Action Models for Coalition Logic

Rustam Galimullin[(✉)1][0000−0003−4195−8189] and Thomas Ågotnes[1,2]

[1] University of Bergen, Bergen, Norway
[2] Southwest University, Chongqing, China
{rustam.galimullin, thomas.agotnes}@uib.no

Abstract. In the paper, we study the dynamics of coalitional ability by proposing an extension of coalition logic (CL). CL allows one to reason about what a coalition of agents is able to achieve through a joint action, no matter what agents outside of the coalition do. The proposed dynamic extension is inspired by dynamic epistemic logic, and, in particular, by action models. We call the resulting logic coalition action model logic (CAML), which, compared to CL, includes additional modalities for coalitional action models. We investigate the expressivity of CAML, and provide a complexity characterisation of its model checking problem.

Keywords: Dynamic Coalition Logic · Coalition Logic · Dynamic Epistemic Logic · Action Model Logic.

1 Introduction

Coalition logic (CL) [24, 23] is one of the most well-known formalisms for reasoning about strategic abilities of groups of agents in the presence of opponents. Modalities $\langle\langle C\rangle\rangle\varphi$ of CL express the fact that 'there is a joint action by agents from coalition C such that no matter what agents outside of the coalition do at the same time, φ will be true'. CL was conceived as a formal language for strategic games, and constructs $\langle\langle C\rangle\rangle\varphi$ characterise the existence of a winning strategy for agents in C.

One way to approach models of CL is to view them as protocols or contracts specifying what agents can and cannot do in different states. In this paper, we propose an extension of CL, which we call coalition action model logic (CAML), that includes modalities for updating those models. Such updates are carried out with respect to *action models* that are expressed in the language with formulas $[\mathsf{M_s}]\varphi$ meaning 'after executing action model $\mathsf{M_s}$, φ is the case'.

In creating CAML we followed the lead of dynamic epistemic logic (DEL) [12], and, in particular, of action model logic (AML) [9, 12]. Action models in AML model various epistemic events that can influence agents' knowledge about facts of the world and about knowledge of other agents. In a similar vein, coalitional action models of CAML can influence strategic abilities of coalitions of agents. On a more general scale, we hope that CAML will be a step towards a study of *dynamic coalition logic* (DCL). To make the link between DEL and DCL even more explicit, we can say that while DEL captures the *the dynamics of knowledge*, DCL should be able to capture the *dynamics of ability*.

CAML is not the first dynamic coalition logic. In [15] the authors proposed dictatorial dynamic coalition logic (DDCL) that was inspired by arrow update logic [21] and relation-changing logics [6]. DDCL updates strategic abilities of single agents by granting them dictatorial powers or revoking such powers. Compared to DDCL, action models of CAML allow for more fine-tuned updates that may affect more than one agent in various ways. On the other hand, modalities of CAML neither grant agents new actions they have not had before, nor remove such actions. Thus, coalitional action models can be viewed as prescriptions of how protocols or contracts between agents should be modified while taking into account what agents actually can and cannot do.

The implementation of an action model in CAML is not, however, merely a restriction (submodel) of the initial model. The action model might prescribe several different modifications compatible with the same state in the initial model, and the resulting updated model might have *more* states than the initial one. We capture this by using a definition of a product update very similar to the one used in AML. Restrictions on transition systems corresponding to policies, norms, or social laws is far from a new idea [27, 28], and logical formalisms for reasoning about such restrictions have been extensively studied, particularly using systems based on computation tree logic (CTL) [3]. In [4] a language similar to CL is used: an expression of the form $\langle C \rangle \varphi$, where C is a coalition and φ is a temporal formula, expresses the fact that if coalition C *complies* with the normative system, then φ will be true. Here, formulas are interpreted in the context of a single, given, restriction on legal actions, and although one can quantify over different parts of that restriction by varying the coalition C, the resulting submodel will always be a *restriction* of the initial model. The conceptual overlap notwithstanding, CAML is significantly different: as mentioned earlier, the updated models, obtained using action models, are not necessarily submodels, and the underlying models are CL models with joint actions rather than Kripke models of CTL.

After we briefly present necessary background information on CL in Section 2, we introduce CAML in Section 3. In Section 4 we show that CAML is strictly more expressive than CL. This result shows a crucial difference between AML and CAML: whereas the former is as expressive as the underlying epistemic logic, and thus completeness of AML follows trivially from reduction axioms, we cannot have reduction axioms for CAML. Moreover, we claim that CAML is incomparable to alternating-time temporal logic (ATL) [5]. Added expressivity of CAML comes at a price. In Section 5 we show that the complexity of the model checking problem jumps from P for CL to *PSPACE*-complete for CAML. Finally, we conclude in Section 6.

2 Language and Semantics of Coalition Logic

In this section we briefly provide all the necessary background information on coalition logic [2, 23]. Let A be a finite set of agents, and P be a countable set of propositional variables.

Definition 1. *The* language of coalition logic $\mathcal{CL}$ *is defined by BNF:*

$$\varphi ::= p \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \langle\langle C\rangle\rangle\varphi$$

where $p \in P$ and $C \subseteq A$. Formulas $\langle\langle C\rangle\rangle\varphi$ are read 'coalition C can force φ'. We denote $A \setminus C$ as $\overline{C}$. The dual of $\langle\langle C\rangle\rangle\varphi$ is $[\![C]\!]\varphi := \neg\langle\langle C\rangle\rangle\neg\varphi$. We will call subsets of A 'coalitions', and we will also call complements of C, $\overline{C}$, 'the anti-coalition'.

The semantics of CL is given with respect to concurrent game models. A *concurrent game model* (CGM), or a *model*, is a tuple $M = (S, Act, act, out, L)$. S is a non-empty set of states, and Act is a non-empty set of actions.

The function $act : A \times S \to 2^{Act} \setminus \emptyset$ assigns to each agent and each state a non-empty set of actions. A C-action at a state $s \in S$ is a tuple α_C such that $\alpha_C(i) \in act(i, s)$ for all $i \in C$. The set of all C-actions in s is denoted by $act(C, s)$. We will also write $\alpha_{C_1} \cup \alpha_{C_2}$ to denote a $C_1 \cup C_2$-action with $C_1 \cap C_2 = \emptyset$.

A tuple of actions $\alpha = \langle\alpha_1, \ldots, \alpha_k\rangle$ with $k =\mid A \mid$ is called an *action profile*. An action profile is *executable* in state s if for all $i \in A$, $\alpha_i \in act(i, s)$. The set of all action profiles executable in s is denoted by $act(s)$. An action profile α *extends* a C-action α_C, written $\alpha_C \sqsubseteq \alpha$, if for all $i \in C$, $\alpha(i) = \alpha_C(i)$.

The function *out* assigns to each state s and each $\alpha \in act(s)$ a unique output state. We write $Out(s, \alpha_C)$ for $\{out(s, \alpha) \mid \alpha \in act(s) \text{ and } \alpha_C \sqsubseteq \alpha\}$. Intuitively, $Out(s, \alpha_C)$ is the set of all states reachable by action profiles that extend some given C-action α_C. Finally, $L : S \to 2^P$ is the valuation function.

We will also denote a CGM M with a designated, or current, state s as M_s, and will sometimes call it a *pointed model*. We call M *finite* if S is finite.

Definition 2. *Let M_s be a pointed CGM. The semantics of CL is defined inductively as follows:*

$$\begin{array}{ll}
M_s \models p & \textit{iff } s \in L(p) \\
M_s \models \neg\varphi & \textit{iff } M_s \not\models \varphi \\
M_s \models \varphi \wedge \psi & \textit{iff } M_s \models \varphi \textit{ and } M_s \models \psi \\
M_s \models \langle\langle C\rangle\rangle\varphi & \textit{iff } \exists\alpha_C, \forall\alpha_{\overline{C}} : M_t \models \varphi, \textit{ where } t = out(s, \alpha_C \cup \alpha_{\overline{C}}) \\
M_s \models [\![C]\!]\varphi & \textit{iff } \forall\alpha_C, \exists\alpha_{\overline{C}} : M_t \models \varphi, \textit{ where } t = out(s, \alpha_C \cup \alpha_{\overline{C}})
\end{array}$$

Informally, the semantics of the coalition modality $\langle\langle C\rangle\rangle\varphi$ means that in the current state of a given CGM there is a choice of actions by the members of coalition C such that no matter what the opponents from the anti-coalition $\overline{C}$ choose to do at the same time, φ holds after the execution of the corresponding action profile.

Definition 3. *We call a formula φ* valid *if for all M_s it holds that $M_s \models \varphi$.*

Example 1. An example of a CGM is presented in Figure 1 on the left. The model is called M and it describes the following protocol. There are two states: s, where agents receive a *p*rize (propositional variable p), and state t, where agents do not receive a prize. Each agent has two actions in each state, and they can switch states by 'synchronisation', i.e. by choosing actions with the same number, either a_0b_0 or a_1b_1. Formally, $M_s \models p \wedge \langle\langle\{a, b\}\rangle\rangle\neg p$. At the same time, no agent alone can force the transition to state t, or, in symbols, $M_s \models [\![a]\!]p \wedge [\![b]\!]p$.

The classic notion of indistinguishability between models in modal logic is bisimulation. In this paper, we will use a CGM-specific version of bisimulation [1].

Definition 4. *Let* $M = (S^M, Act^M, act^M, out^M, L^M)$ *and* $N = (S^N, Act^N, act^N, out^N, L^N)$ *be two CGMs. A relation* $Z \subseteq S^M \times S^N$ *is called* bisimulation *if and only if for all* $C \subseteq A$, $s_1 \in S^M$ *and* $s_2 \in S^N$, $(s_1, s_2) \in Z$ *implies*

- *for all* $p \in P$, $s_1 \in L^M(p)$ *iff* $s_2 \in L^N(p)$;
- *for all* $\alpha_C \in act^M(C, s_1)$, *there exists* $\beta_C \in act^N(C, s_2)$ *such that for every* $s_2' \in Out^N(s_2, \beta_C)$, *there exists* $s_1' \in Out^M(s_1, \alpha_C)$ *such that* $(s_1', s_2') \in Z$.
- *The same as above with 1 and 2 swapped.*

If there is a bisimulation between M *and* N *linking states* s_1 *and* s_2, *we call the pointed models* bisimilar *(*$M_{s_1} \leftrightarrows N_{s_2}$*).*

The crucial property of bisimilar models, that will be of use later in the paper, is that bisimilar models satisfy the same set of formulas of coalition logic.

Theorem 1 ([1]). *Let* M *and* N *be CGMs such that* $M \leftrightarrows N$ *and there is a bisimulation between* $s \in S^M$ *and* $t \in S^N$. *Then for all* $\varphi \in \mathcal{CL}$, $M_s \models \varphi$ *iff* $N_t \models \varphi$.

3 Coalition Action Model Logic

Before providing formal definitions, we introduce coalitional action models intuitively with an example.

3.1 Informal Exposition and Example

Coalitional action models are inspired by action models of DEL [9, 12], and they are, basically, models, where each state has an assigned formula that is called a *precondition*. Preconditions indicate which states of action models are executable in which states of a given CGM. An action model can be viewed as a *policy*, explicitly describing *legal* joint actions and implicitly imposing *restrictions* on existing joint actions in different states. However, the result of implementing an action model policy is not necessarily a submodel of the initial model: it can in fact have more states, if the action model describes more possible actions compatible with the same state in the initial model. We capture this by a *product update* using a restricted Cartesian product, very similar to product updates in AML. In particular, in the update of a CGM with an action model, we take a product of states of the CGM and those states of the action model that are satisfied according to preconditions. In the resulting *updated model*, a transition labelled with an action profile is preserved, if there is a corresponding transition in both CGM and the action model. To differentiate CGMs and coalitional action models, we will use sans-serif font for the elements of the latter.

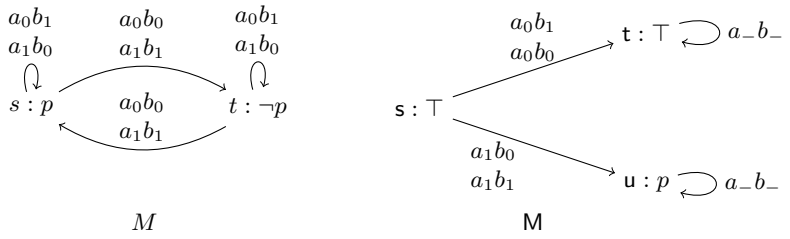

Fig. 1. Model M and action model M.

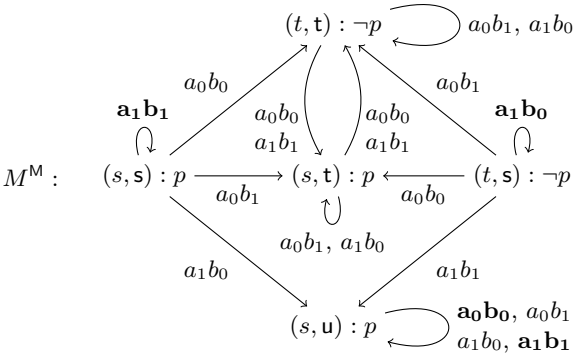

Fig. 2. Updated model M^{M} with added action profiles in bold font.

Example 2. As a continuation of our prize example, consider action model M in Figure 1. In the figure, $a\_b\_$ is a shorthand that the corresponding transition is labelled by all of a_0b_0, a_0b_1, a_1b_0, and a_1b_1.

Action model M describes a policy that prescribes the following modification of the protocol expressed by CGM M. In all states, s or t, if the first agent chooses action a_0, then follow the prize protocol without any modifications (expressed by state t). If the first agent chooses action a_1, then agents enter a state, where each their joint action gets a prize (expressed by state u).

The result of updating CGM M with action model M is updated model M^{M} (Figure 2) that is based on a product of states of M with those states of M that satisfy preconditions. For example, precondition of state s is satisfied by both s and t, and thus we have both (s, s) and (t, s) in the updated model. There is an arrow labelled with an action profile α between some (s, s) and (t, t) if there are arrows labelled with α from s to t and from s to t. Finally, p is satisfied by (s, s) if $p \in L(s)$.

Observe that although in the example for both M and M transitions from each state were defined for each action profile, it is not the case for the corresponding function of M^{M}. The reason for this is that the intersection of transitions from M and M is not guaranteed to include all executable action profiles.

Indeed, in the example, in M_s action profile a_1b_1 takes the agents to a $\neg p$-state, while the same profile in $\mathsf{M_s}$ takes the agents to a p-state. Similarly for action profiles a_1b_0 in states t and s, and a_0b_0 and a_1b_1 in states s and u.

This can be interpreted as the uncertainty (or a conflict) agents may have when a new modification contradicts the existing protocol. We deal with such situations by making the agents remain in the current state in the cases of such uncertainty. In other words, we follow the rule that says *when in doubt, remain where you are*. On the level of updated models this means that for all action profiles α that are not defined, we put $out^{M^{\mathsf{M}}}((s,\mathsf{s}),\alpha) = (s,\mathsf{s})$. That is why in M^{M} action profiles a_1b_0 and a_1b_1 (in bold font) loop back to states (t,s) and (s,s) correspondingly. Moreover, action profiles a_0b_0 and a_1b_1 loop back to (s,u). In this paper, we will write labels of added self-loops in bold font.

Of course, our approach to managing these conflicts is quite conservative, and one can imagine more radical ways of updating a CGM. We leave the exploration of such alternatives for future work.

All in all, action model M updates agents' strategic abilities by taking into account what they actually can achieve in a given CGM M. Thus, in Figure 2 we can indeed see that in state (s,s) action a_0 by the first agents leads to agents executing the same protocol as described by CGM M (states (s,t) and (t,t) in the updated model). On the other hand, contrary to the situation in the initial model, now in the updated model agents can reach state (s,u), where they always receive prizes. Formally, we can write $M_s \not\models \langle\!\langle \{a,b\} \rangle\!\rangle [\![\{a,b\}]\!] p$ and $M_s \models [\mathsf{M_s}] \langle\!\langle \{a,b\} \rangle\!\rangle [\![\{a,b\}]\!] p$, where construct $[\mathsf{M_s}]$ means execution of action model M with the actual state s.

3.2 Syntax and Semantics of Coalition Action Modal Logic

Definition 5. *A* coalitional action model M*, or an* action model*, is a tuple* $(\mathsf{S}, \mathsf{Act}, \mathsf{act}, \mathsf{out}, \mathsf{pre})$*, where* S *is a finite non-empty set of states,* Act *is a non-empty set of actions,* $act : A \times \mathsf{S} \to 2^{\mathsf{Act}} \setminus \emptyset$ *assigns to each agent and each state a non-empty set of actions. Definitions related to action profiles are the same as in the definition of CGM. Function* out *is a* partial function *that maps action profiles executable in a state to a unique output state. Finally,* $\mathsf{pre} : \mathsf{S} \to \mathcal{CL}$ *assigns to each state a formula of coalition logic. We denote action model* M *with a designated state* s *by* $\mathsf{M_s}$*.*

Definition 6. *The* language of coalition action model logic $\mathcal{CAML}$ *is given recursively by the following grammar:*

$$\varphi ::= p \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \langle\!\langle C \rangle\!\rangle \varphi \mid [\pi]\varphi$$
$$\pi ::= \mathsf{M_s} \mid (\pi \cup \pi)$$

where $[\pi]\varphi$ *is read 'after execution of* π*,* φ *is true', and the union operator stands for a non-deterministic choice. Dual* $\langle\pi\rangle\varphi$ *is defined as* $\neg[\pi]\neg\varphi$*.*

Definition 7. *Let* $M_s = (S, Act, act, out, L)$ *be a pointed CGM and* $\mathsf{M_s} = (\mathsf{S}, \mathsf{Act}, \mathsf{act}, \mathsf{out}, \mathsf{pre})$ *be a coalitional action model. The* semantics of CAML *extends the semantics of CL in Definition 2 with the following:*

$$
\begin{array}{ll}
M_s \models [\mathsf{M_s}]\varphi & \textit{iff } M_s \models \mathsf{pre(s)} \textit{ implies } M^{\mathsf{M}}_{(s,\mathsf{s})} \models \varphi \\
M_s \models \langle \mathsf{M_s} \rangle \varphi & \textit{iff } M_s \models \mathsf{pre(s)} \textit{ and } M^{\mathsf{M}}_{(s,\mathsf{s})} \models \varphi \\
M_s \models [\pi \cup \rho]\varphi & \textit{iff } M_s \models [\pi]\varphi \textit{ and } M_s \models [\rho]\varphi \\
M_s \models \langle \pi \cup \rho \rangle \varphi & \textit{iff } M_s \models \langle \pi \rangle \varphi \textit{ or } M_s \models \langle \rho \rangle \varphi
\end{array}
$$

The updated model M^{M} *is a tuple* $(S^{M^{\mathsf{M}}}, Act, act, out^{M^{\mathsf{M}}}, L)$*, where*

$$S^{M^{\mathsf{M}}} = \{(s, \mathsf{s}) \mid s \in S, \mathsf{s} \in \mathsf{S}, \textit{ and } M_s \models \mathsf{pre(s)}\},$$

$$out^{M^{\mathsf{M}}}((s,\mathsf{s}),\alpha) = \begin{cases} (t,\mathsf{t}) & (t,\mathsf{t}) \in S^{M^{\mathsf{M}}}, out(s,\alpha) = t \textit{ and } \mathsf{out(s},\alpha) = \mathsf{t} \\ (s,\mathsf{s}), & \textit{otherwise.} \end{cases}$$

According to the definition of an updated model, we assume that action models do not grant agents new actions, and, moreover, the valuation of propositional variables remains the same. Thus, action models can be viewed as policy updates that deal only with agents' strategic abilities, and take into account what agents can actually do in the current CGM. Another point worth mentioning is that in our definition of action models we do not require the function out to be total.

There are many similarities between AML and CAML. In particular, all of the following properties are valid for both logics. Their validity in the case of CAML can be shown by application of the definition of the semantics.

Proposition 1. *All of the following are valid.*

1. $\langle \mathsf{M_s} \rangle \varphi \to [\mathsf{M_s}]\varphi$
2. $[\mathsf{M_s}]p \leftrightarrow (\mathsf{pre(s)} \to p)$
3. $[\mathsf{M_s}]\neg\varphi \leftrightarrow (\mathsf{pre(s)} \to \neg[\mathsf{M_s}]\varphi)$
4. $[\mathsf{M_s}](\varphi \wedge \psi) \leftrightarrow ([\mathsf{M_s}]\varphi \wedge [\mathsf{M_s}]\psi)$
5. $[\pi \cup \rho]\varphi \leftrightarrow [\pi]\varphi \wedge [\rho]\varphi$

The first item states that there is only one way to execute an action model. This is similar to public announcements [25]. Note, however, that in general, $\langle \pi \rangle \varphi \to [\pi]\varphi$ is not valid, since $\langle \pi \rangle \varphi$ is executed non-deterministically. The second property shows that updating a model does not affect propositional variables. Interaction between action models and negation is captured by the third item. Property number four states distributivity of action model updates over conjunction. Finally, the fifth item shows how we can get rid of the union.

Even though Proposition 1 shows that AML and CAML have much in common, the logics are different in a very crucial way. Items two, three, and four of the proposition, interpreted as AML formulas, in conjunction with an interaction principle for AML action models and knowledge modality, constitute *reduction axioms* of AML. This means, that in the context of AML, each formula with an action model can be equivalently rewritten into a formula without it, thus showing that AML is as expressive as epistemic logic, and providing a completeness proof for AML 'for free' (see more on this [12, Sections 6,7, and 8]). We show in the next section that this is not true for CAML.

4 Expressivity

In this section, we argue that, unlike the case of DEL, CAML is strictly more expressive than CL, and thus no reduction axioms are possible. Apart from that, we compare CAML to alternating-time temporal logic (ATL) [5].

Definition 8. *Let φ and ψ be formulas of a language interpreted on CGMs. We say that they are* equivalent *if for all pointed CGMs M_s it holds that $M_s \models \varphi$ iff $M_s \models \psi$.*

Definition 9. *Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be two languages. We say that $\mathcal{L}_1$* is at least as expressive as *$\mathcal{L}_2$ ($\mathcal{L}_2 \leqslant \mathcal{L}_1$) if and only if for all $\varphi \in \mathcal{L}_2$ there is an equivalent $\psi \in \mathcal{L}_1$. If $\mathcal{L}_1$ is* not *at least as expressive as $\mathcal{L}_2$, we write $\mathcal{L}_2 \not\leqslant \mathcal{L}_1$. If $\mathcal{L}_2 \leqslant \mathcal{L}_1$ and $\mathcal{L}_1 \not\leqslant \mathcal{L}_2$, we write $\mathcal{L}_2 < \mathcal{L}_1$ and say that $\mathcal{L}_1$* is strictly more expressive than *$\mathcal{L}_2$. Finally, if $\mathcal{L}_1 \not\leqslant \mathcal{L}_2$ and $\mathcal{L}_2 \not\leqslant \mathcal{L}_1$, we say that $\mathcal{L}_1$ and $\mathcal{L}_2$ are incomparable.*

Theorem 2. $\mathcal{CL} < \mathcal{CAML}$.

Proof. That $\mathcal{CL} \leqslant \mathcal{CAML}$ follows from the fact that $\mathcal{CL} \subset \mathcal{CAML}$. To show that $\mathcal{CAML} \not\leqslant \mathcal{CL}$, consider models M_s from and N_s from Figure 3. In the models, $a\_b\_$ is a shorthand for all of a_0b_0, a_0b_1, a_1b_0, and a_1b_1. Observe that the two models are quite similar and the difference is that in M_s the agents can force $\neg p$ if they choose actions labelled with the same number, e.g. a_0b_0, and in N_s the agents can force $\neg p$ is they choose actions labelled with different numbers, e.g. a_0b_1. It is easy to check that the two models are bisimilar, and thus satisfy the same formulas of $\mathcal{CL}$ by Theorem 1.

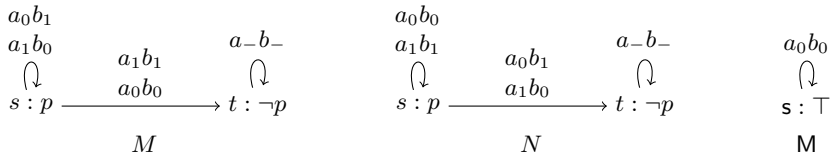

Fig. 3. Models M, N, and action model M.

Now consider action model M in Figure 3. The action model has only one state with the precondition $\top$ and one self-loop labelled with a_0b_0. The results of updating M_s and N_s with $\mathsf{M_s}$ are presented in Figure 4.

It is clear that $M_s \models \langle \mathsf{M_s} \rangle \langle\!\langle \{a, b\} \rangle\!\rangle \neg p$ and $N_s \not\models \langle \mathsf{M_s} \rangle \langle\!\langle \{a, b\} \rangle\!\rangle \neg p$. Thus we have that, first, no formula of $\mathcal{CL}$ can distinguish M_s and N_s, and, second, that formula $\langle \mathsf{M_s} \rangle \langle\!\langle \{a, b\} \rangle\!\rangle \neg p$ of $\mathcal{CAML}$ distinguishes the models. Hence, $\mathcal{CL} \leqslant \mathcal{CAML}$. □

From Theorem 2 it follows that there cannot be any reduction axioms for CAML. There is, however, yet another interesting corollary. In the proof, we

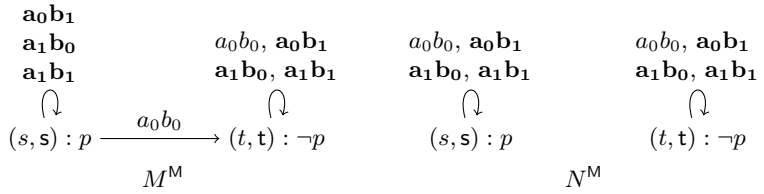

Fig. 4. Updated models M^{M} and N^{M} with added action profiles in bold font.

started with two bisimilar models, and the results of updating them with the same coalitional action model turned out to be not bisimilar. This is quite different from DEL, where updates with action models preserve bisimulation [12, Proposition 6.21].

Corollary 1. *Coalitional action models do not preserve bisimulation.*

Now we turn to the comparison of CAML and ATL [5], with the latter being, probably, the most well-known logic for reasoning about strategic abilities. Other notable examples of such logics include ATL* [5] and strategy logic (SL) [22]. We will consider only ATL in this paper, and the expressivity proofs below can be reused for both ATL* and SL.

ATL extends CL with temporal operators $\mathsf{X}\varphi$ for 'φ is true in the next step', $\mathsf{G}\varphi$ for 'φ is always true', and $\psi\mathsf{U}\varphi$ for 'ψ is true until φ'. Below, we concisely present ATL, and the the interested reader can find more details in the literature [5, 1, 2].

Definition 10. *The $\mathcal{ATL}$ is defined recursively as follows:*

$$\varphi ::= p \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \langle\langle C\rangle\rangle \mathsf{X}\varphi \mid \langle\langle C\rangle\rangle \mathsf{G}\varphi \mid \langle\langle C\rangle\rangle \varphi \mathsf{U} \varphi$$

Before we provide the semantics of the logic, we need some additional definitions.

Definition 11. *A* memoryless strategy *for agent i in model M is a function $str_i : S \to Act$ such that $str_i(s) \in act(i, s)$. A memoryless strategy for coalition C, denoted str_C is a tuple of memoryless strategies for each $i \in C$.*

Definition 12. *Given a CGM $M = (S, Act, act, out, L)$, a* play *$\lambda$ is an infinite sequence of states in S such that for all $i \geq 0$, state s_{i+1} is a successor of s_i. We will denote the i'th element of play λ as $\lambda[i]$. The set of all plays that can be realised by coalition C following strategy str_C from some given state s, denoted by $Plays(s, str_C)$, is defined as*

$$\{\lambda \mid \lambda[0] = s \text{ and } \lambda[i+1] \in Out(\lambda[i], str_C(\lambda[i])) \text{ for all } i \in \mathbb{N}\}.$$

Definition 13. *Let M_s be a CGM. The semantics of ATL (omitting Boolean cases) is defined as follows:*

$$
\begin{array}{l}
M_s \models \langle\langle C\rangle\rangle \mathsf{X}\varphi \quad \textit{iff } \exists \alpha_C : M_t \models \varphi \textit{ for all } t \in Out(s, \alpha_C) \\
M_s \models \langle\langle C\rangle\rangle \mathsf{G}\varphi \quad \textit{iff } \exists str_C, \forall \lambda \in Plays(s, str_C) : M_{\lambda[i]} \models \varphi \textit{ for all } i \geq 0 \\
M_s \models \langle\langle C\rangle\rangle \psi \mathsf{U} \varphi \textit{ iff } \exists str_C, \forall \lambda \in Plays(s, str_C), \exists i \geq 0 : \\
\qquad\qquad\qquad\qquad M_{\lambda[i]} \models \varphi, \textit{ and } M_{\lambda[j]} \models \psi \textit{ for all } 0 \leq j < i
\end{array}
$$

It is immediate that $\langle\langle C\rangle\rangle \mathsf{X}\varphi$ in ATL is equivalent to $\langle\langle C\rangle\rangle \varphi$ in CL.

To show that the languages of ATL and CAML are incomparable with respect to expressive power, we need only *Always* operator out of all temporal features of ATL. The intuition behind the proof is that formula $\langle\langle C\rangle\rangle \mathsf{G}\varphi$ can possibly reach states on any distance from the current one. At the same time, the distance formulas of $\mathcal{CAML}$ can reach is bounded by their sizes.

Theorem 3. *$\mathcal{CAML}$ and $\mathcal{ATL}$ are incomparable.*

Proof. To argue that $\mathcal{CAML} \not\leqslant \mathcal{ATL}$, we can reuse the proof of Theorem 2 without any modification with only mentioning that for bisimilar models, Theorem 1 can be extended to $\mathcal{ATL}$ [1].

For the other direction, consider an $\mathcal{ATL}$ formula $\langle\langle a\rangle\rangle \mathsf{G}p$, and assume, towards a contradiction, that there is an equivalent formula of $\mathcal{CAML}$ φ of size $n = |\varphi|$. Also, consider models M and N in Figure 5.

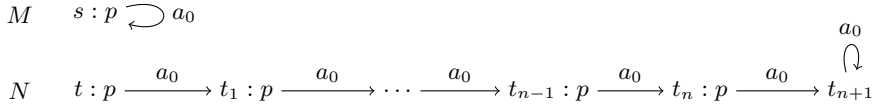

Fig. 5. Models M and N.

Model M is a single-state model with a loop, where state s satisfies p. Model N is a chain of size $n+2$, where states t, and t_i for $1 \leqslant i \leqslant n$ satisfy p, and state t_{n+1} does not satisfy p.

We have that $M_s \models \langle\langle a\rangle\rangle \mathsf{G}p$ since there is only one state available, s, and the state satisfies p. On the other hand, it is clear that $N_t \not\models \langle\langle a\rangle\rangle \mathsf{G}p$. Indeed, we can see that $\mathsf{G}p$ does not hold in t (or any other state of N): a $\neg p$-state, t_{n+1}, is reachable from t.

That $M_s \models \varphi$ if and only if $N_t \models \varphi$ can be shown by induction on the structure of φ. Boolean and coalitional cases are straightforward. Also note that state t_{n+1}, which could be used to distinguish M_s and N_t, is n steps away from t. Thus, there is not enough modal depth to witness the difference using only coalitional modalities. For the case of action models, assume that in the resulting updated model there is a state (t_k, s). Since action models do not change values of propositional variables, (t_k, s) will satisfy the same propositions as t_k satisfied in N. Moreover, recall that there is a transition from one state to another in the updated model, if the transition was present both in N and the action model. Thus, (t_k, s) can be connected via a_0 only to states (t_{k+1}, t).Hence, a

distinguishing non-p-state will still remain $n - i$ steps away after i steps of the induction. □

5 Model Checking

Now we show that the model checking problem for CAML is *PSPACE*-complete. It is known that model checking coalition logic can be done in polynomial time, and, thus, in the case of CAML, we have to pay for increased expressivity with higher complexity. This is similar to the situation with DEL, where model checking epistemic logic can be done in polynomial time [19], and the complexity of model checking action model logic is *PSPACE*-complete [8].

Theorem 4. *The model checking problem for CAML is PSPACE-complete.*

Proof. To show that the model checking problem for CAML is in *PSPACE*, we present Algorithm 1. Boolean cases and the case of coalition modalities take polynomial time and we omit them for brevity. For an overview of model checking of strategic logics, including CL, see [11].

Algorithm 1 An algorithm for model checking CAML

```
1: procedure MC(M, s, φ)
2:     case φ = [M_s]ψ
3:         if MC(M, s, pre(s)) then
4:             return MC(M^M, (s, s), ψ)
5:         else
6:             return true
7:     case φ = [π ∪ ρ]ψ
8:         return MC(M, s, [π]ψ) and MC(M, s, [ρ]ψ)
```

The algorithm follows the semantics and its correctness can be shown via induction on φ. Now we argue that the algorithm takes at most polynomial space. The interesting case here is $\varphi = [\mathsf{M}_\mathsf{s}]\psi$. Since preconditions are formulas of coalition logic, $\mathrm{MC}(M, s, \mathsf{pre(s)})$ is computed in polynomial time, and hence space. The size of updated model M^M is bounded by $\mathcal{O}(|M| \times |\mathsf{M}|) \leqslant \mathcal{O}(|M| \times |\varphi|)$. Finally, since there at most $|\varphi|$ symbols in φ, the total space required by $\mathrm{MC}(M, s, \varphi)$ is bounded by $\mathcal{O}(|M| \times |\varphi|^2)$.

To show hardness, we take the *PSPACE*-hardness proof of the model checking problem for AML [8] as a starting point, and adapt the technique to CGMs and coalitional action models. The main difficulty we face here is that we need to fine-tune models and action models used in the proof in order to ensure that *out* functions behave as expected.

We use the classic reduction from the satisfiability of quantified Boolean formula (QBF) that is known to be *PSPACE*-complete. Also, without loss of generality, we assume that our QBFs have $2k$ variables with alternating quantifiers. See more on satisfiability of such QBFs in [7, p. 83].

For a given QBF $\Psi := \forall x_1 \exists x_2 \dots \forall x_{2k-1} \exists x_{2k} \psi(x_1, \dots, x_{2k})$ we construct in polynomial time a CGM M_s over $A = \{a\}$, action models $\mathsf{AddChain}i_{\mathsf{s}^{\mathsf{i}}_0}$ for all x_i, action model $\mathsf{Copy}_{\mathsf{t}}$, and a formula of $\mathcal{CAML}$ ψ' such that

$$\Psi \text{ is satisfiable iff}$$
$$M_s \models [\mathsf{AddChain1}_{\mathsf{s}^1_0} \cup \mathsf{Copy}_{\mathsf{t}}]\langle \mathsf{AddChain2}_{\mathsf{s}^2_0} \cup \mathsf{Copy}_{\mathsf{t}}\rangle \dots$$
$$[\mathsf{AddChain}(2k-1)_{\mathsf{s}^{2k-1}_0} \cup \mathsf{Copy}_{\mathsf{t}}]\langle \mathsf{AddChain2}k_{\mathsf{s}^{2k}_0} \cup \mathsf{Copy}_{\mathsf{t}}\rangle \psi'.$$

Model M is a tuple (S, Act, act, out, L), where $S = \{s_i \mid 0 \leqslant i \leqslant 2k+1\}$, $Act = \{a_i \mid 0 \leqslant i \leqslant 2k\}$, $act(a, s_i) = Act$ for $0 \leqslant i \leqslant 2k$, and $act(a, s_{2k+1}) = \{a_0\}$, $out(s_i, \alpha) = s_{i+1}$ for $0 \leqslant i \leqslant 2k$, and $out(s_{2k+1}, \alpha) = s_{2k+1}$, and $\{x_i\} = L(s_i)$ for $0 \leqslant i \leqslant 2k+1$. The model is a chain of states of length $2k+1$ such that each next step is reachable via actions a_0, ...,a_{2k} of agent a, and there is a self-loop labelled with a_0 in the last state s_{2k+1}. Each state satisfies exactly one propositional variable x_i.

Coalitional action model $\mathsf{AddChain}i$ is a tuple $(\mathsf{S}, \mathsf{Act}, \mathsf{act}, \mathsf{out}, \mathsf{L})$, where $\mathsf{S} = \{\mathsf{s}^i_j \mid 0 \leqslant j \leqslant i\} \cup \{\mathsf{s}^i_*\}$, $\mathsf{Act} = \{a_i \mid 0 \leqslant i \leqslant 2k\}$, $\mathsf{act}(a, \mathsf{s}^i_j) = \{a_l \mid 1 \leqslant l \leqslant i\}$ for $j \neq i$ and $j \neq 0$, $\mathsf{act}(a, \mathsf{s}^i_0) = \mathsf{Act}$, $\mathsf{act}(a, \mathsf{s}^i_*) = \{a_l \mid 0 \leqslant l \leqslant 2k \text{ and } l \neq i\}$, $\mathsf{out}(\mathsf{s}^i_j, a_l) = \mathsf{s}^i_{j+1}$ for $0 \leqslant j < i$, $1 \leqslant l \leqslant i$ and $l \neq 0$, $\mathsf{out}(\mathsf{s}^i_0, a_l) = \mathsf{s}^i_*$ for $l = 0$ and $l > i$, $\mathsf{out}(\mathsf{s}^i_i, a_i) = \mathsf{s}^i_i$, $\mathsf{out}(\mathsf{s}^i_*, a_j) = \mathsf{s}^i_*$ for $0 \leqslant j \leqslant 2k$ and $j \neq i$, $\mathsf{pre}(\mathsf{s}^i_j) = x_j$ for $0 \leqslant j \leqslant i$, and $\mathsf{pre}(\mathsf{s}^i_*) = \neg x_0$. Action model $\mathsf{AddChain}i$ is a chain of length i where each next state is reachable via all actions a_i excluding a_0, the final state in the chain has a self loop labelled with a_i, and a special state $\mathsf{s}^{\mathsf{i}}_*$ is reachable from the first state of the chain via a_0 and all a_j such that $j > i$. The intuition behind the action models is that $\mathsf{AddChain}i$'s add chains of length i to M_s meaning that variable x_i has been set to 1. Moreover, all other chains that were already in a CGM are not affected.

Coalitional action model Copy is a tuple $(\mathsf{S}, \mathsf{Act}, \mathsf{act}, \mathsf{out}, \mathsf{L})$, where $\mathsf{S} = \{\mathsf{t}\}$, $\mathsf{Act} = \{a_i \mid 0 \leqslant i \leqslant 2k\}$, $\mathsf{act}(a, \mathsf{t}) = \mathsf{Act}$, $\mathsf{out}(\mathsf{t}, \alpha) = \mathsf{t}$, and $\mathsf{pre}(\mathsf{t}) = \top$. Action model Copy just copies a given model so that no new chain appears meaning that the current x_i has been set to 0.

Finally, we translate $\psi(x_1, \dots, x_{2k})$ by substituting every x_i with $(\langle\!\langle a \rangle\!\rangle)^i [\![a]\!] x_i$, where $(\langle\!\langle a \rangle\!\rangle)^i$ is a stack of size i of $\langle\!\langle a \rangle\!\rangle$'s. In the resulting translated formula, subformula $(\langle\!\langle a \rangle\!\rangle)^i [\![a]\!] x_i$ holds if in a model there is a chain of length i with a loop at the end. This means that variable x_i has been set to 1.

As an example, consider a QBF $\forall x_1 \exists x_2 (x_1 \to x_2)$. We translate the formula into a CAML formula

$$[\mathsf{AddChain1}_{\mathsf{s}^1_0} \cup \mathsf{Copy}_{\mathsf{t}}]\langle \mathsf{AddChain2}_{\mathsf{s}^2_0} \cup \mathsf{Copy}_{\mathsf{t}}\rangle (\langle\!\langle a \rangle\!\rangle [\![a]\!] x_1 \to \langle\!\langle a \rangle\!\rangle \langle\!\langle a \rangle\!\rangle [\![a]\!] x_2).$$

The corresponding model M and action models $\mathsf{AddChain1}$, $\mathsf{AddChain2}$, and Copy are presented in Figure 6.

According to the semantics,

$$M_s \models [\mathsf{AddChain1}_{\mathsf{s}^1_0} \cup \mathsf{Copy}_{\mathsf{t}}]\langle \mathsf{AddChain2}_{\mathsf{s}^2_0} \cup \mathsf{Copy}_{\mathsf{t}}\rangle (\langle\!\langle a \rangle\!\rangle [\![a]\!] x_1 \to \langle\!\langle a \rangle\!\rangle \langle\!\langle a \rangle\!\rangle [\![a]\!] x_2)$$

if and only if

$$M_s \models [\mathsf{AddChain1}_{\mathsf{s}^1_0}]\langle \mathsf{AddChain2}_{\mathsf{s}^2_0} \cup \mathsf{Copy}_{\mathsf{t}}\rangle (\langle\!\langle a \rangle\!\rangle [\![a]\!] x_1 \to \langle\!\langle a \rangle\!\rangle \langle\!\langle a \rangle\!\rangle [\![a]\!] x_2)$$

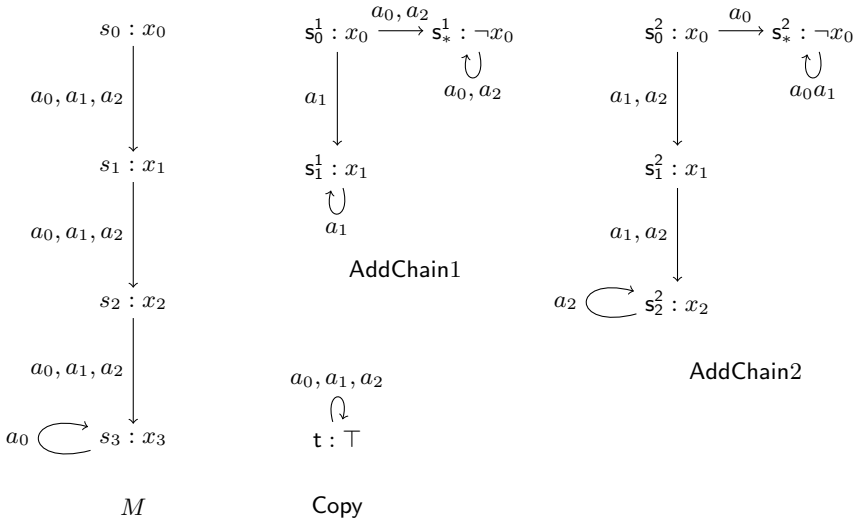

Fig. 6. Model M, and action models AddChain1, AddChain2, and Copy.

and

$$M_s \models [\mathsf{Copy_t}]\langle \mathsf{AddChain2}_{\mathsf{s}_0^2} \cup \mathsf{Copy_t}\rangle(\langle\!\langle a\rangle\!\rangle [\![a]\!] x_1 \to \langle\!\langle a\rangle\!\rangle\langle\!\langle a\rangle\!\rangle [\![a]\!] x_2).$$

In other words, $\langle \mathsf{AddChain2}_{\mathsf{s}_0^2} \cup \mathsf{Copy_t}\rangle(\langle\!\langle a\rangle\!\rangle [\![a]\!] x_1 \to \langle\!\langle a\rangle\!\rangle\langle\!\langle a\rangle\!\rangle [\![a]\!] x_2)$ should hold in both $M^{\mathsf{AddChain1}}_{(s_0,\mathsf{s}_0^1)}$ and $M^{\mathsf{Copy}}_{(s_0,\mathsf{t})}$. Updated model $M^{\mathsf{AddChain1}}_{(s_0,\mathsf{s}_0^1)}$ is depicted in Figure 7, and model $M^{\mathsf{Copy}}_{(s_0,\mathsf{t})}$ will just copy M, so we do not provide the figure.

Now, for each of $M^{\mathsf{AddChain1}}_{(s_0,\mathsf{s}_0^1)}$ and $M^{\mathsf{Copy}}_{(s_0,\mathsf{t})}$ there must be a subsequent update with either AddChain2 or Copy such that $\langle\!\langle a\rangle\!\rangle [\![a]\!] x_1 \to \langle\!\langle a\rangle\!\rangle\langle\!\langle a\rangle\!\rangle [\![a]\!] x_2$ will hold in the resulting model.

The result of updating M^{Copy} with AddChain2 is shown in Figure 7. Note that $M^{\mathsf{Copy,AddChain2}}_{(s_0,\mathsf{t},\mathsf{s}_0^2)} \models \langle\!\langle a\rangle\!\rangle [\![a]\!] x_1 \to \langle\!\langle a\rangle\!\rangle\langle\!\langle a\rangle\!\rangle [\![a]\!] x_2$ as the antecedent is not satisfied. Also observe that $M^{\mathsf{Copy,Copy}}_{(s_0,\mathsf{t},\mathsf{t})}$ would satisfy the formula for the same reason. All in all, this corresponds to setting x_1 to 0 in the original QBF, and thus the QBF will be true irregardless of the value of x_2.

Consider updated model $M^{\mathsf{AddChain1}}_{(s_0,\mathsf{s}_0^1)}$. It has only chains of lengths 1 and 3, and thus we have that $M^{\mathsf{AddChain1}}_{(s_0,\mathsf{s}_0^1)} \models \langle\!\langle a\rangle\!\rangle [\![a]\!] x_1$ and at the same time $M^{\mathsf{AddChain1}}_{(s_0,\mathsf{s}_0^1)} \not\models \langle\!\langle a\rangle\!\rangle\langle\!\langle a\rangle\!\rangle [\![a]\!] x_2$. So, $M^{\mathsf{AddChain1}}_{(s_0,\mathsf{s}_0^1)}$ does not satisfy formula $\langle\!\langle a\rangle\!\rangle [\![a]\!] x_1 \to \langle\!\langle a\rangle\!\rangle\langle\!\langle a\rangle\!\rangle [\![a]\!] x_2$. Hence, updating it with Copy would also result in a model, where the formula is not satisfied. This corresponds to choosing value 1 for x_1 in our QBF, and setting x_2 to 0 will not make the QBF true. However, choosing 1 for x_2 satisfies the formula. In terms of updated models this corresponds to updating $M^{\mathsf{AddChain1}}_{(s_0,\mathsf{s}_0^1)}$ with AddChain2, and the result of such an update is depicted in Figure 8. Note that in Figure 8 we take the connected component that includes state $(s_0, \mathsf{s}_0^1, \mathsf{s}_0^2)$,

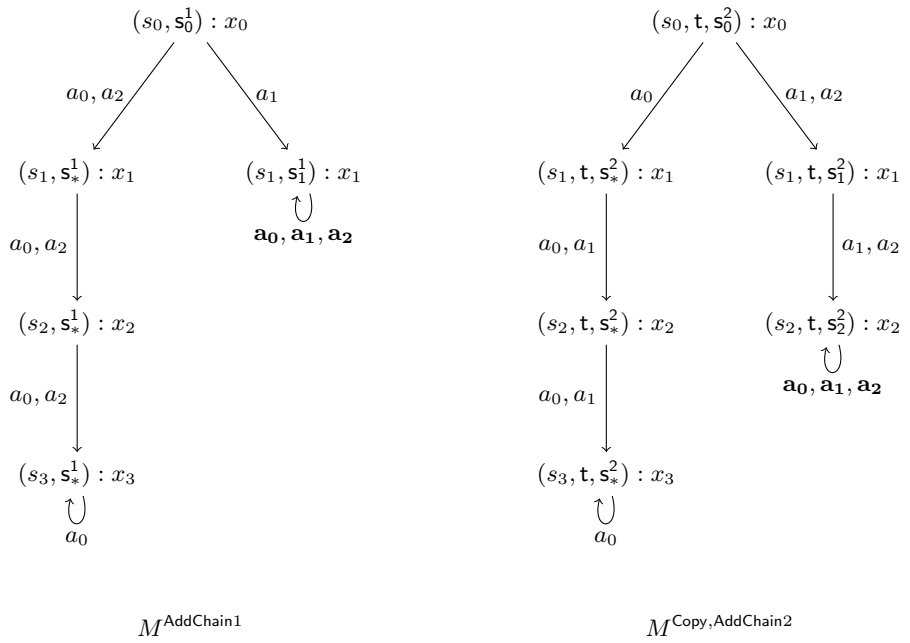

Fig. 7. Updated models $M^{\mathsf{AddChain1}}$ and $M^{\mathsf{Copy,AddChain2}}$ with added action profiles in bold font.

and we disregard state $(s_1, \mathsf{s}_1^1, \mathsf{s}_*^2)$ that will not be connected to the chosen component. It is clear that $M^{\mathsf{AddChain1,AddChain2}}_{(s_0, \mathsf{s}_0^1, \mathsf{s}_0^2)}$, which corresponds to setting both x_1 and x_2 to 1, satisfies $\langle\!\langle a \rangle\!\rangle [\![a]\!] x_1 \to \langle\!\langle a \rangle\!\rangle \langle\!\langle a \rangle\!\rangle [\![a]\!] x_2$.

Our construction mimics QBFs in the following way. For a universal quantifer $\forall x_i$ we use $[\mathsf{AddChain}i_{\mathsf{s}_0^i} \cup \mathsf{Copy}_{\mathsf{t}}]$ that corresponds to producing an updated model with a chain of length i, setting x_i to 1, and an updated model without such a chain, setting x_i to 0. In the case of $\exists x_i$, the choice between $\mathsf{AddChain}i_{\mathsf{s}_0^i}$ and $\mathsf{Copy}_{\mathsf{t}}$ is existential, which is expressed by $\langle \mathsf{AddChain}i_{\mathsf{s}_0^i} \cup \mathsf{Copy}_{\mathsf{t}} \rangle$. As a result of such a choice, we will have an updated model with a chain of length i, or an updated model without such a chain. □

Remark 1. Our hardness reduction relied heavily on non-deterministic choice, i.e. constructs $[\pi \cup \rho]$ and $\langle \pi \cup \rho \rangle$. As we have already mentioned in Proposition 1, item five, we can equivalently rewrite formulas with unions to formulas without it. This rewriting, however, can result in a formula of exponential size. We leave the problem of determining hardness of model checking CAML without union open, and conjecture that it is still *PSPACE*-hard. On a similar note, a more complicated construction than the one used in [8] was employed to show that DEL without union is *PSPACE*-hard [17, Theorem 4].

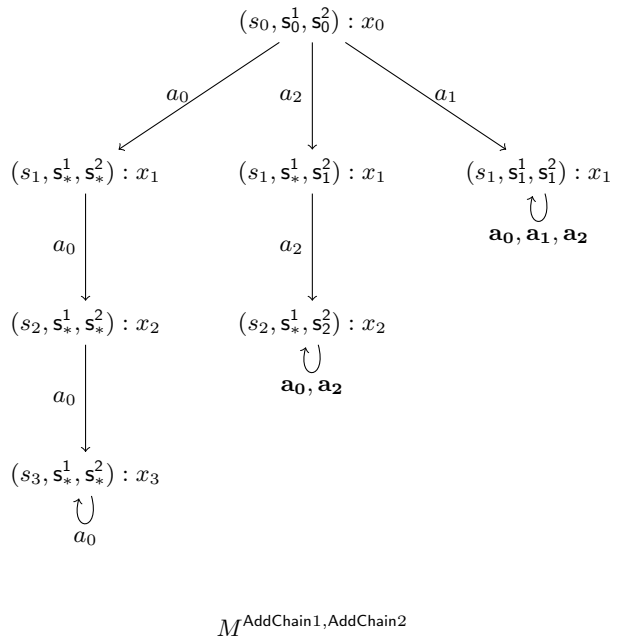

Fig. 8. Updated model $M^{\mathsf{AddChain1},\mathsf{AddChain2}}$ with added action profiles in bold font.

6 Discussion

We presented coalition action model logic (CAML) for reasoning about how agents' abilities change as a result of updating a CGM with a coalitional action model. Even though we took inspiration from DEL, CAML turned out quite different. In particular, CAML is strictly more expressive than the base CL, and thus no reduction axioms are possible. We also proved that CAML is incomparable to ATL, and conjecture that the same holds for other logics for reasoning about strategic abilities, namely ATL* and SL. Finally, we investigated the complexity of the model checking problem for CAML, and showed that it is *PSPACE*-complete even in the case of a single agent.

Since this is the first proposal of DEL-like action models for CGMs, there is a plethora of open questions. First, the non-existence of reduction axioms leaves open the problem of providing a sound and complete axiomatisation of CAML. Moreover, it is also worthwhile to investigate coalitional action models with postconditions (similar to those considered in [14]), i.e. action models that allow changing valuations of propositional variables. While we expect that postconditions will not affect the complexity of model-checking, expressivity results may turn out to be more surprising. Taking into account that updated models are *refinements* [10] of the corresponding original models, it is also quite tempting to investigate the problem of *synthesis* [18, 13] of coalitional action models:

having a starting CGM M and a desired target model N, synthesize a coalitional action model M such that $M^{\mathsf{M}} = N$.

Another avenue of further research is having a more expressive base language than CL. In particular, we plan to use action models with ATL and ATL*. Apart from that, we had to make a design decision that whenever the result of executing an action profile is undefined (or, there is a conflict between the existing model and a proposed modification), then a given system remains in the same state. However, there may be other intuitively natural ways to handle situations like that. Moreover, our action models are quite conservative in the sense that they neither grant agents new actions nor revoke any actions. It would be exciting to come up with action models that affect agents' sets of available actions.

Yet another avenue of further research is adding a temporal dimension to our framework in the vein of [20, 26]. Such an extension would allow us to store the *history of updates* directly in a model. This approach was investigated for the case of DDCL in [16], where it is used to reason about smart contract upgrades on blockchain structures.

Acknowledgments We would like to thank anonymous reviewers of AiML 2022 and DaLí 2022 for their careful reading of the paper and encouraging comments. We would also like to thank attendees of DaLí 2022 for fruitful discussions, and in particular Maksim Gladyshev for pointers to relevant literature.

References

1. Ågotnes, T., Goranko, V., Jamroga, W.: Alternating-time temporal logics with irrevocable strategies. In: Samet, D. (ed.) Proceedings of the 11th TARK. pp. 15–24 (2007). https://doi.org/10.1145/1324249.1324256
2. Ågotnes, T., Goranko, V., Jamroga, W., Wooldridge, M.: Knowledge and ability. In: van Ditmarsch, H., Halpern, J.Y., van der Hoek, W., Kooi, B. (eds.) Handbook of Epistemic Logic, pp. 543–589. College Publications (2015)
3. Ågotnes, T., van der Hoek, W., Juan A. Rodriguez-Aguilar, C.S., Wooldridge, M.: On the logic of normative systems. In: Veloso, M.M. (ed.) Proceedings of the 20th IJCAI. pp. 1175–1180 (2007)
4. Ågotnes, T., van der Hoek, W., Wooldridge, M.: Robust normative systems and a logic of norm compliance. Logic Journal of the IGPL **18**(1), 4–30 (2010). https://doi.org/10.1093/jigpal/jzp070
5. Alur, R., Henzinger, T.A., Kupferman, O.: Alternating-time temporal logic. Journal of the ACM **49**, 672–713 (2002). https://doi.org/10.1145/585265.585270
6. Areces, C., Fervari, R., Hoffmann, G.: Relation-changing modal operators. Logic Journal of the IGPL **23**(4), 601–627 (2015). https://doi.org/10.1093/jigpal/jzv020
7. Arora, S., Barak, B.: Computational Complexity: A Modern Approach. CUP (2009)
8. Aucher, G., Schwarzentruber, F.: On the complexity of dynamic epistemic logic. In: Schipper, B.C. (ed.) Proceedings of the 14th TARK (2013)
9. Baltag, A., Moss, L.S.: Logics for epistemic programs. Synthese **139**(2), 165–224 (2004). https://doi.org/10.1023/B:SYNT.0000024912.56773.5e

10. Bozzelli, L., van Ditmarsch, H., French, T., Hales, J., Pinchinat, S.: Refinement modal logic. Information and Computation **239**, 303–339 (2014). https://doi.org/10.1016/j.ic.2014.07.013
11. Bulling, N., Dix, J., Jamroga, W.: Model checking logics of strategic ability: Complexity. In: Dastani, M., Hindriks, K.V., Meyer, J.J.C. (eds.) Specification and Verification of Multi-agent Systems, pp. 125–159. Springer (2010). https://doi.org/10.1007/978-1-4419-6984-2_5
12. van Ditmarsch, H., van der Hoek, W., Kooi, B.: Dynamic Epistemic Logic, Synthese Library, vol. 337. Springer (2008)
13. van Ditmarsch, H., van der Hoek, W., Kooi, B., Kuijer, L.B.: Arrow update synthesis. Information and Computation **275** (2020). https://doi.org/10.1016/j.ic.2020.104544
14. van Ditmarsch, H., Kooi, B.: Semantic results for ontic and epistemic change. In: Bonanno, G., van der Hoek, W., Wooldridge, M. (eds.) Proceedings of the 7th LOFT. Texts in Logic and Games, vol. 3, pp. 88–101. Amsterdam University Press (2008)
15. Galimullin, R., Ågotnes, T.: Dynamic coalition logic: Granting and revoking dictatorial powers. In: Ghosh, S., Icard, T. (eds.) Proceedings of the 8th LORI. LNCS, vol. 13039, pp. 88–101. Springer (2021). https://doi.org/10.1007/978-3-030-88708-7_7
16. Galimullin, R., Ågotnes, T.: Coalition logic for specification and verification of smart contract upgrades. In: Proceedings of the 24th PRIMA. p. (to appear) (2022)
17. de Haan, R., van de Pol, I.: On the computational complexity of model checking for dynamic epistemic logic with S5 models. FLAP **8**(3), 621–658 (2021)
18. Hales, J.: Arbitrary action model logic and action model synthesis. In: Proceedings of the 28th LICS. pp. 253–262. IEEE Computer Society (2013). https://doi.org/10.1109/LICS.2013.31
19. Halpern, J.Y., Moses, Y.: A guide to completeness and complexity for modal logics of knowledge and belief. Artificial Intelligence **54**(2), 319–379 (1992). https://doi.org/10.1016/0004-3702(92)90049-4
20. Hoshi, T.: Merging DEL and ETL. Journal of Logic, Language and Information **19**(4), 413–430 (2010). https://doi.org/10.1007/s10849-009-9116-7
21. Kooi, B., Renne, B.: Arrow update logic. Review of Symbolic Logic **4**(4), 536–559 (2011). https://doi.org/10.1017/S1755020311000189
22. Mogavero, F., Murano, A., Vardi, M.Y.: Reasoning about strategies. In: Lodaya, K., Mahajan, M. (eds.) Proceedings of the 30th FSTTCS. LIPIcs, vol. 8, pp. 133–144. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2010). https://doi.org/10.4230/LIPIcs.FSTTCS.2010.133
23. Pauly, M.: Logic for Social Software. Ph.D. thesis, ILLC, University of Amsterdam, The Netherlands (2001)
24. Pauly, M.: A modal logic for coalitional power in games. Journal of Logic and Computation **12**(1), 149–166 (2002). https://doi.org/10.1093/logcom/12.1.149
25. Plaza, J.: Logics of public communications. Synthese **158**(2), 165–179 (2007). https://doi.org/10.1007/s11229-007-9168-7
26. Renne, B., Sack, J., Yap, A.: Logics of temporal-epistemic actions. Synthese **193**(3), 813–849 (2016). https://doi.org/10.1007/s11229-015-0773-6
27. Shoham, Y., Tennenholtz, M.: On the synthesis of useful social laws for artificial agent societies. In: Proceedings of the 10th AAAI. pp. 276–281 (1992)
28. Shoham, Y., Tennenholtz, M.: On social laws for artificial agent societies: Offline design. In: Agre, P.E., Rosenschein, S.J. (eds.) Computational Theories of Interaction and Agency, pp. 597–618. MIT Press (1996)