Jumping Automata for Uniform Strategies

Bastien Maubert and Sophie Pinchinat

Université de Rennes 1, IRISA, Rennes, France

Abstract

The concept of uniform strategies has recently been proposed as a relevant notion in game theory for computer science. It relies on properties involving sets of plays in two-player turn-based arenas equipped with a binary relation between plays. Among the two notions of *fully-uniform* and *strictly-uniform* strategies, we focus on the latter, less explored. We present a language that extends CTL^* with a quantifier $\boxminus$ over all related plays, which enables to express a rich class of uniformity constraints on strategies. We show that the existence of a uniform strategy is equivalent to the language non-emptiness of a *jumping tree automaton*. While the existence of a uniform strategy is undecidable for *rational* binary relations, restricting to *recognizable* relations yields a 2Exptime-complete complexity, and still captures a class of two-player imperfect-information games with epistemic temporal objectives. This result relies on a translation from jumping tree automata with recognizable relations to two-way tree automata.

1998 ACM Subject Classification F.4 Mathematical Logic and Formal Languages

Keywords and phrases Games, Imperfect information, Uniform strategies, Jumping automata

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2013.287

1 Introduction

Infinite-duration game models have been intensively studied for their applications in computer science [2] and logic [10]. First, infinite-duration games provide a natural abstraction of computing systems' non-terminating interaction [1] (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 [8], leading to a powerful theory of automata, logics and infinite games [10] and to the development of algorithms for the automatic verification ("model-checking") and synthesis of hardware and software systems. In all cases, solving games aims at computing a strategy (of some distinguished player) whose outcomes fulfill ω-regular conditions meant to describe some desirable property.

In essence, ω-regular conditions are evaluated on individual plays, independently of other plays that result from the strategy. However, turning to imperfect-information games raises the need to deal with *sets* of plays, as the strategic decision has to be the same in indistinguishable situations [19]. This typical property of strategies in imperfect-information games is in general dealt aside from the ω-regular winning conditions. However, this splitting is a real issue when considering properties of strategies that mix, *e.g.* knowledge and time.

In an attempt to study this problem in depth, [14] introduced a general notion of *uniform strategies* and showed that it captures a variety of settings from the literature. Uniformity properties of strategies are expressed in a logic that combines standard temporal modalities with two new quantifiers, $\boxminus$ and $\boxbox$, that universally quantify over "related" plays according to some binary relation between plays. The difference between the two quantifiers is in their range. While the *strict* quantifier $\boxminus$ only quantifies over related plays that follow the strategy, the *full* quantifier $\boxbox$ ranges over all related plays in the arena.

33rd Int'l Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2013).
Editors: Anil Seth and Nisheeth K. Vishnoi; pp. 287–298

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

These quantifiers generalize the knowledge operator K of epistemic temporal logics [11]: classically, the semantics of K is a universal quantification over histories related to the actual one by some observational equivalence relation that captures the capabilities of the agent – perfect/imperfect recall, synchronous/asynchronous, ... [11]. In contrast, the setting of [14] allows for arbitrary binary relations for the semantics of the quantifiers, as long as they are *rational*, *i.e.* recognized by *finite state transducers* [6, 3]. Noticeably, most equivalence relations used in epistemic temporal logics are recognized by very simple transducers. Additionally, rational relations need not be equivalences, and can capture relations used in belief revision, and for modelling plausibility, with K45 or KD45 axiomatization [9].

Restricting to formulas that use only one kind of quantifier, the strict one or the full one, yields the notions of *strictly-uniform strategies* and *fully-uniform strategies* respectively. Deciding the existence of a fully-uniform strategy has already been investigated [15]: the problem is k-Exptime-complete for logical specifications that involve up to k nested $\boxminus$ quantifiers – 2Exptime-complete if $k \leq 2$.

In this work, we focus on the even more involved case of strict-uniformity which, as opposed to full-uniformity, does not allow for bottom-up constructions, as one cannot evaluate inner-most formulas before knowing the whole searched strategy. Actually, this intricate feature yields undecidability of the *strictly-uniform strategy problem* (i.e. the existence of a strictly uniform strategy) when the whole class of rational relations is considered. More precisely, we first show that this undecidability result holds even if we restrict to the subclass of regular[1] equivalence relations.

In an effort to better understand the difficulty of this problem, we propose an automata-based approach inspired by [21] for solving LTL games. We introduce and study *jumping tree automata (JTA)*, a class of tree automata which generalizes standard alternating tree automata. JTA are equipped with a binary relation between branches of trees and, in addition to normal behaviour of alternating automata, they allow for jumps between related nodes of the input tree. Intuitively, the jumps of JTA "implement" the meaning of the $\boxminus$ operator in the logic $\boxminus\mathsf{CTL}^*$, that we also define in this contribution as an extension of the logic $\boxminus\mathsf{LTL}$ introduced in [14]. We show that JTA capture the full logic $\boxminus\mathsf{CTL}^*$, and that from a uniformity property a JTA can be built that accepts the tree unfoldings of strictly-uniform strategies.

Although the language emptiness problem for JTA is unsurprisingly undecidable when considering arbitrary rational relations over branches of trees, we identify a decidable case when the class of binary relations between branches is confined to the well-known family of *recognizable* relations; basically, such relations only challenge a bounded amount of information in each branch. Decidability of JTA emptiness in this case is shown by an effective transformation of JTA with recognizable relations into equivalent two-way tree automata, and decidability of the strictly-uniform strategy problem for recognizable relations follows. More precisely, we establish that the emptiness problem for JTA with recognizable relations is Exptime-complete, and that the strictly-uniform strategy problem for this class of relations is 2Exptime-complete.

This latter result sheds light on phenomena in games of imperfect information. In such games, only *observation-based* strategies are allowed, enforcing players to play identically along plays that share the same sequence of observations. A weaker form of this requirement is the *knowledge-based*[2] strategies, *i.e.* players play identically along plays that yield the

[1] captured by synchronous transducers

[2] This vocabulary is highly misleading, and we now rather say *information-set-based* instead.

same information set. Although being rational, the observation-based equivalence relation is not recognizable, unlike the information-set-based equivalence relation. Interestingly, in two-player games with ω-regular winning objectives, the existence of an observation-based strategy implies the existence of an information-set-based one, hence looking for the latter is enough; but this does no longer hold for more players. Our results (on undecidability/decidability) distinguishing arbitrary rational relations from recognizable ones gives a new insight on the frontier between imperfect-information games with two players and games with more players.

Additionally, our automata-theoretic approach based on jumping tree automata yields an effective method to solve two-player imperfect-information games with epistemic temporal logic specifications of winning conditions. By interpreting the $\boxminus$ quantifier of the logic $\boxminus\mathsf{CTL}^*$ as the information-set-based knowledge operator, we can in a unified formalism express that, *e.g.* a strategy is information-set-based on the one hand, and fulfills some branching-time epistemic temporal property, on the other hand. Actually, we may even extend the setting to deal with several $\boxminus_i$ modalities, referring to different binary relations in a single specification: in particular, our decidability result for recognizable relations would still hold, with no additional complexity cost. It would then be possible to seek for a single player's information-set-based strategy with a fairly rich epistemic condition involving the knowledge of other players. Also, because recognizable relations are closed under intersection, such an extension would enable to reason on multi-player imperfect-information arenas about the existence of a coalition strategy (involving two meta players: the coalition and the anti-coalition) with distributed knowledge, as long as this knowledge can be bounded. Due to lack of space, we do not present this significant generalization in the paper.

The rest of the paper is organized as follows. In Section 2, we remind some notions concerning trees and game arenas. We present the language $\boxminus\mathsf{CTL}^*$ to specify uniform strategies in Section 3, and we prove in Section 4 that the strictly-uniform strategy problem is undecidable for regular equivalence relations. In Section 5 we define jumping tree automata, we show that they capture $\boxminus\mathsf{CTL}^*$ and we present a reduction of the strictly-uniform strategy problem to their non-emptiness. Finally, we prove in Section 6 that JTA with recognizable relations can be turned into equivalent two-way tree automata, yielding the decidability of the strictly-uniform strategy problem for recognizable relations.

2 Preliminaries

2.1 Basic notions on trees

For the rest of the paper, let $k \in \mathbb{N}$ be a natural number, and let $[k]$ denote the set $\{1, \dots, k\}$. An *infinite tree* is a nonempty set $t \subseteq [k]^*$ such that if $x \cdot i \in t$, then $x \in t$, and if $x \in t$, there exists $i \in [k]$ such that $x \cdot i \in t$. The elements of t are called *nodes*, and the empty word ϵ is the *root* of the tree. If $x \cdot i \in t$, we say that $x \cdot i$ is a *child* of x, and that x is the *parent* of $x \cdot i$. We will note $x \cdot \uparrow$ the parent of a node x: $(x \cdot i) \cdot \uparrow := x$. Note that $\epsilon \cdot \uparrow$ is undefined as the root has no parent. Given a node x of a tree t, we let $Paths(x)$ be the set of infinite paths $\pi = x_0 x_1 \dots$ in t such that $x_0 = x$ and for all i, x_{i+1} is a child of x_i. Also, for a path $\pi = x_0 x_1 x_2 \dots$, π^i denotes $x_i x_{i+1} x_{i+2} \dots$

Given a finite alphabet Σ, a Σ-*labelled tree* is a pair $T = (t, \ell)$, where t is a tree and $\ell : t \to \Sigma$ is a *labelling*. For a node $x = i_1 i_2 \dots i_n$ in t, $n \geq 0$, we define its *node word* $w(x)$ made of the sequence of labels from the root to this node: $w(x) := \ell(\epsilon)\ell(i_1)\ell(i_1 i_2) \dots \ell(i_1 \dots i_n)$.

2.2 Two-player turn-based game arenas

We consider two-player turn-based games played on finite graphs with vertices labelled with propositions. For the rest of the paper, we let AP be an infinite countable set of *atomic propositions*.

An *arena* is a finite structure $\mathcal{G} = (V, E, v_0, \nu)$ where $V = V_1 \uplus V_2$ is the finite set of *positions*, partitioned between positions of Player 1 (V_1) and those of Player 2 (V_2), $E \subseteq V \times V$ is the set of *edges*, $v_0 \in V$ is the *initial position* and $\nu : V \to 2^{AP}$ is a *valuation function*, mapping each position to a *finite* set of atomic propositions. We will assume that for each position v there is an atomic proposition p_v such that $p_v \in \nu(v')$ if, and only if, $v = v'$. For $v \in V$, we write $E(v) = \{v' \mid (v, v') \in E\}$ for the set of successors of v, and we assume that for all $v \in V$, $E(v)$ is nonempty. $Plays_*$ and $Plays_\omega$ are, respectively, the set of finite and infinite *plays*. For an infinite play $\pi = v_0 v_1 \dots$ and $i \in \mathbb{N}$, $\pi[i] := v_i$ and $\pi[0, i] := v_0 \dots v_i$.

A *strategy* for Player 1 is a partial function $\sigma : Plays_* \to V$ that maps a finite play ending in $v \in V_1$ to some successor of v. 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. In the following, it is convenient to see a strategy σ as the tree T_σ, obtained by unfolding the arena and pruning moves that do not follow the strategy. Formally, a *strategy tree* $T = (t, \ell)$ is a 2^{AP}-labelled tree such that $\ell(\epsilon) = \nu(v_0)$ and for every $x \in t$, letting v be the unique position such that $p_v \in \ell(x)$:

- if $v \in V_1$ then x has a unique child x', and $\ell(x') = \nu(v')$ for some $v' \in E(v)$
- if $v \in V_2$ then x has exactly one child x' for each $v' \in E(v)$, and $\ell(x') = \nu(v')$.

3 Uniform strategies

A uniform strategy is a strategy subject to a uniformity constraint, such as being *observation-based* (see Example 2). In the purpose of automated synthesis of uniform strategies, we introduce a logical formalism $\boxminus\mathsf{CTL}^*$, in the spirit of [14], but extending the temporal features from LTL to CTL^* [7] (enabling us to capture, *e.g.* module-checking [13]). The logic $\boxminus\mathsf{CTL}^*$ is interpreted over infinite trees, and we use it to reason on strategies seen as trees. It contains a quantifier $\boxminus$ that universally quantifies over "related" finite plays, or equivalently over nodes in the strategy tree related by some binary relation $\rightsquigarrow$.

The set of well-formed $\boxminus\mathsf{CTL}^*$ state formulas is given by the following grammar:

$$\begin{array}{lll} \text{State formulas:} & \varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid A\psi \mid \boxminus\varphi & \text{where } p \in AP \\ \text{Path formulas:} & \psi ::= \varphi \mid \neg\psi \mid \psi \vee \psi \mid \mathbf{X}\psi \mid \psi\mathbf{U}\psi & \end{array}$$

Classically, we define the Boolean conjunction for both types of formula $\varphi \wedge \varphi$ and $\psi \wedge \psi$ with their expected meaning. Also, for each path formula ψ, we write $E\psi$ for the state formula $\neg A \neg\psi$, $\mathbf{F}\psi$ for $\top\mathbf{U}\psi$, $\mathbf{G}\psi$ for $\neg\mathbf{F}\neg\psi$ and for each state formula φ, we write $\diamondminus\varphi$ for $\neg\boxminus\neg\varphi$.

Given a binary relation $\rightsquigarrow$ over $(2^{AP})^*$, a 2^{AP}-labelled tree [3] $T = (t, \ell)$ and two nodes $x, y \in t$, we will abuse notations by writing $x \rightsquigarrow y$ for $w(x) \rightsquigarrow w(y)$ (that is their node words are related by $\rightsquigarrow$). We define the semantics of $\boxminus\mathsf{CTL}^*$ as follows, where $x \in t$ is a node and π is an infinite path:

[3] In fact, by 2^{AP}, we mean $2^{AP'}$, where AP' is some finite subset of AP.

$x \models p$ if $p \in \ell(x)$ $\quad$ $x \models \neg\varphi$ if $x \not\models \varphi$
$x \models \varphi_1 \vee \varphi_2$ if $w \models \varphi_1$ or $w \models \varphi_2$ $\quad$ $x \models A\psi$ if for all $\pi \in Paths(x)$, $\pi \models \psi$
$x \models \boxminus\varphi$ if for all $y \in t$ such that $x \rightsquigarrow y$, $y \models \varphi$
$\pi \models \varphi$ if $x_0 \models \varphi$, where $\pi = x_0x_1\ldots$ $\quad$ $\pi \models \neg\psi$ if $\pi \not\models \psi$
$\pi \models \psi_1 \vee \psi_2$ if $\pi \models \psi_1$ or $\pi \models \psi_2$ $\quad$ $\pi \models \mathbf{X}\psi$ if $\pi^1 \models \psi$
$\pi \models \psi_1 \mathbf{U} \psi_2$ if there exists $i \geq 0$ such that $\pi^i \models \psi_2$ and for all $0 \leq j < i$, $\pi^j \models \psi_1$

We extend the semantics to trees by writing $T \models \varphi$ whenever $\epsilon \models \varphi$. In particular, $T \models A\psi$ if every branch of T satisfies ψ.

▶ **Definition 1.** Let $\mathcal{G}$ be an arena, $\rightsquigarrow$ be a relation over $(2^{AP})^*$ and φ be a $\boxminus$CTL* formula. A strategy σ is $(\rightsquigarrow, \varphi)$-*uniform* if the strategy tree of σ satisfies φ, *i.e.* $T_\sigma \models \varphi$.

▶ **Example 2.** Consider the case of observation-based strategies in two-player games with imperfect information. Such games are played on graphs with edges labelled by *actions* and proceed as follows. In position v, Player 1 chooses an action a, and Player 2 chooses a new position reachable from v via an a-edge. The imperfect information is caused by the inability of Player 1 to see the very position chosen by Player 2, but only its *observation*; note that the same observation may be shared by several positions. In such games, strategies for Player 1 are required to assign the same action to observationally equivalent situations.

Such a requirement can be specified as a uniformity constraint in our framework. To do so, we convert actions into positions: choosing action a in node v is simulated by moving to the newly created position (v, a), and we enrich the arena labelling by atomic propositions p_o for observations, and by p_a in all positions of the form (v, a). Noting $\rightsquigarrow$ the observational equivalence relation, and defining $\psi_{Obs} := \mathbf{G} \bigwedge_{a \in Act} (\mathbf{X}p_a \rightarrow \boxminus E\mathbf{X}p_a)$, we have: a strategy is observation-based if, and only if, it is a $(\rightsquigarrow, A\psi_{Obs})$-uniform strategy. Indeed, a strategy tree that verifies $A\psi_{Obs}$ represents a strategy that, whenever it recommends action a after some finite play, it does so in all $\rightsquigarrow$-related (*i.e.* observationally equivalent) finite plays.

Beyond the example above, the approach clearly enables to represent various kinds of observation-based strategies by tuning relation $\rightsquigarrow$ according to the desired player's observational abilities and/or memory resources (perfect-recall, memoryless, synchronous, asynchronous, etc). Note that in this framework, a formula like $A\psi_{Obs} \wedge \varphi$ naturally combines on observation-based constraint with an arbitrary epistemic temporal winning condition φ. Also we refer the interested reader to [14] for more examples of uniform strategies.

4 A first undecidability result

In order to address automated synthesis of uniform strategies one has to make assumptions on the relation used in the semantics of the language $\boxminus$CTL*, and in particular how it can be finitely represented. We therefore consider the expressive class of *rational relations*.

4.1 Rational relations

We briefly recall the notions of rational relations and transducers. A *finite state transducer* over the alphabet Σ is a tuple $\mathcal{M} = (Q, q_0, Q_F, \Delta)$, where Q is a finite set of states, $q_0 \in Q$ is a distinguished *initial state*, $Q_F \subseteq Q$ is a set of *accepting states*, and $\Delta \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times (\Sigma \cup \{\epsilon\}) \times Q$ is a finite set of *transitions*. A transducer is *synchronous* if $\Delta \subseteq Q \times \Sigma \times \Sigma \times Q$.

Intuitively, a transducer reads a finite word $w \in \Sigma^*$ on its input tape, writes a finite word $w' \in \Sigma^*$ on its output tape, and accepts (w, w') if it reaches an accepting state. The relation $[\mathcal{M}]$ *recognized by* $\mathcal{M}$ is the set of pairs (w, w') accepted by $\mathcal{M}$.

It is well known that transducers recognize *rational* relations [3], and that synchronous transducers recognize *regular* relations [12].

For example, the observational equivalence relation for a player with asynchronous perfect-recall is rational, and accepted by a fairly simple transducer with one state per observation. For synchronous perfect-recall, the relation is regular and is accepted by an even simpler synchronous transducer with only one state.

4.2 The problem SUS and its undecidability

Back to our central topic on uniform strategies, we consider the following decision problem:

$$\mathrm{SUS} := \left\{ (\mathcal{G}, \mathcal{M}, \varphi) \;\middle|\; \begin{array}{l} \mathcal{G} \text{ is a finite arena, } \mathcal{M} \text{ is a transducer, } \varphi \in \boxminus\mathsf{CTL}^* \\ \text{and there exists a } ([\mathcal{M}], \varphi)\text{-uniform strategy in } \mathcal{G}. \end{array} \right\}$$

It can be proven, by reduction of the Post Correspondence Problem, that SUS is undecidable, but we propose here the stronger result.

▶ **Theorem 3.** SUS *is undecidable for rational relations, even if we restrict to regular equivalence relations.*

The rest of the section is dedicated to the proof of Theorem 3. We reduce the *distributed strategy problem for three-player imperfect-information games with safety objective*, as addressed by [16, 4]. We present the problem as stated in [4], in which two players with imperfect information (Player A and Player B) play against nature (the third player). Formally, let Act_A (resp. Act_B) be a finite set of available actions for Player A (resp. Player B), and Γ_A (resp. Γ_B) be a finite set of observations for Player A (resp. Player B). We write $Act = Act_A \times Act_B$.

A finite *safety imperfect-information game* is a tuple $\mathcal{G}^{imp} = (V, E, v_0, \gamma_A, \gamma_B)$ where $E \subseteq V \times Act \times V$ is a set of transitions, $\gamma_X : V \to \Gamma_X$ is an observation function ($X \in \{A, B\}$), and with a designated subset $Bad \subseteq V$ of "bad" positions that Player A and Player B should avoid. In each round, Player X chooses an action $c_X \in Act_X$, which gives an action profile $x = (c_A, c_B)$, and nature chooses a next position in $E(v, x) = \{v' \mid (v, x, v') \in E\}$. We suppose that all actions are allowed in every position: for all $v \in V$, $a \in Act_A$ and $b \in Act_B$, we have $E(v, (a, b)) \neq \emptyset$. The observation functions are extended to finite plays $\rho = v_0 x_0 v_1 \dots x_{n-1} v_n$ by letting $\gamma_X(\rho) = \gamma_X(v_0)\gamma_X(v_1)\dots\gamma_X(v_n)$. Note that actions are not observed.

A strategy for Player X is a partial mapping $\sigma_X : (V \cdot Act)^* \cdot V \to Act_X$ that assigns an action to each finite play. It must be observation-based: for any finite plays ρ and ρ' such that $\gamma_X(\rho) = \gamma_X(\rho')$, $\sigma_X(\rho) = \sigma_X(\rho')$. A *distributed strategy* is a pair (σ_A, σ_B) of strategies for Player A and Player B. The *outcome* of a distributed strategy is the set of infinite plays that are both induced by σ_A and σ_B, and a distributed strategy is winning if no play in the outcome ever visits a position in Bad.

It is well known [16, 4] that the following problem is undecidable : *given a safety imperfect-information game, does there exist a winning distributed strategy?*

We reduce this problem to SUS. Let $\mathcal{G}^{imp} = (V, E, v_0, \gamma_A, \gamma_B)$ be an imperfect-information arena with observations Γ_A and Γ_B and bad states Bad. We build a game arena $\mathcal{G} = (V', E', v_0', \nu)$ in which Player 1 plays for both Player A and Player B, and Player 2 plays for nature; Figure 1 shows how each transition in $\mathcal{G}^{imp}$ is transformed into a widget in $\mathcal{G}$.

The set of positions $V' = V_1^A \uplus V_1^B \uplus V_2$ is split into three: in positions of $V_1^A = V$, Player 1 simulates moves of Player A, in positions of $V_1^B = V \times Act_A$, Player 1 simulates moves of Player B, and in positions of $V_2 = V \times Act_A \times Act_B$, Player 2 simulates moves

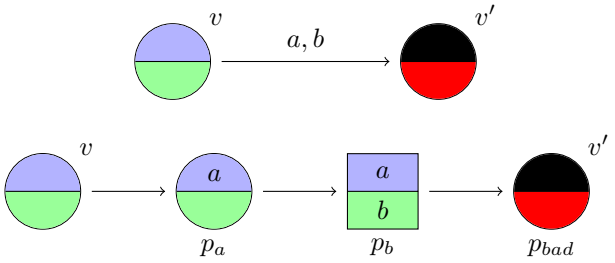

Figure 1 Coding in $\mathcal{G}$ a transition $(v,(a,b),v')$ of $\mathcal{G}^{imp}$, with $v' \in Bad$. Colors represent the observations of Player A and Player B.

of nature. Hence for all v, v', a, b, we have $(v,(v,a)) \in E'$, $((v,a),(v,a,b)) \in E'$, and if $(v,(a,b),v') \in E$ then $((v,a,b),v') \in E'$. For each action $c \in Act_X$, p_c labels positions in which the last move was Player 1 simulating the choice of action c by Player X. In addition, "bad" positions are marked with proposition p_{bad}. As it is assumed in our definition of game arenas that each position v is identified by an atomic proposition p_v, we keep it silent in the following formal definition of the labelling. We consider the set of atomic propositions $\{p_c \mid c \in Act_A \cup Act_B\} \cup \{p_{bad}\}$, and we label the arena as follows: if $v \in Bad$, $\nu(v) = \{p_{bad}\}$, $\nu(v,a) = \{p_a, p_{bad}\}$ and $\nu(v,a,b) = \{p_b, p_{bad}\}$; otherwise, $\nu(v) = \emptyset$, $\nu(v,a) = \{p_a\}$ and $\nu(v,a,b) = \{p_b\}$. Finally, we set $v'_0 = v_0$.

Clearly, since Player 1 plays for the coalition $\{A,B\}$, we expect each branch of her strategy to satisfy the following path formula:

$$\psi_{Safe} := \mathbf{G}\neg p_{bad}. \tag{1}$$

For a finite play $\rho = v_0(v_0,a_0)(v_0,a_0,b_0)v_1\ldots$, we note $\gamma_X(\rho) = \gamma_X(v_0)\gamma_X(v_1)\ldots$

We want to enforce that when Player 1 simulates a move of Player X, her choice is only based on Player X's observation. To do so, we define the symmetric and transitive relation $\rightsquigarrow$ over V'^* that relates two sequences of positions if they end in positions belonging to the same player (A or B), and are observationally equivalent for this player:[4]

$$\rightsquigarrow := \left\{ (\rho,\rho') \;\middle|\; \begin{array}{l} \text{last}(\rho) \in V_1^1 \text{ and last}(\rho') \in V_1^A \text{ and } \gamma_A(\rho) = \gamma_A(\rho'), \text{ or} \\ \text{last}(\rho) \in V_1^B \text{ and last}(\rho') \in V_1^B \text{ and } \gamma_B(\rho) = \gamma_B(\rho') \end{array} \right\}. \tag{2}$$

Note that for the sake of clarity we defined $\rightsquigarrow$ on V^* instead of $(2^{AP})^*$, but by considering the propositions $p_v \in AP$ ($v \in V$) and working with the alphabet $\Sigma = \{\{p_v\} \cup \nu(v) \mid v \in V\}$, one can easily rephrase relation $\rightsquigarrow$ of Equation (2) as a binary relation over Σ^*.

The following path formula states that whenever Player 1 simulates a move of Player X, she chooses the same action in all plays observationally equivalent for Player X:

$$\psi_{Obs} := \mathbf{G} \bigwedge_{c \in Act_A \cup Act_B} \mathbf{X}p_c \rightarrow \boxminus E\mathbf{X}p_c. \tag{3}$$

Combining Equations (1) and (3) we get the following reduction.

▶ **Lemma 4.** *There is a winning distributed strategy in $\mathcal{G}^{imp}$ if and only if there is a $(\rightsquigarrow, A(\psi_{Obs} \wedge \psi_{Safe}))$-uniform strategy in $\mathcal{G}$.*

[4] For a finite word w, last(w) classically denotes its last letter.

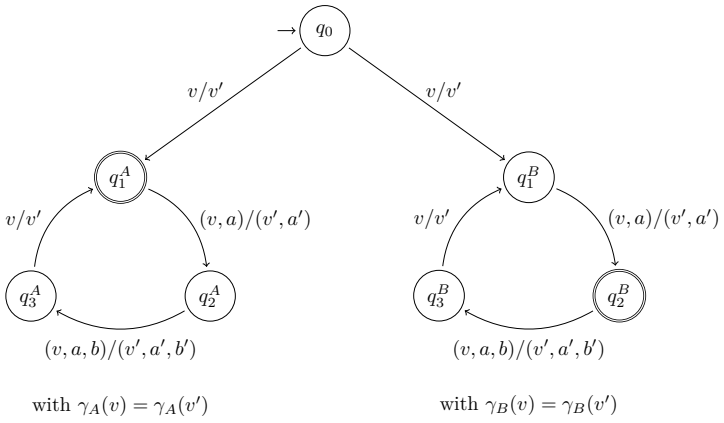

Figure 2 The synchronous transducer $\mathcal{M}_{A,B}$.

We show that $\rightsquigarrow$ is regular.

Consider the synchronous transducer $\mathcal{M}_{A,B}$ of Figure 2. State q_0 is the initial state (ingoing arrow), q_1^A and q_2^B are final states (doubly circled). Transducer $\mathcal{M}_{A,B}$ works as follows: before reading a word w, the transducer guesses whether we are interested in Player A or Player B's observation. In the first case it goes to the left, reads w and writes a word w' observationally equivalent for Player A (remember that actions are not observed). The pair (w, w') is accepted if w (and w') indeed ends in a position of Player A. The second case is symmetric. So $\mathcal{M}_{A,B}$ recognizes $\rightsquigarrow$, hence $\rightsquigarrow$ is regular. Note that $\rightsquigarrow$ is not reflexive, but its reflexive $\sim$ closure is also regular (plug in $\mathcal{M}_{A,B}$ the synchronous transducer for the identity relation). Lemma 4 would also hold for $\sim$, which concludes the proof of Theorem 3.

5 Intermezzo: jumping tree automata

We remind the notions of alternating tree automata and two-way tree automata, and we define *jumping tree automata (JTA)*. For an introduction to the theory of automata on infinite trees see [20].

For a set X, $\mathbb{B}^+(X)$ is the set of positive boolean formulas over X, *i.e.* formulas built with elements of X as atomic propositions and using only connectives $\vee$ and $\wedge$. We also allow for formulas $\top$ and $\bot$, and $\wedge$ has precedence over $\vee$. Elements of $\mathbb{B}^+(X)$ are denoted by $\alpha, \beta \ldots$ Let $D \subseteq [k] \cup \{\epsilon, \uparrow, \diamondsuit, \boxminus\}$ be a set of *directions*. A D-automaton is a tuple $\mathcal{A} = (\Sigma, Q, \delta, q_0, C)$ where Σ is a finite alphabet, Q a finite set of states, $q_0 \in Q$ an initial state, C an accepting condition, and $\delta : Q \times \Sigma \to \mathbb{B}^+(D \times Q)$ a transition function. If D contains $\diamondsuit$ or $\boxminus$ then we additionally require the automaton to be equipped with a binary relation $\rightsquigarrow$ over Σ^*.

We note $D_A = [k]$, $D_\uparrow = D_A \cup \{\epsilon, \uparrow\}$ and $D_\rightsquigarrow = D_A \cup \{\diamondsuit, \boxminus\}$. D_A-automata are *alternating tree automata*, $D_\uparrow$-automata are *two-way alternating tree automata*, and $D_\rightsquigarrow$-automata are *jumping alternating tree automata (JTA)*.

In this work we are not interested in identifying children of a node in a tree, but only in existentially or universally quantifying over them. Therefore, for the sake of readability, we use the abstract directions $D_A = \{\Diamond, \Box\}$ instead of $[k]$, as in alternating automata on graphs [5, 17]. We use notation $[\Diamond, q]$ as a macro for $[1, q] \vee \ldots \vee [k, q]$, and similarly $[\Box, q]$ for $[1, q] \wedge \ldots \wedge [k, q]$. All that we establish on this version of jumping tree automata can be easily adapted to the setting where directions are made explicit.

Acceptance of an input tree $T = (t, \ell)$ in a designated node $x_0 \in t$ by a D-automaton $\mathcal{A}$ is classically defined on a two-player game between Eve (the proponent) and Adam (the opponent): Let $T = (t, \ell)$ be a Σ-labelled tree, $x_0 \in t$, and $\mathcal{A} = (\Sigma, Q, \delta, q_0, C)$ be a D-automaton. We define the game $\mathcal{G}^{x_0}_{\mathcal{A},T} = (V, E, v_0)$: The set of positions is $V = t \times Q \times \mathbb{B}^+(D \times Q)$, the initial position is $(x_0, q_0, \delta(q_0, \ell(x_0)))$, and a position (x, q, α) belongs to Eve if α is of the form $\alpha_1 \vee \alpha_2$, $[\Diamond, q']$ or $[\diamondsuit, q']$; otherwise it belongs to Adam.

Moves in $\mathcal{G}^{x_0}_{\mathcal{A},T}$ are defined by Rules (4a)-(4e).

$$(x, q, \alpha_1 \dagger \alpha_2) \to (x, q, \alpha_i) \qquad \text{where } \dagger \in \{\vee, \wedge\} \text{ and } i \in \{1, 2\} \tag{4a}$$

$$(x, q, [\bigcirc, q']) \to (y, q', \delta(q', \ell(y))) \text{ where } \bigcirc \in \{\Diamond, \Box\} \text{ and } y \text{ is a child of } x \tag{4b}$$

$$(x, q, [\epsilon, q']) \to (x, q', \delta(q', \ell(x))) \tag{4c}$$

$$(x, q, [\uparrow, q']) \to (y, q', \delta(q', \ell(y))) \qquad \text{where } y \text{ is } x\text{'s parent} \tag{4d}$$

$$(x, q, [\ominus, q']) \to (y, q', \delta(q', \ell(y))) \qquad \text{where } \ominus \in \{\diamondsuit, \boxminus\} \text{ and } x \rightsquigarrow y \tag{4e}$$

Positions of the form $(x, q, \top)$ and $(x, q, \bot)$ are deadlocks, winning for Eve and Adam respectively. Positions of the form $(\epsilon, q, [\uparrow, q'])$ are also deadlocks as the root of a tree has no parent; they are winning for Adam.

Most of the time the starting node x_0 will be the root ϵ of the tree, and in this case we simply write $\mathcal{G}_{\mathcal{A},T}$ instead of $\mathcal{G}^{\epsilon}_{\mathcal{A},T}$. The winning condition of $\mathcal{G}^{x_0}_{\mathcal{A},T}$ results from the acceptance condition C of $\mathcal{A}$. In this work, we consider *parity condition*: C is a mapping that assigns to each state of the automaton a natural number called its *colour*, and an infinite play is winning for Eve if the least colour seen infinitely often during the play is even. A tree T is *accepted* by $\mathcal{A}$ if Eve has a winning strategy in $\mathcal{G}_{\mathcal{A},T}$, and we denote by $\mathcal{L}(\mathcal{A})$ the set of trees accepted by $\mathcal{A}$.

We first prove that the class of JTA is closed by complementation, by a construction inspired from classical alternating automata. To this aim we classically define the *dualization* $\widetilde{\alpha}$ of a formula $\alpha \in \mathbb{B}^+(D_{\rightsquigarrow} \times Q)$ by induction as follows: $\widetilde{\top} = \bot$, $\widetilde{\bot} = \top$, $\widetilde{\alpha \vee \beta} = \widetilde{\alpha} \wedge \widetilde{\beta}$, $\widetilde{\alpha \wedge \beta} = \widetilde{\alpha} \vee \widetilde{\beta}$, $\widetilde{[\Diamond, q]} = [\Box, q]$, $\widetilde{[\Box, q]} = [\Diamond, q]$, and, as expected, $\widetilde{[\diamondsuit, q]} = [\boxminus, q]$ and $\widetilde{[\boxminus, q]} = [\diamondsuit, q]$.

▶ **Definition 5.** Let $\mathcal{A} = (\Sigma, Q, \delta, q_0, C)$ be a jumping tree automaton. We define *the complement* of $\mathcal{A}$ by $\widetilde{\mathcal{A}} = (\Sigma, Q, \widetilde{\delta}, q_0, \widetilde{C})$, where $\widetilde{C}(q) = C(q) + 1$, and $\widetilde{\delta}(q, a) = \widetilde{\delta(q, a)}$.

▶ **Lemma 6.** *Eve wins $\mathcal{G}^{x_0}_{\widetilde{\mathcal{A}},T}$ if, and only if, Eve loses $\mathcal{G}^{x_0}_{\mathcal{A},T}$.*

Proof. The arenas of both games are isomorphic, and if a position belongs to Eve in $\mathcal{G}^{x_0}_{\mathcal{A},T}$ then its counterpart in $\mathcal{G}^{x_0}_{\widetilde{\mathcal{A}},T}$ belongs to Adam, and vice versa. Also, a play is winning for a player in one game if and only if its counterpart in the other game is winning for the opponent. From this we have that a winning strategy for a player in one game gives a winning strategy for its opponent in the other, and because parity games are determined [23], the result follows. ◀

We now establish that JTA capture $\boxminus\mathsf{CTL}^*$.

▶ **Theorem 7.** *Let φ be an $\boxminus\mathsf{CTL}^*$ formula and $\rightsquigarrow$ a binary relation over $(2^{AP})^*$. There exists a jumping tree automaton $\mathcal{A}_\varphi$ of size exponential in $|\varphi|$ and equipped with $\rightsquigarrow$ such that $T \in \mathcal{L}(\mathcal{A}_\varphi)$ if, and only if, $T \models \varphi$.*

The following expresses that JTA are adequate machines for the decision problem SUS.

▶ **Theorem 8.** *Let $\mathcal{G}$ be a finite arena, $\rightsquigarrow$ be a binary relation and $\varphi \in \boxminus\mathsf{CTL}^*$. There is a jumping tree automaton $\mathcal{A}$ equipped with $\rightsquigarrow$ such that σ is a $(\rightsquigarrow, \varphi)$-uniform strategy if, and only if, $T_\sigma \in \mathcal{L}(\mathcal{A})$. Moreover, $\mathcal{A}$ can be chosen of size $|\mathcal{A}| = O(|\mathcal{G}| \times 2^{|\varphi|})$.*

Proof. One simply builds from $\mathcal{G}$ a nondeterministic tree automaton $\mathcal{A}_\mathcal{G}$ that accepts the set of strategy trees for Player 1 in $\mathcal{G}$. $\mathcal{A}_\mathcal{G}$ is of size linear in $|\mathcal{G}|$. Then, by Theorem 7, one can build a JTA $\mathcal{A}_\varphi$ equipped with $\rightsquigarrow$ that accepts the models of φ. This automaton is of size $|\mathcal{A}_\varphi| = O(2^{|\varphi|})$. The product $\mathcal{A} = \mathcal{A}_\mathcal{G} \times \mathcal{A}_\varphi$ is thus a JTA that accepts precisely the strategy trees that verify φ. ◀

The following is a direct consequence of Theorems 3 and 8.

▶ **Corollary 9.** *The emptiness problem for jumping tree automata with regular equivalence relations is undecidable.*

6 The special case of recognizable relations

We first show that JTA with recognizable relations (Definition 10) can be effectively transformed into equivalent two-way tree automata of linear size (Theorem 12). We then get two central corollaries: first, the emptiness problem for JTA with recognizable relations is decidable (Corollary 13), and second, the problem SUS becomes decidable when restricted to recognizable relations (Corollary 14) and it is 2ExpTime-complete.

▶ **Definition 10.** A relation $\rightsquigarrow \subseteq \Sigma^* \times \Sigma^*$ is *recognizable* if there is a family of regular languages $\mathcal{L}_1, \mathcal{L}'_1, \ldots, \mathcal{L}_n, \mathcal{L}'_n \subseteq \Sigma^*$ such that $\rightsquigarrow = \bigcup_{i=1}^{n} \mathcal{L}_i \times \mathcal{L}'_i$.

We rather use another, easily shown equivalent, characterization. Let $\overline{w} \in \Sigma^*$ denote the mirror image of a word $w \in \Sigma^*$ (*i.e.* $\overline{\epsilon} = \epsilon$ and $\overline{aw} = \overline{w}a$).

▶ **Proposition 11.** *A relation $\rightsquigarrow$ over Σ^* is recognizable if the language $\mathcal{L}_\rightsquigarrow := \{\overline{u}\#v \mid u \rightsquigarrow v\}$ is regular (accepted by a finite state automaton), with fresh symbol $\# \notin \Sigma$.*

Given a recognizable relation $\rightsquigarrow$, we write $\mathcal{B}_\rightsquigarrow = (\Sigma \cup \{\#\}, Q_\rightsquigarrow, \delta_\rightsquigarrow, s_0, F_\rightsquigarrow)$ for the canonical deterministic finite state automaton of $\mathcal{L}_\rightsquigarrow$, with standard interpretation of the components of $\mathcal{B}_\rightsquigarrow$. We will abuse vocabulary by saying that "$\mathcal{B}_\rightsquigarrow$ *recognizes relation* $\rightsquigarrow$".

A typical example of recognizable relation relates to *information-set-based* strategies in two-player imperfect-information games, where finite plays are related whenever they share the same *information set* according to the player under imperfect information, *i.e.* the set of states the player considers possible after "observing" the course of the game. As in finite arenas information sets are finitely many, a mere powerset construction provides the deterministic finite state automaton that recognizes the information-set-based relation.

A central result of our contribution is an alternative characterization of JTA with recognizable relations.

▶ **Theorem 12.** *Let $\mathcal{A}$ be a jumping tree automaton with a recognizable relation $\rightsquigarrow$. Then, there is a two-way tree automaton $\widehat{\mathcal{A}}$ of size $O(|\mathcal{A}| \times |\mathcal{B}_\rightsquigarrow|)$ such that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\widehat{\mathcal{A}})$.*

We sketch the proof: when JTA $\mathcal{A}$ goes down along a branch of a tree, $\widehat{\mathcal{A}}$ behaves likewise. The critical points are the jump instructions of $\mathcal{A}$, say in a node x of the tree. At this point, $\widehat{\mathcal{A}}$ stops behaving like $\mathcal{A}$ and enters a *jump mode* that simulates this jump: $\widehat{\mathcal{A}}$ triggers automaton $\mathcal{B}_\rightsquigarrow$ and goes up to the root while running $\mathcal{B}_\rightsquigarrow$ on the reversed branch. When reaching the root, $\mathcal{B}_\rightsquigarrow$ has read $\overline{w(x)}$, the mirror of the node word of x, and $\widehat{\mathcal{A}}$ feeds $\mathcal{B}_\rightsquigarrow$ with

the # symbol. Then $\widehat{\mathcal{A}}$ goes down along some or all (depending on the jump: respectively existential or universal) branch(es) of the tree while still running $\mathcal{B}_{\rightsquigarrow}$. Each time $\mathcal{B}_{\rightsquigarrow}$ reaches a final state in a node y, it has read $\overline{w(x)}\#w(y)$, which by Proposition 11 means that $x \rightsquigarrow y$; the jump mode can then be closed. In this case, $\widehat{\mathcal{A}}$ resumes the simulation of $\mathcal{A}$.

From Theorem 12, we immediately infer two significant corollaries.

▶ **Corollary 13.** *The non-emptiness problem for jumping tree automata with recognizable relations is decidable in time exponential in the size of the jumping automaton and of the word automaton recognizing the relation.*

Proof. This is a direct consequence of Theorem 12 along with the EXPTIME complexity of the non-emptiness problem for two-way alternating tree automata [22]. ◀

Also, combining Theorem 8 with Corollary 13 gives a decidable subproblem of SUS.

▶ **Corollary 14.** *The problem*

$$\mathrm{SUS}_{rec} := \left\{ (\mathcal{G}, \mathcal{B}_{\rightsquigarrow}, \varphi) \;\middle|\; \begin{array}{c} \mathcal{G} \text{ is a finite arena, } \rightsquigarrow \text{ is a recognizable relation, } \varphi \in \boxminus\mathsf{CTL}^* \\ \text{and there exists a } (\rightsquigarrow, \varphi)\text{-uniform strategy in } \mathcal{G}. \end{array} \right\}$$

is 2EXPTIME*-complete, and the decision procedure synthesizes a solution strategy (if any).*

Proof. Let $\mathcal{G}$ be a finite arena, $\rightsquigarrow$ be a recognizable relation, and $\varphi \in \boxminus\mathsf{CTL}^*$. By Theorem 8, there is a JTA $\mathcal{A}$ equipped with relation $\rightsquigarrow$ of size $O(|\mathcal{G}| \times 2^{|\varphi|})$ whose language is the tree unfoldings of $(\rightsquigarrow, \varphi)$-uniform strategies. By Corollary 13, emptiness of $\mathcal{L}(\mathcal{A})$ can be decided in time $2^{O(|\mathcal{G}| \times 2^{|\varphi|} \times |\mathcal{B}_{\rightsquigarrow}|)}$, hence $(\mathcal{G}, \mathcal{B}_{\rightsquigarrow}, \varphi) \in \mathrm{SUS}_{rec}$ can be decided in 2EXPTIME. Furthermore, a strategy (if any) is synthesized when checking the non-emptiness of the JTA $\mathcal{A}$: we can make $\mathcal{A}$ an equivalent two-way tree automaton, then an equivalent non-deterministic tree automaton and apply classic algorithms to exhibit a regular tree (if any).

The 2EXPTIME-hardness of SUS_{rec} comes from the 2EXPTIME-completeness of solving LTL games [18]: given an arena $\mathcal{G}$ and an LTL formula ψ, we consider the instance $(\mathcal{G}, \mathcal{B}_{\rightsquigarrow}, A\psi)$ of SUS_{rec}, where $\rightsquigarrow$ is *e.g.* the full relation. Clearly, as there is no $\boxminus$ quantifier in ψ, a strategy in $\mathcal{G}$ which satisfies $A\psi$ is a $(\rightsquigarrow, A\psi)$-uniform strategy in $\mathcal{G}$, and vice versa. ◀

Future work: We want to study the links between jumping tree automata and the logic $\boxminus\mathsf{L}_\mu$, *i.e.* extending the full μ-calculus with the $\boxminus$ quantifier. We conjecture that jumping automata capture $\boxminus\mathsf{L}_\mu$, but the other direction deserves further investigations. Also, in order to handle richer uniformity constraints, we seek for a class of relations for which jumping tree automata languages would exceed the line of ω-regulararity, and still enjoy a decidable emptiness problem. Finally, the notion of uniform strategy generalizes to the case of concurrent game structures, and investigating what results are preserved is yet another interesting perspective, as well as possible extensions to strategic logics, with modalities quantifying over uniform strategies. We foresee that questions of strategy context may become even trickier than usual when rational relations are involved.

References

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

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

3 Jean Berstel. *Transductions and context-free languages*, volume 4. Teubner Stuttgart, 1979.

4 Dietmar Berwanger and Lukasz Kaiser. Information tracking in games on graphs. *Journal of Logic, Language and Information*, 19(4):395–412, 2010.

5 Mikolaj Bojanczyk. Two-way alternating automata and finite models. In Peter Widmayer, Francisco Triguero Ruiz, Rafael Morales Bueno, Matthew Hennessy, Stephan Eidenbenz, and Ricardo Conejo, editors, *ICALP*, volume 2380 of *Lecture Notes in Computer Science*, pages 833–844. Springer, 2002.

6 Samuel Eilenberg. *Automata, languages, and machines*, volume 1. Access Online via Elsevier, 1974.

7 E. A. Emerson and J. Y. Halpern. "Sometimes" and "Not Never" revisited: On branching versus linear time temporal logic. *Journal of the ACM*, 33(1):151–178, 1986.

8 E. Allen Emerson and Charanjit S. Jutla. Tree automata, mu-calculus and determinacy (extended abstract). In *FOCS*, pages 368–377. IEEE Computer Society, 1991.

9 Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. *Reasoning about knowledge*, volume 4. MIT press Cambridge, 1995.

10 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.

11 Joseph Y. Halpern and Moshe Y. Vardi. The complexity of reasoning about knowledge and time. 1. Lower bounds. *Journal of Computer and System Sciences*, 38(1):195–237, 1989.

12 Bakhadyr Khoussainov and Sasha Rubin. Automatic structures: overview and future directions. *Journal of Automata, Languages and Combinatorics*, 8(2):287–301, 2003.

13 Orna Kupferman and Moshe Y. Vardi. Module checking revisited. In Orna Grumberg, editor, *CAV*, volume 1254 of *Lecture Notes in Computer Science*, pages 36–47. Springer, 1997.

14 Bastien Maubert and Sophie Pinchinat. A general notion of uniform strategies. *To appear in International Game Theory Review*, 2013.

15 Bastien Maubert, Sophie Pinchinat, and Laura Bozzelli. The complexity of synthesizing uniform strategies. In Proceedings 1st International Workshop on *Strategic Reasoning,* Rome, Italy, March 16-17, 2013, volume 112 of *Electronic Proceedings in Theoretical Computer Science*, pages 115–122. Open Publishing Association, 2013.

16 Gary Peterson, John H. Reif, and Salman Azhar. Lower bounds for multiplayer noncooperative games of incomplete information. *Computers & Mathematics with Applications*, 41(7):957–992, 2001.

17 Nir Piterman and Moshe Y. Vardi. Global model-checking of infinite-state systems. In Rajeev Alur and Doron Peled, editors, *CAV*, volume 3114 of *Lecture Notes in Computer Science*, pages 387–400. Springer, 2004.

18 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.

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

20 Wolfgang Thomas. Automata on infinite objects. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B)*, pages 133–192. 1990.

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

22 Moshe Y. Vardi. Reasoning about the past with two-way automata. In Kim Guldstrand Larsen, Sven Skyum, and Glynn Winskel, editors, *ICALP*, volume 1443 of *Lecture Notes in Computer Science*, pages 628–641. Springer, 1998.

23 Wiesław Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science*, 200(1–2):135–183, June 1998.