Temporal Explorability Games

Pete Austin
University of Liverpool, UK

Sougata Bose
UMONS – Université de Mons, Belgium

Nicolas Mazzocchi
Slovak University of Technology in Bratislava, Slovakia

Patrick Totzke
University of Liverpool, UK

Abstract

Temporal graphs extend ordinary graphs with discrete time that affects the availability of edges. We consider solving games played on temporal graphs where one player aims to explore the graph, i.e., visit all vertices. The complexity depends majorly on two factors: the presence of an adversary and how edge availability is specified.

We demonstrate that on static graphs, where edges are always available, solving explorability games is just as hard as solving reachability games. In contrast, on temporal graphs, the complexity of explorability coincides with generalized reachability (NP-complete for one-player and PSPACE-complete for two player games). We show that if temporal graphs are given symbolically, even one-player reachability (and thus explorability and generalized reachability) games are PSPACE-hard. For one player, all these are also solvable in PSPACE and for two players, they are in PSPACE, EXP and EXP, respectively.

2012 ACM Subject Classification Theory of computation → Network games; Theory of computation → Representations of games and their complexity; Mathematics of computing → Graphs and surfaces; Theory of computation → Formal languages and automata theory

Keywords and phrases Temporal Graphs, Explorability, Reachability, Games

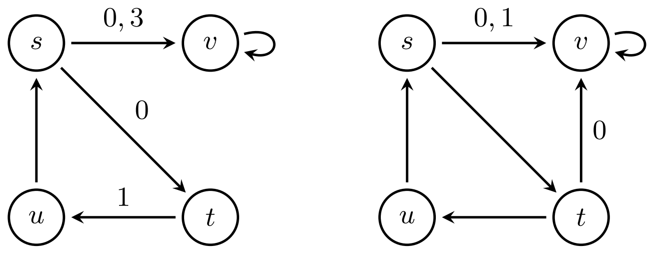
Digital Object Identifier 10.4230/LIPIcs.CONCUR.2025.7

Funding This work has been supported by the EPSRC, projects EP/V025848/1 and EP/X042596/1 and by the Fonds de la Recherche Scientifique – FNRS under Grant n° T.0188.23 (PDR ControlleRS). The research was mainly performed when N. Mazzocchi was affiliated with the Institute of Science and Technology Austria where he was financed by the ERC-2020-AdG 101020093.

1 Introduction

Two player zero-sum games on graphs are a common formalism in formal verification, especially reactive synthesis (see e.g. [21, 16, 34, 35]). The two players jointly determine an infinite path stepwise, where the owner of the current vertex gets to extend the path to a valid successor. One player aims to satisfy a given winning condition, such as reaching a target vertex, and the opposing player aims to prevent that. Under very general conditions, that are satisfied here, such games are determined, meaning that exactly one player has a winning strategy that guarantees a favourable outcome for them no matter their opponent's choices. Solving a game refers to the algorithmic task to determine which player has a winning strategy.

Temporal graphs are a way to model dynamic systems: they extend graphs with discrete and global time and can specify for each edge at which times it can be traversed. Temporal graphs are typically given as sequence of graphs over the same vertex set, an encoding we refer to as *explicit*. For our purposes this is polynomially equivalent to encoding temporal



(a) Arena $\mathcal{A}_1$: explorable from s, not from t, u, v.

(b) Arena $\mathcal{A}_2$: not explorable from any vertex.

(c) Arena $\mathcal{A}_3$: explorable from s but the order is not enforceable.

Figure 1 Examples of games on temporal graphs. Vertices owned by Player 1 are drawn as circles, those owned by Player 2 as diamonds. The labels on edges denote the times at which they are available. Edges without labels are always available.

graphs as ordinary graphs in which (directed) edges carry an explicit list of timestamps, which we also refer to as explicit encodings[1]. Alternatively, and more interestingly, we consider succinct symbolic representations where the availability of edges is given as logical predicate in the existential fragment of Presburger Arithmetic.

In this paper we study the computational complexity of solving games played on temporal graphs (where players' choices are required to respect the temporal constraints on edges). Our focus lies on *explorability* games, where Player 1 wins if and only if all vertices of the graph are visited. Explorability generalizes reachability conditions in the sense that reachability games straightforwardly and in logarithmic space reduce to explorability games. On the other hand, explorability is a special case of so-called *generalized reachability* conditions of [17], where several target sets are given and at least one vertex must be reached in each.

For the benefit of readers unfamiliar with temporal graphs, or turn-based games, we start by providing some intuition to help appreciate the difficulties caused by the dynamic changes of the arena, as well as the presence of antagonistic choice. Consider first the one-player explorability game played on the arena $\mathcal{A}_1$ in Figure 1a. Starting in s, Player 1 wins by exploring the graph as $s \to t \to u \to v$. From any other vertex there is no exploring path due to the non-availability of edges at time 0. An easy property that implies non-explorability of static graphs is the presence of two pairwise non-reachable vertices. Indeed, if two such vertices exists then any path can at most see one of them, hence not explore the graph. This can be turned into a full characterization of explorability even in the presence of an opponent (see Lemma 4). However, due to the temporal constraints on edges, this cannot naïvely be extended to temporal graphs. For example, arena $\mathcal{A}_1$ is explorable yet v is not reachable from t starting at time 0, and vice versa. On the other hand, in arena $\mathcal{A}_2$ (Figure 1b), for every pair of vertices at least one can reach the other starting at time 0. Yet, from no initial vertex and at no time, $\mathcal{A}_2$ is explorable.

Figure 1c demonstrates that even if Player 1 wins the explorability game (as here from vertex s), she may not be able to enforce visiting the vertices in a particular order, as that can be influenced by her opponent's moves. Dually, as for the game starting in vertex s in $\mathcal{A}_2$, even if Player 2 wins, he may not be able to dictate which vertex is left unexplored.

[1] This is trivial for timestamps in unary encoding; Allowing binary-encoded timestamps in the input does not impact the complexity of solving explorability games since winning plays necessarily use consecutive times and thus cannot exist if the edge relation is too sparse.

Related Work. Temporal graphs have been used to analyse dynamic networks and distributed systems in dynamic topologies, such as dissemination/propagation of information [9, 12, 24, 36] or the spread of diseases [38]. There is a large body of mathematical work that considers temporal generalizations of various graph-theoretic notions and properties [11, 18, 31]. Related algorithmic questions include graph colouring [28], travelling salesman [32], maximal matchings [27], and checking the existence of temporal cliques [29], Eulerian circuits [7], vertex-cover sets [3], and explorability [1, 2, 14] (called exploration). Several prior works on explorability assume structural properties that ensure that the graph is explorable. Questions then focus on the minimal time to explore. Pelc [33] provides several sufficient conditions on temporal graphs to be explorable. Spirakis and Michail [32] showed that without assumptions on the graph, checking explorability can be done in linear time for static graphs and is NP-complete for temporal graphs. In most of the works, a path in the temporal graph is allowed to wait at a vertex for an unbounded amount of time.

The edge relation is often deliberately left unspecified and sometimes only assumed to satisfy some weak assumptions about connectedness, frequency, or fairness to study the worst or average cases in uncontrollable environments. Depending on the application, one distinguishes between "online" questions (e.g. [19, 26]), where the edge availability is revealed stepwise, as opposed to the "offline" variant where all is given in advance. We refer to [13, 22, 30] for overviews of temporal graph theory and its applications.

Fijalkow and Horn [17] study games with *generalized reachability* objectives, which generalize not only reachability but also explorability games. Generalized reachability conditions are conjunctions of reachability conditions: *Player* 1 aims to reach at least one vertex out of each of several target sets. Explorability thus corresponds to the case where every vertex forms a singleton target set. Solving generalized reachability games is PSPACE-complete [17], but in polynomial time if all target sets are singletons.

Reachability and parity games played on (symbolically represented) temporal graphs have been introduced in [6]; solving these games is PSPACE-complete. The lower bound was shown by reduction from the emptiness problem for unary alternating finite automata, crucially relying on the presence of antagonistic choice. As in [6], our notion of temporal graphs assumes that some edge must be traversed at every unit time. Waiting (not moving the token for a while) as occasionally permitted [30, 3] can be modelled by adding self-loops.

Turn-based games involving temporal constraints have also been studied in the context of games played on the configuration graphs of timed automata [4]. Solving timed parity games is complete for EXP [8, 25] and the lower bound already holds for reachability games on timed automata with only two clocks [23]. However, the time in temporal graphs is discrete as opposed to the continuous time semantics in timed automata. A direct translation of (games on) temporal graphs to equivalent timed automata games requires two clocks: one to hold the global time used to check the edge predicate and one to ensure that time progresses one unit per step. Fearnley and Jurdziński [15] showed that the reachability problem (solving one-player reachability games) for two-clock timed automata is already PSPACE-hard. Our lower bound constructions have a similar flavour but are incomparable in that they do not re-prove nor are implied by their result. They make crucial use of resetting of clocks which is impossible in our model. Our construction in turn uses either antagonistic choice (Theorem 8) or the more powerful transition guards in symbolic representations (Theorem 9).

Contributions. We study the complexity of solving explorability games and contrast the worst-case complexity for related decision questions for games played on static graphs versus temporal graphs. It turns out that explorability is no harder than reachability on static

		Static	Explicit	Symbolic
One-player	**Reachability**	NL-complete [5, Theorem 4.18]	NL-complete [Theorem 6]	**PSPACE-complete** [Corollary 14]
	Explorability	**NL-complete** [Theorem 5]	**NP-complete** [Theorem 7]	**PSPACE-complete** [Corollary 14]
	Gen. Reach	NP-complete [17, Theorem 3]	NP-complete [Theorem 6]	**PSPACE-complete** [Corollary 14]
Two-player	**Reachability**	P-complete [21]	P-complete [Theorem 6]	PSPACE-complete [6, Theorem 2]
	Explorability	**P-complete** [Theorem 5]	**PSPACE-complete** [Theorem 8]	**PSPACE-hard; In EXP** [Corollary 14]
	Gen. Reach	PSPACE-complete [17, Theorem 1]	PSPACE-complete [Theorem 6]	**PSPACE-hard; In EXP** [Corollary 14]

Figure 2 A table detailing the computational complexities for the one and two-player variants of reachability, explorability, and generalized reachability games played on static, explicitly and symbolically represented temporal graphs. New results are in boldface.

graphs whereas on temporal graphs, explorability games exhibit the hardness of generalized reachability when the temporal edge availability is given explicitly. For temporal explorability games with a succinct, symbolic representation, we have PSPACE-hardness for both variants but a PSPACE-EXP complexity gap for two-player games. Specifically,

1. on static graphs, solving explorability games is complete for polynomial time (and NL-complete for one-player games);
2. on explicitly represented temporal graphs, explorability games are PSPACE-complete (NP-complete in the one-player variant);
3. reachability and thus explorability games on symbolically represented temporal graphs are PSPACE-hard even in the single-player variant. This strengthens the known lower bound from [6] which required a second antagonistic player.

Figure 2 summarizes these and related claims. Our most involved constructions are the PSPACE lower bounds, both from the satisfiability of quantified Boolean formulae (QBF).

2 Notations

▶ **Definition 1** (Temporal Graphs). *A temporal graph $G = (V, E)$ is a directed graph where V are vertices and $E : V^2 \to 2^{\mathbb{N}}$ is the edge availability relation that maps each pair of vertices to the set of times at which the respective directed edge can be traversed. If $i \in E(u, v)$ we call v an i-*successor *of u and write $u \xrightarrow{i} v$. We call G* static *if for all $u, v \in V$, $E(u, v)$ is either $\mathbb{N}$ or $\emptyset$. Its* horizon *is $h(G) = \sup_{u,v \in V}(E(u, v))$, that is, the largest finite time at which any edge is available, or ∞ if no such finite time exists.*

Naturally, one can unfold a temporal graph G into its *expansion* up to time $T \in \mathbb{N} \cup \{\infty\}$, which is the static graph G_T with vertices $V \times \{0, 1, \ldots, T, T+1\}$ and $(u, \theta) \to (v, \theta + 1)$ if and only if $\theta \in E(u, v)$. We denote by *the expansion* of a temporal graph G, its expansion up to its horizon $h(G)$. See for instance Figure 3 for the expansion of a temporal graph $\mathcal{A}_1$ (up to its horizon $h(\mathcal{A}_1) = 3$).

Complexity bounds for decision problems about temporal graphs very much depend on the representation of the input. We will say a temporal graph is represented *explicitly* if it is given as sequence of directed graphs, one per unit time from 0 up to its (finite) horizon.

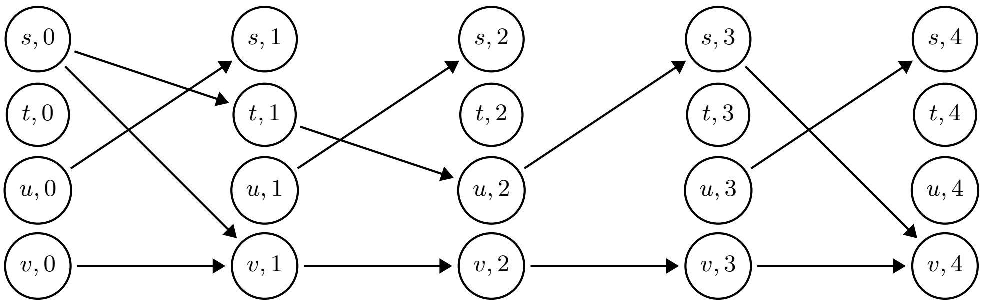


Figure 3 The expansion of $\mathcal{A}_1$.

We also consider a *symbolic* representation where the edge relation E is represented as a formula in the existential fragment of Presburger Arithmetic ($\exists$PA), the first-order theory over natural numbers with equality and addition. That is, the formula $\Phi_{u,v}(x)$ with one free variable x represents the set of times at which an edge from u to v is available as $E(u,v) = \{n \mid \Phi_{u,v}(n) \equiv \mathit{true}\}$. We use common syntactic sugar including inequality and multiplication with constants.

In this representation, checking if an edge is available at a given time, i.e., checking whether a given $\exists$PA formula is satisfied by a given valuation, is NP-complete (and in polynomial time if the number of quantifiers are fixed) [37, Corollary 1]. The symbolic representation of a temporal graph can be exponentially more succinct than the explicit one: using repeated doubling one can express exponentially large values. This representation is also at most exponentially larger because the Presburger-definable edge relation must be ultimately periodic with base and period at most exponential [20].

▶ **Definition 2** (Games on graphs). *A* game *is played by two opposing players. It consists of a directed graph (V, E), a partitioning $V = V_1 \uplus V_2$ of the vertices into those controlled by Player 1 and Player 2 respectively, and a* winning condition*. We refer to $\mathcal{A} = (V_1, V_2, E)$ as the* arena *of the game.*

The game starts with a token on an initial vertex $s_0 \in V$ and is turn-based, where in round j, the owner of the vertex occupied by the token moves it to some successor. This way an infinite path $\rho = s_0 s_1 \ldots$ called a play *is generated based on choices made by each player given the current round and vertex. A play is won by Player 1 if it satisfies the given winning condition, and by Player 2 otherwise.*

A strategy *for Player i is a recipe for how to move. It is a function $\sigma_i : V^* V_i \to V$ from finite paths ending in a vertex s in V_i to some successor. A strategy is* winning *from $s \in V$ if Player i wins every play that starts in s and during which all decisions are according to σ_i.*

We call a vertex s winning for Player i if there exists a winning strategy from s, and call the subset of all such vertices the *winning region* for that player. The main algorithmic question is to *solve* a game, meaning in other words to compute the winning regions for a given arena and winning condition. We consider the following winning conditions.

- *Reachability* towards a given set $F \subseteq V$ of target vertices. This is satisfied by those plays that eventually visit a vertex in F.
- *Generalized Reachability* towards target sets $F_1, F_2, \ldots, F_k \subseteq V$. This is satisfied by those plays that visit at least one vertex of every target set F_i.
- *Explorability* is the condition that asks that eventually every vertex in V is visited.

Note that explorability is a special case of generalized reachability where every vertex $s_i \in V$ corresponds to one target set $F_i = \{s_i\}$.

We study games played on the expansion of a given temporal graph $G = (V, E)$, where the ownership of vertices and winning condition are defined on the underlying temporal graph G and lifted to *the expansion* of G. That is, for any time θ and $u \in V_i$, the vertex (u, θ) is owned by Player i. Similarly, (generalized) reachability conditions are defined in terms of target sets $F \subseteq V$ and explorability asks to visit every vertex in V.

3 Static Graphs

We first discuss the bounds of explorability games for both one and two player variants, on static graphs, i.e., every edge is available at any time.

▶ **Lemma 3.** *For every arena $\mathcal{A} = (V_1, V_2, E)$ and vertices $s, t \in V$ one can construct (in logarithmic space) an arena $\mathcal{B} = (V_1', V_2', E')$ such that $V' = V_1' \cup V_2'$, $V \subseteq V'$, $V_2 = V_2'$ and Player 1 wins the reachability game on $\mathcal{A}$ from s to t if, and only if, she wins the explorability game on $\mathcal{B}$.*

Proof. Without loss of generality, assume that $t \in V_1$ and that each vertex has at least one outgoing edge. We construct $\mathcal{B}$ as follows. Firstly, for every edge between vertices v, u, introduce a new vertex $[v, u] \in V_1'$ that can either move to u or reset, i.e. move back to s. Secondly, from target t there exists an edge to every other vertex.

We claim that Player 1 wins the reachability game on $\mathcal{A}$ if and only if she wins the explorability game on $\mathcal{B}$. Indeed, if Player 1 does not win on $\mathcal{A}$, then she cannot visit t on either arena and therefore loses in both. Conversely, if Player 1 does win on $\mathcal{A}$, then she can explore $\mathcal{B}$ by repeatedly moving from s to t; visit a previously unseen vertex, then reset to s. Note that a reduction from a one-player reachability game where Player 1 owns all the vertices will construct another one-player explorability game where this still holds. ◀

The following characterization of explorability games is adapted from [17, Theorem 4], where $s \preceq t$ denotes that Player 1 wins the reachability game from s with target t.

▶ **Lemma 4.** *Consider an arena with vertex set V and let $s \in V$ be an initial vertex. Player 1 wins the explorability game from s if, and only if, both 1) for all $u, v \in V$ either $u \preceq v$ or $v \preceq u$; and 2) for all $u \in V$, $s \preceq u$.*

Proof. Suppose both conditions 1 and 2 are true. Point 1 implies that there exists a linearization of all n vertices such that $v_1 \preceq v_2 \preceq \ldots \preceq v_n$. By Point 2 we have that $s \preceq v_1$ and therefore there is such a linearization of $\preceq$ with $v_1 = s$. Consequently, for all $1 \leq i < n$ there exists a Player 1 strategy σ_i that wins the reachability game from v_i to v_{i+1}. Player 1 can follow the σ_i from v_i until v_{i+1} is reached, then switch to σ_{i+1} and so on, until all vertices have been visited.

Suppose we have an arena where the first condition is false: there are $u, v \in V$ with $u \not\preceq v$ and $v \not\preceq u$. Regardless of the starting vertex s, once a play visits u, Player 1 cannot ensure to visit v from then on (or vice versa). Finally, if the second condition is false then there must be one vertex u that Player 1 cannot ensure to visit from s. ◀

▶ **Theorem 5.** *Solving one-player (respectively two-player) explorability games on a static graph is complete for* NL *(respectively* P*).*

Proof. The hardness is a consequence of Lemma 3. Note that the explorability game constructed from a one-player reachability game is also one-player. As one and two-player reachability games are NL-hard and P-hard respectively, the lower bounds follow. The upper bounds follow from Lemma 4, observing that at most n^2 reachability queries are necessary to verify the two conditions in the lemma. These queries can be done in NL in the one-player case, and P in the two-player case. ◀

4 Explicitly represented Temporal Graphs

Before discussing explorability games, we first remark that solving reachability and generalized reachability games have the same complexity on explicitly represented temporal graphs and on static graphs. This is because for any temporal graph $\mathcal{A}$ one can construct its expansion $\mathcal{A}'$, which is only polynomially larger assuming explicit encodings, and modify the winning conditions appropriately to get a game on this static graph: every target vertex v in $\mathcal{A}$ gives rise to target vertices (v, θ) in $\mathcal{A}'$ for all times $0 \leq \theta \leq h(\mathcal{A})$.

This reduction allows transferring complexity upper bounds for solving games on static arenas [17].

▶ **Theorem 6.** *Assuming explicit encodings, solving one-player games on temporal graphs is* NL*-complete for reachability conditions and* NP*-complete for generalized reachability. Solving two-player games on temporal graphs is* P*-complete for reachability and* PSPACE*-complete for generalized reachability.*

Note that an explorability game on a temporal graph corresponds to a generalized reachability game on its expansion as outlined above. However, the target sets are no longer singleton sets and therefore the improved (NL. resp. P) upper bounds provided in [17] for (one resp. two-player) generalized reachability with singleton targets do not apply to explorability on temporal graphs. For example, the explorability game on $\mathcal{A}_1$ (Figure 1) corresponds to the generalised reachability game on its expansion (Figure 3) with targets $\{s\} \times \underline{4}$, $\{t\} \times \underline{4}$, $\{u\} \times \underline{4}$, and $\{v\} \times \underline{4}$, where $\underline{4} = \{0, 1, 2, 3, 4\}$ are all possible times up to the horizon.

In contrast to games on static arenas, where explorability is not harder than reachability, we will see that on temporal graphs explorability is as hard as generalized reachability.

▶ **Theorem 7.** *Solving one-player explorability games on an explicit temporal graph is* NP*-complete.*

Proof. The NP-hardness is due to Michail and Spirakis [32, Proposition 2] by reduction from the Hamiltonian Path problem. Indeed, a directed graph with n vertices has a Hamiltonian path if and only if it can be explored in exactly n steps. A matching upper bound can be achieved by stepwise guessing an exploring path of length at most $h(G)$, which is polynomial given that the input graph is explicitly represented. ◀

We now consider the impact of an antagonistic player for explorability on temporal graphs. A key element of our lower bound construction is the interaction of antagonistic choice and the passing of time. Our reduction, from QBF, relies on a time-bounded final “flooding phase” that allows exploration only if sufficient time is left, and which can only be guaranteed by winning the preceding QBF game.

▶ **Theorem 8.** *Solving two-player explorability games on explicit temporal graphs is* PSPACE*-complete.*

Proof. The upper bound for solving explorability games on a temporal arena $\mathcal{A}$ follows from solving the generalized reachability game on the expansion of $\mathcal{A}$, where we have a target set $F_i = \{(v_i, \theta) \mid \forall\, \theta \in \{0, 1, \ldots, h(\mathcal{A})\}\}$ for every vertex v_i of $\mathcal{A}$. Generalized reachability games can be solved in PSPACE [17, Theorem 1].

We provide a matching lower bound using a reduction from QSAT. Given a QBF formula $\Phi = \exists x_1 \forall x_2 \ldots \exists x_n.\ C_1 \wedge C_2 \wedge \ldots \wedge C_k$, we construct a two-player explorability game G_Φ on a temporal graph so that Player 1 wins iff Φ is true. Let φ refer to the matrix $C_1 \wedge C_2 \wedge \ldots \wedge C_k$ of the given formula Φ, that is, each C_i is a disjunction of at most 3 literals. The game G_Φ has vertices $V = V_\forall \uplus V_\exists \uplus \{q_\varphi\} \uplus V_{\text{literals}} \uplus V_{\text{clause}}$, where

- the vertices $V_\exists = \{q_1, q_3, \ldots, q_n\}$, $V_{literals} = \{q_{x_1}, q_{\neg x_1}, q_{x_2}, q_{\neg x_2}, \ldots, q_{x_n}, q_{\neg x_n}\}$ and $V_{clauses} = \{q_{C_1}, q_{C_2}, \ldots, q_{C_k}\}$ are controlled by Player 1.
- the vertices $V_\forall\{q_2, q_4, \ldots, q_{n-1}\}$ and $\{q_\varphi\}$ are controlled by Player 2.

The game is played in 4 phases, each of a fixed length, as follows.

1. In the *initial phase*, from time 0 to $k-1$, the game starts from q_{C_1} and visits all clause vertices of $V_{clauses}$ in order and then ends up in the vertex q_1 corresponding to the first existential quantifier. Formally, the edges available during this phase are, for all $0 \leq i < k$, $E(q_{C_i}, q_{C_{i+1}}) = E(q_{C_k}, q_1) = [0, k)$. Edges of the initial phase are drawn in red in the example in Figure 4.
2. The *selection phase*, from time k to $k + 2n - 1$, the game visits the vertices corresponding to the quantifiers in order of their appearance in the formula. Depending on whether the quantifier is $\exists$ (or $\forall$), Player 1 (respectively Player 2) chooses to visit a vertex corresponding to either a positive or negative literal for each variable. The edges available during this phase are for all $0 \leq i < n$, $E(q_i, q_{x_i}) = E(q_i, q_{\neg x_i}) = E(q_{x_i}, q_{i+1}) = E(q_{\neg x_i}, q_{i+1}) = E(q_{x_n}, q_\varphi) = E(q_{\neg x_n}, q_\varphi) = [k, k + 2n)$. Corresponding edges are in blue in Figure 4 and corresponds to choosing a valuation for the variables in the formula Φ.
3. The *evaluation phase*, at time $k + 2n$ and $k + 2n + 1$, starts from vertex q_φ, where Player 2 selects a clause C_i by moving to vertex q_{C_i}. Formally, we have for all $0 < i \leq k$, $E(q_\varphi, q_{C_i}) = \{k + 2n\}$. From q_{C_i}, Player 1 has the choice to visit a literal vertex q_ℓ at time $k + 2n + 1$ if $\neg\ell$ is contained in the clause C_i, where ℓ is a literal. After this phase, all vertices in $V_\forall \uplus V_\exists \uplus \{q_\varphi\} \uplus V_{\text{clause}}$ have been visited as well as half of $V_{literal}$. In Figure 4, edges for the clause selection and literal selection are in green.
4. The game ends after a *flooding phase* that lasts for exactly $n - 1$ steps starting from time $k + 2n + 2$. That is, all edges become unavailable from time $k + 3n + 1$ onwards. During this phase, an edge is available from a literal vertex q_{ℓ_i} to both literal vertex $q_{x_{i+1}}$ and $q_{\neg x_{i+1}}$. Formally, the edges $(q_{\ell_i}, q_{\ell_{i+1}})$, for all $0 < i < n$ and (q_{ℓ_n}, q_{ℓ_1}), where $\ell_j \in \{x_j, \neg x_j\}$, are available at times $[k + 2n + 2, k + 3n + 1]$. Thus, in this phase Player 1 can visit exactly $n - 1$ of the possibly unexplored vertices. The flooding phase is shown in yellow in Figure 4.

Suppose the QBF formula Φ is true. Then the $\exists$-player has a strategy to assign values to the variables x_i (for odd i) to ensure that $\varphi(\vec{x})$ is true. By following the same choices in the selection phase (moving $q_i \longrightarrow q_{x_i}$ if x_i is set to *true* and $q_i \longrightarrow q_{\neg x_i}$ otherwise), Player 1 guarantees that when q_φ is reached, the jointly chosen variable assignment $\vec{x}$ satisfies φ. At this point the play has already visited exactly one vertex among q_{x_j} and $q_{\neg x_j}$ for all $j \leq n$. Moreover, no matter which vertex q_{C_i} is chosen by Player 2 in the next step, the corresponding clause C_i is also satisfied by assignment $\vec{x}$. By construction, this means there exists a literal $\ell_j \in C_i$ such that vertex q_{ℓ_j} has been visited before. The Player 1 can then move $C_i \longrightarrow q_{\neg \ell_j}$ to end the evaluation phase. Finally, in the flooding phase, Player 1 can freely move to visit the remaining $n - 1$ literal vertices not visited thus far. This guarantees that indeed, all vertices have been visited and Player 1 wins the explorability game.

If the QBF instance Φ is not satisfiable, then the $\forall$-player has a strategy to assign values to the x_{2i} to ensure that $\varphi(\vec{x})$ is false. By following the same choices during the selection phase (moving $q_i \longrightarrow q_{x_i}$ if x_i is set to *true* and $q_i \longrightarrow q_{\neg x_i}$ otherwise), Player 2 guarantees

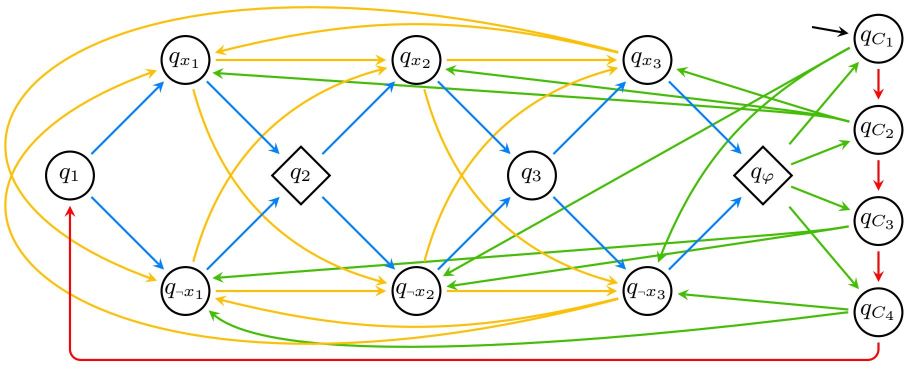


Figure 4 The exploration game for QBF formula $\exists x_1{:}\forall x_2{:}\exists x_3{:}(C_1 = x_2 \vee x_3) \wedge (C_2 = \neg x_1 \vee \neg x_2 \vee \neg x_3) \wedge (C_3 = x_1 \vee x_2) \wedge (C_4 = x_1 \vee x_3)$, where colours encode edge availibility: red=$[0, k)$, blue=$[k, k + 2n)$, green=$[k + 2n, k + 2n + 1]$, yellow=$(k + 2n + 1, k + 3n + 1]$.

that when q_φ is reached, the jointly chosen variable assignment does not satisfy Φ, meaning that there is some clause C_i that is not satisfied. In the evaluation phase, the strategy τ then picks q_{C_i} as the successor from q_φ. Consequently, all successors of q_{C_i} must be vertices that have already been seen in the play. This means that exactly n of the literal vertices are not yet visited. However, the game ends after the following $n - 1$ of the flooding phase. Therefore, one of the literal vertices must be left unexplored. ◀

5 Symbolically represented Temporal Graphs

We now consider games on temporal graphs that are given in a succinct representation. More precisely, we assume that the arena is given as the underlying (static) graph and for every edge, the set of times at which it is available is represented by a formula in the existential fragment of Presburger Arithmetic (∃PA). Recall that evaluating $\varphi(x)$ for a given ∃PA formula φ and time $x \in \mathbb{N}$ is in NP. We call this representation of temporal graphs *symbolic*. Symbolic representations that are equally as powerful as ∃PA, such that they can represent semi-linear sets, include the likes of solving a union of linear equations and commutative context-free grammars [10]. Finding a satisfying valuation for both of these encodings of semi-linear sets has also been shown to be in NP, much like ∃PA. We use ∃PA as the formulae produced are concise and easy to understand after only a brief viewing. There are other concise encodings (such as the inclusion of the universal fragment of PA) that could be used for a symbolic temporal graph, however the problem of finding membership or satisfiability of a given value may be too complex for the encoding to be efficient. For example, satisfiability with respects to the universal fragment of PA is exponential and not in NP.

▶ **Theorem 9.** *Solving one-player symbolic temporal reachability games is* PSPACE*-hard.*

Proof. We reduce QSAT to one-player temporal reachability with symbolic time encoding. Consider a quantified Boolean formula $\Phi = \exists x_1 \forall x_2 \ldots \exists x_n : \varphi$ where φ is a propositional formula with variables in $X = \{x_1, \ldots, x_n\}$. Without loss of generality assume that the quantifiers alternate strictly, starting and ending with existential quantifiers (introduce at most $n + 1$ useless variables otherwise). We show how to construct the symbolic one-player game G_Φ such that Φ is satisfiable if and only if some target vertex of G_Φ can be reached. We use elapse of time to encode valuations of variables of Φ. At a time $\theta \geq 0$, since φ

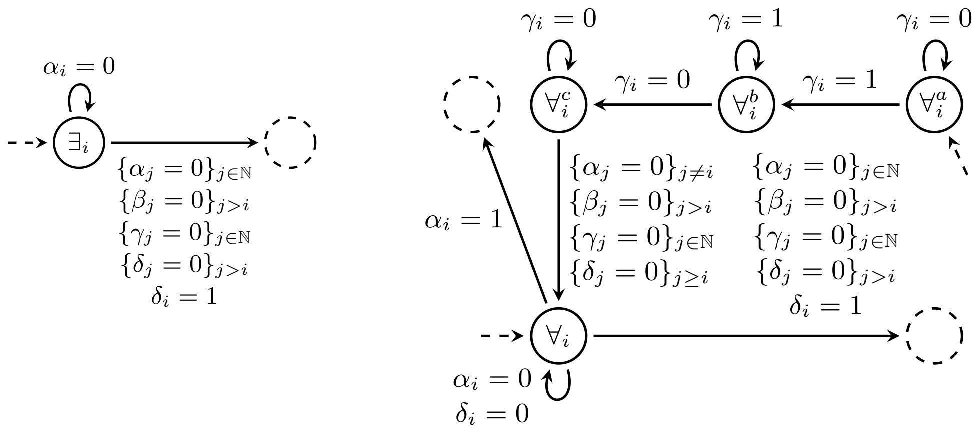


Figure 5 Gadgets to construct G_Φ.

is quantifier-free, determining whether the valuation $\nu_\theta\colon X \to \{0,1\}$ makes Φ true can be checked with a single transition that is available exactly at time θ. The difficulty arises from encoding of the "adversarial" behaviour of universal quantifiers. To do so, we leverage both the graph structure and the Presburger arithmetic.

In our encoding, time is sectored into n segments of 4 bits. The horizon of G_Φ is thus $h(G_\Phi) = 2^{4n}$. For every $i \in \{1,\dots,n\}$ and time $\theta \in \{0,...,2^{4n}-1\}$, we call α_i, β_i, γ_i, and δ_i the four bits of the ith most significant sector, i.e., the bits at indices $(4n-1)-(4i-4)$, $(4n-1)-(4i-3)$, $(4n-1)-(4i-2)$ and $(4n-1)-(4i-1)$ in the binary expansion of θ. In particular, $\alpha_1 = 4n-1$ and $\delta_n = 0$ are respectively the most and the least significant bits. Intuitively, in the sequel we use β_i to represent the chosen value of variable $x_i \in X$, we control overflows of β_i with assumption on α_i and γ_i, and δ_i is an extra bit acting as buffer, to allow spending time without influencing the others. In the sequel, we define constrains on such bits. To do so, for all $k \in \{0,\dots,4n-1\}$ and all $v \in \{0,1\}$, we require the kth least significant bit of a given time $\theta \in \{0,\dots,2^{4n}-1\}$ to be v. This can be expressed thanks to following Presburger formula, which states that the kth least significant bit in the binary expansion of θ equals v.

$$\Psi_{k,v}(\theta) = \exists y_0 : \dots \exists y_{4n} : \exists v_0 : \dots \exists v_{4n-1} : \wedge \begin{cases} \bigwedge_{j=0}^{4n-1}(v_j = 0 \vee v_j = 1) & \text{i.e., } v_j \in \{0,1\} \\ \bigwedge_{j=0}^{4n-1}(y_j = 2y_{j+1} + v_j) & \text{i.e., } y_{j+1} \text{ is } \left\lfloor \frac{\theta}{j+1} \right\rfloor \\ (y_0 = \theta) \wedge (v_k = v) & \text{and } v_k \text{ is } \theta\text{'s } k\text{th bit} \end{cases}$$

In the following, we write $\xi_i = v$, where $\xi \in \{\alpha,\beta,\gamma,\delta\}$ and $v \in \{0,1\}$ as a shorthand for the formula for checking the corresponding bits.

In G_Φ, each quantifier will have a corresponding vertex. If $x_i \in X$ is existentially quantified, the vertex q_i evaluates x_i thanks to a self-loop which also set $\delta_i = 1$. In other words, Player 1 can use the self-looping transition of q_i for spending time (not more than 2^{α_i} steps since α_i must be 0) which in particular leaves him possibility to set the bit β_i as an encoding the value of x_i. Additionally, the non-looping transition of q_i is available only when the bit δ_i is 1. See the left gadget of Figure 5. Observe that, if q_i is entered at time θ where $\delta_i = 0$, there are exactly two times θ_0, θ_1 when q_i can be exited, and θ_0 and θ_1 only differ on the bit β_i. Formally, $\theta_0 = \theta + 2^{(4n-1)-(4i-1)}$, $\theta_1 = \theta_0 + 2^{(4n-1)-(4i-3)}$. This is a direct consequence of the availability of both outgoing transitions of q_i leaving only the bit β_i as degree of freedom to Player 1. We shall prove that vertices encoding existential quantifiers

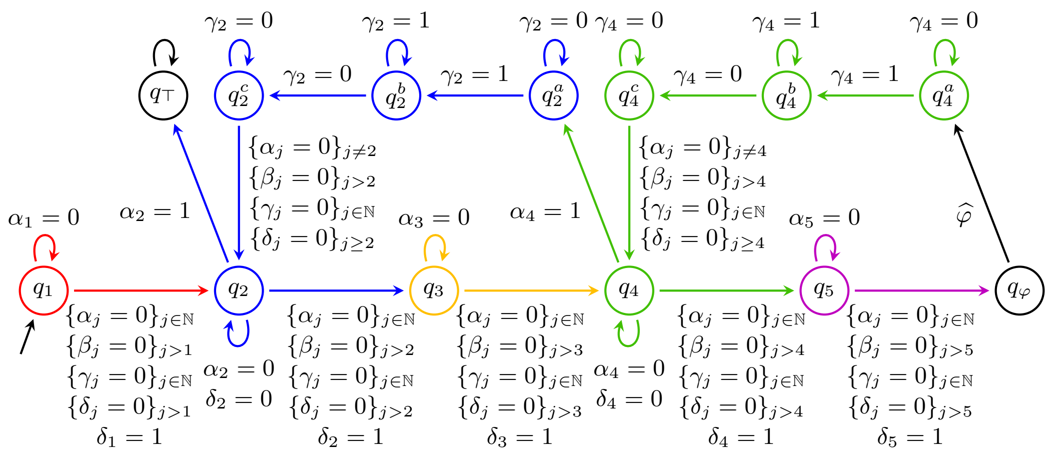


Figure 6 Encoding of a formula with 5 quantifiers to a symbolic one-player game.

are the only source of branching in G_Φ. If $x_i \in X$ is universally quantified, the self-loop of the vertex q_i only serves to set $\delta_i = 1$, the evaluation of x_i being driven by vertices encoding existential quantifiers and auxiliary vertices q_i^a, q_i^b and q_i^c. See the right gadget of Figure 5. The vertex q_i admits two behaviours. If q_i is entered at a time when $\alpha_i = 0$, then q_i exits toward the vertex encoding the next existential quantifier. Otherwise, $\alpha_i = 1$ and then q_i backtracks toward the auxiliary vertices of the previous universal quantifier. The elapsing of time while backtracking will set γ_{i-2} to 1, and then sets it back to 0. If $\beta_{i-2} = 0$, the side effect of backtracking is to switch β_{i-2} from 0 to 1. Otherwise, $\beta_{i-2} = 1$, and the side effect of backtracking is to switch α_{i-2} from 0 to 1, which triggers a further backtrack in q_{i-2}. To end the construction of G_Φ, we add two more vertices q_φ (to check the valuation of Φ encoded by elapse of time) and $q_\top$ (to end the cascade of backtrack). A complete picture is given when Φ has 5 quantifiers in Figure 6. The quantifier-free Presburger formula $\widehat{\varphi}$ is defined as φ where every positive occurrence x_i in φ is replaced by $\beta_i = 1$ in $\widehat{\varphi}$ and every negative occurrence $\neg x_i$ is replaced by $\beta_i = 0$. The size of G_Φ, in particular the encoding of edges availability is discussed at the end of the proof.

Next, we prove that Φ is satisfiable if and only if G_Φ admits a path from q_1 to $q_\top$ composed of exactly 2^{4n} edges, thanks to the unrestricted self-loop on $q_\top$. The argument holds by induction on the odd number $n \geq 1$ of quantifiers. In the base case, $n = 1$ and G_Φ has three vertices q_1, q_φ and $q_\top$. We show that when q_1 is entered at time 0, it can be exited only at time $\theta_0 = 1$ and $\theta_1 = 5$. By construction, q_1 can only be exited toward q_φ that has a single outgoing edge toward $q_\top$. Hence, the edge from q_φ to $q_\top$ can only be traversed at time $\theta_0 + 1 = 2$ or $\theta_1 + 1 = 6$ where $\beta_1 = 0$ and $\beta_1 = 1$ respectively. By definition of $\widehat{\varphi}$, the vertex $q_\top$ is reachable if and only if Φ is satisfiable.

Now, assume that Φ has an odd number $n \geq 3$ of quantifiers. Let $\Phi = \exists x_1 \forall x_2 \Phi'$ where $\Phi' = \exists x_3 \forall x_4 \ldots \exists x_n : \varphi$. For all $v_1, v_2 \in \{\texttt{false}, \texttt{true}\}$, we denote $\Phi'[x_1 \leftarrow v_1, x_2 \leftarrow v_2]$ the formula Φ' where x_1 takes value v_1 and x_2 takes value v_2. Observe that $\Phi'[x_1 \leftarrow v_1, x_2 \leftarrow v_2]$ is a quantified Boolean formula without free-variable. For all $v_1, v_2 \in \{\texttt{false}, \texttt{true}\}$, by induction hypothesis $G_{\Phi'[x_1 \leftarrow v_1, x_2 \leftarrow v_2]}$ admits a path from q_1' to $q_\top'$ composed of exactly $2^{4(n-2)}$ edges if and only if $\Phi'[x_1 \leftarrow v_1, x_2 \leftarrow v_2]$ is satisfiable. Intuitively, we construct such a path of G_Φ from q_1 to $q_\top$ based on a path of $G_{\Phi'}$ from q_1' to $q_\top'$. This is effective since $G_{\Phi'[x_1 \leftarrow v_1, x_2 \leftarrow v_2]}$ is a subgraph of $G_{\Phi[x_1 \leftarrow v_1, x_2 \leftarrow v_2]}$ where q_3 and q_2^a in $G_{\Phi[x_1 \leftarrow v_1, x_2 \leftarrow v_2]}$ match respectively with q_1' and $q_\top'$ in $G_{\Phi'[x_1 \leftarrow v_1, x_2 \leftarrow v_2]}$. In fact, $G_{\Phi'[x_1 \leftarrow v_1, x_2 \leftarrow v_2]}$ has an horizon of $2^{4(n-2)}$ because Φ' has $n-2$ quantifiers, and q_2^a has a self-loop available at all time in this horizon, that is, $G_{\Phi'[x_1 \leftarrow v_1, x_2 \leftarrow v_2]}$ cannot modify the bits of bit sectors 1 and 2. Also, q_3 is entered only when the $4(n-2)$ least significant bits of the time are zero due to the availability of the edge from q_2 to q_3, and q_2^a is leaved only if the $4(n-2)$ bits overflow due to as a direct consequence of the availability of the edge from q_2^a to q_2^b.

We show that when q_1 is entered at time 0, it can be exited only at time $\theta_0 = 2^{4n-4}$, $\theta_1 = 2^{4n-4} + 2^{4n-2}$. Then q_2 is visited and since in both cases $\alpha_2 = 0$, by construction, the vertex q_3 is visited at time $\theta_0 + 2^{4n-8}$ or $\theta_1 + 2^{4n-8}$ (where δ_2 switched to 1). Let $\theta'_0 = \theta_0 + 2^{4n-8}$ where $\beta_1 = 0$ and $\theta'_1 = \theta_1 + 2^{4n-8}$ where $\beta_1 = 1$. In both cases, $\beta_2 = 0$. For all $k \in \{0,1\}$, by induction hypothesis, there is a path from q_3 at time θ'_k to q_2^a at time $\theta'_k + 2^{4(n-2)} - 1$ if and only if $\Phi'[x_1 \leftarrow k, x_2 \leftarrow 0]$ is satisfiable. If $\Phi'[x_1 \leftarrow 0, x_2 \leftarrow 0]$ and $\Phi'[x_1 \leftarrow 1, x_2 \leftarrow 0]$ are not satisfiable, then G_Φ cannot reach $q_\top$ from q_1 and Φ is not satisfiable. Otherwise, the two paths in G_Φ that reach q_2^a at $\theta'_0 + 2^{4(n-2)} - 1$ and $\theta'_1 + 2^{4(n-2)} - 1$ can be extended to reach q_2 at time $\theta_0 + 2^{4n-6}$ and $\theta_1 + 2^{4n-6}$ respectively (where β_2 switched to 1). Since $\alpha_2 = 0$ in both cases, the vertex q_3 is visited at time $\theta_0 + 2^{4n-6} + 2^{4n-8}$ and $\theta_1 + 2^{4n-6} + 2^{4n-8}$ respectively (where switched to $\delta_2 = 1$). Let $\theta''_0 = \theta_0 + 2^{4n-6} + 2^{4n-8}$ where $\beta_1 = 0$ and $\theta''_1 = \theta_1 + 2^{4n-6} + 2^{4n-8}$ where $\beta_1 = 1$. For all $v_1, v_2 \in \{\texttt{false}, \texttt{true}\}$, by induction hypothesis, there is a path from q_3 at time θ''_k to q_2^a at time $\theta_k + 2^{4(n-2)} - 1$ if and only if $\Phi'[x_1 \leftarrow k, x_2 \leftarrow 1]$ is satisfiable. If $\Phi'[x_1 \leftarrow 0, x_2 \leftarrow 1]$ and $\Phi'[x_1 \leftarrow 1, x_2 \leftarrow 1]$ are not satisfiable, then G_Φ cannot reach $q_\top$ and Φ is not satisfiable. Otherwise, the two paths in G_Φ that reach q_2^a at $\theta''_0 + 2^{4(n-2)} - 1$ and $\theta''_1 + 2^{4(n-2)} - 1$ can be extended to reach q_2 at time $\theta_0 + 2^{4n-5}$ and $\theta_1 + 2^{4n-5}$ respectively (where α_2 switched to 1). Since $\alpha_2 = 1$, the vertex $q_\top$ is visited at time $\theta_0 + 2^{4n-5} + 1$ and $\theta_1 + 2^{4n-5} + 1$ respectively. Hence, G_Φ admit a path from q_1 to $q_\top$ if and only if there is $k \in \{0,1\}$ such that $\Phi'[x_1 \leftarrow k, x_2 \leftarrow 0]$ and $\Phi'[x_1 \leftarrow k, x_2 \leftarrow 1]$ are satisfiable, i.e., if and only if Φ is satisfiable.

It is worth emphasizing that, when G_Φ admits a path from q_1 to $q_\top$, it can be constructed with a memoryless strategy. In the above reduction, for each vertex q_i that encodes an existential quantifier, the inductively constructed strategy aims in q_i at evaluating β_i such that the next time q_φ is visited its outgoing edge will be available. Hence, in q_i, the strategy is solely based on the value $\beta_{i-1}, \ldots, \beta_1$. Since the vertices encoding existential quantifiers are the only source of branching in G_Φ, the strategy is memoryless.

Finally, we show that the size of G_Φ is polynomial in the size of Φ. Here after, the size of the formula corresponds to the number of symbols to write it. The size of a symbolic temporal graphs G, is defined by $|G| = |V| + \sum_{(u,v)\in V^2} |E(u,v)|$. The graph G_Φ has at most $4n+2$ vertices, and all availability are expressible by a Presburger formula of polynomial size in $|\Phi|$. Since the horizon of G_Φ is 2^{4n}, each edge availability can be expressed as a conjunction of at most $4n$ formulas of the form $\Psi_{k,v}$. ◀

▶ **Corollary 10.** *Solving reachability games on one-player symbolic temporal graphs with at most K temporal edges is Σ_m^P-hard, i.e, hard for the m-th level of the polynomial hierarchy, where $K \geq 2\lceil \frac{m}{2} \rceil + 9\lfloor \frac{m}{2} \rfloor + 1$.*

Proof. In the proof of Theorem 9, the number of quantifiers in Φ determines the number of temporal edges used in the constructed symbolic temporal graph. The gadget to encode existential quantifiers uses two temporal edges and the one for universal quantifiers uses nine temporal edges, and there is one edge for the quantifier-free formula φ (see Figure 5). Note that if a quantifier block contains more than m variables, we can blow up the number of bits polynomially by having bits β_i^j for $1 \leq j \leq m$, while keeping the number of temporal edges the same, thus obtaining a reduction from QSAT with a fixed number of quantifiers. ◀

A similar Π_m^P-hardness also holds for a different choice of m. Note that this is in contrast with the PSPACE-hardness proof for two-player reachability games on symbolic temporal graphs, where the problem is PSPACE-hard even with just one temporal edge [17, Theorem 1].

A restriction that makes the problem easier is allowing *waiting*. It is typical in study of temporal graphs to allow waiting for an arbitrary amount of time in temporal paths, i.e., instead of having to take an edge at every timestep, edges are traversed at increasing (but not necessarily consecutive) timesteps. This corresponds to having a loop on all vertices that can be traversed at any time. In case of two-player games, this allows Player 2 to stall infinitely and therefore is not interesting. The following remark states that the explorability on a one-player temporal arena with symbolic encoding becomes easy if waiting is allowed. We show that this holds even for generalized reachability objectives.

▶ **Theorem 11.** *Assuming symbolic encodings, solving one-player games on temporal graphs with waiting is* NP*-complete for reachability, explorability and generalized reachability.*

Proof. The NP lower bound follows from checking satisfiability of an existential Presburger formula, which is known to be NP-complete [37]. Given a $\exists$PA formula ϕ with no free variables, consider the symbolic temporal graph $V = \{s, t\}$, such that $E(s,t) \overset{\text{def}}{=} \phi(x)$ is the only available edge, where x is a variable that does not occur in ϕ. The vertex t is reachable from s if, and only if, ϕ is true. For the upper bound, note that the length of a shortest witnessing path, measured by the number of edges traversed, is at most quadratic in the number of vertices of the graph. The NP upper bound can be shown by guessing the vertices visited along a play, together with the time they are visited first. The intermediate path between two such vertices can only visit linearly many vertices as no vertex is repeated in such a path. If a revisit is necessary, there is another path which waits instead. If a path does exist, then the time at which it is visited will at most be exponentially large and consequentially can be guessed in polynomial time. Therefore, by guessing the path and the times at which corresponding edges are taken, we obtain an NP algorithm. ◀

▶ Remark 12. In [15], Fearnley and Jurdziński introduce counter-stack automata to prove that reachability in two-clock timed automata is PSPACE-complete. The model of counter-stack automata bears similarity with the technique used in proof of Theorem 9. Indeed, both rely on a linear ordering of counters (being bits in our model) and that incrementation of a counter can only be performed when the value of all smaller counters is known. When elapse of time sets a bit to 1, the smaller bits are reset to 0. In [15] counters can be reset, which correspond to reset of clocks. Our proof cannot be derived from [15] because time cannot be reset in our setting. Vice versa, our proof uses existential Presburger formula with arbitrarily many variables which is not known to be expressible with a two-clock automata.

▶ **Theorem 13.** *Solving two-player symbolic temporal generalized reachability games is in* EXP*. It is in* PSPACE *for one-player games.*

Proof. Consider a generalized reachability game G on arena $\mathcal{A} = (V_1, V_2, E)$ and with targets $\mathcal{F} = F_1, \dots F_k$. Note that under symbolic encodings, temporal graphs are ultimately periodic, meaning that there are $b, p \in \mathbb{N}$ so that for every $i > b$, $i \in E(u,v)$ if, and only if, $i + p \in E(u,v)$. This is because the set of times at which any given edge is available is defined using a Presburger formula and thus semilinear [20]. Moreover, the least common multiple of the periods of the individual edges is therefore a period of (all edges in) the temporal graph and we can bound b and p to be at most exponential in the size of the input.

We can therefore construct an at most exponentially larger reachability game G' in which control states record the set of states (of G) that have already been visited, and the time up to and within the period. That is, define the arena $\mathcal{A}' = (V_1', V_2', E')$ where $V' = V \times 2^V \times \{0, 1, \ldots, b+p-1\}$ and with edge relation E' so that $(u, S, \theta) \longrightarrow (v, S \cup \{v\}, \theta')$ if $\theta \in E(u, v)$ and either $\theta' = \theta + 1$ or both $\theta = b + p - 1$ and $\theta' = b$. Player 1 owns a state (u, S, θ) in G' iff she owns u in G. Finally, define the reachability target set in the constructed game G' as

$$F' = \{(v, S, \theta) \in V' : \forall F_j \in \mathcal{F}, \exists v \in F_j, \text{ such that } v \in S\}.$$

Notice that any play from $(v, \emptyset, 0)$ in G' uniquely corresponds to a play from state s at time 0 in the original game G. The second components within control states in G' are non-decreasing and record the set of states visited by the corresponding play in G. By definition of our reachability target F', we see that a play is winning for Player 1 in the constructed reachability game G' if, and only if, the corresponding play is winning for Player 1 in the generalized reachability game G.

Note that the size of G' is $|V| \cdot |2^V| \cdot (b + p)$, where $b, p \leq 2^{\mathcal{O}(|G|)}$. The claim thus follows from the (known) facts that two-, and one-reachability games can be solved in polynomial time and logarithmic space, respectively. ◀

▶ **Corollary 14.** *Assuming symbolic encodings, solving one-player games on temporal graphs is* PSPACE*-complete for reachability, explorability and generalized reachability. Solving two-player games on temporal graphs is* PSPACE*-complete for reachability and* PSPACE*-hard and in* EXP *for explorability and generalized reachability.*

6 Conclusion

We study the complexities of solving explorability games on temporal graphs, splitting into several cases based on the way temporal edges are defined and whether one or two-players are active.

First, we transfer results for games on static graphs and show that there, solving explorability games is NL-complete for the one-player and P-complete for the two-player case.

On explicitly represented temporal graphs in which waiting is permitted, it was known that checking the existence of an exploring path, i.e. the solving one-player explorability games is NP-complete [30]. We show here that even for the more general case where waiting is not always allowed the problem remains in NP. We further show that solving two-player games on explicitly represented temporal graphs is PSPACE-complete for reachability, explorability, and generalized reachability conditions. The existing lower bound for reachability [6] crucially relies on binary encodings and the presence of an antagonistic player.

The main technical contribution of this work is a strong lower bound for symbolically represented temporal graphs: We show that already single-player reachability (and thus explorability) games are PSPACE-hard. Our construction uses specially crafted Presburger constraints to ensure that all possible choices of an "opponent" in a QBF game are verified. Towards upper bounds, we show that even two-player generalized reachability games are solvable in deterministic exponential time, which is better than the exponential *space* procedure one gets for free (by encoding time into the state space) but not quite matching our lower bound. We conjecture that indeed, solving two-player games for each of these three objectives remains in PSPACE, mainly because raising the lower bound remained elusive. To show EXP-hardness one needs to encode non-trivial and recoverable information into timestamps. Our attempts to reduce from CTL Satisfiability or Countdown Games so far required to either construct Presburger constraints to compare bits that are exponentially far apart in the binary expansion of a given free variable (time), or to construct gadgets that "copy" bitstrings from lower to higher significant bit positions. It is worth noting that any such construction must only be correct for explorability but not for the simpler reachability conditions (which are PSPACE-complete [6]).

On the other hand, showing PSPACE-membership would require either find a polynomially bounded untimed game gadgets to evaluate any given ∃PA formula or to encode the antagonistic behaviours arithmetically.

References

1 Duncan Adamson, Vladimir V. Gusev, Dmitriy Malyshev, and Viktor Zamaraev. Faster Exploration of Some Temporal Graphs. In *1st Symposium on Algorithmic Foundations of Dynamic Networks (SAND 2022)*, volume 221 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 5:1–5:10. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. `doi:10.4230/LIPIcs.SAND.2022.5`.

2 Eleni C. Akrida, George B. Mertzios, Paul G. Spirakis, and Christoforos Raptopoulos. The temporal explorer who returns to the base. *Journal of Computer and System Sciences*, 120:179–193, 2021. `doi:10.1016/j.jcss.2021.04.001`.

3 Eleni C. Akrida, George B. Mertzios, Paul G. Spirakis, and Viktor Zamaraev. Temporal vertex cover with a sliding time window. *Journal of Computer and System Sciences*, 107:108–123, 2020. `doi:10.1016/j.jcss.2019.08.002`.

4 Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994. `doi:10.1016/0304-3975(94)90010-8`.

5 Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 1st edition, 2009.

6 Pete Austin, Sougata Bose, and Patrick Totzke. Parity games on temporal graphs. In *International Conference on Foundations of Software Science and Computational Structures*, pages 79–98. Springer Nature Switzerland, 2024. `doi:10.1007/978-3-031-57228-9_5`.

7 Benjamin Merlin Bumpus and Kitty Meeks. Edge exploration of temporal graphs. In *Combinatorial Algorithms*, pages 107–121. Springer International Publishing, 2021. `doi:10.1007/978-3-030-79987-8_8`.

8 Krishnendu Chatterjee, Thomas A. Henzinger, and Vinayak S. Prabhu. Timed Parity Games: Complexity and Robustness. *Logical Methods in Computer Science*, Volume 7, Issue 4, December 2011. `doi:10.2168/LMCS-7(4:8)2011`.

9 Hongjiang Chen, Pengfei Jiao, Huijun Tang, and Huaming Wu. Temporal graph representation learning with adaptive augmentation contrastive. In *Machine Learning and Knowledge Discovery in Databases: Research Track*, pages 683–699. Springer Nature Switzerland, 2023. `doi:10.1007/978-3-031-43415-0_40`.

10 Dmitry Chistikov, Christoph Haase, and Simon Halfon. Context-free commutative grammars with integer counters and resets. *Theoretical Computer Science*, 735:147–161, 2018. Reachability Problems 2014: Special Issue. `doi:10.1016/j.tcs.2016.06.017`.

11 Jean-Lou De Carufel, Paola Flocchini, Nicola Santoro, and Frédéric Simard. Cops & robber on periodic temporal graphs: Characterization and improved bounds. In *Structural Information and Communication Complexity*, pages 386–405. Springer Nature Switzerland, 2023. `doi:10.1007/978-3-031-32733-9_17`.

12 Argyrios Deligkas, Eduard Eiben, and George Skretas. Minimizing reachability times on temporal graphs via shifting labels. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pages 5333–5340. International Joint Conferences on Artificial Intelligence Organization, August 2023. `doi:10.24963/ijcai.2023/592`.

13 Stefan Dobrev, Rastislav Královič, and Euripides Markou. Online graph exploration with advice. In *Structural Information and Communication Complexity*, pages 267–278. Springer Berlin Heidelberg, 2012. `doi:10.1007/978-3-642-31104-8_23`.

14 Thomas Erlebach, Michael Hoffmann, and Frank Kammer. On temporal graph exploration. *Journal of Computer and System Sciences*, 119:1–18, 2021. `doi:10.1016/j.jcss.2021.01.005`.

15 John Fearnley and Marcin Jurdziński. Reachability in two-clock timed automata is pspace-complete. In *Automata, Languages, and Programming*, pages 212–223. Springer Berlin Heidelberg, 2013.

16 Nathanaël Fijalkow, Nathalie Bertrand, Patricia Bouyer-Decitre, Romain Brenguier, Arnaud Carayol, John Fearnley, Hugo Gimbert, Florian Horn, Rasmus Ibsen-Jensen, Nicolas Markey, Benjamin Monmege, Petr Novotný, Mickael Randour, Ocan Sankur, Sylvain Schmitz, Olivier Serre, and Mateusz Skomra. Games on graphs, 2023. `doi:10.48550/arXiv.2305.10546`.

17 Nathanaël Fijalkow and Florian Horn. The surprizing complexity of generalized reachability games, 2012. `arXiv:1010.2420`.

18 Paola Flocchini, Bernard Mans, and Nicola Santoro. Exploration of periodically varying graphs. In *Algorithms and Computation*, pages 534–543. Springer Berlin Heidelberg, 2009. `doi:10.1007/978-3-642-10631-6_55`.

19 Janosch Fuchs, Christoph Grüne, and Tom Janßen. The complexity of online graph games. In *SOFSEM 2024: Theory and Practice of Computer Science*, pages 269–282. Springer Nature Switzerland, 2024. `doi:10.1007/978-3-031-52113-3_19`.

20 Seymour Ginsburg and Edwin H. Spanier. Semigroups, Presburger formulas, and languages. *Pacific Journal of Mathematics*, 16(2):285–296, 1966.

21 Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata Logics, and Infinite Games: A Guide to Current Research.* Springer-Verlag, 2002.

22 Petter Holme and Jari Saramäki. *Temporal Network Theory.* Springer, January 2019. `doi:10.1007/978-3-030-23495-9`.

23 Marcin Jurdziński and Ashutosh Trivedi. Reachability-time games on timed automata. In *International Colloquium on Automata, Languages and Programming*, pages 838–849. Springer Berlin Heidelberg, 2007.

24 Fabian Kuhn, Nancy Lynch, and Rotem Oshman. Distributed computation in dynamic networks. In *Symposium on Theory of Computing*, STOC '10, pages 513–522. Association for Computing Machinery, 2010. `doi:10.1145/1806689.1806760`.

25 Oded Maler, Amir Pnueli, and Joseph Sifakis. On the synthesis of discrete controllers for timed systems. In *International Symposium on Theoretical Aspects of Computer Science*, pages 229–242. Springer Berlin Heidelberg, 1995.

26 Nicole Megow, Kurt Mehlhorn, and Pascal Schweitzer. Online graph exploration: New results on old and new algorithms. In *Automata, Languages and Programming*, pages 478–489. Springer Berlin Heidelberg, 2011. `doi:10.1007/978-3-642-22012-8_38`.

27 George B. Mertzios, Hendrik Molter, Rolf Niedermeier, Viktor Zamaraev, and Philipp Zschoche. Computing maximum matchings in temporal graphs. *Journal of Computer and System Sciences*, 137:1–19, 2023. `doi:10.1016/j.jcss.2023.04.005`.

28 George B. Mertzios, Hendrik Molter, and Viktor Zamaraev. Sliding window temporal graph coloring. *Journal of Computer and System Sciences*, 120:97–115, 2021. `doi:10.1016/j.jcss.2021.03.005`.

29 George B. Mertzios, Sotiris Nikoletseas, Christoforos Raptopoulos, and Paul G. Spirakis. Brief Announcement: On the Existence of δ-Temporal Cliques in Random Simple Temporal Graphs. In *3rd Symposium on Algorithmic Foundations of Dynamic Networks (SAND 2024)*, volume 292 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 27:1–27:5. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. `doi:10.4230/LIPIcs.SAND.2024.27`.

30 Othon Michail. *An Introduction to Temporal Graphs: An Algorithmic Perspective*, pages 308–343. Springer International Publishing, 2015. `doi:10.1007/978-3-319-24024-4_18`.

31 Othon Michail, Ioannis Chatzigiannakis, and Paul G Spirakis. Causality, influence, and computation in possibly disconnected synchronous dynamic networks. *Journal of Parallel and Distributed Computing*, 74(1):2016–2026, 2014. `doi:10.1016/J.JPDC.2013.07.007`.

32 Othon Michail and Paul G. Spirakis. Traveling salesman problems in temporal graphs. *Theoretical Computer Science*, 634:1–23, 2016. `doi:10.1016/j.tcs.2016.04.006`.

33 Andrzej Pelc. Explorable families of graphs. In *Structural Information and Communication Complexity*, pages 165–177. Springer International Publishing, 2018. `doi:10.1007/978-3-030-01325-7_17`.

34 A. Pnueli and R. Rosner. On the synthesis of a reactive module. In *Annual Symposium on Principles of Programming Languages*, POPL '89, pages 179–190. Association for Computing Machinery, 1989. `doi:10.1145/75277.75293`.

35 Amir Pnueli. The temporal logic of programs. In *Annual Symposium on Foundations of Computer Science*, SFCS '77, pages 46–57. IEEE Computer Society, 1977. `doi:10.1109/SFCS.1977.32`.

36 Ramamoorthi Ravi. Rapid rumor ramification: Approximating the minimum broadcast time. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 202–213, 1994.

37 Bruno Scarpellini. Complexity of subcases of presburger arithmetic. *Transactions of the American Mathematical Society*, 284(1):203–218, 1984. `doi:10.2307/1999283`.

38 Yuejiao Wang, Dajun Daniel Zeng, Qingpeng Zhang, Pengfei Zhao, Xiaoli Wang, Quanyi Wang, Yin Luo, and Zhidong Cao. Adaptively temporal graph convolution model for epidemic prediction of multiple age groups. *Fundamental Research*, 2(2):311–320, 2022. `doi:10.1016/j.fmre.2021.07.007`.