Uniform Strategies

Bastien Maubert , Sophie Pinchinat

RESEARCH REPORT
N° 8144
December 2012
Project-Team S4

ISSN 0249-6399   ISRN INRIA/RR--8144--FR+ENG

Uniform Strategies

Bastien Maubert[*] [†], Sophie Pinchinat[‡] [†]

Project-Team S4

Research Report n° 8144 — December 2012 — 30 pages

Abstract: We consider turn-based game arenas for which we investigate uniformity properties of strategies. These properties involve bundles of plays, that arise from some semantical motive. Typically, we can represent constraints on allowed strategies, such as being observation-based. We propose a formal language to specify uniformity properties and demonstrate its relevance by rephrasing various known problems from the literature. Note that the ability to correlate different plays cannot be achieved by any branching-time logic if not equipped with an additional modality, so-called R in this contribution. We also study an automated procedure to synthesize strategies subject to a uniformity property, which strictly extends existing results based on, say standard temporal logics. We exhibit a generic solution for the synthesis problem provided the bundles of plays rely on any binary relation definable by a finite state transducer. This solution yields a non-elementary procedure.

Key-words: game theory, temporal logic, epistemic logic, uniform strategies

[*] bastien.maubert@irisa.fr
[†] Team S4, Université de Rennes 1 / IRISA, Rennes, France
[‡] sophie.pinchinat@irisa.fr

RESEARCH CENTRE
RENNES – BRETAGNE ATLANTIQUE

Campus universitaire de Beaulieu
35042 Rennes Cedex

Stratégies uniformes

Résumé : Nous considérons des arènes de jeux pour lesquelles nous étudions des propriétés d'uniformité des stratégies. Ces propriétés font intervenir des ensembles de parties, qui émergent d'une quelconque motivation sémantique. Typiquement, nous pouvons représenter des contraintes sur les stratégies autorisées, comme par exemple être observationnelles. Nous proposons un language formel pour spécifier les propriétés d'uniformité et démontrons sa pertinence en reformulant divers problèmes connus de la littérature. Noter que la capacité à corréler différentes parties ne peut être obtenue par aucune logique du temps arborescent à moins de l'équiper d'une modalité supplémentaire, appelée R dans cette contribution. Nous étudions aussi une procédure de synthèse automatique de stratégies soumises à une propriété d'uniformité, qui étend strictement les résultats existants basés sur la logique temporelle standard. Nous proposons une solution générique pour le problème de la synthèse dans le cas où les ensembles de parties sont caractérisés par une relation binaire définissable par un transducteur fini. Cette solution donne une procédure non–élémentaire.

Mots-clés : théorie des jeux, logique temporelle, logique épistémique, stratégies uniformes

1 Introduction

In extensive (finite or infinite duration) games, the arena is represented as a graph whose vertices denote positions of players and whose paths denote plays. In this context, a strategy of a player is a mapping prescribing to this player which next position to select provided she has to make a choice at this current point of the play. As mathematical objects, strategies can be seen as infinite trees those of which are obtained by pruning the infinite unfolding of the arena according to the selection prescribed by this strategy; outcomes of a strategy are therefore the branches of the trees.

Strategies of players are not arbitrary in general, since players aim at achieving some objectives: in classic game theory with finite-duration plays, the reasonable *rationality assumption* leads players to play in such a way that they maximize their pay-off. More recently, (infinite-duration) game models have been intensively studied for their applications in computer science [AG11] and logic [GTW02]. First, infinite-duration games provide a natural abstraction of computing systems' non-terminating interaction [AHK02] (think of a communication protocol between a printer and its users, or control systems). Second, infinite-duration games naturally occur as a tool to handle logical systems for the specification of non-terminating behaviors, such as for the propositional μ-calculus [EJ91], leading to a powerful theory of automata, logics and infinite games [GTW02] and to the development of algorithms for the automatic verification ("model-checking") and synthesis of hardware and software systems.

Additionally, the cross fertilization of multi-agent systems and distributed systems theories has led to equip logical systems with additional modalities, such as epistemic ones, to capture uncertainty [Sat77, Leh84, FHV91, PR85, LR86, HV89], and more recently, these logical systems have been adapted to game models in order to reason about knowledge, time and strategies [vdHW03, JH04, DEG10]. The whole picture then becomes intricate, mainly because time and knowledge are essentially orthogonal, yielding a complex theoretical universe to reason about. In order to understand to which extent knowledge and time are orthogonal, the angle of view where strategies are infinite trees is helpful: Time is about the *vertical* dimension of the trees as it relates to the ordering of encountered positions along plays (branches) and to the branching in the tree. On the contrary, Knowledge is about the *horizontal* dimension, as it relates plays carrying, e.g., the same information.

As far as we know, this horizontal dimension, although extensively studied when interpreted as knowledge or observation [AVW03, vdHW03, JH04, Ben05, PR05, CDHR06, AČC07, DEG10], has not been addressed in its generality. In this paper, we aim at providing a unified setting to handle it. We introduce the generic notion of *uniformity properties* and associated so-called *uniform strategies* (those satisfying uniformity properties). Some notions of "uniform" strategies have already been used, e.g., in the setting of strategic logics [VB01, Ben05, JH04] and in the evaluation game of Dependence Logic [Vää07], which both fall into the general framework we present here.

We have chosen to tell our story in a simple framework where games are described by two-player turn-based arenas in which all information is put inside the positions, and not on the edges. However, the entire theory can be adapted to more sophisticated models, *e.g.* with labels on edges, multi-players, concurrent games, ... Additionally, although uniformity properties can be described in a set-theoretic framework, we have chosen to use a logical formalism which can be exploited to address fundamental automated techniques such as the verification of uniformity properties and the synthesis of uniform strategies – arbitrary uniformity properties are in general hopeless for automation. The formalism we use combines the Linear-time Temporal Logic LTL [GPSS80] and a new modality R (for "for all related plays"), the semantics of which is given by a binary relation between plays. Modality R generalizes the knowledge operator K of [HV89] for the epistemic relations of agents in Interpreted Systems. The semantic binary relations between plays are very little constrained: they are not necessarily equivalences, to capture, *e.g.* plausibility (pre)orders one finds in doxastic logic [Hin62], neither are they knowledge-based, to capture particular strategies in games where epistemic aspects are irrelevant. Formulas of the logic are interpreted over outcomes of a strategy. The R modality allows to universally quantify over all plays that are in relation with the current play. Distinguishing between the universal quantification over all plays in the game and the universal quantification over all the outcomes in the strategy tree yields two kinds of uniform strategies: the *fully-uniform strategies* and the *strictly-uniform strategies*.

As announced earlier, we illustrate the suitability of our notions by borrowing many frameworks from the literature: strategies for games with imperfect information, games with opacity conditions, the non-interference properties of computing systems, diagnosability of discrete-event systems (with a proposal for a formal definition of *prognosis*), and finally the evaluation game for Dependence Logic. Proofs of Section 3 are omitted due to lack of space, but they are quite simple. Through these examples we show that both notions, strict uniformity and full uniformity, are relevant and incomparable. These examples also demonstrate that defining uniformity properties with our formal language is convenient and intuitive. There are even more instances of uniform strategies in the literature, but the numerous examples we give here are already convincing enough to justify the relevance of the notion.

Next we turn to the automated synthesis of uniform strategies. For this purpose, we unsurprisingly restrict to finite arenas and to binary relations between plays that are finitely representable: we use *finite state transducers* [Ber79], an adequate device to characterize a large class of binary relations between sequences of symbols, hence they can be used to relate sequences of positions, *i.e.* plays. Incidentally, all binary relations that are involved in the relevant literature seem to follow this restriction. In this context, we address the problem of *the existence of a fully-uniform strategy*: "given a finite arena, a finite state transducer describing a binary relation between plays, and a formula expressing a uniformity property, does there exist a fully-uniform strategy for Player 1?". We prove that this problem is decidable by designing an algorithm. This algorithm involves a non-trivial powerset construction from the arena and the finite state transducer. This construction needs being iterated a number of times that matches the maximum number of nested R modalities in the formula specifying the uniformity property. As each powerset construction is computed in exponential time, the overall procedure is non-elementary. Regarding the decision problem for the existence of a strictly-uniform strategy, its decidability is an open problem.

The paper is organized as follows. in Section 2 we set the mathematical framework: we introduce the notion of uniform strategies and we present the formal language to specify uniformity properties. Next, in Section 3, we expose a significant set of six instances of uniform strategy problems from the literature. Section 4 is dedicated to a short reminder about finite state transducers that are used in the strategy synthesis problem addressed and solved in Section 5. We finish by a discussion on the work done and perspectives in Section 6.

2 Uniform properties

In this section we define a very general notion of uniform strategies. We consider two-player turn-based games that are played on graphs with vertices labelled with propositions.These propositions represent the relevant information for the uniformity properties one wants to state. If the game models a dynamic system interacting with its environment, relevant information can be the value of some (Boolean) variables. If it models agents interacting in a network, interesting information can be the state of the communication channels. In games with imperfect information, it can be what action has just been played.

From now on and for the rest of the paper, we let AP be an infinite set of *atomic propositions*.

An *arena* is a structure $\mathcal{G} = (V, E, v_I, \ell)$ where $V = V_1 \uplus V_2$ is the set of *positions*, partitioned between positions of Player 1 (V_1) and those of Player 2 (V_2), $E \subseteq (V_1 \times V_2) \cup (V_2 \times V_1)$ is the set of *edges*, $v_I \in V$ is the *initial position* and $\ell : V \to \mathcal{P}(AP)$ is a *valuation function*, mapping each position to the finite set of atomic propositions that hold in this position.

For $v \in V$, $Traces_\omega(v) \subseteq vV^\omega$ is the set of *infinite traces* starting in v, *i.e.* the set of infinite paths $v_0v_1v_2\dots$ in the game graph (V, E), with $v_0 = v$, and similarly $Traces_*(v) \subseteq vV^*$ is the set of *finite traces* starting in v, *i.e.* the set of finite paths $v_0v_1\dots v_n$ in (V, E), with $v_0 = v$ and $n \geq 0$. We let $Traces_\omega = \cup_{v\in V} Traces_\omega(v)$ and $Traces_* = \cup_{v\in V} Traces_*(v)$. Typical elements of $Traces_\omega$ are π, π', and λ, λ' are typical elements of $Traces_*$. $Plays_\omega$ denotes $Traces_\omega(v_I)$ and $Plays_*$ denotes $Traces_*(v_I)$, respectively the set of infinite and finite plays in the game. We shall write ρ instead of λ to distinguish finite plays from other finite traces.

For an infinite trace $\pi = v_0v_1\dots$ and $i, j \in \mathbb{N}$, $\pi[i] := v_i$, $\pi[i, \infty] := v_iv_{i+1}\dots \in Traces_\omega$ and $\pi[i, j] := v_i \dots v_j \in Traces_*$. We will use similar notations for finite traces, and $|.| : Traces_* \cup Traces_\omega \to \mathbb{N} \cup \{\omega\}$ denotes the length of the trace. If $\lambda \in Traces_*$, we let $last(\lambda) := \lambda[|\lambda| - 1]$ be the last position of λ.

A *strategy* for Player k, $k \in \{1, 2\}$, is a partial function $\sigma : Plays_* \to V$ that assigns the next position to choose in every situation in which it is Player k's turn to play. In other words $\sigma(\rho)$ is defined if $last(\rho) \in V_k$. Let σ be a strategy for Player k. We say that a play $\pi \in Plays_\omega$ is *induced by* σ if for all $i \geq 0$ such that $\pi[i] \in V_1$, $\pi[i+1] = \sigma(\pi[0, i])$, and the *outcome of* σ, noted $\mathrm{Out}(\sigma) \subseteq Plays_\omega$, is the set of all infinite plays that are induced by σ.

We want to express properties of strategies that do not concern only single traces but rather sets of correlated traces. We first give a very abstract definition.

Definition 1 *Let $\mathcal{G}$ be an arena. A* uniformity property *$U \subseteq \mathcal{P}(Plays_\omega)$ is a set of sets of plays in $\mathcal{G}$.*

Definition 2 *Let $\mathcal{G}$ be an arena and U a uniformity property. A strategy σ is U-*uniform *if $Out(\sigma) \in U$.*

This definition gives an idea of the notion we want to capture, but first this set-theoretic definition is not very intuitive, and moreover it is so expressive that automatically handling this notion in its generality is hopeless. We therefore restrict the notion of uniform strategy by fixing a formal language to specify uniformity properties. As demonstrated in the next section, the language is powerful enough to capture plethora of instances from the literature.

The proposed language enables to express properties of the dynamics of plays, and resembles the Linear Temporal Logic (*LTL*) [GPSS80]. However, while *LTL* formulas are evaluated on individual plays (paths), we want here to express properties on "bundles" of plays. To this aim, we equip arenas with a binary relation between finite plays, and we enrich the logic with a modality R that quantifies over related plays: the intended meaning of $R\varphi$ is that φ holds in every related play.

For the general presentation of the logic, we do not make yet assumptions concerning the binary relation over plays, as opposed to Section 5 dedicated to decidability issues.

We now give the syntax and semantics of the language $\mathcal{L}$.

2.1 Syntax

The syntax of the language $\mathcal{L}$ is the following :

$$\mathcal{L}: \quad \varphi, \psi ::= p \mid \neg\varphi \mid \varphi \wedge \psi \mid \bigcirc\varphi \mid \varphi\,\mathcal{U}\,\psi \mid R\varphi$$

where p is in AP. As usual we will use the following notations : $true := p \vee \neg p$, $false := \neg true$, $\mathbf{F}\varphi := true\,\mathcal{U}\,\varphi$, $\mathbf{G}\varphi := \neg\mathbf{F}\neg\varphi$, and $\varphi\mathcal{W}\psi := \varphi\,\mathcal{U}\,\psi \vee \mathbf{G}\varphi$. In addition we will use the following notation: $\langle R\rangle\varphi := \neg R\neg\varphi$, and for a formula $\varphi \in \mathcal{L}$, $Sub(\varphi)$ denotes the set of all its subformulas.

The syntax of $\mathcal{L}$ is similar to that of linear temporal logic with knowledge [HV89]. However, we use R instead of the usual knowledge operator K to emphasize that though it has a strong epistemic flavour, notably in various application instances we present here, it need not be interpreted in terms of knowledge in general, but merely as a way to state properties of bundles of plays.

Definition 3 *For a formula $\varphi \in \mathcal{L}$, we define the R-*depth *of φ, denoted $d_R(\varphi)$, as the maximum number of nested R modalities in φ. For $n \geq 0$, let $\mathcal{L}_n = \{\varphi \in \mathcal{L} \mid d_R(\varphi) = n\}$ be the set of formulas of R-depth n.*

We note LTL the language $\mathcal{L}_n$ as it matches the syntax (and also the semantics) of the Linear-time Temporal Logic of [GPSS80].

2.2 Semantics

To give the semantics of $\mathcal{L}$ we take an arena $\mathcal{G} = (V, E, v_I, \ell)$ and a relation $\rightsquigarrow \subseteq Plays_* \times Plays_*$. A formula φ of $\mathcal{L}$ is evaluated at some point $i \in \mathbb{N}$ of a trace $\pi \in Plays_\omega$, within a *universe* $\Pi \subseteq Plays_\omega$. The semantics is given by induction over formulas.

$\Pi, \pi, i \models p$	if	$p \in \ell(\pi[i])$
$\Pi, \pi, i \models \neg\varphi$	if	$\Pi, \pi, i \not\models \varphi$
$\Pi, \pi, i \models \varphi \wedge \psi$	if	$\Pi, \pi, i \models \varphi$ and $\Pi, \pi, i \models \psi$
$\Pi, \pi, i \models \bigcirc\varphi$	if	$\Pi, \pi, i+1 \models \varphi$
$\Pi, \pi, i \models \varphi\,\mathcal{U}\,\psi$	if	there is $j \geq i$ such that $\Pi, \pi, j \models \psi$ and for all $i \leq k < j$, $\Pi, \pi, k \models \varphi$
$\Pi, \pi, i \models R\varphi$	if	for all $\pi' \in \Pi, j \in \mathbb{N}$ such that $\pi[0,i] \rightsquigarrow \pi'[0,j]$, $\Pi, \pi', j \models \varphi$

The LTL part is classic. $R\varphi$ is true at some point of a trace if φ is true in every related finite trace in the universe.

We will sometimes need to evaluate an LTL-formula φ in a position v of an arena, with the classic semantics that φ holds in v if it holds in every trace starting in v.

Formally, for an arena $\mathcal{G} = (V, E, v_I, \ell)$, a position $v \in V$ and a formula $\varphi \in$ LTL, we write

$$\mathcal{G}, v \models \varphi \text{ if } \pi, 0 \models \varphi \text{ for all } \pi \in Traces_\omega(v)$$

Here we can omit Π because the formula has no R modality.

Definition 4 *Given an arena $\mathcal{G} = (V, E, v_I, l)$, a* uniformity property *is a pair $(\rightsquigarrow, \varphi)$ where $\rightsquigarrow$ is a relation over $Plays_*$ and $\varphi \in \mathcal{L}$ is a formula.*

Now we define two notions of uniform strategies, which differ only in the universe the R modality quantifies over: $\mathrm{Out}(\sigma)$ or $Plays_\omega$ (with the latter, related plays not induced by the strategy also count). As we shall see in the examples of the next section, making a nuance is worthwhile.

Definition 5 *Let $\mathcal{G}$ be an arena and $(\rightsquigarrow, \varphi)$ be a uniformity property. A strategy σ for Player 1 is*

- $(\rightsquigarrow, \varphi)$-strictly-uniform *if for all $\pi \in Out(\sigma)$, $Out(\sigma), \pi, 0 \models \varphi$.*
- $(\rightsquigarrow, \varphi)$-fully-uniform *if for all $\pi \in Out(\sigma)$, $Plays_*, \pi, 0 \models \varphi$.*

The notion of fully-uniform strategy is in a sense weaker than the strictly-uniform one. Indeed, the fact that a particular play in an arena verifies a formula φ in the fully-uniform semantics, *i.e.* with $Plays_\omega$ and not $\mathrm{Out}(\sigma)$ as a universe, is independent of the strategy. Hence in the general definition of uniform strategies, a uniformity property U could be defined to be a set of plays instead of a set of sets of plays, and a strategy σ could be said to be uniform if $\mathrm{Out}(\sigma) \subseteq U$ instead of $\mathrm{Out}(\sigma) \in U$, it would still contain the notion of fully-uniform strategies. Strictly-uniform strategies could no longer fit in this definition though, as deciding whether a play verifies a formula in this semantics cannot be done without knowing the strategy. In this sense, strict uniformity is "stronger" than full uniformity. However, the notion of fully-uniform strategies still enables to characterize tree languages that are not even μ-calculus definable: indeed, as observed by [AČZ06], given a infinite tree, Property (*) that at every depth $d > 0$, there exist two nodes, one of which being labeled by, say p, and one of which not being labeled by p, cannot be ω-regular (a pumpimg lemma argument suffices). Hence this property cannot be defined by any μ-calculus formula. However, we are able to characterize arenas whose tree unfolding has this property: consider the equivalence relation $=_{length}$ which relates plays with equal length. One easily sees that the existence of a $(=_{length}, \mathbf{G}(\langle R\rangle p \wedge \langle R\rangle \neg p))$-fully-uniform strategy is equivalent to say that the tree unfolding of the arena has Property (*).

Remark also that the relation $\rightsquigarrow$ plays no role in Definition 5 if φ does not contain any R operator, hence it is a mere LTL formula. Notice that in this latter case, some standard ω-regular (winning) conditions can be expressed over plays. The extension to a more powerful logic, such as the full propositional μ-calculus, in order to capture all ω-regular properties is *a priori* possible. However, for the examples considered in Section 3 this full power is not needed.

3 Frameworks from the literature

In this section we demonstrate that the notion of uniform strategy of previous Section 2 enables to embed various problems from the literature, and in particular that it subsumes two existing notions of uniform strategies.

3.1 Games with imperfect information

Games with imperfect information, in general, are games in which some of the players do not know exactly what is the current position of the game. This can occur in real games, *e.g.* poker since one does not know what cards her opponents have in hands, but also in situations arising from computer science, like for example a program that observes or controls a system by means of a sub-part of its variables, the interface, while other variables remain hidden. One important aspect of imperfect-information games is that not every strategy is "playable". Indeed, a player who has imperfect information cannot follow a strategy in which different moves are chosen for situations that are indistinguishable to her. This is why strategies are required to choose moves uniformly over observationally equivalent situations. This kind of strategies is sometimes called *uniform strategies* in the community of strategic logics ([VB01, Ben05, JH04]), or *observation-based strategies* in the community of computer-science oriented game theory ([CDHR06]).

In games with imperfect information, the player's ability to remember what happened so far along a play is a key point to achieve a winning strategy, as opposed to *e.g.* perfect-information parity games, where memoryless strategies are sufficient. Moreover in an imperfect information configuration, it is necessary to define what situations are indistinguishable to the player, and this requires defining how much memory she has of what occurs in a play. It is therefore relevant under an imperfect information assumption to distinguish the *perfect recall* setting, as opposed to the *imperfect recall* one. In the former, the player remembers the whole history of the observation she had of a play, no matter how long it is, while in the latter the player forgets a part of the information. An agent with imperfect information can either be memoryless, *i.e.* he does not remember anything and takes his decisions only based on the current position, or have a bounded memory, but this case can be reduced to the memoryless case by putting the different possible configurations of his memory in the positions of the arena.

While games with imperfect information and perfect recall have been studied intensively [Rei84, CDHR06, BD08], the case of imperfect recall has received much less attention since paradoxes concerning the interpretation of such games were raised [PR97]. Nonetheless, relevant problems may be modeled with imperfect recall: typically, particular computing resources have very limited memory and cannot remember arbitrarily long histories.

In this subsection, we show that the notion of "uniform" or "observation-based" strategy can be easily embedded in our notion of uniform strategy, and this no matter the assumption made on the memory of the player.

We first consider two-player imperfect-information games as studied for example in [Rei84, CDHR06, BD08]. In these games, Player 1 only partially observes the positions of the game, such that some positions are indistinguishable to her, while Player 2 has perfect information (the asymmetry is due to the focus being on the existence of strategies for Player 1). Arenas are labelled directed graphs together with a finite set of *actions* Act, and in each round, if the position is a node v, Player 1 chooses an available action a, and Player 2 chooses a next position v' reachable from v through an a-labelled edge.

We equivalently define this framework in a manner that fits our setting by putting Player 1's actions inside the positions. We have two kinds of positions, of the form v and of the form (v, a). In a position v, when she chooses an action a, Player 1 actually moves to position (v, a), then Player 2 moves from (v, a) to some v'. So an imperfect-information game arena is a structure $\mathcal{G}_{imp} = (\mathcal{G}, \sim)$ where $\mathcal{G} = (V, E, v_I, \ell)$ is a two-player game arena with positions in V_1 of the form v and positions in V_2 of the form (v, a). For a position $(v, a) \in V_2$, we note $(v, a).act := a$. $E \subseteq V_1 \times V_2 \cup V_2 \times V_1$, $vE(v', a)$ implies $v = v'$, $v_I \in V_1$. We assume that $p_1 \in AP$ and for every action a in Act,, $p_a \in AP$. p_1 holds in positions belonging to Player 1, and p_a holds in positions of Player 2 where the last action chosen by Player 1 is a: $\ell(v) = \{p_1\}$ for $v \in V_1$, $\ell(v, a) = \{p_a\}$ for $(v, a) \in V_2$. Finally, $\sim \subseteq V_1^2$ is an observational equivalence relation on positions, that relates indistinguishable positions for Player 1. We define its extension to finite plays as the least relation $\approx$ such that $\rho \cdot v \approx \rho' \cdot v'$ whenever $\rho \approx \rho'$ and $v \sim v'$, and $\rho \cdot (v, a) \approx \rho' \cdot (v', a')$ whenever $\rho \approx \rho'$, $v \sim v'$ and $a = a'$.

We add the classic requirement that the same actions must be available in indistinguishable positions: for all $v, v' \in V_1$, if $v \sim v'$ then $vE(v, a)$ if, and only if, $v'E(v', a)$. In other words, if a player has different options, she can distinguish the positions.

Definition 6 *A strategy σ for Player* 1 *is* observation-based *if for all $\rho, \rho' \in v(V_2V_1)^*$, $\rho \approx \rho'$ implies $\sigma(\rho).act = \sigma(\rho').act$.*

We define the formula

$$\texttt{SameAct} := \mathbf{G}(p_1 \to \bigvee_{a \in Act} R\bigcirc p_a)$$

which expresses that whenever it is Player 1's turn to play, there is an action a that is played in all equivalent finite play.

Theorem 1 *A strategy σ for Player 1 is observation-based if, and only if, it is $(\approx, \texttt{SameAct})$-strictly-uniform.*

Here we have to make use of the notion of strict uniformity, and not the full uniformity. Indeed, after a finite play $\pi[0,i]$ ending in V_1, we want to enforce that the actions in the next positions of equivalent plays are the same only in those plays that are induced by the strategy we consider, and not in every possible play in the game. This would of course not hold as soon as several choices are possible in $\pi[i]$.

Also, it is interesting to see that this notion could also be embedded with a simpler formula by using a relation that is not an equivalence. Define $\rightsquigarrow$ as: $\pi[0,i] \rightsquigarrow \pi'[0,j]$ if $\pi[i] \in V_1$, $j = i+1$ and $\pi[0,i] \approx \pi'[0,i]$, and define the formula:

$$\texttt{SameAct'} := \mathbf{G} \bigvee_{a \in Act} Rp_a.$$

Then a strategy σ for Player 1 is observation-based if, and only if, it is $(\rightsquigarrow, \texttt{SameAct'})$-strictly-uniform.

In this version, the relation is not reflexive, in particular plays ending in V_2 are linked to no play, making $R\varphi$ trivially true for any φ. This is the reason why we no longer need to mark positions of V_1 with the proposition p_1 and test whether we are in V_1 before we ask for some p_a to hold in every reachable position.

Finally, notice that to embed the case of imperfect-recall for example, one would just have to replace $\approx$ with the appropriate relation.

3.2 Games with epistemic condition

Uniform strategies enable to express winning conditions that have epistemic features, the relevance of which is exemplified by the *games with opacity condition* studied in [MPB11]. In that case, R can represent a players' knowledge, or distributed knowledge between a group of players, or common knowledge, depending on the winning condition one wants to define.

Games with opacity condition are based on two-player imperfect-information arenas with a particular winning condition, called the *opacity condition*, which relies on the knowledge of the player with imperfect information. In such games, some positions are "secret" as they reveal a critical information that the imperfect-information player wants to know (in the epistemic sense).

More formally, assume that a proposition $p_S \in AP$ represents the secret. Let $\mathcal{G}_{inf} = (\mathcal{G}, \sim)$ be an imperfect-information arena as described in Section 3.1, with a distinguished set of positions $S \subseteq V_1$ that denotes the secret. Let $\mathcal{G} = (V, E, v_I, \ell)$ be the arena with $\ell^{-1}(\{p_S\}) = S$ (positions labeled by p_S are exactly positions $v \in S$). The *opacity winning condition* is as follows.

The *knowledge* or *information set* of Player 1 after a finite play is the set of positions that she considers possible according to the observation she has: let $\rho \in Plays_*$ be a finite play with $last(\rho) \in V_1$. The *knowledge* or *information set* of Player 1 after ρ is $I(\rho) := \{last(\rho') \mid \rho' \in Plays_*, \rho \approx \rho'\}$.

An infinite play is winning for Player 1 if there exists a finite prefix ρ of this play whose information set is contained in S, *i.e.* $I(\rho) \subseteq S$, otherwise Player 2 wins. Again, strategies for Player 1 are required to be observation-based. It can easily be shown that:

Theorem 2 *A strategy σ for Player 1 is winning if, and only if, σ is $(\approx, \mathbf{F}Rp_S)$-strictly-uniform. A strategy σ for Player 2 is winning if, and only if, σ is $(\approx, \mathbf{G}\neg Rp_S)$-fully-uniform.*

For the second statement of Theorem 2, we make use of the notion of full uniformity because we are interested in the knowledge of Player 1. We consider that she does not know what strategy Player 2 is playing, hence she may consider possible some plays that are observationally equivalent to her but not induced by this strategy.

On the other hand, for the first statement, we use strict uniformity but could have used full uniformity instead. Indeed, since the actions chosen by Player 1 are part of what she observes, she cannot consider possible a finite play that has not followed her strategy until the point i considered. In fact, if π is induced by σ and $\pi[0,i] \approx \pi'[0,i]$, then $\pi'[0,i]$ is also induced by σ. The future of the play may not follow sigma, but this is not a problem here. Indeed, the property that we consider on equivalent plays, p_S, does not depend on the future.

Notice that though we chose to illustrate with an example, any winning condition that could be expressed as a formula of the linear temporal logic with one knowledge operator would fit in our setting.

3.3 Non-interference

Non-interference, as introduced by [GM82], is a property evaluated on labelled transition systems which handle Boolean variables. Such systems are tuples $(S, \mathcal{I}, \mathcal{O}, \delta, s_I, \mathit{Output})$ where S is the set of states, $\mathcal{I} = H \uplus L$ is a set of Boolean input variables partitioned into *high security variables* H and *low security variables* L, $\mathcal{O}$ is the set of Boolean output variables, $\delta : S \times 2^{\mathcal{I}} \to S$ is the transition function that maps each pair of state and input variables valuation[1] to a next state, s_I is the initial state, and $\mathit{Output} : S \to 2^{\mathcal{O}}$ is the output function that represents a mapping of states onto valuations of the Boolean output variables. We extend the transition function δ to $S \times (2^{\mathcal{I}})^* \to S$ as expected: $\delta(s, \epsilon) = s$ and $\delta(s, ua) = \delta(\delta(s, u), a)$.

We define the L-equivalence, $\sim_L$ over $(2^{\mathcal{I}})^*$ by $u \sim_L u'$ whenever u and u' have the same length and they coincide on the values of the low security input variables, *i.e.* for all $1 \leq i \leq \mathit{length}(u)$, for all $l \in L$, $l \in u(i) \Leftrightarrow l \in u'(i)$. Given an infinite sequence of inputs $w \in (2^{\mathcal{I}})^\omega$, we abuse notation by writing $\mathit{Output}(w)$ for the infinite sequence of output variables valuations encountered in the states along the execution of the system on input w. A system $(S, \mathcal{I}, \mathcal{O}, \delta, s_I, \mathit{Output})$ verifies the *non-interference property* if for any two finite sequences of inputs $w, w' \in (2^{\mathcal{I}})^*$, $w \approx_L w'$ implies $\mathit{Output}(w) = \mathit{Output}(w')$. In other words, the valuations of high security variables have no consequence on the observation of the system.

A first natural problem is to decide the non-interference property of a system. A second more general problem is a control problem: we want to decide whether there is a way of restricting the set of input valuations along the executions, or equivalently to control the environment, so that the system is non-interfering. By constraining the applied restriction to be trivial, the former problem is a particular case of the latter. We can encode the control problem in our setting.

Let $\mathit{Sys} = (S, \mathcal{I}, \mathcal{O}, \delta, s_I, \mathit{Output})$ be an instance of the problem, and write Σ for $2^{\mathcal{I}}$ with typical elements $a, b, \ldots$ Without loss of generality, we can assume that Sys is *complete*: every input valuation yields a transition. We define a two-player game arena that simulates the system, in which Player 1 fixes the *environment*, *i.e.* a subset of the possible inputs in the current state, and Player 2 chooses a particular input among those. More formally, let $\mathcal{G}_{\mathit{Sys}} = (V, E, v_I, \ell)$, with $V = V_1 \uplus V_2$, $V_1 = (\Sigma \uplus \{\epsilon\}) \times S$ and $V_2 = S \times 2^{\Sigma}$. A position $(a, s) \in V_1$ denotes a situation where the system reaches state s by an a-transition, and $(s, A) \in V_2$ denotes a situation where in state s, the set of possible inputs is A. The set of edges E of the arena is the smallest set such that $(a, s)E(s, A)$ for all $s \in S$, $a \in \Sigma$ and $A \subseteq \Sigma$, and $(s, A)E(a, \delta(s, a))$ whenever $s \in S$ and

[1] we classically confuse valuations over a set B of Boolean variables with elements of 2^B.

$a \in A$. The initial position of the arena is $v_I = (\epsilon, s_I)$, and by assuming that $\{p_o \mid o \in 2^{\mathcal{O}}\} \subseteq AP$, we set $\ell(a, s) = \ell(s, A) = \{p_{Output(s)}\}$.

By writing ι for the canonical projection from $V_1 \cup V_2$ onto $2^{\mathcal{I}}$ (that is $\iota(\epsilon, s_I) = \iota(s, A) = \epsilon$ and $\iota(a, s) = a$) and by extending ι to finite plays as expected, we let $\rho \equiv_L \rho'$ hold whenever $\iota(\rho) \sim_L \iota(\rho')$.We now define the formula

$$\texttt{SameOutput} := \mathbf{G} \bigwedge_{p_o \in AP} (p_o \to Rp_o)$$

which captures the property that the valuations of output variables along $\equiv_L$-equivalent executions of the system coincide, and we can establish the following.

Theorem 3 *There is a one-to-one correspondence between* $(\equiv_L, \texttt{SameOutput})$*-strictly-uniform strategies of Player* 1 *and the controllers which ensure the non-interference property of the system.*

In particular, the trivial strategy of Player 1*, where from any position* (a, s) *she chooses to move to* (s, Σ)*, is* $(\equiv_L, \texttt{SameOutput})$*-strictly-uniform if, and only if, the system has the non-interference property.*

Here we have to use the strict uniformity as we only want to consider the executions of the machine allowed by the control represented by the strategy.

Notice that in order to make this control problem more realistic, one would seek a maximal permissive strategy/controller so that environments as "large" as possible are computed, but this is out of the scope of the paper.

3.4 Diagnosis and Prognosis

Diagnosis has been intensively studied, in particular by the discrete-event systems community (see for example [SSL+95, YL02, CL99]). Informally, in this setting, a discrete-event system is *diagnosable* if any occurrence of a faulty event during an execution is eventually detected. More formally, *diagnosability* is a property of discrete-event systems which are structures of the form $Sys = (S, \Sigma, \Sigma_o, \Delta, s_I, F)$, with S the set of states, Σ the set of events, $\Sigma_0 \subseteq \Sigma$ the *observable* events, $\Delta \subseteq S \times \Sigma \times S$ the transition relation, s_I the initial state and $F \subseteq S$ the faulty states; we assume that once a faulty state is reached, only faulty states can be reached (the fault is persistent). We can rephrase this problem in our setting, with a single player simulating the system. Notice that since there is only one player, a strategy defines a unique infinite play.

Here we assume that $p_f \in AP$ represents the fact of being faulty. Let $\mathcal{G}_{Sys} = (V, E, v_I, \ell)$, with $V_1 = \emptyset$, $V_2 = (\Sigma \uplus \{\epsilon\}) \times S$, $(a, s)E(b, s')$ whenever $(s, b, s') \in \Delta$, $v_I = (\epsilon, s_I)$, and $\ell(a, s) = \{f\}$ if $s \in F$, $\emptyset$ otherwise. We write $\rho \equiv_{\Sigma_o} \rho'$ whenever the sequences of observable events underlying ρ and ρ' are the same (these sequences are obtained from the sequences of positions in the play: for each position of the form (a, s), keep its letter a if $a \in \Sigma_o$, and delete it otherwise). In this game Player 1 never plays, all she does is look at Player 2 simulate the system. There is only one strategy for her, which is to do nothing, and all possible plays, representing all possible executions of Sys, are in the outcome of this strategy.

Theorem 4 *Sys is diagnosable if, and only if, Player 1 has a* $(\equiv_{\Sigma_o}, \mathbf{F}p_f \to \mathbf{F}Rp_f)$*-fully-uniform strategy in* $\mathcal{G}_{Sys}$*.*

Here again, since the outcome of the only possible strategy for Player 1 is the whole set of plays, we could equivalently choose full or strict uniformity.

Prognosis is a companion of diagnosis, but focuses on the ability to predict that a fault will occur. Prognosability-like properties can be defined in our setting. As an example, we aim at saying that a system is *prognosable* whenever the fact that a fault occurs in a system is known at least one step in advance. We define the following formula, which means that either a fault never occurs, or it occurs but we know it one step before it does.

$$\texttt{Prognose} := (\neg p_f)\mathcal{W}(\neg p_f \wedge R\bigcirc p_f))$$

Using the same framework as for diagnosis, we can propose:

Definition 7 *A system Sys is* prognosable *if there is a* $(\equiv_{\Sigma_o}, \texttt{Prognose})$*-fully-uniform strategy for Player 1 in* $\mathcal{G}_{Sys}$.

3.5 Dependence Logic

Dependence Logic is a flourishing topic introduced recently by Väänänen [Vää07]. It extends first order logic by adding atomic dependence formulas $\texttt{dep}(t_1, \ldots, t_n)$, which express functional dependence of the term t_n on the terms $t_1, \ldots, t_{n-1}$. A dependence atom $\texttt{dep}(x_0, x_1)$ can be interpreted as "the value of x_1 depends only on the value of x_0", or "the value of x_1 is fully determined by the value of x_0". Evaluating a dependence between terms on a single assignment of the free variables is meaningless: in order to tell whether t depends on t', one must vary the values of t' and see how the values of t are affected. This is why a formula of Dependence Logic is evaluated on a first-order model $\mathcal{M}$ and a *set of assignments* for the free variables, called a *team*. If t is a term, $\mathcal{M}$ a model and s an assignment for the free variables in t, we note $[\![t]\!]_s^{\mathcal{M}} \in M$ the interpretation of t in the model $\mathcal{M}$ with the assignment s.

Dependence Logic is inspired by Independence Friendly logic (IF logic), a logic defined by Hintikka and Sandu [HS89]. Van Benthem gives in [Ben05] an imperfect-information evaluation game for IF logic, using a notion of uniform strategy that corresponds with the classic notion of imperfect-information or observation-based strategy.

For Dependence Logic, an evaluation game is also given in [Vää07]. It is presented as a game with imperfect information, because strategies must verify some “uniformity constraint”, which makes the game undetermined. However, the notion of uniform strategy in these games is not defined as “playing uniformly in positions of an information set”, but rather as “playing such that, when positions of an information set are reached, some property uniformly holds in these positions”. For this reason it is a notion of uniform strategy different from the one used by Van Benthem in [VB01, Ben05], and this game is not a game with imperfect information *stricto sensu.*

We present the evaluation game, the uniformity requirement and then show that this notion fits in our setting.

Let Φ be a sentence (formula with no free variable) of Dependence Logic in negation normal form, *i.e.* only atomic formulas can be negated, and let $\mathcal{M}$ be a first order model. $\mathcal{G}^{\mathcal{M}}(\varphi)$ is a two player game between Player 1 and Player 2; positions are of the form (φ, n, s), where φ is a subformula of Φ, n is the position in Φ of the first symbol of φ and s is an assignment whose domain contains the free variables of φ. The index n is used to decide, given two positions containing the same dependence atom, whether they are the same *syntactic* subformulas of φ or not. For a subformula φ, $len(\varphi)$ is the number of symbols in φ. The game starts in position $(\varphi, 1, \emptyset)$ and the rules are as follows:

position $(t_1 = t_2, n, s)$:	if $[\![t_1]\!]_s^{\mathcal{M}} = [\![t_2]\!]_s^{\mathcal{M}}$, Player 1 wins.
position $(\neg(t_1 = t_2), n, s)$:	if $[\![t_1]\!]_s^{\mathcal{M}} = [\![t_2]\!]_s^{\mathcal{M}}$, Player 2 wins
position $(Rt_1 \dots t_m, n, s)$:	if $[\![R]\!]^{\mathcal{M}}[\![t_1]\!]_s^{\mathcal{M}} \dots [\![t_m]\!]_s^{\mathcal{M}}$, Player 1 wins.
position $(\neg Rt_1 \dots t_m, n, s)$:	if $[\![R]\!]^{\mathcal{M}}[\![t_1]\!]_s^{\mathcal{M}} \dots [\![t_m]\!]_s^{\mathcal{M}}$, Player 2 wins.
position $(\mathtt{dep}(t_1, \dots, t_m), n, s)$:	Player 1 wins.
position $(\neg\mathtt{dep}(t_1, \dots, t_m), n, s)$:	Player 2 wins.
position $(\varphi \vee \psi, n, s)$:	Player 1 chooses between position (φ, n, s) and $(\psi, n + 1 + \text{len}(\varphi), s)$.
position $(\varphi \wedge \psi, n, s)$:	Player 2 chooses between position (φ, n, s) and $(\psi, n + 1 + \text{len}(\varphi), s)$.
position $(\exists x\varphi, n, s)$:	Player 1 chooses a value $a \in M$ and moves to $(\varphi, n + 2, s(x \mapsto a))$
position $(\forall x\varphi, n, s)$:	Player 2 chooses a value $a \in M$ and moves to $(\varphi, n + 2, s(x \mapsto a))$

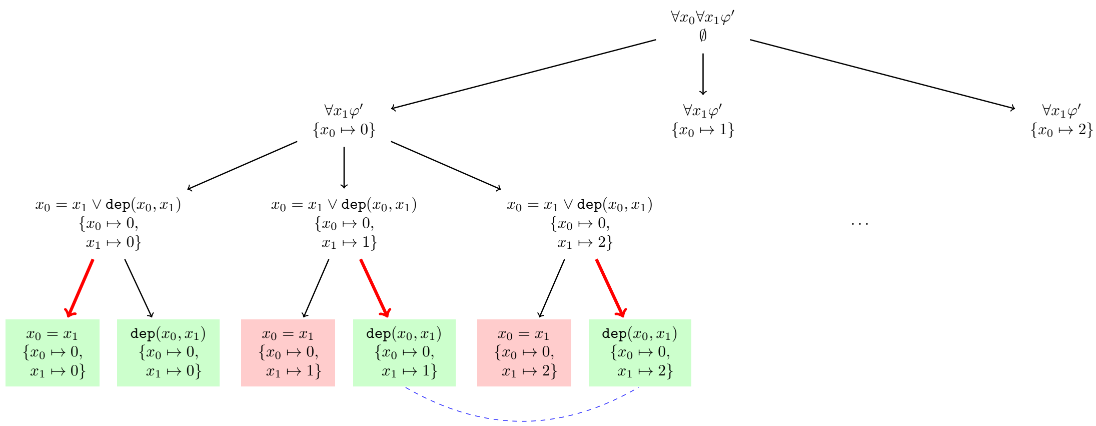

Figure 1: Evaluation game for $\forall x_0 \forall x_1(x_0 = x_1 \vee \mathtt{dep}(x_0, x_1))$ with $M = \{0, 1, 2\}$

Figure 1 represents (a part of) the game for the evaluation of the Dependence Logic formula $\forall x_0 \forall x_1(x_0 = x_1 \vee \mathtt{dep}(x_0, x_1))$ on a model $\mathcal{M}$ with domain $M = \{0, 1, 2\}$. This formula is not true in this model. Indeed, intuitively, if there are at least three elements in the domain, it is not because x_0 and x_1 do not have the same value that the value of x_0 determines the value of x_1: there remain two possibilities for the value of x_1 that are not the one of x_0. So there should not be a winning strategy for Player 1 for the evaluation game to be correct.

In the first two rounds, Player 2 chooses a value for each of the universally quantified variables x_0 and x_1. Then Player 1 chooses a disjunct and we reach atomic formulas. Green positions are winning for Player 1, red ones are winning for Player 2. The red arrows indicate a strategy for Player 1 (we focus on the subtree for $x_0 = 0$, we assume that Player 1 plays the same way in the others). We see that this strategy is winning for Player 1, while the formula is not true in the model.

The problem comes from the fact that, as said earlier, a dependence atom must not be evaluated on a single assignment but on a *set* of assignments, a team. In the evaluation game, the team in which a dependence atom $\mathtt{dep}(t_1, \dots, t_n)$ should be evaluated is the set of assignments in leafs that contain $\mathtt{dep}(t_1, \dots, t_n)$ and are reached by the strategy. In the example, the assignments in the two leafs linked with the dashed line, $\{x_0 \mapsto 0, x_1 \mapsto 1\}$ and $\{x_0 \mapsto 0, x_1 \mapsto 2\}$, are thus part of the team in which $\mathtt{dep}(x_0, x_1)$ should be evaluated (there are more in the two other subtrees not shown here). Then we see that this strategy should not be allowed as while both leaves agree on the value of x_0, they do not agree on the value of x_1. This observation leads to defining a certain notion of uniform strategy.

A strategy σ for Player 1 is *uniform* in the sense of [Vää07] if, for every two finite plays $\rho, \rho' \in \mathrm{Out}(\sigma)$ such that $last(\rho) = (\mathtt{dep}(t_1, \ldots, t_m), n, s, 1)$ and $last(\rho') = (\mathtt{dep}(t_1, \ldots, t_m), n, s', 1)$ contain the same (syntactically speaking) atomic dependence subformula, if s and s' agree on $t_1, \ldots, t_{m-1}$, then they also agree on t_m. Then we have the expected property that a sentence φ of Dependence Logic is true in a model $\mathcal{M}$ if Player 1 has a winning *uniform* strategy in $\mathcal{G}^{\mathcal{M}}(\varphi)$.

We characterize uniform strategies in the sense of [Vää07] as uniform strategies in our sense. The game described above easily fits in our setting (we add loops on terminal positions so as to obtain infinite plays). Let Φ be a sentence of Dependence Logic, and $\mathcal{M}$ be a finite model. We call $G^{\mathcal{M}}_{\Phi} = (V, E, v_I)$ the evaluation game defined above. For each object $a \in M$ of the domain we use one atomic proposition p_a, and we also use the proposition p_d to mark positions that contain dependence atoms. So we assume that $\{p_a \mid a \in M\} \uplus \{p_d\} \subset AP$, and we define $\mathcal{G}^{\mathcal{M}}_{\Phi} = (V, E, v_I, \ell)$, where the valuation ℓ is as follows :

$$\begin{array}{rcl} \ell(\mathtt{dep}(t_1, \ldots, t_m), n, s) & = & \{p_a, p_d\} \text{ with } a = [\![t_m]\!]^{\mathcal{M}}_s \\ \ell(\ \_\ , n, s) & = & \emptyset \end{array}$$

We define the equivalence relation $\simeq$ on finite plays as the smallest reflexive relation such that if there is $\varphi = \mathtt{dep}(t_1, \ldots, t_m)$ and n s.t $last(\rho) = (\varphi, n, s)$, $last(\rho') = (\varphi, n, s')$, and $[\![t_i]\!]^{\mathcal{M}}_s = [\![t_i]\!]^{\mathcal{M}}_{s'}$ for $i = 1, \ldots, n-1$, then $\rho \simeq \rho'$. Now we define the formula

$$\mathtt{AgreeOnLast} := \mathbf{G}(p_d \to \bigvee_{a \in M} Rp_a)$$

which expresses that whenever the current position contains a dependence atom $\mathtt{dep}(t_1, \ldots, t_m)$, it agrees with all equivalent finite plays on some value a for t_m. Since equivalent plays are those ending in a position that has the same dependence atom and agrees on the first $m-1$ terms, it is easy to prove:

Theorem 5 *A strategy σ for Player 1 in $G^{\mathcal{M}}_{\Phi}$ is uniform if, and only if, it is $(\simeq, \mathtt{AgreeOnLast})$-strictly-uniform in $\mathcal{G}^{\mathcal{M}}_{\Phi}$.*

In this example again, the strict uniformity is needed, as we only want to catch leaves that are hit by the strategy. Also, note that here $\simeq$ is an equivalence because we take the reflexive closure of some transitive and reflexive relation. But as for observation based strategies, if we did not take the reflexive closure, we could avoid using the proposition p_d and use the simpler formula

$$\mathtt{AgreeOnLast'} := \mathbf{G} \bigvee_{a \in M} Rp_a$$

Indeed the relation would not be reflexive: in particular plays not ending in a dependence atom would not be linked to any play, and $R\varphi$ would trivially hold for any φ in these plays; this is why testing whether we are in a dependence atom before enforcing that some p_a must hold everywhere would no longer be needed.

3.6 Dependence logic and games with imperfect information

As we said, the evaluation game for Dependence Logic presented in the previous subsection is said to be a game with imperfect information. We do not agree, because the difference between games with perfect information and games with imperfect information (at least in the perfect recall setting, it is not as clear otherwise, see [PR97]) lies in the fact that in the latter, some finite plays are related, in the sense that they are indistinguishable to one of the players, and this player must behave the same way in these related situations. Concerning the evaluation game for Dependence Logic, the difference with perfect-information games is that some plays are related, those ending in positions bound to the same atomic dependence formula $\mathtt{dep}(t_1, \ldots, t_n)$ with valuations agreeing on $t_1, \ldots, t_{n-1}$, and the valuations in these related positions must agree on t_n. So it is not that players should behave the same way in related situations, but rather that the players should *have behaved* in such a way that the valuations for t_n are the same in related situations.

But it is true that there is a similarity between these two constraints on allowed strategies, as shown by looking at the formulas of the uniformity properties capturing observation-based strategies (`SameAct`) and uniform strategies in the sense of Dependence Logic (`AgreeOnLast`):

$$\mathtt{SameAct} = \mathbf{G}(p_1 \rightarrow \bigvee_{a \in Act} R\bigcirc p_a) \quad \text{and} \quad \mathtt{AgreeOnLast} = \mathbf{G}(p_d \rightarrow \bigvee_{a \in M} Rp_a)$$

In the first case, the same thing must *happen* in equivalent situations, whereas in the second case, the same thing must *hold* in equivalent situations.

The resemblance is even more striking if we take the second versions:

$$\mathtt{SameAct'} = \mathbf{G} \bigvee_{a \in Act} Rp_a \quad \text{and} \quad \mathtt{AgreeOnLast'} = \mathbf{G} \bigvee_{a \in M} Rp_a$$

Neither semantics games for Dependence Logic are games with imperfect information in the classical sense, nor games with imperfect information can be easily described using the uniform strategy notion of [Vää07], but both can be characterized in a very similar way with our notion of uniform strategies.

4 Regular relations

The previous examples from the literature all fall into a particular class of binary relations, so-called *regular*[2] relations. They are characterized by some finite state machines called *transducers* [Ber79]. One way to see a transducer is to picture a nondeterministic automaton with two tapes, an *input* tape and an *output* tape. The transducer reads an input finite word on its input tape and writes out a finite word on its output tape. Notice that this machine is in general nondeterministic so that it may have several outputs for a given input word. Hence, transducers define binary relations.

Definition 8 *A* Finite State Transducer *(FST) is a tuple $T = (Q, \Sigma, \Gamma, q_i, Q_F, \Delta)$, where Q is a finite set of states, Σ is the* input alphabet*, Γ is the* output alphabet*, $q_i \in Q$ is a distinguished* initial state*, $Q_F \subseteq Q$ is a set of* accepting states*, and $\Delta \subset Q \times (\Sigma \cup \{\epsilon\}) \times (\Gamma \cup \{\epsilon\}) \times Q$ is a finite set of* transitions*.*

Intuitively, $(q, a, b, q') \in \Delta$ means that the transducer can move from state q to state q' by reading a and writing b (both possibly ϵ). We also define the *extended transition relation* Δ^*, which is the smallest relation such that:

- for all $q \in Q$, $(q, \epsilon, \epsilon, q) \in \Delta^*$, and
- if $(q, w, w', q') \in \Delta^*$ and $(q', a, b, q'') \in \Delta$, then $(q, w \cdot a, w' \cdot b, q'') \in \Delta^*$.

[2] Actually, the genuine vocabulary is "rational relations", but we prefer to use "regular relations" to avoid any misleading terminology in the context of game theory and rational players.

In the following, for $q, q' \in Q$, $w \in \Sigma^*$ and $w' \in \Gamma^*$, notation $q \text{-}[w/w'] \rightarrow q'$ means $(q, w, w', q') \in \Delta^*$.

Definition 9 *Let $T = (Q, \Sigma, \Gamma, q_i, Q_F, \Delta)$ be an FST. The* relation recognized by T *is*

$$[T] := \{(w, w') \mid w \in \Sigma^*, w' \in \Gamma^*, \exists q \in Q_F,\ q_i \text{-}[w/w'] \rightarrow q\}.$$

In other words, a couple (w, w') is in the relation recognized by T if there is an accepting execution of T that reads w and outputs w'.

Definition 10 *Let Σ and Γ be two alphabets. A binary relation $\rightsquigarrow \subseteq \Sigma^* \times \Gamma^*$ is* regular *if it is recognized by an FST.*

We now recall some basic properties of regular relations that will be useful later on. The first property regards intersection (regular relations are not closed under intersection in general) and the second property regards composition. The interested reader is referred to [Ber79] for technical details.

Property 6 *Let Σ and Γ be two alphabets. Let $\rightsquigarrow \subseteq \Sigma^* \times \Gamma^*$ be a regular relation, and let L and L' be two regular languages over Σ and Γ respectively. Then $\rightsquigarrow \cap (L \times L')$ is also a regular relation.*

Let $\Sigma, \Sigma', \Sigma''$ be three alphabets. Let $\rightsquigarrow_1 \subseteq \Sigma^* \times \Sigma'^*$ and $\rightsquigarrow_2 \subseteq \Sigma'^* \times \Sigma''^*$ be two binary relations.

Definition 11 *The* composition *of $\rightsquigarrow_1$ and $\rightsquigarrow_2$ is $\rightsquigarrow_1 \circ \rightsquigarrow_2 \subseteq \Sigma^* \times \Sigma''^*$, defined by:*

$$\rightsquigarrow_1 \circ \rightsquigarrow_2 = \{(\rho, \rho'') \mid \exists \rho' \in \Sigma'^*, \rho \rightsquigarrow_1 \rho' \text{ and } \rho' \rightsquigarrow_2 \rho''\}$$

Property 7 *[EM65] If $\rightsquigarrow_1$ and $\rightsquigarrow_2$ are two regular relations recognized respectively by T_1 and T_2 , then $\rightsquigarrow_1 \circ \rightsquigarrow_2$ is also regular, and the composition of the transducers $T_1 \circ T_2$ recognizes $\rightsquigarrow_1 \circ \rightsquigarrow_2$.*

We close this section by taking two examples of binary relations involved in Section 3 and showing that they are regular. In fact, one can check that they all are.

Example 1 *We first consider an example induced by the imperfect-information setting of Section 3.1. Let $\mathcal{G}_{imp} = (V, E, v_I, \ell, \sim)$ be an imperfect-information game arena as described in Section 3.1. Relation $\approx \subseteq Plays_*^2$ is an* observational equivalence *over plays, generated by the equivalence $\sim$ between positions. Consider the FST T_{obs} depicted in Figure 2, with a unique initial state (ingoing arrow) that is also the final state (two concentric circles). It reads an input letter (a position) and outputs any position that is $\sim$-equivalent to it. This FST recognizes a relation $\simeq$ over $V^* \times V^*$, such that $\approx = \simeq \cap Plays_*^2$. Because $Plays_*$ is a regular language (one can see $\mathcal{G}$ as a finite state automaton), Proposition 6 gives that $\simeq \cap Plays_*^2$ is also a regular relation, and in fact $\mathcal{G} \times T_{obs} \times \mathcal{G}$ is a transducer that precisely recognizes $\approx$.*

Example 2 *Consider another binary relation that is also an equivalence, but induced by some alphabetic morphism $h : V \rightarrow \mathcal{O} \cup \{\epsilon\}$: two plays ρ and ρ' are equivalent, written $\rho \equiv_{\mathcal{O}} \rho'$, whenever $h(\rho) = h(\rho')$. This example generalizes the one of Section 3.4 where the alphabetic morphism is a mere projection. In order to draw an FST for the relation $\equiv_{\mathcal{O}}$, we need to fix*

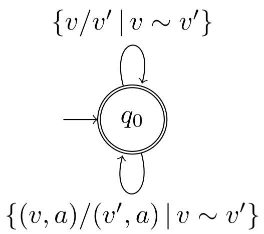

Figure 2: T_{obs}, an FST for the equivalence relation $\simeq$ of Sections 3.1 and 3.2.

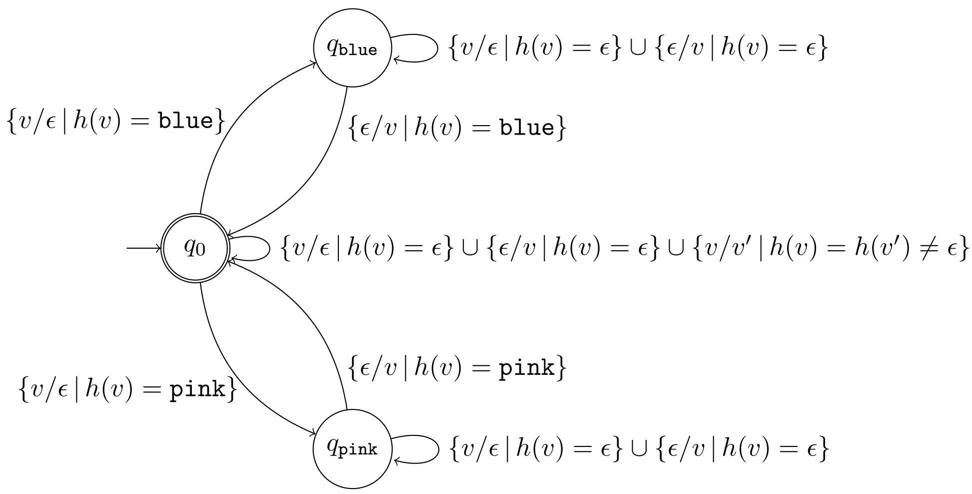

Figure 3: $T_{\mathcal{O}}$, an FST for the equivalence relation $\equiv_{\mathcal{O}}$.

the set $\mathcal{O}$. Assume $\mathcal{O}$ has only two elements `blue` *and* `pink` *so that any position in the game is either observed as $h(v) = \texttt{blue}$ or $h(v) = \texttt{pink}$ or unobserved (the case $h(v) = \epsilon$). The FST $T_{\mathcal{O}}$ that recognizes $\equiv_{\mathcal{O}}$ is drawn in Figure 3. Once again, one should take the product of $T_{\mathcal{O}}$ with the game arena to restrict the relation to $Plays_* \times Plays_*$. Remark that contrary to the case of equivalence $\approx$ in Figure 2, the equivalence $\equiv_{\mathcal{O}}$ does not preserve the length of plays.*

Example 1 is an example of a simple transducer, but in fact all relations of Section 3 can be recognized by the transducer given in Example 2, for an appropriate alphabetic morphism. In the next section, we will only consider regular relations over plays, i.e. the relation $\rightsquigarrow$ in the model is recognized by some FST.

5 Automated synthesis of fully-uniform strategies

In this section, we study the problem of synthesizing fully-uniform strategies. We restricy finite arenas. Motivated by Section 4, we only consider regular relations, and as a consequence we always assume that the semantic relation between plays is described by a finite state transducer T. Still, when it is clear from the context, we write $\rightsquigarrow$ instead of $[T]$. Also, because we only consider full uniformity and no longer the strict one, it is understood in the semantics of a formula that the universe Π is the set of plays of the considered game, hence we omit it when it is clear from the context. Furthermore, we will sometimes make the semantic relation between plays explicit. All these conventions yield a notation of the form $\pi, i \models_{\rightsquigarrow} \varphi$ meaning that $Plays_\omega, \pi, i \models \varphi$ with the R-modality semantics based on the binary relation $\rightsquigarrow$.

Finally, in this section, the size $|\mathcal{G}|$ of an arena $\mathcal{G}$ is the number of positions, the size $|T|$ of a transducer T is the number of states, and $|\varphi|$ is the size of formula φ.

Definition 12 *For each $n \in \mathbb{N}$, we define the decision problem* FUS_n *by:*

$$\mathrm{FUS}_n := \{(\mathcal{G}, T, \varphi) \mid \quad \mathcal{G} \textit{ is an arena and } ([T], \varphi) \textit{ is a uniformity property}^3 \textit{ with } \varphi \in \mathcal{L}_n \textit{ such that there exists a } ([T], \varphi)\textit{-fully-uniform strategy for Player 1 in } \mathcal{G}\}$$

The fully-uniform strategy problem *is* $\mathrm{FUS} := \bigcup_{n \in \mathbb{N}} \mathrm{FUS}_n$

For an instance $(\mathcal{G}, T, \varphi)$ of FUS, its size is defined as $|(\mathcal{G}, T, \varphi)| := |\mathcal{G}| + |T| + |\varphi|$.

Theorem 8 FUS_n *is in* 2-EXPTIME *for $n \leq 2$, and in n*-EXPTIME *for $n > 2$.*

Corollary 9 *The fully-uniform strategy problem is decidable.*

The rest of this section is dedicated to the proof of Theorem 8. We describe a powerset construction for a new arena (Section 5.1) and a way to lift the semantic relation between plays to this powerset construction (Section 5.2). Next we show how to exploit this construction to reduce membership in FUS_{n+1} to membership in FUS_n (Section 5.3).

5.1 Powerset arena

In games with imperfect information, the information set of a player after a finite play is the set of positions that are consistent with what she has observed. We define a similar notion in our setting, and we show that the regularity assumption on the relation is sufficient to compute information sets and build a powerset construction arena in which formulas of the kind $R\varphi$ where $\varphi \in \mathrm{LTL}$ can be evaluated positionally.

Definition 13 *Let $\mathcal{G}$ be an arena, $\rightsquigarrow \subseteq Plays_*^2$ and $\rho \in Plays_*$. The* information set *after the finite play ρ is the set of terminating positions of related finite plays. Formally, $I(\rho) = \{v' \mid \exists \rho' \cdot v' \in Plays_*,\ \rho \rightsquigarrow \rho' \cdot v'\}$.*

For an arena $\mathcal{G}$ and a transducer T over $Plays_*$, we construct a powerset arena $\widehat{\mathcal{G}}$ in which formulas of the form $R\varphi$ can be evaluated positionally when $\varphi \in \mathcal{L}_0$.

Unlike classic powerset constructions [Rei84, CDHR06], in our setting, the new information set after a move in the game cannot be computed knowing only the previous information set and the new position. To compute information sets, we need to simulate the nondeterministic execution of T, taking as an input the sequence of positions played and writing as output the related plays. This is why in our construction, positions do not contain directly information sets, but rather we add in the positions of the game sufficient information on the current configuration of the transducer.

More precisely, two things are necessary: the set of states the transducer may be in after reading the sequence of positions played so far, plus for each of these states the set of possible last positions written on the output tape (because of nondeterminacy, different executions can end up in configurations with same state but different last letters on the output tape). We only need to remember the last letter on the output tape, and not the whole tape, because the information set which we aim at computing is just the set of the *last* positions of related plays.

So positions are of the form $(v, S, Last)$, where $S \subseteq Q$ is the set of possible current states of T, and $Last : S \to \mathcal{P}(V)$ associates to a state $q \in S$ the set of the possible last positions on the output tape of T if the current state is q. The transitions in this arena follow the ones in $\mathcal{G}$, we just have to maintain the additional information about the configuration of the transducer. In order to define the initial position $\widehat{v}_I = (v_I, S_I, Last_I)$ of $\widehat{\mathcal{G}}$ we need to simulate the execution of T starting from its initial state and reading v_I. To do so, we introduce an artificial position $\widehat{v}_{-1}$ that initializes the transducer before reading the first position of a play. Concretely, $\widehat{v}_{-1} = (v_{-1}, S_{-1}, Last_{-1})$, with $v_{-1} \notin V$ a fresh position, $S_{-1} = \{q_i\}$ because before starting the transducer is in its initial state, and $Last_{-1}(q_i) = \emptyset$ because nothing is written on the output tape.

Definition 14 *Let $\mathcal{G} = (V, E, v_I, \ell)$ be an arena and $T = (Q, V, q_i, Q_F, \Delta)$ be an FST such that $[T] \subseteq Plays_*^2$. We define the arena $\widehat{\mathcal{G}} = (\widehat{V}, \widehat{E}, \widehat{v}_I, \widehat{\ell})$ by:*

- $\widehat{V} = V \times \mathcal{P}(Q) \times (Q \to \mathcal{P}(V)) \uplus \{\widehat{v}_{-1}\}$
- $(u, S, Last) \mathrel{\widehat{\to}} (v, S', Last')$ *if*
  - $u = v_{-1}$ *and* $v = v_I$, *or* $u \to v$,
  - $S' = \{q' \mid \exists q \in S, \exists \lambda' \in V^*,\ q \xrightarrow{[v/\lambda']} q'\}$ *and*
  - $Last'(q') = \{v' \mid \exists q \in S, \exists \lambda' \in V^*, q \xrightarrow{[v/\lambda' \cdot v']} q',$ *or* $q \xrightarrow{[v/\epsilon]} q'$ *and* $v' \in Last(q)\}$
- $\widehat{v}_I$ *is the only* $\widehat{v} \in \widehat{V}$ *such that* $\widehat{v}_{-1} \mathrel{\widehat{\to}} \widehat{v}$.
- $\widehat{\ell}(\widehat{v}) = \ell(v)$ *if* $\widehat{v} = (v, S, Last)$.

Notice that $\widehat{\mathcal{G}}$ has size $|\widehat{\mathcal{G}}| = O(|\mathcal{G}| \times 2^{|T|} \times 2^{|T||\mathcal{G}|})$.

Regarding the definition of transitions, the first point means that transitions in $\widehat{E}$ follow those in E, except for the only transition leaving $\widehat{v}_{-1}$, that is used to define $\widehat{v}_I$. The second point for the definition of S' expresses that when we move from u to v in $\mathcal{G}$, we give v as an input to the transducer. So the set of states the transducer can be in after this move is exactly the set of states that can be reached from one of the previous possible states by reading v and writing some sequence of positions (possibly ϵ). We use the notation λ because without loss of generality one could assume that the transducer can only output sequences of positions that form a valid path in the game graph. But it is not important here, the assumption that $[T] \subseteq Plays^2_*$ is sufficient. Finally, the third point for the definition of $Last'$ captures that if some position v' is at the end of the output tape after the transducer read v and reached q', it is either because while reading v the last letter it wrote is v', or it wrote nothing and v' was already the end of the output tape before reading v.

To finish with, $\widehat{v}_I$ is the only successor of $\widehat{v}_{-1}$, and the valuation of a position in the powerset construction is the valuation of the underlying position in the original arena.

Remark 1 *We need to clarify the following definitions:*

- $S' = \{q' \mid \exists q \in S, \exists \lambda' \in V^*,\ q \xrightarrow{[v/\lambda']} q'\}$ *and*
- $Last'(q') = \{v' \mid \exists q \in S, \exists \lambda' \in V^*, q \xrightarrow{[v/\lambda' \cdot v']} q',$ *or* $q \xrightarrow{[v/\epsilon]} q'$ *and* $v' \in Last(q)\}$

Indeed, in each one, there can be infinitely many such λ' because of transitions that read nothing on the input tape. Still, S' and $Last'$ can be computed in linear time in the size of Δ. To do so, for each q in S, we compute $S_{q,v} = \{(q', v') \mid \exists \lambda' \in V^*, q \xrightarrow{[v/\lambda' \cdot v']} q',$ or $q \xrightarrow{[v/\epsilon]} q'$ and $v' \in Last(q)\}$. S' and $Last'$ can be easily reconstructed from $\cup_{q \in S} S_{q,v}$. For $q \in S$, computing $S_{q,v}$ can be done by depth-first search, by first reading v and then only ϵ, and remembering the last output. The search can be stopped when we reach a state that has already been visited.

Let us take an arena $\mathcal{G}$ and an FST T such that $[T] \subseteq Plays^2_*$. For a position $\widehat{v}$ of $\widehat{\mathcal{G}}$, we will access the different components of the position with the notations $\widehat{v}.v, \widehat{v}.S, \widehat{v}.Last$.

There is a natural mapping $f : Plays_\omega \to \widehat{Plays_\omega}$: for an infinite play $\pi \in \mathcal{G}$, we define $f(\pi)$ as the only play $\widehat{\pi}$ in $\widehat{Plays_\omega}$ such that $\widehat{\pi}[0].v \cdot \widehat{\pi}[1].v \cdot \widehat{\pi}[2].v \ldots = \pi$. This is well defined because from a position $\widehat{u} = (u, S, Last)$ in $\widehat{V}$, for $v \in V$ such that $u \to v$, there is a unique move $\widehat{u} \widehat{\to} \widehat{v}$ such that $\widehat{v}.v = v$. It is easy to see that f is a bijection. From now on we will slightly abuse notations: for a play $\pi \in Plays_\omega$, $\widehat{\pi}$ will denote $f(\pi)$, and for $\widehat{\pi} \in \widehat{Plays_\omega}$, π will denote $f^{-1}(\widehat{\pi})$. *idem* for finite plays.

Definition 15 *Let $\widehat{v} = (v, S, Last) \in \widehat{V}$ be a position in the powerset construction. Its* local information set *$\widehat{v}.I$ is defined by:*

$$\widehat{v}.I := \bigcup_{q \in S \cap Q_F} Last(q)$$

Definition 15 means that a position v' is in the information set after a play $\widehat{\rho}$ if and only if there is an execution of the transducer on the word seen so far that terminates in an accepting state with v' the last output. Actually, the local information sets correspond to the real information sets as expressed by the following proposition.

Proposition 10 *For all $\widehat{\rho} \in \widehat{Plays_*}$, $last(\widehat{\rho}).I = I(\rho)$.*

The rest of this subsection is dedicated to the proof of Proposition 10.

Lemma 11 *Let $\widehat{\rho} \in \widehat{Plays_*}$, and let $\widehat{v} := last(\widehat{\rho})$. Then, $\widehat{v}.S = \{q \mid \exists \lambda' \in V^*, q_i \xrightarrow{[\rho/\lambda']} q\}$, and for each $q \in \widehat{v}.S$, $\widehat{v}.Last(q) = \{v' \mid \exists \lambda' \in V^*, q_i \xrightarrow{[\rho/\lambda' \cdot v']} q\}$.*

Proof The proof is by induction on $\widehat{\rho}$.

Case $\widehat{v}_I$. We note $\widehat{v}_I = (v_I, S_I, Last_I)$, and start with the left-right inclusions for both equalities. Let $q' \in S_I$ and $v' \in Last_I(q)$. By definition of $Last_I$ and S_I there is a q in $S_{-1} = \{q_i\}$ (so $q = q_i$) and a $\lambda' \in V^*$ such that $q_i \text{ -}[v_I/\lambda']\!\!\rightarrow q'$, which proves the first inclusion. Since $Last_{-1}(q_i) = \emptyset$ by definition, the case $\lambda' = \epsilon$ is not possible. So there exists λ'' such that $\lambda' = \lambda'' \cdot v'$, which gives us the second inclusion.

The proofs for the two right-left inclusions are straightforward applications of the definitions of S_I and $Last_I$.

Case $\widehat{\rho} \cdot \widehat{u} \cdot \widehat{v}$, $\widehat{\rho} \in \widehat{Plays_*} \cup \{\epsilon\}$. We note $\widehat{u} = (u, S, Last)$ and $\widehat{v} = (v, S', Last')$. For the two left-right inclusions, let $q' \in S'$ and $v' \in Last'(q')$. By definition of S' and $Last'$ there is a q in S and a $\lambda'_1 \in V^*$ such that $q \text{ -}[v/\lambda'_1]\!\!\rightarrow q'$, and either $\lambda'_1 = \lambda' \cdot v'$ for some λ', or $\lambda'_1 = \epsilon$ and $v' \in Last(q)$. By induction hypothesis, we have that $S = \{q \mid \exists \lambda' \in V^*, q_i \text{ -}[\rho \cdot u/\lambda']\!\!\rightarrow q\}$, so there exists $\lambda'_2 \in V^*$ such that $q_i \text{ -}[\rho \cdot u/\lambda'_2]\!\!\rightarrow q$, and by transitivity, $q_i \text{ -}[\rho \cdot u \cdot v/\lambda'_2 \cdot \lambda'_1]\!\!\rightarrow q'$. This proves the first left-right inclusion. For the second one we split the two cases for λ'_1.

- If $\lambda'_1 = \lambda'_3 \cdot v'$ for some λ'_3, then by transitivity we have $q_i \text{ -}[\rho \cdot u \cdot v/\lambda'_2 \cdot \lambda'_3 \cdot v']\!\!\rightarrow q'$, which proves the second left-right inclusion.
- If $\lambda'_1 = \epsilon$, then $v' \in Last(q)$. By induction hypothesis there is some λ'_3 such that $q_i \text{ -}[\rho \cdot u/\lambda'_3 \cdot v']\!\!\rightarrow q$. By transitivity we obtain $q_i \text{ -}[\rho \cdot u \cdot v/\lambda'_3 \cdot v']\!\!\rightarrow q'$, which also proves the second left-right inclusion.

Now for the first right-left inclusion, take q' and λ' such that $q_i \text{ -}[\rho \cdot u \cdot v/\lambda']\!\!\rightarrow q'$. Necessarily there exist λ'_1, λ'_2 and q such that $q_i \text{ -}[\rho \cdot u/\lambda'_1]\!\!\rightarrow q$, $q \text{ -}[v/\lambda'_2]\!\!\rightarrow q'$ and $\lambda'_1 \cdot \lambda'_2 = \lambda'$. By induction hypothesis $q \in S$, so by definition of S', $q' \in S'$. For the second right-left inclusion, take $q' \in S'$, and take v' and λ' such that $q_i \text{ -}[\rho \cdot u \cdot v/\lambda' \cdot v']\!\!\rightarrow q'$. Again, necessarily there exist λ'_1, λ'_2 and q such that $q_i \text{ -}[\rho \cdot u/\lambda'_1]\!\!\rightarrow q$, $q \text{ -}[v/\lambda'_2]\!\!\rightarrow q'$ and $\lambda'_1 \cdot \lambda'_2 = \lambda' \cdot v'$. By induction hypothesis $q \in S$. We distinguish two cases.

- If $\lambda'_2 = \epsilon$, then $\lambda'_1 = \lambda' \cdot v'$, hence $q_i \text{ -}[\rho \cdot u/\lambda' \cdot v']\!\!\rightarrow q$. By induction hypothesis, $v' \in Last(q)$, so by definition of $Last'$, because $q \in S$ and $q \text{ -}[v/\epsilon]\!\!\rightarrow q'$, we obtain $v' \in Last'(q')$.
- If $\lambda'_2 = \lambda'_3 \cdot v'$ for some λ'_3, then by definition of $Last'$, because $q \in S$, we have $v' \in Last'(q')$.

This finishes the induction. $\square$

We can now terminate the proof of Proposition 10: Let $\widehat{\rho} \in \widehat{Plays_*}$, and let $\widehat{v} = \text{last}(\widehat{\rho})$ be of the form $\widehat{v} = (v, S, Last)$. We remind that (Definition 13) $I(\rho) = \{v' \in V \mid \exists \rho' \cdot v' \in Plays_*,\ \rho \rightsquigarrow \rho' \cdot v'\}$.

We start with the left-right inclusion. Let $v' \in \widehat{v}.I$. By definition, $\widehat{v}.I = \bigcup_{q \in S \cap Q_F} Last(q)$, so $v' \in Last(q)$ for some $q \in S \cap Q_F$. By Lemma 11, there exists $\lambda' \in V^*$ such that $q_i \text{ -}[\rho/\lambda' \cdot v']\!\!\rightarrow q$, and because $q \in Q_F$, we have that $(\rho, \lambda' \cdot v') \in [T]$, which implies that $\rho \rightsquigarrow \lambda' \cdot v'$. Since $\rightsquigarrow \subseteq Plays_*^2$, $\lambda' \cdot v' \in Plays_*$, hence $v' \in I(\rho)$.

For the right-left inclusion, take $v' \in I(\rho)$. There exists ρ' such that $\rho' \cdot v' \in Plays_*$ and $\rho \rightsquigarrow \rho' \cdot v'$. By definition of T, there exists $q \in Q_F$ such that $q_i \text{ -}[\rho/\rho' \cdot v']\!\!\rightarrow q$. By Lemma 11, $q \in S$, and $v' \in Last(q)$. Since $q \in S \cap Q_F$, $v' \in \widehat{v}.I$.

5.2 Lifting transducers

Let $\mathcal{G}$ be an arena, T an FST such that $[T] \subseteq Plays_*^2$, and let $\widehat{\mathcal{G}} = \text{Power}(\mathcal{G}, T)$. We describe how to build a transducer $\widehat{T}$ that lifts $[T] \subseteq Plays_* \times Plays_*$ to $\widehat{Plays_*} \times \widehat{Plays_*}$.

We note $T{\downarrow}$ for the deterministic transducer that computes f, the bijection that maps a play $\widehat{\rho} \in \widehat{Plays_*}$ to the underlying play $\rho \in Plays_*$, and $T{\uparrow}$ for the deterministic transducer that computes f^{-1}. Both are easily built from $\widehat{\mathcal{G}}$, and $|T{\downarrow}| = |T{\uparrow}| = O(|\widehat{\mathcal{G}}|)$.

Definition 16 *The* lift *of transducer T is $\widehat{T} = T{\downarrow} \circ T \circ T{\uparrow}$.*

Notice that $|\widehat{T}| = O(|\widehat{\mathcal{G}}| \times |T| \times |\widehat{\mathcal{G}}|)$.

In the following we let $\rightsquigarrow\, = [T]$ and $\widehat{\rightsquigarrow} = [\widehat{T}]$. The following proposition follows directly from the definitions of $\widehat{\mathcal{G}}$ and $\widehat{T}$:

Proposition 12 *For every $\varphi \in \mathcal{L}$, $\pi \in Plays_\omega$, $i \geq 0$,*

$$\pi, i \models_{\rightsquigarrow} \varphi \text{ iff } \widehat{\pi}, i \models_{\widehat{\rightsquigarrow}} \varphi$$

5.3 R-elimination

We establish that given an instance of FUS_{n+1}, we can build in exponential space and time an equivalent instance of FUS_n.

Proposition 13 *For all instance $(\mathcal{G}, T, \varphi)$ of FUS_{n+1}, there exists an instance $(\mathcal{G}', T', \varphi')$ of FUS_n computable in time exponential in $|(\mathcal{G}, T, \varphi)|$ such that:*

- $(\mathcal{G}, T, \varphi) \in \text{FUS}_{n+1}$ *iff* $(\mathcal{G}', T', \varphi') \in \text{FUS}_n$
- $|\mathcal{G}'| = O(2^{(|\mathcal{G}|+|T|)^2})$
- $|T'| = O(2^{O(|\mathcal{G}|+|T|)^2})$
- $|\varphi'| = O(|\varphi|)$

The rest of the section is dedicated to the proof of Proposition 13. Let $(\mathcal{G}, T, \varphi)$ be an instance of FUS_{n+1}.

Lemma 14 *Let $\pi \in Plays_\omega$, $i \geq 0$, and φ be an LTL−formula.*

$$\pi, i \models_{\rightsquigarrow} R\varphi \text{ iff } \mathcal{G}, u \models \varphi \text{ for all } u \in I(\pi[0, i]).$$

Proof We start with the left-right implication. Suppose that $\pi, i \models R\varphi$ holds, and take $u \in I(\pi[0, i])$. We need to prove that $\mathcal{G}, u \models \varphi$. To do so, we take $\pi' \in Traces_\omega(u)$ an infinite trace starting in u and we prove that $\pi', 0 \models \varphi$. Since $u \in I(\pi[0, i])$, by definition of the information set, there exists $\rho \cdot u \in Plays_*$ such that $\pi[0, i] \rightsquigarrow \rho \cdot u$. We let $j = |\rho|$ and $\pi'' = \rho \cdot \pi'$ be such that $\pi''[j] = u$. Clearly, $\pi'' \in Plays_\omega$, and $\pi[0, i] \rightsquigarrow \pi[0, j]$. Since $\pi, i \models R\varphi$ holds, we have that $\pi'', j \models \varphi$. And because $\varphi \in \text{LTL}$, the fact that it holds at some point of a trace only depends on the future of this point, hence $\pi''[j, \infty], 0 \models \varphi$, *i.e.* $\pi', 0 \models \varphi$.

For the right-left implication, suppose that $\mathcal{G}, u \models \varphi$ for all $u \in I(\pi[0, i])$, and take $\pi' \in Plays_\omega$, $j \geq 0$ such that $\pi[0, i] \rightsquigarrow \pi'[0, j]$. We have that $\pi'[j] \in I(\pi[0, i])$, so $\mathcal{G}, \pi'[j] \models \varphi$. Because $\pi'[j, \infty]$ is in $Traces_\omega(\pi'[j])$, we have that $\pi'[j, \infty], 0 \models \varphi$, hence $\pi', j \models \varphi$. $\square$

Lemma 15 *Let $\widehat{\pi} \in \widehat{Plays_\omega}$, $i \geq 0$, and let φ be an LTL−formula.*

$$\widehat{\pi}, i \models_{\widehat{\sim}} R\varphi \textit{ iff } \mathcal{G}, u \models \varphi \textit{ for all } u \in \widehat{\pi}[i].I.$$

Proof Let $\widehat{\pi} \in \widehat{Plays_\omega}$ and $i \geq 0$. By Proposition 12, for any $\varphi \in LTL$, $\widehat{\pi}, i \models_{\widehat{\sim}} R\varphi$ iff $\pi, i \models_{\sim} R\varphi$, and by Lemma 14, $\pi, i \models_{\sim} R\varphi$ iff $\mathcal{G}, u \models \varphi$ for all $u \in I(\pi[0,i])$. Now, by Proposition 10, $\widehat{\pi}[i].I = I(\pi[0,i])$, which concludes the proof. □

We now define how formulas of the kind $R\varphi$ can be replaced by new atomic propositions, and how positions of the powerset arena can be marked with these new propositions: To an arena $\mathcal{G}$, a formula $\varphi \in \mathcal{L}$ and a subformula $R\psi \in \mathcal{L}_1 \cap Sub(\varphi)$ (if any), we associate a fresh atomic proposition $p_{R\psi}$ that occurs neither in $\mathcal{G}$ nor in φ.

Definition 17 *For $\varphi \in \mathcal{L}_{n+1}$, we define $\widehat{\varphi} := \varphi[p_{R\psi}/R\psi \mid R\psi \in \mathcal{L}_1 \cap Sub(\varphi)]$.*

Example 3 $\widehat{R\bigcirc Rq} = R\bigcirc p_{Rq}$

Definition 18 *For an instance $(\mathcal{G}, T, \varphi)$ of* FUS_{n+1}*, we define the instance $\widehat{(\mathcal{G}, T, \varphi)}$ of* FUS_n *as $(\mathcal{G}', T', \varphi')$ by:*

- *If $\widehat{\mathcal{G}} = (\widehat{V}, \widehat{E}, \widehat{v_I}, \widehat{\ell})$, then $\mathcal{G}' = (\widehat{V}, \widehat{E}, \widehat{v_I}, \widehat{\ell}')$, with*
$$\widehat{\ell}'(\widehat{v}) = \widehat{\ell}(\widehat{v}) \cup \{p_{R\psi} \mid R\psi \in \mathcal{L}_1 \cap Sub(\varphi) \textit{ and } \forall u \in \widehat{v}.I, \mathcal{G}, u \models \psi\}.$$
- $T' = \widehat{T}$
- $\varphi' = \widehat{\varphi}$

From now on, for an instance $(\mathcal{G}, T, \varphi)$ of FUS_{n+1}, we abuse notation by writing $\widehat{\mathcal{G}} = (\widehat{V}, \widehat{E}, \widehat{v_I}, \widehat{\ell})$ for the *modified* powerset construction of Definition 18.

Lemma 16 *Take an instance $(\mathcal{G}, T, \varphi)$ of* FUS_{n+1}*, and let $(\widehat{\mathcal{G}}, \widehat{T}, \widehat{\varphi}) = \widehat{(\mathcal{G}, T, \varphi)}$. Then for all $\pi \in Plays_\omega$ and $i \geq 0$,*

$$\pi, i \models_{\sim} \varphi \textit{ iff } \widehat{\pi}, i \models_{\widehat{\sim}} \widehat{\varphi}$$

Proof By Proposition 12, $\pi, i \models_{\sim} \varphi$ iff $\widehat{\pi}, i \models_{\widehat{\sim}} \varphi$, so it only remains to show that $\widehat{\pi}, i \models_{\widehat{\sim}} \varphi$ iff $\widehat{\pi}, i \models_{\widehat{\sim}} \widehat{\varphi}$. We prove it by induction on φ. The cases $\varphi = p, \varphi = \neg\psi, \varphi = \psi \vee \psi', \varphi = \bigcirc\psi, \varphi = \psi\, \mathcal{U}\, \psi'$ are trivial. It remains to consider the case $\varphi = R\psi$, which decomposes into two subcases depending on $d_R(\psi)$:

- If $d_R(\psi) > 0$, then $\widehat{\varphi} = R\widehat{\psi}$. We then have:

$$\begin{aligned}
\widehat{\pi}, i \models_{\widehat{\sim}} R\psi &\text{ iff } \forall \widehat{\pi}', j \text{ s.t. } \widehat{\pi}[0,i] \mathrel{\widehat{\sim}} \widehat{\pi}'[0,j],\ \widehat{\pi}', j \models_{\widehat{\sim}} \psi \\
&\text{ iff } \forall \widehat{\pi}', j \text{ s.t. } \widehat{\pi}[0,i] \mathrel{\widehat{\sim}} \widehat{\pi}'[0,j],\ \widehat{\pi}', j \models_{\widehat{\sim}} \widehat{\psi} && \text{(by induction hypothesis)} \\
&\text{ iff } \widehat{\pi}, i \models_{\widehat{\sim}} R\widehat{\psi}
\end{aligned}$$

- If $d_R(\psi) = 0$, that is $\psi \in LTL$, then $\widehat{\varphi} = p_{R\psi}$.

$$\begin{aligned}
\widehat{\pi}, i \models_{\widehat{\sim}} R\psi &\text{ iff } \mathcal{G}, u \models \psi \text{ for all } u \in \widehat{\pi}[i].I && \text{(by Lemma 15)} \\
&\text{ iff } p_{R\psi} \in \widehat{\ell}(\widehat{\pi}[i]) && \text{(by Definition 18)} \\
&\text{ iff } \widehat{\pi}, i \models_{\widehat{\sim}} p_{R\psi}
\end{aligned}$$

$\square$

We can now achieve the proof of Proposition 13. Take an instance $(\mathcal{G}, T, \varphi)$ of FUS_{n+1}. We show that $(\widehat{\mathcal{G}}, \widehat{T}, \widehat{\varphi}) = (\widehat{\mathcal{G}, T, \varphi})$ is a good candidate. Notice that the natural bijection between $Plays_*$ and $\widehat{Plays_*}$ induces a bijection between strategies σ in $\mathcal{G}$ and strategies $\widehat{\sigma}$ in $\widehat{\mathcal{G}}$, such that for every strategy σ in $\mathcal{G}$, if we note $\widehat{\mathrm{Out}(\sigma)} := \{\widehat{\pi} \mid \pi \in \mathrm{Out}(\sigma)\}$, then $\mathrm{Out}(\widehat{\sigma}) = \widehat{\mathrm{Out}(\sigma)}$.

Let σ be $([T], \varphi)$-fully-uniform in $\mathcal{G}$. If $\widehat{\pi} \in \mathrm{Out}(\widehat{\sigma})$, then $\widehat{\pi} \in \widehat{\mathrm{Out}(\sigma)}$, hence $\pi \in \mathrm{Out}(\sigma)$. Because σ is $([T], \varphi)$-fully-uniform, $\pi, 0 \models_{\sim} \varphi$. By Lemma 16 we conclude that $\widehat{\pi}, 0 \models_{\widehat{\sim}} \widehat{\varphi}$, which means that $\widehat{\sigma}$ is $([\widehat{T}], \widehat{\varphi})$-fully-uniform in $\widehat{\mathcal{G}}$. Since $d_R(\widehat{\varphi}) = d_R(\varphi) - 1 = n$, $(\widehat{\mathcal{G}, T, \varphi}) \in \mathrm{FUS}_n$.

Assume $(\widehat{\mathcal{G}, T, \varphi}) \in \mathrm{FUS}_n$, that is there exists a strategy $\widehat{\sigma}$ which is $([\widehat{T}], \widehat{\varphi})$-fully-uniform in $\widehat{\mathcal{G}}$. Any play $\pi \in \mathrm{Out}(\sigma)$ is uniquely associated to a play $\widehat{\pi} \in \widehat{\mathrm{Out}(\sigma)} = \mathrm{Out}(\widehat{\sigma})$, which by assumption satisfies $\widehat{\pi}, 0 \models_{\widehat{\sim}} \widehat{\varphi}$. By Lemma 16, $\pi, 0 \models_{\sim} \varphi$, which shows that σ is $([T], \varphi)$-fully-uniform in $\mathcal{G}$.

This achieves the proof of the first point of Proposition 13. For the second point, recall Definition 14 that gives $|\widehat{\mathcal{G}}| = O(|\mathcal{G}| \times 2^{|T|} \times 2^{|T||\mathcal{G}|})$, so that $|\widehat{\mathcal{G}}| = O(2^{(|\mathcal{G}|+|T|)^2})$. The third point is derived from this second point and Definition 16: $|\widehat{T}| = O(|\widehat{\mathcal{G}}| \times |T| \times |\widehat{\mathcal{G}}|) = O(2^{(|\mathcal{G}|+|T|)^2} \times |T| \times 2^{(|\mathcal{G}|+|T|)^2}) = O(2^{O(|\mathcal{G}|+|T|)^2})$. Finally, the fourth point stating that $|\widehat{\varphi}| = O(|\varphi|)$ is immediate by Definition 17.

It remains to prove that $(\widehat{\mathcal{G}, T, \varphi})$ is computed in time exponential in $|(\mathcal{G}, T, \varphi)|$. Clearly, the powerset construction (Section 5.1) and the lifting of the transducer (Section 5.2) both take exponential time in $|\mathcal{G}| + |T|$, hence in $|(\mathcal{G}, T, \varphi)|$. The marking phase in the R-elimination (Definition 18) involves model checking at most $|\varphi|$ LTL-formulas on each position of the original arena $\mathcal{G}$. Model checking an LTL-formula in a given position requires polynomial space in $|\mathcal{G}|+|\varphi|$ [SC85]. Since PSPACE $\subseteq$ EXPTIME, it is exponential in time. All in all, we need to model check an LTL-formula at most $|\mathcal{G}| \times |\varphi|$ times, so the whole marking phase is done in time exponential in $|(\mathcal{G}, T, \varphi)|$. We conclude that $(\widehat{\mathcal{G}, T, \varphi})$ can be computed in time exponential in $|(\mathcal{G}, T, \varphi)|$, which terminates the proof of Proposition 13.

5.4 Complexity of FUS_n

In this subsection we describe an algorithm that decides whether an instance $(\mathcal{G}, T, \varphi)$ is in FUS, and we establish upper bounds for the FUS_n problem, for each $n \in setn$.

Algorithm 1 describes our decision procedure. It takes as an entry an instance $(\mathcal{G}, T, \varphi)$ of FUS, and returns `true` if it is a positive instance[4], `false` otherwise. To do so, starting from $(\mathcal{G}_0, T_0, \varphi_0) = (\mathcal{G}, T, \varphi)$, it successively applies the construction described in Subsection 5.3 to eliminate R operators in φ and to ultimately reduce the problem to solving an equivalent LTL game.

It is known that solving LTL games has a time complexity doubly-exponential in the size of the formula, and that it is actually 2EXPTIME-complete [PR89]. We remind that solving an LTL game $(\mathcal{G}, \varphi)$, in the automata-theoretic formulation of this problem [Var91], can be done by the following procedure, that we will call in *LTLGameSolver* (see [PR89, ALT04]). First, compute a nondeterministic Büchi tree automaton that accepts trees whose branches all verify the formula. This automaton is of size exponential in $|\varphi|$. Then, by for example Safra's construction [Saf88], build an equivalent deterministic Rabin automaton $\mathcal{A}_\varphi$ with a number of states doubly-exponential in φ, and a number of pairs exponential in φ. Then, with a linear cost in $|\mathcal{G}|$, transform the arena $\mathcal{G}$ into a nondeterministic tree automaton $\mathcal{A}_\mathcal{G}$ that accepts all

[4] i.e., there exists a $([T], \varphi)$-fully-uniform strategy in $\mathcal{G}$.

strategies of Player 1 in $\mathcal{G}$. Then, there exists a strategy whose outcomes all satisfy φ if and only if the product Rabin automaton $\mathcal{A}_\varphi \times \mathcal{A}_\mathcal{G}$ accepts some tree. Deciding the emptiness of a Rabin tree automaton can be done time $O((\ell m)^{3m})$, where ℓ is the number of states and m is the number of pairs of the Rabin automaton [Ros91]. Provided that for the product Rabin automaton $\mathcal{A}_\varphi \times \mathcal{A}_\mathcal{G}$ we have $\ell = |\mathcal{G}| \times 2^{2^{|\varphi|}}$ and $m = 2^{|\varphi|}$, we finally obtain the following upper bound:

Proposition 17 *Solving an LTL game $(\mathcal{G}, \varphi)$ takes time $|\mathcal{G}|^{2^{O(|\varphi|)}}$.*

It is important to keep the size of the arena and the size of the formula apart for the moment, instead of just saying that it is doubly-exponential in the size of the entry, because in our decision procedure, the size of the iterated powerset constructions suffers a exponential blow-up, contrary to the successive formulas whose sizes remains unchanged (and even decrease since subformulas are replaced with atomic propositions).

Input: $(\mathcal{G}, T, \varphi)$
Output: `true` if $(\mathcal{G}, T, \varphi) \in$ FUS, `false` otherwise
$(\mathcal{G}_0, T_0, \varphi_0) := (\mathcal{G}, T, \varphi)$;
$k := 0$;
while $d_R(\varphi_k) > 0$ **do**
  $(\mathcal{G}_{k+1}, T_{k+1}, \varphi_{k+1}) := (\widehat{\mathcal{G}_k, T_k, \varphi_k})$;
  $k := k + 1$;
end
return *LTLGameSolver*$(\mathcal{G}_k, \varphi_k)$

Algorithm 1: Decision procedure for the problem FUS.

Theorem 8 as announced at the beginning of Section 5 can now be proved.

Let $n \in \mathbb{N}$, and let $(\mathcal{G}, T, \varphi)$ be an instance of FUS$_n$. If $n = 0$, the body of the **while** instruction is not executed, and we immediately call *LTLGameSolver*$(\mathcal{G}_0, \varphi_0)$. By Proposition 17, this call takes time $|\mathcal{G}_0|^{2^{O(|\varphi_0|)}}$, hence it is 2-EXPTIME in $|(\mathcal{G}, T, \varphi)|$. We next answer the case $n > 0$, and will distinguish the two particular cases $n = 1$ and $n = 2$.

For convenience, we introduce notations for iterated exponential functions: for $k, n \in \mathbb{N}$, $\exp^k(n) = 2^{2^{\cdot^{\cdot^{2^n}}}} \}k$.

Lemma 18 *For every $0 \leq k \leq n$, $|\mathcal{G}_k| = |T_k| = exp^k(O(|\mathcal{G}| + |T|)^2)$.*

Proof By induction on k, using Proposition 13. □

If $(\mathcal{G}, T, \varphi)$ is an instance of FUS$_n$, for $1 \leq k \leq n$, by Proposition 13, the execution of the k-th loop takes time exponential in $|(\mathcal{G}_{k-1}, T_{k-1}, \varphi_{k-1})|$. Hence by Lemma 18, the time complexity for the k-th loop is $\exp^1(\exp^{k-1}(O(|\mathcal{G}| + |T|)^2) + |\varphi|) = 2^{|\varphi|}\exp^k(O(|\mathcal{G}| + |T|)^2)$, and the execution of the whole **while** instruction takes time θ_{WHILE} where:

$$\begin{aligned}\theta_{\text{WHILE}} &= \sum_{k=1}^{n} 2^{|\varphi|}\exp^k(O(|\mathcal{G}| + |T|)^2) \\ &= 2^{|\varphi|}\exp^n(O(|\mathcal{G}| + |T|)^2)\end{aligned}$$

By Proposition 17 and Lemma 18, solving the final LTL game $(\mathcal{G}_n, \varphi_n)$ takes time θ_{LTL}, where:

$$\begin{aligned}\theta_{\mathrm{LTL}} &= |\mathcal{G}_n|^{2^{O(|\varphi_n|)}} \\ &= \exp^1(\exp^{n-1}(O(|\mathcal{G}| + |T|)^2) * 2^{O(|\varphi|)})\end{aligned}$$

We obtain that Algorithm 1 runs in time $\theta = \theta_{\mathrm{WHILE}} + \theta_{\mathrm{LTL}}$, but since (for $n > 0$), θ_{WHILE} is negligible compared to θ_{LTL}, we obtain:

$$\theta = \exp^1(\exp^{n-1}(O(|\mathcal{G}| + |T|)^2) * 2^{O(|\varphi|)}) \tag{1}$$

Additionally, for $n = 1$, the double exponential complexity stems from the size of the formula. For $n = 2$, because the size of the arena has taken two exponentials, the double exponential complexity comes both from the size of the formula and the size of the arena. Afterwards, since the arena keeps growing exponentially while the size of the formula remains the same, the complexity comes essentially from the size of the arena. This achieves the proof of Theorem 8.

Note that the subroutine $LTLGameSolver(\mathcal{G}_n, \varphi_n)$ of Algorithm 1, based on the automata-theoretic procedure of [PR89], does not merely decide the existence of a winning strategy, but actually builds one (if any). Recall also that forgetful strategies are sufficient for LTL games, as they are particular cases of regular games which enjoy the "Forgetful Determinacy" [Zei94]. By the natural bijection invoked in the proof of Proposition 13 between strategies in a powerset arena and strategies in the original arena, one can trace the strategy in $\mathcal{G}_n$ back to a $([T_0], \varphi_0)$-fully-uniform strategy in the original game $\mathcal{G}_0$.

Corollary 19 *Forgetful strategies are sufficient for full-uniformity properties.*

6 Discussion

We have investigated the concept of uniform strategies in two-player turn-based infinite-duration games, motivated by the many instances from the literature: games with imperfect information, games with epistemic condition, non-interference, diagnosis and prognosis, and Dependence Logic. Uniformity is addressed in the context of a semantic binary relation between plays of the arena, which can arise from any reason to relate plays with each others, *e.g.* an epistemic feature.

In order to embrace all the examples we have encountered, and likely many potential others, we were led to designing a formal language whose sentences express the very uniformity properties of strategies. Clearly, the language-based approach offers intuitive definitions, while the set-theoretic one, which may capture a larger class of uniformity properties, is much less readable. The particular uniformity properties that have been addressed in the literature so far (Section 3) can now more easily be compared. Our language is an enrichment of the Linear-time Temporal logic LTL [GPSS80], hence it is interpreted over plays. The additional feature is captured by the modality R which quantifies universally over related plays. Whether this quantification ranges over all plays in the arena or just over outcomes of the considered strategy yields two variants of uniform strategies, namely fully-uniform and strictly-uniform strategies.

The general procedure to decide the fully-uniform strategy problem is non-elementary. This may be the price to pay for a generic solution for arbitrarily complex uniformity properties, and we conjecture that the fully-uniform strategy problem is non-elementary hard. However, bounding the R-depth of the formulas gives an elementary bound complexity, which seems incidentally to be the case for all the examples of Section 3: only formulas whose R-depth is one are needed, so that the generic procedure has "only" a double exponential time complexity. Notice that [MPB11] obtained a tighter (optimal) single exponential time bound for solving games with opacity condition, which corresponds to the fixed formula $\mathbf{G}\neg Rp_S$ of R-depth equal to one and to a simple fixed binary relation between plays (see Section 3.2). Notice that if we fix a formula of R-depth 1, the time complexity Equation (1) of our procedure collapses to a single exponential time complexity in the size of the arena and of the transducer.

Our results can be extended and commented in many respects. We give here some of them.

First, the choice we have made to rely on an enrichment of the LTL logic can be questioned – although this logic regarding properties of time is acknowledged in many respects. We may try to extend the synthesis procedure to a much richer logic like the Linear-time μ-calculus, a language extending standard linear time temporal logic with fix-point operators. But the current procedure relies on a bottom-up traversal of the parse-tree of the formula φ, which cannot be generalized to formulas with arbitrary fix-points. The LTL logic falls into the very particular so-called *alternation-free* fragment of the Linear-time μ-calculus, where fix-points do not interplay. Significant progress in understanding this extended setting need being pursued.

Second, we considered a single semantic binary relation between plays. One may wonder whether the case of several relations $\rightsquigarrow_i$, yielding modalities R_i at the language level, can be investigated at the algorithmic level. We foresee a generalization of our powerset construction by synchronizing the execution of all the transducers of the relations. We however remain cautious regarding the success of this approach since closely related topics such as the *decentralized* diagnosis problem is known to be undecidable [ST02]. Still the question is important as it would unify our setting with the Epistemic Temporal Logic ETL of [HV89] and bring light on the automated verification of ETL definable properties for *open systems* (see the *module*-checking problem of [KV97]).

Additional comments are needed to fully understand the contribution, in particular regarding the recent developments of alternating-time epistemic logic [vdHW03, JH04, DEG10]. The two settings are close but incomparable. With the uniform strategy concept, we aim at extending the range of (qualitative) properties of strategies by means of binary relations between plays, and at exploiting those properties to synthesize particular strategies. Instead, alternating-time epistemic logics offer a way to quantify over strategies that achieve ETL-like properties, hence they are not synthesis-oriented, and moreover, they do not handle arbitrary relations between plays. Unifying the two settings is a real challenge; we would need to design a (necessarily more complex) language that incorporates the specification of the relation(s) between plays.

References

[AČC07] R. Alur, P. Černỳ, and S. Chaudhuri. Model checking on trees with path equivalences. *Tools and Algorithms for the Construction and Analysis of Systems*, pages 664–678, 2007.

[AČZ06] Rajeev Alur, Pavol Černý, and Steve Zdancewic. Preserving secrecy under refinement. In *ICALP '06: Proceedings (Part II) of the 33rd International Colloquium on Automata, Languages and Programming*, pages 107–118. Springer, 2006.

[AG11] K.R. Apt and E. Grädel. *Lectures in Game Theory for Computer Scientists.* Cambridge University Press, 2011.

[AHK02] R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM (JACM)*, 49(5):672–713, 2002.

[ALT04] R. Alur and S. La Torre. Deterministic generators and games for ltl fragments. *ACM Transactions on Computational Logic (TOCL)*, 5(1):1–25, 2004.

[AVW03] A. Arnold, A. Vincent, and I. Walukiewicz. Games for synthesis of controllers with partial observation. *Theoretical Computer Science*, 1(303):7–34, 2003.

[BD08] D. Berwanger and L. Doyen. On the power of imperfect information. In *Proc. of FSTTCS*, pages 73–82. Citeseer, 2008.

[Ben05] J. Benthem. The epistemic logic of if games. *The Philosophy of Jaakko Hintikka*, 30, 2005.

[Ber79] J. Berstel. *Transductions and context-free languages*, volume 4. Teubner Stuttgart, 1979.

[CDHR06] K. Chatterjee, L. Doyen, T. Henzinger, and J.F. Raskin. Algorithms for omega-regular games with imperfect information. In *Computer Science Logic*, pages 287–302. Springer, 2006.

[CL99] C.G. Cassandras and S. Lafortune. Discrete event systems- the state of the art and new directions. *Applied and computational control, signals, and circuits.*, 1:1–65, 1999.

[DEG10] C. Dima, C. Enea, and D. Guelev. Model-checking an alternating-time temporal logic with knowledge, imperfect information, perfect recall and communicating coalitions. *Electronic Proceedings in Theoretical Computer Science*, 25, 2010.

[EJ91] E. A. Emerson and C. S. Jutla. Tree automata, mu-calculus and determinacy. In *Proceedings 32nd Annual IEEE Symp. on Foundations of Computer Science, FOCS'91, San Jose, Puerto Rico, 1–4 Oct 1991*, pages 368–377. IEEE Computer Society Press, Los Alamitos, California, 1991.

[EM65] C.C. Elgot and J.E. Mezei. On relations defined by generalized finite automata. *IBM Journal of Research and Development*, 9(1):47–68, 1965.

[FHV91] R. Fagin, J.Y. Halpern, and M.Y. Vardi. A model-theoretic analysis of knowledge. *Journal of the ACM (JACM)*, 38(2):382–428, 1991.

[GM82] J.A. Goguen and J. Meseguer. Security policies and security models. In *IEEE Symposium on Security and privacy*, volume 12, 1982.

[GPSS80] D. Gabbay, A. Pnueli, S. Shelah, and J. Stavi. On the temporal analysis of fairness. In *Proc. 7th ACM Symp. Principles of Distributed Computing, Toronto, Ontario, Canada*, pages 163–173, January 1980.

[GTW02] E. Grädel, W. Thomas, and T. Wilke, editors. *Automata, Logics, and Infinite Games: A Guide to Current Research [outcome of a Dagstuhl seminar, February 2001]*, volume 2500 of *Lecture Notes in Computer Science*. Springer, 2002.

[Hin62] J. Hintikka. *Knowledge and belief*, volume 13. Cornell University Press Ithaca, 1962.

[HS89] J. Hintikka and G. Sandu. Informational independence as a semantical phenomenon. *Studies in Logic and the Foundations of Mathematics*, 126:571–589, 1989.

[HV89] J.Y. Halpern and M.Y. Vardi. The complexity of reasoning about knowledge and time. 1. lower bounds. *Journal of Computer and System Sciences*, 38(1):195–237, 1989.

[JH04] W. Jamroga and W.V.D. Hoek. Agents that know how to play. *Fundamenta Informaticae*, 63(2-3):185–220, 2004.

[KV97] O. Kupferman and M. Y. Vardi. Module checking revisited. In *Computer Aided Verification, Proc. 9th Int. Conference*, volume 1254 of *Lecture Notes in Computer Science*, pages 36–47. Springer-Verlag, 1997.

[Leh84] D. Lehmann. Knowledge, common knowledge and related puzzles (extended summary). In *Proceedings of the third annual ACM symposium on Principles of distributed computing*, pages 62–67. ACM, 1984.

[LR86] R.E. Ladner and J.H. Reif. The logic of distributed protocols: preliminary report. In *Proceedings of the 1986 Conference on Theoretical aspects of reasoning about knowledge*, pages 207–222. Morgan Kaufmann Publishers Inc., 1986.

[MPB11] Bastien Maubert, Sophie Pinchinat, and Laura Bozzelli. Opacity issues in games with imperfect information. In Giovanna D'Agostino and Salvatore La Torre, editors, Proceedings Second International Symposium on *Games, Automata, Logics and Formal Verification,* Minori, Italy, 15-17th June 2011, volume 54 of *Electronic Proceedings in Theoretical Computer Science*, pages 87–101. Open Publishing Association, 2011.

[PR85] R. Parikh and R. Ramanujam. Distributed processes and the logic of knowledge. *Logics of Programs*, pages 256–268, 1985.

[PR89] A. Pnueli and R. Rosner. On the synthesis of an asynchronous reactive module. In *Proc. 16th Int. Coll. on Automata, Languages and Programming, ICALP'89, Stresa, Italy, LNCS 372*, pages 652–671. Springer-Verlag, July 1989.

[PR97] M. Piccione and A. Rubinstein. The absent-minded driver's paradox: synthesis and responses. *Games and Economic Behavior*, 20(1):121–130, 1997.

[PR05] S. Pinchinat and S. Riedweg. A decidable class of problems for control under partial observation. *Inf. Process. Lett.*, 95(4):454–460, 2005.

[Rei84] J.H. Reif. The complexity of two-player games of incomplete information. *Journal of computer and system sciences*, 29(2):274–301, 1984.

[Ros91] R. Rosner. Modular synthesis of reactive systems. *Ann Arbor*, 1050:48106–1346, 1991.

[Saf88] S. Safra. On the complexity of ω-automata. In *Proc. 29th IEEE Symp. Foundations of Computer Science, White Plains, New York*, pages 319–327, October 1988.

[Sat77] JM Sato. A study of kripke style methods for some modal logic by gentzen's sequential method. Technical report, Technical report, Publication Research Institute for Mathematical Science, 1977.

[SC85] A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logics. *Journal of the ACM*, 32(3):733–749, July 1985.

[SSL+95] M. Sampath, R. Sengupta, S. Lafortune, K. Sinaamohideen, and D. Teneketzis. Diagnosability of discrete event systems. *IEEE Transactions on Automatic Control*, 40(9):1555–1575, 1995.

[ST02] R. Sengupta and S. Tripakis. Decentralized diagnosability of regular languages is undecidable. In *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, volume 1, pages 423–428. IEEE, 2002.

[Vää07] J. Väänänen. Dependence logic, 2007.

[Var91] M.Y. Vardi. Verification of concurrent programs: The automata-theoretic framework. *Annals of Pure and Applied Logic*, 51(1):79–98, 1991.

[VB01] J. Van Benthem. Games in dynamic-epistemic logic. *Bulletin of Economic Research*, 53(4):219–248, 2001.

[vdHW03] W. van der Hoek and M. Wooldridge. Cooperation, knowledge, and time: Alternating-time temporal epistemic logic and its applications. *Studia Logica*, 75(1):125–157, 2003.

[YL02] T.S. Yoo and S. Lafortune. Polynomial-time verification of diagnosability of partially observed discrete-event systems. *Automatic Control, IEEE Transactions on*, 47(9):1491–1495, 2002.

[Zei94] S. Zeitman. Unforgettable forgetful determinacy. *Journal of Logic and Computation*, 4(3):273–283, 1994.

RESEARCH CENTRE
RENNES – BRETAGNE ATLANTIQUE

Campus universitaire de Beaulieu
35042 Rennes Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399