two different settings: when agents by themselves are able to achieve their individual goals and when agents need other agents to reach their individual goals. The authors extend the classical SATPLAN planner to deal with the problem of multi-agent planning by handling negative (i.e., harmful) and positive interactions in order to finally discover consistent plans for multiple agents. The concept of commitments can be used in this approach to model flexible interactions among agents where positive and negative interactions can be captured in terms of satisfaction and violation of agents' commitments. The formalism of interpreted systems can be used to coordinate synchronous agents' actions working in the same environment from a global perspective. By so doing, our model checking approach can be used to check the consistency of plans.

8. Conclusion

Interpreted systems provide a useful formalism for representing agent commitments. In this paper, we refined CTLC, a new logic for social commitments and their fulfillment introduced in [18] and investigated its axiomatization. In order to verify commitments, instead of translating them into the input language of an existing model checker, we developed a new BDD-based model checking algorithm that extends MCMAS. We proved that the time complexity of model checking CTLC is P-complete with regard to the explicit models and its space complexity is PSPACE-complete for concurrent programs. In our implementation, we used a widely studied industrial case study and conducted eight experiments with large state-space (approximately 6.85373e+12 states), which demonstrated the effectiveness and applicability of our approach in terms of execution time and memory in use. There are many directions for future work. We plan to theoretically prove the completeness, soundness and decidability of our logic to complement our practical work on model checking. We also plan to extend the logic and its model checking to consider conditional commitments and other commitment actions such as assign and delegate. Finally, we plan to extend our model checker with graphical notation for a user-friendly specification using a similar technique as the one proposed in [40], which uses an XML-based transformation system.

Acknowledgements

We thank the four anonymous reviewers for their valuable comments and suggestions for improvements. We also would like to thank Prof. Orna Kupferman for her highly helpful explanations, suggestions, and comments regarding the complexity of model checking for concurrent programs. Our thanks go also to NSERC, SSHRC, and FQRSC for their financial supports.

References

[1] M. Alberti, D. Daolio, P. Torroni, M. Gavanelli, E. Lamma, P. Mello, Specification and verification of agent interaction protocols in a logic-based system, in: H. Haddad, A. Omicini, R.L. Wainwright, L.M. Liebrock (Eds.), SAC, ACM, 2004, pp. 72–78.

[2] M. Baldoni, C. Baroglio, E. Marengo, Behavior oriented commitment-based protocols, in: H. Coelho, R. Studer, M. Wooldridge (Eds.), ECAI, vol. 215, IOS Press, 2010, pp. 137–142.

[3] J. Bentahar, Z. Maamar, W. Wan, D. Benslimane, P. Thiran, S. Subramanian, Agent-based communities of web services: an argumentation-driven approach, Service Oriented Computing and Applications 2 (4) (2008) 219–238.

[4] J. Bentahar, J.-J. Meyer, W. Wan, Model checking communicative agent-based systems, Knowledge-Based Systems 22 (3) (2009) 142–159.

[5] J. Bentahar, J.-J.C. Meyer, W. Wan, Model checking agent communication, in: M. Dastani, K.V. Hindriks, J.-J.C. Meyer (Eds.), Specification and Verification of Multi-Agent Systems, first ed., Springer, 2010, pp. 67–102 (chapter 3).

[6] J. Bentahar, B. Moulin, J.-J.C. Meyer, B. Chaib-draa, A logical model for commitment and argument network for agent communication, in: Proceedings of the 3rd International Joint Conference on AAMAS, IEEE Computer Society, 2004, pp. 792–799.

[7] J. Bentahar, B. Moulin, J.-J.C. Meyer, Y. Lespérance, A new logical semantics for agent communication, in: K. Inoue, K. Satoh, F. Toni (Eds.), CLIMA VII, LNCS, vol. 4371, Springer, 2007, pp. 151–170.
[8] G. Bhat, R. Cleaveland, A. Groce, Efficient model checking via Büchi Tableau automata, in: G. Berry, H. Comon, A. Finkel (Eds.), CAV, LNCS, vol. 2102, Springer, 2001, pp. 38–52.
[9] F. Chesani, P. Mello, M. Montali, P. Torroni, Commitment tracking via the reactive event calculus, in: Proceedings of IJCAI, 2009, pp. 91–96.
[10] E.M. Clarke, E.A. Emerson, A.P. Sistla, Automatic verification of finite-state concurrent systems using temporal logic specifications, ACM Transactions on Programming Languages and Systems 8 (2) (1986) 244–263.
[11] E.M. Clarke, O. Grumberg, D.A. Peled, Model Checking, The MIT Press, Cambridge, Massachusetts, 1999.
[12] S.A. Cook, The complexity of theorem-proving procedures, in: M.A. Harrison, R.B. Banerji, J.D. Ullman (Eds.), Proceedings of the Third Annual ACM Symposium on Theory of Computing (STOC '71), ACM, New York, 1971, pp. 151–158.
[13] N. Desai, Z. Cheng, A.K. Chopra, M. Singh, Toward verification of commitment protocols and their compositions, in: E.H. Durfee, M. Yokoo, M.N. Huhns, O. Shehory, (Eds.), AAMAS. IFAAMAS, 2007, pp. 144–146.
[14] N. Desai, A.K. Chopra, M.P. Singh, Amoeba: a methodology for modeling and evolution of cross-organizational business processes, ACM Transaction on Software Engineering and Methodology 19 (2) (2009) 1–40.
[15] Y. Dimopoulos, M.A. Hashmi, P. Moraitis, μ-SATPLAN: multi-agent planning as satisfiability, Knowledge-Based Systems 29 (2012) 54–62.
[16] M. El-Menshawy, J. Bentahar, R. Dssouli, Verifiable semantic model for agent interactions using social commitments, in: M. Dastani, A.E. Fallah-Seghrouchni, J. Leite, P. Torroni, (Eds.), LADS, LNCS, vol. 6039, 2010, pp. 128–152.
[17] M. El-Menshawy, J. Bentahar, R. Dssouli, Symbolic model checking commitment protocols using reduction, in: A. Omicini, S. Sardina, W. Vasconcelos (Eds.), DALT, LNAI, vol. 6619, Springer, 2011, pp. 185–203.
[18] M. El-Menshawy, J. Benthar, H. Qu, R. Dssouli, On the verification of social commitments and time, in: Proceedings of the 10th International Conference on AAMAS, 2011b, pp. 483–890.
[19] R. Fagin, J.Y. Halpern, Y. Moses, M.Y. Vardi, Reasoning About Knowledge, The MIT Press, Cambridge, MA, 1995.
[20] N. Fornara, F. Viganò, M. Verdicchio, M. Colombetti, Artificial institutions: a model of institutional reality for open multi-agent systems, AI and Law 16 (1) (2008) 89–105.
[21] S.N. Gerard, M.P. Singh, Protocol refinement: formalization and verification, in: A. Artikis, J. Bentahar, A.K. Chopra, F. Dignum, (Eds.), AAMAS Workshop on Agent Communication (AC), 2010, pp. 19–36.
[22] G. Hughes, M. Cresswell, A New Introduction to Modal Logic, Routledge, London, 1996.
[23] W. Jamroga, T. Ågotnes, Modular Interpreted Systems, in: Proceedings of the 6th International Conference on AAMAS, ACM, 2007, pp. 131:1–131:8.
[24] D.S. Johnson, A catalog of complexity classes, in: J. van Leeuwen (Ed.), Handbook of Theoretical Computer Science, Algorithms and Complexity, vol. A, Elsevier and MIT Press, 1990, pp. 67–161.
[25] N.D. Jones, Space-bounded reducibility among combinatorial problems, Journal of Computer and System Sciences 11 (1) (1975) 68–85.
[26] B. Khosravifar, J. Bentahar, M. Gomrokchi, R. Alam, CRM: an efficient trust and reputation model for agent computing, Knowledge-Based Systems (2011) (On line first) <http://dx.doi.org/10.1016/j.knosys.2011.01.004>.
[27] Y.A. Kim, H.S. Song, Strategies for predicting local trust based on trust propagation in social networks, Knowledge-Based Systems 24 (8) (2011) 1360–1371.
[28] O. Kupferman, M. Vardi, P. Wolper, An automata-theoretic approach to branching-time model checking, Journal of the ACM 47 (2) (2000) 312–360.
[29] A. Lomuscio, H. Qu, F. Raimondi, MCMAS: a model checker for the verification of multi-agent systems, in: A. Bouajjani, O. Maler (Eds.), CAV, LNCS, vol. 5643, Springer, 2009, pp. 682–688.
[30] A. Lomuscio, M.J. Sergot, Deontic interpreted systems, Studia Logica 75 (1) (2003) 63–92.
[31] J. Ma, G. Zhang, J. Lu, A state-based knowledge representation approach for information logical inconsistency detection in warning systems, Knowledge-Based Systems 23 (2) (2011) 125–131.
[32] A.U. Mallya, M.N. Huhns, Commitments among agents, IEEE Internet Computing 7 (4) (2003) 90–93.
[33] A.U. Mallya, M.P. Singh, An algebra for commitment protocols, Autonomous Agents and Multi-Agent Systems 14 (2) (2007) 143–163.
[34] A.U. Mallya, P. Yolum, M.P. Singh, Resolving commitments among autonomous agents, in: F. Dignum (Ed.), ACL'03, LNCS, vol. 2922, Springer, 2004, pp. 166–182.
[35] J.-J. Meyer, F. Veltman, Intelligent agents and common sense reasoning, in: P. Blackburn, J. van Benthem, F. Wolter (Eds.), Handbook of Modal Logic, vol. 3, Elsevier, 2007, pp. 991–1029 (chapter 18).
[36] M. Nakano, Y. Hasegawa, K. Funakoshi, J. Takeuchi, T. Torii, K. Nakadai, N. Kanda, K. Komatani, H.G. Okuno, H. Tsujino, A multi-expert model for dialogue and behavior control of conversational robots and agents, Knowledge-Based Systems 24 (2) (2011) 248–256.
[37] R. Nkambou, P. Fournier-Viger, E.M. Nguifo, Learning task models in ill-defined domain using an hybrid knowledge discovery framework, Knowledge-Based Systems 24 (1) (2011) 176–185.
[38] W. Penczek, A. Lomuscio, Verifying epistemic properties of multi-agent systems via bounded model checking, Fundamenta Informaticae 55 (2) (2003) 167–185.
[39] D.Q. Pham, J. Harland, Temporal linear logics as a basis for flexible agent interactions, in: E.H. Durfee, M. Yokoo, M.N. Huhns, O. Shehory, (Eds.), Proceedings of the 6th International Joint Conference on AAMAS, IFAAMAS, 2007, pp. 124–131.
[40] E. Pulvermueller, S. Feja, A. Speck, Developer-friendly verification of process-based systems, Knowledge-Based Systems 23 (7) (2010) 667–676.
[41] N. Roos, Maximal-confirmation diagnoses, Knowledge-Based Systems 24 (2011) 467–477.
[42] D. Rosaci, Trust measures for competitive agents, Knowledge-Based Systems 28 (2012) 38–46.
[43] W.J. Savitch, Relationships between nondeterministic and deterministic tape complexities, Journal of Computer and System Sciences 4 (2) (1970) 177–192.
[44] P. Schnoebelen, Advances in modal logic, vol. 4. King's College Publications, 2003, pp. 393–436 (chapter, The Complexity of Temporal Logic Model Checking).
[45] M.P. Singh, An ontology for commitments in multiagent systems: toward a unification of normative concepts, AI and Law 7 (1) (1999) 97–113.
[46] M.P. Singh, A social semantics for agent communication languages, in: F. Dignum, M. Greaves (Eds.), Issues in Agent Communication, LNCS, vol. 1916, Springer, 2000, pp. 31–45.
[47] M.P. Singh, Semantical considerations on dialectical and practical commitments, in: D. Fox, C.P. Gomes (Eds.), AAAI, AAAI Press, 2008, pp. 176–181.
[48] R. Tarjan, Depth first search and linear graph algorithms, SIAM Journal of Computing 1 (2) (1972) 146–160.
[49] P. Telang, M. Singh, Enhancing tropos with commitments: a business meta-model and methodology, in: A. Borgida, V.K. Chaudhri, P. Giorgini, E.S.K. Yu (Eds.), Conceptual Modeling: Foundations and Applications, LNCS, vol. 5600, Springer, 2009, pp. 417–435.
[50] P.R. Telang, M.P. Singh, Specifying and verifying cross-organizational business models: an agent-oriented approach, IEEE Transactions on Services Computing 4 (2011), http://dx.doi.org/10.1109/TSC.2011.4.
[51] M.Y. Vardi, P. Wolper, Reasoning about infinite computations, Information and Computation 115 (1) (1994) 1–37.
[52] M. Venkatraman, M.P. Singh, Verifying compliance with commitment protocols: enabling open web-based multiagent systems, Autonomous Agents and Multi-Agent Systems 2 (3) (1999) 217–236.
[53] M. Verdicchio, M. Colombetti, A logical model of social commitment for agent communication, in: Proceeding of the International Joint Conference on AAMAS, ACM, 2003, pp. 528–535.
[54] M. Verdicchio, M. Colombetti, A logical model of social commitment for agent communication, in: F. Dignum (Ed.), ACL'03, LNCS, vol. 2922, Springer, 2004, pp. 128–145.
[55] M. Winikoff, Implementing commitment-based interactions, in: E. Durfee, M. Yokoo, M. Huhns, O. Shehory, (Eds.), AAMAS, 2007, pp. 873–880.
[56] M. Wooldridge, An Introduction to MultiAgent Systems, second ed., John Wiley and Sons, 2009.
[57] L. Wu, J. Su, K. Su, X. Luo, Z. Yang, A concurrent dynamic logic of knowledge, belief, and certainty for multiagent systems, Knowledge-Based Systems 23 (2) (2010) 162–168.
[58] J. Xing, M.P. Singh, Formalization of commitment-based agent interaction, in: SAC, ACM, 2001, pp. 115–120.
[59] J. Xing, M.P. Singh, Engineering commitment-based multiagent systems: a temporal logic approach, in: Proceedings of the 2nd International Joint Conference on AAMAS, ACM, 2003, pp. 891–898.
[60] P. Yolum, M.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) (2004) 227–253.

Please cite this article in press as: J. Bentahar et al., Communicative commitments: Model checking and complexity analysis, Knowl. Based Syst. (2012), http://dx.doi.org/10.1016/j.knosys.2012.04.010