Model Checking Coalition Logic on Implicit Models is $\Delta^P_3$-complete

Nils Bulling
The IfI Review Board

Prof. Dr. Jürgen Dix (Theoretical Computer Science/Computational Intelligence)
Prof. i.R. Dr. Klaus Ecker (Applied Computer Science)
Prof. Dr. Barbara Hammer (Theoretical Foundations of Computer Science)
Prof. Dr. Sven Hartmann (Databases and Information Systems)
Prof. i.R. Dr. Gerhard R. Joubert (Practical Computer Science)
apl. Prof. Dr. Günter Kemnitz (Hardware and Robotics)
Prof. i.R. Dr. Ingbert Kupka (Theoretical Computer Science)
Prof. i.R. Dr. Wilfried Lex (Mathematical Foundations of Computer Science)
Prof. Dr. Jörg Müller (Business Information Technology)
Prof. Dr. Niels Pinkwart (Business Information Technology)
Prof. Dr. Andreas Rausch (Software Systems Engineering)
apl. Prof. Dr. Matthias Reuter (Modeling and Simulation)
Prof. Dr. Harald Richter (Technical Computer Science)
Prof. Dr. Gabriel Zachmann (Computer Graphics)
Prof. Dr. Christian Siemers (Hardware and Robotics)
Model Checking Coalition Logic on Implicit Models is $\Delta_3^P$-complete

Nils Bulling

Clausthal University of Technology, Department of Informatics, Germany
bulling@in.tu-clausthal.de

Abstract

In this note we show that model checking Coalition Logic over Concurrent Game Structures in which the transition function is given implicitly by a set of Boolean formulae is $\Delta_3^P$-complete.

1 Introduction

Coalition Logic [Pauly, 2002] (CL) is a strategic logic that allows to model and to reason about one-step abilities of agents. It is well known, that CL can be understood as the next-time fragment of Alternating-time Temporal Logic (ATL) [Alur et al., 2002]. Hence, model checking CL is at most as hard as for ATL. From [Alur et al., 2002] we know that model checking ATL is $P$-complete over Concurrent Game Structures (CGS's). In [Bulling et al., 2010] it is shown that the proof of the lower bound can easily be modified to provide also a $P$-hardness proof for CL. All these results are with respect to the size of a given CGS which is defined as the number of transitions in the model.

Often, the number of transitions in a CGS is exponential in terms of states and agents. Taking this into account it is reasonable to encode the transition function symbolically resulting in a more compact model; that is, in a model of smaller size. We will call such models implicit CGS’s. The size of such a model is measured with respect to the number of states and the size of the encoded transition function [Laroussinie et al., 2008]. Given this new representation/measure the model checking complexity of ATL is proven to be $\Delta_3^P$-complete [Laroussinie et al., 2008, Jamroga and Dix, 2005] and [Jamroga and Dix, 2008]. In this note we prove that CL is also $\Delta_3^P$-complete over implicit CGS's.
2 Coalition Logic: Syntax and Semantics

Firstly, we present the language $L_{CL}$; subsequently, we introduce implicit CGS’s and define a semantics for $L_{CL}$. In the following let $\Pi$ be a non-empty set of propositions and $\text{Agt} = \{1, \ldots, k\}$ be a non-empty and finite set of agents.

2.1 The language

*Coalition Logic (CL)*, introduced in [Pauly, 2002], is a logic for modeling and reasoning about strategic abilities of agents. The main construct of CL, $[A] \psi$, expresses that coalition $A$ can bring about $\psi$ in a single-step game.

**Definition 1 (Language $L_{CL}$ [Pauly, 2002])** The language $L_{CL}$ is given by all formulae generated by the following grammar: $\phi ::= p | \neg \phi | \phi \land \psi | [A] \phi$, where $p \in \Pi$ and $A \subseteq \text{Agt}$.

In [Pauly, 2002], coalitional models were chosen as semantics for $L_{CL}$. These models are given by $(St, E, \pi)$ consisting of a set of states $St$, a playable effectivity function $E$, and a valuation function $\pi$. The effectivity function determines the outcome that a coalition is effective for, i.e., given a set $X \subseteq St$ of states a coalition $C$ is said to be effective for $X$ iff it can enforce the next state to be in $X$. However, in [Goranko and Jamroga, 2004] it was shown that concurrent game structures (CGS’s) provide an equivalent semantics, and that CL can be seen as the *next-time fragment* of ATL.

2.2 Semantics

The semantics for $L_{CL}$ is defined over a variant of transition systems where transitions are labeled with combinations of actions, one per agent. Formally, a concurrent game structure (CGS) is a tuple

$$\mathfrak{M} = (\text{Agt}, St, \Pi, \pi, Act, d, o)$$

which includes a non-empty finite set of all agents $\text{Agt} = \{1, \ldots, k\}$, a non-empty finite set of states $St$, a set of atomic propositions $\Pi$ and their valuation $\pi : \Pi \rightarrow 2^{St}$, and a non-empty finite set of (atomic) actions $Act$. Function $d : \text{Agt} \times St \rightarrow 2^{Act}$ defines non-empty sets of actions available to agents at each state, and $o$ is a (deterministic) transition function that assigns the outcome state $q' = o(q, \alpha_1, \ldots, \alpha_k)$ to state $q$ and a tuple of actions $(\alpha_1, \ldots, \alpha_k)$ for $\alpha_i \in d(i, q)$ and $1 \leq i \leq k$, that can be executed by $\text{Agt}$ in $q$. We also write $d_a(q)$ instead of $d(a, q)$. So, it is assumed that all the agents execute their actions synchronously: The combination of the actions, together with the current state, determines the next transition of the system.
A strategy of agent $a$ is a conditional plan that specifies what $a$ is going to do in each state; that is, a function $s_a : St \rightarrow Act$ where $s_a(q) \in d_a(q)$. The set of such strategies is denoted by $\Sigma_a$.

A collective strategy for a group of agents $A = \{a_1, \ldots, a_r\} \subseteq \text{Agt}$ is simply a tuple $s_A = (s_{a_1}, \ldots, s_{a_r})$ of strategies, one per agent from $A$. By $s_A|a_i$, we denote agent $a$’s part $s_a$ of the collective strategy $s_A$ where $a \in A$. The set of $A$’s collective perfect information strategies is given by $\Sigma_A = \prod_{a \in A} \Sigma_a$.

Function $\text{out}(q, s_A)$ returns the set of all paths $\lambda$ that may occur when agents $A$ execute strategy $s_A$ from state $q$ onward:

$$\text{out}(q, s_A) = \{ \lambda = q_0q_1q_2\ldots | q_0 = q \text{ and for each } i = 1, 2, \ldots \text{ there exists a tuple of agents’ decisions } \langle \alpha_{a_1}^{-1}, \ldots, \alpha_{a_k}^{-1} \rangle \text{ such that } \alpha_{a_j}^{-1} \in d_a(q_{i-1}) \text{ for every } a_j \in \text{Ag}, \text{ and } o(q_{i-1}, \alpha_{a_1}^{-1}, \ldots, \alpha_{a_k}^{-1}) = q_i \}.$$  

The semantics for $\mathcal{L}_{CL}$ is shown below. Informally speaking, $\mathcal{M}, q \models [\Sigma] \varphi$ if, and only if, there exists a collective strategy $s_A$ such that $\varphi$ holds in the next state on each computation from $\text{out}(q, s_A)$.

**Definition 2 (Semantics $\models$)** Let $\mathcal{M}$ be a CGS. The semantics for $\mathcal{L}_{CL}$, denoted by $\models$, is defined as follows:

- $\mathcal{M}, q \models p$ iff $\lambda[0] \in \pi(p)$ and $p \in \Pi$;
- $\mathcal{M}, q \models \neg \varphi$ iff $\mathcal{M}, q \not\models \varphi$;
- $\mathcal{M}, q \models \varphi \land \psi$ iff $\mathcal{M}, q \models \varphi$ and $\mathcal{M}, q \models \psi$;
- $\mathcal{M}, q \models [\Sigma] \varphi$ iff there is a strategy $s_A \in \Sigma_A$ for $A$ such that for every path $\lambda \in \text{out}(s_A, q)$, we have $\mathcal{M}, \lambda[1] \models \varphi$.

Formally, the logic CL is given by $(\mathcal{L}_{CL}, \models)$; that is, by the language $\mathcal{L}_{CL}$ and the semantics just introduced.

An implicit concurrent game structure (to the best of our knowledge, this has been introduced for the first time in [Laroussinie et al., 2008], but already present in the ISPL modeling language behind MCMAS [Raimondi and Lomuscio, 2004, Raimondi, 2006]) is defined similarly to a CGS but the transition function is encoded in a particular way often allowing for a more compact representation than the explicit transition table. Formally, an implicit CGS is given by $\mathcal{M} = (\text{Ag}, \text{St}, \Pi, \pi, \text{Act}, d, \hat{o})$ where $\hat{o}$, the encoded transition function, is given by a sequence

$$( (\varphi_{r_0}^0, q_{r_0}^0), \ldots, (\varphi_{r_e}^e, q_{r_e}^e) )_{r=1, \ldots, |\text{St}|}$$

A path $\lambda$ is an infinite sequence of states such that subsequent states are connected by a transition. We use $\lambda[i]$ to refer to state $q_i$, i.e. $\lambda[i] = q_i$, provided that $\lambda = q_0q_1\cdots \in \text{St}^{\omega}$.
where \( t_r \in \mathbb{N} \), \( q_r^i \in St \) and each \( \varphi_r^i \) is a Boolean combination of propositions \( \text{exec}_j^i \), where \( j \in \text{Agent}, \alpha \in \text{Act}, i = 1, \ldots, t \) and \( r = 1, \ldots, |St| \). It is required that \( \varphi_r^i = \top \). The term \( \text{exec}_j^i \) stands for “agent \( j \) executes action \( \alpha \)”. We use \( \varphi[\alpha_1, \ldots, \alpha_k] \) to refer to the Boolean formula over \( \{\top, \bot\} \) obtained by replacing \( \text{exec}_j^i \) with \( \top \) (resp. \( \bot \)) if \( \alpha_j = \alpha \) (resp. \( \alpha_j \neq \alpha \)).

The encoded transition function induces a standard transition function \( o_\hat{o} \) as follows:

\[
o_\hat{o}(q_i, \alpha_1, \ldots, \alpha_k) = q^j_i \text{ where } j = \min\{\kappa \mid \varphi^i_{\kappa}[\alpha_1, \ldots, \alpha_k] \equiv \top\}
\]

That is, \( o_\hat{o}(q_i, \alpha_1, \ldots, \alpha_k) \) returns the state belonging to the formula \( \varphi^i_{\kappa} \) (associated with state \( q_i \)) with the minimal index \( \kappa \) that evaluates to “true” given the actions \( \alpha_1, \ldots, \alpha_k \). We also use \( \hat{o}(q_i, \alpha_1, \ldots, \alpha_k) \) to refer to \( o_\hat{o}(q_i, \alpha_1, \ldots, \alpha_k) \). Note that the function is well defined as the last formula in each sequence is equivalent to \( \top \): no deadlock can occur. The size of \( \hat{o} \) is defined as

\[
|\hat{o}| = \sum_{r=1}^{t} \sum_{j=1}^{|St|} |\varphi^j_r|
\]

that is, the sum of the sizes of all formulae. Hence, the size of an implicit CGS is given by \( |St| + |\text{Agent}| + |\hat{o}| \). Recall, that the size of an explicit CGS is \( |St| + |\text{Agent}| + m \) where \( m \) is the number of transitions. Finally, we require that the encoding of the transition function is reasonably compact, that is, \( |\hat{o}| \leq O(|o_\hat{o}|) \).

3 Model Checking Complexity

Firstly, we recall two well-known results.

**Theorem 1** ([Laroussinie et al., 2008; Jamroga and Dix, 2005]) Model checking ATL over implicit CGS’s is \( \Delta^P_3 \)-complete with respect to the size of the model and the length of the formula.

The \( \Delta^P_3 \)-hardness proof of [Laroussinie et al., 2008] uses the “nexttime” and “until” temporal operators in the construction of an ATL formula that is used in the reduction of SNSAT\(_2\). We give a proof that uses only the language \( \mathcal{L}_{\text{CL}} \).

**Theorem 2** Model checking CL over implicit CGS’s is \( \Delta^P_3 \)-complete with respect to the size of the model and the length of the formula.

**Proof.** The upper bound follows from the result that model checking ATL is in \( \Delta^P_3 \).
We extend the proof from [Laroussinie et al., 2008] such that only the next-time operator is used. The proof is done by reducing the $\Delta_2^0$-complete problem SNSAT$_2$. A SNSAT$_2$ instance $\mathcal{I}$ consists of formulae

$$z_i = \exists X_i \forall Y_i \psi_i(z_1, \ldots, z_{i-1}, X_i, Y_i)$$

where $X_i = \{x_i^1, \ldots, x_i^s\}$ and $Y_i = \{y_i^1, \ldots, y_i^s\}$ are sets of variables and $s \in \mathbb{N}$ for $i = 1, \ldots, m$. According to the truth of the formulae $\psi_i$, the value of each $z_i$ is uniquely defined. A valuation of $\mathcal{I}$ is a mapping $v_{\mathcal{I}}$ assigning these unique values to each variable $z_i$. Moreover, if $v_{\mathcal{I}}(z_i) = \top$ we define

$$v_{\mathcal{I}}^{z_i} : X_i \to \{\top, \bot\}$$

to be some valuation of the variables $X_i$ that witnesses the truth of $z_i$. Note, that each $z_i$ recursively depends on $z_{i-1}, \ldots, z_1$. In the following we will often omit the subscript $\mathcal{I}$.

We construct the following implicit CGS $\mathcal{M}_{\mathcal{I}}$ for a given SNSAT$_2$ instance $\mathcal{I}$. Firstly, we introduce agents, each controlling one variable. There are agents $a_i^1$ (one agent per variable $x_i^1$) with actions $\{\top, \bot\}$, $b_i^1$ (one agent per variable $y_i^1$) with actions $\{\top, \bot\}$, $c_i$ (one agent per $z_i$) with actions $\{\top, \bot\}$, and $d$ (the “selector”) with actions $\{1, \ldots, m\}$ for $i = 1, \ldots, m$ and $j = 1, \ldots, s$. We use $A$ (resp. $C$ and $B$) to denote the set of all agents $a_i^1$ (resp. $c_i$ and $b_i^1$).

The states of the model are given by states $q_i$ and $\bar{q}_i$ (one per $z_i$) and the two states $q_{\top}, q_{\bot}$. States $q_i$ are labelled with proposition neg and state $q_{\top}$ is labelled with sat.

Before giving the formal definition of the encoded transition function, we explain the role of the agents. Agents $a_i^1$ (resp. $b_i^1$ and $c_i$) determine the value of the variables $x_i^1$ (resp. $y_i^1$ and $z_i$). Action $\top$ (resp. $\bot$) sets them true (resp. false). Agent $d$ has a more elaborated function. Once, all moves of the other agents are fixed, the agent can decide to “check” whether formula $\psi_i$ holds regarding the actions of the other agents by executing action $i$. If the check is successful, the system goes to the winning state $q_{\top}$. If not, it goes to the losing state $q_{\bot}$. However, there are some exceptions to that which will be presented in the formal definition of the encoded transition function.

The part $(\varphi_1^0, q_0^0), \ldots, (\varphi_m^0, q_m^0)$ in the encoded transition function associated with state $q_i$ is defined as follows (where $\psi_i^0$ denotes the formula $\psi_i$ in $(\ast)$ in which each occurrence of $x_i^1$ (resp. $y_i^1$ and $z_i$) is replaced by $\text{exec}_{\top}^d$ (resp. $\text{exec}_{\top}^b$ and $\text{exec}_{\top}^c$)) (recall, that $\text{exec}_a^i$ means that agent $a$ executes action $i$):

1. $$(\text{exec}_k^d \land (\land_{j=1,\ldots,k} \text{exec}_{\top}^c) \land \psi_i^0, q_{\top})_{k=i,\ldots,1,$$
2. $$(\text{exec}_k^b \land (\land_{j=1,\ldots,k} \text{exec}_{\top}^c), q_{\bot})_{k=i,\ldots,1},$
3. $$(\text{exec}_k^d \land \neg \text{exec}_{\top}^c, \bar{q}_k)_{k=i-1,\ldots,1},$
4. $$(\top, q_{\top})$$

Moreover, there are loops at states $q_{\top} \text{ and } q_{\bot}$ and transitions from $\bar{q}_i \text{ to } q_i$ for $i = 1, \ldots, m$. The following lemma is fundamental to our reduction.
Lemma 3 Let $\chi_0 = \top$ and

$$\chi_{r+1} = [A \cup C](\text{sat} \lor (\neg \land [0] \neg \chi_r))$$

for $r = 0, \ldots, m - 1$ where $\text{sat}$ and $\neg$ are propositional symbols. Then, for all $i \leq m$ and $r \geq i$ it holds that

$$\mathcal{M}, q_i \models \chi_r \text{ iff } v_T(z_i) = \top.$$

Proof of Lemma. We proceed by induction on $i$. Firstly, we consider the base case $i = 1$.

$\Rightarrow$: Suppose that $\mathcal{M}, q_1 \models \chi_r$ for $r \geq 1$. Due to the definition of the transition function only rules $(1,2,4)$ are present; hence, only $q_{\top}$ and $q_{\bot}$ are reachable. That is, the formula $\mathcal{M}, q_1 \models [A \cup C] \text{sat}$ must be satisfied (as the label $\neg$ cannot become true). But then, there must be a valuation of the $x_i$'s such that for all valuations of the $y_i$'s, $\psi_1$ evaluates true; hence, $v(z_1) = \top$.

$\Leftarrow$: Suppose $v(z_1) = \top$. Then, there is a valuation of the variables $x_i$ such that for all valuations of the $y_i$'s the formula $\psi_1$ evaluates true. It is easily seen that the strategy in which each agent in $A$ plays according to the valuation given by $v^1$ and $c_1$ plays $\top$ witnesses that $q_1 \models [A \cup C] \bigcirc \text{sat}$ (and thus also $\mathcal{M}, q_1 \models \chi_r$ for $r \geq 1$).

For the inductive step suppose the assumption holds up to index $i \geq 1$.

$\Rightarrow$: Suppose $\mathcal{M}, q_{i+1} \models \chi_r+1$ for $r \geq i$. Firstly, we prove the following claim.

Claim: Suppose $\mathcal{M}, q_{i+1} \models \chi_{r+1}$, then each $c_l$ with $l \leq i$ plays according to the valuation $v(z_i)$.

Proof of claim. Suppose $c_l$ plays $\bot$ and $d$ plays $l$. Then, the next state of the system is $q_i$ and consequently, $\mathcal{M}, q_i \models \neg \chi_r$ and by induction hypothesis $v(z_i) = \bot$.

The other case is proven by induction. Suppose $i = 1$, $\mathcal{M}, q_2 \models \chi_{r+1}$, and $c_1$ plays $\top$. We have to show that $v(z_1) = \top$. Suppose the contrary. Then, for any strategy of $A \cup C$ there is a strategy of $B$ such that $\psi_1^i$ evaluates false. Hence, if $d$ plays $1$ rule $(2)$ is firing and the next state is $q_{\bot}$ and thus $\mathcal{M}, q_2 \not\models \neg \chi_{r+1}$. Contradiction!

For the induction step, suppose that all agents $c_l$ for $l < i$ play according to $v(z_i)$, that $\mathcal{M}, q_{i+1} \models \chi_{r+1}$, and $c_i = \top$. We show that $v(z_i) = \top$. For the sake of contradiction, suppose that $v(z_i) = \bot$. Again, for any strategy of $A \cup C$ witnessing $\chi_{r+1}$ we have that there is a strategy of $B$ that falsifies $\psi_1^i$ (note, that by assumption $c_1, \ldots, c_{i-1}$ play according to $v(z_1), \ldots, v(z_{i-1})$). So, if $d$ plays $i$ rule $(2)$ is firing and the next state is $q_{\bot}$ which implies $\mathcal{M}, q_{i+1} \not\models \chi_{r+1}$. Contradiction!

Now let $s_{AC}$ be the strategy of agents $A \cup C$ that witnesses $\chi_{r+1}$ in $q_{i+1}$. Suppose player $d$ plays $i + 1$. Irrelevant of the move of $c_{i+1}$ either rule $(1)$ or
rule (2) is firing. This does only depend on the valuation of $\psi'_{i+1}$. By assumption, we must have that $\psi'_{i+1}$ is true for all strategies of $B$ else $M, q_i \Vdash \chi_{i+1}$. Because of the previous claim, we must also have that $v(z_{i+1}) = \top$.

“$\Leftarrow$”: Suppose $v(z_{i+1}) = \top$. Let $s_{AC}$ be the strategy in which players $c_j$ play according to $v(z_j)$ and players $a_j$ play according to $v^{\top}$ if $v(z_j) = \top$ and arbitrarily if $v(z_j) = \bot$ for $a = 1, \ldots, s$. Suppose player $d$ plays $l \leq i + 1$. Now, if each $c_j$ for $j = i, \ldots, l$ plays $\top$ we have that $\psi'$ is true as there is no valuation of variables $Y$ that makes $\psi'$ false given the choices of $A \cup C$; hence, the next state is $q_{\top}$. Secondly, if $d$ plays $l$ and there is some agent $c_j$, $j > l$, that plays $\bot$; then rule (4) fires and the next state is also $q_{\top}$; the same holds if $d$ plays $l > i + 1$. Finally, suppose $d$ plays $l$ and $c_i = \bot$. Then, by the definition of the actions of agents $C$, $v(z_i) = \bot$ and by induction hypothesis $M, q_i \Vdash \neg \chi_r$; thus, $M, q_i \Vdash \neg \chi_r$ is true. Taking all these cases together we have $M, q_{i+1} \Vdash \chi_{i+1}$.

This gives us the following polynomial reduction:

$$z_m = \top \iff M, q_m \Vdash \chi_m$$

4 Conclusions

We have shown (Theorem 2) that model checking $L_{CL}$ over implicit CGS’s is already $\Delta^P_3$-complete; thus, resides in the same complexity class as model checking the more expressive language $L_{ATL}$. This mirrors the situation for (explicit) models over which model checking each of these two logics is $P$-complete.

References


