Dictatorial Dynamic Coalition Logic*

Rustam Galimullin† Thomas Ågotnes‡

Abstract

Coalition logic (CL) allows one to reason about what a coalition of agents may bring about no matter what agents outside of the coalition do at the same time. To study the dynamics of coalitional ability, we enrich CL with model-changing operators that update coalitional abilities of single agents by either granting the agents dictatorial powers, or revoking them. We investigate the model checking problem for the two extensions. Moreover, we compare their expressive power relative to each other, and situate them in the broader context of the logics for reasoning about strategic abilities.

1 Introduction

One of the most exciting manifestations of the dynamic turn in logic [8] is dynamic epistemic logic (DEL) [11] that encompasses various formalisms for modelling *information change*. In particular, DELs capture *the dynamics of knowledge* by allowing one to reason about how agents' knowledge and beliefs change as a result of different types of epistemic events.

The interplay between *knowledge* and *ability* has been in the focus of computer science and philosophy research for quite a while [3] with notable examples being alternating-time temporal epistemic logic [17] and epistemic coalition logic [1]. This research, however, focused mostly on how agents' knowledge and ignorance affect their choice of available strategies, and did not capture how those available strategies may change.

In the current paper, we propose a study of *the dynamics of ability,* where ability-changing operations are treated as first-class citizens. One may view *ability change* as an update of a protocol, contract, or an agreement between agents, that specifies how what they can and cannot do should be modified. In this regard, we follow the lead of DEL, where knowledge-changing actions are explicitly expressed in a language.

Apart from a purely philosophical interest of 'dynamifying' logics of ability, there is also a very practical interest. Reasoning about actions and abilities of single agents and groups

*Extended version of [14].

†University of Bergen, Bergen, Norway; rustam.galimullin@uib.no

‡Southwest University, Chongqing, China, and University of Bergen, Bergen, Norway; thomas.agotnes@uib.no

of agents is ubiquitous in AI. Notable examples are classical and epistemic [9] multi-agent planning, game theory, software verification, and so on. One of the most recent challenges is verification of the safety of blockchains, and, in particular, smart contracts [16].

To simplify an example problem from [16], assume that there is a newly-founded company, and the initial block of the smart contract specifies that the board of directors consists of Alice, Bob, and Carol, and all financial decisions are made according to the majority rule. Apart from the board of directors, there are also employees, Dave and Ellen, with fewer privileges. Now assume that Bob was caught making suspicious transactions and, according to some financial regulation, they can no longer be on the board. Moreover, to substitute Bob, Ellen was promoted. The resulting new situation is recorded in the next block of the blockchain, where it is specified that Bob loses the ability to make financial decisions, while Ellen obtains such an ability.

Clearly, in the described scenario, it is vital to specify what an agent, or a group thereof, is able or unable to do. One of the most popular languages for reasoning about abilities of groups of agents is called *coalition logic* (CL) [23] (which can be considered as a Next-time fragment of alternating-time temporal logic (ATL) [4]). CL extends propositional logic with constructs $\langle\langle C\rangle\rangle\varphi$ meaning that 'there is a joint action by agents from coalition C such that no matter what agents outside of the coalition do, φ'.

While CL captures the abilities of agents to force certain outcomes, it provides only a static snapshot and thus is inadequate for the situations where new policies or regulations override agents' abilities. We thus propose the development and study of *dynamic coalition logic*, with dynamic operators in the spirit of dynamic epistemic logics[1] [11] that can modify or update the abilities of agents and coalitions. In the current paper we take the first step and focus on granting and revoking *dictatorial powers*, i.e. the ability of single agents to force an outcome.

To model updates of dictatorial powers, we borrow syntax and basic intuition from arrow update logic (AUL) [18], where constructs $U = \{(\chi_1, a_1, \psi_1), \ldots, (\chi_n, a_n, \psi_n)\}$ specify which belief relations should be preserved in a current model. AUL, being a dynamic epistemic logic [11], models such epistemic events as public and private announcements, lying, etc. Arrow updates were also used to reason about norms [19].

First, we consider *positive dictatorial dynamic coalition logic* (DDCL$^+$) that extends CL with updates $+U = \{(\chi_1, a_1, \psi_1)^+, \ldots, (\chi_n, a_n, \psi_n)^+\}$. In this case, $+U$ specifies between which states an agent should be granted the dictatorial power. In particular, $(\chi, a, \psi)^+$ means that agent a will be able to force any state where ψ is true, from any state where χ holds. In terms of models, this means that in the updated model there will be a set of new arrows satisfying the requirement. In this regard, DDCL$^+$ is slightly reminiscent of bridge logics [5]. However, in our case, the interpretation of arrows and the mechanism of adding relations are completely different.

Apart from the logic of granting dictatorial powers, we also study the logic of revoking

[1]Dynamic epistemic logics with coalitional operators have been studied only in the setting of public announcements (see [12, 13]). These logics, however, are not strictly coalitional in the sense of [23] since they are defined on epistemic models, not concurrent game models.

such powers, which we call *negative dictatorial dynamic coalition logic* (DDCL$^-$). The logic extends CL with updates $-U = \{(\chi_1, a_1, \psi_1)^-, \ldots, (\chi_n, a_n, \psi_n)^-\}$ that, similarly to the updates of AUL, specify which dictatorial powers should be preserved, while all other such powers, not satisfying the specification, are removed.

After we recall some background information about CL in Section 2, we present syntax and semantics of DDCL$^+$ and DDCL$^-$ (Section 3). In particular, we argue that updates $+U$ and $-U$ are not always executable. In Section 4, which is entirely new, we show that the complexity of the model checking problem is P-complete for both DDCL$^+$ and DDCL$^-$. This result comes hand in hand with the fact, demonstrated in Section 5, that both logics are strictly more expressive than CL. Thus, not only there cannot be reduction axioms for the logics in the fashion of AUL, but also this additional expressivity comes 'for free', at least for the case of model checking. In Section 5, we additionally show that DDCL$^+$ and DDCL$^-$ are incomparable w.r.t. expressive power, and then situate the logics in the wider context of logics for reasoning about strategic abilities. Finally, we discuss further research in Section 6.

2 Coalition Logic

As the logics introduced in this paper are dynamic extensions of coalition logic, we first provide all the necessary background information on it (see [23, 22, 3]). Let P be a countable set of propositional variables, and A be a finite set of agents.

Definition 2.1. The *language of coalition logic* $\mathcal{CL}$ is given recursively by the following grammar:

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

where $p \in P$ and $C \subseteq A$. Constructs $\langle\langle C \rangle\rangle \varphi$ 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 $[\![C]\!]\varphi := \neg\langle\langle C \rangle\rangle \neg\varphi$.

Formulas of coalition logic are interpreted on concurrent game models.

Definition 2.2. A *concurrent game model* (CGM)[2], or a *model*, is a tuple $M = (A, S, Act, act, out, L)$. A is a non-empty finite set of agents, and the subsets of A are called *coalitions*. 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

[2]Note that although in [23] the semantics of CL are given relative to effectivity models, we still can use CGMs as these types of models are semantically equivalent (see more on this topic in [15]).

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, $L : S \to P$ is the valuation function.

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

Definition 2.3. Let M_s be a pointed CGM. The semantics of CL are defined as follows:

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

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. Given φ and M, we define $[\![\varphi]\!]_M := \{s \in S \mid M_s \models \varphi\}$.

Definition 2.4. We call a formula φ *valid* if for all M_s it holds that $M_s \models \varphi$.

Definition 2.5. Let $M = (A, S^M, Act^M, act^M, out^M, L^M)$ and $N = (A, S^N, Act^N, act^N, out^N, L^N)$ 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 M linking states s_1 and s_2, we call the pointed models *bisimilar* ($M_{s_1} \leftrightarrows N_{s_2}$).

Theorem 1 ([2])**.** Let M and N be CGMs such that $M \leftrightarrows N$ and there is a bisimilation between $s \in S^M$ and $t \in S^N$. Then for all $\varphi \in \mathcal{CL}$, $M_s \models \varphi$ iff $N_t \models \varphi$.

Before we continue, we define an auxiliary set of forcing actions for each state and agent. Intuitively, an action is a forcing action if all action profiles it appears in lead to the same state.

Definition 2.6. Let M be a CGM. The set of *forcing actions* for agent i and state s, denoted as $\mathfrak{f}(i,s)$, is defined as follows:

$$\{\alpha_i \in act(i,s) \mid \forall \alpha, \beta \in act(s) : (\alpha_i \sqsubseteq \alpha \text{ and } \alpha_i \sqsubseteq \beta) \text{ implies } out(\alpha, s) = out(\beta, s)\}$$

Without loss of generality and to make the following technical presentation clearer, we assume that each action in the set of forcing actions is labelled with a pair of states it connects. Thus, elements of $\mathfrak{f}(i,s)$ are $a_i^{(s,t_1)}$, $b_i^{(s,t_2)}$, ...

3 Dictatorial Dynamic Coalition Logic

In this section we introduce the ways of granting and revoking dictatorial powers of agents. We borrow the syntax from arrow update logic [18].

3.1 Granting Dictatorial Powers

Definition 3.1. The *language of positive dictatorial dynamic coalition logic* $\mathcal{DDCL}^+$ is given by the following BNF:

$$\begin{array}{lcl} \varphi & ::= & p \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \langle\langle C \rangle\rangle \varphi \mid [+U]\varphi \\ +U & ::= & (\varphi, a, \varphi)^+ \mid (\varphi, a, \varphi)^+, +U \end{array}$$

where $p \in P$, $a \in A$, and $C \subseteq A$. Constructs $+U$ are called *(positive) updates*, and formulas $[+U]\varphi$ are read as 'after (positive) update $+U$, φ is true'. We will abuse the notation and treat the list $+U := (\psi_1, a_1, \varphi_1)^+, \ldots, (\psi_n, a_n, \varphi_n)^+$ as the set $+U := \{(\psi_1, a_1, \varphi_1)^+, \ldots, (\psi_n, a_n, \varphi_n)^+\}$. The dual of $[+U]\varphi$ is $\langle +U\rangle\varphi := \neg[+U]\neg\varphi$.

The intended meaning of $(\varphi, a, \psi)^+$ is as follows: in each φ-state, in the updated model, there will be a *new* action for agent a such that no matter which actions other agents choose, the target state is a ψ-state. In case of multiple φ- and ψ-states, we have a new action for each pair.

Example 1. Before giving the formal definition of the semantics, let us consider an example. In Figure 1, in model M there are three states, s, t, and u, and two agents, a and b. In s, agent a has three actions, a_0, a_1, and a_2, and she has the ability to decide which state will be next. Agent b does not have the ability to force anything in the model.

To make the example more relatable, assume that p means that agent a has a cup of coffee, and q stands for b having coffee. Action a_1 signifies agent a pouring coffee just for herself; if she chooses a_2, then she pours coffee for b as well; and actions a_0 and b_0 are 'do nothing', or 'enjoy oneself' actions. It is clear from the figure that in s, where neither a nor b have coffee, a can choose to either get a cup for herself (transition to state u), or for both of them (transition to t), or just do nothing (self-loop in s). Agent b, being a polite guest of a, cannot do anything.

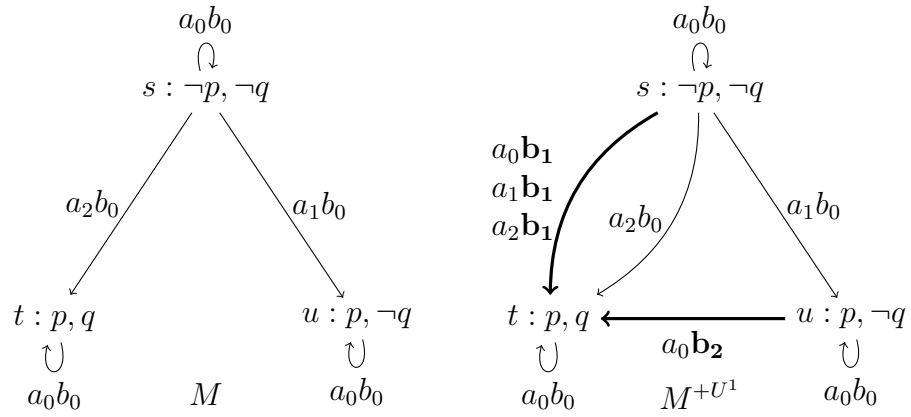

Figure 1: Models M (left) and M^{+U^1} (right), where thick arrows depict new transitions, and new actions are in bold font.

The set of forcing actions for a in s is $\mathfrak{f}(a, s) = \{a_0^{(s,s)}, a_1^{(s,u)}, a_2^{(s,t)}\}$ and the set of forcing actions for b in s is empty. In states t and u both a and b have one forcing action each: a has a_0 and b has b_0.

Now, let us consider $+U^1 = \{(\neg q, b, q)^+\}$. Informally, we want to give agent b the power to get a cup of coffee whenever she does not have one. The result of updating our model with $+U^1$ is presented in Figure 1 on the right. In the figure, agent b gains two new actions (in bold font in the figure): b_1 in s, and b_2 in u. Then, for each action profile with b_1 or b_2 there is a transition (depicted by thick arrows) to state t, where q is true. This means that after the update, agent b has the dictatorial power to force q from any of the states where q does not hold. Formally, we have, for example, that $M_s \not\models \langle\langle b \rangle\rangle q$ and $M_s \models [+U^1]\langle\langle b \rangle\rangle q$. With the update, sets of forcing actions are changed as well. In state s, $\mathfrak{f}(a, s) = \{a_2^{(s,t)}\}$ and $\mathfrak{f}(b, s) = \{b_1^{(s,t)}\}$; in state t both a and b have $a_0^{(t,t)}$ and $b_0^{(t,t)}$ respectively; in state u, $\mathfrak{f}(b, u) = \{b_0^{(u,u)}, b_2^{(u,t)}\}$ and a does not have forcing actions.

As another example, consider $+U^2 = \{(\top, b, \neg p)^+\}$. We can imagine an informal reading as agent b gets rid of a's coffee no matter what, and precludes her from getting one if she hasn't got one yet (state s). The update of the initial model is depicted in Figure 2 on the left.

In the updated model, agent a does not have any strategy to escape $\neg p$, or, formally, $M_s \models [+U^2][\![a]\!]\neg p$. The non-empty sets of forcing actions in the updated model are the following: $\mathfrak{f}(a, s) = \{a_0^{(s,s)}\}$, $\mathfrak{f}(b, s) = \{b_1^{(s,s)}\}$, $\mathfrak{f}(b, t) = \{b_0^{(t,t)}, b_2^{(t,s)}\}$, and $\mathfrak{f}(b, u) = \{b_0^{(u,u)}, b_3^{(u,s)}\}$.

Before we continue with the definition of the semantics, it should be noted that not every update can be implemented. For example, in our initial model in Figure 1, we may require the following update: $+U = \{(\neg q, b, q)^+, (\neg p, a, p)^+\}$. In this case, we have a clash of control while assigning a transition from s to t: both a and b should have the ability to force t. See the model on the right in Figure 2 for a representation of

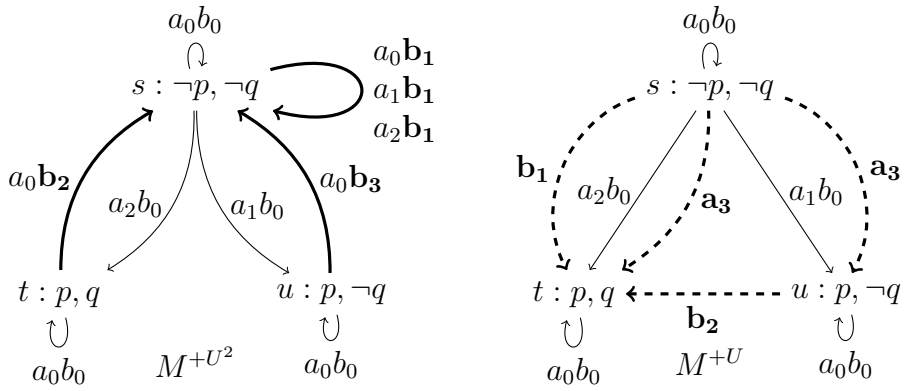

Figure 2: Model M^{+U^2} (left) and a tentative updated model M^{+U} (right), where thick arrows depict new transitions, dashed arrows depict new tentative transitions required by update $+U$, and new actions are in bold font. Observe that in M^{+U} both a and b require dictatorial powers in s.

the problem. To mitigate the problem, we consider only updates that are clash-free, or *executable*, throughout the given model.

Definition 3.2. Let M be a CGM, and $+U$ be an update. We call $+U$ *executable in* M iff for all $(\varphi, i, \psi)^+, (\chi, j, \tau)^+ \in +U$: $[\![\varphi]\!]_M \cap [\![\chi]\!]_M = \emptyset$ whenever $i \neq j$.

Informally, the definition says that an update is executable if it is not granting dictatorial powers to different agents in the same state.

Definition 3.3. Let $M_s = (A, S, Act, act, out, L)$ be a CGM. The semantics of DDCL$^+$ extends Definition 2.3 with the following clause for updates:

$$M_s \models [+U]\varphi \quad \text{iff} \quad +U \text{ is executable in } M \text{ implies } M_s^{+U} \models \varphi$$

where $M^{+U} = (A, S, Act^{+U}, act^{+U}, out^{+U}, L)$ is the *updated model*.

To define $M^{+U} = (A, S, Act^{+U}, act^{+U}, out^{+U}, L)$, we first define the set of *new* forcing actions for each agent i in each state s. Let $Pairs^{+U}_{(\varphi,i,\psi)^+} = \{s \mid M_s \models \varphi\} \times \{s \mid M_s \models \psi\}$ be all pairs of states between which we need to add transitions according to some $(\varphi, i, \psi)^+ \in +U$. The set of new forcing actions $\mathfrak{f}^{+U}(i, s)$ consists of actions $\alpha_k^{(s,t)}$ for each $(s, t) \in Pairs^{+U}_{(\varphi,i,\psi)^+}$ and all $(\varphi, i, \psi)^+ \in +U$, where k starts with $|act(i, s)| + 1$ and is increased by 1 for each such (s, t). Intuitively, the set of new forcing actions is constructed according to $+U$ such that each new action has a unique ordinal number k.

Then Act^{+U} is $Act \cup \bigcup_{i \in A, s \in S} \mathfrak{f}^{+U}(i, s)$. Function $act^{+U}(i, s) = act(i, s) \cup \mathfrak{f}^{+U}(i, s)$. Finally,

$$out^{+U}(\alpha, s) = \begin{cases} t, & \exists \alpha_i^{(s,t)} \sqsubseteq \alpha : \alpha_i^{(s,t)} \in \mathfrak{f}^{+U}(i, s) \text{ for some } i \in A \\ out(\alpha, s), & \text{otherwise} \end{cases}$$

Intuitively, $out^{+U}(\alpha, s)$ takes the system into state t if there is a forcing action of agent i labelled with (s,t) in action profile α, and works as the original $out(\alpha, s)$ if there are no new forcing actions in α.

To see that an updated model M^{+U} is indeed a model it is enough to argue that the function out^{+U} is well-defined. Consider arbitrary state s and action profile α that contains a new forcing action $\alpha_i^{(s,t)}$ by an agent i and that is executable in s. Due to the executability of $+U$, all new forcing actions in s belong to agent i. Since the updated set of actions $act^{+U}(i, s)$ of i will contain $\alpha_i^{(s,t)}$, the updated set of executable action profiles $act^{+U}(s)$ will have all possible α such that $\alpha_i^{(s,t)} \sqsubseteq \alpha$. Finally, according to the definition of out^{+U}, for all such α, $out^{+U}(\alpha, s)$ will return exactly state t. Thus, function out^{+U} is total and deterministic.

As the first step towards the systematic study of DDCL$^+$, we consider some valid and not valid formulas of the logic. In the proposition below, property 1 states that positive updates are monotonic operators. That we cannot, in general, unite or commute updates is captured by items 2 and 3. The intuition behind possible counterexamples is that such actions may result in updates that are not executable. Properties 4 and 5 claim that we cannot decompose a single update into a series of consecutive ones, and vice versa. However, such a decomposition is possible if the starting and target states are specified by formulas of propositional logic, as claimed by item 6[3].

Proposition 1. The following holds for formulas of DDCL$^+$.

1. $\langle +U\rangle\varphi \wedge \langle +U\rangle\psi \leftrightarrow \langle +U\rangle(\varphi \wedge \psi)$ is valid.
2. $\langle +U^1\rangle\varphi \wedge \langle +U^2\rangle\varphi \rightarrow \langle +U^1 \cup +U^2\rangle\varphi$ is not valid.
3. $\langle +U^1\rangle\langle +U^2\rangle\varphi \rightarrow \langle +U^2\rangle\langle +U^1\rangle\varphi$ is not valid.
4. $\langle (\psi_1, a, \chi_1)^+, (\psi_2, b, \chi_2)^+\rangle\varphi \rightarrow \langle (\psi_1, a, \chi_1)^+\rangle\langle (\psi_2, b, \chi_2)^+\rangle\varphi$ is not valid.
5. $\langle (\psi_1, a, \chi_1)^+\rangle\langle (\psi_2, b, \chi_2)^+\rangle\varphi \rightarrow \langle (\psi_1, a, \chi_1)^+, (\psi_2, b, \chi_2)^+\rangle\varphi$ is not valid.
6. $\langle (\psi_1, a, \chi_1)^+, (\psi_2, b, \chi_2)^+\rangle\varphi \rightarrow \langle (\psi_1, a, \chi_1)^+\rangle\langle (\psi_2, b, \chi_2)^+\rangle\varphi$, where ψ_1, χ_1, ψ_2, and χ_2 are propositional, is valid.

Proof. All the results can be shown by application of the definition of the semantics, or the intuition that some updates may become not executable if placed after some other update. Here we provide a brief proof sketch of items 4 and 6.

To see that formula in the item 4 is not valid, take $\varphi := \neg\langle\langle b\rangle\rangle\chi_2$ and $\psi_2 := \langle\langle a\rangle\rangle\chi_1$. Now, let M be a CGM such that $M_s \models \psi_1 \wedge \neg\psi_2 \wedge \varphi$. Since $\neg\psi_2$ is false in state s, update

[3]Items 4, 5, and 6 of Proposition 1 hint at possible interaction between DDCL$^+$ and a fragment thereof with only single-agent updates, where we grant dictatorial powers to one agent at a time. In particular, such updates are always executable: there is no clash of power if we consider only a single agent per update. We leave the discussion of the fragment and how it relates to DDCL$^+$ for the future.

$\langle(\psi_1, a, \chi_1)^+, (\psi_2, b, \chi_2)^+\rangle\varphi$ will grant dictatorial powers only to agent a in s. Thus, agent b will still be not able to force χ_2 after the update, satisfying φ. Next, let us consider the consecutive updates $\langle(\psi_1, a, \chi_1)^+\rangle\langle(\psi_2, b, \chi_2)^+\rangle\varphi$. After the first update, $\langle(\psi_1, a, \chi_1)^+\rangle$, agent a gains the ability to force χ_1, and therefore the updated model satisfies ψ_2. This leads to the fact that updating the model with the second update, $\langle(\psi_2, b, \chi_2)^+\rangle$, grants agent b with the power to force χ, which makes φ false.

To show that item 4 is not valid, we used the fact that some formulas in updates may change their truth values as a result of previous updates. Thus, earlier updates may 'unlock' later updates. To argue that formula in item 6 is valid, it is enough to notice that such unlocking is not relevant for propositional formulas as updates do not change the valuation of propositional variables. □

3.2 Revoking Dictatorial Powers

Apart from granting dictatorial powers, another way of updating coalitional abilities is by revoking such powers. Similarly to DDCL$^+$, we approach this problem from the perspective of arrow updates.

Definition 3.4. The *language of negative dictatorial dynamic coalition logic* $\mathcal{DDCL}^-$ is given by the following grammar:

$$\begin{array}{lcl} \varphi & ::= & p \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \langle\!\langle C\rangle\!\rangle\varphi \mid [-U]\varphi \\ -U & ::= & (\varphi, a, \varphi)^- \mid (\varphi, a, \varphi)^-, -U \end{array}$$

where $p \in P$, $a \in A$, and $C \subseteq A$. Constructs $-U$ are called *(negative) updates*, and formulas $[-U]\varphi$ are read as 'after (negative) update $-U$, φ is true'. We treat the list $-U$ as the set $-U := \{(\psi_1, a_1, \varphi_1)^-, \ldots, (\psi_n, a_n, \varphi_n)^-\}$. The dual of $[-U]\varphi$ is $\langle -U\rangle\varphi := \neg[-U]\neg\varphi$.

The idea behind $-U$ updates is similar to how arrow updates work in AUL [18]: the list $-U$ specifies which dictatorial powers should be *preserved*. In other words, for each χ-state where there are local dictatorial agents forcing a ψ-state, we check whether there is a corresponding $(\chi, i, \psi)^-$ in $-U$ for all such agents i. If yes, then we leave the corresponding arrows as they are; if not, we delete all arrows with the corresponding forcing action.

Example 2. Recall model M^{+U^2} in Figure 2 with the non-empty sets of forcing actions $\mathfrak{f}(a, s) = \{a_0^{(s,s)}\}$, $\mathfrak{f}(b, s) = \{b_1^{(s,s)}\}$, $\mathfrak{f}(b, t) = \{b_0^{(t,t)}, b_2^{(t,s)}\}$, and $\mathfrak{f}(b, u) = \{b_0^{(u,u)}, b_3^{(u,s)}\}$.

Now, assume that the abuse of power by b after update $+U^2$ was not tolerated in the office, and a new policy was issued specifying that once a has got a coffee, she can enjoy it in peace. The corresponding update is $-U^3 = \{(p, b, p)^-, (\neg p, b, \neg p)^-\}$, where the first clause preserves self-loops in states t and u, and the second clause preserves some of the self-loops in state s. The result of updating M^{+U^2} with $-U^3$ is shown in Figure 3 on the left.

To obtain the updated model $M^{+U^2,-U^3}$, we check for each state and each action profile whether it contains a forcing action by an agent. If it does, then we see whether the states

the corresponding arrow connects are marked by formulas satisfying one of the elements of $-U^3$, and whether the forcing action belong to an agent specified by that element of $-U^3$. If it is the case, we leave the profile as it is, if it is not, we remove the corresponding arrow.For example, M^{+U^2} had an arrow from p-state t to a $\neg p$-state s marked with action profile a_0b_2, where b_2 is a forcing action by b. None of $(p, b, p)^-$ and $(\neg p, b, \neg p)^-$ specifies that such a dictatorial power should be preserved, so in the resulting updated model $M^{+U^2,-U^3}$ the corresponding arrow is removed. The new sets of forcing actions in updated model $M^{+U^2,-U^3}$ are $\mathfrak{f}(a, s) = \{a_0^{(s,s)}\}$, $\mathfrak{f}(b, s) = \{b_1^{(s,s)}\}$, $\mathfrak{f}(a, t) = \{a_0^{(t,t)}\}$, $\mathfrak{f}(b, t) = \{b_0^{(t,t)}\}$, $\mathfrak{f}(a, u) = \{a_0^{(u,u)}\}$, and $\mathfrak{f}(b, u) = \{b_0^{(u,u)}\}$. Now it holds that $M_t^{+U^2} \models \langle\langle b\rangle\rangle \neg p \wedge [-U^3][\![b]\!]p$.

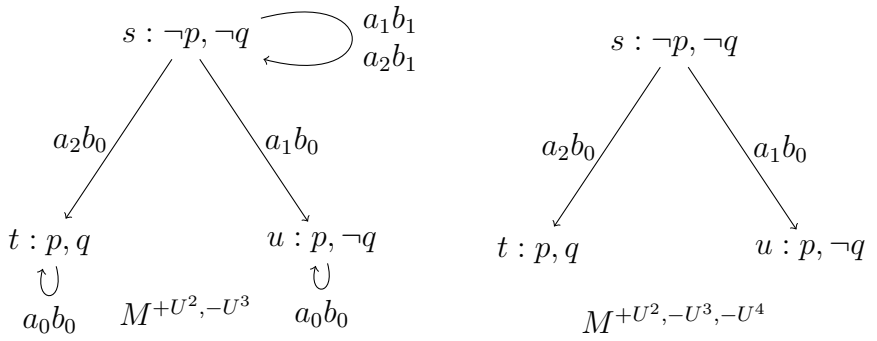

Figure 3: Model $M^{+U^2,-U^3}$ (left) and a tentative updated model $M^{+U^2,-U^3,-U^4}$ (right).

Similarly to the case of $+U$, we need to be careful with $-U$ since we do not want to end up in a situation, where an agent does not have any actions in some state, or, equivalently, some state does not have any outgoing arrows. Indeed, consider the further update of the model from Figure 3 with $-U^4 = \{(p \wedge \neg p, a, p \wedge \neg p)^-\}$ meaning that we are required to revoke all dictatorial powers from all the agents (since none of the states satisfy $p \wedge \neg p$). The resulting model would look like the one presented in Figure 3 on the right.

In the resulting tentative model there are no actions in states t and u, hence the structure in Figure 3 is not a CGM at all. To tackle this issue, we, once again, require a corresponding condition of executability.

We first specify which forcing actions should be *preserved* in $\mathfrak{f}(i, s)$ according to $-U$ in a given CGM M. Set $\mathfrak{f}^{-U}(i, s)$ of forcing actions to be preserved is defined as

$$\mathfrak{f}^{-U}(i, s) = \{\alpha_i^{(s,t)} \in \mathfrak{f}(i, s) \mid \exists(\varphi, i, \psi)^- \in -U : M_s \models \varphi \text{ and } M_t \models \psi\}$$

Intuitively, a forcing action $\alpha^{(s,t)}$ should be preserved if there is a $(\varphi, i, \psi)^- \in -U$ such that s satisfies φ, and t satisfies ψ.

Definition 3.5. Let M be a CGM, and $-U$ be an update. We call $-U$ *executable in* M iff for all $i \in A$ and $s \in S$ at least one of the following conditions is true:

- $|\mathfrak{f}^{-U}(i, s)| \neq 0$, or

- $\exists \alpha_i \in act(i,s) : \alpha_i \notin \mathfrak{f}(i,s)$

In other words, executability of $-U$ means that for all forcing actions, either we have a clause in $-U$ that allows us to preserve at least one forcing action from a state, or there are other, non-forcing, actions by the agent in the state. This ensures that agents do not run out of actions as a result of an update.

Definition 3.6. Let $M_s = (A, S, Act, act, out, L)$ be a CGM. The semantics of DDCL$^-$ extends Definition 2.3 with the following clause for updates:

$$M_s \models [-U]\varphi \quad \text{iff} \quad -U \text{ is executable in } M \text{ implies } M_s^{-U} \models \varphi$$

where $M^{-U} = (A, S, Act^{-U}, act^{-U}, out^{-U}, L)$ is the *updated model*.

We denote by $\mathfrak{f}^-(i,s)$ the set $\mathfrak{f}(i,s) \setminus \mathfrak{f}^{-U}(i,s)$ of forcing actions of agent i in state s to be *removed* from the model. Then function $act^{-U}(i,s) = act(i,s) \setminus \mathfrak{f}^-(i,s)$, and the updated set of executable action profiles is $act^{-U}(s)$. Set Act^{-U} is $\bigcup_{i \in A, s \in S} act^{-U}(i,s)$. Finally, $out^{-U}(\alpha, s)$ is restricted to those action profiles α that are in $act^{-U}(s)$.

To see that M^{-U} is a model, we argue that out^{-U} is a total function. First, notice that executability of $-U$ guarantees that for each agent there is at least one action left in each state. Now, assume to the contrary that for some state s and action profile α executable in s, the value of $out^{-U}(\alpha, s)$ is not defined. This means that α must contain a forcing action $\alpha_i^{(s,t)}$ by an agent i such that $\alpha_i^{(s,t)} \in \mathfrak{f}^-(i,s)$. However, according to the definition of $act^{-U}(i,s)$, $\alpha_i^{(s,t)}$ cannot appear within executable action profiles $act^{-U}(s)$. Hence, a contradiction.

One of the side-effects of considering forcing actions in the setting of ability updates is that if for a given model there are no forcing actions, then all $-U$'s are executable and none of $-U$ has any effect on the model. This result follows directly from the definition of the semantics.

Proposition 2. Let M be a CGM such that for all $i \in A$ and $s \in S$, $\mathfrak{f}(i,s) = \emptyset$. Then for any $-U$ and $\varphi \in \mathcal{DDCL}^-$ it holds that $M_s \models [-U]\varphi$ iff $M_s \models \varphi$.

Similarly to DDCL$^+$, we mention some properties of DDCL$^-$.

Proposition 3. The following holds for formulas of DDCL$^-$.

1. $\langle -U \rangle \varphi \wedge \langle -U \rangle \psi \leftrightarrow \langle -U \rangle (\varphi \wedge \psi)$ is valid.
2. $\langle -U^1 \rangle \varphi \wedge \langle -U^2 \rangle \varphi \rightarrow \langle -U^1 \cup -U^2 \rangle \varphi$ is not valid.
3. $\langle -U^1 \rangle \langle -U^2 \rangle \varphi \rightarrow \langle -U^2 \rangle \langle -U^1 \rangle \varphi$ is not valid.

Property 1 states that negative updates are monotonic, while items 2 and 3 say that in general we cannot take a union of negative updates or change the order of their application. The counterexamples can be provided by exploiting the fact that negative updates may become not executable once united or applied in a different order.

4 Model checking

In this section we consider the global model checking problem for both DDCL$^+$ and DDCL$^-$, and argue that the problem is P-complete in both cases. To achieve the polynomial upper bound, we first prepare the set of subformulas of a given formula similarly to [19]. After that, we label states of a given finite model to calculate the extension of the formula. The labelling is inspired by the model checking algorithm for computation tree logic [10].

4.1 DDCL$^+$

Let a finite model $M = (A, S, Act, act, out, L)$ and a formula $\varphi \in \mathcal{DDCL}^+$ be given. We denote by $sub(\varphi)$ the list of subformulas of φ . Next, we label each subformula in the list $sub(\varphi)$ by the sequence of updates inside the scope of which it appears. Finally, we organise the list in the following way. If ψ^σ and χ^τ are subformulas with labellings σ and τ, then ψ^σ precedes χ^τ if

- both ψ^σ and χ^τ are subformulas of formulas within updates and σ is a proper prefix of τ;
- if ψ^σ is a subformula of a formula within an update and χ^τ is not;
- neither ψ^σ or χ^τ are subformulas of formulas within updates and τ is a proper prefix of σ;
- $\sigma = \tau$ and ψ^σ is a part of χ^τ;
- otherwise, ψ appears to the left of χ in φ.

All ties in the described procedure are broken arbitrarily, and for simplicity we remove all identical elements from the resulting list.

Intuitively, we order list $sub(\varphi)$ in such a way that we first have subformulas of formulas within updates starting with the outermost updates and continuing with inner updates, and then we have subformulas appearing in the scopes of updates ordered from innermost to outermost. Such an organisation of list $sub(\varphi)$ ensures that by the time we need to consider a subformula within the scope of some $[+U]$, we already know what effect $[+U]$ and all preceding updates have on a model.

As an example, consider formula $\varphi := [(q, b, \neg q)^+][(p, a, \neg p)^+]\langle\langle\{a, b\}\rangle\rangle\neg p \wedge \langle\langle\{a, b\}\rangle\rangle p$ with $+U^1 := (q, b, \neg q)^+$ and $+U^2 := (p, a, \neg p)^+$. The ordered labelled list $sub(\varphi)$ looks as follows: q, $\neg q$, $(p)^{+U^1}$, $(\neg p)^{+U^1}$, $(p)^{+U^1,+U^2}$, $(\neg p)^{+U^1,+U^2}$, $(\langle\langle\{a, b\}\rangle\rangle\neg p)^{+U^1,+U^2}$, $([+U^2]\langle\langle\{a, b\}\rangle\rangle\neg p)^{+U^1}$, p, $[+U^1][+U^2]\langle\langle\{a, b\}\rangle\rangle\neg p$, $\langle\langle\{a, b\}\rangle\rangle p$, φ.

In the algorithm below, we will denote by $pref(\sigma)$ the list of all proper prefixes of σ ordered by length starting with longest and including the empty prefix. Moreover, function $last(\sigma)$ returns the last element of σ.

Algorithm 1 An algorithm for global DDCL$^+$ model checking

procedure GLOBALDDCL$^+(M, \varphi)$
  for all $\psi^\sigma \in sub(\varphi)$ **do**
    for all $s \in S$ **do**
      case $\psi^\sigma = (\langle\langle C \rangle\rangle \chi)^\sigma$
        $check \leftarrow false$
        for all $\tau \in pref(\sigma)$ **do**
          if *check* **then**
            break
          else if τ is empty **then**
            for all $\alpha_C \in act(C, s)$ **do**
              $check \leftarrow false$
              for all $\alpha_{A \setminus C} \in act(A \setminus C, s)$ **do**
                if $out(\alpha_C \sqcup \alpha_{A \setminus C}, s)$ is not labelled with χ^σ **then**
                  $check \leftarrow true$
                  break
              if not *check* **then**
                label s with $(\langle\langle C \rangle\rangle \chi)^\sigma$
          else
            for all $i \in A$ **do**
              for all $(\chi_1^\tau, i, \chi_2^\tau)^+ \in last(\tau)$ **do**
                if s is labelled with χ_1^τ **then**
                  if $i \in A \setminus C$ **then**
                    for all $t \in S$ labelled with χ_2^τ **do**
                      if t is **not** labelled with χ^σ **then**
                        $check \leftarrow true$
                        break
                  else
                    for all $t \in S$ labelled with χ_2^τ **do**
                      if t is labelled with χ^σ **then**
                        label s with $(\langle\langle C \rangle\rangle \chi)^\sigma$
                        $check \leftarrow true$
                        break
                  if *check* **then**
                    break
              if *check* **then**
                break
      case $\psi^\sigma = ([+U]\chi)^\sigma$
        if EXECUTABILITY$(+U, \sigma)$ is *true* **then**
          if s is labelled with $\chi^{\sigma,[+U]}$ **then**
            label s with $([+U]\chi)^\sigma$
        else
          label s with $([+U]\chi)^\sigma$
end procedure

```
44: procedure EXECUTABILITY(+U, σ)
45:     for all (χ_1^σ, i, χ_2^σ) ∈ +U do
46:         for all (χ_3^σ, j, χ_4^σ) ∈ +U do
47:             if i ≠ j then
48:                 for all s ∈ S do
49:                     if s is labelled with χ_1^σ and χ_3^σ then
50:                         return false
51:     return true
52: end procedure
```

Algorithm GLOBALDDCL$^+$ labels each state of S with subformulas from $sub(\varphi)$. Labelling proceeds from formulas within updates with a wider scope to formulas within updates with a smaller scope starting with simpler subformulas and proceeding with more complex ones. After all subformulas occurring in updates have been processed, the algorithm labels formulas within the scopes of updates.

Labelling of Boolean cases is straightforward and omitted for brevity. Case for coalitional modalities $(\langle\langle C\rangle\rangle\chi)^\sigma$ is handled in the following way. For a given subformula, we check the closest update within the scope of which the subformula occurs. In other words, we check the last element of σ. Then, if the update has a triple $(\chi_1^\tau, i, \chi_2^\tau) \in last(\tau)$ such that it grants agent i dictatorial powers from some current state, i.e. the current state is labelled with χ_1^τ, we check whether the agent belongs to the anti-coalition $A \setminus C$. If it does, we check whether the agent can force a non-χ-state. This is due to the fact that even though dictatorial powers can be granted to an agent outside of C, the agent may still lack an action to force a non-χ-state. If $i \in C$, then we check whether one of states t labelled with χ_2^τ is also labelled with χ^σ. If yes, then we label s with $(\langle\langle C\rangle\rangle\chi)^\sigma$. Informally, this means that in order to ensure that $(\langle\langle C\rangle\rangle\chi)^\sigma$ holds in the current state, it is enough that some local dictator from C can force at least one χ^σ-state.

If the loop does not terminate with τ, we check the next update, with a wider scope, within which formula $\langle\langle C\rangle\rangle\chi$ occurs. Finally, if none of the updates in σ settle $(\langle\langle C\rangle\rangle\chi)^\sigma$, i.e. none of the updates influence the abilities of coalition C, we check whether C can force χ in a usual way. The reason we check updates starting from the innermost is that later updates may override earlier dictatorial powers.

Finally, the case of $([+U]\chi)^\sigma$ handled according to the semantics with an additional subroutine checking the executability of $+U$. The correctness of the algorithm can be shown by an induction on the formula structure.

Theorem 2. Given a CGM M and a formula φ, $M_s \models \varphi$ if and only if GLOBALDDCL$^+$ labels s with φ.

Preparation of $sub(\varphi)$ can be done in $\mathcal{O}(|\varphi| \times |\varphi|)$ steps. Procedure GLOBALDDCL$^+$ is bounded by $\mathcal{O}(|\varphi|^3 \times |A| \times |S|^2)$ (the case of coalitional modality), and procedure EXECUTABILITY is bounded by $\mathcal{O}(|\varphi|^2 \times |S|)$. The lower bound follows from P-completeness of CL model checking .

Theorem 3. Complexity of DDCL$^+$ model checking problem is P-complete.

4.2 DDCL$^-$

Similarly to the case of DDCL$^+$, DDCL$^-$ global model checking requires preparation of the list of subformulas $sub(\varphi)$ of $\varphi \in \mathcal{DDCL}^-$. The labelling of subformulas follows the same procedure as in Section 4.1 with the only difference that now we also include update symbols $-U$ in the list. Moreover, labelled updates $(-U)^\sigma$ will follow immediately after subformulas of formulas within the update $-U$. We will use updates to label action profiles in the algorithm below.

As an example, we consider a formula which is almost identical to φ from Section 4.1. Let the new φ be $[(q,b,\neg q)^-][(p,a,\neg p)^-]\langle\langle\{a,b\}\rangle\rangle\neg p \wedge \langle\langle\{a,b\}\rangle\rangle p$ with $-U^1 := (q,b,\neg q)^-$ and $-U^2 := (p,a,\neg p)^-$. The ordered labelled list $sub(\varphi)$ looks as follows: q, $\neg q$, $-U^1$, $(p)^{-U^1}$, $(\neg p)^{-U^1}$, $(-U^2)^{-U^1}$,$(p)^{-U^1,-U^2}$, $(\neg p)^{-U^1,-U^2}$, $(\langle\langle\{a,b\}\rangle\rangle\neg p)^{-U^1,-U^2}$, $([-U^2]\langle\langle\{a,b\}\rangle\rangle\neg p)^{-U^1}$, p, $[-U^1][-U^2]\langle\langle\{a,b\}\rangle\rangle\neg p$, $\langle\langle\{a,b\}\rangle\rangle p$, φ.

Algorithm 2 An algorithm for global DDCL$^-$ model checking

```
1: procedure GlobalDDCL⁻(M, φ)
2:     prefix ← null
3:     for all ψ^σ ∈ sub(φ) do
4:         if prefix = null or not prefix = σ then
5:             for all i ∈ A do
6:                 for all s ∈ S do
7:                     FS(i, s, σ) = ForcingSet(i, s, σ)
8:                     prefix ← σ
9:         for all s ∈ S do
10:            case ψ^σ = −U^σ
11:                for all α ∈ act(s) do:
12:                    check ← true
13:                    for all α_i ∈ α do:
14:                        if α_i ∈ FS(i, s, σ) then
15:                            check ← false
16:                            for all (χ_1^σ, i, χ_2^σ)^- ∈ −U^σ do
17:                                if s is labelled with χ_1^σ and out(s, α) is labelled with χ_2^σ and α is
   labelled with σ then
18:                                    check ← true
19:                                    break
20:                        if not check then
21:                            break
22:                    if check then
23:                        label α with σ, −U
24:            case ψ^σ = (⟨⟨C⟩⟩χ)^σ
25:                for all α_C ∈ act(C, s) do
26:                    check ← false
27:                    for all α_{A\C} ∈ act(A \ C, s) do
28:                        if α_C ⊔ α_{A\C} is labelled with σ then
```

```
                        if out(α_C ⊔ α_{A\C}, s) is not labelled with χ^σ then
                            check ← true
                            break
                    if not check then
                        label s with (⟪C⟫χ)^σ
            case ψ^σ = ([−U]χ)^σ
                if Executability(−U, σ) is true then
                    if s is labelled with χ^{σ,[−U]} then
                        label s with ([−U]χ)^σ
                else
                    label s with ([−U]χ)^σ
end procedure
procedure ForcingSet(i, s, σ)
    ρ ← ∅
    for all α_i ∈ act(i, s) do
        check ← false
        for all α_{A\i} ∈ act(A \ i, s) do
            if α_i ⊔ α_{A\i} is labelled with σ then
                for all β_{A\i} ∈ act(A \ i, s) do
                    if not out(s, α_i ⊔ α_{A\i}) = out(s, α_i ⊔ β_{A\i}) then
                        check ← true
                        break
                if check then
                    break
        if not check then
            ρ ← ρ ∪ {α_i}
    return ρ
end procedure
procedure Executability(−U, σ)
    for all s ∈ S do
        check ← false
        for all α ∈ act(s) do
            if α is labelled with σ, −U then
                check ← true
                break
        if not check then
            return false
    return true
end procedure
```

In the global model checking algorithm for DDCL$^-$ we omit Boolean cases since they are as expected. Due to the fact that the set of forcing actions of an agent may change after an update, we use procedure ForcingSet to calculate forcing sets for all agents and labels σ. This allows us to know exactly what actions are forcing for a given agent in a

given state after some given series of updates. To preclude ForcingSet from running several times on the same input, we store sets of forcing actions in memory. The required space is linear and bounded by $\mathcal{O}(|S| \times |Act| \times |\varphi|)$. Time required for computing all necessary forcing sets is bounded by $\mathcal{O}(|\varphi| \times |A|^3 \times |S| \times |Act|^3)$.

Cases of $[-U]^\sigma$ label action profiles that should be preserved after the sequence of updates $\sigma, -U$. For each action profile we first check whether all forcing actions α_i in the profile satisfy some triple $(\chi_1, i, \chi_2)^-$ in the update. If it is the case, we label the action profile with $\sigma, -U$. Following the semantics of negative updates, we also label the action profile with $\sigma, -U$, if it does not contain any forcing actions, and thus is not affected by the update.

The case of coalitional modality is handled in a usual way with an additional check that a current action profile is labelled with σ. Such a check ensures that the corresponding transition has been preserved after the sequence of updates σ. Finally, the case of subformulas of type $([-U]\chi)^\sigma$ follows the semantics. The executability of $-U$ is checked in procedure Executability$(-U, \sigma)$ that returns *true* if for each state there is at least one action profile labelled with $\sigma, -U$ meaning that the profile has been preserved after updates $\sigma, -U$. Observe that since subformulas $-U^\sigma$ come before $([-U]\chi)^\sigma$, by this time all qualified action profiles have been labelled with $\sigma, -U$.

The algorithm follows closely the semantics of DDCL$^-$, and its correctness can be shown by the induction on the formula structure.

Theorem 4. Given a CGM M and a formula φ, $M_s \models \varphi$ if and only if GlobalDDCL$^-$ labels s with φ.

Preparation of $sub(\varphi)$ can be done in $\mathcal{O}(|\varphi| \times |\varphi|)$ steps. Procedure GlobalDDCL$^-$ is bounded by $\mathcal{O}(|\varphi|^2 \times |M|^3)$ for the case $-U$, and the lower bound follows from P-completeness of CL model checking.

Theorem 5. Complexity of DDCL$^-$ model checking problem is P-complete.

5 Expressivity

In this section we explore the expressivity of dictatorial dynamic coalition logic. First, we show that both the positive and the negative versions are strictly more expressive than CL, and then we demonstrate that the two dynamic extensions of CL are incomparable relative to each other. After that, we situate DDCLs in the wider context of logics for strategic reasoning, and show that DDCLs stand quite apart from them by proving the corresponding incomparability results.

5.1 How DDCLs are related to CL and to each other

Definition 5.1. Let φ and ψ be formulas. We say that they are *equivalent* if for all M_s it holds that $M_s \models \varphi$ iff $M_s \models \psi$.

Definition 5.2. 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.

We first show that both logics, DDCL$^+$ and DDCL$^-$, are strictly more expressive than $\mathcal{CL}$, and thus, in contrast to the situation with AUL [18], positive and negative updates cannot be eliminated.

Proposition 4. $\mathcal{CL} < \mathcal{DDCL}^+$ and $\mathcal{CL} < \mathcal{DDCL}^-$.

Proof. The fact that both $\mathcal{DDCL}^+$ and $\mathcal{DDCL}^-$ are at least as expressive as $\mathcal{CL}$ follows from the fact that $\mathcal{CL} \subseteq \mathcal{DDCL}^+$ and $\mathcal{CL} \subseteq \mathcal{DDCL}^-$.

For $\mathcal{DDCL}^+ \not\leqslant \mathcal{CL}$, consider models M_s and N_s in Figure 4. There is only one agent a with the only available action a_0. It is immediate that M_s and N_s are bisimilar and thus cannot be distinguished by any formula of $\mathcal{CL}$. At the same time, $\langle (p, a, \neg p)^+ \rangle \langle\!\langle a \rangle\!\rangle \neg p$ is true in N_s (with the resulting updated model N_s^{+U}) and false in M_s (as there are no states where $\neg p$ would hold).

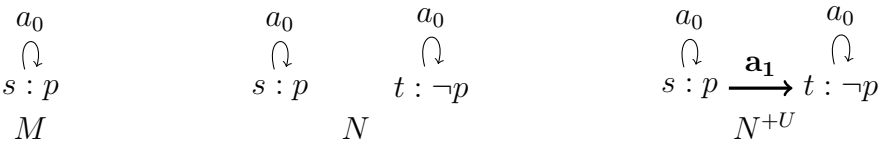

Figure 4: Models, from left to right, M_s, N_s, and N_s^{+U}. New actions are in bold font.

In order to show that $\mathcal{DDCL}^- \not\leqslant \mathcal{CL}$, we will use a technical trick that some $-U$'s are not executable in some models. Consider $\langle (p, a, p)^- \rangle \langle\!\langle a \rangle\!\rangle p \in \mathcal{DDCL}^-$, and assume towards a contradiction that there is an equivalent $\psi \in \mathcal{CL}$ with $|\psi| = n$.

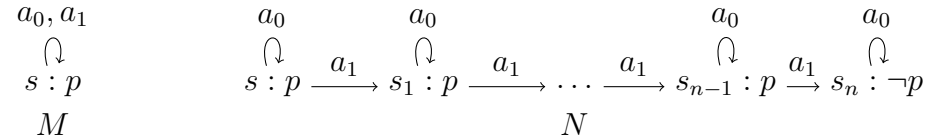

Figure 5: Models M_s (left) and N_s (right).

Consider models M_s and N_s in Figure 5. The former has only one state with a loop, and the latter is a chain of length $n+1$ with $\neg p$ being the case only at the farthest state from s. Although M and N are not bisimilar, it is clear from the construction of the models that both of them satisfy the same ψ up to modal depth n: there is simply not enough modal depth to witness a difference. On the other hand, $M_s \models \langle (p, a, p)^- \rangle \langle\!\langle a \rangle\!\rangle p$ (all the forcing actions remain intact), and $N_s \not\models \langle (p, a, p)^- \rangle \langle\!\langle a \rangle\!\rangle p$. Indeed, since the update requires us to remove all forcing actions that do not conform to $(p, a, p)^-$, we have to remove the loop at

the last state s_n, which results in s_n being without any actions, and thus the whole update is not executable in N_s. □

Another non-obvious question is whether DDCL$^+$ and DDCL$^-$ are different. We show that it is indeed the case, and, in particular, that the logics are incomparable.

Theorem 6. $\mathcal{DDCL}^- \not\leqslant \mathcal{DDCL}^+$.

Proof. Consider models M_s and N_s in Figure 6. Observe that they are bisimilar, and thus satisfy the same formulas of $\mathcal{CL}$. Moreover, it can be argued that the models also satisfy the same formulas of $\mathcal{DDCL}^+$. Indeed, we can reason by induction that adding a forcing transition in one model, adds an equivalent forcing transition in the other model. Intuitively, some $+U$ is executable in one model if and only if it is executable in the other model, and no new forcing arrow can take us to a non-bisimilar state.

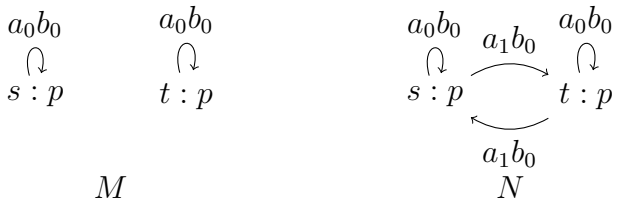

Figure 6: Models M_s (left) and N_s (right).

Updates $-U$ depend, on the other hand, on the sets of forcing actions. For M, the sets of forcing actions are $\mathfrak{f}(a,s) = \{a_0^{(s,s)}\}$, $\mathfrak{f}(b,s) = \{b_0^{(s,s)}\}$, $\mathfrak{f}(a,t) = \{a_0^{(t,t)}\}$, and $\mathfrak{f}(b,t) = \{b_0^{(t,t)}\}$. The thing to notice here is that both agents have forcing actions in both states. For model N, the non-empty sets of forcing actions are $\mathfrak{f}(a,s) = \{a_0^{(s,s)}, a_1^{(s,t)}\}$ and $\mathfrak{f}(a,t) = \{a_0^{(t,t)}, a_1^{(t,s)}\}$. Since there are no forcing actions for agent b, we can exploit it with $-U$'s. In particular, consider $\langle (p,a,p)^- \rangle p$, which intuitively orders to preserve only a's forcing actions. It is easy to see that $\langle (p,a,p)^- \rangle$ is not executable in M_s, and hence $M_s \not\models \langle (p,a,p)^- \rangle p$. At the same time $N_s \models \langle (p,a,p)^- \rangle p$, since updating N_t with $\langle (p,a,p)^- \rangle$ leaves the model intact. □

Theorem 7. $\mathcal{DDCL}^+ \not\leqslant \mathcal{DDCL}^-$.

Proof. Consider models M_s and N_s in Figure 7. Observe that model N is actually a disjoint union of two models. Moreover, M_s and N_s are bisimilar.

The models are constructed in such a manner that the sets of forcing actions in all states for all agents of both models are empty. Hence, given an arbitrary formula φ of $\mathcal{DDCL}^-$ we can use Proposition 2 to get a translation $t(\varphi)$ into an equivalent formula $\mathcal{CL}$. Finally, since M_s and N_s agree on formulas of $\mathcal{CL}$ we can conclude that they also agree on all formulas of $\mathcal{DDCL}^-$.

Now consider $\langle (p,a,\neg p)^+ \rangle \langle\langle a \rangle\rangle \neg p \in \mathcal{DDCL}^+$. Since there are no states that satisfy $\neg p$ in M, updating the model with $\langle (p,a,\neg p)^+ \rangle$, which is executable in M, yields exactly the same model. Because there are no $\neg p$-states, we have $M_s \not\models \langle (p,a,\neg p)^+ \rangle \langle\langle a \rangle\rangle \neg p$.

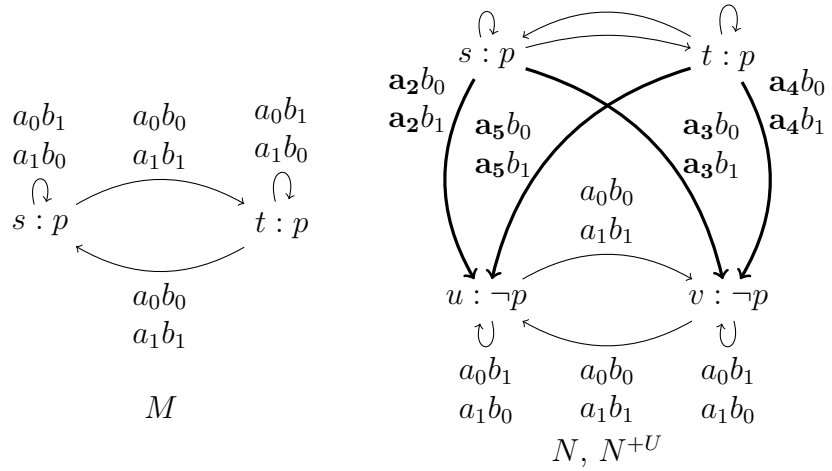

Figure 7: Models M_s (on the left), N_s (on the right minus thick transitions), and N_s^{+U} (including thick transitions). Transitions between states s and t in models N and N^{+U} are exactly like in M, and thus some labels are omitted for readability.

On the other hand, there are states satisfying $\neg p$ in N, and the update of N with $\langle (p, a, \neg p)^+ \rangle$ is shown in Figure 7 on the right including thick transitions. It is clear that $N_s^{+U} \models \langle\!\langle a \rangle\!\rangle \neg p$, and hence $N_s \models \langle (p, a, \neg p)^+ \rangle \langle\!\langle a \rangle\!\rangle \neg p$. □

5.2 How DDCLs are related to other logics for reasoning about strategic abilities

Coalition logic is not the only formalism for reasoning about strategic abilities on concurrent game models. Other notable examples of such logics include ATL, ATL* [4], and strategy logic [21]. Before presenting the expressivity results, we provide a very concise overview of the logics; the reader can find further details in the cited literature.

ATL. The language of alternating-time temporal logic [4] extends the language of CL with temporal operators $\mathsf{X}\varphi$ for 'φ is true in the next moment', $\mathsf{G}\varphi$ for 'φ is alway true', and $\psi \mathsf{U} \varphi$ for 'ψ is true until φ'.

Definition 5.3. The $\mathcal{ATL}$ is defined recursively as follows:

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

Observe that in $\mathcal{ATL}$, temporal operators are in the immediate scope of the coalition modality.

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

Definition 5.4. 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 5.5. Given a CGM M, a *play* λ 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 5.6. Let M_s be a CGM. The semantics of ATL (omitting Boolean cases) is defined as follows:

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

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.

ATL*. In ATL, temporal operators appear immediately within the scope of a coalitional modality. Relaxing this condition results in a generalisation of ATL that is called ATL*.

Definition 5.7. The language of $\mathcal{ATL}^*$ is defined by the following BNF:

$$\begin{array}{lll} \text{State formulas} & \varphi & ::= \; p \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \langle\langle C\rangle\rangle \varphi \\ \text{Path formulas} & \psi & ::= \; \varphi \mid \neg\psi \mid (\psi \wedge \psi) \mid \mathsf{X}\psi \mid \mathsf{G}\psi \mid \psi \mathsf{U} \psi \end{array}$$

ATL*, in contrast to ATL, allows agents to have strategies that take into account the entire history of a play.

Definition 5.8. Let λ be a play. We call a finite segment $\lambda[0, i]$ a *history* h. Given a model M with the set of states S, we denote by S^+ the set of all histories. A *perfect recall strategy* for agent i in model M is a function $str_i^+ : S^+ \to Act$ such that $str_i(h) \in act(i, s)$ and s is the last element of h. A perfect recall strategy for coalition C, denoted str_C is a tuple of perfect recall strategies for each $i \in C$.

Finally, 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

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

The semantics of ATL* are defined by the mutual induction on state and path formulas, and in the definition below we omit Boolean cases.

Definition 5.9. Let M_s be a CGM. The semantics of ATL* is as follows:

$$\begin{array}{lll} M_s \models \langle\langle C\rangle\rangle \varphi & \text{iff} & \exists str_C^+ : M_\lambda \models \varphi \text{ for all } \lambda \in Plays^+(s, str_C^+) \\ M_\lambda \models \varphi & \text{iff} & M_{\lambda[0]} \models \varphi, \text{ for all state formulas } \varphi \\ M_\lambda \models \mathsf{X}\psi & \text{iff} & M_{\lambda[1,\infty)} \models \psi \\ M_\lambda \models \mathsf{G}\psi & \text{iff} & M_{\lambda[i,\infty)} \models \psi \text{ for all } i \geq 0 \\ M_\lambda \models \psi \mathsf{U} \varphi & \text{iff} & \exists i \geq 0 : M_{\lambda[i,\infty)} \models \varphi, \text{ and } M_{\lambda[j,\infty)} \models \psi \text{ for all } 0 \leq j < i \end{array}$$

Strategy logic. In all logics considered so far in the paper, strategies of agents are implicit. Although the semantics of strategy logic (SL) [21] is defined on CGMs, the syntax of the logic has explicit strategy quantifiers: $\langle\langle x \rangle\rangle$ meaning 'there is a strategy x' and $[\![x]\!]$ meaning 'for all strategies x'. Moreover, to associate strategy variables with particular agents, there is a binding operator (a, x) that means 'bind agent a to the strategy associated with x'.

Definition 5.10. The language of $\mathcal{SL}$ is defined recursively by the following grammar:

$$\varphi \quad ::= \quad p \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \mathsf{X}\varphi \mid \varphi \mathsf{U} \varphi \mid \langle\langle x \rangle\rangle \varphi \mid [\![x]\!]\varphi \mid (a, x)\varphi$$

Having variables in a language leads to the necessity of plethora of additional definitions regarding variable assignments and variables being free. We omit these details, and instead provide the semantics of SL only on intuitive level. Sacrificing preciseness for brevity will not, however, backfire, since the result of this section do not require the full machinery of SL. The interested reader is referred to [21, 20] for more details.

Let us consider the truth condition of the existential strategy quantifier. Having a model M_s and assignment μ of variables and agents to perfect recall strategies, $M_s^\mu \models \langle\langle x \rangle\rangle \varphi$ holds if and only if there is a strategy str^+ from the set of all strategies such that $M_s^{\mu[x \mapsto str^+]} \models \varphi$, where $\mu[x \mapsto str^+]$ means that variable x has been assigned strategy str^+. Similarly, $M_s^\mu \models [\![x]\!]\varphi$ if and only if for all strategies str^+ it holds that $M_s^{\mu[x \mapsto str^+]} \models \varphi$. Finally, binding $(a, x)\varphi$ assigns to agent a the strategy assigned to variable x. In symbols, $M_s^\mu \models (a, x)\varphi$ if and only if $M_s^{\mu[a \mapsto \mu(x)]} \models \varphi$.

Having direct access to strategies in the language leads to high expressivity. Indeed, in SL one can express that two agents share the same strategy, or that one agent changes their strategy during a play. In particular, any formula with a coalition modality can be expressed in SL. For example, in the case of $A = \{a, b, c, d\}$ and formula $\langle\langle \{a, b\} \rangle\rangle \varphi$, an equivalent SL formula would be $\langle\langle x_a \rangle\rangle \langle\langle x_b \rangle\rangle [\![x_c]\!][\![x_d]\!](a, x_a)(b, x_b)(c, x_c)(d, x_d)\varphi$.

Expressivity. Not only are ATL, ATL*, and SL interesting logics on CGMs, each next one of them is strictly more expressive than the previous one. In particular, $\mathcal{CL} < \mathcal{ATL} < \mathcal{ATL}^* < \mathcal{SL}$. Below we show that even though DDCLs are strictly more expressive than CL, they stand quite apart from their strategic cousins. Formally, we claim that both DDCLs are incomparable with any of ATL, ATL*, and SL.

Theorem 8. $\mathcal{DDCL}^+ \not\leqslant \mathcal{ATL}$, $\mathcal{DDCL}^+ \not\leqslant \mathcal{ATL}^*$, $\mathcal{DDCL}^+ \not\leqslant \mathcal{SL}$, $\mathcal{DDCL}^- \not\leqslant \mathcal{ATL}$, $\mathcal{DDCL}^- \not\leqslant \mathcal{ATL}^*$, and $\mathcal{DDCL}^- \not\leqslant \mathcal{SL}$

Proof. For the case of $\mathcal{DDCL}^+$, recall models M_s and N_s in Figure 4. As claimed in the proof of Proposition 4, the models are distinguished by a $\mathcal{DDCL}^+$ formula. However, without invoking the notion of bisimilarity, it is clear that none of the strategic logics can 'jump' from s to t in model N, and thus spot the difference between M_s and N_s.

To get the results for $\mathcal{DDCL}^-$, we recall models M_s and N_s in Figure 6. That the models can be distinguished by a $\mathcal{DDCL}^-$ formula is shown in the proof of Proposition 6.

Again, it is clear that none of the strategic logics can tell M_s from N_s due to the fact that p is true everywhere, and agents can only force a transition to some p-state. □

To prove the other direction, it is enough to show that there is an ATL formula that can distinguish two models that are not distinguishable by neither of DDCLs. Results for ATL* and SL will follow trivially as they are strictly more expressive than ATL.

Theorem 9. $\mathcal{ATL} \not\leqslant \mathcal{DDCL}^+$.

Proof. Consider an ATL formula $\langle\langle\{a\}\rangle\rangle p \mathsf{U} q$, and assume that there is an equivalent DDCL$^+$ formula φ of size $|\varphi|$. Now consider model M in Figure 8. The model is an almost symmetric chain of length $2 \cdot |\varphi| + 4$. In M, state s_{n+1} satisfies neither p nor q, and state s_{n+2} satisfies only q.

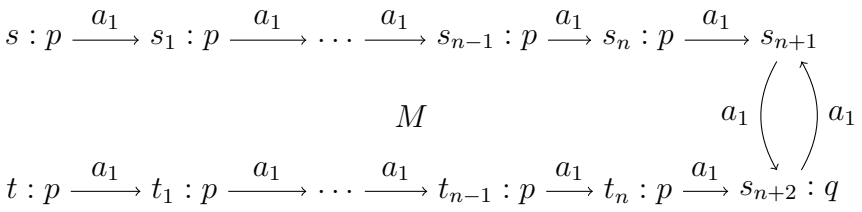

Figure 8: Models M. If a state does not satisfy a propositional variable, we do not list that variable next to the state.

Observe that in M_t agent a can maintain p until the agent finally reaches the state satisfying q. On the other hand, the same is impossible if the agent starts in M_s due to the fact that in order to reach the state satisfying q, the agent has to pass through s_{n+1}, where both p and q are false, and there is no way to avoid s_{n+1}. Thus, $M_s \not\models \langle\langle\{a\}\rangle\rangle p \mathsf{U} q$ and $M_t \models \langle\langle\{a\}\rangle\rangle p \mathsf{U} q$.

To argue that φ cannot distinguish the pointed models, we first note that states s_{n+1} and s_{n+2} lie more than $|\varphi|$ steps away from both s and t. Thus there is not enough modal depth to witness the difference using just coalitional modalities.

Second, notice that if a positive update allows us to reach a state satisfying some ψ from the s-side of the model, then the same update will allow us to reach the same state from the t-side of the model, and vice versa. This is guaranteed by the fact that models M_s and M_t are n-bisimilar. As an example, consider state t_n. This is the only state satisfying $p \wedge \langle\langle\{a\}\rangle\rangle q$, and thus it could be used to distinguish s- and t-sides of the model. However, notice that t_n cannot be reached via coalitional modalities as M_s and N_t are n-bisimilar. State t_n can be reached, though, if we use some positive update to create a transition from some current state to t_n. In this case, there will also be a transition from a 'mirror state' on the other side of the model, since after i steps in evaluating φ we are still in $n - i$-bisimilar states. Therefore, $M_s \models \varphi$ if and only if $M_t \models \varphi$. □

Theorem 10. $\mathcal{ATL} \not\leqslant \mathcal{DDCL}^-$.

Proof. Consider an ATL formula $\langle\langle\{a,b\}\rangle\rangle p \mathsf{U} q$, and assume that there is an equivalent DDCL$^-$ formula φ of size $|\varphi|$. Now consider model M_s in Figure 7 and model N_s in Figure 9. Notice that model N is a chain of length $|\varphi|+1$ with the rightmost state being the only one where p does not hold.

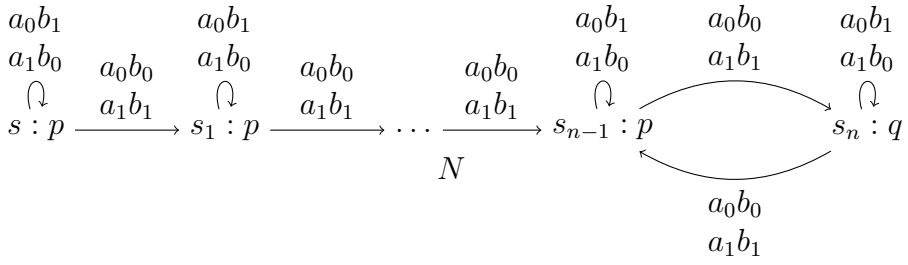

Figure 9: Model N_s.

It is immediate that $M_s \not\models \langle\langle\{a,b\}\rangle\rangle p \mathsf{U} q$, since there is no q-state in the model, and $N_s \models \langle\langle\{a,b\}\rangle\rangle p \mathsf{U} q$. As for φ, it is enough to notice that the models are constructed in such a way that there are no forcing actions for any agent in any state. Thus, by Proposition 2, negative updates do not affect the models. Moreover, since the length of N is $|\varphi|+1$ there is not enough modal depth in $|\varphi|$ to witness a difference between the models. Therefore, $M_s \models \varphi$ and $N_s \models \varphi$. □

The overall expressivity landscape of DDCLs and other logics for reasoning about strategic abilities is depicted in Figure 10.

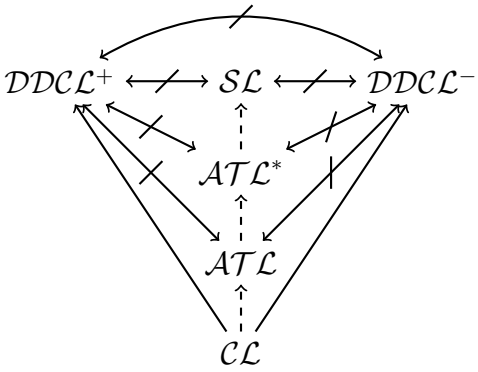

Figure 10: The relative expressivity of the logics. An arrow from $\mathcal{L}_1$ to $\mathcal{L}_2$ means that $\mathcal{L}_1 \leqslant \mathcal{L}_2$. If there is no symmetric arrow, then $\mathcal{L}_1 < \mathcal{L}_2$. The arrow relation is transitive. An arrow from $\mathcal{L}_1$ to $\mathcal{L}_2$ is crossed-out if $\mathcal{L}_1 \not\leqslant \mathcal{L}_2$. Dashed arrows represent previously known results.

6 Conclusion

We made a first step in exploring the *dynamics of ability* by presenting two dynamic extensions of CL. The first extension, DDCL$^+$, deals with granting dictatorial powers to single agents. The second extension, DDCL$^-$, reasons about revoking dictatorial powers. We showed that both formalisms are strictly more expressive than CL, mutually incomparable, and have P-complete model checking problems. We also considered both DDCLs in the context of other logics of strategic ability, namely ATL, ATL*, and SL.

Since this work is an initial exposition of dynamic coalition logic, there is a plethora of open questions and further research directions. For example, it is not clear how to combine granting and revoking dictatorial powers together in the same update. Apart from that, the next natural step is reasoning about granting powers to coalitions, rather than to single agents, i.e. we will consider $(\chi, C, \psi)^+$ and $(\chi, C, \psi)^-$ in the future. The challenge here is that while we may want to grant a coalition some forcing power, we may also want that none of the members of the coalition has such a power on their own.

Another exciting avenue of further research is is providing sound and complete axiomatisations of the logics. However, it seems particularly difficult as axiomatisations of many well-known relation changing logics are still unknown [5, 6]. Finally, there is also a conceptual subtlety worth exploring. In our definition of forcing actions we called an action forcing if for all action profiles it appears in, the outcome state is the same. In other words, forcing actions in our interpretation force *single states*. This is a reasonable interpretation of forcing/dictatorship, but it is not the only one. Another possible interpretation of a forcing action is that the action forces a (not necessarily singleton) set of φ-states.

On a more global scale, we would like to 'dynamify' CL in the vein of action models of DEL [7], thus coming up with a general dynamic coalition logic. The same goal can be set out for finding dynamic extensions of ATL and SL.

References

[1] Thomas Ågotnes and Natasha Alechina. Coalition logic with individual, distributed and common knowledge. *Journal of Logic and Computation*, 29(7):1041–1069, 2019.

[2] Thomas Ågotnes, Valentin Goranko, and Wojciech Jamroga. Alternating-time temporal logics with irrevocable strategies. In Dov Samet, editor, *Proceedings of the 11th TARK*, pages 15–24, 2007.

[3] Thomas Ågotnes, Valentin Goranko, Wojciech Jamroga, and Michael Wooldridge. Knowledge and ability. In Hans van Ditmarsch, Joseph Y. Halpern, Wiebe van der Hoek, and Barteld Kooi, editors, *Handbook of Epistemic Logic*, pages 543–589. College Publications, 2015.

[4] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49:672–713, 2002.

[5] Carlos Areces, Raul Fervari, and Guillaume Hoffmann. Relation-changing modal operators. *Logic Journal of the IGPL*, 23(4):601–627, 2015.

[6] Guillaume Aucher, Johan van Benthem, and Davide Grossi. Modal logics of sabotage revisited. *Journal of Logic and Computation*, 28(2):269–303, 2018.

[7] Alexandru Baltag, Lawrence S. Moss, and Sławomir Solecki. The logic of public announcements, common knowledge, and private suspicions. In Itzhak Gilboa, editor, *Proceedings of the 7th TARK*, pages 43–56. Morgan Kaufmann, 1998.

[8] Johan van Benthem. *Logical Dynamics of Information and Interaction*. CUP, 2011.

[9] Thomas Bolander and Mikkel Birkegaard Andersen. Epistemic planning for single and multi-agent systems. *Journal of Applied Non-Classical Logics*, 21(1):9–34, 2011.

[10] Edmund M. Clarke and E. Allen Emerson. Design and synthesis of synchronization skeletons using branching-time temporal logic. In Dexter Kozen, editor, *Logics of Programs*, volume 131 of *LNCS*, pages 52–71. Springer, 1981.

[11] Hans van Ditmarsch, Wiebe van der Hoek, and Barteld Kooi. *Dynamic Epistemic Logic*, volume 337 of *Synthese Library*. Springer, 2008.

[12] Rustam Galimullin. *Coalition announcements*. PhD thesis, University of Nottingham, UK, 2019.

[13] Rustam Galimullin. Coalition and relativised group announcement logic. *Journal of Logic, Language and Information*, 30(3):451–489, 2021.

[14] Rustam Galimullin and Thomas Ågotnes. Dynamic coalition logic: Granting and revoking dictatorial powers. In Sujata Ghosh and Thomas Icard, editors, *Proceedings of the 8th LORI*, volume 13039 of *LNCS*, pages 88–101. Springer, 2021.

[15] Valentin Goranko and Wojciech Jamroga. Comparing semantics of logics for multi-agent systems. *Synthese*, 139(2):241–280, 2004.

[16] Maurice Herlihy and Mark Moir. Blockchains and the logic of accountability: Keynote address. In Martin Grohe, Eric Koskinen, and Natarajan Shankar, editors, *Proceedings of the 31st LICS*, pages 27–30. ACM, 2016.

[17] Wiebe van der Hoek and Michael J. Wooldridge. Cooperation, knowledge, and time: Alternating-time temporal epistemic logic and its applications. *Studia Logica*, 75(1):125–157, 2003.

[18] Barteld Kooi and Bryan Renne. Arrow update logic. *Review of Symbolic Logic*, 4(4):536–559, 2011.

[19] Louwe B. Kuijer. An arrow-based dynamic logic of norms. In Julian Gutierrez, Fabio Mogavero, Aniello Murano, and Michael Wooldridge, editors, *Proceedings of the 3rd SR*, pages 1–11, 2015.

[20] Fabio Mogavero, Aniello Murano, Giuseppe Perelli, and Moshe Y. Vardi. Reasoning about strategies: On the model-checking problem. *ACM Transactions on Computational Logic*, 15(4):34:1–34:47, 2014.

[21] Fabio Mogavero, Aniello Murano, and Moshe Y. Vardi. Reasoning about strategies. In Kamal Lodaya and Meena Mahajan, editors, *Proceedings of the 30th FSTTCS*, volume 8 of *LIPIcs*, pages 133–144. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2010.

[22] Marc Pauly. *Logic for Social Software*. PhD thesis, ILLC, University of Amsterdam, The Netherlands, 2001.

[23] Marc Pauly. A modal logic for coalitional power in games. *Journal of Logic and Computation*, 12(1):149–166, 2002.