Perspective Games

ORNA KUPFERMAN, School of Computer Science and Engineering, The Hebrew University, Israel
GAL VARDI, School of Computer Science and Engineering, The Hebrew University, Israel, and TTI-Chicago, USA

We introduce and study *perspective games*, which model multi-agent systems in which agents can view only the parts of the system that they own. As in standard multi-player turn-based games, the vertices of the game graph are partitioned among the players. Starting from an initial vertex, the players jointly generate a computation, with each player deciding the successor vertex whenever the generated computation reaches a vertex she owns. A perspective strategy for a player depends only on the history of visits in her vertices. Thus, unlike observation-based models of partial visibility, where uncertainty is longitudinal—players partially observe all vertices in the history, uncertainty in the perspective model is transverse—players fully observe part of the vertices in the history.

We consider deterministic and probabilistic perspective games, with structural (e.g., Büchi or parity) and behavioral (e.g., LTL formulas) winning conditions. For these settings, we study the theoretical properties of the game as well as the decidability and complexity of the problem of deciding whether a player has a winning perspective strategy, in terms of both the game graph and the objectives. We compare perspective strategies with memoryless ones, and study an extension of the temporal logic ATL* with path quantifiers that capture perspective and memoryless strategies.

CCS Concepts: • **Theory of computation → Modal and temporal logics**; **Automata over infinite objects;**

Additional Key Words and Phrases: Multi-agent systems, deterministic and probabilistic games, partial visibility

ACM Reference format:
Orna Kupferman and Gal Vardi. 2024. Perspective Games. *ACM Trans. Comput. Logic* 25, 1, Article 4 (January 2024), 26 pages.
https://doi.org/10.1145/3627705

4

1 INTRODUCTION

Design and control of multi-agent systems correspond to the synthesis of winning strategies in a game that models the interaction between the agents. The game is played on a graph whose paths correspond to computations of the system. We study here settings in which each of the players has control in different parts of the system. Thus, the game is *turn-based*: starting from an initial vertex, the players jointly generate a *play*, namely a path in the graph, with each player deciding the successor vertex when the play reaches a vertex she controls. The objectives of the players in

An extended abstract of this article appeared in the proceedings of LICS 2019 [36].
Authors' addresses: O. Kupferman, School of Computer Science and Engineering, The Hebrew University, Jerusalem, Israel; e-mail: orna@cs.huji.ac.il; G. Vardi, School of Computer Science and Engineering, The Hebrew University, Israel, and TTI-Chicago, USA; e-mail: galvardi@ttic.edu.

1529-3785/2024/01-ART4
https://doi.org/10.1145/3627705

the game refer to the infinite play that they generate. Each objective is a Borel set α in the Cantor topology on V^ω [44], where V is the set of vertices of the graph. In some settings, the specification of α is *structural*: it is specified as an ω-regular condition on V (e.g., Büchi or parity). In other settings, the specification of α is *behavioral*: the vertices in V are labeled by assignments to a set AP of atomic propositions—these with respect to which the system is defined, and α is a language of infinite words in $(2^{AP})^\omega$.

A *strategy* for a player directs her how to continue a play that reaches her vertices. We distinguish between *deterministic* (a.k.a. *pure*) strategies, which choose a successor vertex, and *randomized* strategies, which choose a probability distribution over the successor vertices [20]. We also distinguish between games with *full visibility*, where strategies may depend on the full history of the play, and games with *partial visibility*, where strategies depend only on visible components of the history. The traditional approach to partial visibility assumes *longitudinal uncertainty*, where in all vertices, the players observe the assignment only to an observable subset of the atomic propositions [5, 16, 18, 37, 50]. With longitudinal uncertainty (a.k.a. *observation-based* uncertainty), we model systems in which each of the underlying components can only view and control a subset of the system's variables. For example, a program that interacts with a user with private variables. There, the strategies of the players cannot distinguish between different paths in which the observable atomic propositions behave in the same manner.

We introduce and study *perspective games*, which model *transverse uncertainty*—a new type of partial visibility. As in standard turn-based games, the vertices of the game graph are partitioned among the players, who jointly generate a computation. In a perspective game, the visibility of each player is restricted to her vertices. Thus, a perspective strategy for a player cannot distinguish among histories that differ in visits to vertices owned by other players. Perspective games capture multi-agent systems in which agents can view only the parts of the system that they control. For example, a communication network in which a company that owns part of the routers has to make routing decisions based only on information about visits to its routers [1], a component in a composite reactive system that does not observe the interaction of the environment with the other components [42], and switched systems where components are activated by a scheduler and are not aware of the evolution of the system while being switched off [40, 43]. As another example, consider a system that consists of a few clients that take turns using a server. Each client can observe the current state of the server while using it, but it cannot monitor the server while other clients are using it. Thus, each client has perspective visibility. Perspective games are different from all models of partial visibility studied so far. Indeed, in a perspective game, both players have full visibility on the parts of the system they control, and no visibility (in particular, even no information on the number of transitions taken) on the parts they do not control. Thus, while in games with longitudinal uncertainty players observe all vertices, but partially, in perspective games visibility and lack of visibility are *transverse*—some vertices the players do not see at all, and some they fully see.

Perspective strategies are related to *memoryless* strategies, which depend only on the current vertex of the game. Clearly, every memoryless strategy is perspective. Indeed, the current vertex is visible to the player who controls the vertex. Perspective strategies are also related to *stuttering-invariant strategies* in *asynchronous* games [27, 29]. In these games, the players are unaware of the time that has elapsed (and number of steps that other components have made) between their steps. As the strategies in these games are memoryless, the setting is simpler than that of perspective games. The model closest to perspective games is that of [49], which considers games with partial visibility in an asynchronous setting. There, moves that have no visible effect to a player are hidden from that player. In other words, whenever a play traverses an edge from vertex v to vertex v', then the traversal is visible only to players that can distinguish between v and v'. Thus, perspective games are a special case of the model in [49]. Unlike perspective games, partial visibility in [49] is longitudinal. It is shown in [49] that the problem of deciding whether a player has a winning strategy is decidable, yet the complexity of the described algorithm is not tight. Another related model is the one studied in [42], of control-flow composition. Motivated by software and web services systems, the authors study systems composed from a library of components. Components gain and relinquish control over the computation, and their behavior is independent of the history of the computation. A similar setting is that of switched systems that are turned on and off along the computation [23]. Beyond differences in the visibility model (for example, in [42] a component has no information about the history, including earlier calls to itself, and [23] distinguishes between active and dormant switched systems, depending on the awareness of components to the environment when they are switched off), the problems studied in [23, 42] are different and concern the synthesis of a system from cooperative components. In contrast, we study a game setting, where components have zero-sum objectives.

We start by studying some theoretical aspects of perspective games. We consider two-player games with a winning condition L that is either structural, namely $L \subseteq V^{\omega}$, or behavioral, namely $L \subseteq (2^{AP})^{\omega}$. Player 1 aims for a play whose computation is in L, thus a winning strategy of Player 1 is one that guarantees that the generated play is in L no matter how Player 2 proceeds. We show that in the deterministic setting, perspective strategies (P-strategies) are weaker than ones with full visibility (F-strategies). Thus, there are games that Player 1 wins with an F-strategy yet does not win with a P-strategy. The weakness of P-strategies applies; however, only for Player 1. Thus, Player 1 has a P-strategy that wins against all P-strategies of Player 2 iff Player 1 has a P-strategy that wins against all F-strategies of Player 2. In the probabilistic setting, P-strategies are weaker for both players. Finally, in both the deterministic and probabilistic settings, perspective games are not determined. Thus, there are games in which Player 1 does not have a winning (or almost-surely winning, in the probabilistic setting) P-strategy for L nor Player 2 has a winning P strategy for the complement of L. While the proofs and examples required for establishing the above results are novel, the obtained picture is similar to that known for other studied models of partial visibility [16, 18].

The prime problem when reasoning about games is to decide whether a player has a winning strategy. Here, the differences between perspective games and other models of partial visibility become significant: handling longitudinal uncertainty typically involves some subset-construction-like transformation of the game graph into a game graph of exponential size with full visibility. Accordingly, deciding games with longitudinal uncertainty is EXPTIME-complete in the graph [5, 11, 16, 18]. In perspective games, we can avoid this exponential blow-up in the size of the graph and trade it with an exponential blow-up in the (typically much smaller) winning condition! Essentially, while in the longitudinal setting we have uncertainty about the current position of the graph, which forces an exponential blow-up in the graph, the transverse setting induces uncertainty about the current position of an automaton for the winning condition, which forces an exponential blow-up in the winning condition. Technically, our algorithm constructs an alternating tree automaton that accepts exactly all trees that describe winning P-strategies for Player 1. The branches of the trees correspond to the possible choices of Player 1, and the automaton sends to each branch requirements induced by all the possible behaviors of Player 2. We analyse the complexity of our algorithm for behavioral winning conditions given by deterministic or universal co-Büchi or parity automata, as well as by LTL formulas, and show that the problem is EXPTIME-complete for all above types of automata and is 2EXPTIME-complete for LTL. In all cases, the complexity in terms of the graph is polynomial. As for the lower bound, we show that the problem is EXPTIME-hard already for a fixed-size graph and a winning condition given by a deterministic automaton on finite words. Our lower-bound proof is by a reduction from a linear-space

alternating Turing machine (ATM), and it uses the partial visibility in order to force Player 1, who generates an accepting computation, to respect the transition function of the Turing machine in all positions of the configurations. Indeed, the position that the winning condition checks is decided by Player 2 in a preamble to the game that Player 1 does not observe. To sum up, while solving perspective games is exponentially harder than solving full-visibility games, the exponential blow-up is only in the winning condition. The graph-complexity of perspective games coincides with that of games with no uncertainty and is exponentially lower than that of games with longitudinal uncertainty.

In the probabilistic setting, we show that partial observability enables the players to draw numbers uniformly at random. Essentially, by letting Player j choose a number $x_j \in \{0, \ldots, n-1\}$, we get that $x_1 + x_2 \mod n$ is uniformly distributed in $\{0, \ldots, n-1\}$. This enables perspective games with winning conditions specified by deterministic co-Büchi automata to model the emptiness problem for probabilistic co-Büchi automata. Since the latter problem is undecidable [7, 17], so is the problem of deciding whether Player 1 has a randomized P-strategy that almost-surely wins the corresponding perspective game.

Deciding games with behavioral winning condition is strongly related to ATL* model checking [5]. The temporal logic ATL* offers selective quantification over computations in multi-agents system. Specifically, the path quantifier $\langle\langle A\rangle\rangle$, for a set A of players, ranges over the set of computations that the players in A can force the system into. In particular, the initial vertex of a game G satisfies $\langle\langle$Player 1$\rangle\rangle\psi$ iff Player 1 has a winning strategy in the game G with winning condition ψ. The path quantifier $\langle\langle A\rangle\rangle$ assumes F-strategies. The general extension of ATL* to multi-agents systems with partial visibility is undecidable, as it enables cooperation between players that proceed asynchronously [46, 48]. Numerous decidable settings have been studied [5, 6, 9, 10, 12, 45]. As has been the case with games, uncertainty is longitudinal and results in model-checking algorithms whose graph complexity is exponentially higher than that of ATL* with no uncertainty.

We introduce Perspective-ATL*, which extends ATL* with two new path quantifiers, $\langle\langle A\rangle\rangle_P$ and $\langle\langle A\rangle\rangle_M$, that range over the set of computations that the players in A can force the system into using perspective and memoryless strategies, respectively. We solve the model-checking problem for Perspective-ATL* with two players, and show that it is 2EXPTIME-complete, as is the one for ATL*. For formulas that use only $\langle\langle A\rangle\rangle$ and $\langle\langle A\rangle\rangle_P$ path quantifiers, the graph complexity of our algorithm is only polynomial. Thus, handling transverse uncertainty is exponentially easier than longitudinal uncertainty. On the other hand, model checking of fixed-size formulas with a path quantifier $\langle\langle A\rangle\rangle_M$ is NP-hard, making the graph complexity of Perspective-ATL* hard for NP and co-NP. On the positive side, we show that for Perspective-ATL, which extends ATL with perspective and memoryless path quantification, model-checking is polynomial in both the system and the specification. Essentially, this follows from the fact that the strategies required for satisfying ATL objectives are memoryless, and hence perspective, and so $\langle\langle A\rangle\rangle_P$ and $\langle\langle A\rangle\rangle_M$ coincide with $\langle\langle A\rangle\rangle$. The $\langle\langle A\rangle\rangle_M$ path quantifier can be viewed as a special case of path quantifiers with imperfect recall, namely when the players in A have no uncertainty, but are limited in their memory [52].

Finally, we study perspective games with structural winning conditions. Since memoryless strategies are perspective, decision procedures for games with winning conditions that admit memoryless strategies apply for perspective games. This includes the Büchi, parity, and Rabin conditions. For the generalized Büchi and Streett conditions, which do not admit memoryless strategies, we study the power of P-strategies and show that while the generalized Büchi winning condition does not always admit memoryless strategies, it does admit perspective ones. Thus, the solution of perspective games with a generalized Büchi condition amounts to solving games with full visibility. On the other hand, the Streett winning condition does not admit perspective strategies. Still, we are able to describe an algorithm that decides perspective Streett games whose complexity is exponential in the number of pairs in the Streett condition and only polynomial in the graph. We use the characterization of structural winning conditions that admit perspective strategies in order to point to a fragment of LTL that admits such strategies in behavioral perspective games.

2 PERSPECTIVE GAMES

A *game graph* is a tuple $G = \langle AP, V_1, V_2, v_0, E, \tau \rangle$, where AP is a finite set of atomic propositions, V_1 and V_2 are disjoint sets of vertices, owned by Player 1 and Player 2, respectively, and we let $V = V_1 \cup V_2$. Then, $v_0 \in V_1$ is an initial vertex, which we assume to be owned by Player 1, and $E \subseteq V \times V$ is a total edge relation, thus for every $v \in V$ there is $v' \in V$ such that $\langle v, v' \rangle \in E$. The function $\tau : V \to 2^{AP}$ maps each vertex to a set of atomic propositions that hold in it. The *size* $|G|$ of G is $|E|$, namely the number of edges in it. In the beginning of a play in the game, a token is placed on v_0. Then, in each turn, the player that owns the vertex that hosts the token chooses a successor vertex and moves there the token. A *play* $\rho = v_0, v_1, \ldots$ in G, is an infinite path in G that starts in v_0; thus $\langle v_i, v_{i+1} \rangle \in E$ for all $i \geq 0$. The play ρ induces a *computation* $\tau(\rho) = \tau(v_0), \tau(v_1), \ldots \in (2^{AP})^\omega$. A *game* is a pair $\mathcal{G} = \langle G, L \rangle$, where G is a game graph, and $L \subseteq (2^{AP})^\omega$ is a *behavioral winning condition*, namely an ω-regular language over the atomic propositions, given by an LTL formula or an automaton.[1] Intuitively, Player 1 aims for a play whose computation is in L, while Player 2 aims for a play whose computation is in $comp(L) = (2^{AP})^\omega \setminus L$. Formally, we distinguish between different classes of strategies and objectives for the players, defined below. Sometimes we also refer to games with a *structural winning condition*. There, $L \subseteq V^\omega$ is given by an ω-regular winning condition (e.g., Büchi or parity) over V. Accordingly, the graph G may not be labeled by atomic propositions.

2.1 The Deterministic Setting

Let $\mathsf{Prefs}(G)$ be the set of nonempty prefixes of plays in G. For a sequence $\rho = v_0, \ldots, v_n$ of vertices, let $\mathsf{Last}(\rho) = v_n$. For $j \in \{1, 2\}$, let $\mathsf{Prefs}_j(G) = \{\rho \in \mathsf{Prefs}(G) : \mathsf{Last}(\rho) \in V_j\}$. Thus, $\mathsf{Prefs}_j(G)$ is the subset of $\mathsf{Prefs}(G)$ consisting of prefixes of plays whose last vertex is in V_j. A *strategy* for Player j is a function $f_j : \mathsf{Prefs}_j(G) \to V$ such that for every $\rho \in \mathsf{Prefs}_j(G)$, we have that $\langle \mathsf{Last}(\rho), f_j(\rho) \rangle \in E$. That is, a strategy for Player j maps prefixes of plays that end in a vertex v she owns to a successor of v. For technical convenience, we add ϵ to $\mathsf{Prefs}_1(G)$ and require all strategies f_1 of Player 1 to have $f_1(\epsilon) = v_0$. Thus, all plays start with Player 1 placing the token on v_0. The *outcome* of two strategies f_1 and f_2 of Player 1 and Player 2, respectively, is the play obtained when the players follow the strategies f_1 and f_2. Formally, $\mathsf{Outcome}(f_1, f_2) = v_0, v_1, \ldots$ is such that for all $i \geq 0$, if $v_i \in V_j$, then $v_{i+1} = f_j(v_0, \ldots, v_i)$.

The above definition assumes that both players have full visibility of the play generated by their strategies. We now consider a setting where the visibility of Player j is restricted to vertices she owns. Note that since Player j decides the successor in these vertices, she also knows about visits to the successors, even if she does not own them. Formally, for a prefix $\rho = v_0, \ldots, v_i \in \mathsf{Prefs}(G)$ and $j \in \{1, 2\}$, *the perspective of* Player j *on* ρ, denoted $\mathsf{Persp}_j(\rho)$, is the restriction of ρ to vertices $v_i \in V_j$. We denote the perspectives of Player j on prefixes in $\mathsf{Prefs}_j(G)$ by $\mathsf{PPrefs}_j(G)$. Formally, $\mathsf{PPrefs}_j(G) = \{\mathsf{Persp}_j(\rho) : \rho \in \mathsf{Prefs}_j(G)\}$. Note that $\mathsf{PPrefs}_j(G) \subseteq V_j^*$. A *perspective strategy* for Player j is then a function $f_j : \mathsf{PPrefs}_j(G) \to V$ such that for every $\rho \in \mathsf{PPrefs}_j(G)$, we have that $\langle \mathsf{Last}(\rho), f_j(\rho) \rangle \in E$. That is, a perspective strategy for Player j maps her perspective of prefixes of plays that end in a vertex v she owns to a successor of v. A well known special case of perspective strategies are memoryless ones: The strategy f_j is *memoryless* if for every $\rho \in \mathsf{PPrefs}_j(G)$, we have

[1] In Sections 2.3 and 3, we define LTL and automata formally. For the examples in this section, we specify L in LTL using the temporal operators $\Box$ (always), $\Diamond$ (eventually), and $\bigcirc$ (next).

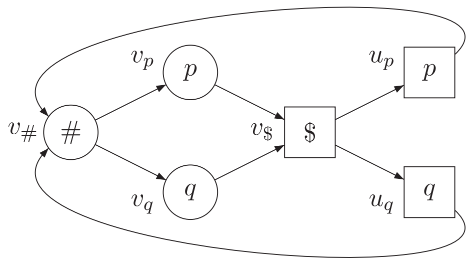

Fig. 1. The game graph G_{match} over $AP = \{p, q, \#, \$\}$. The vertices of Player 1 are circles, and those of Player 2 are squares. The initial vertex is $v_\#$.

that $f_j(\rho)$ depends only on Last(ρ). That is, a strategy is memoryless if it does not distinguish between prefixes of plays that reach the same vertex. Note that a memoryless strategy for Player j can be viewed as a function $f_j : V_j \to V$.

We use F and P to indicate the visibility type of strategies, namely whether they are full (F) or perspective (P). Consider a game $\mathcal{G} = \langle G, L\rangle$. For $\alpha, \beta \in \{F, P\}$, we say that Player 1 (α, β)*-wins* $\mathcal{G}$ if there is an α-strategy f_1 for Player 1 such that for every β-strategy f_2 for Player 2, we have that $\tau(\text{Outcome}(f_1, f_2)) \in L$. Similarly, Player 2 (α, β)*-wins* $\mathcal{G}$ if there is an α-strategy f_2 for Player 2 such that for every β-strategy f_1 for Player 1, we have that $\tau(\text{Outcome}(f_1, f_2)) \notin L$.

Example 1. Consider the game graph G_{match} appearing in Figure 1. Let $\mathcal{G} = \langle G_{match}, \varphi\rangle$ be a game with $\varphi = \Box \Diamond ((p \wedge \bigcirc\bigcirc p) \vee (q \wedge \bigcirc\bigcirc q))$. Thus, Player 1 aims for computations that include infinitely many occurrences of windows of the form $p \cdot \mathbf{true} \cdot p$ or $q \cdot \mathbf{true} \cdot q$. It is easy to see that Player 1 (F, F)-wins $\mathcal{G}$. Indeed, consider a strategy f_1 for Player 1 in which she chooses to proceed from $v_\#$ to v_p whenever the visit to $v_\#$ was preceded by a visit to u_p, and chooses to proceed to v_q whenever the visit to $v_\#$ was preceded by a visit to u_q. Then, for every strategy f_2 of Player 2, the computation $\tau(\text{Outcome}(f_1, f_2))$ satisfies $\psi = \Box(((\$ \wedge \bigcirc p) \to \bigcirc\bigcirc\bigcirc p) \wedge ((\$ \wedge \bigcirc q) \to \bigcirc\bigcirc\bigcirc q))$, which implies φ. In fact, Player 1 also (P, F)-wins $\mathcal{G}$. To see this, consider a strategy f_1' for Player 1 in which she proceeds from $v_\#$ to v_p and v_q alternately. That is, in odd visits to the vertex $v_\#$ she chooses v_p, and in even visits to $v_\#$ she chooses v_q. Then, for every strategy f_2' for Player 2, the computation Outcome(f_1', f_2') is such that every visit in v_p is followed by $u_\$, u_p$ or $u_\$, u_q, v_\#, v_q$, and every visit in v_q is followed by $u_\$, u_q$ or $u_\$, u_p, v_\#, v_p$, guaranteeing that φ is satisfied.

The winning strategies of Player 1 in Example 1 assume full visibility of Player 2. The following theorem states that the visibility type of Player 2 does not matter.

Theorem 2. *For every game $\mathcal{G}$, we have that* Player 1 (F, F)*-wins $\mathcal{G}$ iff* Player 1 (F, P)*-wins $\mathcal{G}$, and* Player 1 (P, F)*-wins $\mathcal{G}$ iff* Player 1 (P, P)*-wins $\mathcal{G}$.*

Proof. Let $\mathcal{G} = \langle G, L\rangle$. First, consider an F or P strategy f_1 of Player 1, and assume that $\tau(\text{Outcome}(f_1, f_2)) \in L$ for every F-strategy f_2 of Player 2. Clearly, $\tau(\text{Outcome}(f_1, f_2)) \in L$ for every P-strategy f_2 for Player 2.

For the other direction, consider an F or P strategy f_1 of Player 1, and assume that we have $\tau(\text{Outcome}(f_1, f_2)) \notin L$ for some F-strategy f_2 of Player 2. Let $\rho = \text{Outcome}(f_1, f_2)$. We define a P-strategy f_2' for Player 2 such that for every prefix ρ' of ρ with Last$(\rho') \in V_2$ we have $f_2'(\text{Persp}_2(\rho')) = f_2(\rho')$. Note that for every two distinct prefixes ρ', ρ'' of ρ with Last(ρ'), Last$(\rho'') \in V_2$, the lengths of Persp$_2(\rho')$ and Persp$_2(\rho'')$ are different, thus f_2' is well defined. Now, as Outcome$(f_1, f_2') = $ Outcome(f_1, f_2), we have that $\tau(\text{Outcome}(f_1, f_2')) \notin L$, and we are done. □

Since the visibility type of Player 2 does not matter, we can omit it from our notation. Namely, for $\alpha \in \{F, P\}$, we say that Player 1 α-wins $\mathcal{G}$ if there is an α-strategy f_1 for Player 1 such that for every strategy f_2 for Player 2 (either F or P), we have that $\tau(\text{Outcome}(f_1, f_2)) \in L$.

On the other hand, the visibility type of Player 1 does matter. That is, F-strategies for Player 1 are strictly stronger than P-strategies. Formally, we have the following:

Theorem 3. *There is a game $\mathcal{G}$ such that* Player 1 *F-wins $\mathcal{G}$ yet* Player 1 *does not P-win $\mathcal{G}$.*

Proof. Let $\mathcal{G} = \langle G_{match}, \psi \rangle$, with $\psi = \Box(((\$ \wedge \bigcirc p) \rightarrow \bigcirc\bigcirc\bigcirc p) \wedge ((\$ \wedge \bigcirc q) \rightarrow \bigcirc\bigcirc\bigcirc q))$. Thus, ψ requires Player 1 to proceed to v_p after visits to u_p and to proceed to v_q after visits to u_q. In Example 1, we showed that Player 1 F-wins $\mathcal{G}$. On the other hand, Player 1 does not P-win $\mathcal{G}$. Indeed, when Player 1 has a perspective visibility, her choices between v_p and v_q are independent of the choices of Player 2. Therefore, for every P-strategy f_1 of Player 1, there is a strategy f_2 of Player 2 such that ψ is not satisfied in $\tau(\text{Outcome}(f_1, f_2))$. □

Games with full visibility are *determined*. That is, for every game $\mathcal{G} = \langle G, L \rangle$, either Player 1 has a strategy that ensures the satisfaction of L, or Player 2 has a strategy that ensures the satisfaction of $comp(L)$ [18, 44]. In the game from the proof of Theorem 3, the strategy f_2 of Player 2 is not winning—it just prevents f_1 from winning. In fact, no strategy of Player 2 is winning. Formally, we have the following:

Theorem 4. *Perspective games are not determined.*

Proof. Consider the game graph G_{match}, and let $\psi = \Box(((\$ \wedge \bigcirc p) \rightarrow \bigcirc\bigcirc\bigcirc p) \wedge ((\$ \wedge \bigcirc q) \rightarrow \bigcirc\bigcirc\bigcirc q))$. As argued above, Player 1 does not P-win $\langle G_{match}, \psi \rangle$. In addition, as Player 1 does F-win $\langle G_{match}, \psi \rangle$, we have that Player 2 does not P-win $\langle G_{match}, \neg\psi \rangle$, and we are done. □

2.2 A Probabilistic Setting

We now extend the setting to a probabilistic one. A probability distribution on a finite set A is a function $\kappa : A \rightarrow [0, 1]$ such that $\sum_{a \in A} \kappa(a) = 1$. The *support* of κ is the set $\text{Supp}(\kappa) = \{a \in A : \kappa(a) > 0\}$. We denote by $\mathcal{D}(A)$ the set of probability distributions on A. A *randomized strategy* for Player j is a function $g_j : \text{Prefs}_j(G) \rightarrow \mathcal{D}(V)$ such that for every $\rho \in \text{Prefs}_j(G)$ and for every $v \in \text{Supp}(g_j(\rho))$, we have $\langle \text{Last}(\rho), v \rangle \in E$. *Perspective randomized strategies* are defined similarly, as $g_j : \text{PPrefs}_j(G) \rightarrow \mathcal{D}(V)$. An *event* is a measurable set $L \subseteq (2^{AP})^\omega$ of computations. Given (possibly perspective) randomized strategies g_1 and g_2 of the two players, the probabilities of events are uniquely defined [32]. Intuitively, the probability of an event L is the probability to obtain a play whose computation is in L. We denote by $Pr_{g_1,g_2}(L)$ the probability of L when the randomized strategies g_1 and g_2 are used. We also use $Pr_{g_1,g_2}(\psi)$, for an LTL formula ψ, referring to the event $L(\psi) = \{w : w \models \psi\}$. It is known that ω-regular languages, and hence also LTL formulas, are measurable [55].

For a randomized strategy g_1 of Player 1, we say that g_1 is an *almost-winning strategy* if $Pr_{g_1,g_2}(L) = 1$ for every randomized strategy g_2 of Player 2. As in the deterministic case, we use F and P to indicate whether the randomized strategies have full or perspective visibility and talk about (α, β)-almost winning, for $\alpha, \beta \in \{F, P\}$.

In games with full visibility for both players, it is known that Player 1 has an almost winning (randomized) strategy iff she has a winning (deterministic) strategy [44]. This no longer holds for perspective games:

Theorem 5. *There is a game $\mathcal{G}$ such that* Player 1 *(P, F)-almost wins $\mathcal{G}$ yet* Player 1 *does not P-win $\mathcal{G}$.*

Proof. Let $\mathcal{G} = \langle G_{match}, \theta_{\#} \rangle$, for $\theta_{\#} = \Diamond ((p \wedge \bigcirc \# \wedge \bigcirc\bigcirc p) \vee (q \wedge \bigcirc \# \wedge \bigcirc\bigcirc q))$. Consider the randomized P-strategy g_1 of Player 1 in which whenever she visits the vertex $v_{\#}$, she moves to the successor v_p with probability $\frac{1}{2}$ and to the successor v_q with probability $\frac{1}{2}$. It is easy to see that g_1 is a (P, F)-almost winning strategy in $\mathcal{G}$. However, for every P-strategy f_1 for Player 1, there is a strategy f_2 for Player 2 such that $\rho(\text{Outcome}(f_1, f_2))$ does not satisfy $\theta_{\#}$, thus Player 1 does not P-win $\mathcal{G}$. □

In Theorem 2, we show that in deterministic games, the visibility type of Player 2 does not matter. In the probabilistic setting, it does matter.

Theorem 6. *There is a game $\mathcal{G}$ such that* Player 1 (P, P)*-almost wins $\mathcal{G}$ yet* Player 1 *does not* (P, F)*-almost win $\mathcal{G}$.*

Proof. Let $\mathcal{G} = \langle G_{match}, \theta_{\$} \rangle$, for $\theta_{\$} = \Diamond ((p \wedge \bigcirc \$ \wedge \bigcirc\bigcirc p) \vee (q \wedge \bigcirc \$ \wedge \bigcirc\bigcirc q))$. Consider the strategy g_1 described in the proof of Theorem 5. Note that g_1 is a (P, P)-almost winning strategy in $\mathcal{G}$. Indeed, for every randomized P-strategy g_2 of Player 2, we have $Pr_{g_1, g_2}(\theta_{\$}) = 1$. On the other hand, for every randomized P-strategy g_1' of Player 1, there is an F-strategy g_2' of Player 2 such that $Pr_{g_1', g_2'}(\theta_{\$}) = 0$. Hence, Player 1 does not have a (P, F)-almost winning strategy in $\mathcal{G}$. □

Finally, as in the deterministic case, it may be that no player almost wins a given perspective game:

Theorem 7. *Perspective games are not almost determined.*

Proof. Consider the game graph G_{match}, and let $\psi = \bigcirc\bigcirc\bigcirc ((p \rightarrow \bigcirc\bigcirc p) \wedge (q \rightarrow \bigcirc\bigcirc q))$. It is easy to see that neither Player 1 almost P-wins $\langle G_{match}, \psi \rangle$ nor Player 2 almost P-wins $\langle G_{match}, \neg\psi \rangle$. □

2.3 Perspective Alternating Temporal Logic

The logic ATL* offers selective quantification over computations in multi-agents systems [5]. There are two types of formulas in ATL*: *state formulas*, whose satisfaction is related to a specific vertex in a game that models the system, and *path formulas*, whose satisfaction is related to a specific computation. Formally, an ATL* state formula is one of the following:

- **(S1)** p, for $p \in AP$.
- **(S2)** $\neg\varphi_1$ or $\varphi_1 \vee \varphi_2$, where φ_1 and φ_2 are ATL* state formulas.
- **(S3)** $\langle\langle A \rangle\rangle \psi$, where $A \subseteq \{$Player 1, Player 2$\}$ is a set of players and ψ is an ATL* path formula.

An ATL* path formula is one of the following:

- **(P1)** An ATL* state formula.
- **(P2)** $\neg\psi_1$ or $\psi_1 \vee \psi_2$, where ψ_1 and ψ_2 are ATL* path formulas.
- **(P3)** $\bigcirc \psi_1$ or $\psi_1 \mathcal{U} \psi_2$, where ψ_1 and ψ_2 are ATL* path formulas.

The logic ATL* consists of the set of state formulas generated by the rules (S1–3). Additional Boolean connectives and temporal operators are defined from $\neg, \vee, \bigcirc$, and $\mathcal{U}$ in the usual manner; in particular, $\Diamond \psi = \mathbf{true} \mathcal{U} \psi$ and $\Box \psi = \neg \Diamond \neg \psi$. The same way CTL is a fragment of CTL*, the logic ATL is the fragment of ATL* that consists of all formulas in which every temporal operator is immediately preceded by a path quantifier. The logic LTL consists of path formulas obtained by applying rules P1-3 above with P1 including only atomic propositions.

The semantics of ATL* is defined with respect to a game graph $G = \langle AP, V_1, V_2, v_0, E, \tau \rangle$. Before we define the semantics, we need some notations. Recall that given two strategies f_1 and f_2 of Player 1 and Player 2, respectively, we define Outcome(f_1, f_2) as the play obtained by letting Player 1 and Player 2 follow f_1 and f_2 from the initial state of G. Here, we adjust the outcome function to get as a parameter a vertex from which the play starts and a set of two strategies, one for each player. Thus, given a vertex v and a set $\mathcal{F} = \{f_1, f_2\}$, of Player 1 and Player 2 strategies, we define $\text{Outcome}(v, \mathcal{F}) = v^0, v^1, \ldots$, where $v^0 = v$ and for all $j \in \{1, 2\}$ and $i \geq 0$, if $v^i \in V_j$, then $v^{i+1} = f_j(v^0, \ldots, v^i)$.

We write $v \models \varphi$ to indicate that vertex v satisfies a state formula φ and write $\rho \models \psi$ to indicate that the play $\rho = \rho_0, \rho_1, \ldots$ satisfies a path formula ψ. For $i \geq 0$, let ρ^i denote the suffix of ρ from position i. Thus, $\rho^i = \rho_i, \rho_{i+1}, \rho_{i+2}, \ldots$. The satisfaction relation $\models$ is defined, for all vertices v and plays ρ of G, inductively as follows:

- **(S1)** $v \models p$, for $p \in AP$, iff $p \in \tau(v)$.
- **(S2)** $v \models \neg\varphi_1$ iff $v \not\models \varphi_1$, and $v \models \varphi_1 \vee \varphi_2$ iff $v \models \varphi_1$ or $v \models \varphi_2$.
- **(S3)** $v \models \langle\langle A \rangle\rangle \psi$ iff there exists a set $\mathcal{F}$ of strategies, one for each player in A, such that for all sets $\tilde{\mathcal{F}}$ of strategies, one for each player not in A, and for all plays $\rho \in (\text{Outcome}(v, \mathcal{F} \cup \tilde{\mathcal{F}}))$, we have $\rho \models \psi$.
- **(P1)** $\rho \models \varphi$ for a state formula φ iff $\rho_0 \models \varphi$.
- **(P2)** $\rho \models \neg\psi_1$ iff $\rho \not\models \psi_1$, and $\rho \models \psi_1 \vee \psi_2$ iff $\rho \models \psi_1$ or $\rho \models \psi_2$.
- **(P3)** $\rho \models \bigcirc \psi_1$ iff $\rho^1 \models \psi_1$, and $\rho \models \psi_1 \mathcal{U} \psi_2$ iff there exists a position $i \geq 0$ such that $\rho^i \models \psi_2$ and for all positions $0 \leq j < i$, we have $\rho^j \models \psi_1$.

For example, the ATL* formula $\langle\langle$Player 1$\rangle\rangle((\Diamond \Box \neg req) \vee (\Box \Diamond grant))$ asserts that Player 1 has a strategy to enforce computations in which either only finitely many requests are sent, or infinitely many grants are given.

The logic Perspective-ATL* extends ATL* by augmenting the $\langle\langle A \rangle\rangle$ path quantifier by parameters on the class of the strategies of the players in A (by Theorem 2, the class of the strategies of the players not in A is not important). In addition to $\langle\langle A \rangle\rangle$, which corresponds to full visibility, we have the path quantifiers $\langle\langle A \rangle\rangle_P$ and $\langle\langle A \rangle\rangle_M$, where the strategies are perspective and memoryless, respectively. Formally, we have the following:

- $v \models \langle\langle A \rangle\rangle_P \psi$ iff there exists a set $\mathcal{F}$ of P-strategies, one for each player in A, such that for all sets $\tilde{\mathcal{F}}$ of (F or P) strategies, one for each player not in A, and for all plays $\rho \in (\text{Outcome}(v, \mathcal{F} \cup \tilde{\mathcal{F}}))$, we have $\rho \models \psi$.
- $v \models \langle\langle A \rangle\rangle_M \psi$ iff there exists a set $\mathcal{F}$ of memoryless strategies, one for each player in A, such that for all sets $\tilde{\mathcal{F}}$ of F-strategies, one for each player not in A, and for all plays $\rho \in (\text{Outcome}(v, \mathcal{F} \cup \tilde{\mathcal{F}}))$, we have $\rho \models \psi$.

In the *model-checking problem* for Perspective-ATL*, we are given a game graph G and a Perspective-ATL* formula ψ, and we must decide whether $v_0 \models \psi$.

We also consider Perspective-ATL, which extends ATL with perspective and memoryless path quantification.

3 AUTOMATA

Given a set D of directions, a *D-tree* is a set $T \subseteq D^*$ such that if $x \cdot c \in T$, where $x \in D^*$ and $c \in D$, then also $x \in T$. The elements of T are called *nodes*, and the empty word ε is the *root* of T. For every $x \in T$, the nodes $x \cdot c$, for $c \in D$, are the *successors* of x. Nodes without successors are called *leaves*. A *path* π of a tree T is a set $\pi \subseteq T$ such that $\varepsilon \in \pi$ and for every $x \in \pi$, either x is a leaf or there exists a unique $c \in D$ such that $x \cdot c \in \pi$. Given an alphabet Σ, a *Σ-labeled D-tree* is a pair $\langle T, \tau \rangle$ where T is a tree and $\tau : T \to \Sigma$ maps each node of T to a letter in Σ.

For a set X, let $\mathcal{B}^+(X)$ be the set of positive Boolean formulas over X (i.e., Boolean formulas built from elements in X using $\wedge$ and $\vee$), where we also allow the formulas **true** and **false**. For a set $Y \subseteq X$

and a formula $\theta \in \mathcal{B}^+(X)$, we say that Y *satisfies* θ iff assigning **true** to elements in Y and assigning **false** to elements in $X \setminus Y$ makes θ true. An *alternating tree automaton* is $\mathcal{A} = \langle \Sigma, D, Q, q_{in}, \delta, \alpha \rangle$, where Σ is the input alphabet, D is a set of directions, Q is a finite set of states, $\delta : Q \times \Sigma \to \mathcal{B}^+(D \times Q)$ is a transition function, $q_{in} \in Q$ is an initial state, and α specifies the acceptance condition (a condition that defines a subset of Q^ω; we define several types of acceptance conditions below). For a state $q \in Q$, we use $\mathcal{A}^q$ to denote the automaton obtained from $\mathcal{A}$ by setting the initial state to be q. The *size* of $\mathcal{A}$, denoted $|\mathcal{A}|$, is the sum of lengths of formulas that appear in δ.

The alternating automaton $\mathcal{A}$ runs on Σ-labeled D-trees. A *run* of $\mathcal{A}$ over a Σ-labeled D-tree $\langle T, \tau \rangle$ is a $(T \times Q)$-labeled $\mathbb{N}$-tree $\langle T_r, r \rangle$. Each node of T_r corresponds to a node of T. A node in T_r, labeled by (x, q), describes a copy of the automaton that reads the node x of T and visits the state q. Note that many nodes of T_r can correspond to the same node of T. The labels of a node and its successors have to satisfy the transition function. Formally, $\langle T_r, r \rangle$ satisfies the following:

(1) $r(\varepsilon) = \langle \varepsilon, q_{in} \rangle$.
(2) Let $y \in T_r$ with $r(y) = \langle x, q \rangle$ and $\delta(q, \tau(x)) = \theta$. Then there is a (possibly empty) set $S = \{(c_0, q_0), (c_1, q_1), \ldots, (c_{n-1}, q_{n-1})\} \subseteq D \times Q$, such that S satisfies θ, and for all $0 \le i \le n-1$, we have $y \cdot i \in T_r$ and $r(y \cdot i) = \langle x \cdot c_i, q_i \rangle$.

For example, if $\langle T, \tau \rangle$ is a $\{0,1\}$-tree with $\tau(\varepsilon) = a$ and $\delta(q_{in}, a) = ((0, q_1) \vee (0, q_2)) \wedge ((0, q_3) \vee (1, q_2))$, then, at level 1, the run $\langle T_r, r \rangle$ includes a node labeled $(0, q_1)$ or a node labeled $(0, q_2)$, and includes a node labeled $(0, q_3)$ or a node labeled $(1, q_2)$. Note that if, for some y, the transition function δ has the value **true**, then y need not have successors. Also, δ can never have the value **false** in a run.

A run $\langle T_r, r \rangle$ is accepting if all its infinite paths satisfy the acceptance condition. Given a run $\langle T_r, r \rangle$ and an infinite path $\pi \subseteq T_r$, let $\inf(\pi) \subseteq Q$ be such that $q \in \inf(\pi)$ if and only if there are infinitely many $y \in \pi$ for which $r(y) \in T \times \{q\}$. That is, $\inf(\pi)$ contains exactly all the states that appear infinitely often in π. We consider here three acceptance conditions defined as follows. A path π satisfies:

— a *Büchi* condition $\alpha \subseteq Q$ if and only if $\inf(\pi) \cap \alpha \neq \emptyset$.
— a *co-Büchi* condition $\alpha \subseteq Q$ iff $\inf(\pi) \cap \alpha = \emptyset$.
— a *parity* condition $\alpha : Q \to \{1, \ldots, k\}$ iff the minimal color $i \in \{1, \ldots, k\}$ for which $\inf(\pi) \cap \alpha^{-1}(i) \neq \emptyset$, is even. The number k of colors in α is called the *index* of the automaton.
— a *Rabin* condition $\alpha = \{\langle \alpha_1, \beta_1 \rangle, \ldots, \langle \alpha_k, \beta_k \rangle\} \subseteq 2^Q \times 2^Q$ iff for some $1 \le i \le k$ we have that $\inf(\pi) \cap \alpha_i \neq \emptyset$ and $\inf(\pi) \cap \beta_i = \emptyset$.

For the three conditions, an automaton accepts a tree iff there exists a run that accepts it. We denote by $L(\mathcal{A})$ the set of all Σ-labeled D-trees that $\mathcal{A}$ accepts.

Below we discuss some special cases of alternating automata. The alternating automaton $\mathcal{A}$ is *nondeterministic* if for all the formulas that appear in δ, if (c_1, q_1) and (c_2, q_2) are conjunctively related, then $c_1 \neq c_2$. (i.e., if the transition is rewritten in disjunctive normal form, there is at most one element of $\{c\} \times Q$, for each $c \in D$, in each disjunct). The automaton $\mathcal{A}$ is *universal* if all the formulas that appear in δ are conjunctions of atoms in $D \times Q$, and $\mathcal{A}$ is *deterministic* if it is both nondeterministic and universal. The automaton $\mathcal{A}$ is a *word* automaton if $|D| = 1$. Then, we can omit D from the specification of the automaton and denote the transition function of $\mathcal{A}$ as $\delta : Q \times \Sigma \to \mathcal{B}^+(Q)$. If the word automaton is nondeterministic or universal, then $\delta : Q \times \Sigma \to 2^Q$, and we often extend δ to sets of states and to finite words: for $S \subseteq Q$, we have that $\delta(S, \epsilon) = S$ and for a word $w \in \Sigma^*$ and a letter $\sigma \in \Sigma$, we have $\delta(S, w \cdot \sigma) = \delta(\delta(S, w), \sigma)$. Sometimes we are interested in reachability via a nonempty path that visits some $\alpha \subseteq Q$. For this, we define $\delta_\alpha : 2^Q \times \Sigma^+ \to 2^Q$ as follows. First, $\delta_\alpha(S, \sigma) = \delta(S, \sigma) \cap \alpha$. Then, for a word $w \in \Sigma^+$, we define

$\delta_\alpha(S, w\cdot\sigma) = \delta(\delta_\alpha(S,w),\sigma) \cup (\delta(S, w\cdot\sigma)\cap\alpha)$. Thus, either α is visited in the prefix of the run that reads w after leaving S, or the last state of the run is in α. It is not hard to prove by an induction on the length of w that for all states $q \in Q$, we have that $q \in \delta_\alpha(S,w)$ iff there is a run from S on w that reaches q and visits α after leaving S. Finally, we say that a Σ-labeled D-tree $\langle T, \tau\rangle$ is *regular* if for all letters $\sigma \in \Sigma$, we have that $\tau^{-1}(\sigma)$ is a regular language over D. Note that a regular tree can be generated by a (D,Σ)-*transducer*: a deterministic automaton over D in which each state is labeled by a letter in Σ. Then, $\tau(x)$, for a node $x \in D^*$, is the letter that labels the transducer state that is reachable by reading x.

We denote each of the different types of automata by three-letter acronyms in $\{D,N,U,A\} \times \{B,C,P,R\} \times \{W,T\}$, where the first letter describes the branching mode of the automaton (deterministic, nondeterministic, universal, or alternating), the second letter describes the acceptance condition (Büchi, co-Büchi, parity, or Rabin), and the third letter describes the object over which the automaton runs (words or trees). For example, APT are alternating parity tree automata and UCT are universal co-Büchi tree automata.

4 DECIDING BEHAVIORAL PERSPECTIVE GAMES IN THE DETERMINISTIC SETTING

In this section, we study the problem of deciding whether Player 1 has a winning perspective strategy in a given behavioral game, thus where the winning condition is given by an automaton or an LTL formula. We also study the complexity of the Perspective-ATL* model-checking problem.

4.1 Upper Bound for Universal Automata

Consider a game graph $G = \langle AP, V_1, V_2, E, v_0, \tau\rangle$. For a vertex $v \in V_2$, a $(V_2^+ \cdot V_1)$-*path from* v is a finite path $v_1, v_2, \ldots, v_k \in V_2^+ \cdot V_1$ in G such that $v_1 = v$. A V_2^ω-*path from* v is an infinite path $v_1, v_2, \ldots \in V_2^\omega$ in G such that $v_1 = v$. When Player 1 moves the token to a vertex $v \in V_2$, the token may traverse a $(V_2^+ \cdot V_1)$-path ρ from v, in which case it returns to V_1 in $\mathsf{Last}(\rho)$, or it may traverse a V_2^ω-path from v, in which case it never returns to a vertex in V_1.

Consider a UCW $\mathcal{U} = \langle 2^{AP}, Q, q_0, \delta, \alpha\rangle$ and a state $q \in Q$. Suppose that the token is placed in some vertex $v \in V_1$ and that the objective of Player 1 is to force the token into computations in $L(\mathcal{U}^q)$. Assume further that Player 1 chooses to move the token to a successor v' of v. We distinguish between two possibilities.

- $v' \in V_1$. Then, the new objective of Player 1 is to force the token from v' into computations in $L(\mathcal{U}^{q'})$, for all the states $q' \in \delta(q, \tau(v))$.
- $v' \in V_2$. Then, we distinguish between two cases.
  - There is a V_2^ω-path ρ from v' and $\tau(\rho) \notin L(\mathcal{U}^{q'})$ for some $q' \in \delta(q,\tau(v))$. We then say that v' *is a trap for* $\langle v, q\rangle$. Indeed, Player 2 can stay in vertices in V_2 and force the token into a computation not in $L(\mathcal{U}^{q'})$, ensuring the desired behavior for Player 1 is not satisfied.
  - v' is not a trap for $\langle v,q\rangle$, in which case, for every $(V_2^+ \cdot V_1)$-path $\rho\cdot v''$ from v', Player 1 should force a token that is placed in v'' into computations in $L(\mathcal{U}^{q'})$, for all states $q' \in \delta(q, \tau(v)\cdot\tau(\rho))$.

The above intuition motivates the following definition of *updated objectives*. For a pair $\langle v,q\rangle \in V_1 \times Q$, standing for an objective of Player 1 to force a token placed on v to be accepted from q, and a choice $v' \in V$ of a successor of v, we define the set $S^{v'}_{v,q} \subseteq (V \times Q \times \{\bot,\top\}) \cup \{\mathit{false}\}$ of updated objectives – these that Player 1 has to satisfy in order to fulfil her $\langle v,q\rangle$ objective after choosing v'. The $\{\bot,\top\}$ flag in the updated objectives is used for tracking visits in α: an updated objective $\langle v'', q', c\rangle \in S^{v'}_{v,q}$ has $c = \top$ if Player 2 can force a visit in α when $\mathcal{U}$ runs from q to q' along a word that labels a path from v via v' to v''. Formally, we define $S^{v'}_{v,q}$ as follows. First, if

v' is a trap for $\langle v, q\rangle$, then $S^{v'}_{v,q} = \{false\}$. Indeed, once Player 1 chooses a vertex that is a trap for $\langle v, q\rangle$, she cannot fullfil her objective. Otherwise, $S^{v'}_{v,q} \subseteq (V \times Q \times \{\bot, \top\})$, and a triple $\langle v'', q', c\rangle$ is in $S^{v'}_{v,q}$ iff one the following holds:

— $v' \in V_1$, $v'' = v'$, and $q' \in \delta(q, \tau(v))$. Then, $c = \top$ iff $q' \in \alpha$.
— $v' \in V_2$, there is a $(V_2^+ \cdot V_1)$-path $\rho \cdot v''$ from v', and $q' \in \delta(q, \tau(v) \cdot \tau(\rho))$. Then, $c = \top$ iff there is a $(V_2^+ \cdot V_1)$-path $\rho \cdot v''$ from v' such that $q' \in \delta_\alpha(q, \tau(v) \cdot \tau(\rho))$.

Note that it may be that v' is not a trap for $\langle v, q\rangle$, yet there is no $(V_2^+ \cdot V_1)$-path from v'. That is, all the paths from v' stay in vertices in V_2 and are in $L(\mathcal{U}^{q'})$ for all $q' \in \delta(q, \tau(v))$. Then, $S^{v'}_{v,q} = \emptyset$.

Theorem 8. *Let $\mathcal{G} = \langle G, \mathcal{U}\rangle$ be a game, where G is a game graph and $\mathcal{U}$ is a UCW. We can construct a UCT $\mathcal{A}_\mathcal{G}$ over V-labeled V_1-trees such that $\mathcal{A}_\mathcal{G}$ accepts a V-labeled V_1-tree $\langle V_1^*, \eta\rangle$ iff $\langle V_1^*, \eta\rangle$ is a winning P-strategy for* Player 1. *The size of $\mathcal{A}_\mathcal{G}$ is polynomial in $|G|$ and $|\mathcal{U}|$.*

Proof. Let $\mathcal{U} = \langle 2^{AP}, Q, q_0, \delta, \alpha\rangle$. We define $\mathcal{A}_\mathcal{G} = \langle V, V_1, Q', q'_0, \delta', \alpha'\rangle$, where

— $Q' = V \times Q \times \{\bot, \top\}$. Intuitively, when $\mathcal{A}_\mathcal{G}$ is in state $\langle v, q, c\rangle$, it accepts strategies that force a token placed on v into a computation accepted by $\mathcal{U}^q$.
— $q'_0 = \langle v_0, q_0, \bot\rangle$.
— For all $\langle v, q, b\rangle \in V \times Q \times \{\bot, \top\}$ and letter $v' \in V$, if $S^{v'}_{v,q} = \{false\}$ or $\neg E(v, v')$, then $\delta'(\langle v, q, b\rangle, v') = false$. Otherwise,

$$\delta'(\langle v, q, b\rangle, v') = \bigwedge_{\langle v'', q', c\rangle \in S^{v'}_{v,q}} (v'', \langle v'', q', c\rangle).$$

Thus, for every updated requirement $\langle v'', q', c\rangle \in S^{v'}_{v,q}$, the automaton sends a copy in state $\langle v'', q', c\rangle$ to direction v''. Note that several updated requirements may be sent to the same direction. Indeed, different $(V_2^+ \cdot V_1)$-paths from v' may induce different words that $\mathcal{U}$ reads from q. Moreover, since $\mathcal{U}$ is universal, it may send copies in different states even for a single word. Note; however, that all states sent to direction v'' agree on their V-element, which is v''. Note also that when $S^{v'}_{v,q} = \emptyset$, we get that $\delta'(\langle v, q, b\rangle, v') = \mathbf{true}$.
— $\alpha' = V \times Q \times \{\top\}$. Recall that a $\top$ flag indicates that Player 2 may reach the Q-element in an updated requirement traversing a path that visits α. Accordingly, the co-Büchi requirement to visit α only finitely many times amounts to a requirement to visit states with $\top$ only finitely many times. □

Theorem 9. *Let $\mathcal{G} = \langle G, \mathcal{U}\rangle$ be a game, where G is a game graph and $\mathcal{U}$ is a UCW. We can construct an NBT $\mathcal{A}'_\mathcal{G}$ over V-labeled V_1-trees such that there is a winning P-strategy for* Player 1 *in $\mathcal{G}$ iff $L(\mathcal{A}'_\mathcal{G})$ is not empty. The size of $\mathcal{A}'_\mathcal{G}$ is polynomial in $|G|$ and exponential in $|\mathcal{U}|$.*

Proof. By Theorem 8, we can construct a UCT $\mathcal{A}_\mathcal{G}$ over V-labeled V_1-trees such that $L(\mathcal{A}_\mathcal{G})$ is not empty iff there is a winning P-strategy for Player 1 in $\mathcal{G}$. The size of $\mathcal{A}_\mathcal{G}$ is polynomial in $|G|$ and $|\mathcal{U}|$. The transformation from $\mathcal{A}_\mathcal{G}$ to $\mathcal{A}'_\mathcal{G}$ can be done by the method of [39]. Below we analyze the construction and show how the fact that $\mathcal{A}_\mathcal{G}$ is deterministic in the V component implies that it is polynomial in $|G|$ and exponential in $|\mathcal{U}|$.

For $k \geq 1$, we denote $[k] = \{1, \ldots, k\}$. The construction in [39] transforms the UCT $\mathcal{A}_\mathcal{G} = \langle V, V_1, Q', q'_0, \delta', \alpha'\rangle$ to an NBT $\mathcal{A}'_\mathcal{G}$ with states $S = 2^{Q' \times [k]} \times 2^{Q' \times [k]}$, where k is such that $|Q'| \cdot k$ bounds the size of an NRT that is equivalent to $\mathcal{A}_\mathcal{G}$, which is exponential in $|Q'|$. Also, for every state $\langle P, O\rangle \in S$, we have $O \subseteq P$, and if $\langle q, i\rangle$ and $\langle q', i'\rangle$ are in P with $q = q'$, then $i = i'$. Therefore, the states in S can be written as $2^{Q'} \times 2^{Q'} \times \mathcal{F}$, where $\mathcal{F}$ is the set of functions $f : Q' \to [k]$. Recall that the states of the UCT $\mathcal{A}_\mathcal{G}$ are $Q' = V \times Q \times \{\bot, \top\}$, and that $\mathcal{A}_\mathcal{G}$ is deterministic in the V-component. Hence, the translation of $\mathcal{A}_\mathcal{G}$ to an NRT is polynomial in $|G|$ and exponential in $|\mathcal{U}|$, and thus k is only polynomial in $|G|$. Also, for every $\langle P, O\rangle \in S$, if $\langle v, q, c, i\rangle$ and $\langle v', q', c', i'\rangle$ are in P, then since $\mathcal{A}_\mathcal{G}$ is deterministic in the V-component, we have $v = v'$. Therefore, the states in S can be written as $V \times 2^{Q\times\{\bot,\top\}} \times 2^{Q\times\{\bot,\top\}} \times \mathcal{F}$, where $\mathcal{F}$ is the set of functions $f : Q\times\{\bot, \top\} \to [k]$. Hence, $|S|$ is polynomial in $|G|$ and exponential in $|\mathcal{U}|$. □

Since the nonemptiness problem for an NBT $\mathcal{A}$ can be solved in quadratic time [56], and we can return a transducer of size $O(|\mathcal{A}|)$ that witnesses the nonemptiness, we can conclude with an upper bound:

Corollary 10. *Deciding whether* Player 1 *P-wins in a perspective game $\langle G, \mathcal{U}\rangle$ for a UCW $\mathcal{U}$ is in EXPTIME. The problem can be solved in time polynomial in $|G|$ and exponential in $|\mathcal{U}|$. Moreover, when* Player 1 *P-wins, the algorithm returns a witness P-strategy by means of a transducer of size polynomial in $|G|$ and exponential in $|\mathcal{U}|$.*

While, as we show in Section 4.3, UCWs are powerful enough in order to reason about objectives in LTL, we now show that Corollary 10 holds, in fact, also when the objective is given by a UPW.

Theorem 11. *Deciding whether* Player 1 *P-wins in a perspective game $\langle G, \mathcal{U}\rangle$ for a UPW $\mathcal{U}$ is in EXPTIME. The problem can be solved in time polynomial in $|G|$ and exponential in $|\mathcal{U}|$. Moreover, when* Player 1 *P-wins, the algorithm returns a witness P-strategy by means of a transducer of size polynomial in $|G|$ and exponential in $|\mathcal{U}|$.*

Proof. In the proof of Theorem 8, we constructed a UCT where the updated objectives included a flag in $\{\bot, \top\}$ to indicate visits in the co-Büchi condition. When the objective is given by a UPW $\mathcal{U}$, we define the updated objective to include the minimal color visited. That is, $S^{v'}_{v,q} \subseteq (V\times Q\times\{1, \ldots, k\})\cup\{false\}$, where k is the index of $\mathcal{U}$, is such that an updated objective $\langle v'', q', c\rangle$ is in $S^{v'}_{v,q}$ if Player 2 can force a run in $\mathcal{U}$ from q to q' in which the minimal color that is visited along a word that labels a path from v via v' to v'' is c. Then, by a construction similar to the one in the proof of Theorem 8, we obtain a UPT $\mathcal{A}_\mathcal{G}$ over V-labeled V_1-trees such that $\mathcal{A}_\mathcal{G}$ accepts a V-labeled V_1-tree $\langle V_1^*, \eta\rangle$ iff $\langle V_1^*, \eta\rangle$ is a winning P-strategy for Player 1. The size of $\mathcal{A}_\mathcal{G}$ is polynomial in $|G|$ and $|\mathcal{U}|$. By [39], UPT emptiness can be reduced to UCT emptiness with a polynomial blowup.[2] Thus, we can obtain a UCT $\mathcal{A}'_\mathcal{G}$ of size polynomial in $|G|$ and $|\mathcal{U}|$, such that $\mathcal{A}'_\mathcal{G}$ accepts a V-labeled V_1-tree $\langle V_1^*, \eta\rangle$ iff $\langle V_1^*, \eta\rangle$ is a winning P-strategy for Player 1. Similar to the UPT $\mathcal{A}_\mathcal{G}$, the UCT $\mathcal{A}'_\mathcal{G}$ is deterministic in the V-component. Hence, we can transform the UCT $\mathcal{A}'_\mathcal{G}$ to an NBT $\mathcal{A}''_\mathcal{G}$ by the method of [39] as shown in the proof of Theorem 9. The size of $\mathcal{A}''_\mathcal{G}$ is polynomial in $|G|$ and exponential in $|\mathcal{U}|$. Now, the claim follows from the fact that the nonemptiness problem for NBT can be solved in quadratic time [56], and we can return a transducer of size $O(|\mathcal{A}''_\mathcal{G}|)$ that witnesses the nonemptiness. □

4.2 Lower Bound for Deterministic Automata

We show an EXPTIME lower bound for *co-safe* specifications. An ω-regular language L is co-safety if for every word $w \in \Sigma^\omega$, if $w \in L$, then w has a prefix x such that $x \cdot y \in L$ for all $y \in \Sigma^\omega$ [4]. It is easy to see that a co-safety language can be recognized by a DBW whose only state in α is an accepting sink, and similarly by a DCW whose only state not in α is an accepting sink. Such

[2]The transformation in [39] is from APT to UCT, and it involves an exponential blowup in the size of the alphabet. The extended alphabet directs the UCT in resolving nondeterminism. Hence, in the case of UPT, the alphabet does not change.

automata are at the bottom of Wagner's hierarchy [58], they are termed *weak[−+]* automata, to indicate the automaton is weak with at most one transition from a rejecting component to an accepting component, and we denote the deterministic weak[−+] word automata by DWW[−+].

Theorem 12. *Let $\mathcal{G} = \langle G, \mathcal{A} \rangle$ be a game, where G is a game graph and $\mathcal{A}$ is a DWW[−+]. Deciding whether* Player 1 *has a winning P-strategy in $\mathcal{G}$ is EXPTIME-hard. Furthermore, it is EXPTIME-hard already for a fixed-size G.*

Proof. We show a reduction from the membership problem for a linear-space ATM. An ATM is a tuple $M = \langle Q_e, Q_u, \Sigma, \Gamma, \Delta, q_{init}, q_{acc}, q_{rej} \rangle$, where Q_e and Q_u are finite sets of *existential* and *universal* states, and we let $Q = Q_e \cup Q_u$. Then, Σ and Γ are input and working alphabets, respectively, with $\Sigma \subseteq \Gamma$, and $\Delta \subseteq Q \times \Gamma \times Q \times \Gamma \times \{L, R\}$ is a transition relation. Finally, q_{init}, q_{acc}, and q_{rej} are the initial, accepting, and rejecting states, respectively, and we assume that $q_{init} \in Q_e$. In the membership problem, we get as input an ATM M and an input word $w \in \Sigma^*$, and we decide whether M accepts w.

A configuration of M describes its state, the content on the working tape, and the location of the reading head. Since M is a linear space ATM, there is some linear function $p : \mathbb{N} \to \mathbb{N}$ such that the number of cells used by the working tape in every configuration of M on its run on w is bounded by $p(|w|)$. We describe a configuration of M by a word $u \cdot q \cdot \gamma \cdot v$, for $u, v \in \Gamma^*$, $\gamma \in \Gamma$, and $q \in Q$. Then, M is in state q, the content of the tape is $u \cdot \gamma \cdot v$, and the reading head points to γ. The initial configuration of M on w, is then $q_{init} \cdot w \cdot \llcorner\!\lrcorner^{\,p(|w|)-|w|}$, for the special letter $\llcorner\!\lrcorner \in \Gamma$. If the current state is q_{acc} or q_{rej}, then the configuration is final and has no successors. Otherwise, the successors of a configuration $u \cdot q \cdot \gamma \cdot v$ are determined by Δ. For every tuple $\langle q, \gamma, q', \gamma', d \rangle \in \Delta$, there is a successor configuration obtained by moving to state q', writing γ' instead of γ, and moving the head one cell to the left or right, depending on d. If $q \in Q_e$, the configuration is *existential*, and M can choose a successor configuration and continue the run from it. If $q \in Q_u$, the configuration is *universal*, and M continues from all successor configurations. Thus, the possible computations of M on w induce a game graph whose vertices are M's configurations, and w is accepted iff Player 1, which proceeds in vertices associated with existential configurations, has a strategy to win from the initial configuration in a reachability game whose target are vertices associated with configurations with q_{acc}. The membership problem for linear-space ATM is EXPTIME-hard already for M of a fixed size, and when Δ alternates between existential and universal states [15], thus $\Delta \subseteq (Q_e \times \Gamma \times Q_u \times \Gamma \times \{L, R\}) \cup (Q_u \times \Gamma \times Q_e \times \Gamma \times \{L, R\})$.

The game graph described above assumes F-strategies and its size is exponential in $|w|$. The big challenge in our reduction is to use P-strategies in order to work with a fixed-size graph. We first describe the main ideas of the reduction, and then describe it formally. The vertices of Player 1 are going to maintain information about the last transition (in particular, the current state of M), but no information about the tape content. The vertices of Player 2 are going to maintain information about the last transition (in particular, the current state of M), and the letter under the tape head. In each Player 1 turn, she chooses a transition in Δ that corresponds to the current state and letter, and moves to a Player 2 vertex accordingly. Since the current letter is not encoded in Player 1's vertices, then Player 1 might lie, but then the DWW[−+] would make sure that she loses the game. Also, the Player 2 vertex that Player 1 chooses to move to must correspond to the current letter. Again, if Player 1 lies about it, then the DWW[−+] makes sure that she loses the game. In a Player 2 turn, she chooses a transition according to the current state and letter—both encoded in her vertices, and moves to a corresponding Player 1 vertex. Recall that the transitions in M alternate between existential and universal states. Accordingly, there is exactly one Player 2 vertex between two Player 1 vertices in the play. This fact enables Player 1 to maintain the tape configuration although she sees only her vertices. Player 1 wins whenever a vertex that corresponds to q_{acc} is reached.

The DWW[−+] $\mathcal{A}$ makes sure that Player 1 does not lie about the current letter, both when choosing her transitions, and when passing the control to Player 2. Since there are exponentially many possible tape contents, $\mathcal{A}$ cannot maintain the full tape content. Instead, $\mathcal{A}$ maintains only the letter in some specific position $0 \leq k \leq p(|w|) - 1$ on the tape. The position k is chosen by Player 2 during a preamble we add to the game. Player 1 does not see the preamble, and thus she does not know k. Accordingly, in order to avoid losing, Player 1 should not lie about any of the tape cells and thus should faithfully simulate the computation of M on w. Hence, Player 1 has a winning P-strategy iff M accepts w.

We now describe the reduction formally. Given an ATM $M = \langle Q_e, Q_u, \Sigma, \Gamma, \Delta, q_{init}, q_{acc}, q_{rej} \rangle$ with a linear-space complexity function $p : \mathbb{N} \to \mathbb{N}$, and a word $w = w_0, w_1, w_2, \ldots \in \Sigma^*$, we construct a game $\mathcal{G} = \langle G, \mathcal{A} \rangle$ such that G is of a fixed size, $\mathcal{A}$ is polynomial in $|w|$, and Player 1 P-wins $\mathcal{G}$ iff M accepts w. For $j \in \{e, u\}$, let $\Delta_j \subset \Delta$ be the transitions from states in Q_j. The game graph $G = \langle AP, V_1, V_2, v_0, E, \tau \rangle$ is defined as follows:

— $AP = \{v_0, u_0, q_{init}\} \cup \Delta$.
— $V_1 = \{v_0, q_{init}\} \cup \Delta_u$. The vertex v_0 is the initial vertex, the vertex q_{init} corresponds to the computation being in state q_{init} before traversing any transition, and vertices $\langle q, \gamma, q', \gamma', d \rangle \in \Delta_u$ correspond to the computation being in state q' after traversing the transition $\langle q, \gamma, q', \gamma', d \rangle$.
— $V_2 = \{u_0\} \cup (\Delta_e \times \Gamma)$. The vertex u_0 is where Player 2 chooses the location k that is monitored by the DWW[−+] $\mathcal{A}$. A vertex $\langle t, \gamma \rangle \in \Delta_e \times \Gamma$ corresponds to a transition $t \in \Delta_e$ chosen by Player 1 at the previous round, and of the letter γ that Player 1 claims to be under the current tape head.
— The set E contains the following edges:
  – $\langle v_0, u_0 \rangle$, $\langle u_0, u_0 \rangle$, and $\langle u_0, q_{init} \rangle$. We call these edges *the preamble* of G.
  – $\langle q_{init}, \langle t, \gamma \rangle \rangle$ for every $t = \langle q_{init}, \gamma_1, q', \gamma_2, d \rangle \in \Delta_e$ and $\gamma \in \Gamma$.
  – $\langle t, \langle t', \gamma \rangle \rangle$, for every $t = \langle q_1, \gamma_1, q_2, \gamma_2, d \rangle \in \Delta_u$, $t' = \langle q_2, \gamma_1', q', \gamma_2', d' \rangle \in \Delta_e$, and $\gamma \in \Gamma$.
  – $\langle \langle t, \gamma_1' \rangle, t' \rangle$, for every $t = \langle q_1, \gamma_1, q_2, \gamma_2, d \rangle \in \Delta_e$ and $t' = \langle q_2, \gamma_1', q', \gamma_2', d' \rangle \in \Delta_u$.

  The self loop at u_0 enables Player 2 to choose how long to stay in u_0. Since Player 1 has perspective visibility, she does not know the number of rounds that Player 2 chooses to stay in u_0. Edges involving transitions describe the computation, to be checked by $\mathcal{A}$.
— For every $v \in \{v_0, u_0, q_{init}\} \cup \Delta_u$, we have $\tau(v) = \{v\}$, and for every $v = \langle t, \gamma \rangle \in \Delta_e \times \Gamma$, we have $\tau(v) = \{t\}$.

The DWW[−+] $\mathcal{A} = \langle \Sigma, S, s_0, \delta, \alpha \rangle$ is defined as follows:

— $\Sigma = \{v_0, u_0, q_{init}\} \cup \Delta$.
— $S = \{s_0, s_{acc}, s_{rej}\} \cup \bigcup_{0 \leq k \leq p(|w|)-1} S^k$, where $S^k = \{\langle k, \gamma, h \rangle : \gamma \in \Gamma, 0 \leq h \leq p(|w|) - 1\}$. A triple $\langle k, \gamma, h \rangle$ in S^k indicates that the letter in position k in the tape is γ, and that the position of the reading head is h. For $0 \leq k \leq |w| - 1$, let $s_0^k = \langle k, w_k, 0 \rangle$, and for $|w| \leq k \leq p(|w|) - 1$, and let $s_0^k = \langle k, \llcorner\!\lrcorner, 0 \rangle$.
— $\delta : S \times \Sigma \to S$ is defined as follows:
 (1) For $s \in \{s_{acc}, s_{rej}\}$ and $\sigma \in \Sigma$, we have $\delta(s, \sigma) = s$.
 (2) For every $s \in S$ and $\sigma \in \{v_0, q_{init}\}$ we have $\delta(s, \sigma) = s$.
 (3) $\delta(s_0, u_0) = s_0^0$, and for $t \in \Delta$ we have $\delta(s_0, t) = s_0$.
 (4) Let $s = s_0^k$. If $0 \leq k \leq p(|w|) - 2$ then $\delta(s, u_0) = s_0^{k+1}$. If $k = p(|w|) - 1$ then $\delta(s, u_0) = s_{acc}$. Thus, the behavior of Player 2 in the preamble of G determines the position k that $\mathcal{A}$ monitors.

(5) For every k and $s \in S^k \setminus \{s_0^k\}$ we have $\delta(s, u_0) = s$.
(6) Let $s = \langle k, \gamma, h \rangle \in S^k$ and $t = \langle q, \gamma_1, q', \gamma_2, d \rangle \in \Delta$. If $d = R$, then we denote $d' = 1$ and otherwise $d' = -1$. Then, we have the following:
 – If $h + d' \notin \{0, \ldots, p(|w|) - 1\}$, then $\delta(s, t) = s_{rej}$.
 – If $h + d' \in \{0, \ldots, p(|w|) - 1\}$, then
  ∗ If $h = k$, then
   · If $\gamma = \gamma_1$ and $q' = q_{acc}$, then $\delta(s, t) = s_{acc}$.
   · If $\gamma = \gamma_1$ and $q' \neq q_{acc}$, then $\delta(s, t) = \langle k, \gamma_2, h + d' \rangle$.
   · If $\gamma \neq \gamma_1$, then $\delta(s, t) = s_{rej}$.
  ∗ If $h \neq k$, then
   · If $q' = q_{acc}$ then $\delta(s, t) = s_{acc}$.
   · If $q' \neq q_{acc}$ then $\delta(s, t) = \langle k, \gamma, h + d' \rangle$.

— $\alpha = \{s_{acc}\}$. □

4.3 Tight Complexities

We are now ready to show tight complexities for the problem of deciding a given perspective game for different classes of behavioral winning conditions. Recall that the input $\mathcal{G} = \langle G, L \rangle$ to the problem has two parameters. Thus, in addition to the *joint complexity* of the problem, namely the complexity in terms of both $|G|$ and the automaton or formula that describes L, we are interested also in its *graph complexity*, namely the complexity in terms of $|G|$, assuming L is given by a fixed-size automaton or formula. Indeed, typically the size of G is much bigger than the size of the winning condition, and is the computational bottleneck [41].

Theorem 13. *Let $\mathcal{G} = \langle G, \psi \rangle$ be a game, where ψ is an LTL formula. Deciding whether* Player 1 *has a winning P-strategy in $\mathcal{G}$, and finding a winning P-strategy, is 2EXPTIME-complete. Furthermore, the problem is 2EXPTIME-hard already for a fixed-size game graph. The graph complexity of the problem is PTIME-complete.*

Proof. For the upper bound, we construct an NBW $\mathcal{A}_{\neg\psi}$ of size exponential in $|\psi|$ such that $L(\mathcal{A}_{\neg\psi}) = \{w : w \not\models \psi\}$ [57]. We dualize $\mathcal{A}_{\neg\psi}$ to obtain a UCW for ψ, and then use Corollary 10.

The lower bound for the graph complexity follows from the PTIME-hardness of alternating reachability [28]. We now prove the 2EXPTIME lower bound. In [5] it is shown that deciding whether Player 1 has a winning F-strategy in a game $\mathcal{G} = \langle G, \psi \rangle$ is 2EXPTIME-hard already for a fixed-size game graph G, and an LTL formula ψ with a fixed number of atomic propositions. The proof in [5] reduces the realizability problem [51] for an LTL formula ψ with a fixed number of atomic propositions to a game $\mathcal{G} = \langle G, \psi \rangle$ with $G = \langle 2^{AP} \times \{1\}, 2^{AP} \times \{2\}, ((2^{AP} \times \{1\}) \times (2^{AP} \times \{2\})) \cup ((2^{AP} \times \{2\}) \times (2^{AP} \times \{1\})), \langle \emptyset, 1 \rangle, AP, \tau \rangle$, where for every $u \in 2^{AP}$ we have $\tau(\langle u, 1 \rangle) = \tau(\langle u, 2 \rangle) = u$. Thus, G alternates between vertices of Player 1 and Player 2. Hence, although Player 1 has perspective visibility, she knows the entire play. Therefore, Player 1 has a winning F-strategy in $\mathcal{G}$ iff she has a winning P-strategy. Therefore, deciding whether Player 1 has a winning P-strategy is 2EXPTIME-hard already for a fixed-size game graph. □

Theorem 14. *Let $\mathcal{G} = \langle G, \mathcal{A} \rangle$ be a game, where $\mathcal{A}$ is a DWW[−+] or UPW. Deciding whether* Player 1 *has a winning P-strategy in $\mathcal{G}$, and finding a winning P-strategy, is EXPTIME-complete. Furthermore, the problem is EXPTIME-hard already for a fixed size game graph. The graph complexity of the problem is PTIME-complete.*

Proof. The EXPTIME lower bound follows from Theorem 12, and the lower bound for the graph complexity follows from the PTIME-hardness of alternating reachability [28]. The upper bound follows from Theorem 11. □

4.4 The Case of Memoryless Strategies

Recall that memoryless strategies are a special case of perspective ones. In this section we study the problem of deciding games in which Player 1 is restricted to memoryless strategies. Surprisingly, the graph complexity in this seemingly simpler setting is NP-complete, thus it is harder than the polynomial graph complexity for perspective strategies.

Theorem 15. *Let $\mathcal{G} = \langle G, L\rangle$ be a game. Deciding whether* Player 1 *has a winning memoryless strategy in $\mathcal{G}$, and finding a winning memoryless strategy, is PSPACE complete when L is given by an LTL formula, and is NP-complete when L is given by a DWW[−+] or a UPW. In both cases, the graph complexity of the problem is NP-complete.*

Proof. We start with the case where L is given by an LTL formula, thus $\mathcal{G} = \langle G, \psi\rangle$. Given $\mathcal{G}$, we guess a memoryless strategy for Player 1 in G. Thus, for each vertex $v \in V_1$, we choose an outgoing edge. Let G' be the graph obtained from G by removing all outgoing edges from vertices in V_1 that are not chosen. It is easy to see that the guessed strategy for Player 1 is winning from vertex v_0 iff all the computations from v_0 in G' satisfy ψ. This can be checked in PSPACE using LTL model checking [53]. Hence, going over all possible memoryless strategies can be done in PSPACE. Hardness in PSPACE follows from LTL model checking. Indeed, when all vertices belong to Player 2, the problem coincides with checking that all computations of $\mathcal{G}$ satisfy ψ.

We continue to the case where L is given by a DWW[−+] or a UPW $\mathcal{A}$. Here too, we guess and check a winning strategy for Player 1. Here, however, we can check a guessed witness in polynomial time by checking the emptiness of the intersection of the language of G' with the complement of $\mathcal{A}$, which is an NPW or a DWW[+-] (that is, a deterministic weak automaton for the safety property obtained by dualizing a DWW[−+]). Hence, the problem is in NP.

We describe a reduction from the problem 2DP (two vertex-disjoint paths), proved to be NP-complete in [24]. In 2DP, we are given a directed graph $G = \langle V, E\rangle$ and two vertices $v, u \in V$, and have to determine whether there are disjoint paths from u to v and from v to u. That is, whether there are paths $s_1, s_2, \ldots, s_k$ and $s'_1, s'_2, \ldots, s'_{k'}$ such that $s_1 = s'_{k'} = u$, $s'_1 = s_k = v$, and $s_1, s_2, \ldots, s_{k-1}, s'_1, s'_2, \ldots, s'_{k'-1}$ are all different.

Given a graph $G = \langle V, E\rangle$, and two vertices $v, u \in V$, define the game graph $G' = \langle AP, V, \emptyset, v, E, \tau\rangle$, where $AP = \{p_1, p_2\}$, $\tau(v) = \{p_1\}$, $\tau(u) = \{p_2\}$, and $\tau(u') = \emptyset$ for all $u' \notin \{v, u\}$. Thus, Player 1 owns all vertices, the initial vertex is v, it is labeled by p_1, and it is the only vertex in which p_1 holds. Also, u is labeled by p_2, and it is the only vertex in which p_2 holds. Now, consider the (co-safety) language $L \subseteq (2^{\{p_1,p_2\}})^\omega$ where $w \in L$ iff w has a prefix in $p_1 \cdot \mathbf{true}^* \cdot p_2 \cdot \mathbf{true}^* \cdot p_1$. Thus, L is the set of plays that satisfy the LTL formula $\psi = p_1 \wedge \bigcirc \diamond (p_2 \wedge \bigcirc \diamond p_1)$. Clearly, L can be defined also by a DWW[−+] or a UPW of a fixed size.

We prove that $\langle G, v, u\rangle \in$ 2DP iff Player 1 wins $\langle G', L\rangle$ using a memoryless strategy. Assume first that $\langle G, v, u\rangle \in$ 2DP. Let $v = s_1, s_2, \ldots, s_k = u = s'_1, s'_2, \ldots, s'_{k'} = v$ be as described above. Consider the memoryless strategy $f : V \to V$ with $f(s_i) = s_{i+1}$, for $1 \le i < k$ and $f(s'_i) = s'_{i+1}$, for $1 \le i < k'$. It is easy to see that Outcome(v, f) is an infinite path that repeatedly traverses disjoint paths from u to v and from v to u. Thus, Outcome$(v, f) \in L$ and Player 1 wins $\langle G', L\rangle$.

For the other direction, assume that Player 1 wins $\langle G', L\rangle$ using a memoryless strategy. Let $f : V \to V$ be a memoryless strategy for Player 1 such that Outcome$(v, f) \in L$. Recall that Outcome$(v, f) = v, f(v), f^2(v), f^3(v), \ldots$. Since Outcome$(v, f)$ is in L, there is a minimal $j > 0$ such that $f^j(v) = v$ and there is $0 < i < j$ such that $f^i(v) = u$. We claim that for all $1 \le l_1 < l_2 \le j$, we have that $f^{l_1}(v) \ne f^{l_2}(v)$. Note that this would complete the proof, as it implies that Outcome(v, f) witnesses that $\langle G, v, u\rangle \in$ 2DP. So, assume by way of contradiction that there are $1 \le l_1 < l_2 \le j$ with $f^{l_1}(v) = f^{l_2}(v)$. Then, Outcome$(v, f) = v, f(v), \ldots (f^{l_1}(v), \ldots f^{l_2-1}(v))^\omega$. We distinguish between two cases. First, if $l_2 = j$, we get a contradiction either to the minimality of j or to the existence of $0 < i < j$ such that $f^i(v) = u$. Then, if $l_2 < j$, we get a contradiction to Outcome(v, f) being in L. □

Remark 1. NP-hardness of the 2DP problem is used in [31] to prove the NP-hardness of the problem of deciding whether Player 1 has a memoryless winning strategy in generalized-Büchi games. Thus, ones with a structural generalized-Büchi winning condition. We could have used a similar idea, or describe a direct reduction from the problem in [31] for the NP-lower bound in the proof of Theorem 15. Indeed, our reduction there is valid also with $\psi = \Box\Diamond p_1 \wedge \Box\Diamond p_2$, which corresponds to the structural generalized-Büchi condition $\{\{v\}, \{u\}\}$. The formula ψ, however, is not co-safety, and the above would have covered only the case L is given by a UPW. Unlike games with a structural generalized-Buchi winning condition, games with a structural DWW[−+] winning condition correspond to reachability games, and deciding whether Player 1 has a memoryless winning strategy in structural DWW[−+] games amounts to deciding the game, which can be done in linear time. In Section 6, we elaborate on memoryless and perspective strategies in games with structural winning conditions. □

Remark 2. In [8], the authors study the complexity of deciding whether an edge-labeled graph contains a simple path that is labeled by a word from a given regular language. Showing that the problem is NP-hard, the authors describe a reduction from 3SAT. The reduction constructs, given a propositional formula θ in 3CNF, a labeled complete directed grid and a regular expression that forces a simple path in the grid to correspond to an assignment satisfying θ. The reduction and its proof are very complicated. The outcome of memoryless strategies are simple lasso-shaped paths. This connection suggests that the reduction we describe in the proof of Theorem 15 can be adjusted for showing NP-hardness of the problem studied in [8]. Below we describe such a reduction.

Given $\langle G, v, u\rangle$, consider the graph G' obtained from G by duplicating v to v_{in} (with edges that enter v in G) and v_{out} (with edges that leave v in G), and labeling edges from v_{out} and to v_{in} by p_1, and duplicating u to u_{in} (with edges that enter u in G) and u_{out} (with edges that leave u in G), and adding an edge labeled p_2 from u_{in} to u_{out}. It is not hard to see that the same arguments used in our proof can be used to show that $\langle G, v, u\rangle$ is in 2DP iff G' has a simple path that starts in v_{out} that is labeled by a word in $p_1 \cdot \mathbf{true}^* \cdot p_2 \cdot \mathbf{true}^* \cdot p_1$. □

4.5 Perspective-ATL* Model Checking

For ATL* and ATL, the model-checking problems are known to be 2EXPTIME-complete and PTIME-complete, respectively, and the graph complexity is PTIME-complete [5]. Adding longitudinal uncertainty makes the graph complexity EXPTIME-complete [5]. As we now show, perspective partial visibility comes at no cost, whereas memoryless one increases the graph complexity.

Theorem 16. *The model-checking problem for* Perspective-*ATL** *is 2EXPTIME-complete. The problem is 2EXPTIME-hard already for a fixed-size game graph. When the only path quantifiers are* $\langle\langle A\rangle\rangle$ *and* $\langle\langle A\rangle\rangle_P$*, the graph complexity is PTIME-complete. With path quantifiers* $\langle\langle A\rangle\rangle_M$*, the graph complexity is in* Δ_2^P*, and is NP- and coNP-hard. The model-checking problem for* Perspective-*ATL is PTIME-complete.*

Proof. We start with Perspective-ATL*. The lower bounds follow from Theorems 13 and 15. In particular, co-NP-hardness for the graph complexity of Perspective-ATL* with $\langle\langle A\rangle\rangle_M$ path quantifiers follows by a reduction from the problem complementing 2DP used in the proof of Theorem 15.

For the upper bound, let G be a game graph and let φ be a Perspective-ATL* formula. As in the algorithm for CTL* model checking [21], we label each vertex v in G by all state subformulas of φ that hold in v. We do this in a bottom-up fashion, starting from the innermost state subformulas of φ. For an innermost state subformula φ_i and a vertex v, we decide whether φ_i holds in v as follows. If $\varphi_i = \langle\langle A \rangle\rangle \psi$ for some $A \subseteq \{\textsc{Player } 1, \textsc{Player } 2\}$, then we use ATL* model-checking [5]. If $\varphi_i = \langle\langle A \rangle\rangle_P \psi$, then we proceed as follows. If $A = \{\textsc{Player } 1\}$ or $A = \{\textsc{Player } 2\}$, then we use Theorem 13. If $A = \{\textsc{Player } 1, \textsc{Player } 2\}$ or $A = \emptyset$, then we claim that $\varphi_i = \exists\psi$ or $\varphi_i = \forall\psi$, and use CTL* model checking. Note that while the equivalences are trivial for $\langle\langle A \rangle\rangle$, they require a proof for $\langle\langle A \rangle\rangle_P$, as it is not clear that P-strategies can force the play to all computations. To see that they can, assume that $v \models \exists\psi$ for a vertex v. So, there is a play $\rho = v_1, v_2, \ldots$ from v that satisfies ψ. Then, Player 1 and Player 2 can cooperate to ensure the satisfaction of ψ using P-strategies f_1 and f_2 such that for every prefix $\rho' = v_1, \ldots, v_i$ of ρ, if $v_i \in V_j$, then $f_j(\text{Persp}_j(\rho')) = v_{i+1}$. Since for every two distinct prefixes ρ' and ρ'' of ρ that end in V_j, the lengths of $\text{Persp}_j(\rho')$ and $\text{Persp}_j(\rho'')$ are not equal, then f_j is well-defined.

We continue to the case $\varphi_i = \langle\langle A \rangle\rangle_M \psi$. As in the upper bound in Theorem 15, we guess a memoryless strategy for the players in A, and check that all the paths in the graph obtained by removing edges that are not chosen by the strategy satisfy ψ. The complexity is PSPACE, with graph complexity NP. Now, when we proceed in a bottom-up fashion, each evaluation of an $\langle\langle A \rangle\rangle_M \psi$ subformula requires such a call, which increases the graph complexity to Δ_2^P.[3]

We continue to Perspective-ATL. Consider a formula of the form $\langle\langle A \rangle\rangle_P \theta$ or $\langle\langle A \rangle\rangle_M \theta$, for θ of the form $\bigcirc \varphi_1$, $\Box \varphi_1$ or $\varphi_1 \mathcal{U} \varphi_2$, for some state formulas φ_1 and φ_2. Note that in all three cases, we have that $\langle\langle A \rangle\rangle_P \theta$ and $\langle\langle A \rangle\rangle_M \theta$ are both equivalent to $\langle\langle A \rangle\rangle \theta$. Indeed, if the players A can ensure the satisfaction of θ by F-strategies, then they can do it using memoryless strategies, and such strategies are also P-strategies. Hence, every Perspective-ATL formula is equivalent to the ATL formula obtained by replacing each $\langle\langle A \rangle\rangle_P$ or $\langle\langle A \rangle\rangle_M$ path quantifiers by $\langle\langle A \rangle\rangle$. The claim then follows from the PTIME-completeness of ATL model-checking [5]. □

Remark 3. While finding the exact graph complexity of model checking Perspective-ATL* formulas with $\langle\langle A \rangle\rangle_M$ path quantifiers is interesting from a complexity-theoretical point of view, it does not contribute much to our story. A possible tightening of our analysis is via the complexity class BH, which is based on a Boolean hierarchy over NP. Essentially, it is the smallest class that contains NP and is closed under union, intersection, and complement [59]. BH is contained in Δ_2^P, and we conjecture that as has been the case with *flow logic* [35], the graph complexity of the fragments of Perspective-ATL* obtained by restricting the number of subformulas of form $\langle\langle A \rangle\rangle_M \psi$ correspond to levels in the BH hierarchy. □

5 DECIDING PERSPECTIVE GAMES IN THE PROBABILISTIC SETTING

In Theorem 5, we showed that there are games in which Player 1 has a (P, F)-almost winning strategy, but no P-winning strategy. In this section, we show that reasoning about perspective games in the probabilistic setting is undecidable for behavioral objectives given by a DCW (and thus also for DPW and NBW).

Theorem 17. *The problems of deciding whether* Player 1 (P, F)*-almost wins and whether she* (P, P)*-almost wins a DCW perspective game are undecidable.*

Proof. We show a reduction from the emptiness problem of probabilistic co-Büchi word automata (PCW, for short), proved to be undecidable [7, 17]. A PCW is $\mathcal{P} = \langle \Sigma, Q, q_0, \delta, \alpha \rangle$, where Σ is the alphabet, Q are the states, q_0 is the initial state, $\alpha \subseteq Q$ is a co-Büchi acceptance condition, and the transition function $\delta : Q \times \Sigma \times Q \to [0, 1]$ is such that for all $q \in Q$ and $\sigma \in \Sigma$, we have $\sum_{q' \in Q} \delta(q, \sigma, q') = 1$. A word $w \in \Sigma^\omega$ is accepted by $\mathcal{P}$ if the acceptance probability of w in $\mathcal{P}$,

[3] The complexity class Δ_2^P includes all problems that can be solved by a deterministic polynomial-time Turing machine that has an oracle to a nondeterministic polynomial-time Turing machine, *a.k.a* P^{NP}.

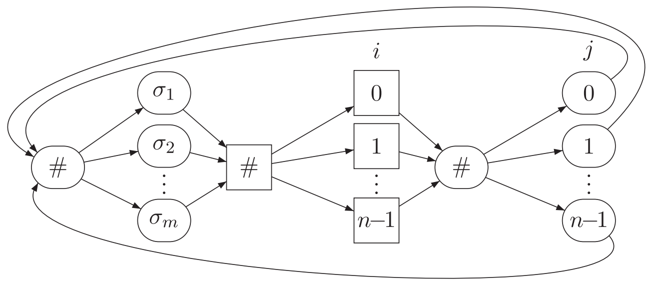

Fig. 2. The game graph G. In every vertex exactly one atomic proposition in $\Sigma \cup \{0, \ldots, n-1\} \cup \{\#\}$ holds.

denoted by $Pr_{\mathcal{P}}(w)$, is 1. The undecidability proof in [7, 17] applies already to PCWs for which the range of δ are rational numbers. Accordingly, we can assume that there is a common divisor r of all probabilities in $\mathcal{P}$, such that $\frac{1}{r}$ is an integer. Let $n = \frac{1}{r}$. By duplicating states in $\mathcal{P}$ we can further assume that all the positive probabilities in $\mathcal{P}$ are equal to $\frac{1}{n}$. Thus, for every $q \in Q$ and $\sigma \in \Sigma$, there are exactly n states, denoted $q^\sigma_0, \ldots, q^\sigma_{n-1}$, such that $\delta(q, \sigma, q^\sigma_k) = \frac{1}{n}$ for every $k \in \{0, \ldots, n-1\}$.

We construct a game $\mathcal{G} = \langle G, \mathcal{A} \rangle$, where $\mathcal{A}$ is a DCW, such that $\mathcal{P}$ is nonempty iff Player 1 (P, F)-almost wins $\mathcal{G}$. Intuitively, the probabilistic transitions of $\mathcal{P}$ are simulated by randomized strategies of the players in $\mathcal{G}$.

The game graph G with $AP = \Sigma \cup \{0, \ldots, n-1\} \cup \{\#\}$ is shown in Figure 2. A play in G has an infinite sequence of rounds, such that in each round Player 1 chooses $\sigma \in \Sigma$, Player 2 chooses an index $i \in \{0, \ldots, n-1\}$, and then Player 1 chooses an index $j \in \{0, \ldots, n-1\}$. Since Player 1 has perspective visibility, the choices of i and j are independent. Accordingly, each player in $\mathcal{G}$ has the possibility to ensure exact simulation of the probabilistic transitions of $\mathcal{P}$ by choosing transitions to the $\{0, \ldots, n-1\}$ vertices in G uniformly at random. Indeed, if Player 2 chooses $i \in \{0, \ldots, n-1\}$ uniformly at random and then Player 1 chooses j without knowing i, then irrespective of the (possibly randomized) choice of j by Player 1, the index $(i + j) \mod n$ is distributed uniformly in $\{0, \ldots, n-1\}$. Likewise, if Player 1 chooses j uniformly at random, then $(i+j) \mod n$ is distributed uniformly no matter what Player 2 does.

The DCW $\mathcal{A} = \langle \Sigma', Q', q'_0, \delta', \alpha' \rangle$ is obtained from $\mathcal{P}$ as follows:

— $\Sigma' = \Sigma \cup \{0, \ldots, n-1\} \cup \{\#\}$.
— $Q' = Q \cup (Q \times \Sigma) \cup (Q \times \Sigma \times \{0, \ldots, n-1\})$.
— $q'_0 = q_0$.
— $\delta' : Q' \times \Sigma' \to Q'$ is defined as follows:
  - For $q \in Q$ and $\sigma \in \Sigma$, we have $\delta'(q, \sigma) = \langle q, \sigma \rangle$.
  - For $\langle q, \sigma \rangle \in Q \times \Sigma$ and $i \in \{0, \ldots, n-1\}$, we have $\delta'(\langle q, \sigma \rangle, i) = \langle q, \sigma, i \rangle$.
  - For $\langle q, \sigma, i \rangle \in Q \times \Sigma \times \{0, \ldots, n-1\}$ and $j \in \{0, \ldots, n-1\}$, we have $\delta'(\langle q, \sigma, i \rangle, j) = q^\sigma_{(i+j) \mod n}$.
  - For every $q \in Q'$ we have $\delta'(q, \#) = q$.
  - For every $q \in Q$ and $k \in \{0, \ldots, n-1\}$ we have $\delta'(q, k) = q$.
  - For every $q \in (Q \times \Sigma) \cup (Q \times \Sigma \times \{0, \ldots, n-1\})$ and $\sigma \in \Sigma$ we have $\delta'(q, \sigma) = q$.
— $\alpha' = \alpha$.

We now prove the correctness of the reduction, namely that $\mathcal{P}$ is nonempty iff Player 1 (P, F)-almost-wins $\langle G, \mathcal{A} \rangle$, and that $\mathcal{P}$ is nonempty iff Player 1 (P, P)-almost-wins.

Assume first that $\mathcal{P}$ is nonempty, and let $w \in L(\mathcal{P})$. Let g_1 be a randomized P-strategy of Player 1 such that the transitions to the Σ vertices are chosen deterministically according to w

(namely, in the kth round Player 1 chooses the kth letter of w), and the choice of $j \in \{0, \ldots, n-1\}$ is done uniformly at random. Since for every random choice of $i \in \{0, \ldots, n-1\}$ by Player 2 the index $(i + j) \mod n$ is distributed uniformly, then for every randomized strategy g_2 of Player 2 we have $Pr_{g_1,g_2}(L(\mathcal{A})) = Pr_{\mathcal{P}}(w) = 1$.

Assume now that $\mathcal{P}$ is empty. Let g_2 be a randomized P-strategy of Player 2 such that $i \in \{0, \ldots, n-1\}$ is chosen uniformly at random. Note that in a randomized P-strategy of Player 1, the choice of j cannot depend on i. Hence, for every randomized P-strategy of Player 1, the index $(i + j) \mod n$ is distributed uniformly. Therefore, for every strategy g_1 of Player 1, we have that $Pr_{g_1,g_2}(L(\mathcal{A}))$ is the probability that $\mathcal{P}$ accepts a word w that is drawn according to some distribution that is induced by g_1. Since for every word w we have $Pr_{\mathcal{P}}(w) < 1$, then $Pr_{g_1,g_2}(L(\mathcal{A})) < 1$.

Thus, we showed that $\mathcal{P}$ is nonempty iff Player 1 (P, F)-almost-wins. In fact, since the strategy g_2 of Player 2 defined above is perspective, then we also have that $\mathcal{P}$ is nonempty iff Player 1 (P, P)-almost-wins. □

6 DECIDING STRUCTURAL PERSPECTIVE GAMES

An extensively studied problem is to determine which games *admit memoryless strategies*, namely, in which games Player 1 has a winning F-strategy iff she has a winning memoryless strategy [54]. Recall that every memoryless strategy is perspective. Indeed, vertices in V_j are included in the perspective viewpoint of Player j. Therefore, games that admit memoryless strategies (for example, reachability, Büchi, co-Büchi, Rabin, and parity games [13]) also admit P-strategies.[4]

In this section, we study which structural winning conditions admit P-strategies. The question is particularly interesting for conditions that do not admit memoryless strategies. Let us recall the generalized-Büchi and Streett winning conditions, which do not admit memoryless strategies. A path $\rho \in V^\omega$ satisfies:

- a *generalized Büchi* condition $\alpha = \{\alpha_1, \ldots, \alpha_k\} \subseteq 2^V$ iff $\inf(\rho) \cap \alpha_i \neq \emptyset$ for all $1 \leq i \leq k$.
- a *Streett* condition $\alpha = \{\langle\beta_1, \alpha_1\rangle, \ldots, \langle\beta_k, \alpha_k\rangle\} \subseteq 2^V \times 2^V$ iff $\inf(\rho) \cap \beta_i \neq \emptyset$ implies $\inf(\rho) \cap \alpha_i \neq \emptyset$ for all $1 \leq i \leq k$.

As we now show, while generalized Büchi games do not admit memoryless strategies, they do admit perspective ones. Intuitively, it follows from the fact that Player 1 can satisfy the different conjuncts of the generalized Büchi condition in a round-robin fashion by maintaining a counter that directs her which conjunct to satisfy next.

Theorem 18. *Generalized-Büchi games admit P-strategies.*

Proof. Let $G = \langle V_1, V_2, v_0, E\rangle$ be a game graph, let $V = V_1 \cup V_2$ with $|V| = n$, and let $\alpha = \{\alpha_1, \ldots, \alpha_k\} \subseteq 2^V$ be the generalized Büchi winning condition. For a vertex v, we denote by G^v the game graph obtained from G by changing the initial vertex to be v. We show that if Player 1 F-wins $\langle G, \alpha\rangle$, then she also P-wins it. Let f_1 be a winning F-strategy for Player 1 in $\langle G, \alpha\rangle$, and let $U \subseteq V$ be the set of vertices that are reachable when Player 1 plays according to f_1. That is, $v \in U$ iff there is a strategy f_2 of Player 2 such that v is in Outcome(f_1, f_2). Note that since U is the set of reachable vertices, a vertex $v \in U \cap V_2$ cannot have successors in $V \setminus U$. Since f_1 is a winning strategy in $\langle G, \alpha\rangle$, and satisfaction of the winning condition is independent of finite prefixes, then f_1 is also a winning strategy in $\langle G^v, \alpha\rangle$, for every $v \in U$. Hence, for every $1 \leq i \leq k$ and for every $v \in U$, the strategy f_1 induces a winning strategy for Player 1 in the Büchi game on G^v

[4]The above also suggests that the study of probabilistic P-strategies for these objectives is not interesting, as the corresponding perspective games are determined, and thus there is no advantage to a probabilistic strategy over a deterministic one.

with the objective $\alpha_i \cap U$. We denote this game by $\mathcal{G}^{v,i} = \langle G^v, \alpha_i \cap U\rangle$. Since Büchi games admit memoryless strategies, Player 1 has a memoryless winning strategy $g_1^{v,i}$ for the Büchi game $\mathcal{G}^{v,i}$ for every $v \in U$ and $1 \leq i \leq k$. Consider the following P-strategy f_1' for Player 1. Starting with $i = 1$, she plays according to $g_1^{v_0,i}$ and maintains a counter that is increased each time the play visits a vertex in V_1. Since $g_1^{v_0,i}$ is winning and memoryless, the play must reach some vertex in $\alpha_i \cap U$ within at most n rounds. When the counter is at least n and the play reaches a vertex $v \in U \cap V_1$, then Player 1 resets the counter, increases i to $(i \mod k) + 1$, and starts playing according to $g_1^{v,i}$ (with the new i). Note that if at some point the play reaches some $v \in \alpha_i \cap U \cap V_2$ and then stays in V_2, then the play must satisfy α according to the definition of U. Also, since vertices in $U \cap V_2$ do not have successors in $V \setminus U$, if Player 2 does not stay in V_2 for infinitely many rounds after visiting $v \in \alpha_i \cap U \cap V_2$, then when the play moves back to a vertex $v' \in V_1$, we have $v' \in U$, and therefore eventually the play reaches $u' \in U \cap V_1$ when the counter is at least n. Hence, f_1' is a winning P-strategy for Player 1 in $\langle G, \alpha\rangle$. □

For a Streett condition, maintaining a counter is not sufficient, as Player 1 should also be aware of vertices that Player 2 may have visited, making P-strategies weaker than ones with full visibility:

Theorem 19. *Streett games do not admit P-strategies.*

Proof. We prove the claim already for Streett games with index 2. Let G_{match} be the game graph from Figure 1, and let $\alpha = \{\langle\{v_p\}, \{u_p\}\rangle, \langle\{u_p\}, \{v_p\}\rangle\}$ be a Streett winning condition. Thus, α requires that the vertex v_p is visited infinitely often iff the vertex u_p is visited infinitely often. It is easy to see that Player 1 has a winning F-strategy, in which from the vertex $v_\#$ she chooses to proceed to v_p whenever the visit to $v_\#$ was preceded by a visit to u_p, and chooses to proceed to v_q whenever the visit to $v_\#$ was preceded by a visit to u_q. However, Player 1 does not have a winning P-strategy. Indeed, a P-strategy for Player 1 in G_{match} must be independent of the choices of Player 2, and for every such strategy of Player 1, Player 2 has a strategy with which α is not satisfied. □

We can now conclude with the complexity of deciding whether Player 1 has a winning P-strategy in all common structural perspective games.

Theorem 20. *Consider a structural perspective game $\mathcal{G} = \langle G, \alpha\rangle$. Let k be the index of α. Deciding whether* Player 1 *P-wins $\mathcal{G}$ can be done in time polynomial in $|G|$ and k for α that is Büchi, co-Büchi, or generalized Büchi, and in time polynomial in $|G|$ and exponential in k for α that is parity, Rabin, or Streett.*

Proof. For Büchi, co-Büchi, generalized Büchi, parity, and Rabin, the complexity follows from known results about games with full visibility[5] [14, 38, 47, 56]. Indeed, all above winning conditions admit P-strategies. For all but generalized Büchi, this follows from the fact they admit memoryless strategies, and for generalized Büchi it follows from Theorem 18.

For Streett, Theorem 19 implies that we should develop a new algorithm. We denote $\alpha = \{\langle\beta_1, \alpha_1\rangle, \ldots, \langle\beta_k, \alpha_k\rangle\}$. First, we reduce the Streett game $\langle G, \alpha\rangle$ to a game $\langle G', \psi\rangle$, where G' is obtained from G by assigning to vertices atomic propositions in $AP = \{p_1, \ldots, p_k, q_1, \ldots, q_k\}$ such that $p_i \in \tau(v)$ iff $v \in \alpha_i$ and $q_i \in \tau(v)$ iff $v \in \beta_i$. Let $\psi = \bigwedge_{1 \leq i \leq k} \Box\Diamond q_i \rightarrow \Box\Diamond p_i$. Clearly, Player 1 has a winning P-strategy in $\langle G, \alpha\rangle$ iff she has a winning P-strategy in $\langle G', \psi\rangle$. Now, by Theorem 11, it is sufficient to show that we can reduce the LTL formula ψ to a UPW $\mathcal{A}$ of size polynomial in k. Essentially, $\mathcal{A}$ consists of k disjoint components, each with three states, and the

[5]Thus, improvements in the classical game setting lead to improvements in perspective games. In particular, for parity it is $O(|G|^{\log(k)+6})$ and for Rabin it is $O(|G|^{k+1} \cdot k!)$.

i-th component is responsible for checking whether the ith conjunct in ψ is satisfied. Formally, a UPW $\mathcal{A}$ such that $L(\mathcal{A}) = L(\psi)$ we constructed as follows:

Recall that ψ specifies a Streett winning condition, thus $\psi = \bigwedge_{1 \leq i \leq k} \square \diamond q_i \rightarrow \square \diamond p_i$. We construct a UPW $\mathcal{A}$ such that $L(\mathcal{A}) = L(\psi)$. The UPW $\mathcal{A}$ consists of k disjoint components, and the ith component is responsible for checking whether the ith conjunct in ψ is satisfied. Every word $w \in (2^{AP})^\omega$ has exactly k runs in $\mathcal{A}$, one run in each component. The ith component consists of three states $S^i = \{s^i_q, s^i_p, s^i_\perp\}$ that monitor visits in vertices with labels q_i and p_i. The parity acceptance condition α' is such that for every $1 \leq i \leq k$ we have $\alpha'(s^i_p) = 2$, $\alpha'(s^i_q) = 3$ and $\alpha'(s^i_\perp) = 4$. If a run π is contained in the ith component then $\inf(\pi) \subseteq S^i$. Thus, α' requires that if a run visits infinitely often in s^i_q then it should also visit infinitely often in s^i_p. Formally, we have $\mathcal{A} = \langle 2^{AP}, S, s_0, \delta, \alpha' \rangle$, where:

- $S = \{s_0\} \cup \bigcup_{1 \leq i \leq k} S^i$.
- $\delta : S \times 2^{AP} \rightarrow S$ is defined as follows. For a letter $\sigma \in 2^{AP}$ we have:
  - $\delta(s_0, \sigma) = \{s^i_\perp : 1 \leq i \leq k\}$. Thus, we start k runs, one for each component S^i.
  - For every $1 \leq i \leq k$ and every $s \in S^i$ we have:
    * If $p_i \in \sigma$ then $\delta(s, \sigma) = \{s^i_p\}$.
    * If $p_i \notin \sigma$ and $q_i \in \sigma$ then $\delta(s, \sigma) = \{s^i_q\}$.
    * If $p_i, q_i \notin \sigma$ then $\delta(s, \sigma) = \{s^i_\perp\}$.
- $\alpha' : S \rightarrow \{1, \ldots, 4\}$ is such that $\alpha'(s_0) = 1$ and for every $1 \leq i \leq k$ we have $\alpha'(s^i_p) = 2$, $\alpha'(s^i_q) = 3$ and $\alpha'(s^i_\perp) = 4$. □

We note that deciding the existence of a winning memoryless strategy in the case of generalized-Büchi or Streett games is NP-complete in the size of the game graph, with NP-hardness applied already for generalized-Büchi winning conditions with index 2 [31]. By Theorem 20, the graph complexity of deciding the existence of winning perspective (rather than memoryless) strategies is exponentially lower.

Finally, the study of structural winning conditions that admit P-strategies also induce fragments of LTL that admit P-strategies in the behavioral setting:

Theorem 21. *Consider a game $\mathcal{G} = \langle G, \psi \rangle$. If ψ is an LTL formula of the form $\bigvee_{1 \leq i \leq k} (\square \diamond p_i \wedge \diamond \square q_i)$ or $\bigwedge_{1 \leq i \leq k} \square \diamond p_i$ for propositional assertions p_i and q_i over AP, then* Player 1 *F-wins $\mathcal{G}$ iff* Player 1 *P-wins $\mathcal{G}$.*

7 DISCUSSION

Traditional partial visibility is longitudinal —in all vertices, the players observe the assignment only to an observable subset of the atomic propositions. We introduced and studied perspective games, which model a new type of partial visibility in multi-agent systems. Perspective games model settings with transverse uncertainty—players observe the assignment to all the atomic propositions, but only in the vertices they own. As discussed in Section 1, transverse uncertainty is present in communication networks, switched systems, and other composite systems in which each of the underlying components can view only the parts of the system it controls. We showed that while transverse uncertainty shares many theoretical properties with longitudinal uncertainty, it is easier to handle. The bottom line of our results is that unless the specification formalism is a deterministic automaton, the complexities of the problems we study coincide with their complexity in a setting with no uncertainty, and for all classes of specifications, the complexity stays polynomial in the size of the game graph. Intuitively, this follows from the fact that the transverse setting adds uncertainty only about the state of an automaton that follows the play traversed by the token, and such uncertainty anyway exists in nondeterministic or alternating automata. This is in contrast with longitudinal uncertainty, where algorithms are exponential in the game graph.

Synchronizing uncertainty due to partial visibility with uncertainty due to branches in the automaton is not easy. From a technical point of view, we left open the complexity of deciding perspective games with objectives given by NBWs. In order to handle LTL, our results on UCWs suffice. It would still be interesting to investigate the complexity for NBWs, as the challenge there is similar to challenges one faces in other problems in which a solution for LTL is based on its exponential translation to UCW, which is not legitimate when the starting point is an NBW (cf., Safraless synthesis [30, 39]). Another problem we left open is the decidability of probabilistic perspective games with LTL objectives. Indeed, the undecidability result for DCWs uses undecidability results known for probabilistic co-Büchi automata, and which have no temporal-logic analogue. Additional future work includes an extension of the results to *strategy logic* [19], as well as the development of symbolic algorithms for solving perspective games, which proved interesting and useful in the longitudinal setting [16, 18].

From a more conceptual point of view, we see several interesting directions for future research. The extension to games with more than two players is straightforward if the game stays *zero-sum*, thus the objectives of the players form a partition of $(2^{AP})^\omega$. Once we consider settings in which each player has her own objective, and the objectives may overlap, the game becomes *non-zero-sum*, and we care about its stable outcomes. Non-zero-sum games with longitudinal uncertainty are studied in [22, 26]. The combination of rationality with uncertainty is computationally challenging. For example, deciding whether a Nash Equilibrium exists in games with three or more players is undecidable [26]. In a follow-up work, [34] studies non-zero-sum games with transverse uncertainty, namely perspective multi-player games. It is shown there that transverse uncertainty leads to undecidability in settings with three or more players that include coalitions or non-zero-sum objectives. On the positive side, in two-player non-zero-sum perspective games, finding and reasoning about stable outcomes is decidable, and in fact, unlike the case with longitudinal uncertainty, can be done in the same complexity as in games with full visibility. Another interesting issue in both types of uncertainty is games with *coalitions*. There, one should distinguish between collaboration that is reflected in joint objectives and collaboration that is reflected in sharing of knowledge. Several extensions to ATL* formalize different collaboration schemes in the context of partial visibility [10, 25], possibly in conjunction with other restrictions on the architecture or the strategies of the players [2, 3]. We conjecture that the computational advantage of the perspective viewpoint is carried over to these models. The computational advantage also motivates a characterization of LTL properties that admit perspective strategies. The general problem is PSPACE-hard, with an easy reduction from LTL satisfiability, and the challenge is to extend the syntactic fragment we describe in Section 6.

Finally, a variant of perspective games in which uncertainty is reduced are perspective games *with notifications*, studied in [33]. There, uncertainty is still transverse, yet a player may be notified about events that happen between visits in vertices she owns. It would be interesting to combine longitudinal and transverse uncertainty. Thus, to consider games where a player can view only the parts of the system she controls, and even in them, her visibility is partial. We conjecture that reasoning about such games is exponential in both the graph and its winning condition.

REFERENCES

[1] S. Agarwal, M. S. Kodialam, and T. V. Lakshman. 2013. Traffic engineering in software defined networks. In *Proceedings of the 32nd IEEE International Conference on Computer Communications.* 2211–2219.
[2] Th. Ågotnes. 2006. Action and knowledge in alternating-time temporal logic. *Synthese* 149, 2 (2006), 375–407.
[3] N. Alechina, B. Logan, N. Hoang Nga, and A. Rakib. 2009. A logic for coalitions with bounded resources. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence.* 659–664.

[4] B. Alpern and F.B. Schneider. 1987. Recognizing safety and liveness. *Distributed Computing* 2, 3 (1987), 117–126.

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

[6] B. Aminof, G. De Giacomo, A. Murano, and S. Rubin. 2018. Synthesis under assumptions. In *Proceedings of the 16th International Conference on Principles of Knowledge Representation and Reasoning*. 615–616.

[7] C. Baier, N. Bertrand, and M. Größer. 2008. On decision problems for probabilistic Büchi automata. In *Proceedings of the 11th International Conference on Foundations of Software Science and Computation Structures*. Springer, 287–301.

[8] C. Barrett, R. Jacob, and M. Marathe. 2000. Formal-language-constrained path problems. *SIAM Journal on Computing* 30, 3 (2000), 809–837.

[9] F. Belardinelli, A. Lomuscio, A. Murano, and S. Rubin. 2017. Verification of broadcasting multi-agent systems against an epistemic strategy logic. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, Vol. 17. 91–97.

[10] R. Berthon, B. Maubert, A. Murano, S. Rubin, and M.Y. Vardi. 2017. Strategy logic with imperfect information. In *Proceedings of the 32nd IEEE Symposium on Logic in Computer Science*. 1–12.

[11] D. Berwanger, K. Chatterjee, M. De Wulf, L. Doyen, and T. A. Henzinger. 2010. Strategy construction for parity games with imperfect information. *Information and Computation* 208, 10 (2010), 1206–1220.

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

[13] T. Cachat. 2002. Two-way tree automata solving pushdown games. In *Proceedings of the Automata Logics, and Infinite Games.* E. Grädel, W. Thomas, and T. Wilke (Eds.), Lecture Notes in Computer Science, Vol. 2500, Springer, 303–317.

[14] C. S. Calude, S. Jain, B. Khoussainov, W. Li, and F. Stephan. 2017. Deciding parity games in quasipolynomial time. In *Proceedings of the 49th ACM Symposium on Theory of Computing*. 252–263.

[15] A. K. Chandra and L. J. Stockmeyer. 1976. Alternation. In *Proceedings of the 17th IEEE Symposium on Foundations of Computer Science*. 98–108.

[16] K. Chatterjee and L. Doyen. 2010. The complexity of partial-observation parity games. In *Proceedings of the 16th International Conference on Logic for Programming Artificial Intelligence and Reasoning*. Springer, 1–14.

[17] K. Chatterjee, L. Doyen, H. Gimbert, and T.A. Henzinger. 2010. Randomness for free. In *Proceedings of the 35th International Symposium on Mathematical Foundations of Computer Science*. Springer, 246–257.

[18] K. Chatterjee, L. Doyen, T. Henzinger, and J.-F. Raskin. 2007. Algorithms for omega-regular games with imperfect information. *Logical Methods in Computer Science* 3, 3 (2007).

[19] K. Chatterjee, T. A. Henzinger, and N. Piterman. 2010. Strategy logic. *Information and Computation* 208, 6 (2010), 677–693.

[20] L. de Alfaro, T.A. Henzinger, and O. Kupferman. 1998. Concurrent reachability games. In *Proceedings of the 39th IEEE Symposium on Foundations of Computer Science*. 564–575.

[21] E.A. Emerson and C.-L. Lei. 1987. Modalities for model checking: Branching time logic strikes back. *Science of Computer Programming* 8, 3 (1987), 275–306.

[22] E. Filiot, R. Gentilini, and J.-F. Raskin. 2018. Rational synthesis under imperfect information. In *Proceedings of the 33rd IEEE Symposium on Logic in Computer Science*. ACM, 422–431.

[23] D. Fisman and O. Kupferman. 2009. Reasoning about Finite-State Switched Systems. In *Peroceedings of the 5th International Haifa Verification Conference,* Lecture Notes in Computer Science, Vol. 6405. Springer, 71–86.

[24] S. Fortune, J. Hopcroft, and J. Wyllie. 1980. The directed subgraph homeomorphism problem. *Theoretical Computer Science* 10, 2 (1980), 111 – 121.

[25] V. Goranko and W. Jamroga. 2016. State and path coalition effectivity models of concurrent multi-player games. *Autonomous Agents and Multi-Agent Systems* 30, 3 (2016), 446–485.

[26] J. Gutierrez, G. Perelli, and M. J. Wooldridge. 2018. Imperfect information in Reactive Modules games. *Information and Computation* 261, 4 (2018), 650–675.

[27] J.Y. Halpern and M.Y. Vardi. 1986. The complexity of reasoning about knowledge and time: Extended abstract. In *Proceedings of the 18th ACM Symposium on Theory of Computing*. ACM, 304–315.

[28] N. Immerman. 1981. Number of quantifiers is better than number of tape cells. *Journal of Computer and Systems Science* 22, 3 (1981), 384–406.

[29] W. Jamroga, W. Penczek, P. Dembinski, and A.W. Mazurkiewicz. 2018. Towards partial order reductions for strategic ability. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems.* 156–165.

[30] B. Jobstmann and R. Bloem. 2006. Optimizations for LTL synthesis. In *Proceedings of the 6th International Conference on Formal Methods in Computer-Aided Design.* 117–124.

[31] B. Jobstmann, A. Griesmayer, and R. Bloem. 2005. Program repair as a game. In *Proceedings of the 17th International Conference on Computer Aided Verification,* Lecture Notes in Computer Science, Vol. 3576. 226–238.

[32] J.G. Kemeny, J.L. Snell, and A.W. Knapp. 1976. *Denumerable Markov Chains.* Springer.

[33] O. Kupferman and N. Shenwald. 2020. Perspective games with notifications. In *Proceedings of the 40th Conference on Foundations of Software Technology and Theoretical Computer Science*, Vol. 182. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 51:1–51:16.
[34] O. Kupferman and N. Shenwald. 2021. Perspective multi-player games. In *Proceedings of the 36th IEEE Symposium on Logic in Computer Science*. 1–13.
[35] O. Kupferman and G. Vardi. 2019. Flow logic. *Logical Methods in Computer Science* 15, 4 (2019).
[36] O. Kupferman and G. Vardi. 2019. Perspective games. In *Proceedings of the 34th IEEE Symposium on Logic in Computer Science*. 1 – 13.
[37] O. Kupferman and M.Y. Vardi. 2000. Synthesis with incomplete information. In *Advances in Temporal Logic*. Kluwer Academic Publishers, 109–127.
[38] O. Kupferman and M.Y. Vardi. 2004. From complementation to certification. In *Proceedings of the 10th International Conference on Tools and Algorithms for the Construction and Analysis of Systems,* Lecture Notes in Computer Science, Vol. 2988. Springer, 591–606.
[39] O. Kupferman and M.Y. Vardi. 2005. Safraless decision procedures. In *Proceedings of the 46th IEEE Symposium on Foundations of Computer Science*. 531–540.
[40] D. Liberzon. 2003. *Switching in Systems and Control*. Birkhauser.
[41] O. Lichtenstein and A. Pnueli. 1985. Checking that finite state concurrent programs satisfy their linear specification. In *Proceedings of the 12th ACM Symposium on Principles of Programming Languages*. 97–107.
[42] Y. Lustig and M.Y. Vardi. 2013. Synthesis from component libraries. *Software Tools for Technology Transfer* 15, 5–6 (2013), 603–618.
[43] M. Margaliot. 2006. Stability analysis of switched systems using variational principles: An introduction. *Automatica* 42, 12 (2006), 2059–2077.
[44] D.A. Martin. 1975. Borel determinacy. *Annals of Mathematics* 102, 2 (1975), 363–371.
[45] B. Maubert and A. Murano. 2018. Reasoning about knowledge and strategies under hierarchical information. In *Proceedings of the 16th International Conference on Principles of Knowledge Representation and Reasoning*.
[46] G.L. Peterson and J.H. Reif. 1979. Multiple-person alternation. In *Proceedings of the 20th IEEE Symposium on Foundations of Computer Science*. 348–363.
[47] N. Piterman and A. Pnueli. 2006. Faster solutions of rabin and streett games. In *Proceedings of the 21st IEEE Symposium on Logic in Computer Science*. 275–284.
[48] A. Pnueli and R. Rosner. 1990. Distributed reactive systems are hard to synthesize. In *Proceedings of the 31st IEEE Symposium on Foundations of Computer Science*. 746–757.
[49] B. Puchala. 2010. Asynchronous omega-regular games with partial information. In *Proceedings of the 35th International Symposium on Mathematical Foundations of Computer Science*. Springer, 592–603.
[50] J.H. Reif. 1984. The complexity of two-player games of incomplete information. *Journal of Computer and Systems Science* 29, 2 (1984), 274–301.
[51] R. Rosner. 1992. *Modular Synthesis of Reactive Systems*. Ph.D. Dissertation. Weizmann Institute of Science, Israel. UMI Order Number: DP17517.
[52] P.Y. Schobbens. 2004. Alternating-time logic with imperfect recall. *Electronic Notes in Theoretical Computer Science* 85, 2 (2004), 82–93.
[53] A.P. Sistla and E.M. Clarke. 1985. The complexity of propositional linear temporal logic. *Journal of ACM* 32, 3 (1985), 733–749.
[54] W. Thomas. 1995. On the synthesis of strategies in infinite games. In *Proceedings of the 12th Symposium on Theoretical Aspects of Computer Science,* Lecture Notes in Computer Science, Vol. 900. Springer, 1–13.
[55] M.Y. Vardi. 1985. Automatic verification of probabilistic concurrent finite-state programs. In *Proceedings of the 26th IEEE Symposium on Foundations of Computer Science*. 327–338.
[56] M.Y. Vardi and P. Wolper. 1986. Automata-theoretic techniques for modal logics of programs. *Journal of Computer and Systems Science* 32, 2 (1986), 182–221.
[57] M.Y. Vardi and P. Wolper. 1994. Reasoning about infinite computations. *Information and Computation* 115, 1 (1994), 1–37.
[58] K. Wagner. 1979. On ω-regular sets. *Information and Control* 43, 2 (1979), 123–177.
[59] G. Wechsung. 1985. On the boolean closure of NP. In *Proceedings of the International Conference on Fundamentals of Computation Theory,*Lecture Notes in Computer Science, Vol. 199. Springer, 485–493.

Received 17 January 2023; accepted 14 September 2023