Prompt Alternating-Time Epistemic Logics

Benjamin Aminof
Technische Universiät Wien

Aniello Murano and **Sasha Rubin**
Università di Napoli "Federico II"

Florian Zuleger
Technische Universiät Wien

Abstract

In temporal logics, the operator F expresses that at some time in the future something happens, e.g., a request is eventually granted. Unfortunately, there is no bound on the time until the eventuality is satisfied which in many cases does not correspond to the intuitive meaning system designers have, namely, that F abstracts the idea that there is a bound on this time although its magnitude is not known. An elegant way to capture this meaning is through Prompt-LTL, which extends LTL with the operator $\mathsf{F_P}$ ("prompt eventually"). We extend this work by studying alternating-time epistemic temporal logics extended with $\mathsf{F_P}$.

We study the model-checking problem of the logic Prompt-KATL*, which is ATL* extended with epistemic operators and prompt eventually. We also obtain results for the model-checking problem of some of its fragments. Namely, of Prompt-KATL (ATL with epistemic operators and prompt eventually), Prompt-KCTL* (CTL* with epistemic operators and prompt eventually), and finally the existential fragments of Prompt-KATL* and Prompt-KATL.

Introduction

Alternating-time temporal logics are expressive tools for reasoning about multi-agent systems (Alur, Henzinger, and Kupferman 2002; van der Hoek and Wooldridge 2002; Chatterjee, Henzinger, and Piterman 2010; Mogavero et al. 2014). These powerful logics allow one to express individual or common goals of the agents throughout time, as well as specify the interactions among the agents (cooperation or adversarial). *Model checking* (Clarke and Emerson 1981; Queille and Sifakis 1981) specifications written in these logics allows one to verify the correct behavior of multi agent systems using recently developed practical automatic tools (Lomuscio, Qu, and Raimondi 2009; Čermák et al. 2014; Čermák, Lomuscio, and Murano 2015).

A pioneering logic in this field is *Alternating-Time Temporal Logic (*ATL**)* and its fragments ATL (Alur, Henzinger, and Kupferman 2002), and CTL* (Emerson and Halpern 1986). ATL* formulas are usually interpreted over *concurrent game structures (CGS)*, which are labeled-state transition-systems with the ability of modeling the interaction among agents. For example, in a system with multiple agents and shared resources, the fact that a set of agents A can ensure that, regardless of the actions of the other agents, every request to access a resource is eventually granted, can be expressed by the ATL* formula $\langle\langle A\rangle\rangle\, \mathsf{G}(req \rightarrow \mathsf{F}\, grant)$.

A crucial shortcoming in real-life temporal-logic verification, deeply ingrained in the definition of linear-temporal logic (LTL), is that the satisfaction of a formula like $\mathsf{F}\, grant$ implies no *a priori bound* on *when* the $grant$ occurs. I.e., the system may admit executions with longer and longer delays before the $grant$, and yet still satisfy the formula $\mathsf{F}\, grant$. Replacing the above formula with a formula specifying that the grant should occur within some fixed number of steps (say 3) is usually not an option, since one usually does not know the maximal delay that should be expected. This fact, that the F operator of temporal-logics fails to capture the intuitive meaning of "within some bounded amount of time" has motivated the introduction of an extension of LTL, called "prompt-LTL", in (Kupferman, Piterman, and Vardi 2009; Alur et al. 2001) that includes a new operator $\mathsf{F_P}$ called "prompt eventually". The semantics of $\mathsf{F_P}\,\phi$ is such that it is satisfied only if there is some bound k, which is shared by all behaviours/computations of the system, such that whenever $\mathsf{F_P}\,\psi$ should hold at some point along a computation then ψ holds within at most k steps. (Kupferman, Piterman, and Vardi 2009) goes on to show that prompt-LTL model checking is not more costly and slightly more complicated than LTL-model checking. It is important to note that prompt-LTL formulas are in positive normal form (i.e., with negations pushed all the way to the atoms), but it does not include the dual operator $\mathsf{G_P}$ of the $\mathsf{F_P}$ operator (however, the operator G is included). As argued in (Kupferman, Piterman, and Vardi 2009), on the one hand $\mathsf{G_P}$ is less useful than $\mathsf{F_P}$ (its meaning is that there is some global bound k such that whenever $\mathsf{G_P}\,\psi$ should hold then ψ holds for at least k steps, and we do not care afterwards), and on the other hand adding it to the logic seriously complicates the decision procedures.

Since ATL* inherits its temporal operators from LTL it is natural to consider extending it with the $\mathsf{F_P}$ operator, thus allowing one to specify that eventualities should not be delayed for an unbounded number of steps[1]. We believe that, in a multi-agent setting, the need for the $\mathsf{F_P}$ operator is ar-

[1] It is an intriguing open question whether one can write in ATL* (or CTL*) a formula that is equivalent to $\mathsf{F_P}$.

guably even more natural than for closed single-agent systems (for which LTL is suited) as it allows one to specify that certain agents should not have the power to unboundedly delay other agents. Hence, we extend ATL* to include the F_P operator. We combine this with the extension of ATL* with epistemic operators that allow one to express what different agents know in a setting with imperfect information.

Reasoning about agents' strategies in open system verification may require to act under partial information about the states of the system (Aminof, Murano, and Vardi 2007; Jamroga and Ågotnes 2007; Aminof et al. 2013; Bulling and Jamroga 2014; Huang and van der Meyden 2014; Jamroga and Murano 2015).In many practical scenarios such as web-banking attacks, card games, market auctions, etc., agents have indeed a limited observability about the state content. One may think of states containing some private information that are visible only to a (possibly empty) subset of players (Reif 1984; Kupferman and Vardi 1997a). Each agent chooses his strategy based on what he can observe. To work with incomplete information systems, the syntax and the semantics of ATL* has been properly extended with epistemic operators (van der Hoek and Wooldridge 2002). The resulting logic is known as KATL*, sometimes simply called ATL* *with imperfect information*.

Our contribution We address the question of verifying prompt branching-time specifications in multi-agent systems for the first time. We present an extension of KATL* with the prompt eventually temporal operator, called Prompt-KATL*. We study the model-checking problem of this logic and some of its fragments. Namely, of Prompt-KATL (ATL with epistemic operators and prompt eventually), Prompt-KCTL* (CTL* with epistemic operators and prompt eventually), and finally the existential fragments of Prompt-KATL* and Prompt-KATL. We show that, for the case of perfect information, model-checking is decidable and not harder than for these logics without the prompt eventually operator. For the case of imperfect information, note that model checking of ATL*, and even ATL, over concurrent game structures with more than two agents and imperfect information is undecidable ((Pnueli and Rosner 1989; Dima and Tiplea 2011)). Moreover, by (Vester 2013), this is already the case for the existential fragment of ATL. However, we show that the imperfect information case is always decidable for Prompt-KCTL*, and for Prompt-KATL* and Prompt-KATL it is decidable in the following two cases: (i) the players are constrained to memoryless strategies, or (ii) the players use memory-full but cooperative strategies and one restricts to the existential fragments of Prompt-KATL* and Prompt-KATL. Furthermore, in all cases, the complexity of our procedures is as good as for the non-prompt version of these logics.[2]

Related work In (Alur et al. 2001), the authors introduce a parameterised extension of LTL in which the temporal operators are associated with variables in order to count the steps between successive occurrences of different events. A fragment of this logic is closely studied in (Kupferman, Piterman, and Vardi 2009), i.e., the extension of LTL by the prompt eventuality operator F_P, called *Prompt* LTL. In (Almagor, Hirshfeld, and Kupferman 2010) the automata-theoretic counterpart of the F_P operator has been also introduced and studied.

Only recently has prompt LTL been studied outside the realm of closed systems. (Zimmermann 2013) studies two-player turn-based games of perfect information with respect to prompt LTL. (Chatterjee, Henzinger, and Horn 2009) lift the prompt semantics to ω-regular games, under the parity winning condition, by introducing *finitary parity* games. They make use of the concept of "*distance*" between positions in a play that refers to the number of edges traversed in the game arena. The classical parity winning condition is then reformulated to take into consideration only those states occurring with a bounded distance. This idea has also been generalised to deal with more involved prompt parity conditions (Fijalkow and Zimmermann 2012; Mogavero, Murano, and Sorrentino 2013). Finally, from a practical point of view, one can see connections with bounded model checking of open systems. Indeed, a central question there is whether a model satisfies a given formula by looking at the model up to depth k. We refer to (Huang, Luo, and Van Der Meyden 2011) for an overview.

Due space limitation, most of the proofs are just sketched.

Definitions

Basic Notation

We denote the set of integers by $\mathbb{N}$, and write $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$. For a set Γ, we write Γ^ω (resp. Γ^*) for the set of infinite (resp. finite) sequences (also called words) of elements in Γ, and Γ^+ for the non-empty finite sequences. We count positions in a sequence starting with 0, and write w_i for the $i'th$ element (called *letter*) of a word w. The length (in $\mathbb{N}_0 \cup \{\infty\}$) of w is written $|w|$. The suffix $w_i w_{i+1} \cdots$ of w is written $w_{\geq i}$ and $w_{\leq i}$ is the prefix $w_0 \cdots w_i$ of w. We usually write a instead of the singleton set $\{a\}$. Given $n \in \mathbb{N}$, we consider a function f with domain $\{1, \ldots, n\}$ as a vector with n coordinates. Thus we may write $f = (f_1, \ldots, f_n)$, and use $f(i)$ and f_i interchangeably.

Game Structures

As for KATL*, models of Prompt-KATL* are *Imperfect Information Concurrent Game Structures (iCGS)*, i.e., structures of the form $\mathsf{S} = \langle Ag, \mathsf{AP}, Act, S, \lambda, \delta, \{\sim_a : a \in Ag\}\rangle$ where:

- Ag is a finite non-empty set of *agents* (also called *players*;
- AP is a finite non-empty set of atoms; Act is a finite non-empty set of *actions* for the agents;
- S is the set of *states* of the game structure;
- $\lambda : S \to 2^{\mathsf{AP}}$ is a *labeling* function that assigns to a state s the set $\lambda(s)$ of atoms that hold in that state;
- $\delta : S \times Act^{Ag} \to S$ is a *transition* function that assigns to every state, and every choice of actions — one for each agent — a successor state;

[2] Except for the case of Prompt-KATL with memoryless strategies for which we only show membership in PSPACE.

- and $\sim_a \subseteq S \times S$ is an equivalence relation representing the imperfect information of agent a, i.e., $s \sim_a s'$ means that agent a cannot distinguish between s and s'.

The equivalence-classes of $\sim_a$ are called the *observation sets* of agent a. The set Act^{Ag} is called the set of *decisions*. An iCGS is called *finite* if the set S is finite. An iCGS has *perfect information* if for every $a \in Ag$ we have that $\sim_a$ is the equality relation, i.e., if $s \sim_a s'$ implies that $s = s'$. In this case it may be written CGS.

Observe that we assume that all agents use the same set of actions Act. However, it is sometimes convenient to assume that each agent a uses only some non-empty subset $Act_a \subseteq Act$ of actions (one can simply define the transition relation to have all actions in $Act \setminus Act_a$ duplicate the effect of some action in Act_a)[3].

Computations and Strategies. A path in S is a finite or infinite sequence $\pi_0\pi_1 \cdots \in S^\omega \cup S^+$ such that for all i there exists a decision $d \in Act^{Ag}$ such that $\pi_{i+1} = \delta(\pi_i, d)$. We call finite paths *histories*, and infinite ones *computations* or *plays*. The set of computations in S is written $\mathsf{cmp}(\mathsf{S})$, and the set of computations in S that start with s is written $\mathsf{cmp}(\mathsf{S}, s)$. We define $\mathsf{hist}(\mathsf{S})$ and $\mathsf{hist}(\mathsf{S}, s)$ similarly.

A *strategy* (for a single agent) is a function $\sigma : \mathsf{hist}(\mathsf{S}) \to Act$. For a non-empty set $A \subseteq Ag$ of agents, and a strategy σ_a for each $a \in A$, write $\Sigma_A := \{\sigma_a : a \in A\}$ for the set of strategies. A path π is *consistent with* Σ_A if it can be obtained by having the agents in A follow their strategies in Σ_A, i.e., if for every position π_i of π there exists $d \in Act^{Ag}$ such that: i) $\pi_{i+1} \in \delta(\pi_i, d)$ and; ii) for every $a \in A$, $d(a) = \sigma_a(\pi_{\leq i})$. The set of computations starting with s that are consistent with Σ_A, written $out(s, \Sigma_A)$, is called the set of *outcomes* of Σ_A from s.

For $A \in Ag$, we derive the following equivalence relations: $\sim_A := \cap_{a \in A} \sim_a$, and $\sim_A^{\mathbb{C}} := (\cup_{a \in A} \sim_a)^*$, where $*$ denotes the transitive closure (with respect to composition). Note that $\sim_{\{a\}} = \sim_a$, and we use these interchangeably.

Extend $\sim_A$ to histories point-wise: if hs and $h's'$ are histories and $h \sim_A h'$ and $s \sim_A s'$ then $hs \sim_A h's'$. A strategy σ is *observational for agent* a if for all $h \sim_a h'$, we have $\sigma(h) = \sigma(h')$. A set Σ_A of strategies for the agents in A is *cooperatively observational* if for all $h \sim_A h'$ and $a \in A$, we have $\sigma_a(h) = \sigma_a(h')$.[4]

2-Player Games. The proofs of Propositions 3 and 4 use the following notions. A *2-player concurrent game* is a pair $\langle \mathsf{S}, \Upsilon \rangle$ where S is an iCGS (called an *arena*) with two players (usually $Ag = \{0, 1\}$), and $\Upsilon \subseteq \mathsf{cmp}(\mathsf{S})$ is a *goal*. A play $\pi \in \mathsf{cmp}(\mathsf{S})$ is *won* by Player 0 if $\pi \in \Upsilon$, and is won by Player 1 otherwise. Given $s \in S$, an observational strategy σ_i for Player $i \in \{0, 1\}$ is called *winning from* s (and we say that Player i *wins from* s) if all plays starting in s that are consistent with σ_i are won by Player i (i.e., if $out(s, \sigma_i) \subseteq \Upsilon$). An LTL formula ψ induces a goal $\Upsilon := \{\pi \in \mathsf{cmp}(\mathsf{S}) \mid \pi \models \psi\}$, and we usually just say that the game has goal ψ. If Player 0 wins from s in the game with goal ψ we say that he *can enforce* ψ *from* s.

[3]More formally, require that for every agent a, there is some action $\alpha \in Act_a$, such that for every $d \in Act^{Ag}$ and $s \in S$, we have: if $d(a) \in Act \setminus Act_a$ then $\delta(s, d) = \delta(s, d')$, where d' is obtained from d by letting $d'(a) = \alpha$.

[4]Agents using cooperative strategies are sometimes referred to as having *distributive knowledge*.

Syntax of Prompt-KATL*.

Fix a finite set of *atomic propositions (atoms)* AP, and a finite set of *agents* Ag. The *Prompt-*KATL^* *state* (φ) *and path* (ψ) *formulas over* AP *and* Ag are built using the following context-free grammar:

$$\varphi ::= p \mid \neg p \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \langle\langle A \rangle\rangle \psi \mid [[A]]\psi \mid \mathbb{K}_a\varphi \mid \mathbb{D}_A\varphi \mid \mathbb{C}_A\varphi \mid \widetilde{\mathbb{K}}_a\varphi \mid \widetilde{\mathbb{D}}_A\varphi \mid \widetilde{\mathbb{C}}_A\varphi$$

and

$$\psi ::= \varphi \mid \psi \vee \psi \mid \psi \wedge \psi \mid \mathsf{X}\psi \mid \psi \,\mathsf{U}\, \psi \mid \psi \,\mathsf{R}\, \psi \mid \mathsf{F_P}\, \psi$$

where p varies over AP, A varies over subsets of Ag, and a over elements of Ag. The class of *Prompt-*KATL^* *formulas* is the set of state formulas generated by the grammar.

The *temporal operators* are X (next), U (until), R (releases, the dual of until) and $\mathsf{F_P}$ (prompt eventually); the *strategy quantifiers* are $\langle\langle A \rangle\rangle, [[A]]$, where $\langle\langle A \rangle\rangle \psi$ is read "the agents in A can enforce ψ", and its dual $[[A]]\psi$ is read "the agents in A cannot avoid ψ"; and the *epistemic operators* are $\mathbb{K}_a$ (agent a knows that), $\mathbb{D}_A$ (the agents in A distributively know that), and $\mathbb{C}_A$ (amongst the agents in A it is common knowledge that), as well as the dual operators $\widetilde{\mathbb{K}}_a, \widetilde{\mathbb{D}}_A, \widetilde{\mathbb{C}}_A$. Observe that, as discussed in the introduction, we do not include a dual operator to $\mathsf{F_P}$. Also note that if one removes the prompt eventually operator then the grammar above generates exactly all the formulas of KATL^* in positive normal form (i.e., with negations pushed all the way to the atoms).

We have the usual syntactic sugar: we write $\mathsf{F}\,\varphi$ (*eventually*) instead of $\mathsf{true}\,\mathsf{U}\,\varphi$; (*globally*) $\mathsf{G}\,\varphi$ instead of $\mathsf{false}\,\mathsf{R}\,\varphi$; E (*exists*) instead of $\langle\langle Ag \rangle\rangle$, and A (*for all*) instead of $[[Ag]]$; $\mathbb{E}$ (*everybody knows*) instead of $\wedge_{a \in Ag}\mathbb{K}_a$ (and its dual $\widetilde{\mathbb{E}}$ for $\vee_{a \in Ag}\widetilde{\mathbb{K}}_a$). Finally, we use a shorthand for repeated next: X^k (for $k \in \mathbb{N}$) is defined as: $\mathsf{X}^1 := \mathsf{X}$, and $\mathsf{X}^{k+1} := \mathsf{X}\,\mathsf{X}^k$.

We now define some important syntactic fragments.

1. Prompt-ATL^* formulas consists of the formulas of Prompt-KATL^* in which no epistemic operator occurs.
2. Prompt-ATL formulas consists of the formulas of Prompt-ATL^* in which every temporal operator is immediately preceded by a strategy quantifier.
3. Prompt-KCTL^* is obtained from Prompt-KATL^* by only allowing strategy quantifiers of the form E and A.
4. Prompt-LTL is the class of path formulas generated by the grammar above in which no strategy quantifier or epistemic operator appears.
5. The *existential fragments* of Prompt-KATL^* and Prompt-KATL consist of those formulas in which the $[[A]]$ quantifier does not occur.

Semantics of Prompt-KATL*.

We first define the semantics of KATL^* (i.e., formulas that don't mention the prompt operator $\mathsf{F_P}$), and then define the semantics of Prompt-KATL^*.

Semantics of KATL*. The satisfaction relation $\models$ is defined inductively, as usual. Formally, for all $s \in S, p \in \mathsf{AP}$:

- $(\mathsf{S}, s) \models p$ iff $p \in \lambda(s)$, and $(\mathsf{S}, s) \models \neg p$ iff $p \notin \lambda(s)$.
- $(\mathsf{S}, s) \models \varphi_1 \wedge \varphi_2$ iff $(\mathsf{S}, s) \models \varphi_1$ and $(\mathsf{S}, s) \models \varphi_2$.
- $(\mathsf{S}, s) \models \varphi_1 \vee \varphi_2$ iff $(\mathsf{S}, s) \models \varphi_1$ or $(\mathsf{S}, s) \models \varphi_2$.
- $(\mathsf{S}, s) \models \langle\langle A \rangle\rangle \psi$ iff there exists a set of observational strategies Σ_A, one for each agent in A, s.t. $(\mathsf{S}, \pi) \models \psi$ for all $\pi \in out(s, \Sigma_A)$.
- $(\mathsf{S}, s) \models [[A]] \psi$ iff for every set of observational strategies Σ_A, one for each agent in A, there is a computation $\pi \in out(s, \Sigma_A)$ s.t. $(\mathsf{S}, \pi) \models \psi$.
- $(\mathsf{S}, s) \models \mathbb{K}_a \varphi$ (resp. $\widetilde{\mathbb{K}}_a \varphi$) iff $(\mathsf{S}, s') \models \varphi$ for every s' (resp. some s') s.t. $s' \sim_a s$.
- $(\mathsf{S}, s) \models \mathbb{D}_A \varphi$ (resp. $\widetilde{\mathbb{D}}_A \varphi$) iff $(\mathsf{S}, s') \models \varphi$ for every s' (resp. some s') s.t. $s' \sim_A s$.
- $(\mathsf{S}, s) \models \mathbb{C}_A \varphi$ (resp. $\widetilde{\mathbb{C}}_A \varphi$) iff $(\mathsf{S}, s') \models \varphi$ for every s' (resp. some s') s.t. $s' \sim_A^{\mathbb{C}} s$.

and for all $\pi \in \mathsf{cmp}(\mathsf{S})$:

- $(\mathsf{S}, \pi) \models \varphi$, for φ a state formula, iff $(\mathsf{S}, \pi_0) \models \varphi$.
- $(\mathsf{S}, \pi) \models \psi_1 \wedge \psi_2$ iff $(\mathsf{S}, \pi) \models \psi_1$ and $(\mathsf{S}, \pi) \models \psi_2$.
- $(\mathsf{S}, \pi) \models \psi_1 \vee \psi_2$ iff $(\mathsf{S}, \pi) \models \psi_1$ or $(\mathsf{S}, \pi) \models \psi_2$.
- $(\mathsf{S}, \pi) \models \mathsf{X}\, \psi$ iff $(\mathsf{S}, \pi_{\geq 1}) \models \psi$.
- $(\mathsf{S}, \pi) \models \psi_1 \,\mathsf{U}\, \psi_2$ iff there $i \in \mathbb{N}$ such that $(\mathsf{S}, \pi_{\geq i}) \models \psi_2$ and for all $j < i$, $(\mathsf{S}, \pi_{\geq j}) \models \psi_1$.
- $(\mathsf{S}, \pi) \models \psi_1 \,\mathsf{R}\, \psi_2$ iff for all i, either $(\mathsf{S}, \pi_{\geq i}) \models \psi_2$ or there exists $j < i$ such that $(\mathsf{S}, \pi_{\geq j}) \models \psi_1$.

We emphasise two points: strategies have perfect recall, and epistemic operators depend only on the current state and not on the history (the latter appears, e.g., in (Čermák et al. 2014)).

Semantics of Prompt-KATL*. For $k \in \mathbb{N}$, we use the notation $(\mathsf{S}, s) \models^k \varphi$ and $(\mathsf{S}, \pi) \models^k \psi$ to denote that the formula is satisfied in which every prompt eventuality is fulfilled within at most k steps. Formally: define $(\mathsf{S}, s) \models^k \varphi$ and $(\mathsf{S}, \pi) \models^k \psi$ inductively, as above,[5] with the following additional rule:

- $(\mathsf{S}, \pi) \models^k \mathsf{F_P}\, \psi$ iff there exists $j \leq k$ such that $(\mathsf{S}, \pi_{\geq j}) \models^k \psi$. Say that π *models* ψ *with bound* k.

Definition 1. *For a Prompt-*KATL* *state-formula* φ *and* $s \in S$*, define* $(\mathsf{S}, s) \models \varphi$ *iff there exists* $k \in \mathbb{N}$ *such that* $(\mathsf{S}, s) \models^k \varphi$*. Say that* φ *is* satisfied *at* s*.*

Linearising branching-formulas and Prompt-LTL

Like CTL*, and ATL* after it, one can think of a Prompt-KATL* path formula ψ over atoms AP as a Prompt-LTL formula $lin(\psi)$ over atoms which are the maximal state subformulas of ψ, as follows (Kupferman, Vardi, and Wolper 2000).

A formula φ is a *state subformula* of ψ if φ is a state formula as well as a subformula of ψ. A formula φ is a *maximal state subformula* of ψ if $\varphi \neq \psi$, it is a state subformula of ψ, and it is not a proper subformula of any other state subformula of ψ. Let $max(\psi)$ be the set of maximal state subformulas of ψ.

Every Prompt-KATL* path formula ψ can be viewed as a Prompt-LTL formula, call it $lin(\psi)$, whose atoms are elements of $max(\psi)$. Formally:

Definition 2. *For a path formula* ψ*, define* $lin(\psi)$ *as follows — in each case note that* $lin(\psi)$ *is a formula over atoms* $max(\psi)$*:*

- $lin(p) := p$ *and* $lin(\neg p) := \neg p$*;*
- *For* $\circ \in \{\langle\langle A \rangle\rangle, [[A]], \mathbb{K}_a, \mathbb{D}_A, \mathbb{C}_A\}$*,* $lin(\circ \psi) := \circ \psi$*;*
- *If* $\psi = \phi_1 \circ \phi_2$ *for* $\circ \in \{\vee, \wedge\}$*, then* $lin(\phi_1 \circ \phi_2)$ *is defined to be* $lin(\phi_1) \circ lin(\phi_2)$ *(note this is well defined since at least one* ϕ_i *is not a state formula);*
- *For* $\circ \in \{\mathsf{X}, \mathsf{F_P}\}$*,* $lin(\circ \phi) := \circ lin(\phi)$*;*
- *For* $\circ \in \{\mathsf{U}, \mathsf{R}\}$*,* $lin(\psi_1 \circ \psi_2) := lin(\psi_1) \circ lin(\psi_2)$*.*

For example, if $\psi = (p \,\mathsf{U} \langle\langle A \rangle\rangle \mathsf{F_P}\, q) \vee \mathsf{X} \neg p$, then its state subformulas are $\{p, \langle\langle A \rangle\rangle \mathsf{F_P}\, q, q, \neg p\}$, and $max(\psi) = \{p, \langle\langle A \rangle\rangle \mathsf{F_P}\, q, \neg p\}$, and thus $lin(\psi)$ is the Prompt-LTL formula $(\boxed{p} \,\mathsf{U}\, \boxed{\langle\langle A \rangle\rangle \mathsf{F_P}\, q}) \vee \mathsf{X} \boxed{\neg p}$ over the atoms $max(\psi)$ (for illustration we box the subformulas that are treated as atoms).

For an iCGS $\mathsf{S} := \langle Ag, \mathsf{AP}, Act, S, \lambda, \delta, \{\sim_a : a \in Ag\}\rangle$, and a Prompt-KATL* path formula ψ over AP, define the iCGS $\mathsf{S}_\psi := \langle Ag, max(\psi), Act, S, \lambda, \delta_\psi, \{\sim_a : a \in Ag\}\rangle$ with atoms $max(\psi)$ and a labeling $\lambda_\psi : S \to 2^{\max(\psi)}$ defined by letting $\varphi \in \lambda_\psi(s)$ iff $(\mathsf{S}, s) \models \varphi$. The next lemma says that a computation π of S satisfies ψ iff, when viewed as a computation of S_ψ, it satisfies $lin(\psi)$:

Lemma 1. *For every iCGS* S*,* Prompt-KATL* *path formula* ψ *over atoms* AP*, and computation* π *of* S*, we have that* $(\mathsf{S}, \pi) \models \psi$ *if and only if* $(\mathsf{S}_\psi, \pi) \models lin(\psi)$*.*

Proof. The proof is by induction on the structure of ψ, using the following inductive hypothesis on the subformulas ϕ of ψ (that are not proper subformulas of any formula in $\max(\psi)$): for every position $i \in \mathbb{N}$ of π, we have that $(\mathsf{S}, \pi_{\geq i}) \models \phi$ iff $(\mathsf{S}_\psi, \pi_{\geq i}) \models lin(\phi)$. □

Prompt linear-temporal logic.

We now consider Prompt-LTL in more detail.

Notation. For a path $\pi \in \mathsf{cmp}(\mathsf{S})$ and a Prompt-LTL formula ψ, write $\pi \models \psi$ instead of $(\mathsf{S}, \pi) \models \psi$. Also, if $v \in (2^{\mathsf{AP}})^\omega$ then we abuse notation and write $v \models \psi$.

Recall from the definitions of syntax that we define Prompt-LTL as the syntactic fragment of Prompt-KATL* in which $\langle\langle A \rangle\rangle$ and $[[A]]$ do not occur. Thus, if ψ is a Prompt-LTL formula and S is a CGS, then $(\mathsf{S}, s) \models \mathsf{A}\, \psi$ expresses that there is a bound $k \in \mathbb{N}$ such that for every $\pi \in \mathsf{cmp}(\mathsf{S})$ starting in s we have that $(\mathsf{S}, \pi) \models^k \psi$.[6]

[5] Thus, e.g., $(\mathsf{S}, \pi) \models^k \psi_1 \,\mathsf{U}\, \psi_2$ iff there $i \in \mathbb{N}$ such that $(\mathsf{S}, \pi_{\geq i}) \models^k \psi_2$ and for all $j < i$, $(\mathsf{S}, \pi_{\geq j}) \models^k \psi_1$.

[6] Remark: the syntax in (Kupferman, Piterman, and Vardi 2009) is different. There they write, e.g., $\mathsf{S} \models \psi$ instead of $\mathsf{S} \models \mathsf{A}\, \psi$.

Proposition 1. *(Kupferman, Piterman, and Vardi 2009) The following problem is decidable: given* Prompt-LTL *formula ψ, a finite CGS S, and a state $s \in S$, decide whether $(\mathsf{S}, s) \models \mathsf{A}\,\psi$. Moreover, the complexity is* PSPACE *in the size of ψ and* NLOGSPACE *in the size of S.*

The previous proposition allows us to deal with universal quantifiers. We now deal with the existential quantifier.

Definition 3. *If ψ is a* Prompt-LTL *formula, define $live(\psi)$ as the* LTL *formula that results from ψ by replacing every $\mathsf{F_P}$ by F.*

The following lemma says that $(\mathsf{S}, s) \models \mathsf{E}\,\psi$ if and only if $(\mathsf{S}, s) \models \mathsf{E}\,live(\psi)$. The reason, roughly, is as follows. For the forward direction, note that if ψ holds promptly on a computation, then in particular, it holds eventually (this is because our formulas are in positive-normal form and so the prompt eventualities can not be negated). For the reverse direction, use the fact that if S has a computation satisfying $live(\psi)$ then it has such a computation that is a lasso, i.e., of the form uv^ω (this holds for all LTL formulas, and thus $live(\psi)$ in particular, (Vardi and Wolper 1994)). In this case, every subformula of ψ that holds in a suffix $\pi_{\geq j}$ of π (with $j \geq |u| + |v|$) also holds in the suffix $\pi_{\geq j-|v|}$. Thus all eventualities hold promptly (i.e., within $|u|+|v|$ steps). The formal proof is in the full version of the paper.

Lemma 2. *If $\pi = uv^\omega$ is a lasso, and ψ is a* Prompt-LTL *formula, then the following are equivalent:*

a) $\pi \models \psi$,

b) $\pi \models live(\psi)$,

c) $\pi \models^b \psi$ *where* $b = |u| + |v|$.

Proof. Clearly $c) \rightarrow a)$. For $a) \rightarrow b)$ an easy induction on ψ shows that for every computation π, and every $k \in \mathbb{N}$, if $\pi \models^k \psi$ then $\pi \models live(\psi)$. For $b) \rightarrow c)$ we prove, by induction on ψ, that for all $i \in \mathbb{N}$, and all subformulas ψ' of ψ: if $\pi_{\geq i} \models live(\psi')$ then $\pi_{\geq i} \models^b \psi'$ (to complete the proof take $i = 0$ and $\psi' = \psi$). The only non-trivial case is $\psi' = \mathsf{F_P}\,\psi_1$. Thus, suppose $\pi_{\geq i} \models live(\mathsf{F_P}\,\psi_1)$. Then $\pi_{\geq i} \models \mathsf{F}\,live(\psi_1)$, and so there exists $j \geq i$ such that $\pi_{\geq j} \models live(\psi_1)$. There are two cases.

Suppose $j < |u|$. By induction, $\pi_{\geq j} \models^b \psi_1$, and so $\pi_{\geq i} \models^{\max\{j-i,b\}} \mathsf{F_P}\,\psi_1$. Since $j - i < |u| - i < b$, we have that $\pi_{\geq i} \models^b \mathsf{F_P}\,\psi_1$, as required.

Suppose $j \geq |u|$. Pick n_0 so that $\max\{i, |u|\} \leq j - |v|n_0 \leq \max\{i, |u|\} + |v|$. Since $\pi = uv^\omega$, we have that $\pi_{\geq j-|v|n_0}$ is equal to π_j. Thus $\pi_{\geq j-|v|n_0} \models live(\psi_1)$, and so by induction $\pi_{\geq j-|v|n_0} \models^b \psi_1$, and so $\pi_{\geq i} \models^{\max\{j-|v|n_0-i,b\}} \mathsf{F_P}\,\psi_1$. Since $j - |v|n_0 - i \leq \max\{i, |u|\} + |v| - i \leq |u| + |v| = b$, we have $\pi_{\geq i} \models^b \mathsf{F_P}\,\psi_1$, as required. □

We now get:

Proposition 2. *For every* Prompt-LTL *formula ψ, CGS S, and state $s \in S$, we have that $(\mathsf{S}, s) \models \mathsf{E}\,\psi$ if and only if $(\mathsf{S}, s) \models \mathsf{E}\,live(\psi)$. Moreover, the cost of checking this fact is* PSPACE *in the size of ψ and* NLOGSPACE *in the size of S.*

Proof. Since $live(\psi)$ is an LTL formula, we have that $(\mathsf{S}, s) \models \mathsf{E}\,live(\psi)$ if and only if there is a lasso $\pi = uv^\omega$ starting in s such that $(\mathsf{S}, \pi) \models live(\psi)$ (Vardi and Wolper 1994). By Lemma 2 this is equivalent to $(\mathsf{S}, \pi) \models^b \psi$ where $b = |u| + |v|$. By definition of $\models^b$, this is equivalent to $(\mathsf{S}, s) \models \mathsf{E}\,\psi$. The complexity follows from the model-checking complexity of LTL (Sistla and Clarke 1985; Kupferman, Piterman, and Vardi 2009). □

The final lemma of this section will be used in the proof of Proposition 4 (used as part of Theorem 4 on co-operative strategies). The lemma relies on the alternating-color technique from (Kupferman, Piterman, and Vardi 2009), that we now describe. Given a computation π of an iCGS S, add a new atomic proposition red, and imagine that states in which red holds are colored red, and otherwise white. Given a Prompt-LTL formula ψ, derive from it an LTL formula $col(\psi)$ by replacing every subformula of the form $\mathsf{F_P}\,\phi$ with a subformula that says that ϕ holds before the color changes twice. Now, if we can come up with a coloring of π in which the colors alternate fast enough (say every k steps or less), such that the colored version of π models $col(\psi)$, then we can deduce that π models ψ with bound $2k$; conversely, if π models ψ with bound k, then by changing colors every k steps we can ensure that $col(\psi)$ is satisfied. This allows us to replace reasoning about Prompt-LTL formulas with reasoning about colorings and LTL formulas. We now formally describe the required elements for applying this reasoning.

For $w \in (2^{\mathsf{AP} \cup \{red\}})^\omega$, a *block* is a maximal subword $w_i \cdots w_j$ of w such that $red \in w_l$ for all $l \in [i, j]$ or $red \notin w_l$ for all $l \in [i, j]$. For $k \in \mathbb{N}$, say that w is *k-coloured* if the size of every block of w has length at most k. Say that $w \in (2^{\mathsf{AP} \cup \{red\}})^\omega$ is a *colouring* of $v \in (2^{\mathsf{AP}})^\omega$ if and only if for every i, $w_i \cap \mathsf{AP} = v_i$. Say that w is a *k-colouring* of v if w is a colouring of v and is k-coloured.

For a Prompt-LTL formula ψ, let ψ' be the LTL formula obtained by replacing every subformula of ψ, of the form $\mathsf{F_P}\,\phi$, by $\mathsf{within}(\phi) := (red\,\mathsf{U}(\neg red\,\mathsf{U}\,\phi)) \vee (\neg red\,\mathsf{U}(red\,\mathsf{U}\,\phi))$, which states that ϕ should hold at some point within this color block or the next. Let $col(\psi) := \psi' \wedge \rho$, where $\rho := \mathsf{G}\,\mathsf{F}(red \wedge \mathsf{X}\neg red)$ states that the colors change infinitely often. It is important to note that while $col(\psi)$ is exponentially larger than ψ, the number of subformulas it has is linear in the number of subformulas of ψ.

Lemma 3. *For every* Prompt-LTL *formula ψ, word $v \in (2^{\mathsf{AP}})^\omega$ and $k \in \mathbb{N}$: (i) if there is a k-colouring w of v such that $w \models col(\psi)$ then $v \models^{2k} \psi$; (ii) if $v \models^k \psi$ then there is a k-colouring w of v such that $w \models col(\psi)$, moreover, w can be chosen with all blocks of size exactly k.*

Proof. Although this lemma is easily extractable from (Kupferman, Piterman, and Vardi 2009), for completeness we provide the proof. For (i), given a k-colouring w of v, if $w \models col(\psi)$ then the result (that $w \models^{2k} \psi$) follows by induction on ψ and the following observation: for every $i \in \mathbb{N}$ and every subformula $\mathsf{F_P}\,\phi$ of ψ, we have $w_{\geq i} \models \mathsf{within}(\phi)$ implies that $v_{\geq i} \models^{2k} \mathsf{F_P}\,\phi$.

For (ii), if $v \models^k \psi$ then colour v by blocks of size exactly k to get w. The result (that $w \models col(\psi)$) follows by induc-

tion on ψ and the following observation: for every $i \in \mathbb{N}$ and every subformula $\mathsf{F_P}\,\phi$ of ψ, we have $v_{\geq i} \models^k \mathsf{F_P}\,\phi$ implies that $w_{\geq i} \models \mathsf{within}(\phi)$. □

Deciding the model-checking problems

The model-checking problem for a logic $\mathcal{L}$ is the following: given a formula φ from $\mathcal{L}$ and a finite iCGS S, decide whether $\mathsf{S} \models \varphi$.

Fact 1. *Model checking* ATL^* *is undecidable over concurrent game structures with* $|Ag| \geq 3$*, and imperfect information, already for the existential fragment (Pnueli and Rosner 1989; Dima and Tiplea 2011).*

Thus, also model checking Prompt-KATL^* is undecidable for three or more agents with imperfect information. We show that one can regain decidability (and we provide the optimal complexity) in four ways: (i) restricting to iCGS with perfect information, or (ii) restricting to path quantifiers (instead of strategy quantifiers), or (iii) restricting to memoryless strategies, or (iv) restricting to cooperative strategies and the existential fragment.

Prompt-ATL^* with perfect information

Proposition 3. *The following problems are decidable: given a* Prompt-LTL *formula* ψ*, a finite (perfect-information) CGS* S*, a set of agents* $A \subseteq Ag$*, and state* $s \in S$*, decide whether* $(\mathsf{S}, s) \models \langle\!\langle A \rangle\!\rangle \psi$*; and, similarly, decide whether* $(\mathsf{S}, s) \models [[A]]\psi$*. Moreover, the complexity of these problems is* 2EXPTIME *in the number of subformulas of* ψ*, and polynomial in the size of* S*.*

Proof. We will reduce each question to the problem of deciding if a given player has a winning strategy in Prompt-LTL turn-based games. The latter are solvable in 2EXPTIME (Zimmermann 2013).

We first consider the case of $\langle\!\langle A \rangle\!\rangle \psi$. In the first step, build a two-player arena G such that $(\star)$: $(\mathsf{S}, s) \models \langle\!\langle A \rangle\!\rangle \psi$ if and only if there exists $k \in \mathbb{N}$ such that Player 0 can enforce (in G) the LTL goal ψ_k from s, where ψ_k is formed from ψ by replacing every subformula of the form $\mathsf{F_P}\,\phi$ by $\bigvee_{i \leq k} \mathsf{X}^i\,\phi$. The idea is that Player 0 corresponds to the coalition A and Player 1 to the coalition $Ag \setminus A$. In more detail: $\mathsf{S} = \langle Ag, \mathsf{AP}, Act, S, \lambda, \delta \rangle$, define G to be the (perfect information) CGS with two agents, Player 0 and Player 1, $\langle \{0,1\}, \mathsf{AP}, Act_0 \cup Act_1, S, \lambda, \delta_G \rangle$ as follows:

- action sets $Act_0 := Act^A$ and $Act_1 := Act^{Ag \setminus A}$,
- state set S, labeling λ,
- transition function $\delta_G : S \times Act_0 \times Act_1 \to S$ that maps $(s, (d_1, d_2)) \mapsto \delta(s, d_1 \otimes d_2)$ where $d_1 \otimes d_2 \in Act^{Ag}$ maps a to $d_1(a)$ for $a \in A$ and otherwise (for $a \in Ag \setminus A$) to $d_2(a)$.

It is immediate from the definitions (of G and $\models$) that $(\star)$ holds. For the next step, recall the following folk fact $(\dagger)$: Player 0 has a winning strategy in a concurrent game G with goal Υ if and only if Player 0 has a winning strategy in the turn-based game G^{tb}, which simulates G as follows: first, Player 0 moves (thus revealing his chosen action) and then Player 1 chooses his action and the simulated move is completed. To "skip" the intermediate nodes that G^{tb} introduces, we use the goal Υ^{tb} which is defined to be the set of all computations such that the subsequence consisting of only the even positioned nodes is in Υ. The intuitive reason that this lemma is true is that in the game G^{tb} Player 0 has no new information available to it, and thus it is exactly as hard (or easy) for him to win as in G.[7]

Formally, we build the turn-based game G^{tb} of perfect information with goal Υ^{tb}_k as follows. Let G^{tb} be the 2-player game, over atoms AP, and such that:

- the state set S^{tb} is $S \cup (S \times Act_0)$,
- the labeling function λ^{tb} maps $s \mapsto \lambda(s)$ and $(s, d) \mapsto \emptyset$,
- the transition function $\delta^{tb} : S^{tb} \times Act_0 \times Act_1 \to S^{tb}$ maps state $s \in S$ and actions (d_0, d_1) to (s, d_0), and maps a state $(s, d_0) \in S \times Act_0$ and actions (d_0', d_1) to $\delta(s, d_0 \otimes d_1)$ (for all $s \in S$, $d_0, d_0' \in Act_0$ and $d_1 \in Act_1$).

Moreover, Υ^{tb}_k consists of those plays whose subsequence consisting of every other node satisfies ψ_k.

Thus, we have reduced our problem to deciding if there exists $k \in \mathbb{N}$ such that Player 0 has a winning strategy in the game G^{tb} with goal Υ^{tb}_k. The latter is the problem of solving a two-player turn-based Prompt-LTL game, which, by (Zimmermann 2013), can be done in 2-EXPTIME. Moreover, that procedure is already able to deal with goals which only look at every second node of the play (so called "blinking semantics"). Thus, the procedure can be easily adapted to decide if there exists $k \in \mathbb{N}$ such that Player 0 has a winning strategy in the game G^{tb} with goal Υ^{tb}_k, which completes the $\langle\!\langle A \rangle\!\rangle \psi$ case.

We turn to the $[[A]]\psi$ case. Dually to the case above, build a two-player game H by associating the coalition A with Player 1 and associating the coalition $Ag \setminus A$ with Player 0. Thus, $(\mathsf{S}, s) \models [[A]]\psi$ if and only if there exists $k \in \mathbb{N}$ such that Player 1 does not have a winning strategy in the game H with goal ψ_k. Dually again, build H^{tb} in which Player 1 moves first and use the fact $(\dagger)$ to deduce that, for every $k \in \mathbb{N}$, Player 1 does not have a winning strategy in H with goal ψ_k if and only if Player 1 does not have a winning strategy in the game H^{tb} with goal Υ^{tb}_k. Since turn-based games of perfect information with LTL goals (the "blinking semantics" is easily accommodated) are determined (i.e., one of the players has a winning strategy), then this is equivalent to Player 0 having a winning strategy in the game G^{tb} with goal Υ^{tb}_k. But this is the same type of game we had already solved in the case of $\langle\!\langle A \rangle\!\rangle \psi$. This completes the $[[A]]\psi$ case. □

Theorem 1. *The model checking problem for* Prompt-ATL^* *(resp.* Prompt-ATL*) is* 2EXPTIME*-complete (resp.* PTIME*-complete).*

Proof. By (Alur, Henzinger, and Kupferman 2002), the lower bound already holds for ATL^* (resp. ATL). For the upper bound, we adapt the marking algorithm for model checking ATL^* (Alur, Henzinger, and Kupferman 2002).

[7] However, Player 1 can win in G^{tb} from states in which he can only prevent Player 0 from winning in G, but not ensure he himself wins.

Let φ be a state Prompt-ATL* formula, and mark every state $s \in S$ by the state subformulas φ' of φ that are satisfied at s. This is done inductively. The case that φ' is an atom, a negation of an atom, a disjunction or a conjunction is immediate (e.g., if $\varphi' = \varphi_1 \vee \varphi_2$ and s is marked by φ_1 then also mark s by $\varphi_1 \vee \varphi_2$).

The case that $\varphi' = \langle\langle A \rangle\rangle \psi$ is dealt with as follows. Recall that $lin(\psi)$ is a Prompt-LTL formula over atoms $max(\psi)$. By Lemma 1, deciding if $(\mathsf{S}, s) \models \langle\langle A \rangle\rangle \psi$ is equivalent to deciding if $(\mathsf{S}_\psi, s) \models \langle\langle A \rangle\rangle lin(\psi)$. By induction, the satisfaction of all state subformulas of ψ have already been determined. In particular, we have enough information to form S_ψ. By Proposition 3, deciding whether $(\mathsf{S}_\psi, s) \models \langle\langle A \rangle\rangle lin(\psi)$ can be done in 2EXPTIME. The case that $\varphi' = [[A]]\psi$ is similar.

The case of Prompt-ATL follows by noting that the formula $lin(\psi)$ is always of constant size since Prompt-ATL does not allow temporal operators to be directly nested. □

Prompt-KCTL*

In the case of Prompt-KCTL*, the strategy quantifiers are restricted to E, A. This inherent restriction on the strategy quantifiers results in a decidable model checking problem, also in the presence of imperfect information.

Theorem 2. *Model checking Prompt-KCTL* is* PSPACE*-complete, and the structure complexity is* PTIME*-complete.*

Proof. The lower-bounds already hold for CTL* (Kupferman, Vardi, and Wolper 2000). For the upper-bounds, let φ be a state Prompt-KCTL* formula, and mark every state $s \in S$ by the state subformulas φ' of φ that are satisfied at s. This is done inductively as in Theorem 1. The cases we have to consider are the epistemic operators and the complexity of the path quantifiers. The case that φ' is of the form $\mathbb{K}_a\phi$ is dealt with as follows: mark s by $\mathbb{K}_a\phi$ iff every $s' \sim_a s$ is already marked by ϕ; similarly, mark s by $\mathbb{D}_A\phi$ iff every $s' \sim_A s$ is already marked by ϕ (the remaining epistemic operators, including the duals, are similar). Each of these steps can be performed in time polynomial in S.

We now discuss the case that φ' is of the form $\mathsf{E}\,\psi$ or $\mathsf{A}\,\psi$. Recall that $lin(\psi)$ is a Prompt-LTL formula over atoms $max(\psi)$, and that $live(\cdot)$ replaces every $\mathsf{F_P}$ by F in a Prompt-LTL formula. Thus, $live(lin(\psi))$ is an LTL formula over atoms $max(\psi)$. By induction, the satisfaction of all state subformulas of ψ has already been determined. In particular, we have enough information to form the CGS S_ψ (note that since S is a CGS, so is S_ψ). By Lemma 1 note that (†): for $Q \in \{\mathsf{E}, \mathsf{A}\}$, $(\mathsf{S}, s) \models Q\psi$ if and only if $(\mathsf{S}_\psi, s) \models Q lin(\psi)$.

For $\varphi' = \mathsf{A}\,\psi$ mark s by $\mathsf{A}\,\psi$ if and only if $(\mathsf{S}_\psi, s) \models \mathsf{A}\,lin(\psi)$. To see this is correct use (†).

For $\varphi' = \mathsf{E}\,\psi$, mark s by $\mathsf{E}\,\psi$ if and only if $(\mathsf{S}_\psi, s) \models \mathsf{E}\,live(lin(\psi))$. To see that this is correct use (†), Proposition 2, and the fact that $lin(\psi)$ is a Prompt-LTL formula over atoms $max(\psi)$.

For the complexity, note that i) there are only a linear number of subformulas φ' of φ, and ii) each case φ' of the algorithm can be done in PSPACE in the size of φ' and NLOGSPACE in the size of S (for the $\varphi' = \mathsf{A}\,\psi$ case use Proposition 1, and for the $\varphi' = \mathsf{E}\,\psi$ case use the fact that model checking LTL is in PSPACE (Sistla and Clarke 1985)). □

Prompt-KATL* with memoryless strategies

In this section we prove that model checking Prompt-KATL* with memoryless strategies is PSPACE-complete. We begin with the relevant definitions.

A strategy σ is called *memoryless* if for all histories h, h', and every state s we have $\sigma(hs) = \sigma(h's)$. It is thus common to consider memoryless strategies as functions from states (not histories) to actions.

Define $\models^k_{mem}$ like $\models^k$ except replace the definition of the semantics of the strategy quantifiers to limit the agents to memoryless (observational) strategies as follows:[8]

- $(\mathsf{S}, s) \models^k_{mem} \langle\langle A \rangle\rangle \psi$ (resp. $[[A]]\psi$) iff there exists a set (resp. for all sets) Σ_A of memoryless observational strategies, one strategy for each agent in A, such that for all (resp. for at least one) computation $\pi \in out(s, \Sigma_A)$, we have $(\mathsf{S}, \pi) \models^k \psi$.

Define $(\mathsf{S}, s) \models_{mem} \varphi$ iff there exists $k \in \mathbb{N}$ such that $(\mathsf{S}, s) \models^k_{mem} \varphi$,

Theorem 3. *The following problem is* PSPACE*-complete: given a Prompt-KATL* formula φ and a finite iCGS S, decide whether $\mathsf{S} \models_{mem} \varphi$.*

Proof. The lower-bound already holds for CTL* (Kupferman, Vardi, and Wolper 2000). For the upper bound, we use the marking algorithm as we did in Theorem 1. We show how to deal with the existential strategy quantifier (the universal quantifier is symmetric). Apply Lemma 1 and reduce $(\mathsf{S}, s) \models_{mem} \langle\langle A \rangle\rangle \psi$ to $(\mathsf{S}_\psi, s) \models_{mem} \langle\langle A \rangle\rangle lin(\psi)$. To solve the latter, observe the following: i) each agent has finitely many observational memoryless strategies (each of polynomial size) in S, ii) instantiating a set Σ_A of such strategies, one for each agent in A, results in a sub-area S' of S_ψ in which we have to check whether $(\mathsf{S}', s) \models \mathsf{A}\,lin(\psi)$. By Proposition 1, each step ii) can be done in PSPACE. Thus one can, in PSPACE, search for a set of observational memoryless strategies Σ_A such that ii) holds. □

Existential Fragment of Prompt-KATL* with co-operative strategies

In this section we prove that model checking the existential fragment of Prompt-KATL* with co-operative strategies is 2EXPTIME-complete. We begin with some definitions.

Define $\models^k_{co}$ like $\models^k$ except replace the definition of the semantics of the strategy quantifiers $\langle\langle A \rangle\rangle, [[A]]$ as follows:

- $(\mathsf{S}, s) \models^k_{co} \langle\langle A \rangle\rangle \psi$ (resp. $[[A]]\psi$) iff there exists a set (resp. for all sets) Σ_A of strategies, one strategy for each agent in A, that is co-operatively observational, such that for all computations (resp. for at least one computation) $\pi \in out(s, \Sigma_A)$ we have $(\mathsf{S}, \pi) \models^k \psi$.

[8] One can also define a "uniform" semantics in which also the agents not quantified over (i.e. the ones not in A) are limited to memoryless strategies. Our techniques can be easily adapted also to this uniform semantics.

Define $(\mathsf{S}, s) \models_{co} \varphi$ iff there exists $k \in \mathbb{N}$ such that $(\mathsf{S}, s) \models^{k}_{co} \varphi$.

Theorem 4. *The model-checking problem for the existential fragment of Prompt-KATL* (resp. Prompt-KATL) with cooperative strategies is* 2EXPTIME*-complete (resp. in* PTIME*).*

The lower bound holds already for the existential fragment of ATL* (see the proof of Theorem 5.6 in (Alur, Henzinger, and Kupferman 2002)). For the upper bound, the proof proceeds as in previous constructions, using the marking algorithm from the proof of Theorem 1. The interesting case is the strategy quantifier $\langle\langle A \rangle\rangle$, which is dealt with by the following proposition.

Proposition 4. *The following problem is decidable: given a Prompt-LTL formula ψ, a finite iCGS S, a set of agents $A \subseteq Ag$, and state $s \in S$, decide whether $(\mathsf{S}, s) \models_{co} \langle\langle A \rangle\rangle \psi$. Moreover, this can be done in* 2EXPTIME *in the number of subformulas of ψ, and polynomial time in the size of S.*

Proof. The outline of the proof is as follows: we build a two-player arena G such that $(\star)$: $(\mathsf{S}, s) \models_{co} \langle\langle A \rangle\rangle \psi$ if and only if there exists $k \in \mathbb{N}$ such that Player 0 has a strategy in G that enforces the LTL goal ψ_k from s (as in Proposition 3, ψ_k is formed from ψ by replacing every subformula of the form $\mathsf{F}_{\mathsf{P}}\, \phi$ by $\bigvee_{i \leq k} \mathsf{X}^i\, \phi$). The idea is that Player 0 corresponds to the coalition A and Player 1 to the coalition $Ag \setminus A$. We are justified in treating all players in A as a single player by our assumption of cooperative strategies for them (the antagonists in $Ag \setminus A$ are always treated as a single player since all of their strategies have to be defeated).

We then modify G to obtain a new arena G', which allows Player 0 to chooses colours (red or $\neg red$) at every second step. We then prove $(\star\star)$ for every $k \in \mathbb{N}$, if Player 0 has an observational strategy in G that enforces goal ψ_k from s then it has an observational strategy in G' that enforces, from s, the following goal $(\dagger)$: $col(\psi)$ holds and no colour consecutively repeats more than k times[9], and vice versa (but with ψ_{2k} substituted for ψ_k, i.e., that if Player 0 has an observational strategy in G' that enforces $(\dagger)$ then it has an observational strategy in G that enforces ψ_{2k}). Finally, using the fact that LTL games of imperfect information admit finite-state strategies, we will show how to decide (in 2EXPTIME) whether there exists $k \in \mathbb{N}$ such that Agent 0 has a strategy in G' that enforces $(\dagger)$ from s. Here are some more details.

If $\mathsf{S} = \langle Ag, \mathsf{AP}, Act, S, \lambda, \delta, \{\sim_a : a \in Ag\}\rangle$, define G to be the iCGS $\langle \{0,1\}, \mathsf{AP}, Act_0 \cup Act_1, S, \lambda, \delta_G, \{\sim_0, \sim_1\}\rangle$ as follows:

- action sets $Act_0 := Act^A$ and $Act_1 := Act^{Ag \setminus A}$,
- state set S, labeling λ,
- transition function $\delta_G : S \times Act_0 \times Act_1 \to S$ that maps $(s, (d_1, d_2)) \mapsto \delta(s, d_1 \otimes d_2)$ where $d_1 \otimes d_2 \in Act^{Ag}$ maps a to $d_1(a)$ for $a \in A$ and otherwise (for $a \in Ag \setminus A$) to $d_2(a)$,
- relation $\sim_0, \sim_1$ defined as follows: $s \sim_0 r$ if $s \sim_A r$, and $s \sim_1 r$ if $s \sim_{Ag \setminus A} r$.

[9] W remind the reader that $col(\psi)$ is defined before Lemma 3.

It is immediate from the definitions (of G and $\models_{co}$) that $(\star)$ holds. Define G' to be the iCGS obtained from G by adding a new atom red and splitting every transition from s to s' using decision d, into a diamond shape, where the first decision is either to go left from s (decision d with red) or go right (decision d with $\neg red$) to one of two intermediate nodes. The choice of colour is made entirely by Player 0. Then, from each of these intermediate nodes every decision leads to the node s'. The observation sets are defined such that the intermediate nodes that are successors of indistinguishable nodes are themselves indistinguishable (and thus no new information leaks). Formally, G' is the iCGS[10] $\langle \{0,1\}, \mathsf{AP} \cup \{red\}, Act', S', \lambda', \delta', \{\sim'_0, \sim'_1\}\rangle$ where:

- $Act'_0 := Act_0 \times \{red, \neg red\}$ and $Act'_1 := Act_1$,
- $S' := S \cup (S \times Act^{Ag} \times \{red, \neg red\})$,
- the labeling λ' maps $s \mapsto \lambda(s)$ and $(s, d, x) \mapsto \{x\}$,
- transition function $\delta' : S' \times Act'_0 \times Act'_1 \to S'$ that, for actions $(d_0, x) \in Act'_0$ and $d_1 \in Act_1$, maps state $s \in S$ and actions $((d_0, x), d_1)$ to $(s, d_0 \otimes d_1, x)$; and that maps state (s, d, x) and all actions of the players to $\delta_G(s, d)$; REMOVED TEXT THAT DOESNT COMPILE
- observation sets $\sim'_0, \sim'_1$ defined as follows: $s \sim'_i r$ if $s \sim_i r$, and $(s, d, x) \sim'_i (r, d', x')$ if $s \sim_i r$.

We now describe, for every $k \in \mathbb{N}$, the natural transformation of strategies for Player 0 between G and G'. For $h' \in \mathsf{hist}(S')$ define $\mathsf{proj}(h') \in \mathsf{hist}(S)$ to be the string in which every second symbol of h' is removed. Fix an action $\alpha \in Act'_0$. An observational strategy $\sigma_0 : \mathsf{hist}(S) \to Act_0$ in G induces the strategy $\sigma'_0 : \mathsf{hist}(S') \to Act'_0$ in G' defined as follows. If h' ends in an element of S then $\sigma'_0(h') := (\sigma_0(\mathsf{proj}(h')), x)$, where x is red if the integer part of $\frac{|\mathsf{proj}(h')|}{k}$ is odd, and otherwise (if it is even), x is $\neg red$ (in words, use σ_0 and swap the colour every k steps in G). If h' does not end in an element of S then define $\sigma'_0(h') := \alpha$ (recall that all transitions from the intermediate nodes go to the same destination). It is easy to see that σ'_0 is observational since σ_0 is.

Before we prove the converse, we state a useful fact that follows from an induction on the length of histories: if $\mathsf{proj}(h'_1) \sim_0 \mathsf{proj}(h'_2)$ then $h'_1 \sim'_0 h'_2$. Now, an observational strategy $\sigma'_0 : \mathsf{hist}(S') \to Act'_0$ in G' induces the strategy $\sigma_0 : \mathsf{hist}(S) \to Act_0$ in G defined as follows: $\sigma_0(h) := \sigma'_0(h')$ where $\mathsf{proj}(h') = h$. This is well defined because, if $\mathsf{proj}(h'_1) = \mathsf{proj}(h'_2) = h$ then $h'_1 \sim'_0 h'_2$ (by the useful fact), and so $\sigma'_0(h'_1) = \sigma'_0(h'_2)$ (since σ'_0 is observational). To see that σ_0 is observational let $h_1 \sim_0 h_2$ be equivalent histories in G, and suppose $\mathsf{proj}(h'_i) = h_i$. Then by the useful fact, also $h'_1 \sim'_0 h'_2$, and thus, since σ'_0 is observational, we have $\sigma'_0(h'_1) = \sigma'_0(h'_2)$.

We now establish $(\star\star)$. Fix k. A strategy σ_0 in G that ensures ψ_k implies (by Lemma 3) that every play consistent with σ_0 can be k-coloured by blocks of size exactly k in a way that satisfies $col(\psi)$. Thus, the transformed strategy σ'_0 of G' ensures $(\dagger)$. Conversely, suppose strategy σ'_0 in G'

[10] We assume that the two agents use different sets of actions Act'_0 and Act'_1 — see discussion after the definition of iCGS.

ensures (†). Then (by Lemma 3) the transformed strategy σ_0 of G ensures ψ_{2k}.

Finally, we show that Player 0 can enforce (†) from s in G' if and only if Player 0 can enforce $col(\psi)$ from s in G'. The "only if" direction is immediate from the definitions. For the "if" direction, it is implicit in (Kupferman and Vardi 1997a; Kupferman and Vardi 1997b) that LTL games of imperfect information admit finite-state strategies. We claim that if a strategy uses memory k then it $(|S| \times k)$-colours the play. Indeed, let h be a history, and take $i < j \leq |h|$ such that the infix between positions i and j is of length at least $|S| \times k$. Hence, there exists $i \leq i' < j' \leq j$ such that the last states of $x := h_{\leq i'}$ and $y := h_{\leq j'}$ are equal, and the strategy is in the same memory-state after processing the histories x and y. In particular, the strategy gives the same action on x as on y. Thus, the opponent can repeatedly play the infix between i' and j', and thus this infix must contain a colour change since we assumed that the strategy enforces $col(\psi)$ (which states, in particular, that the colours alternate infinitely often). This completes the "if" direction. □

Conclusion

Reasoning about promptness has recently received attention for linear specifications (Alur et al. 2001; Kupferman, Piterman, and Vardi 2009; Zimmermann 2013). This is due to the fact that questions like "a specific state is eventually reached in a computation" have a clear meaning and application in the theory of formal verification, but are useless in practical scenarios if there is no bound on the time before the specific state is reached.

In this work, we initiated the study of prompt requirements over branching time specifications for multi-agent systems. We introduced the logic Prompt-KATL*, an extension of the classic ATL*, in which we added the prompt eventuality temporal operator $\mathsf{F_P}$, and settled the model-checking complexity of many natural fragments, under various restrictions, i.e., perfect information, memoryless strategies, and the existential fragment with co-operative strategies. In particular, we prove that model-checking Prompt-KCTL* is PSPACE-complete without any restrictions. Note that in the case of two players the restriction to co-operative strategies is not a real restriction as these correspond to ordinary strategies. Our results for Prompt-KATL* and Prompt-KATL are summarised in the following tables:

Perfect Info. or co-operative ∃ Fragment	Memoryless
2EXPTIME-complete	PSPACE-complete

Figure 1: Complexity of model checking Prompt-KATL*

Perfect Info. or co-operative ∃ Fragment	Memoryless
in PTIME	in PSPACE

Figure 2: Complexity of model checking Prompt-KATL

As our results show, the complexity of model checking the considered logics with the prompt operator is the same as without the prompt operator [11].

We remark that our upper-bounds for Prompt-KATL were obtained essentially as a side-effect of the results for Prompt-KATL*, i.e., by analysing the complexity of the algorithms we provided for Prompt-KATL* in the restricted case of Prompt-KATL. However, one can directly reason on Prompt-KATL. For example, in the perfect-information case, one can show that (like for Prompt-CTL) $\mathsf{S} \models \varphi$, for a Prompt-ATL formula φ, iff $\mathsf{S} \models \phi$, where ϕ is the ATL formula obtained by replacing every prompt-eventuality $\mathsf{F_P}$ in φ by a regular eventuality F. The underlying reason is that in Prompt-KATL $\mathsf{F_P}$ only ranges over state sub-formulas and thus, reasoning about sub-formulas of the form $\langle\langle A \rangle\rangle \mathsf{F_P}\, \psi$ and $[[A]]\, \mathsf{F_P}\, \psi$ reduces to reasoning about two player games with reachability goals. In such games a (memoryless[12]) winning strategy does not admit a lasso in which ψ does not hold on all its states (since then the opponent can pump the loop of the lasso and win). Thus, if the goal ψ is achieved, it is achieved within a number of steps that is at most the size of the model, and thus promptly. For the case of imperfect information, more complicated reasoning can be employed.

Our work opens up several avenues of research. First, an intriguing open question is whether model-checking Prompt-KATL* under cooperative strategies is decidable, even for two players (in this work we established the optimal complexity for its existential fragment). Second, the satisfiability problem has yet to be tackled, and we believe that the techniques introduces in this paper would yield useful for this investigation.

Acknowledgments. Benjamin Aminof and Florian Zuleger are supported by the Austrian National Research Network S11403-N23 (RiSE) of the Austrian Science Fund (FWF) and by the Vienna Science and Technology Fund (WWTF) through grant ICT12-059. Sasha Rubin is a Marie Curie fellow of the Istituto Nazionale di Alta Matematica. Aniello Murano was supported in part by GNCS 2016 project: Logica, Automi e Giochi per Sistemi Auto-adattivi.

References

[Almagor, Hirshfeld, and Kupferman 2010] Almagor, S.; Hirshfeld, Y.; and Kupferman, O. 2010. Promptness in *omega*-regular automata. In *ATVA 2010*, LNCS 6252, 22–36. Springer.

[Alur et al. 2001] Alur, R.; Etessami, K.; La Torre, S.; and Peled, D. 2001. Parametric temporal logic for model measuring. *ACM Transactions on Computational Logic* 2(3):388–407.

[Alur, Henzinger, and Kupferman 2002] Alur, R.; Henzinger, T.; and Kupferman, O. 2002. Alternating-Time Temporal Logic. *Journal of the ACM* 49(5):672–713.

[Aminof et al. 2013] Aminof, B.; Legay, A.; Murano, A.; Serre, O.; and Vardi, M. Y. 2013. Pushdown module checking with imperfect information. *Inf. Comput.* 223:1–17.

[11] except for the case of Prompt-KATL under observational memoryless strategies.

[12] Winning strategies in such games can be taken w.l.o.g. to be memoryless.

[Aminof, Murano, and Vardi 2007] Aminof, B.; Murano, A.; and Vardi, M. Y. 2007. Pushdown module checking with imperfect information. In *CONCUR 2007*, 460–475.

[Bulling and Jamroga 2014] Bulling, N., and Jamroga, W. 2014. Comparing variants of strategic ability: how uncertainty and memory influence general properties of games. *Journal of Autonomous Agents and Multi-Agent Systems* 28(3):474–518.

[Čermák et al. 2014] Čermák, P.; Lomuscio, A.; Mogavero, F.; and Murano, A. 2014. MCMAS-SLK: A Model Checker for the Verification of Strategy Logic Specifications. In *CAV'14*, LNCS 8559, 524–531. Springer.

[Čermák, Lomuscio, and Murano 2015] Čermák, P.; Lomuscio, A.; and Murano, A. 2015. Verifying and Synthesising Multi-Agent Systems against One-Goal Strategy Logic Specifications. In *AAAI 2015*, 2038–2044. AAAI Press.

[Chatterjee, Henzinger, and Horn 2009] Chatterjee, K.; Henzinger, T. A.; and Horn, F. 2009. Finitary winning in ω-regular games. *ACM Transactions on Computational Logic* 11(1):1.

[Chatterjee, Henzinger, and Piterman 2010] Chatterjee, K.; Henzinger, T.; and Piterman, N. 2010. Strategy Logic. *Information and Computation* 208(6):677–693.

[Clarke and Emerson 1981] Clarke, E., and Emerson, E. 1981. Design and Synthesis of Synchronization Skeletons Using Branching-Time Temporal Logic. In *LP'81*, LNCS 131, 52–71. Springer.

[Dima and Tiplea 2011] Dima, C., and Tiplea, F. 2011. Model-checking ATL under Imperfect Information and Perfect Recall Semantics is Undecidable. Technical report, arXiv.

[Emerson and Halpern 1986] Emerson, E., and Halpern, J. 1986. "Sometimes" and "Not Never" Revisited: On Branching Versus Linear Time. *Journal of the ACM* 33(1):151–178.

[Fijalkow and Zimmermann 2012] Fijalkow, N., and Zimmermann, M. 2012. Cost-parity and cost-streett games. In *FSTTCS*, volume LIPIcs 18, 124–135.

[Huang and van der Meyden 2014] Huang, X., and van der Meyden, R. 2014. An epistemic strategy logic. *arXiv preprint arXiv:1409.2193*.

[Huang, Luo, and Van Der Meyden 2011] Huang, X.; Luo, C.; and Van Der Meyden, R. 2011. Improved bounded model checking for a fair branching-time temporal epistemic logic. In *MoChArt*. Springer. 95–111.

[Jamroga and Ågotnes 2007] Jamroga, W., and Ågotnes, T. 2007. Constructive knowledge: what agents can achieve under imperfect information. *J. Applied Non-Classical Logics* 17(4):423–475.

[Jamroga and Murano 2015] Jamroga, W., and Murano, A. 2015. Module checking of strategic ability. In *AAMAS 2015*, 227–235.

[Kupferman and Vardi 1997a] Kupferman, O., and Vardi, M. Y. 1997a. Module checking revisited. In *CAV'97*, 36–47. Springer.

[Kupferman and Vardi 1997b] Kupferman, O., and Vardi, M. 1997b. Synthesis with incomplete informatio. In *2nd International Conference on Temporal Logic*, 91–106.

[Kupferman, Piterman, and Vardi 2009] Kupferman, O.; Piterman, N.; and Vardi, M. Y. 2009. From liveness to promptness. *Formal Methods in System Design* 34(2):83–103.

[Kupferman, Vardi, and Wolper 2000] Kupferman, O.; Vardi, M.; and Wolper, P. 2000. An Automata Theoretic Approach to Branching-Time Model Checking. *Journal of ACM* 47(2):312–360.

[Lomuscio, Qu, and Raimondi 2009] Lomuscio, A.; Qu, H.; and Raimondi, F. 2009. MCMAS: A Model Checker for the Verification of Multi-Agent Systems. In *CAV'09*, LNCS 5643, 682–688. Springer.

[Mogavero et al. 2014] Mogavero, F.; Murano, A.; Perelli, G.; and Vardi, M. 2014. Reasoning About Strategies: On the Model-Checking Problem. *ACM Transactions on Computational Logic* 15(4):34:1–42.

[Mogavero, Murano, and Sorrentino 2013] Mogavero, F.; Murano, A.; and Sorrentino, L. 2013. On Promptness in Parity Games. In *LPAR'13*, 601–618. Springer.

[Pnueli and Rosner 1989] Pnueli, A., and Rosner, R. 1989. On the Synthesis of a Reactive Module. In *POPL'89*, 179–190. Association for Computing Machinery.

[Queille and Sifakis 1981] Queille, J., and Sifakis, J. 1981. Specification and Verification of Concurrent Programs in Cesar. In *International Symposium on Programming'81*, LNCS 137, 337–351. Springer.

[Reif 1984] Reif, J. H. 1984. The complexity of two-player games of incomplete information. *J. Comput. Syst. Sci.* 29(2):274–301.

[Sistla and Clarke 1985] Sistla, A. P., and Clarke, E. M. 1985. The complexity of propositional linear temporal logics. *J. ACM* 32(3):733–749.

[van der Hoek and Wooldridge 2002] van der Hoek, W., and Wooldridge, M. 2002. Tractable Multiagent Planning for Epistemic Goals. In *Autonomous Agents and Multiagent Systems'02*, 1167–1174.

[Vardi and Wolper 1994] Vardi, M. Y., and Wolper, P. 1994. Reasoning about infinite computations. *Information and Computation* 115(1):1–37.

[Vester 2013] Vester, S. 2013. Alternating-time temporal logic with finite-memory strategies. In *GandALF 2013*, 194–207.

[Zimmermann 2013] Zimmermann, M. 2013. Optimal bounds in parametric LTL games. *Theor. Comput. Sci.* 493:30–45.