A Logic with Revocable and Refinable Strategies

Christophe Chareton, Julien Brunel*, David Chemouil

ONERA/DTIM, *2 avenue Édouard Belin, F-31055 Toulouse*

Abstract

In this article, we present Updatable Strategy Logic (USL), a multi-agent temporal logic which subsumes the main propositions in this area, such as ATL, ATL*, ATL_{sc} and SL. These logics allow to express the capabilities of agents to ensure the satisfaction of temporal properties. USL mainly differs from SL, in two ways. Semantically, USL relies on *multi-strategies* (that is, non-deterministic strategies), and the notion of strategy composition is extended to enable an agent to refine her strategy, that is to update it without revoking it. Syntactically, the logic features a binding operator enabling *multi-strategy* refinements as well as an "unbinding" operator that allows an agent to explicitly revoke a *multi-strategy* (whereas revocation is implicit with the binding operator in SL). We show that USL allows to express a notion of *sustainable control* for an agent, *i.e.*, a capability to always decide the satisfaction of a property, that still holds even after the said capability has been employed. Furthermore, USL allows the definition of a notion of equality between *multi-strategies* that enables to consider quantification over deterministic strategies. This makes USL strictly more expressive than SL. Finally, we also show that the model-checking problem for USL is decidable but non-elementary (as for SL).

Keywords: multi-agent temporal logic, strategic reasoning, non-deterministic strategies, refinement, revocation

1. Introduction

Multi-agent logics are receiving growing interest in contemporary research. Since the seminal work of R. Alur, Th. A. Henzinger, and O. Kupferman [1], increasing efforts have been made to formalise agent interactions and strategies in game structures.

Basically, multi-agent logics allow to formulate assertions about the ability of *agents* to ensure temporal properties. Thus, ATL (resp. ATL*) appears as a generalization of CTL (resp. CTL*) in which the path quantifiers E and A are replaced by quantifiers $\langle\!\langle\cdot\rangle\!\rangle$ and $[\![\cdot]\!]$. Such a quantifier takes a *coalition* (set) of agent(s) as parameter: for instance, $\langle\!\langle A\rangle\!\rangle\varphi$ specifies that *there is* a way for agents in A to ensure the satisfaction of the temporal formula φ (the universal quantifier is written $[\![A]\!]$). Such formulas are usually interpreted in *Concurrent Game Structures* (CGS's), where every agent make choices influencing the execution of the system depending on her *strategy* (a function indicating an action depending on the history of states already met). Here for instance, $\langle\!\langle A\rangle\!\rangle\varphi$ is true in a given CGS if every agent a in A has a strategy ς_a s.t. if all agents in A play their respective strategy ς_a, they force the system execution to satisfy φ, *whatever the other agents in the CGS do*.

1.1. Strategy Composition

Since these logics allow to reason about agent interactions, the ability to address the *composition of strategies*, which comes to nesting strategy quantifiers in a formula, is a major issue that has enjoyed several developments. To illustrate this aspect and introduce our contribution, consider the following ATL formula:

$$\langle\!\langle a_1\rangle\!\rangle\Box(\varphi_1 \wedge \langle\!\langle a_2\rangle\!\rangle\Box\varphi_2) \tag{1}$$

*Corresponding author.

Email addresses: Christophe.Chareton@onera.fr (Christophe Chareton), Julien.Brunel@onera.fr (Julien Brunel), David.Chemouil@onera.fr (David Chemouil)

where $\Box$ is the usual "always" operator and $\langle\langle a_1 \rangle\rangle$ is a shorthand for $\langle\langle \{a_1\} \rangle\rangle$ (similarly for $\langle\langle a_2 \rangle\rangle$), *i.e.* we only consider one-agent coalitions.

In the following, we study the avatars of this formula in various logics, depending on whether a_1 and a_2 stand for the same agent or not (so, in full rigour, a_1 and a_2 are meta-variables here).

In ATL and ATL*, the operator $\langle\langle a_2 \rangle\rangle$ drops the strategies introduced by any earlier quantifier (here $\langle\langle a_1 \rangle\rangle$). So during the evaluation of $\Box\varphi_2$, the strategy adopted by a_1 is not taken into account. Notice that this situation happens whether a_1 and a_2 are *the same agent or not*. Generally speaking, the interpretation of an operator $\langle\langle A \rangle\rangle$ in ATL and ATL* induces an implicit *revocation* of their current strategy by every agent in the system.

ATL$_{\text{sc}}$ [2, 3, 4], while keeping the ATL syntax, adapts the semantics in order to interpret formulas in a *context* which stores strategies introduced by earlier quantifiers. This way, if a_1 and a_2 are two *different agents*, then, during the evaluation of Formula (1) under the ATL$_{\text{sc}}$ semantics, when reaching $\Box\varphi_2$, the strategy adopted by a_1 is read from the current context and taken into account. Said otherwise, it is *not revoked*. Note however that revocation still happens if a_1 and a_2 are the same agent.

1.2. Explicit Strategy Binding

Strategy Logic (SL), which uses similar strategy contexts, is the starting point for our contribution. (Notice there are two versions of SL in the literature: the first one [5] considered interactions between two players in a turn-based game. It was later extended to concurrent games between any finite number of agents [6, 7, 8]. In the remainder of this paper, we only refer to the latter.)

In SL, the ATL operator $\langle\langle a \rangle\rangle$ is split into two different operators: an existential quantifier over strategies $\langle\langle x \rangle\rangle$, where x is a *strategy variable*, and a *binder* (a, x), which stores into a context the information that a follows a strategy ς_x instantiating the variable x. In order to illustrate the SL syntax, consider the following SL formula that is equivalent to Formula (1) under the ATL$_{\text{sc}}$ semantics:

$$\langle\langle x_1 \rangle\rangle(a_1, x_1)[\![x_2]\!](a_2, x_2) \dots [\![x_k]\!](a_k, x_k)\Box(\varphi_1 \wedge \langle\langle x_2 \rangle\rangle(a_2, x_2)\Box\varphi_2) \tag{2}$$

where $[\![x]\!]$ is the universal quantifier over strategy variables (dual of $\langle\langle x \rangle\rangle$) and $\{a_1, \dots a_k\}$ is the set of agents in the system under consideration. (Note that in Formula (2), each agent in the system is explicitly bound, which is due to a syntactic constraint of SL, see Remark 1 later.)

In SL, the composition of contexts is defined in a quite natural way (as far as strategies concern distinct agents) through the semantics of the nesting of binders. For instance, as for ATL$_{sc}$, when evaluating $\Box\varphi_2$ in Formula (2), a_1 remains bound to strategy ς_{x_1}, except if a_1 and a_2 are the same agent. In which case, ς_{x_1} is revoked before adding a binding to ς_{x_2} in the context, *i.e.* the former binding stored in the context is *overwritten*.

1.3. Explicit Strategy Unbinding

Thus, in SL (resp. ATL$_{sc}$), a strategy binder (a, x) (resp. a strategy quantifier $\langle\langle A \rangle\rangle$), does not perform any *implicit revocation* of the current strategies for the agents distinct from a (resp. not in A). However, an implicit revocation still holds for the agent a (or the members of coalition A in ATL$_{\text{sc}}$). So, it is not possible to express what we call a *strategy refinement* for an agent, *i.e.*, the possibility to *enrich* her own strategy, instead of revoking it.

In this paper, building on previous work introduced in [9], we present Updatable Strategy Logic (USL), a logic allowing to consider strategies that are *updatable*, *i.e.* refinable or revocable explicitly. The syntax of USL formulas features (1) a *binder* $(a \rhd x)$ enabling to associate a strategy with an agent *without overwriting strategies previously bound to her*; and (2) an *unbinder* $(a \not\rhd x)$ allowing an agent to revoke a strategy *explicitly.* Thus, if a_1 and a_2 are the same agent, say a, then Formula (2) translates as:

$$\langle\langle x_1 \rangle\rangle(a \rhd x_1)\Box(\varphi_1 \wedge \langle\langle x_2 \rangle\rangle(a \not\rhd x_1)(a \rhd x_2)\Box\varphi_2)$$

while the following formula, without an unbinder, asserts that *a refines* the strategy instantiating x_1 with the one instantiating x_2:

$$\langle\langle x_1 \rangle\rangle(a \rhd x_1)\Box(\varphi_1 \wedge \langle\langle x_2 \rangle\rangle(a \rhd x_2)\Box\varphi_2)$$

A key point is that, instead of classical strategies, USL makes uses of non-deterministic strategies, called *multi-strategies*. The non-determinism of multi-strategies makes them refinable and an agent may then be committed to several multi-strategies. We define the syntax and semantics of USL in detail in Sect. 2.

As shown in Sect. 3, one consequence of considering commitments to several multi-strategies is that it allows to characterise a particular property, called *sustainable control*, which allows to reason about situations where an agent (1) has a choice between different properties she can alternatively ensure, and (2) after using it she still has the choice between the same properties. Another consequence is that, in case of "contradictory" multi-strategies, the "execution" of the system *stops*. We regard this property as fruitful as it allows to *compare* multi-strategies, to characterise *deterministic* multi-strategies (thus covering SL strategies) or to express notions of dependences between multi-strategies.

In Sect. 4, we prove that USL is *strictly more expressive* than SL and that the satisfiability problem for USL is not decidable. On the other hand, as shown in Sect. 5 (adapting a proof from [7]), the model-checking problem on finite models for USL enjoys the same complexity as SL, that is NonElementary in time over the length of the formula and Polynomial over the size of the CGS.

Finally, in Sect. 6 and 7, we compare our results to related works and sketch directions for future work.

Technical Conventions and Notations Used in this Paper

The notation for a *partial function* from a set A to a set B is $f : A \nrightarrow B$. We write $\mathrm{dom}(f)$ for the *domain of definition* of a partial function f.

Given a binary relation $R \subseteq A \times B$, for any sets A and B, we write $R(a)$ for the set $\{b \in B \mid \langle a, b\rangle \in R\}$ and $\mathrm{dom}(R)$ for the *left projection* of R.

Given a set S, we write $|S|$ for the *cardinal* of S, $\mathcal{P}(S)$ for the set of subsets of S, $\mathcal{P}_{>0}(S)$ for the set of non-empty subsets of S, and $\mathcal{P}_{>0}^{<\omega}(S)$ for the set of non-empty finite subsets of S.

Given a set S, we write S^* for the set of possibly-empty finite sequences over S, S^+ for the set of non-empty finite sequences over S, and S^ω for the set of possibly-empty finite and infinite sequences over S.

Given a sequence λ, we write λ_0 for its first element and λ_i for its $(i+1)$-th element. The length of a finite sequence λ (*i.e.* the number of elements in the sequence) if written $|\lambda|$ and its last element is written $\mathrm{last}(\lambda)$. We also use the extensive notation $\lambda_0 \cdot \lambda_1 \cdots \lambda_{|\lambda|-1}$. Note that in the case $|\lambda| = 1$, then λ_0 denotes both a sequence and its unique element.

We also write $\lambda_{\leqslant i}$ for the finite subsequence $(\lambda_0, \ldots, \lambda_i)$ of λ, and $\lambda_{\geqslant i}$ for the subsequence of λ starting at index i.

2. Syntax and Semantics

This section introduces USL. First, we recall the standard definitions regarding the semantic framework used for interpretation, namely *concurrent game structures.* Second, we define the syntax of USL. Then we introduce a number of useful technical definitions before defining the formal semantics of the logic. In all these steps, we follow most of the structure and notations of SL [7] to facilitate the further comparisons in expressive power done in Sect. 4.

2.1. Semantic Framework

The semantic framework used to interpret USL formulas is that of *concurrent game structures*, introduced in [1] and then subsequently used with slight modifications in numerous works [2, 4, 6, 7]. As USL builds upon SL, our definition for CGS's is the one from [6, 7] (the main difference between this and the one from [1] lies in the cardinalities of *St* and *Act*: here they are enumerable whereas they are finite in [1]).

Intuitively, a concurrent game structure is an extension of labelled transition systems dedicated to modelling multi-agent systems. In these systems, transitions are decided by *actions* a set of *agents* perform. At each state of any execution, each agent plays an action and the transition is determined by these actions and the current state.

Definition 1 (Concurrent Game Structure). A *concurrent game structure* is a tuple $\mathcal{G} = \langle Ag, St, At, prop, Act, tr, s_o\rangle$ where:

- Ag is a finite non-empty set of agents
- St is an enumerable non-empty set of *states*
- At is a finite non-empty set of *atomic propositions*

- $prop : St \to \mathcal{P}(At)$ is a valuation function which maps each state s to the set of propositions true at s
- Act is an enumerable non-empty set of actions
- Let us write $Dec = Act^{Ag}$ for the set of *decisions*, *i.e.* maps from agents to actions representing the choices of an action for every agent. Then $tr : St \times Dec \to St$ is the *transition function* which maps a state and a decision to a state.
- $s_0 \in St$ is an *initial state*

Now, we define *tracks* and *paths* as non-empty finite and infinite (resp.), legal, sequences of states in a CGS.

Definition 2 (Track, Path, Execution). Let $\mathcal{G} = \langle Ag, St, At, prop, Act, tr, s_o \rangle$ be a CGS. A *track* in $\mathcal{G}$ is a non-empty finite sequence of states $\theta \in St^+$ s.t. for every $i \in [0, |\theta| - 1[$, there is a decision $\delta \in Dec$ s.t. $\theta_{i+1} = tr(\theta_i, \delta)$.

A *path* in $\mathcal{G}$ is an infinite sequence of states $\eta \in St^\omega$ s.t. every finite prefix of η is a track in $\mathcal{G}$.

The set of all tracks (resp. paths) in $\mathcal{G}$ is written $Track_\mathcal{G}$ (resp. $Path_\mathcal{G}$) (the index will be omitted in case there is no ambiguity).

An *execution* in $\mathcal{G}$ is a track or a path in $\mathcal{G}$.

We can now define the notion of multi-strategy (the use of this word is inspired by [10]). A multi-strategy indicates a set of actions that may be performed by an agent depending on the history of states already met. Notice that, as usual in the literature, a multi-strategy is not attached to a single agent as multiple agents may follow the same strategy.

Note that, contrary to a number of formalisms, we allow our strategies to be non-deterministic. In practice, this approach typically encompasses situations where strategies are under-specified.

In practice, a multi-strategy which is not explicitly specified for a given track is considered to yield the set Act of all actions.

Definition 3 (Multi-strategy). Let $\mathcal{G} = \langle Ag, St, At, prop, Act, tr, s_o \rangle$ be a CGS. A multi-strategy ς in $\mathcal{G}$ is a map which, given a track, yields a non-empty set of actions, *i.e.* $\varsigma : Track \to \mathcal{P}_{>0}(Act)$.

The set of multi-strategies in $\mathcal{G}$ is written $MStrat_\mathcal{G}$ (the index will be omitted in case there is no ambiguity).

We now define the notion of *multi-strategy translation* along a track θ, which represents the way a multi-strategy is to be "updated" to take into account the fact that θ is now added to the history of a game.

Definition 4 (Multi-strategy Translation). Let $\mathcal{G} = \langle Ag, St, At, prop, Act, tr, s_o \rangle$ be a CGS, ς be a multi-strategy and θ be a track in $\mathcal{G}$. We call *translation of ς along θ* the multi-strategy ς^θ s.t. for any track θ', $\varsigma^\theta(\theta') = \varsigma(\theta \cdot \theta'_{\geqslant 1})$.

2.2. Syntax

The syntax of USL makes a distinction between state and path formulas. We first describe a notion of pseudo-formulas and then define formulas as pseudo-formulas where every quantified multi-strategy variable is fresh w.r.t. the scope in which it is introduced.

Definition 5 (Pseudo-formulas). Let Ag be a set of agents, At a set of propositions, and Var a set of (multi-strategy) variables. Then the set of *USL pseudo-formulas* is defined by the following grammar:

- *State pseudo-formulas:*
$$\psi ::= p \mid \neg\psi \mid \psi \wedge \psi \mid \langle\langle x \rangle\rangle \psi \mid (A \rhd x)\varphi \mid (A \not\rhd x)\varphi$$
- *Path pseudo-formulas:*
$$\varphi ::= \psi \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathsf{X}\varphi \mid \varphi \,\mathsf{U}\, \varphi$$

where $p \in At$, $x \in Var$ and A is a *coalition* (*i.e.* $A \subseteq Ag$).

Note that in this grammar, path formulas are introduced exclusively after $(A \rhd x)$ or $(A \not\rhd x)$. In particular, a path formula cannot follow $\langle\!\langle x \rangle\!\rangle$. This restriction avoids counter-intuitive effects in the semantics definitions. As we will see in Sect. 4.2, this is made without loss of generality.

Furthermore, the semantics of these operators includes a quantification over the executions that are made possible by the context. This is not detailed in the present article, but it is possible to simulate branching-time specifications with USL.

Notation. The propositional connective $\vee$ ("or") and constants $\top$ ("true") and $\bot$ ("false") are defined in the obvious way. In the same way, we make use of the common temporal connectors $\Diamond$ ("eventually"), $\Box$ ("always") and R ("release"), defined by: $\Diamond\varphi \triangleq \top\,\mathsf{U}\,\varphi$, $\Box\varphi \triangleq \neg\Diamond\neg\varphi$, and $\varphi_1\,\mathsf{R}\,\varphi_2 \triangleq \neg(\neg\varphi_1\,\mathsf{U}\,\neg\varphi_2)$. Finally, the universal quantifier on multi-strategies is defined as follows: for every variable x and every pseudo-formula φ, $[\![x]\!]\varphi \triangleq \neg\langle\!\langle x \rangle\!\rangle\neg\varphi$.

Definition 6 (Sub-pseudo-formulas). The set $\mathrm{Sub}(\varphi)$ of *sub-pseudo-formulas* of a pseudo-formula φ is defined by induction over φ:

- $\mathrm{Sub}(p) = \{p\}$, with $p \in At$
- $\mathrm{Sub}(\neg\varphi) = \{\neg\varphi\} \cup \mathrm{Sub}(\varphi)$
- $\mathrm{Sub}(\varphi_1 \wedge \varphi_2) = \{\varphi_1 \wedge \varphi_2\} \cup \mathrm{Sub}(\varphi_1) \cup \mathrm{Sub}(\varphi_2)$
- $\mathrm{Sub}(\langle\!\langle x \rangle\!\rangle\varphi) = \{\langle\!\langle x \rangle\!\rangle\varphi\} \cup \mathrm{Sub}(\varphi)$
- $\mathrm{Sub}((A \rhd x)\varphi) = \{(A \rhd x)\varphi\} \cup \mathrm{Sub}(\varphi)$
- $\mathrm{Sub}((A \not\rhd x)\varphi) = \{(A \not\rhd x)\varphi\} \cup \mathrm{Sub}(\varphi)$
- $\mathrm{Sub}(\mathsf{X}\varphi) = \{\mathsf{X}\varphi\} \cup \mathrm{Sub}(\varphi)$
- $\mathrm{Sub}(\varphi_1\,\mathsf{U}\,\varphi_2) = \{\varphi_1\,\mathsf{U}\,\varphi_2\} \cup \mathrm{Sub}(\varphi_1) \cup \mathrm{Sub}(\varphi_2)$

As multi-strategy variable *names* are taken into account in the semantics of formulas, some care must be taken when a quantifier is encountered. Thus, well-formed formulas are pseudo-formulas such that every quantifier introduces a fresh multi-strategy variable w.r.t. the scope in which it appears.

Definition 7 ((Well-Formed) Formula). A pseudo-formula φ is a *(well-formed) formula* if for any sub-pseudo-formula $\langle\!\langle x \rangle\!\rangle\varphi'$ of φ and every sub-pseudo-formula $\langle\!\langle y \rangle\!\rangle\varphi''$ of φ', x and y are distinct variables.

It is straightforward to check that any sub-pseudo-formula of a formula is itself a formula.

Definition 8 (Subformulas). Given a formula φ, the set of *subformulas* of φ is defined as its set of sub-pseudo-formulas and is also denoted by $\mathrm{Sub}(\varphi)$.

Definition 9 (Free Variables). The set of *free variables* of a formula φ, written $\mathrm{FV}(\varphi)$, is defined by induction on φ:

- $\mathrm{FV}(p) = \emptyset$, for $p \in At$
- $\mathrm{FV}(\neg\varphi) = \mathrm{FV}(\mathsf{X}\varphi) = \mathrm{FV}(\varphi)$
- $\mathrm{FV}(\varphi_1 \wedge \varphi_2) = \mathrm{FV}(\varphi_1\,\mathsf{U}\,\varphi_2) = \mathrm{FV}(\varphi_1) \cup \mathrm{FV}(\varphi_2)$
- $\mathrm{FV}(\langle\!\langle x \rangle\!\rangle\varphi) = \mathrm{FV}(\varphi) \setminus \{x\}$
- $\mathrm{FV}((A \rhd x)\varphi) = \mathrm{FV}((A \not\rhd x)\varphi) = \mathrm{FV}(\varphi) \cup \{x\}$

Notation. The *length* $|\varphi|$ of a formula φ is the number of its subformulas.

Definition 10 (Sentence). A formula φ is called a *sentence* if it is *closed,* that is if $\mathrm{FV}(\varphi) = \emptyset$.

Remark 1 (Comparison with SL Sentences). In SL [6, 7], a sentence is a formula that is both variable-closed *and agent-closed.* The latter condition requires that all agents are bound to a strategy variable before any temporal subformula. For instance, given a set of agents $\{a_1, \ldots, a_n\}$ (with $n \in \mathbb{N}$), the formula $\langle\!\langle x_n \rangle\!\rangle (a_n, x_n) \mathsf{X} p$ is not a sentence. In order to be a sentence, a reference to all agents other than a_n must be added, as in the following formula: $\langle\!\langle x_n \rangle\!\rangle (a_n, x_n) [\![x_1]\!] (a_1, x_1) \ldots [\![x_{n-1}]\!] (a_{n-1}, x_{n-1}) \mathsf{X} p$.

As a consequence, writing SL sentences can sometimes be tedious. More importantly, a single informal specification may give rise to different SL sentences depending on the number of agents of the system not mentioned in the specification. In contrast, in USL as in ATL [1] or ATL$_{sc}$ [2, 4], this syntactic constraint is not present.

2.3. Semantics

We now give a number of technical definitions that are required to define the notion of satisfaction between a CGS and a formula. We first define the *evaluation contexts* that are used for USL, and different operations on these contexts (Sect. 2.3.1). In Sect.2.3.2 we introduce the set of *outcomes* of an *evaluation context* and a state. Then we give the definition of satisfaction in Sect.2.3.3.

2.3.1. Evaluation Contexts

As in SL or ATL$_{sc}$, USL relies on a notion of "environment" (called "assignment" in SL [7]) to evaluate the subformulas of a sentence. However, contrary to SL, we make a distinction between two parts, namely *assignments*, that keep track of multi-strategy variable instantiations, and *commitments*, that store the bindings of agents to multi-strategy variables. This distinction is important because USL allows to *compose* multi-strategies so that a given agent may be committed to *several* multi-strategies at once.

Definition 11 (Assignment, Commitment, Context). An *assignment* α is a dictionary which associates a multi-strategy with every multi-strategy variable in its domain of definition, *i.e.* a partial function $\alpha : \mathit{Var} \nrightarrow \mathit{MStrat}$.

A *commitment* γ gathers bindings between a set of agents and variables they are bound to, *i.e.* it is a relation $\gamma \subseteq \mathit{Ag} \times \mathit{Var}$.

A *context* χ is a "well-formed" pair $\langle \alpha, \gamma \rangle$ of an assignment α and a commitment γ, *i.e.* a pair s.t. each multi-strategy variable associated with an agent in the commitment is instantiated: $\gamma(\mathit{Ag}) \subseteq \mathrm{dom}(\alpha)$.

Notation. We write $\alpha_\emptyset$ for the empty assignment (s.t. $\mathrm{dom}(\alpha_\emptyset) = \emptyset$), $\gamma_\emptyset$ for the empty commitment (s.t. $\mathrm{dom}(\gamma_\emptyset) = \emptyset$), and $\chi_\emptyset$ for the empty context (s.t. $\chi_\emptyset = \langle \alpha_\emptyset, \gamma_\emptyset \rangle$).

The notion of multi-strategy translation is now extended to assignments, which will be useful to define the notion of outcome in the next section. A translated assignment is one where every multi-strategy instance is itself translated.

Definition 12 (Assignment Translation, Context Translation). Let $\mathcal{G} = \langle \mathit{Ag}, \mathit{St}, \mathit{At}, \mathit{prop}, \mathit{Act}, \mathit{tr}, s_o \rangle$ be a CGS, $\chi = \langle \alpha, \gamma \rangle$ be a context and θ be a track in $\mathcal{G}$. We call *translation of α along θ* the assignment α^θ s.t. $\mathrm{dom}(\alpha^\theta) = \mathrm{dom}(\alpha)$ and for any $x \in \mathrm{dom}(\alpha^\theta)$, $\alpha^\theta(x) = (\alpha(x))^\theta$. The translation for the context χ is simply defined as $\langle \alpha, \gamma \rangle^\theta = \langle \alpha^\theta, \gamma \rangle$.

During the semantic evaluation of a formula, the context must be transformed as we encounter a multi-strategy quantifier, a binding operator or an unbinding operator. For a quantifier $\langle\!\langle x \rangle\!\rangle$, we simply update the current assignment for x. For a binding operator $(A \rhd x)$, the commitment must be updated with a pair $\langle a, x \rangle$ for every agent a in A. Finally, for an unbinding operator, all pairs $\langle a, x \rangle$ must be removed from the commitment, for any agent a in A.

Definition 13 (Context Transformations). Let $\chi = \langle \alpha, \gamma \rangle$ be a context, $A \subseteq \mathit{Ag}$ be a coalition, x be a multi-strategy variable and ς be a multi-strategy. We define the transformations $[x \mapsto \varsigma]$ on assignments and $[A \oplus x]$ and $[A \ominus x]$ on commitments as follows:

$$\alpha[x \mapsto \varsigma](y) = \begin{cases} \varsigma & \text{if } x = y \\ \alpha(y) & \text{otherwise} \end{cases}$$

$$\gamma[A \oplus x] = \gamma \cup \{\langle a, x \rangle \mid a \in A\}$$

$$\gamma[A \ominus x] = \gamma \setminus \{\langle a, x \rangle \mid a \in A\}$$

This notation is extended to contexts: $\chi[x \mapsto \varsigma] = \langle \alpha[x \mapsto \varsigma], \gamma \rangle$, $\chi[A \oplus x] = \langle \alpha, \gamma[A \oplus x] \rangle$ and $\chi[A \ominus x] = \langle \alpha, \gamma[A \ominus x] \rangle$.

Remark 2. Note that applying any of these three transformations to a context gives a context as result.

2.3.2. Outcomes

In SL [7], a number of definitions ("s-total" strategies and assignments, "complete" assignments...) is introduced to help characterising a play, that is the unique outcome of a game. Here, the situation is a bit different owing to non-determinism of our strategies and, instead of a unique play, we will end up with a set of *outcomes*. More precisely, a context induces an *outcome* function that maps every track θ to a set of executions that can happen if agents, starting from θ, play according to the multi-strategies stored in χ.

To define the outcome function, we must first define the set of possible immediate *successors* of a given track in a context $\chi = \langle \alpha, \gamma \rangle$.

Definition 14 (Successor Function). Let $\mathcal{G} = \langle Ag, St, At, prop, Act, tr, s_o \rangle$ be a CGS and $\chi = \langle \alpha, \gamma \rangle$ be a context. Then the *successor* function $succ_\chi : Track \to \mathcal{P}(St)$ induced by χ is defined, for any track θ, by:

$$succ_\chi(\theta) = \{tr(\text{last}(\theta), \delta) \mid \forall \langle a, x \rangle \in \gamma \cdot \delta(a) \in \alpha(x)(\theta)\}$$

Remark 3 (Finite Executions). Note that, in USL, as the successor function may yield the empty set, some executions may be *finite*. We use a common interpretation of temporal operators over finite executions, already defined in [11]. The semantics of the temporal operator X is less regular than in the usual framework with infinite executions: in particular, it does not commute with $\neg$. Nevertheless, it enables the framework to encompass cases where an agent is committed to contradictory multi-strategies, that is multi-strategies which indicate disjoint sets of actions after a given track. Detecting such cases brings significant additional expressive power, especially through the expression of equality between multi-strategies. The details are given in Sect. 3.2.

We finally come to the definition of *outcomes* of a context and a track, which is the set of *finite* and infinite executions (*i.e.* tracks and paths) that are possible when the history of the game is given by the track.

Definition 15 (Outcomes). Let $\mathcal{G} = \langle Ag, St, At, prop, Act, tr, s_o \rangle$ be a CGS, θ be a track and $\chi = \langle \alpha, \gamma \rangle$ be a context. We define the set of *outcomes* of χ and θ in $\mathcal{G}$ as the set $out(\chi, \theta)$ of tracks and paths λ, in $\mathcal{G}$, s.t. $\lambda_0 = \text{last}(\theta)$ and:

- if λ is a path: for any $i \in \mathbb{N}$, $\lambda_{i+1} \in succ_{\chi^{\lambda_{\leqslant i}}}(\lambda_{\leqslant i})$
- if λ is a track $\lambda_{\leqslant n}$:
  - for any $i < n$, $\lambda_{i+1} \in succ_{\chi^{\lambda_{\leqslant i}}}(\lambda_{\leqslant i})$
  - and $succ_{\chi^\lambda}(\lambda) = \emptyset$.

2.3.3. Satisfaction

Definition 16 (Satisfaction). Let $\mathcal{G} = \langle Ag, St, At, prop, Act, tr, s_o \rangle$ be a CGS and χ be a context. Then the *satisfaction relation* $\models$ is defined by induction on formulas, for every state $s \in St$ and executions λ in $\mathcal{G}$, as follows:

- State formulas
  - $\mathcal{G}, \chi, s \models p$ iff $p \in prop(s)$, with $p \in At$
  - $\mathcal{G}, \chi, s \models \neg\psi$ iff $\mathcal{G}, \chi, s \not\models \psi$
  - $\mathcal{G}, \chi, s \models \psi_1 \wedge \psi_2$ iff $\mathcal{G}, \chi, s \models \psi_1$ and $\mathcal{G}, \chi, s \models \psi_2$
  - $\mathcal{G}, \chi, s \models \langle\!\langle x \rangle\!\rangle \psi$ iff there is a multi-strategy $\varsigma \in MStrat_\mathcal{G}$ s.t. $\mathcal{G}, \chi[x \mapsto \varsigma], s \models \psi$
  - $\mathcal{G}, \chi, s \models (A \rhd x)\varphi$ iff for any $\lambda \in out(\chi[A \oplus x], s), \mathcal{G}, \chi[A \oplus x], \lambda \models \varphi$
  - $\mathcal{G}, \chi, s \models (A \not\rhd x)\varphi$ iff for any $\lambda \in out(\chi[A \ominus x], s), \mathcal{G}, \chi[A \ominus x], \lambda \models \varphi$
- Path formulas
  - $\mathcal{G}, \chi, \lambda \models \psi$ iff $\mathcal{G}, \chi, \lambda_0 \models \psi$, if ψ is a state formula
  - $\mathcal{G}, \chi, \lambda \models \neg\varphi$ iff $\mathcal{G}, \chi, \lambda \not\models \varphi$
  - $\mathcal{G}, \chi, \lambda \models \varphi_1 \wedge \varphi_2$ iff $\mathcal{G}, \chi, \lambda \models \varphi_1$ and $\mathcal{G}, \chi, \lambda \models \varphi_2$

– $\mathcal{G}, \chi, \lambda \models \mathsf{X}\varphi$ iff $|\lambda| > 1$ and $\mathcal{G}, \chi^{\lambda_0\lambda_1}, \lambda_{\geqslant 1} \models \varphi$
– $\mathcal{G}, \chi, \lambda \models \varphi_1 \mathsf{U} \varphi_2$ iff there is a number $i \in \mathbb{N}$ s.t. $|\lambda| > i$, s.t. $\mathcal{G}, \chi^{\lambda_{\leqslant i}}, \lambda_{\geqslant i} \models \varphi_2$ and s.t. for any $0 \leqslant j < i$, $\mathcal{G}, \chi^{\lambda_{\leqslant j}}, \lambda_{\geqslant j} \models \varphi_1$.

Given the empty context $\chi_\emptyset$ and a sentence ψ, we write $\mathcal{G}, s \models \psi$ iff $\mathcal{G}, \chi_\emptyset, s \models \psi$, and $\mathcal{G} \models \psi$ iff $\mathcal{G}, s_0 \models \psi$.

3. Noticeable properties

In this section, we develop on interesting properties we can express in USL. We present properties that are due to the multi-strategy refinement and revocation in Sect. 3.1. Then in Sect. 3.2, we show that USL enables to express equality between multi-strategies and we investigate how this can be useful to express notions such as uniqueness, determinism and dependences between multi-strategies.

3.1. Refinement and revocation of multi-strategies

Let us illustrate the mechanisms of multi-strategies refinement and revocation. The ability to refine multi-strategies in USL allows to reason about situations where an agent

1. has a choice between different properties she can alternatively ensure,
2. and is able to *remain* in capacity to have all these choices available.

The concept of *sustainable control* is a particular case where the agent's choice is about satisfying or falsifying a given property, at any time: when the agent ensures the satisfaction (or the falsification) of the property in a given state, both possibilities remain available in the next state. Let us first informally illustrate this notion with a simple example of a chat server. Then, we will provide a formal definition.

Example 1 (Chat server: sustainable control)**.** *As an illustration (see Fig. 1), suppose one client sends connection requests to the chat server. From the initial state (s_0) the server can always decide whether to* grant *or* deny access *to the client. In addition, the server can* ban *a client and refuse connection once and for all.*

So it can set the atomic proposition access *to true or to false at any moment and as many times as it wants. While the server does not ban the client, both possibilities (set* access *to true or to false) remain available. So intuitively, there is a multi-strategy for the server (namely "not banning the client") such that it always controls the proposition* access*. Thus we say that the server has* sustainable control *over* access*.*

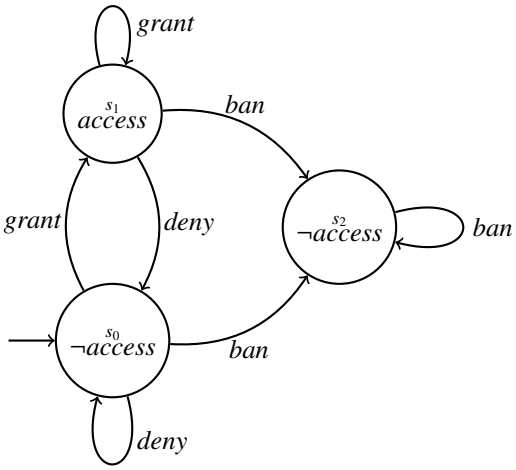

Figure 1: Model $\mathcal{M}_1$ for the server

We now give a formal definition of sustainable control from a semantic perspective. We represent the property to control by a set of states S, and consider that an agent has sustainable control over S from the current state if she has an action that forces the system to go to S and another action that forces the system not to go to S; and after playing either one, she still has sustainable control over S.

Definition 17 (Sustainable control). Given a CGS $\mathcal{G} = \langle Ag, St, At, prop, Act, tr, s_o\rangle$, an agent $a \in Ag$, and a set $S \subseteq St$ of states, we note $\mathsf{SCont}(a, S, s)$ to express that a has *sustainable control* over S from state s, which is defined coinductively as follows: $\mathsf{SCont}(a, S, s)$ iff

$$\begin{array}{lll} \exists ac_1, ac_2 \in Act \cdot \exists S_1, S_2 \subseteq St & \text{s.t.} & \overline{tr}(s, \{a \mapsto ac_1\}) = S_1 \text{ and } \overline{tr}(s, \{a \mapsto ac_2\}) = S_2 \\ & \text{and} & S_1 \subseteq S \text{ and } S_2 \subseteq St \setminus S \\ & \text{and} & \forall s' \in S_1 \cup S_2 \cdot \mathsf{SCont}(a, S, s') \end{array}$$

where $\overline{tr}$ returns the set of all possible successors from a state, given the action performed by one agent[1]:

$$\overline{tr}(s, \{a \mapsto ac\}) = \{s' \in St \mid \exists \delta \in Dec \cdot \delta(a) = ac \text{ and } tr(s, \delta) = s'\}$$

As a matter of fact, sustainable control can be characterised syntactically by a USL formula. We represent the property to control by a (propositional) formula ψ (for the set S of states). Besides, we need three multi-strategy variables to express sustainable control: a multi-strategy variable y to ensure the satisfaction of ψ, a multi-strategy variable z to ensure the falsification of ψ, and a general multi-strategy x, which can compose both with y and z, and ensure that the agent can stay forever in condition to ensure the satisfaction and the falsification of ψ.

Proposition 1. *Given a CGS $\mathcal{G} = \langle Ag, St, At, prop, Act, tr, s_o\rangle$, an agent $a \in Ag$, a set $S \subseteq St$ of states, a state $s \in St$, and a formula ψ such that, for any state $s' \in St$, $s' \in S$ iff $\mathcal{G}, s' \models \psi$, then*

$$\mathsf{SCont}(a, S, s) \quad \textit{iff} \quad \mathcal{G}, s \models \langle\!\langle x\rangle\!\rangle(a \rhd x)\Box(\langle\!\langle y\rangle\!\rangle(a \rhd y)\mathsf{X}\psi \wedge \langle\!\langle z\rangle\!\rangle(a \rhd z)\mathsf{X}\neg\psi)$$

Proof. See Appendix A. □

Let us come back to our chat server example. As we informally explained above, from the initial state s_0 (as well as from state s_1) the server has sustainable control over *access*, *i.e.*, it can set *access* to true and to false at any time. Indeed, one can easily see that the following formula expressing the sustainable control of a over *access* holds in s_0:

$$\langle\!\langle x\rangle\!\rangle(a \rhd x)\Box(\langle\!\langle y\rangle\!\rangle(a \rhd y)\mathsf{X}\mathit{access} \wedge \langle\!\langle z\rangle\!\rangle(a \rhd z)\mathsf{X}\neg\mathit{access})$$

Finally, the presence of the explicit unbinder makes it possible to reason about the revocation of multi-strategies. For instance, we can express that the server, when applying the multi-strategy by which it controls *access*, can decide at any time to revoke this multi-strategy and to ban the client, *i.e.*, to ensure that, from an arbitrary moment chosen by the server, *access* becomes false forever:

$$\langle\!\langle x\rangle\!\rangle(a \rhd x)\Box(\langle\!\langle y\rangle\!\rangle(a \rhd y)\mathsf{X}\mathit{access} \wedge \langle\!\langle z\rangle\!\rangle(a \rhd z)\mathsf{X}\neg\mathit{access} \wedge \langle\!\langle z\rangle\!\rangle(a \not\rhd x)(a \rhd z)\Box\neg\mathit{access})$$

3.2. Relations between multi-strategies

In USL, when an agent is committed to contradictory multi-strategies, in the sense that, after a given track, these specify *disjoint sets of actions,* then the "execution" stops. More precisely, the context that stores the contradictory commitments induces an empty set of outcomes. This aspect of the semantics of USL allows to test, for instance, whether two multi-strategies x and y specify disjoint sets of actions using the formula $(a \rhd x)(a \rhd y)\mathsf{X}\top$. We can then express relations of inclusion and equality between two multi-strategies (Sect. 3.2.1). It is also possible characterise deterministic multi-strategies and define quantifiers over them (Sect. 3.2.2), which is useful to express a simple notion of dependence between two multi-strategies (Sect. 3.2.3). We also show in Sect. 3.2.4 that we can actually characterise a finer-grained relation of *strong dependence* between multi-strategies.

[1] In the case of a single agent ($Ag = \{a\}$), as in the chat server, $\overline{tr}$ always returns a singleton and can be identified with tr.

3.2.1. Inclusion, equality and uniqueness

First, we consider the inclusion for one transition, which expresses that given the current state and the current context, the set of actions specified by the multi-strategy x is included in the one specified by y. This is given by Formula (3):

$$x \subseteq_{\mathsf{X}} y \triangleq [\![z]\!](a \rhd z)((a \rhd x)\mathsf{X}\top \rightarrow (a \rhd y)\mathsf{X}\top) \tag{3}$$

Indeed, Formula (3) asserts that any set of actions in intersection with the one specified by x is also in intersection with the one specified by y. From this definition, we define easily the equality $=_{\mathsf{X}}$ for one transition. We also define the global inclusion $\subseteq$ (and equality $=$) between multi-strategies.

$$\begin{array}{lcl} x =_{\mathsf{X}} y & \triangleq & x \subseteq_{\mathsf{X}} y \wedge y \subseteq_{\mathsf{X}} x \\ x \subseteq y & \triangleq & [\![z]\!](a \rhd z)\Box((a \rhd x)\mathsf{X}\top \rightarrow (a \rhd y)\mathsf{X}\top) \\ x = y & \triangleq & x \subseteq y \wedge y \subseteq x \end{array}$$

Formula $x \subseteq y$ asserts that the set of actions yielded by x is included in the one yielded by y, after any track θ starting from the current state. Similarly, the equality $x = y$ characterises multi-strategies that yield the same sets of actions after any possible track from the current state.

The existence of an equality predicate enables to count the number of possible multi-strategies for a given goal. In particular, we can express the existence of a unique multi-strategy satisfying a goal.

Definition 18 (Uniqueness quantification). The *unique existential quantifier* ($\langle\!\langle !x \rangle\!\rangle$) is defined as follows: for any formula ψ with free variable x:

$$\langle\!\langle !x \rangle\!\rangle \psi \triangleq \langle\!\langle x \rangle\!\rangle(\psi \wedge [\![y]\!](\psi[x := y] \rightarrow x = y))$$

where $\psi[x := y]$ is obtained from ψ by substituting any free occurrence of variable x by y.

Note that if there is a unique multi-strategy such that an agent ensures a goal, then the satisfaction of the goal fully determines the multi-strategy of the agent pursuing it.

3.2.2. Deterministic strategies

Inclusion and equality also enable to express that a multi-strategy is deterministic, *i.e.*, it specifies a single action after any track. In order to express in USL that a variable x represents a multi-strategy which is deterministic *from the current state*, we specify that the only multi-strategy included in x is x itself. It is expressed by Formula (4), with x as free variable:

$$Det(x) \triangleq [\![y]\!](y \subseteq x \rightarrow y = x) \tag{4}$$

A natural way to reason about deterministic strategies consists in using dedicated quantifiers:

Definition 19 (Quantifiers over deterministic strategies). The existential ($\langle\!\langle x \rangle\!\rangle_d$) and universal ($[\![x]\!]_d$) quantifiers over deterministic strategies are defined as follows:

Existential For any formula ψ, $\langle\!\langle x \rangle\!\rangle_d \psi \triangleq \langle\!\langle x \rangle\!\rangle(Det(x) \wedge \psi)$;

Universal For any formula ψ, $[\![x]\!]_d \psi \triangleq [\![x]\!](Det(x) \rightarrow \psi)$.

3.2.3. Alternation of quantifiers and dependence

Of course, these deterministic quantifiers are useful to compare USL to deterministic multi-agent logics such as SL. For instance, in Sect. 4.2, we use them to define an embedding of SL into USL. But they are also of interest in order to express properties that involve an alternation of quantifiers. Indeed, in the case of an alternation of the kind $[\![x]\!]\langle\!\langle y \rangle\!\rangle \psi$, one needs to use the quantifier $[\![x]\!]_d$ instead of $[\![x]\!]$ to enable multi-strategy y to depend on x. Let us develop on this aspect in the following.

Example 2 (Alternation of quantifiers)**.** *Let us consider the CGS $\mathcal{G}_{[\![]\!]\langle\!\langle\rangle\!\rangle}$ illustrated in Fig. 2. There are two agents a and b, playing heads or tails. They both win the game if they get the same side of their respective coin and they both lose otherwise. Both players can play either action heads or action tails. The initial state is s_0. State s_1 is reached if the players win and state s_2 is reached if they lose. The only atomic proposition in the language, p, is true only in state s_1 and testifies that the game is won. We want to express that whatever a plays, b can ensure the victory. From a game-theoretic point of view, where both players are able to control their play and decide between heads and tails, this is obviously true, since b can play the same action as a.*

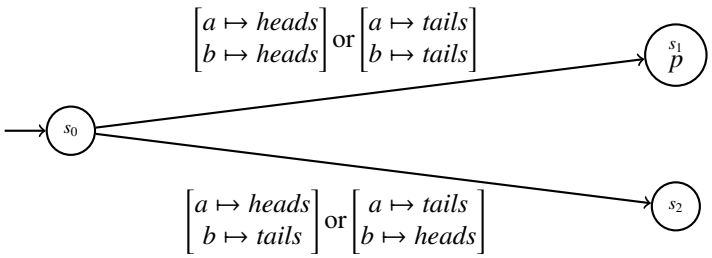

Figure 2: Model $\mathcal{G}_{[\![]\!]\langle\!\langle\rangle\!\rangle}$

First, remark that Formula (5) is inadequate to express the property:

$$[\![x]\!]\langle\!\langle y\rangle\!\rangle(a \rhd x)(b \rhd y)\mathsf{X}p \tag{5}$$

Indeed, this formula specifies that for any multi-strategy ς_x, there is a multi-strategy ς_y s.t., at s_0, if a plays ς_x and b plays ς_y then p will necessarily be true in the next state (which means that the execution goes to s_1). In particular, if ς_x is the multi-strategy that yields the whole set of available actions after any track, then there is obviously no ς_y such that p will be ensured. So Formula (5) is not satisfied in state s_0.

Actually, Formula (5) is true exactly in the same models as Formula (6), obtained by commuting both quantifiers (we leave the proof to the reader):

$$\langle\!\langle x\rangle\!\rangle[\![y]\!](a \rhd x)(b \rhd y)\mathsf{X}p \tag{6}$$

What is at stake in this example is a dependence relation between the actions yielded by the multi-strategy instantiating x and those yielded by the multi-strategy instantiating y. Questions linked to dependence relations between strategies of actors were raised by Hintikka [12] and have given rise to numerous works. Independence Friendly Logic (IF [13]) focuses on the independence relation between strategies: within a set of assignments, a strategy y is *independent* from another strategy x iff changes on x do not affect the value of y. In such a case, y has a constant value over x (whereas it may have functional dependence relations with other variables).

The semantic framework of Dependence Logic (DL [14]), on the other hand, is based on the converse notion of dependence: within a set of assignments, a strategy y is *dependent* on another strategy x iff the value of x determines the value of y. In such a case there is a function from the possible values for x giving the value of y. Following this view, universal non-deterministic quantification is not adequate to express the dependence relation between actions that is due to the alternation of quantifiers. One must rather use the deterministic quantifier, as in Formula (7), which is true in state s_0 of our example:

$$[\![x]\!]_d\langle\!\langle y\rangle\!\rangle(a \rhd x)(b \rhd y)\mathsf{X}p \tag{7}$$

Actually, Formula (7) is true in every model where the actions yielded by y depend on those yielded by x. But it is also true in every models where y is completely independent of x, that is, where there is a multi-strategy y such that for every strategy x, $(a \rhd x)(b \rhd y)\mathsf{X}p$ is true. In such a case, y is both independent from x, with the IF perspective, and dependent on x, with the DF perspective. These two notions are not contradictory. Indeed, Formula (7) is a semantic consequence of $\langle\!\langle y\rangle\!\rangle[\![x]\!]_d(a \rhd x)(b \rhd y)\mathsf{X}p$. We say that Formula (7) expresses a *potential dependence*.

Then, using both deterministic and non-deterministic quantifiers enables us to characterise *actual dependence*, that is dependence without independence:

$$([\![x]\!]_d\langle\!\langle y\rangle\!\rangle(a \rhd x)(b \rhd y)\mathsf{X}p) \wedge \neg(\langle\!\langle y\rangle\!\rangle[\![x]\!]_d(a \rhd x)(b \rhd y)\mathsf{X}p) \tag{8}$$

Indeed, this formula asserts that the multi-strategy y must not be the same for every x, to satisfy $(a \rhd x)(b \rhd y)\mathsf{X}p$.

3.2.4. Strong dependence

Intuitively, expressing dependence aims at answering the following question: if x varies, does y vary? The actual dependence expressed in Formula (8) asserts that the whole set of possible variations of x induces some changes on y. However it would be interesting to give finer means to characterise the variations of x due to those of y. Thanks to equality, we can propose a first step in that direction: we say that y *strongly depends* on x for the satisfaction of φ if any change of x induces a change of y for the satisfaction of φ. Let us illustrate this aspect with the following example.

Example 3 (Magic trick)**.** *Consider a game with two players, the magician and the spectator. The spectator picks up a card from a pack of* 52 *pairwise different cards, then replaces it in the pack. Then the magician picks up a card. The game is iterated infinitely often. We distinguish between two different levels:*

Beginner level *The magician wins iff he never picks up the same card as the spectator.*

Expert level *The magician wins iff he always picks up the same card as the spectator.*

We wish to express the fact that the expert level is harder than the beginner level for the magician.

The game is represented by model $\mathcal{G}_M$ in Fig. 3. There are three states s_0, s_1 and s_2. The initial state is s_0 and there is one action per card in the game (for the sake of readability, Fig. 3 does not show actions but only whether they correspond to picking the same card or not). The transition function specifies that from any state the execution goes to:

- s_1 if both players play the same card,
- s_2 if they play different card.

Proposition *same* is true exactly at s_1. It states that both players picked up the same card during the previous turn.

For both levels, the magician has a winning multi-strategy, which depends on x. So the following formulas, similar to Formula (8), are true in our model.

$$\mathcal{G}_M \models ([\![x]\!]_d\langle\!\langle y\rangle\!\rangle(\mathit{spectator} \rhd x)(\mathit{magician} \rhd y)\mathsf{X}\Box \mathit{same}) \wedge \neg(\langle\!\langle y\rangle\!\rangle[\![x]\!]_d(\mathit{spectator} \rhd x)(\mathit{magician} \rhd y)\mathsf{X}\Box \mathit{same})$$

and

$$\mathcal{G}_M \models ([\![x]\!]_d\langle\!\langle y\rangle\!\rangle(\mathit{spectator} \rhd x)(\mathit{magician} \rhd y)\mathsf{X}\Box \neg \mathit{same}) \wedge \neg(\langle\!\langle y\rangle\!\rangle[\![x]\!]_d(\mathit{spectator} \rhd x)(\mathit{magician} \rhd y)\mathsf{X}\Box \neg \mathit{same})$$

The fact that the expert level is harder than the beginner level comes from the fact that, for the former, *y strongly depends* on *x*: any change in the strategy played by the spectator induces a change in the magician's multi-strategy. This is not the case for the beginner level, since for any action from the spectator, the magician has 51 available favourable actions. Intuitively, to win in the expert level, the magician needs complete information about the spectator's play; while in the beginner level, partial information only may be sufficient. Formula (9) expresses that the magician's multi-strategy strongly depends on the spectator's strategy in order to win the expert level game:

$$\begin{aligned}[\![x]\!]_d\langle\!\langle y\rangle\!\rangle(\mathit{spectator} \rhd x)(\mathit{magician} \rhd y)\mathsf{X}\Box \mathit{same} \\ \wedge \Big([\![x]\!]_d[\![y]\!][\![x']\!]_d[\![y']\!]\Big(((\mathit{spectator} \rhd x)(\mathit{magician} \rhd y)\mathsf{X}\Box \mathit{same}) \\ \wedge ((\mathit{spectator} \rhd x')(\mathit{magician} \rhd y')\mathsf{X}\Box \mathit{same}) \wedge x \neq x'\Big) \rightarrow y \neq y'\Big)\end{aligned} \tag{9}$$

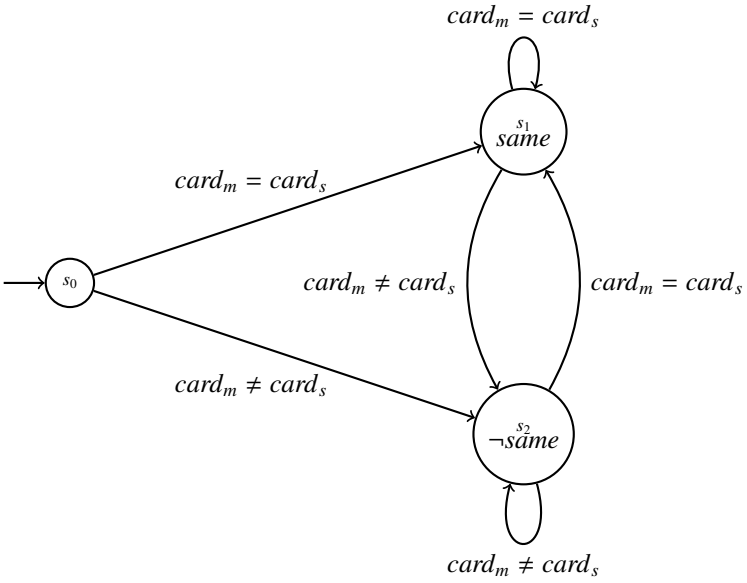

Figure 3: Illustration of the magic trick: $\mathcal{G}_M$

The first part of the formula states that for any strategy played by the spectator, the magician has a winning multi-strategy. The rest of the formula states that if two pairs $\langle x, y\rangle$ and $\langle x', y'\rangle$ of a strategy and a multi-strategy (x and x' for the spectator and y and y' for the magician), such that $x \neq x'$, enable to ensure that the magician wins the expert level game, then $y \neq y'$. Said otherwise, if the strategy of the spectator changes, then the magician's multi-strategy must change in order to win the expert level game.

Formula (9) is true in $\mathcal{G}_M$. But the formula expressing strong dependence of the magician's multi-strategy on the spectator's strategy to win the *beginner* level (*i.e.* the one obtained from Formula (9) by replacing $p_=$ with $\neg p_=$) is obviously false in $\mathcal{G}_M$. This shows that the beginner level is easier that the expert level.

4. Comparison with SL

In this section, we show that USL is strictly more expressive than SL. We first recall technical facts regarding the notion of expressive power. Then we show that USL is at least as expressive as SL. Finally, we show that the converse does not hold.

4.1. Comparing Logics

The comparison of expressive powers between USL and SL uses two classical partial ordering relations between logics. Let us first recall their definitions.

Definition 20 (Expressiveness). Let $\mathcal{L}$ and $\mathcal{L}'$ be two logics interpreted over CGS's. Let φ be an $\mathcal{L}$ sentence and φ' be an $\mathcal{L}'$ sentence. We say that φ and φ' are *equivalent* iff for any CGS $\mathcal{G}$: $\mathcal{G} \models_{\mathcal{L}} \varphi$ iff $\mathcal{G} \models_{\mathcal{L}'} \varphi'$.

We say that $\mathcal{L}$ is *at least as expressive as* $\mathcal{L}'$, written $\mathcal{L}' \preccurlyeq_{exp} \mathcal{L}$ iff for any sentence in $\mathcal{L}'$, there is an equivalent sentence in $\mathcal{L}$. We also say that $\mathcal{L}$ is *strictly more expressive* than $\mathcal{L}'$, written $\mathcal{L}' \prec_{exp} \mathcal{L}$, iff $\mathcal{L}' \preccurlyeq_{exp} \mathcal{L}$ and $\mathcal{L} \not\preccurlyeq_{exp} \mathcal{L}'$.

Definition 21 (Distinguishing power). Let $\mathcal{L}$ and $\mathcal{L}'$ be two different logics interpreted over CGS's. We say that $\mathcal{L}$ *distinguishes* between two CGS's $\mathcal{G}_1$ and $\mathcal{G}_2$ iff there is a sentence φ in $\mathcal{L}$ such that $\mathcal{G}_1 \models_{\mathcal{L}} \varphi$ and $\mathcal{G}_2 \not\models_{\mathcal{L}} \varphi$.

We say that $\mathcal{L}$ is *at least as distinguishing as* $\mathcal{L}'$, written $\mathcal{L}' \preccurlyeq_{dis} \mathcal{L}$, iff for any pair of CGS's $\mathcal{G}_1$ and $\mathcal{G}_2$, if $\mathcal{L}'$ distinguishes between $\mathcal{G}_1$ and $\mathcal{G}_2$, then $\mathcal{L}$ also distinguishes between $\mathcal{G}_1$ and $\mathcal{G}_2$

Lemma 1. *Let $\mathcal{L}$ and $\mathcal{L}'$ be two logics interpreted over CGS's. If $\mathcal{L}' \preccurlyeq_{exp} \mathcal{L}$ then $\mathcal{L}' \preccurlyeq_{dis} \mathcal{L}$.*

4.2. USL is at least as expressive as SL

We show in Sect. 4.2.2 that USL is at least as expressive as SL by defining an *embedding* of the latter into the former. We first make a few considerations on SL formulas in Sect. 4.2.1 then come to the embedding itself in Sect 4.2.2.

4.2.1. Technical Considerations on SL Formulas

First, we make a distinction between path and state formulas for SL as the embedding will depend on this. Notice that we do not change the syntax of SL, not even its grammar: we only distinguish between two classes of formulas. We recall in Appendix B the facts related to SL that we rely on in the following.

Definition 22 (State and path formulas of SL). Let Ag be a set of agents, At be a set of propositions and Var be a set of variables. The set of SL formulas can be divided into two subsets described by the following grammar:

- state formulas: $\psi ::= p \mid \neg\psi \mid \psi \wedge \psi \mid \psi \vee \psi \mid \langle\!\langle x \rangle\!\rangle \varphi \mid [\![x]\!] \varphi \mid (a, x)\varphi$
- path formulas: $\varphi ::= \psi \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \mathsf{X}\varphi \mid \varphi \,\mathsf{U}\, \varphi \mid \varphi \,\mathsf{R}\, \varphi$

where $p \in At, a$ is an agent and $x \in Var$.

Second, to ensure that our embedding yields well-formed USL formulas, and not just pseudo-formulas, we transfer our well-formedness criterion for USL formulas (cf. Def. 7) to SL ones.

Definition 23 (USL-compliant SL formula). An SL formula φ is *USL-compliant* iff for any subformula $\langle\!\langle x \rangle\!\rangle \varphi'$ or $[\![x]\!]\varphi'$ of φ and for any subformula $\langle\!\langle y \rangle\!\rangle \varphi''$ or $[\![y]\!]\varphi''$ of φ', x and y are distinct variables.

Notice that in the remainder of Sect. 4, *we only consider USL-compliant SL formulas*, which is without loss of generality as one can easily check that any SL formula can be made USL-compliant by an adequate renaming of variables.

4.2.2. Embedding SL into USL

Here, we establish that USL is at least as expressive as SL by exhibiting an embedding of SL into USL that applies to SL formulas and to SL contexts:

- The embedding is applied to USL-compliant SL formulas (cf. Def. 23).
- For any SL formula φ, for any subformula $\varphi' = (a, x)\varphi''$ of φ, the operator (a, x) is replaced by :
  - $(a \rhd x)$ if there is no other binding (a, y) above φ' in φ
  - $(a \not\rhd y)(a \rhd x)$ otherwise.

  Thus, for any agent a, the translation of subformulas depends on the unique (if any) strategy variable to which a is bound in a strategy context used for evaluating φ'. We address this by parameterising the embedding with a dictionary f keeping track of bindings of strategy variables to agents, *i.e.* $f : Ag \rightharpoonup X$. Then, given an agent a and a variable x, we write $f[a \mapsto x]$ for the update of f with the binding of x to a. (Formally, except for typing, this works exactly as for assignments and transformations thereof, see Def. 11 and 13.)
- SL quantifiers are simulated by the USL quantifiers $\langle\!\langle x \rangle\!\rangle_d$ and $[\![x]\!]_d$ over *deterministic* strategies (cf. Def. 19).
- In SL, quantifiers apply to path formulas while, in USL, for any pseudo-formula $\langle\!\langle x \rangle\!\rangle \psi$ or $[\![x]\!]\psi$, ψ is a state formula. To address constraint in our embedding, we give a different treatment of SL formulas $\langle\!\langle x \rangle\!\rangle \psi$ or $[\![x]\!]\psi$, depending on whether ψ is a state formula or not (cf. Def. 22).

  If ψ is *not* a state formula, we insert $[\![x_\emptyset]\!](\emptyset \rhd x_\emptyset)$ in the embedding (here, $\emptyset$ is the empty coalition and $x_\emptyset$ is fresh in the source formula). This insertion enables to transform any formula that is not a state formula into a state formula, by universally quantifying over the executions that are made possible within the current context. Indeed, one can check that the semantics of USL is such that for any $\mathcal{G}, \chi, s$ and for any formula ψ which is not a state formula, $\mathcal{G}, \chi, s \models_{\text{USL}} [\![x_\emptyset]\!](\emptyset \rhd x_\emptyset)\psi$ iff for any $\lambda \in out(\chi, s), \mathcal{G}, \chi, \lambda \models_{\text{USL}} \psi$.

Definition 24 (Embedding of SL into USL). Let φ be an (USL-compliant) SL formula. Then the *embedding* $\mathsf{usl}(\varphi)$ of φ in USL is the formula $\mathsf{usl}_{f_\emptyset}(\varphi)$, where $f_\emptyset$ is the function $f_\emptyset : Ag \nrightarrow X$ with empty domain, and where $\mathsf{usl}_f(\varphi)$ is defined by structural recursion over φ as follows:

- $\mathsf{usl}_f(p) = p$
- $\mathsf{usl}_f(\neg\psi) = \neg\, \mathsf{usl}_f(\psi)$
- $\mathsf{usl}_f(\psi_1 \wedge \psi_2) = \mathsf{usl}_f(\psi_1) \wedge \mathsf{usl}_f(\psi_2)$
- $\mathsf{usl}_f(\psi_1 \vee \psi_2) = \mathsf{usl}_f(\psi_1) \vee \mathsf{usl}_f(\psi_2))$
- $\mathsf{usl}_f(\langle\!\langle x \rangle\!\rangle \psi) = \begin{cases} \langle\!\langle x \rangle\!\rangle_d([\![x_\emptyset]\!](\emptyset \rhd x_\emptyset)\, \mathsf{usl}_f(\psi)) \text{ if } \psi \text{ is not a state formula (where } x_\emptyset \text{ is fresh in } \varphi) \\ \langle\!\langle x \rangle\!\rangle_d\, \mathsf{usl}_f(\psi) \text{ if } \psi \text{ is a state formula} \end{cases}$
- $\mathsf{usl}_f([\![x]\!]\psi) = \begin{cases} [\![x]\!]_d([\![x_\emptyset]\!](\emptyset \rhd x_\emptyset)\, \mathsf{usl}_f(\psi)) \text{ if } \psi \text{ is not a state formula (where } x_\emptyset \text{ is fresh in } \varphi) \\ [\![x]\!]_d\, \mathsf{usl}_f(\psi) \text{ if } \psi \text{ is a state formula} \end{cases}$
- $\mathsf{usl}_f((a, x)\psi) = \begin{cases} (a \not\rhd f(a))(a \rhd x)\, \mathsf{usl}_{f[a \mapsto x]}(\psi) \text{ if } a \in \mathrm{dom}(f) \\ (a \rhd x)\, \mathsf{usl}_{f[a \mapsto x]}(\psi) \text{ otherwise} \end{cases}$
- $\mathsf{usl}_f(\mathsf{X}\psi) = \mathsf{X}\, \mathsf{usl}_f(\psi)$
- $\mathsf{usl}_f(\psi_1\, \mathsf{U}\, \psi_2) = \mathsf{usl}_f(\psi_1)\, \mathsf{U}\, \mathsf{usl}_f(\psi_2)$
- $\mathsf{usl}_f(\psi_1\, \mathsf{R}\, \psi_2) = \mathsf{usl}_f(\psi_1)\, \mathsf{R}\, \mathsf{usl}_f(\psi_2)$

Before going any further, let us note that this embedding is such that for any SL formula φ, $|\mathsf{usl}(\varphi)| \in O(|\varphi|)$: in every item of Def. 24, the size of the subformula is at most increased by a constant.

We now consider the translation of SL contexts into USL ones. We first make a few remarks.

Remark 4.

- For any SL context $\overline{\chi}$ used in the evaluation under $\models_{\mathrm{SL}}$ of a USL-compliant sentence φ in a CGS, $\overline{\chi}$ is such that for any agent a in $\mathrm{dom}(\overline{\chi})$, there is a strategy variable x in $\overline{\chi}$ such that $\overline{\chi}(a) = \overline{\chi}(x)$.
- For any SL context $\overline{\chi}$ used in the evaluation under $\models_{\mathrm{SL}}$ of a sentence φ, if $\overline{\chi}$ is used for the evaluation, in a state s, of a subformula of φ that is not a state formula, then $\overline{\chi}$ is a complete and s-total context (cf. Appendix B, Def. 52).

Let us now comment on the definition of the transformation of SL contexts defined hereafter: any strategy context $\overline{\chi}$ for SL is mapped to a *set of contexts* for USL.

Indeed, a strategy context $\overline{\chi}$ for SL maps agents and variables to strategies directly, while a context χ for USL maps agents to variables, and variables to multi-strategies.

Therefore, we transform $\overline{\chi}$ into contexts for USL s.t. each variable x in the domain of $\overline{\chi}$ is mapped to $\overline{\chi}(x)$, and s.t. each agent a in the domain of $\overline{\chi}$ is mapped to a variable x s.t. $\overline{\chi}(x) = \overline{\chi}(a)$. This ensures that a is committed to $\overline{\chi}(a)$. Now, suppose there are different variables in the domain of $\overline{\chi}$ having the same image as a. Then there are different ways to transform $\overline{\chi}$ into a context for USL. This is why the transformation may yield several different contexts.

Definition 25 (Embedding of SL contexts). Let $\overline{\chi}$ be a strategy context for SL, then $\mathsf{usl}(\overline{\chi})$ is the set of contexts $\chi = \langle \alpha, \gamma \rangle$ for USL such that:

- $\mathrm{dom}(\alpha) = \mathrm{dom}(\overline{\chi}) \cap Var$ and for every $x \in \mathrm{dom}(\alpha)$, $\alpha(x) = \overline{\chi}(x)$
- $\mathrm{dom}(\gamma) = \mathrm{dom}(\overline{\chi}) \cap Ag$ and for every $a \in \mathrm{dom}(\gamma)$, $\gamma(a)$ yields a singleton $\{x\}$ and $x \in \mathrm{dom}(\alpha)$ and $\alpha(x) = \overline{\chi}(a)$.

Remark 5.

- For any SL context $\overline{\chi}$ used in the evaluation under $\models_{\text{SL}}$ of a USL-compliant sentence φ, for any $\chi = \langle \alpha, \gamma \rangle \in \mathsf{usl}(\overline{\chi})$, for any agent $a \in Ag$, there is at most one variable x s.t. $\langle a, x \rangle \in \gamma$. Then γ can be seen as a partial function from Ag to X. Let γ be such a commitment, a be an agent and x be a variable. In the following proof we use the transformation $\gamma[a \mapsto x]$ as introduced above for partial functions. We also use the transformations $\gamma[a \oplus x]$ and $\gamma[a \ominus x]$ introduced in Def.13 for commitments. Note that these three transformations are different from one another.

- For any CGS $\mathcal{G}$, for any strategy context $\overline{\chi}$, for any $\chi, \chi' \in \mathsf{usl}(\overline{\chi})$, and for any state s in $\mathcal{G}$, $out(\chi, s) = out(\chi', s)$. Furthermore, if $\overline{\chi}$ is complete and s-total, then $out(\chi, s)$ is the singleton made from the unique path λ determined by χ and s.

Lemma 2. *For any SL formula φ, for any CGS $\mathcal{G}$ and for any strategy context $\overline{\chi}$ such that $\overline{\chi}$ may be used in the evaluation of φ into $\mathcal{G}$, for any $\chi = \langle \alpha, \gamma \rangle \in \mathsf{usl}(\overline{\chi})$, and for any state s:*

- *if φ is a state formula then:*
$$\mathcal{G}, \overline{\chi}, s \models_{SL} \varphi \text{ iff } \mathcal{G}, \chi, s \models_{USL} \mathsf{usl}_\gamma(\varphi)$$

- *if φ is not a state formula then:*
$$\mathcal{G}, \overline{\chi}, s \models_{SL} \varphi \text{ iff the unique path } \Lambda^{\chi,s} \in out(\chi, s) \text{ is s.t. } \mathcal{G}, \chi, \Lambda^{\chi,s} \models_{USL} \mathsf{usl}_\gamma(\varphi)$$

Proof. By structural induction on φ. We present the cases for the negation, the existential quantifier, the binder and the X operator. Note that the strategies for SL are partial functions (see Def. 49) whereas multi-strategies for USL are functions. Consequently, the deterministic strategies defined in USL (see Sect. 3.2.2) slightly differ from SL strategies. In this proof, the terms *strategy* and *s-total strategy* refer to SL definitions, and the strategies in the sense of USL are called *total strategies*.

- Case $\varphi = \neg\psi$:
  - If ψ is a state formula, then $\mathcal{G}, \overline{\chi}, s \models_{\text{SL}} \neg\psi$ iff (by induction hypothesis) for any $\chi \in \mathsf{usl}(\overline{\chi})$) it is not true that $\mathcal{G}, \chi, s \models_{\text{USL}} \mathsf{usl}_\gamma(\psi)$, which is the definition of $\mathcal{G}, \chi, s \models_{\text{USL}} \neg\, \mathsf{usl}_\gamma(\psi)$ and holds iff $\mathcal{G}, \chi, s \models_{\text{USL}} \mathsf{usl}_\gamma(\neg\psi)$.
  - If ψ is not a state formula, then $\mathcal{G}, \overline{\chi}, s \models_{\text{SL}} \neg\psi$ iff $\mathcal{G}, \overline{\chi}, s \not\models_{\text{SL}} \psi$. By induction hypothesis, for any $\chi \in \mathsf{usl}(\overline{\chi})$, this is equivalent to: $\mathcal{G}, \chi, \Lambda^{\chi,s} \not\models_{\text{USL}} \mathsf{usl}_\gamma(\psi)$, which is the definition of $\mathcal{G}, \chi, \Lambda^{\chi,s} \models_{\text{USL}} \neg\, \mathsf{usl}_\gamma(\psi)$ and holds iff $\mathcal{G}, \chi, \Lambda^{\chi,s} \models_{\text{USL}} \mathsf{usl}_\gamma(\neg\psi)$.

- Case $\varphi = \langle\!\langle x \rangle\!\rangle \psi$:
  - If ψ is a state formula, then $\mathcal{G}, \overline{\chi}, s \models_{\text{SL}} \langle\!\langle x \rangle\!\rangle \psi$ iff there is an s-total strategy ς s.t. $\mathcal{G}, \overline{\chi}[x \mapsto \varsigma], s \models_{\text{SL}} \psi$. By induction hypothesis, this is equivalent to: there is an s-total strategy ς s.t. for every $\chi' = \langle \alpha', \gamma' \rangle \in \mathsf{usl}(\overline{\chi}[x \mapsto \varsigma]), \mathcal{G}, \chi', s \models_{\text{USL}} \mathsf{usl}_{\gamma'}(\psi)$. Then, by applying Def. 25, this is true iff for every $\chi = \langle \alpha, \gamma \rangle \in \mathsf{usl}(\overline{\chi})$ there is a total strategy ς such that $\mathcal{G}, \chi[x \mapsto \varsigma], s \models_{\text{USL}} \mathsf{usl}_\gamma(\psi)$, iff for every $\chi = \langle \alpha, \gamma \rangle \in \mathsf{usl}(\overline{\chi}), \mathcal{G}, \chi, s \models_{\text{USL}} \langle\!\langle x \rangle\!\rangle_d \mathsf{usl}_\gamma(\psi)$.
  - If ψ is not a state formula, then $\mathcal{G}, \overline{\chi}, s \models_{\text{SL}} \langle\!\langle x \rangle\!\rangle \psi$ iff there is an s-total strategy ς s.t. $\mathcal{G}, \overline{\chi}[x \mapsto \varsigma], s \models_{\text{SL}} \psi$. By induction hypothesis, this is equivalent to: there is an s-total strategy ς s.t. for every $\chi' = \langle \alpha', \gamma' \rangle \in \mathsf{usl}(\overline{\chi}[x \mapsto \varsigma]), \mathcal{G}, \chi', \Lambda^{\chi[x \mapsto \varsigma]} \models_{\text{USL}} \mathsf{usl}_{\gamma'}(\psi)$. This is true iff for every $\chi = \langle \alpha, \gamma \rangle \in \mathsf{usl}(\overline{\chi})$ there is a total strategy ς such that $\mathcal{G}, \chi[x \mapsto \varsigma], \Lambda^{\chi[x \mapsto \varsigma]} \models_{\text{USL}} \mathsf{usl}_\gamma(\psi)$. This is true iff for every $\chi = \langle \alpha, \gamma \rangle \in \mathsf{usl}(\overline{\chi})$ there is a total strategy ς such that for every $\lambda \in out(\chi, s), \mathcal{G}, \chi[x \mapsto \varsigma], \lambda \models_{\text{USL}} \mathsf{usl}_\gamma(\psi)$, which is true iff there is a total strategy ς such that $\mathcal{G}, \chi[x \mapsto \varsigma], s \models_{\text{USL}} [\![x_\emptyset]\!](\emptyset \rhd x_\emptyset)\, \mathsf{usl}_\gamma(\psi)$, iff $\mathcal{G}, \chi, s \models_{\text{USL}} \langle\!\langle x \rangle\!\rangle_d [\![x_\emptyset]\!](\emptyset \rhd x_\emptyset)\, \mathsf{usl}_\gamma(\psi)$.

- Case $\varphi = (a, x)\psi$: $\mathcal{G}, \overline{\chi}, s \models_{\text{SL}} (a, x)\psi$ iff $\mathcal{G}, \overline{\chi}[a \mapsto \overline{\chi}(x)], s \models_{\text{SL}} \psi$. We can easily check that $\mathsf{usl}(\overline{\chi}[a \mapsto \overline{\chi}(x)]) = \{\langle \alpha, \gamma[a \mapsto x] \rangle \mid \langle \alpha, \gamma \rangle \in \mathsf{usl}(\overline{\chi})\}$. Then by induction hypothesis, $\mathcal{G}, \overline{\chi}, s \models_{\text{SL}} (a, x)\psi$ iff for every $\chi = \langle \alpha, \gamma \rangle \in \mathsf{usl}(\overline{\chi}), \mathcal{G}, \langle \alpha, \gamma[a \mapsto x] \rangle, s \models_{\text{USL}} \mathsf{usl}_{\gamma[a \mapsto x]} \psi$.

– If $a \in \mathrm{dom}(\gamma)$ then $a \in \mathrm{dom}(\overline{\chi})$. Thus, for every $\chi = \langle \alpha, \gamma \rangle \in \mathsf{usl}(\overline{\chi}), \gamma[a \mapsto x] = \gamma[a \ominus \gamma(a)][a \oplus x]$. Then $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} (a, x)\psi$ iff for every $\chi = \langle \alpha, \gamma \rangle \in \mathsf{usl}(\overline{\chi}), \mathcal{G}, \chi, s \models_{\mathrm{USL}} (a \not\rhd \gamma(a))(a \rhd x)\,\mathsf{usl}_{\gamma[a \mapsto x]}(\psi)$.

– If $a \notin \mathrm{dom}(\gamma)$ then $a \notin \mathrm{dom}(\overline{\chi})$. Thus, for every $\chi = \langle \alpha, \gamma \rangle \in \mathsf{usl}(\overline{\chi}), \gamma[a \mapsto x] = \gamma[a \oplus x]$. Then $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} (a, x)\psi$ iff for every $\chi = \langle \alpha, \gamma \rangle \in \mathsf{usl}(\overline{\chi}), \mathcal{G}, \chi, s \models_{\mathrm{USL}} (a \rhd x)\,\mathsf{usl}_{\gamma[a \mapsto x]}(\psi)$.

- Case $\varphi = \mathsf{X}\psi$: let us first introduce some notations for the translation of strategy contexts. Complete definitions are given in Appendix B. Let $\overline{\chi}$ be a strategy context and s a state, then we write $\langle \overline{\chi}, s \rangle^1$ for the pair $\langle \overline{\chi}^{s \cdot s^1}, s^1 \rangle$ s.t. s^1 is the successor of s determined by $\overline{\chi}$: $s^1 = tr(s, \delta)$, where δ is the decision s.t. for every $a \in Ag, \delta(a) = \overline{\chi}(a)(s)$. Now, $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} \mathsf{X}\psi$ iff $\mathcal{G}, \langle \overline{\chi}, s \rangle^1 \models_{\mathrm{SL}} \psi$. By induction hypothesis and Def. 25, this is equivalent to: for any $\chi \in \mathsf{usl}(\overline{\chi}), \mathcal{G}, \chi^{s \cdot s^1}, s^1 \models_{\mathrm{USL}} \mathsf{usl}_\gamma(\psi)$. Thanks to Remark 5, this is true iff $\mathcal{G}, \chi^{\Lambda_0^{\chi,s} \cdot \Lambda_1^{\chi,s}}, \Lambda_{\geqslant 1}^{\chi,s} \models_{\mathrm{USL}} \mathsf{usl}_\gamma(\psi)$, and then iff $\mathcal{G}, \chi, \Lambda^{\chi,s} \models_{\mathrm{USL}} \mathsf{X}\,\mathsf{usl}_\gamma(\psi)$. □

Now, we can prove that USL is at least as expressive as SL:

Lemma 3. *SL $\preccurlyeq_{exp}$ USL.*

Proof. Let $\mathcal{G}$ be a CGS and φ be a sentence of SL. By applying Lemma 2 to the particular case where $\overline{\chi}$ is the empty strategy context and s is the initial state of $\mathcal{G}$, we have that $\mathcal{G} \models_{\mathrm{SL}} \varphi$ iff $\mathcal{G} \models_{\mathrm{USL}} \mathsf{usl}(\varphi)$. □

Corollary 1. *The satisfiability problem for USL is not decidable.*

Proof. The undecidability of the satisfiability problem for SL is established in [7]. If USL had a decision procedure for this problem, thanks to Lemma 3 it would also give a decision procedure for SL. In Sect. 5.1 we draw another corollary of Lemma 3: the model-checking problem for USL has no elementary lower bound. □

4.3. USL is strictly more expressive than SL

The possibility to define an equality predicate between sets of actions in USL induces the possibility to *count* the actions that are available. In particular, the uniqueness existential operator defined in Sect. 3.2 enables to distinguish two models with actions having the same consequences. Therefore, we can compare the expressive powers of USL and SL by help of very simple examples.

Figure 4: Illustration of the comparison between USL and SL: $\mathcal{G}_1$ and $\mathcal{G}_2$

Lemma 4. *USL $\not\preccurlyeq_{dis}$ SL*

Proof. The models $\mathcal{G}_1$ and $\mathcal{G}_2$ in Fig. 4 feature one agent a and one atomic proposition p. Both these models have a single state s. In the figure, every action from every state is represented by an arrow. Each arrow goes from a state to its successor if a plays the corresponding action. Using the operator $\neq_{\mathsf{X}}$ introduced in Sect. 3.2.1, we then have:

- $\mathcal{G}_1, s \models_{\mathrm{USL}} \langle\!\langle !x \rangle\!\rangle (a \rhd x)\mathsf{X}p$
- $\mathcal{G}_2, s \not\models_{\mathrm{USL}} \langle\!\langle !x \rangle\!\rangle (a \rhd x)\mathsf{X}p$
- While, for any SL formula $\varphi, \mathcal{G}_1, s \models_{\mathrm{SL}} \varphi$ iff $\mathcal{G}_2, s \models_{\mathrm{SL}} \varphi$. □

We finally end up with the expected expressiveness result.

Theorem 1. *SL $\prec_{exp}$ USL*

Proof. By Lemmas 4 and 1, we have USL $\not\preccurlyeq_{exp}$ SL. Then combine with Lemma 3. □

Remark 6. Note that the proof of Lemma 4 distinguishes between SL and USL expressive powers without use of the operator $(A \not\rhd x)$. Strategy refinement is indeed enough to make this distinction.

5. Model-Checking

In this section, we tackle the model-checking problem for USL. Let $\mathcal{G}$ be a CGS, we call the *size* of $\mathcal{G}$ and write $|\mathcal{G}|$, the sum $|St| + |Act|$. Given a logic $\mathcal{L}$, we note $\mathsf{MC}(\mathcal{L})$ its finite model checking problem: given a formula φ of $\mathcal{L}$ and a model $\mathcal{G}$ such that $|\mathcal{G}| \in \mathbb{N}$, it decides whether $\mathcal{G} \models_{\mathcal{L}} \varphi$ or not. In Sect. 5.1, we show that the complexity of MC(USL) has no elementary upper bound over the length of the formula. This is given as a corollary of Theorem 1. In Sect. 5.2, we show that it is decidable in time NonElementary over the length of the formula.

5.1. MC(*USL*) *has no elementary upper bound*

The embedding of SL into USL (see Theorem 1) provides us the following theorem:

Theorem 2. *The complexity of* MC(*USL*) *has no elementary upper bound over the length of formulas.*

Proof. Suppose that there is an elementary bound $\mathcal{B}(|\varphi|)$ over the length of formulas for MC(USL). Then, for any SL formula φ, the model-checking of φ is decidable in $\mathcal{B}(|\,\mathsf{usl}(\varphi)|)$, where $\mathsf{usl}(\varphi)$ refers to Def. 24. One checks that $|\,\mathsf{usl}(\varphi)| \in O(|\varphi|)$. So, the model-checking for φ is decidable in $\mathcal{B}(O(|\varphi|^2))$. Then MC(SL) has an elementary bound $\mathcal{B}(O(|\varphi|^2))$, which is in contradiction with the result proved in [7]. □

5.2. MC(*USL*) *is decidable in time* NonElementary

In this section, we establish two results. First, we prove the existence of an algorithm deciding the model-checking of USL in time NonElementary. Second, in Sect. 5.2.5, we split the set of USL formulas in fragments USL^k, where $k \in \mathbb{N}$, and show that, for every $k \in \mathbb{N}$, $\mathsf{MC}(\mathrm{USL}^k)$ is decidable in time $k+1$-Exponential.

For the first result, we use a reduction of MC(USL) to the model-checking problem for the Quantified Computation Tree Logic (QCTL* [15, 16]) under its *tree semantics*. This reduction is made in two steps:

1. In Sect. 5.2.1, we define an alternative semantics for USL, called USL^∞, that uses infinite executions exclusively. Then we define a reduction of MC(USL) to $\mathsf{MC}(\mathrm{USL}^\infty)$ in Sect. 5.2.2.
2. In Sect. 5.2.3, we recall the definitions of QCTL* and its *tree semantics*. Then, in Sect. 5.2.4, we devise a reduction of $\mathsf{MC}(\mathrm{USL}^\infty)$ to MC(QCTL*).

5.2.1. USL$^\infty$

In USL, if a context specifies that an agent is committed to multi-strategies yielding non-intersecting sets of actions after a given track, then the outcomes of this context end after this track. USL^∞ is basically an adaptation of the semantics of USL s.t. outcomes never end. In USL^∞, we exhibit a special state $s_\perp$ and a special atom $p_{s_\perp}$, which holds exactly in $s_\perp$, in order to simulate the end of outcomes. The state $s_\perp$ is not accessible through the transition tr from any other state of the CGS.

However, the outcomes induced by a context may go through $s_\perp$. This is the case if the context includes contradictory sets of actions for a given agent (*e.g.*, sets of actions in empty intersection). In USL, such a context would induce outcomes that end whereas in USL^∞ it induces outcomes going to $s_\perp$ and staying there forever. Consequently, USL^∞ is interpreted in the subclass of CGS's in which $s_\perp$ and $p_{s_\perp}$ are interpreted this way. We call them *CGS's for USL*$^\infty$ and write them CGS$^\infty$*s*.

Definition 26 (CGS$^\infty$*s*). A CGS $\mathcal{G} = \langle Ag, St, At, prop, \mathrm{Ac}, tr, s_0\rangle$ is a CGS$^\infty$ iff:

- $s_\perp \in St$
- $p_{s_\perp} \in At$
- $prop(s_\perp) = \{p_{s_\perp}\}$
- for every $\delta \in Dec$, for every $s \in St$, $tr(s, \delta) = s_\perp$ iff $s = s_\perp$

In USL^∞, the successors of a track θ under a context χ are written $succ^\infty_\chi(\theta)$. They are defined the same way as in USL, except that if the set of transitions enabled by the intersection of expressed choices is empty (that is if the context χ indicates contradictory sets of actions for at least one agent), then the only successor is $s_\perp$:

Definition 27 (Successor Function). Let $\mathcal{G} = \langle Ag, St, At, prop, Act, tr, s_o\rangle$ be a CGS and χ be a context. Then the *successor* function $succ^{\infty}_{\chi} : Track \to \mathcal{P}(St)$ induced by χ is defined, for any track θ, by:

$$succ^{\infty}_{\chi}(\theta) = \begin{cases} succ_{\chi}(\theta) & \text{if it is not empty} \\ \{s_{\perp}\} & \text{otherwise} \end{cases}$$

The set $out^{\infty}(\chi, \theta)$ of outcomes of a context χ and a track θ is defined as in USL, except that they are all infinite.

Definition 28 (Outcomes). Let $\mathcal{G} = \langle Ag, St, At, prop, Act, tr, s_o\rangle$ be a CGS, θ be a track and χ be a context. We define the set of *outcomes* of χ and θ in $\mathcal{G}$ as the set $out^{\infty}(\chi, \theta)$ of paths λ in $\mathcal{G}$ s.t. $\lambda_0 = \text{last}(\theta)$ and for any $i \in \mathbb{N}$, $\lambda_{i+1} \in succ^{\infty}_{\chi^{\lambda_{\leqslant i}}}(\lambda_{\leqslant i})$.

We do not detail the definition for the *satisfaction relation* $\models_{\text{USL}^{\infty}}$. It is defined by structural induction on formulas, the same way as $\models_{\text{USL}}$ except that the outcome function is out^{∞} instead of out. Note also that since in USL$^{\infty}$ any executions in the outcomes of a context and a track execution is a path, we do not need to check for the execution length when interpreting temporal operators.

5.2.2. Reduction of MC(*USL*) *to* MC(*USL*$^{\infty}$)

The reduction of MC(USL) to MC(USL$^{\infty}$) consists in a transformation from any USL formula φ to a USL$^{\infty}$ formula φ^{∞}, and from any CGS $\mathcal{G}$ to a CGS$^{\infty}$ $\mathcal{G}^{\infty}$, such that:

- $\mathcal{G}^{\infty}$ is an adaptation of $\mathcal{G}$ so as to answer the constraints in Def. 26.
- we add occurrences of $\neg p_{s_{\perp}}$ in φ^{∞} in order to force the execution not to go to $s_{\perp}$ before satisfying the subformulas that are in the scope of temporal operators X and U.

Let us first define the transformation of CGS's.

Definition 29 (Tranformation of any CGS into a CGS$^{\infty}$). Let $\mathcal{G} = \langle Ag, St, At, prop, \text{Ac}, tr, s_0\rangle$ be a CGS. We note $\mathcal{G}^{\infty}$ the CGS$^{\infty}$ $\langle Ag, St^{\infty}, At^{\infty}, prop^{\infty}, \text{Ac}, tr^{\infty}, s_0\rangle$ defined by:

- $St^{\infty} = St \cup \{s_{\perp}\}$
- $At^{\infty} = At \cup \{p_{s_{\perp}}\}$
- for every $s \in St, prop^{\infty}(s) = prop(s)$
- for every $(s, \delta) \in St \times Dec, tr^{\infty}(s, \delta) = tr(s, \delta)$

Now we come to the transformation of an USL formula φ. In the treatment of temporal operators X and U, this transformation explicitly specifies that the path is not sent to the state that satisfies $p_{s_{\perp}}$ (*i.e.* to $s_{\perp}$).

Definition 30 (Transformation of USL formulas into USL$^{\infty}$). The transformation of a USL formula φ to the USL$^{\infty}$ formula φ^{∞} is defined by structural recursion on φ:

- $p^{\infty} = p$
- $(\neg\varphi)^{\infty} = \neg\varphi^{\infty}$
- $(\varphi_1 \wedge \varphi_2)^{\infty} = \varphi_1^{\infty} \wedge \varphi_2^{\infty}$
- $(\langle\!\langle x\rangle\!\rangle\varphi)^{\infty} = \langle\!\langle x\rangle\!\rangle\varphi^{\infty}$
- $((A \rhd x)\varphi)^{\infty} = (A \rhd x)\varphi^{\infty}$
- $((A \not\rhd x)\varphi)^{\infty} = (A \not\rhd x)\varphi^{\infty}$
- $(\mathsf{X}\varphi)^{\infty} = \mathsf{X}(\varphi^{\infty} \wedge \neg p_{s_{\perp}})$

- $(\varphi_1 \mathsf{U} \varphi_2)^\infty = (\varphi_1^\infty \wedge \neg p_{s_\perp}) \mathsf{U} (\varphi_2^\infty \wedge \neg p_{s_\perp})$

We have the following correspondence:

Proposition 2. *For any CGS $\mathcal{G}$ and for any formula φ, $\mathcal{G} \models \varphi$ iff $\mathcal{G}^\infty \models_{USL^\infty} \varphi^\infty$.*

Proof. By structural induction on φ. □

*5.2.3. QCTL**

The logic QCTL* [16] is an extension of CTL* by means of quantifiers $\exists p$ and $\forall p$, where p is an atomic proposition. Just as CTL*, it is interpreted in *Kripke structures*. The quantifiers $\exists p$ and $\forall p$ range over the possible valuations for p.

Definition 31 (QCTL* formulas). Let At be a set of propositions. Then the set of *QCTL* formulas* is defined by the following grammar:

- *State formulas:*
$$\psi ::= p \mid \neg\psi \mid \psi \wedge \psi \mid \mathsf{E}\varphi \mid \mathsf{A}\varphi \mid \exists p.\psi \mid \forall p.\psi$$
- *Path formulas:*
$$\varphi ::= \psi \mid \neg\varphi \mid \varphi \wedge \varphi \mid \mathsf{X}\varphi \mid \varphi \mathsf{U} \varphi$$

where $p \in At$.

In [16], the authors give two different semantics for QCTL*: the *structure semantics* and the *tree semantics*. Both semantics use *Kripke structures*. In the *structure semantics*, quantifiers $\exists p$ and $\forall p$ quantify over labellings of the proposition p in the Kripke structure itself. In the *tree semantics*, they quantify over labellings of p in its *execution tree*. Here we are interested in the *tree semantics*. Nevertheless, in order to define the *tree semantics* we first need to present the *structure semantics*.

Let us start with the definition of Kripke structures.

Definition 32 (Kripke structures). A *Kripke structure* is a tuple $\mathcal{K} = \langle St, At, prop, R, s_o \rangle$ where:

- St is an enumerable non-empty set of *states*
- At is a finite non-empty set of *atomic propositions*
- $prop : St \to \mathcal{P}(At)$ is a valuation function which maps each state s to the set of propositions true in s
- $R \subseteq St \times St$ is a relation of accessibility
- $s_0 \in St$ is an *initial state*

Definition 33 (Track, Path, Outcome). Let $\mathcal{K} = \langle St, At, prop, R, s_o \rangle$ be a Kripke structure. A *track* in $\mathcal{K}$ is a non-empty finite sequence of states $\theta \in St^+$ s.t. for every $i \in [0, |\theta| - 1[$, $R(\theta_i, \theta_{i+1})$.

A *path* in $\mathcal{K}$ is an infinite sequence of states $\eta \in St^\omega$ s.t. every finite prefix of η is a track in $\mathcal{G}$.

The set of all tracks (resp. paths) in $\mathcal{K}$ is written $Track_\mathcal{K}$ (resp. $Path_\mathcal{K}$).

Let also s be a state in St, then the *outcomes* of s are the paths starting at s, *i.e.* $out^{\mathrm{QCTL}^*}(s) = \{\lambda \in Path_\mathcal{K} \mid \lambda_0 = s\}$.

The interpretation of quantifiers in QCTL* uses the notion of *At'-equivalence* between Kripke structures:

Definition 34 (At'-equivalence). Let $\mathcal{K} = \langle St, At, prop, R, s_o \rangle$ and $\mathcal{K}' = \langle St, At, prop', R, s_o \rangle$ be two Kripke structures, and let $At' \subseteq At$. We say that $\mathcal{K}$ and $\mathcal{K}$ are At'-equivalent, and we write $\mathcal{K} \equiv_{At'} \mathcal{K}'$, iff for every $s \in St$ and $p \in At \setminus At'$, $p \in prop(s)$ iff $p \in prop'(s)$.

Now, we can define the relation of satisfaction under the *structure semantics*:

Definition 35 (Satisfaction under the structure semantics). Let $\mathcal{K} = \langle St, At, prop, R, s_o \rangle$ be a Kripke structure. The *satisfaction relation* $\models_{\text{QCTL}^*}$ is defined by induction on formulas, for every state $s \in St$ and for every path λ in $\mathcal{K}$, as follows:

- State formulas
  - $\mathcal{K}, s \models_{\text{QCTL}^*} p$ iff $p \in prop(s)$, with $p \in At$
  - $\mathcal{K}, s \models_{\text{QCTL}^*} \neg\psi$ iff $\mathcal{K}, s \not\models_{\text{QCTL}^*} \psi$
  - $\mathcal{K}, s \models_{\text{QCTL}^*} \psi_1 \wedge \psi_2$ iff $\mathcal{K}, s \models_{\text{QCTL}^*} \psi_1$ and $\mathcal{K}, s \models_{\text{QCTL}^*} \psi_2$
  - $\mathcal{K}, s \models_{\text{QCTL}^*} \mathsf{E}\varphi$ iff there is $\lambda \in out^{\text{QCTL}^*}(s)$ s.t. $\mathcal{K}, \lambda \models_{\text{QCTL}^*} \varphi$
  - $\mathcal{K}, s \models_{\text{QCTL}^*} \mathsf{A}\varphi$ iff for any $\lambda \in out^{\text{QCTL}^*}(s), \mathcal{K}, \lambda \models_{\text{QCTL}^*} \varphi$
  - $\mathcal{K}, s \models_{\text{QCTL}^*} \exists p.\psi$ iff there is $\mathcal{K}'$ s.t. $\mathcal{K} \equiv_{\{p\}} \mathcal{K}'$ and $\mathcal{K}', s \models_{\text{QCTL}^*} \psi$
  - $\mathcal{K}, s \models_{\text{QCTL}^*} \forall p.\psi$ iff for every $\mathcal{K}'$ s.t. $\mathcal{K} \equiv_{\{p\}} \mathcal{K}', \mathcal{K}', s \models_{\text{QCTL}^*} \psi$
- Path formulas
  - $\mathcal{K}, \lambda \models_{\text{QCTL}^*} \psi$ iff $\mathcal{G}, \chi, \lambda_0 \models_{\text{QCTL}^*} \psi$, if ψ is a state formula
  - $\mathcal{K}, \lambda \models_{\text{QCTL}^*} \neg\varphi$ iff $\mathcal{K}, \lambda \not\models_{\text{QCTL}^*} \varphi$
  - $\mathcal{K}, \lambda \models_{\text{QCTL}^*} \varphi_1 \wedge \varphi_2$ iff $\mathcal{K}, \lambda \models_{\text{QCTL}^*} \varphi_1$ and $\mathcal{K}, \lambda \models_{\text{QCTL}^*} \varphi_2$
  - $\mathcal{K}, \lambda \models_{\text{QCTL}^*} \mathsf{X}\varphi$ iff $\mathcal{K}, \lambda_{\geqslant 1} \models_{\text{QCTL}^*} \varphi$
  - $\mathcal{K}, \lambda \models_{\text{QCTL}^*} \varphi_1 \mathsf{U} \varphi_2$ iff there is a number $i \in \mathbb{N}$ s.t. $\mathcal{K}, \lambda_{\geqslant i} \models_{\text{QCTL}^*} \varphi_2$ and s.t. for any $0 \leqslant j < i$ $\mathcal{K}, \lambda_{\geqslant j} \models_{\text{QCTL}^*} \varphi_1$

The *tree semantics* for QCTL* uses the notion of *execution trees* of Kripke structures. First, we define Σ-*labelled* S*-trees*, where Σ and S are finite sets:

Definition 36 (Σ-labelled S-trees). Let Σ and S be two finite sets. A Σ-labelled S-tree is a pair $\mathcal{T} = \langle T, \ell \rangle$ where $T \subseteq S^*$ is a non-empty set of finite words on S s.t. for any non-empty word $n = m \cdot s$ in T with $m \in S^*$ and $s \in S$, we have that $m \in T$; and $\ell : T \to \Sigma$ is a labelling function.

Definition 37. Let $\mathcal{K} = \langle St, At, prop, R, s_o \rangle$ be a finite-state Kripke structure. The *execution tree* of $\mathcal{K}$ is the $\mathcal{P}(At)$-labelled St-tree $\mathcal{T}_{\mathcal{K}} = \langle Track_{\mathcal{K}}, \ell \rangle$ s.t. for every $\theta \in Track_{\mathcal{K}}, \ell(\theta) = prop(\text{last}(\theta))$.

Now, under the *tree semantics*, a QCTL* formula is true in a model $\mathcal{K}$ iff it is true in the execution tree of $\mathcal{K}$ under the structure semantics:

Definition 38 (Satisfaction under the tree semantics). Let $\mathcal{K} = \langle St, At, prop, R, s_o \rangle$ be a Kripke structure. Then the *relation of satisfaction under the tree semantics*, written $\models_{\text{QCTL}^*_T}$, is defined, for every state $s \in St$, for every execution $\lambda \in Path_{\mathcal{K}}$ and for every QCTL* formulas φ, by:

- if φ is a state formula, then $\mathcal{K}, s \models_{\text{QCTL}^*_T} \varphi$ iff $\mathcal{T}_{\mathcal{K}}, s \models_{\text{QCTL}^*} \varphi$
- if φ is a path formula, then $\mathcal{K}, \lambda \models_{\text{QCTL}^*_T} \varphi$ iff $\mathcal{T}_{\mathcal{K}}, \lambda \models_{\text{QCTL}^*} \varphi$

From now on, $\mathsf{MC}(\text{QCTL}^*)$ indicates the model-checking problem for QCTL* under the tree semantics.

5.2.4. Reduction of $\mathsf{MC}(USL^{\infty})$ *to* $\mathsf{MC}(QCTL^{*})$

Here, we define a reduction of $\mathsf{MC}(\mathrm{USL}^{\infty})$ to $\mathsf{MC}(\mathrm{QCTL}^{*})$. This is an adaptation of the one of $\mathsf{MC}(\mathrm{ATL}_{sc})$ to $\mathsf{MC}(\mathrm{QCTL}^{*})$ given in [16]. Basically, we define a transformation:

- from any CGS^{∞} $\mathcal{G}$ to a Kripke structure $\mathcal{K}_{\mathcal{G}}$.
- from any pair of an an USL^{∞} formula φ and a commitment γ to a QCTL^{*} formula $\mathsf{qctl}^{*}_{\gamma}(\varphi)$

s.t. for any USL^{∞} sentence $\varphi, \mathcal{G} \models \varphi$ iff $\mathcal{K}_{\mathcal{G}} \models_{\mathrm{QCTL}^{*}_{\mathcal{T}}} \mathsf{qctl}^{*}_{\gamma_{\emptyset}}(\varphi)$, where $\gamma_{\emptyset}$ is the empty commitment.

To transform a CGS $\mathcal{G}$ into a Kripke model $\mathcal{K}_{\mathcal{G}}$, we use new atomic propositions to encode the actions yielded by multi-strategies and played by agents. A multi-strategy can be seen as a function labelling the execution tree of $\mathcal{K}_{\mathcal{G}}$ with the atomic propositions representing the actions indicated by this multi-strategy. Then, the fact an agent a plays along a multi-strategy x is represented by an adequate labelling of the atomic propositions for the actions played by a. So, a labelling of the execution tree of $\mathcal{K}_{\mathcal{G}}$ can encode any context χ applied to any state s in $\mathcal{G}$ (such an encoding is called the *unravelling tree of* $\mathcal{G}$, χ *and* s: it is defined in Appendix C and used for the proof of Prop. 3).

The set of atoms in $\mathcal{K}_{\mathcal{G}}$ also contains, for any state s, an atomic proposition p_s which is true exactly in s. Finally, it contains a proposition p_{contr}, for encoding the cases of contradictory commitments for at least one agent.

Definition 39 (Tranformation of any CGS^{∞} into a Kripke structure). Let $\mathcal{G} = \langle Ag, St, At, prop, Act, tr, s_o\rangle$ be a CGS^{∞}, s.t. $Act = \{ac_1, \dots ac_k\}$ and let X be a set of variables. We consider the following fresh atomic propositions:

- a fresh atomic proposition p_{contr}
- for any state $s \in St \setminus \{s_\perp\}$ a fresh atomic proposition p_s (we call P_{St} the set of these propositions together with $p_{s_\perp}$, the latter being already in At)
- for every variable x in X, a set of fresh atomic propositions $P^x_{Act} = \{p^x_{ac_1}, \dots p^x_{ac_k}\}$
- for every agent a in Ag, a set of fresh atomic propositions $P^a_{Act} = \{p^a_{ac_1}, \dots p^a_{ac_k}\}$

We also write $P_X = \bigcup_{x\in X} P^x_{Act}$ and $P_{Ag} = \bigcup_{a\in Ag} P^a_{Act}$. Then $\mathcal{K}_{\mathcal{G}} = \langle St, At \cup \{p_{contr}\} \cup P_{St} \cup P_X \cup P_{Ag}, prop_{St}, R, s_0\rangle$ is the Kripke structure s.t.:

- $R \subseteq St \times St$ is the relation s.t.:
  - for every $s, s' \in St \setminus \{s_\perp\} \times St \setminus \{s_\perp\}, R(s, s')$ iff there is $\delta \in Dec$ s.t. $tr(s, \delta) = s'$
  - for every $s \in St, R(s, s_\perp)$
- for every $s \in St, prop_{St}(s) = prop(s) \cup \{p_s\}$.

Note that the relation R indicates all the possible successors of any state in the CGS^{∞}, regardless of any context. Thus, it includes all the transitions going to $s_\perp$, so that $\mathcal{K}_{\mathcal{G}}$ includes the outcomes of the CGS^{∞} induced by any possible context.

The transformation of formulas also uses a successor function. Given a state s in a CGS $\mathcal{G}$ and a *partial decision* (*i.e.* a partial function from agents to actions representing the choices of an action for some agents) δ, it specifies the possible successors of θ, given the actions played by agents in $\mathrm{dom}(\delta)$.

Definition 40 ($succ^{\mathcal{G}}$). Let $\mathcal{G} = \langle Ag, St, At, prop, Act, tr, s_o\rangle$ be a CGS^{∞}, $\chi = \langle \alpha, \gamma\rangle$ be a context and s be a state in $\mathcal{G}$. Let also Ag' be a subset of Ag and $\delta \in Act^{Ag'}$. Then $succ^{\mathcal{G}}(s, \delta) = \{s' \mid \exists \delta' \in Dec \cdot \forall a \in Ag', \delta'(a) = \delta(a) \wedge tr(s, \delta') = s'\}$

Before defining the QCTL^{*} formula from the USL^{∞} formula and the context χ, we need to define the following auxiliary formulas, that express the encoding of multi-strategies and of the context:

Definition 41. Let $\mathcal{G} = \langle Ag, St, At, prop, Act, tr, s_o\rangle$ be a CGS^{∞}, x be a strategy variable and γ be a non-empty commitment. Then:

- For any strategy variable x, $\xi^{ms(x)}$ ensures that the propositions in P^x_{Act} label a multi-strategy, *i.e.* that in every state of every possible execution, there is at least one action ac s.t. p^x_{ac} is true:

$$\xi^{ms(x)} \triangleq \mathsf{A}\Box\Big(\bigvee_{ac\in Act} p^x_{ac}\Big)$$

- $deadlock^\gamma$ expresses that in the current state, at least one agent is committed to contradictory multi-strategies according to the context γ:

$$deadlock^\gamma \triangleq \bigvee_{a\in\mathrm{dom}(\gamma)} \bigwedge_{ac\in Act} \bigvee_{x\in\gamma(a)} \neg p^x_{ac}$$

- ξ^γ ensures that the propositions in P_{Ag} encode the fact that the agents bound by γ play as specified by γ:
  - agents do not play iff the execution is in $s_\perp$ or some of them are committed to contradictory multi-strategies (p_{contr} is true in this case);
  - if agents play then
    - for every agent a in the domain of γ, the actions ac s.t. p^{Ag}_{ac} is true are exactly those s.t. for every variable x s.t. $\langle a, x\rangle \in \gamma$, p^x_{ac} is true.
    - for every agent a not in the domain of γ, for every $ac \in Act$, p^a_{ac} is false.

$$\xi^\gamma \triangleq \Big(p_{s_\perp} \rightarrow \bigwedge_{p\in P_{Ag}} \neg p\Big) \wedge \Big(deadlock^\gamma \rightarrow \Big(p_{contr} \wedge \bigwedge_{p\in P_{Ag}} \neg p^a_{ac}\Big)\Big)$$
$$\wedge\Big(\neg deadlock^\gamma \rightarrow \Big(\bigwedge_{\substack{a\in\mathrm{dom}(\gamma)\\ ac\in Act}} \Big(p^a_{ac} \leftrightarrow \bigwedge_{x\in\gamma(a)} p^x_{ac}\Big) \wedge \Big(\bigwedge_{\substack{a\notin\mathrm{dom}(\gamma)\\ ac\in Act}} \neg p^a_{ac}\Big) \wedge \neg p_{contr}\Big)\Big)$$

- $\xi^{out(\gamma)}$ ensures that, from any state of any execution, the successor of this state complies with $succ^\infty_\chi$:

$$\xi^{out(\gamma)} \triangleq \Big((p_{contr} \vee p_{s_\perp}) \wedge \mathsf{X}p_{s_\perp}\Big) \vee \Big[\bigvee_{s\in St\setminus\{s_\perp\}} \Big[p_s \wedge \Big(\bigvee_{\delta\in X^{\mathrm{dom}(\gamma)}} \Big(\Big[\bigwedge_{a\in\mathrm{dom}(\gamma)} p^a_{\delta(a)}\Big] \wedge \mathsf{X}\Big[\bigvee_{s'\in succ^{\mathcal{G}}(s,\delta)} p_{s'}\Big]\Big)\Big)\Big]\Big]$$

Now, we can define the transformation from a pair of a USL$^\infty$ formula and a commitment to a QCTL* formula:

Definition 42 (Transformation of USL$^\infty$ formulas into QCTL*). For any commitment γ and any USL$^\infty$ formula φ, we define a QCTL* formula $\mathsf{qctl}^*_\gamma(\varphi)$ by structural recursion on φ:

- $\mathsf{qctl}^*_\gamma(p) = p$, for every $p \in At$
- $\mathsf{qctl}^*_\gamma(\neg\varphi) = \neg\,\mathsf{qctl}^*_\gamma(\varphi)$
- $\mathsf{qctl}^*_\gamma(\varphi_1 \wedge \varphi_2) = \mathsf{qctl}^*_\gamma(\varphi_1) \wedge \mathsf{qctl}^*_\gamma(\varphi_2)$
- $\mathsf{qctl}^*_\gamma(\langle\!\langle x\rangle\!\rangle\varphi) = \exists p^x_{ac_1}.\ldots\exists p^x_{ac_k}.\xi^{ms(x)} \wedge \mathsf{qctl}^*_\gamma(\varphi)$
- $\mathsf{qctl}^*_\gamma((A \rhd x)\varphi) = \forall p^{a_1}_{ac_1}.\ldots\forall p^{a_1}_{ac_k}.\forall p^{a_n}_{ac_1}.\ldots\forall p^{a_n}_{ac_k}.\forall p_{contr}.\ \mathsf{A}\,((\Box(\xi^{\gamma[A\oplus x]} \wedge \xi^{out(\gamma[A\oplus x])})) \rightarrow \mathsf{qctl}^*_{\gamma[A\oplus x]}(\varphi))$, where $\mathrm{dom}(\gamma) \cup A = \{a_1, \ldots, a_n\}$
- $\mathsf{qctl}^*_\gamma((A \not\rhd x)\varphi) = \forall p^{a_1}_{ac_1}.\ldots\forall p^{a_1}_{ac_k}.\forall p^{a_n}_{ac_1}.\ldots\forall p^{a_n}_{ac_k}.\forall p_{contr}.\ \mathsf{A}\,((\Box(\xi^{\gamma[A\ominus x]} \wedge \xi^{out(\gamma[A\ominus x])})) \rightarrow \mathsf{qctl}^*_{\gamma[A\ominus x]}(\varphi))$, where $\mathrm{dom}(\gamma) \cup A = \{a_1, \ldots, a_n\}$
- $\mathsf{qctl}^*_\gamma(\mathsf{X}\varphi) = \mathsf{X}\,\mathsf{qctl}^*_\gamma(\varphi)$

- $\mathsf{qctl}^*_\gamma(\varphi_1 \mathsf{U} \varphi_2) = \mathsf{qctl}^*_\gamma(\varphi_1) \mathsf{U} \mathsf{qctl}^*_\gamma(\varphi_2)$

We write $\mathsf{qctl}^*(\varphi)$ for $\mathsf{qctl}^*_{\gamma_\emptyset}(\varphi)$, where $\gamma_\emptyset$ is the empty commitment.

We have the following correspondence:

Proposition 3. *For any USL$^\infty$ sentence φ and for any CGS$^\infty$ $\mathcal{G}$,*

$$\mathcal{G} \models_{USL^\infty} \varphi \text{ iff } \mathcal{K}_\mathcal{G} \models_{QCTL^*_\mathcal{T}} \mathsf{qctl}^*(\varphi)$$

Proof. (Sketch) By structural induction over φ. The formulation of the induction hypothesis uses, for any CGS$^\infty$ $\mathcal{G}$, any context χ and any state s in $\mathcal{G}$, the *unravelling tree* of $\langle \mathcal{G}, \chi, s\rangle$ (written $\mathcal{T}_{(\mathcal{G},\chi,s)}$). It is formally defined in Appendix C. The induction hypothesis is the following: for any USL$^\infty$ formula and for any context $\chi = \langle \alpha, \gamma\rangle$,

- if φ is a state formula then for any state s: $\mathcal{G}, \chi, s \models_{\text{USL}^\infty} \varphi$ iff $\mathcal{T}_{(\mathcal{G},\chi,s)}, s \models_{\text{QCTL}^*} \mathsf{qctl}^*_\gamma(\varphi)$
- If φ is a path formula then for any path λ: $\mathcal{G}, \chi, \lambda \models_{\text{USL}^\infty} \varphi$ iff $\mathcal{T}_{(\mathcal{G},\chi,\lambda_0)}, \lambda \models_{\text{QCTL}^*} \mathsf{qctl}^*_\gamma(\varphi)$ □

Corollary 2. MC(*USL*) *is decidable.*

Remark 7. Let us notice that QCTL* has the same expressive power as the monadic second order logic (MSO) on trees, as shown in [16]. Therefore, the reduction of MC(USL) to MC(QCTL*) also provides a reduction of MC(USL) to the model-checking of MSO on trees.

5.2.5. Model-checking for USL^k

Our reduction of MC(USL) to MC(QCTL*) provides a result of decidability for MC(USL) in time NonElementary. We can get a finer result. For any $k \in \mathbb{N}$, we write Q^{k}CTL* the fragment of QCTL* with k alternations of quantifiers.

In [16], the authors also show that, for any $k \in \mathbb{N}$, the problem MC(Q^{k}CTL*) is decidable in time $k+1$-Exponential over the length of φ. As it happens, we can as well split the set of USL formulas into fragments USL^k s.t. for every $k \in \mathbb{N}$, MC(USL^k) reduces to MC(Q^{k}CTL*).

These fragments are defined using an extension of the notion of alternation depth of formulas, where the binder and the unbinder of USL are also treated as quantifiers:

Definition 43 (General alternation depth, USL^k). We call *general alternation depth* of φ, and write $gad(\varphi)$, the alternation depth of φ, where $(A \rhd x)$ and $(A \not\rhd x)$ are counted as:

- universal quantifiers when they are in the scope of an even number of $\neg$
- existential quantifiers otherwise

and $\langle\!\langle x \rangle\!\rangle$ is counted as:

- an existential quantifier when it is in the scope of an even number of $\neg$
- an universal quantifier otherwise.

Now, for any integer k, USL^k is the fragment of USL with at most k general quantifiers alternation, that is the set of USL formulas φ s.t. $gad(\varphi) \leqslant k$.

Theorem 3. *For any $k \in \mathbb{N}$, the problem* MC(*USL^k*)*, of checking the satisfaction of an USL^k sentence φ in a CGS $\mathcal{G}$, is decidable in time k + 1-*Exponential *over the length of φ.*

Proof. Let φ be a USL formula and let $\mathcal{G}$ be a CGS. One checks that $\mathsf{qctl}^*(\varphi^\infty) \in \text{Q}^{gad(\varphi)}\text{CTL}^*$. Furthermore, one easily checks that $|\varphi^\infty|$ is in $O|\varphi|$ and there is an integer k s.t. $|\mathsf{qctl}^*(\varphi^\infty)| \in O(|\varphi^\infty| \times |\mathcal{G}|^k)$. Thus, the complexity bound of MC(Q$^{gad(\varphi)}$CTL*) is preserved when using these transformations. □

Table 1: Strategy contexts, revocation and refinement in multi-agent temporal logics (p stands for "partial support").

	Strategy contexts	Explicit revocation	No systematic revocation	Revocable strategies	Refinable strategies	Sustainable control	Predicate of equality
ATL				×			
BSIL	p	p		×			
IATL	p		×				
SL	×			×			
ATL_{sc}	×	×		×			
USL	×	×	×	×	×	×	×

6. Related Work

Several directions have already been explored for extensions of ATL-ATL* considering the strategies played by different coalitions of agents. Table 1 sums up the main mechanisms at stake in this article and their occurrences in related works.

In BSIL [17], an operator explicitly mentions that a strategy bound to a coalition can be composed with the context. In terms of expressiveness, it extends ATL but is incomparable with ATL*. It is subsumed both by SL and ATL_{sc} (and therefore by USL) but has "only" a Pspace-complete model-checking problem.

The overwriting of strategies is also addressed in IATL [18]. In this proposition, the authors make a distinction between, on the one hand, ATL- and SL-style revocable strategies and, on the other hand, irrevocable strategies. They formalise the latter in a language named *ATL with Irrevocable strategies* (IATL). In IATL, once an agent is bound to a strategy, she never revokes it and she cannot be bound to another strategy anymore. We believe that USL multi-strategies offer an adequate synthesis between this view and the classical one of systematic revocation: in USL, when an agent already bound to a multi-strategy is bound to a new one, she commits to the second one without revoking the first one.

The work presented here deeply refers to SL [6, 7], which fully enables to compose the different strategies followed by agents in a context. Nevertheless, the composition of several strategies for one agent is not possible in that formalism, contrary to USL, since an agent overwrites her previous strategy when she is bound to a new one.

Incidentally (as explained in Remark 1), SL also bears a syntactic constraint that is not present in USL: all agents must be explicitly bound to a strategy before evaluating a temporal formula. In addition to the unnecessary important size of formulas, this raises the more fundamental problem of (lack of) *modularity*: a single informal specification may give rise to *different* SL formulas depending on the number of agents of the system not mentioned in the specification.

The idea of agents explicitly unbound from their current strategies is also present in ATL_{sc} [2, 3] with the operator $\cdot\rangle A\langle\cdot$. Yet, strategies are also automatically revoked in case a given agent is bound to several strategies: it is not possible for an agent to refine her strategy.

All these formalisms enrich the composition of strategies defined in ATL with a notion of strategy context. But they only compose strategies if they concern distinct agents. We figure USL as a further step in this contextualisation. In USL, binding an agent to a strategy commits her to play this strategy. This strategy can then be refined, and its possible revocation must be explicit.

As far as we know, USL is the first proposition of a multi-agent logic featuring finite executions. The USL interpretation of temporal operators in tracks is the traditional one defined in [11] and used in [19] under the name of *neutral interpretation*. The ensuing possibility to characterise equality and dependences is also, as far as we know, new in the field.

7. Conclusion

In this article we defined the logic USL, in which we can reason about agents refining or revoking their strategies. This unifies a rich composition of strategies that allows strategies refinement with the usual revocation of strategies

developed in the literature. It also gathers in a single formal language both a treatment of agent actions through non-deterministic multi-strategies and their classical treatment by use of deterministic strategies. USL strictly extends SL, and is in particular able to express what we called sustainable control and equality of multi-strategies. The syntax of USL is also more flexible than that of SL and is better adapted to modular specifications. Its model-checking problem is NonElementary. Furthermore, we show that one can split the set of USL formulas in fragments USL^k, where $k \in \mathbb{N}$, and show that, for every $k \in \mathbb{N}$, $\mathsf{MC}(\text{USL}^k)$ is decidable in time k + 1-Exponential.

A sound and complete axiomatisation for ATL was given in [20]. It would be interesting to seek the axiomatisation of multi-agent logics bearing the mechanisms of strategy refinement and explicit revocation that are introduced for USL in this article.

It is also important to deepen the comparison of USL with pre-existing formalisms. We have good reasons to think that sustainable control is not expressible in SL. We also want to compare the expressive power of USL with the logic QD_μ [21]. The latter formalism enables to express fixed-point properties about strategies and subsumes SL over CGS's of finite bounded size. Since the refinement of multi-strategies is close to a fixed point, we think that this comparison may yield interesting results.

We are currently working on an extension of CGS in which the transition function is not total, in order to model *conflicting capabilities*, which are capabilities of different agents that cannot be applied together at the same time.

As future work, we plan to study applications of USL. In particular, as we already noticed in [22], temporal multi-agents logics can be useful to investigate goal- and agent-oriented requirements engineering problems. For instance, thanks to multi-strategy refinement, our framework allows to give a precise semantics to the fact that a coalition can at any time choose to help a second *or* a third coalition (both sharing actors with the first one) to satisfy their requirements. We will investigate on the possibilities offered by USL for the verification of requirements engineering models.

A search for fragments of USL with better complexity results should be led. In [8], the authors define a fragment of SL, namely *One-Goal Strategy Logic* (SL $[1_G]$), enjoying the bounded tree-model property and 2-ExpTime complete satisfiability. We intend to investigate whether the corresponding fragment of USL increases the expressive power of SL $[1_G]$ and benefits from its lower complexity results.

We will also study further the expressiveness of USL. In particular, we think the unbinder offers interesting perspectives: in this article we exclusively discussed the meaning of unbinding an agent before she is herself bound again. The expressiveness given by unbinding some agents before binding others should also be investigated.

Searches for further characterisation of the dependences between different multi-strategies should also be led. In particular, USL could provide means to formalise questions such as: which changes of multi-strategy x induce changes on multi-strategy y? if both multi-strategies y and z depend on x, do the changes on x affecting y also affect z? This last question implies the existence of a partial ordering of strategies w.r.t. their dependence relations towards x and a notion of *strength of dependence relations* between strategies. The strong dependence introduced in Sect. 3.2.4 is, in our opinion, a first step in that direction.

Acknowledgements We wish to thank the anonymous referees for their detailed reviews which have helped improve this paper. We are also very grateful to Valentin Goranko, Nicolas Markey and Sophie Pinchinat for their useful feedback on earlier versions of this work.

References

[1] R. Alur, T. A. Henzinger, O. Kupferman, Alternating-time temporal logic, J. ACM 49 (2002) 672–713.
[2] T. Brihaye, A. Da Costa Lopes, F. Laroussinie, N. Markey, ATL with strategy contexts and bounded memory, Logical Foundations of Computer Science (2009) 92–106.
[3] A. Da Costa Lopes, Propriétés de jeux multi-agents, Phd thesis, École normale supérieure de Cachan, 2011.
[4] A. Da Costa Lopes, F. Laroussinie, N. Markey, ATL with strategy contexts: Expressiveness and model checking, in: IARCS Annual Conf. on Foundations of Software Technology and Theoretical Computer Science (FSTTCS, volume 8, 2010, pp. 120–132.
[5] K. Chatterjee, T. A. Henzinger, N. Piterman, Strategy logic, Inf. & Comp. 208 (2010) 677–693.
[6] F. Mogavero, A. Murano, M. Y. Vardi, Reasoning about strategies, in: IARCS Annual Conf. on Foundations of Software Technology and Theoretical Computer Science (FSTTCS), volume 8, 2010, pp. 133–144.
[7] F. Mogavero, A. Murano, G. Perelli, M. Y. Vardi, Reasoning about strategies: On the model-checking problem, CoRR abs/1112.6275 (2011).
[8] F. Mogavero, A. Murano, G. Perelli, M. Y. Vardi, What makes ATL* decidable? a decidable fragment of strategy logic, in: 23rd International Conference on Concurrency Theory (CONCUR), 2012, pp. 193–208.

[9] C. Chareton, J. Brunel, D. Chemouil, Towards an Updatable Strategy Logic, in: Proc. 1st International Workshop on Strategic Reasoning SR, 2013.
[10] P. Bouyer, N. Markey, J. Olschewski, M. Ummels, Measuring permissiveness in parity games: Mean-payoff parity games revisited, in: ATVA, 2011, pp. 135–149.
[11] Z. Manna, A. Pnueli, Temporal verification of reactive systems - safety, Springer, 1995.
[12] J. Hintikka, The Principles of Mathematics Revisited, Cambridge University Press, 1996.
[13] A. L. Mann, G. Sandu, M. Sevenster, Independence-friendly logic: A game-theoretic approach, 386, Cambridge University Press, 2011.
[14] J. Väänänen, Dependence logic: A new approach to independence friendly logic, 70, Cambridge University Press, 2007.
[15] E. A. Emerson, A. P. Sistla, Deciding full branching time logic, Information and Control 61 (1984) 175–201.
[16] A. Da Costa, F. Laroussinie, N. Markey, Quantified CTL: expressiveness and model checking, in: Proceedings of the 23rd International Conference on Concurrency Theory (CONCUR'12), Lecture Notes in Computer Science, Springer, Newcastle, UK, 2012, pp. 177–192.
[17] F. Wang, C.-H. Huang, F. Yu, A temporal logic for the interaction of strategies, in: 22nd International Conference on Concurrency Theory (CONCUR), volume 6901 of *Lecture Notes in Computer Science*, Springer, 2011, pp. 466–481.
[18] T. Ågotnes, V. Goranko, W. Jamroga, Alternating-time temporal logics with irrevocable strategies, in: Theoretical aspects of rationality and knowledge, 2007, pp. 15–24.
[19] C. Eisner, D. Fisman, J. Havlicek, Y. Lustig, A. McIsaac, D. V. Campenhout, Reasoning with temporal logic on truncated paths, in: CAV, 2003, pp. 27–39.
[20] V. Goranko, G. Van Drimmelen, Complete axiomatization and decidability of alternating-time temporal logic, Theoretical Computer Science 353 (2006) 93–117.
[21] S. Pinchinat, Quantified mu-calculus with decision modalities for concurrent game structures, Technical Report, Dept. of Computer Science, Faculty of Engineering and Information Technology, Australian National University, 2007.
[22] C. Chareton, J. Brunel, D. Chemouil, A formal treatment of agents, goals and operations using alternating-time temporal logic, in: Brazilian Symposium on Formal Methods (SBMF), 2011, pp. 188–203.

Appendix A. Proof of Proposition 1

Here, we give the proof of Proposition 1 (see p. 9). It is led through the following steps:

- We express the property *a has sustainable control over S from s* as the existence of a multi-strategy ς having the property *being a multi-strategy by which a has sustainable control over S from s*. The set of pairs $\langle s, \varsigma\rangle \in St \times Strat$ where s is a state and ς is a multi-strategy by which a has sustainable control over S from s is written $\mathsf{Cont}_\omega(a, S)$.
- We exhibit a function Φ from $\mathcal{P}(St \times Strat)$ to $\mathcal{P}(St \times Strat)$ and, for any $k \in \mathbb{N}$, we write $\mathsf{Cont}_n(a, S)$ the set obtained by applying n times Φ starting from the set $St \times Strat$. We show that:
  - $\mathsf{Cont}_\omega(a, S)$ is the meet of $\mathsf{Cont}_n(a, S)$ for $n \in \mathbb{N}$.
  - $\mathcal{G}, s \models \langle\!\langle x\rangle\!\rangle(a \rhd x)\Box(\langle\!\langle y\rangle\!\rangle(a \rhd y)\mathsf{X}\psi \wedge \langle\!\langle z\rangle\!\rangle(a \rhd z)\mathsf{X}\neg\psi)$ iff there is a multi-strategy ς s.t. for every $n \in \mathbb{N}, \langle s, \varsigma\rangle \in \mathsf{Cont}_n(a, S)$.

Let us first introduce a notation for the set of binary partitions of a given set:

Notation. Let E be a set, we denote $\mathsf{Par}_2(E)$ the set of binary partitions of E *i.e.* for every $\langle E_1, E_2\rangle \in \mathcal{P}(E) \times \mathcal{P}(E), \langle E_1, E_2\rangle \in \mathsf{Par}_2(E)$ iff

- $E_1 \neq \emptyset$ and $E_2 \neq \emptyset$
- $E_1 \cap E_2 = \emptyset$ and $E_1 \cup E_2 = E$

Now, we define the property, for a pair of a state s and a multi-strategy ς, to be s.t. by playing ς from s, a has sustainable control over S. In that case we say that ς is a multi-strategy of sustainable control for a from s towards S.

Definition 44 (Multi-strategies of sustainable control). Let S be a set of states and a an agent. The set $\mathsf{Cont}_\omega(a, S)$ of pairs $\langle s, \varsigma\rangle$ of a state and a multi-strategy s.t. ς is a *multi-strategy of sustainable control* for a from s towards S is defined coinductively by: for every $\langle s, \varsigma\rangle \in St \times Strat, \langle s, \varsigma\rangle \in \mathsf{Cont}_\omega(a, S)$ iff

$$\begin{aligned}
&\exists \langle Ac_1, Ac_2\rangle \in \mathsf{Par}_2(\varsigma(s)) \cdot \forall ac_1 \in Ac_1 \cdot \forall ac_2 \in Ac_2 \cdot \exists S_1, S_2 \subseteq St \text{ s.t.} \\
&\qquad\qquad \overline{tr}(s, \{a \mapsto ac_1\}) = S_1 \text{ and } \overline{tr}(s, \{a \mapsto ac_2\}) = S_2 \\
&\qquad \text{and} \quad S_1 \subseteq S \text{ and } S_2 \subseteq St \setminus S \\
&\qquad \text{and} \quad \forall s' \in S_1 \cup S_2 \cdot \langle s', \varsigma^{s\cdot s'}\rangle \in \mathsf{Cont}_\omega(a, S)
\end{aligned}$$

We show that a has sustainable control over S from s iff there is a multi-strategy of sustainable control for a from s towards S.

Lemma 5. *for every s, $\mathsf{SCont}(a, S, s)$ iff there is a multi-strategy ς s.t. $\langle s, \varsigma\rangle \in \mathsf{Cont}_\omega(a, S)$.*

Proof. (Sketch)

- $\Rightarrow$: for every s s.t. $\mathsf{SCont}(a, S, s)$, let ac_1^s and ac_2^s be s.t.:

$$\begin{array}{lll} \exists S_1, S_2 \subseteq St & \text{s.t.} & \overline{tr}(s, \{a \mapsto ac_1\}) = S_1 \text{ and } \overline{tr}(s, \{a \mapsto ac_2\}) = S_2 \\ & \text{and} & S_1 \subseteq S \text{ and } S_2 \subseteq St \setminus S \\ & \text{and} & \forall s' \in S_1 \cup S_2 \cdot \mathsf{SCont}(a, S, s') \end{array}$$

  Then any multi-strategy ς that satisfies $\varsigma(\tau) = \{ac_1^{\mathrm{last}(\tau)}, ac_2^{\mathrm{last}(\tau)}\}$ for every track τ s.t. $\mathsf{SCont}(a, S, \mathrm{last}(\tau))$, also satisfies $\langle s, \varsigma\rangle \in \mathsf{Cont}_\omega(a, S)$.

- $\Leftarrow$: Straightforward. □

We show that $\mathsf{Cont}_\omega(a, S)$ is the limit of applying n times the function Φ to $St \times Strat$, when n approaches ω and Φ is defined as follows:

Definition 45 (Function Φ). Given an agent a and a set S of states, Φ is a map[2] from $\mathcal{P}(St \times Strat)$ to $\mathcal{P}(St \times Strat)$ s.t. for every $U \in \mathcal{P}(St \times Strat)$, $\Phi(U)$ is the set of pairs $\langle s, \varsigma\rangle$ s.t.

$$\begin{array}{l} \exists \langle Ac_1, Ac_2\rangle \in \mathsf{Par}_2(\varsigma(s)) \cdot \forall ac_1 \in Ac_1 \cdot \forall ac_2 \in Ac_2 \cdot \exists S_1, S_2 \subseteq St \text{ s.t.} \\ \qquad\qquad \overline{tr}(s, \{a \mapsto ac_1\}) = S_1 \text{ and } \overline{tr}(s, \{a \mapsto ac_2\}) = S_2 \\ \qquad\qquad \text{and} \quad S_1 \subseteq S \text{ and } S_2 \subseteq St \setminus S \\ \qquad\qquad \text{and} \quad \forall s' \in S_1 \cup S_2 \cdot \langle s', \varsigma^{s \cdot s'}\rangle \in U \end{array}$$

For any n we write $\mathsf{Cont}_n(a, S)$ the set obtained by applying n times the function Φ starting from the set $St \times Strat$. Intuitively, for any state s and multi-strategy ς, $\langle s, \varsigma\rangle \in \mathsf{Cont}_n(a, S)$ iff, by playing ς from s, a is able to refine ς so as to decide whether the next state is in S or not. And a keeps this ability during the first n transitions of any possible execution if she plays ς.

Definition 46 ($\mathsf{Cont}_n(a, S)$). Let S be a set of states and a an agent, s a state and ς a multi-strategy.

- $\mathsf{Cont}_0(a, S) = St \times Strat$
- for every $n \in \mathbb{N}$, $\mathsf{Cont}_{n+1}(a, S) = \Phi(\mathsf{Cont}_n(a, S))$

Now, we can show that $\mathsf{Cont}_\omega(a, S) = \bigcap_{n\in\mathbb{N}} \mathsf{Cont}_n(a, S)$:

Lemma 6. *For every $s, \varsigma, \langle s, \varsigma\rangle \in \mathsf{Cont}_\omega(a, S)$ iff for every $n \in \mathbb{N}$, $\langle s, \varsigma\rangle \in \mathsf{Cont}_n(a, S)$.*

Proof. (Sketch)

- $\Rightarrow$: By recurrence over n, for every n, for any agent a and any set S, $\mathsf{Cont}_\omega(a, S) \subseteq \mathsf{Cont}_n(a, S)$.
- $\Leftarrow$: $\mathsf{Cont}_\omega(a, S)$ is defined coinductively over $St \times Strat$, $\mathcal{P}(St \times Strat)$ is a complete lattice and it is easy to see that Φ is monotone. So $\mathsf{Cont}_\omega(a, S)$ is the greatest fixed point of Φ in $\mathcal{P}(St \times Strat)$.

[2] Although Φ depends on a and S, we do not use the notation $\Phi_{a,S}$ since there is no ambiguity.

Now, in order to show that $\bigcap_{n\in\mathbb{N}} \mathsf{Cont}_n(a,S) \subseteq \mathsf{Cont}_\omega(a,S)$, it is sufficient to show that $\bigcap_{n\in\mathbb{N}} \mathsf{Cont}_n(a,S) \subseteq \Phi(\bigcap_{n\in\mathbb{N}} \mathsf{Cont}_n(a,S))$. Let $\langle s,\varsigma\rangle \in \bigcap_{n\in\mathbb{N}} \mathsf{Cont}_n(a,S)$. For all $k \in \mathbb{N}^*$, there is $\langle Ac_1^k, Ac_2^k\rangle \in \mathsf{Par}_2(\varsigma(s))$ s.t.:

$$\forall k \in \mathbb{N} \cdot \forall ac_1 \in Ac_1^k \cdot \forall ac_2 \in Ac_2^k \cdot \exists S_1, S_2 \subseteq St \text{ s.t.}$$
$$\overline{tr}(s,\{a \mapsto ac_1\}) = S_1 \text{ and } \overline{tr}(s,\{a \mapsto ac_2\}) = S_2$$
$$\text{and} \quad S_1 \subseteq S \text{ and } S_2 \subseteq St \setminus S$$
$$\text{and} \quad \forall s' \in S_1 \cup S_2 \cdot \langle s', \varsigma^{s\cdot s'}\rangle \in \mathsf{Cont}_{k-1}(a,S)$$

The condition $S_1 \subseteq S$ and $S_2 \subseteq St \setminus S$ ensures that the partition $\langle Ac_1^k, Ac_2^k\rangle$ is the same for any $k \in \mathbb{N}$. Let us write it $\langle Ac_1, Ac_2\rangle$. It is s.t.:

$$\forall k \in \mathbb{N} \cdot \forall ac_1 \in Ac_1 \cdot \forall ac_2 \in Ac_2 \cdot \exists S_1, S_2 \subseteq St \text{ s.t.}$$
$$\overline{tr}(s,\{a \mapsto ac_1\}) = S_1 \text{ and } \overline{tr}(s,\{a \mapsto ac_2\}) = S_2$$
$$\text{and} \quad S_1 \subseteq S \text{ and } S_2 \subseteq St \setminus S$$
$$\text{and} \quad \forall s' \in S_1 \cup S_2 \cdot \langle s', \varsigma^{s\cdot s'}\rangle \in \mathsf{Cont}_k(a,S)$$

So, $\forall s' \in S_1 \cup S_2 \cdot \langle s', \varsigma^{s\cdot s'}\rangle \in \bigcap_{n\in\mathbb{N}} \mathsf{Cont}_n(a,S)$. Thus, $\langle s,\varsigma\rangle \in \Phi(\bigcap_{n\in\mathbb{N}} \mathsf{Cont}_n(a,S))$. Then, $\bigcap_{n\in\mathbb{N}} \mathsf{Cont}_n(a,S) \subseteq \mathsf{Cont}_\omega(a,S)$. □

Let us consider now that S is the set of states s in $\mathcal{G}$ s.t. $\mathcal{G}, s \models \psi$. We have the following lemma:

Lemma 7. *Let S be a set of states in $\mathcal{G}$, s.t. there is an USL formula ψ s.t. for every state s of $\mathcal{G}$, $\mathcal{G}, s \models \psi$ iff $s \in S$. Then $\mathcal{G}, s \models \langle\!\langle x\rangle\!\rangle(a \rhd x)\Box(\langle\!\langle y\rangle\!\rangle(a \rhd y)\mathsf{X}\psi \wedge \langle\!\langle z\rangle\!\rangle(a \rhd z)\mathsf{X}\neg\psi)$ iff there is a multi-strategy ς s.t. for every $n \in \mathbb{N}$, $\langle s,\varsigma\rangle \in \mathsf{Cont}_n(a,S)$.*

Proof. (Sketch) Both sides of the equivalence are equivalent to: there is a multi-strategy ς such that for any λ in $out(\langle(x \mapsto \varsigma), \langle a, x\rangle\rangle)$ and for any $n \in \mathbb{N}$,

$$\exists \langle Ac_1, Ac_2\rangle \in \mathsf{Par}_2(\varsigma(\lambda_{\leqslant n})) \cdot \exists S_1, S_2 \subseteq St \quad \text{s.t.} \quad \overline{tr}(s,\{a \mapsto ac_1\}) = S_1 \text{ and } \overline{tr}(s,\{a \mapsto ac_2\}) = S_2$$
$$\text{and} \quad S_1 \subseteq S \text{ and } S_2 \subseteq St \setminus S$$

□

Now, the proof of Proposition 1 is straightforward, by application of Lemmas 5, 6 and 7.

Appendix B. Syntax and semantics of SL

This section only gives a brief presentation of the syntax and semantics of SL. The reader may refer to [7] for a complete presentation.

Definition 47 (SL syntax). Given a set of atomic propositions At, a set of strategy variables X and a set of agents Ag, the syntax of SL is defined by the following grammar:

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

where $p \in At, a \in Ag$ and $x \in X$.

SL uses a notion of *free agents* of a formula φ ($free(\varphi)$), defined inductively as follows:

- $free(p) = \emptyset$, for $p \in At$
- $free(\neg\varphi) = free(\langle\!\langle x\rangle\!\rangle\varphi) = free([\![x]\!]\varphi) = free(\varphi)$
- $free(\varphi_1 \wedge \varphi_2) = free(\varphi_1 \vee \varphi_2) = free(\varphi_1) \cup free(\varphi_2)$

- $free(\varphi_1 \mathsf{U} \varphi_2) = free(\varphi_1 \mathsf{R} \varphi_2) = free(\mathsf{X}\varphi) = Ag$
- $free((a, x \rhd \varphi)) = free(\varphi) \backslash \{a\}$

Definition 48 (Sentences). The sentences of SL are the formulas φ such that $\mathrm{FV}(\varphi) = \emptyset$ and $free(\varphi) = \emptyset$ (with $\mathrm{FV}(\varphi)$ defined similarly as in USL).

SL considers strategies, which can be seen as deterministic multi-strategies, in USL parlance.

Definition 49 (Strategies). Let $\mathcal{G} = \langle Ag, St, At, prop, Act, tr, s_o \rangle$ be a CGS. A *strategy* ς in $\mathcal{G}$ is a partial function which, given a track, yields an action or is not defined, *i.e.* $\varsigma : Track \nrightarrow Act$.

The set of multi-strategies in $\mathcal{G}$ is written $Strat_{\mathcal{G}}$ (the index will be omitted in case there is no ambiguity).

We also call *s-total* a strategy that is defined upon the set of tracks beginning at state s.

The contexts used in SL are also slightly different from those of USL.

Definition 50 (Strategy contexts for SL). Let $\mathcal{G} = \langle Ag, St, At, prop, Act, tr, s_o \rangle$ be a CGS. An *SL strategy context* $\overline{\chi}$ for $\mathcal{G}$ is a partial function from $Ag \cup X$ to *Strat*. It is *s-total* if it defines only s-total strategies and it is *complete* if $Ag \subseteq \mathrm{dom}(\overline{\chi})$.

As the transition function *tr* is defined over the set of decisions, a strategy context must be complete and s-total to determine a transition from a given state s. Conversely, a complete, s-total strategy context and a state determine an execution.

Definition 51 (Global translation). Let s be a state and $\overline{\chi}$ be a complete and s-total strategy context. Then, for any $n \in \mathbb{N}$, we write $\langle \overline{\chi}, s \rangle^n$ the pair made by the state s_n reached after n transitions under $\overline{\chi}$ and the complete and s_n-total strategy context $\overline{\chi}_n = \overline{\chi}^{s \cdots s_n}$. Formally:

- $\langle \overline{\chi}, s \rangle^0 = \langle \overline{\chi}, s \rangle$
- $\langle \overline{\chi}, s \rangle^{n+1} = \langle \overline{\chi}_{n+1}, s_{n+1} \rangle = \langle \overline{\chi}_n^{s_n \cdot tr(s_n, \delta_n)}, tr(s_n, \delta_n) \rangle$

where δ_n is the decision s.t. for every $a \in Ag, \delta_n(a) = \overline{\chi}_n(a)(s_n)$ and the notation $\overline{\chi}^{\theta}$, for $\overline{\chi}$ a strategy context and θ a track, is as in Def. 12.

Now we can define the satisfaction for SL.

Definition 52 (Satisfaction relation for SL). The satisfaction relation $\models_{\mathrm{SL}}$ is defined by induction over the structure of formulas: let $\mathcal{G}$ be a CGS, $\overline{\chi}$ a strategy context for $\mathcal{G}$ and s a state in it, then:

- $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} p$ iff $p \in \lambda(s)$, with $p \in At$
- $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} \neg\varphi$ iff it is not true that $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} \varphi$
- $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} \varphi_1 \wedge \varphi_2$ iff $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} \varphi_1$ and $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} \varphi_2$
- $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} \varphi_1 \vee \varphi_2$ iff $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} \varphi_1$ or $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} \varphi_2$
- $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} \langle\!\langle x \rangle\!\rangle \varphi$ iff there is an s-total strategy ς s.t. $\mathcal{G}, \overline{\chi}[x \mapsto \varsigma], s \models_{\mathrm{SL}} \varphi$
- $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} [\![x]\!] \varphi$ iff for every s-total strategy ς, $\mathcal{G}, \overline{\chi}[x \mapsto \varsigma], s \models_{\mathrm{SL}} \varphi$
- $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} (a, x)\varphi$ iff $\mathcal{G}, \overline{\chi}[a \mapsto \overline{\chi}(x)], s \models_{\mathrm{SL}} \varphi$
- If $\overline{\chi}$ is a complete strategy context:
  - $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} \mathsf{X}\varphi$ iff $\mathcal{G}, \langle \overline{\chi}, s \rangle^1 \models_{\mathrm{SL}} \varphi$
  - $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} \varphi_1 \mathsf{U} \varphi_2$ iff there is $i \in \mathbb{N}$ s.t. $\mathcal{G}, \langle \overline{\chi}, s \rangle^i \models_{\mathrm{SL}} \varphi_2$ and, for every index $j \in \mathbb{N}$ s.t. $0 \leqslant i < j$, $\mathcal{G}, \langle \overline{\chi}, s \rangle^j \models_{\mathrm{SL}} \varphi_1$
  - $\mathcal{G}, \overline{\chi}, s \models_{\mathrm{SL}} \varphi_1 \mathsf{R} \varphi_2$ iff for every index $i \in \mathbb{N}$, it holds that $\mathcal{G}, \langle \overline{\chi}, s \rangle^i \models_{\mathrm{SL}} \varphi_2$ or there is an index $j \in \mathbb{N}$ s.t. $0 \leqslant j \leqslant i, \mathcal{G}, \langle \overline{\chi}, s \rangle^j \models_{\mathrm{SL}} \varphi_1$

where, for every $e \in Var \cup Ag$, strategy ς and context $\overline{\chi}, \overline{\chi}[e \mapsto \varsigma]$ is as in Def.13. We also write $\mathcal{G}, s \models_{\mathrm{SL}} \varphi$ iff $\mathcal{G}, \overline{\chi}_\emptyset, s \models_{\mathrm{SL}} \varphi$ ($\overline{\chi}_\emptyset$ being the empty context).

Appendix C. Unravelling tree of a CGS, a context and a state

We define, for any CGS$^\infty$ $\mathcal{G}$, any context χ and any state s in $\mathcal{G}$, the *unravelling tree* of $(\mathcal{G}, \chi, s)$ (written $\mathcal{T}_{(\mathcal{G},\chi,s)}$) which is used in the proof of Theorem 3. Each state in $\mathcal{T}_{(\mathcal{G},\chi,s)}$ corresponds to a track θ in $\mathcal{G}$ starting by s. Its label contains both the valuation of last(θ) in $\mathcal{G}$ and an encoding of the decisions that can be made by agents in $\mathcal{G}$ after θ, given context χ.

Definition 53 (Unravelling tree). Let $\mathcal{G} = \langle Ag, St, At, prop, Act, tr, s_o\rangle$ be a CGS$^\infty$, $\chi = \langle \alpha, \gamma\rangle$ be a context for $\mathcal{G}$ and s be a state in $\mathcal{G}$. As for the definition of the Kripke structure $\mathcal{K}_\mathcal{G}$ obtained from $\mathcal{G}$ in Def. 39, we consider the fresh atomic proposition p_{contr} and the sets of fresh atomic propositions P_{St}, P_X, P_{Ag}. The *unravelling tree* of $(\mathcal{G}, \chi, s)$ is the $\mathcal{P}(At \cup \{p_{contr}\} \cup P_{St} \cup P_X \cup P_{Ag})$-labelled St-tree $\mathcal{T}_{(\mathcal{G},\chi,s)} = \langle T_{(\mathcal{G},\chi,s)}, \ell_{(\mathcal{G},\chi,s)}\rangle$ s.t.:

- $T_{(\mathcal{G},\chi,s)} = out^\infty(\chi, s)$
- for every $\theta \in T_{(\mathcal{G},\chi,s)}$,
  - for every $p \in At$, $p \in \ell_{(\mathcal{G},\chi,s)}(\theta)$ iff $p \in prop(\text{last}(\theta))$
  - $p_{contr} \in \ell_{(\mathcal{G},\chi,s)}(\theta)$ iff last(θ) $\neq s_\perp$ and there is $a \in \text{dom}(\gamma)$ s.t. $\bigcap_{x\in\gamma(a)} \alpha(x)(\theta) = \emptyset$
  - for every $s \in St$, $p_s \in \ell_{(\mathcal{G},\chi,s)}(\theta)$ iff $s = \text{last}(\theta)$
  - for every $p^x_{ac} \in P_X$, $p^x_{ac} \in \ell_{(\mathcal{G},\chi,s)}(\theta)$ iff $x \in \text{dom}(\alpha)$ and $ac \in \alpha(x)(\theta)$
  - for every $p^a_{ac} \in P_{Ag}$, $p^a_{ac} \in \ell_{(\mathcal{G},\chi,s)}(\theta)$ iff $a \in \text{dom}(\gamma)$, last(θ) $\neq s_\perp$, for every $a' \in \text{dom}(\gamma)$, $\bigcap_{x\in\gamma(a')} \alpha(x)(\theta) \neq \emptyset$ and for every $x \in \gamma(a)$, $ac \in \gamma(x)(\theta)$.

Let us express the following remarks about the labelling of an unravelling tree:

- The propositions in At are true exactly in the tracks ending by a state in which they are true in $\mathcal{G}$.
- The proposition p_{contr} expresses that a track θ is in its last possible state before going in $s_\perp$, *i.e.*, χ specifies a contradictory commitment for at least one of the agents.
- The propositions in P_{St} label any track θ with an encoding of its last state.
- For any variable x, the propositions in P^x_{Act} label any track θ with the actions specified by $\alpha(x)$ after θ.
- The propositions in P_{Ag} label any track θ with the actions that agents included in dom(γ) can play after θ according to χ, provided that none of them is committed to contradictory multi-strategies.