# Restless Exploration of Periodic Temporal Graphs

**Thomas Bellitto**
Sorbonne Université, CNRS, LIP6, F-75005 Paris, France

**Cyril Conchon–Kerjan**
DIENS, Ecole normale supérieure, F-75005 Paris, France

**Bruno Escoffier**
Sorbonne Université, CNRS, LIP6, F-75005 Paris, France
Institut Universitaire de France, Paris, France

## Abstract

A temporal graph is a sequence of graphs, indexed by discrete time steps, with a fixed vertex set but with an edge set that is able to change over time. In the temporal graph exploration problem, an agent wants to visit all the vertices of a given temporal graph. In the classical model, at each time step the agent can either stay where they are, or move along one edge. In this work we add a constraint called *restlessness* that forces the agent to move along one edge at each time step. We mainly focus on (infinite) periodical temporal graphs. We show that if the period is 2 one can decide in polynomial time whether exploring the whole graph is possible or not, while this problem turns out to be NP-hard for any period $p \geq 3$. We also show some time bounds on the explorations of such graphs when the exploration is possible.

**2012 ACM Subject Classification** Theory of computation → Graph algorithms analysis

**Keywords and phrases** Temporal graphs, Graph exploration, NP-completeness

**Digital Object Identifier** 10.4230/LIPIcs.SAND.2023.13

## 1 Introduction

A temporal graph is a sequence of graphs $G = (G_1, G_2, \dots, G_k, \dots)$, where $G_t = (V, E_t)$ is called the snapshot at time step $t$. Vertices remain but edges are susceptible to be removed or added at each time step. When the sequence is finite, the number of steps is called the *lifetime* of the temporal graph. The study of algorithmic aspects of temporal graphs was promoted lately due to the emergence of dynamic networks with change of links over time (e.g., social-, wireless mobile-, transportation networks). Among the several problems that have been studied, the *temporal exploration problem* (TEXP) has received a lot of attention in the last decade. In this problem, an agent aims at visiting all vertices of $V$, in minimal time if visiting all vertices is possible. In the classical model, called the strict variant, the agent can travel on at most one edge at each time step, while in the non-strict variant the agent can use as many edges as they want at each time step [7].

In the strict model, while determining whether it is possible or not to explore the graph (i.e., visit all vertices) is NP-hard [10], it is not hard to see that this is always possible when (1) the graph is connected at each time step and (2) the lifetime is at least $n^2$, where $n = |V|$. A substantial amount of works have been devoted to study the problem on some specific graph classes, both on the computational complexity and on possible improvements of this $O(n^2)$ bound on the lifetime of the graph (which is in fact tight - lifetime $\Omega(n^2)$ might be necessary - for general graphs), see for instance [3, 5, 6, 7, 9]. As a notable example, if each snapshot is connected and of bounded degree, then $O(n^{1.75})$ steps are sufficient to explore the graph [6]. Interestingly, the same upper bound holds in general (connected) graphs in a slightly different model, where the agent is allowed to make at most two moves per step, instead of at most one.


However, the results obtained in previous works do not apply anymore under the constraint called *restlessness*. A $\Delta$-restless temporal path is a path where the agent walking in the graph is not allowed to wait for $\Delta$ steps of time at a vertex before making their next move. For example, this can be used to model non delay tolerant network where a packet can only be stored for a limited amount of time in the nodes, because of memory limitation [4]. Another recent application is the study of paths in virus infection, where a virus can only spread if it infects somebody new before the infected person recovers [8].

We also would like to point out that restless walks in temporal graphs also offer a powerful generalization of properly-colored walks in edge-colored graphs. If the edges of a graph are colored, a walk is said to be properly-colored iff it does not use two edges of the same color consecutively. These graphs themselves offer an interesting generalization of directed graphs (even undirected edge-colored graphs) and there is a rich literature around problems of properly-colored spanning paths or trails (see Chapter 11 of [1]). Given a $k$-edge colored graph, one can build a periodic temporal graph, with infinite lifetime, where the edges available at timeframe $i \mod k$ are exactly the edges colored $i$. If walks have to be $k$-restless, an agent at a vertex $v$ can use edges of any color for their next move, except the one they used to get to $v$, which will take too long to appear again.

In this article, we consider the 1-restless variant of TEXP, i.e. the variant where waiting at a vertex is not possible and the agent therefore has to make exactly one move per step. We denote it by 1-RTEXP. As it is the most restless case, we believe that it is a strong starting point to study the impact of restlessness on the explorability of a graph.

It is easy to see that exploring a temporal graph restlessly can be much more difficult than when we may wait. As a matter of fact, this may not be possible even in a temporal graph with connected snapshots and infinite lifetime, as shown in the simple Example 1.

▶ **Example 1.** For $i > 0$, $G_i$ is the graph on the left of Figure 1 if $i$ is odd, it is the graph on the right of Figure 1 if $i$ is even. Hence, the graph has infinite lifetime, and it is 2-periodical. If the starting vertex (at $t = 1$) is one vertex among $\{D, E, F\}$, then the agent, being forced to move, will be in $\{A, B, C\}$ at time 2, back in $\{D, E, F\}$ at time 3, and so on. It will never be able to visit node $H$.

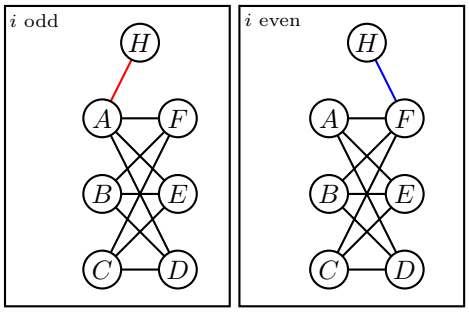


**Figure 1** Snapshots at odd and even timesteps. We use colors to highlight the difference.

Note that the same example works with replacing the subgraph induced by $\{A, B, C, D, E, F\}$ with any (connected) bipartite graph. Following this, a natural question is to find either sufficient conditions for a temporal graph to be explorable restlessly, or tractable cases where 1-RTEXP can be determined in polynomial time. As exploration is not guaranteed even if the lifetime is infinite (and the snapshots are connected), we focus in this work on periodic graphs of infinite lifetime, which are one of the most natural classes of graph of

infinite lifetime that we can encode in finite space (an essential condition for complexity to even make sense). Our main result is to provide a sharp separation between tractability and NP-hardness of 1-RTEXP, based on the period of the graph, summarized in the two following theorems.

▶ **Theorem 2.** *In 2-periodical temporal graphs, 1-RTEXP is polynomially solvable.*

▶ **Theorem 3.** *For any $p \geq 3$, 1-RTEXP is NP-hard in $p$-periodical temporal graphs, even if each snapshot is connected.*

We complement these results by showing some sharp bounds in the number of iterations needed to explore such graphs (whenever exploration is possible).

The article is organized as follows. We give a formal definition of the problem in Section 2. Theorems 2 and 3 are shown respectively in Sections 3 and 4. Exploration time bounds are given in Section 5.

## 2 Preliminaries

▶ **Definition 4.** *A temporal graph is a sequence $G = (G_1, G_2, \dots)$ of graphs $G_t = (V, E_t)$. If the sequence is finite, the length of the sequence is called the lifetime of the graph (otherwise the lifetime is infinite). $G_t$ is called the snapshot of $G$ at time $t$. A (infinite) temporal graph is $p$-periodical if $G_{i+p} = G_i$ for every $i \geq 1$.*

Note that a $p$-periodical temporal graph is fully described by giving the first $p$ snapshots $G_1, \dots, G_p$, thus with a description of (finite) size $O(pn^2)$.

▶ **Definition 5.** *In a temporal graph, a* temporal walk *or* journey *is a sequence of vertices $(v_i, v_{i+1}, \dots, v_j)$ such that for all $k$, $v_k = v_{k+1}$ or $(v_k, v_{k+1}) \in E_k$. The vertices $v_i$ and $v_j$ are called respectively the start vertex and the end vertex and $i$ and $j$ respectively the starting time and ending time. A 1-restless journey, which we simply call a restless journey in this paper, is a journey where $v_k \neq v_{k+1}$, for all $k = i, \dots, j-1$.*

▶ **Definition 6** (1-RTEXP)**.** *Given a temporal graph $G$ and a start vertex $s \in V$, 1-RTEXP asks whether there is a restless journey starting at $s$ at time 1 that contains all vertices of the graph.*

If so, we say that the (temporal) graph is fully explorable, when starting at $s$.

Dealing with periodical temporal graphs, the case of period 1 is trivial, as when $p = 1$ we have $G_{i+1} = G_i$, so the graph is somehow static. Hence, it is fully explorable if and only if it is connected.

Before starting our results, let us mention a result on reachability in $p$-periodical temporal graphs, which easily follows from classical BFS in graphs.

▶ **Lemma 7** (Reachability)**.** *Given a $p$-periodical temporal graph, two vertices $u$ and $w$, and two indices $i, j \in \{1, \dots, p\}$, we can determine in linear time if there exists a restless journey starting at $u$ at some time $t \equiv i \bmod p$ and ending at $w$ at some time $t' \equiv j \bmod p$. Moreover, if such a journey exists, there exists one with length (number of edges) at most $np - 1$.*

**Proof.** We build a graph $G'$ on $np$ vertices $(v, k)$ for $v \in V$ and $k \in \{1, \dots, p\}$. In $G'$ we put an arc from $(v, k)$ to $(v', k+1)$ (or to $(v', 1)$ if $k = p$) if there is an edge $(v, v')$ in snapshot $G_k$. Then, a restless journey in $G$, starting at $u$ at some time $t \equiv i \bmod p$ and ending at $w$ at some time $t' \equiv j \bmod p$, corresponds to a walk in $G'$ starting at $(u, i)$ and ending at $(w, j)$. The existence of such a walk can be determined using a BFS on $G'$.

As $G'$ has linear size (with respect to the input), this can be checked in linear time. Moreover, if such a path exists, there exists a simple one, thus with length at most $np-1$. ◀

If this is the case, we will say that $(v, j)$ *is accessible, or reachable, from* $(u, i)$.

## 3 2-periodical temporal graphs

This section is dedicated to the proof of Theorem 2 which states that 1-RTEXP is polynomially solvable in 2-periodical temporal graphs.

To prove this, we reduce the problem to 2-Sat (restriction of Sat on clauses of size at most 2), which is well known to be polynomial time solvable. The rough idea is to consider two variables per vertex, one saying that we will visit the vertex at an odd time step, the other one saying that we will visit the vertex at an even time step (so at least one of them should be true). We also introduce some variables that represent the order in which we will visit the vertices - more precisely the order of the *first* visit of vertex $v$ at odd and/or even time steps. Corresponding constraints are built thanks to the reachability lemma (Lemma 7). Additional constraints ensure the global feasibility and the fact that the journey starts at $s$.

Let us now formally define the 2-Sat formula. We are given a 2-periodical temporal graph $G$ (i.e., its 2 snapshots $G_1$ and $G_2$), and one start vertex $s \in V$. We construct the following 2-Sat formula $F(G, s)$ :

- Variables:
  - For each vertex $u$ in $V$ we create two variables $u_1$ and $u_2$. As explained above, the variable $u_i$, $i \in \{1, 2\}$, will be true if we visit $u$ at time parity $i$ in our exploration. Let $I$ be the set of variables.
  - For each pair $u_i, v_j$ (for $u, v \in V$, $i, j \in \{1, 2\}$, possibly $u = v$ and/or $i = j$) we create a variable $u_i \rightsquigarrow v_j$. In our construction this variable is true if (1) we visit $u$ at time parity $i$, and (2) either we do not visit $v$ at time parity $j$, or the first visit of $u$ at time parity $i$ is before the first visit of $v$ at time parity $j$.
- Clauses:
  - (VISIT) For each vertex $v$ in $G$ we construct a clause $(v_1 \vee v_2)$, meaning that we have to visit $v$ at time parity 1 or 2 in order to visit the whole graph.
  - (REACH) For each pair $(u_i, v_j)$, if there is *no* restless journey from $u$ at time parity $i$ to $v$ at time parity $j$, we create a clause $((v_j \rightsquigarrow u_i) \vee (\overline{v_j}))$ meaning that we either visit $v$ at time parity $j$ before $u$ at time parity $i$ or do not visit $v$ at time parity $j$ at all.
  - (ORDER) For each pair $u_i, v_j$ in $I$ we create the clause $(\overline{u_i \rightsquigarrow v_j} \vee \overline{v_j \rightsquigarrow u_i})$ ensuring that we do not claim to visit $v$ at time parity $j$ before we visit $u$ at time parity $i$ and at the same time to visit $u$ at time parity $i$ before $v$ at time parity $j$.
  - (START) We create a clause $(s_1)$ for the start vertex $s$ at time parity 1 meaning that we have to go through this state.
  - (FIRST) For other $u_j$ in $I$ we create the clause $(s_1 \rightsquigarrow u_j)$ meaning that we visit $s$ at time 1 before any other vertex, *i.e.* that we start our exploration on $s$ at time odd.

Note that the formula has $2n + 4n^2$ variables and $O(n^2)$ clauses.

We now show in Lemmas 8 and 11 that the temporal graph is fully explorable from $s$ (at time 1) if and only if $F(G, s)$ is satisfiable. This shows Theorem 2. We note that when the graph is fully explorable from $s$, a corresponding restless journey can easily be built from a truth assignment satisfying $F(G, s)$.

▶ **Lemma 8.** *If the graph $G$ can be fully explored starting from $s$ at time 1 then $F(G, s)$ is satisfiable.*

**Proof.** We set the value of each variable according to a restless journey exploring the whole graph starting from $s$ at time 1, as explained in the description of the variables in the formula.

- Clauses (VISIT) are satisfied as the journey visits all vertices, either at time odd or even.
- If we cannot visit $v$ at time parity $j$ after $u$ at time parity $i$ in $G$, either we do not visit $v$ at time parity $j$ ($\overline{v_j}$ is true), or we visit $v$ at time parity $j$ but not $u$ at time parity $i$ ($\overline{v_j \rightsquigarrow u_i}$ is true), or we visit both and then necessarily $v$ at time parity $j$ before $u$ at time parity $i$ ($\overline{v_j \rightsquigarrow u_i}$ is true). Then, clauses (REACH) are satisfied.
- Clauses (ORDER) are verified since following the journey gives a strict order of first visits (note that if $u$ is not visited at time parity $i$ then $u_i \rightsquigarrow v_j$ is false).
- Clauses (START) and (VISIT) are verified since we start our journey from $s$ at time 1. ◀

Assume now that the formula $F(G, s)$ is satisfiable and consider a satisfying truth assignment $S$ of it. Let us define a graph $G_0$ as follows:

- Vertices: For each variable $u_i$ set to true in $S$ the graph contains vertex $u_i$
- Edges: For any two vertices $u_i, v_j$ in the graph, we put the arc $(u_i, v_j)$ if the variable $u_i \rightsquigarrow v_j$ is true in $S$.

▶ **Lemma 9.** *If there is an arc $(u_i, v_j)$ in $G_0$ then $(v, j)$ is accessible from $(u, i)$.*

**Proof.** Assume by contradiction that it is impossible. Then there is by definition in $F(G, s)$ a clause $((v_j \rightsquigarrow u_i) \vee (\overline{v_j}))$. From $v_j \in G_0$ we deduce $v_j$ is true. Thus, to verify the clause, $(v_j \rightsquigarrow u_i)$ must be true. Since we have the above mentioned edge in $G_0$, we also have that $(u_i \rightsquigarrow v_j)$ is true. Consequently, the clause $(\overline{u_i \rightsquigarrow v_j} \vee \overline{v_j \rightsquigarrow u_i})$ of $F(G, s)$ is not satisfied, a contradiction. ◀

Note that by an easy recurrence Lemma 9 shows that when there is a path from $u_i$ to $v_j$ in $G_0$ then $(v, j)$ is accessible from $(u, i)$ in the temporal graph.

▶ **Lemma 10.** *If there is no arc between two vertices $u_i$ and $v_j$ of $G_0$ (neither $(u_i, v_j)$ nor $(v_j, u_i)$), then $(v, j)$ is accessible from $(u, i)$ and vice-versa.*

**Proof.** Assume by contradiction and without loss of generality that $(v, j)$ is not accessible from $(u, i)$. Then there is by definition a clause $((v_j \rightsquigarrow u_i) \vee \overline{v_j})$ in $F(G, s)$. Since $v_j \in G_0$, $v_j$ is true, thus to verify the clause we have that $(v_j \rightsquigarrow u_i)$ is true. By construction, we do have an arc from $u_i$ to $v_j$, a contradiction. ◀

▶ **Lemma 11.** *If $F(G, s)$ is satisfiable then the temporal graph $G$ can be fully explored starting from vertex $s$ at time 1.*

**Proof.** Consider some satisfying assignment $S$ of $F(G, s)$, and the associated graph $G_0$.

Take a topological order of the strongly connected components (SCC) of $G_0$.

This gives us a suitable exploration order of the graph. Indeed, Lemma 9 ensures that if $(u_i, v_j)$ are in the same SCC, then $(v, j)$ is accessible from $(u, i)$ in $G$. If $u_i$ and $v_j$ are in two consecutive SCC in the topological order, then again $(v, j)$ is accessible from $(u, i)$ in $G$: indeed, this follows from Lemma 9 when there is an arc from the SCC of $u_i$ to the one of $v_j$, and from Lemma 10 when there is no arc between the two SCC.

Thus, if we consider an order of vertices of $G_0$ that follow the topological order of SCC, then we can build a journey in the temporal graph that visit all the corresponding vertices. As $G_0$ contains either $u_1$ or $u_2$ for any vertex $u$ of the temporal graph, the journey does explore all the vertices of the temporal graph.

To finish the proof, we shall argue that we can choose a journey that starts at $(s, 1)$. First note that as $s_1$ is true (thanks to the clause (START)) there is a vertex $s_1$ in $G_0$. Moreover, from the clauses (FIRST) we know that there is an arc $(s, u_i)$ for any vertex $u_i$ in $G_0$. Hence, $s_1$ is necessarily in the first SCC in the topological order, and the journey can be chosen to start at $s$ at time 1. ◀

# 4 $p$-periodical temporal graph with $p \geq 3$

We show in this section Theorem 3, i.e., that 1-RTEXP is NP-hard for $p$-periodical graphs, even with connected snapshots, for any $p \geq 3$. We first deal with the case $p = 3$, and then show how the proof can be adapted to the cases $p \geq 4$.

## 4.1 Case $p = 3$

Given a 3-periodical temporal graph $G$ and a start vertex $s$, we study whether the agent can visit all the vertices $V$.

In order to show that the problem is NP-hard, we build a reduction from 3-Sat-(2,2), a restriction of 3-Sat where each variable appears exactly 2 times positively and 2 times negatively. This problem is known to be NP-complete [2].

Given a 3-Sat-(2,2) formula, we build a graph that contains a clause-gadget (described in Section 4.1.1) for every clause, a variable-gadget for every variable (described in Section 4.1.2) and a trap-gadget (described in Section 4.1.3) which makes the snapshots connected, while forcing the agent to explore the graph in some specific order.

### 4.1.1 The clause-gadget

For every clause $C_i$ with literals $\ell_i^1, \ell_i^2, \ell_i^3$, we construct the gadget depicted in Figure 2. We create 8 vertices, $\ell_i^1, \ell_i^2, \ell_i^3, v_i^1, v_i^2, v_i^3, v_i^4, v_i^5$, and 12 edges as follows:

- 3 edges $(v_i^1, \ell_i^j)$, $j \in \{1, 2, 3\}$, present on times $k \equiv 1 \bmod 3$;
- 3 edges $(\ell_i^1, \ell_i^2)$, $(\ell_i^2, \ell_i^3)$, $(\ell_i^3, \ell_i^1)$ present on times $k \equiv 2 \bmod 3$;
- 3 edges $(\ell_i^j, v_i^2)$, $j \in \{1, 2, 3\}$, present on times $k \equiv 3 \bmod 3$;
- 1 edge $(v_i^2, v_i^3)$ present on times $k \equiv 1 \bmod 3$;
- 1 edge $(v_i^3, v_i^4)$ present on times $k \equiv 2 \bmod 3$;
- 1 edge $(v_i^4, v_i^5)$ present on times $k \equiv 3 \bmod 3$.

This construction is illustrated in Figure 2.

Suppose that an agent is at $v_i^1$ at time $k \equiv 1 \bmod 3$ and then moves inside this gadget. The agent will be at one vertex from $\{\ell_i^1, \ell_i^2, \ell_i^3\}$ at time $k + 1$, then on a second vertex from $\{\ell_i^1, \ell_i^2, \ell_i^3\}$ at time $k + 2$ and then joining $v_i^2$ at time $k + 3$. The agent will eventually go through $v_i^3, v_i^4$ and $v_i^5$ in this order as it is the only possible journey. To sum up, the agent will have visited every vertex of the gadget besides one from $\{\ell_i^1, \ell_i^2, \ell_i^3\}$, and will be in $v_i^5$ at $k' \equiv 1 \bmod 3$.

▶ Remark 12. The vertices $\ell_i^1, \ell_i^2, \ell_i^3$ corresponding to literals will be linked to the variable-gadgets. The unvisited vertex will correspond to the literal that will be set to true in the clause in order to verify the formula.

The graph contains one gadget for each clause. More precisely, the gadgets are chained together, where the vertex $v_i^5$ from the gadget of clause $C_i$ is merged with the first vertex $v_{i+1}^1$ of clause $C_{i+1}$. This way, if an agent starts at $v_1^1$ at time $k \equiv 1 \bmod 3$ and moves inside the clause-gadgets, it can traverse all the gadgets and will arrive at $v_m^5$ (where $m$ is the number of clauses) at time $k' \equiv 1 \bmod 3$ and will have visited every vertex except exactly one among $\{\ell_i^1, \ell_i^2, \ell_i^3\}$ for each $i = 1, \ldots, m$.

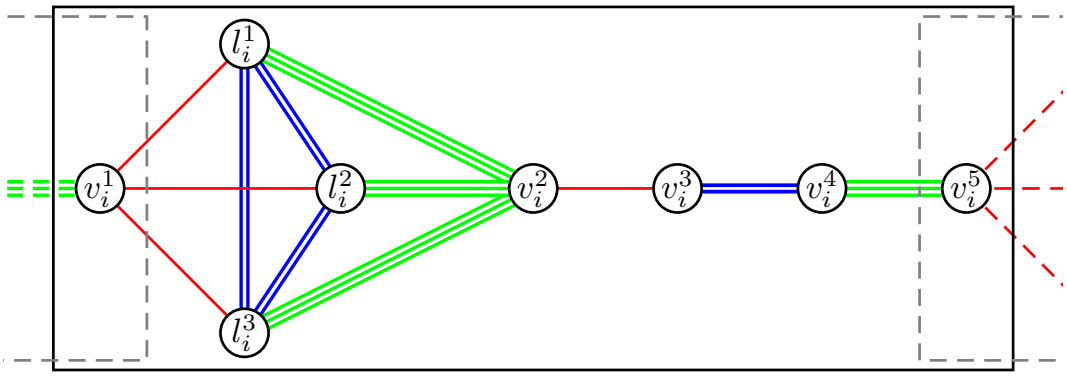


**Figure 2** The gadget associated to the clause $i$. Red edges are present on times $k \equiv 1 \bmod 3$, blue edges on times $k \equiv 2 \bmod 3$ and green edges on times $k \equiv 3 \bmod 3$. For black-and-white readability, the edges present on times 1, 2 and 3 mod 3 are also respectively denoted by simple, double and triple edges.

### 4.1.2 The variable gadget

Every variable $y_i$ appears in 4 clauses, 2 times positively and 2 times negatively. Hence, there are in the clause-gadgets 2 vertices corresponding to $y_i$, and 2 vertices corresponding to $\overline{y_i}$. Then, for every variable $y_i$ we build the gadget depicted in Figure 3: it contains 10 (new) vertices $w_i^j, j = 1, \ldots, 10$, and is linked to the clause-gadgets through the 4 vertices corresponding to $y_i$ and $\overline{y_i}$ (called $y_i^1$, $y_i^2$, $-y_i^1$ and $-y_i^2$ in the figure). Besides the 10 vertices, the gadget contains the following 18 edges:

- Edges $(w_i^3, y_i^1)$, $(w_i^5, -y_i^1)$, $(w_i^4, w_i^8)$, $(w_i^6, w_i^8)$, $(w_i^{10}, y_i^2)$, $(w_i^{10}, -y_i^2)$ and $(w_i^7, w_i^{10})$ present on times $k \equiv 1 \bmod 3$;
- Edges $(w_i^1, w_i^2)$, $(w_i^3, w_i^4)$, $(w_i^5, w_i^6)$ and $(w_i^9, w_i^{10})$ present on times $k \equiv 2 \bmod 3$;
- Edges $(w_i^2, y_i^1)$, $(w_i^2, -y_i^1)$, $(w_i^4, y_i^2)$, $(w_i^6, -y_i^2)$, $(w_i^3, w_i^7)$, $(w_i^5, w_i^7)$ and $(w_i^8, w_i^9)$ present on times $k \equiv 3 \bmod 3$;

This construction is illustrated in Figure 3.

Suppose that an agent is in $w_1$ at some time $k \equiv 2 \bmod 3$ and move inside the variable-gadget (on vertices $w_i^j$, $y_i^1$, $y_i^2$, $-y_i^1$, $-y_i^2$). Then they must go to $w_i^2$, and then can go either to $y_i^1$ or to $-y_i^1$. If they go to $y_i^1$, then they have to go to $w_i^3$ as we consider here the case where they stay inside the gadget (we will show later that they cannot respect the restlessness condition if they leave the gadget). At this point they must go to $w_i^4$, then $y_i^2$ and $w_i^{10}$. They may then visit $w_i^9$, $w_i^8$, $w_i^6$, $w_i^5$, $w_i^7$ and then go back to $w_i^{10}$. Then, either they do the same (now useless) cycle on these vertices, or leave the gadget.

In other words, they can either take $P_1 = (w_i^1, w_i^2, y_i^1, w_i^3, w_i^4, y_i^2, w_i^{10}, w_i^9, w_i^8, w_i^6, w_i^5, w_i^7, w_i^{10})$ (thus visiting the 10 vertices $w_i^j$, and $y_i^1$ and $y_i^2$), or the symmetrical path $P_2 = (w_i^1, w_i^2, -y_i^1, w_i^5, w_i^6, -y_i^2, w_i^{10}, w_i^9, w_i^8, w_i^4, w_i^3, w_i^7, w_i^{10})$ .

Note that in both cases, they are in a vertex in $\{y_i^1, y_i^2, -y_i^1, -y_i^2\}$ only at some time $k' \equiv 1 \bmod 3$.

The variable-gadgets are chained, by merging the vertex $w_i^{10}$ of the gadget of variable $y_i$ to the vertex $w_{i+1}^1$ of the gadget of the variable $y_{i+1}$.

Also, the last vertex $v_m^5$ of the clause-gadgets is linked to the vertex $w_1^1$ of the first variable-gadget by an edge present at time 1 mod 3.

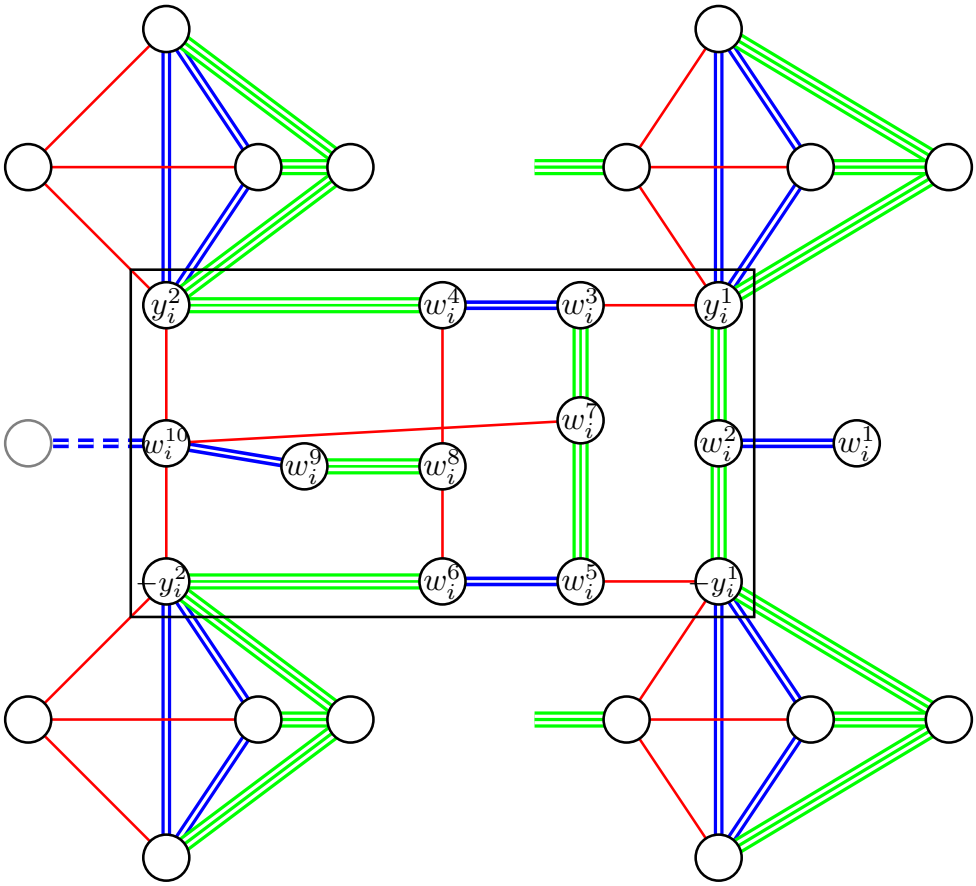


**Figure 3** Unitary variable-gadget (framed). The four adjacent clause-gadgets are depicted too. The meaning of the color is the same as in Figure 2.

### 4.1.3 The trap

With the variable- and clause-gadgets, the snapshots are not connected. We make them connected by adding a gadget, called trap, from which it is impossible to go out. Hence, the trap must be explored last.

The trap, depicted in Figure 4, has 3 vertices $A$, $B$ and $C$ and works as follows. On times $k \equiv 1 \bmod 3$ (resp. $k \equiv 2 \bmod 3$, $k \equiv 0 \bmod 3$), all vertices from the rest of the graphs are adjacent to $A$ (resp. $B$, $C$). If an agent enters the trap, it will alternate between vertices $A$, $B$ and $C$ and cannot go out.

### 4.1.4 Validity of the reduction

We are now ready to prove that the formula is satisfiable if and only if the graph can be fully explored, starting at the vertex $v_1^1$ (gadget of the first clause).

▶ **Lemma 13.** *If the formula is satisfiable then the agent can explore the whole graph.*

**Proof.** Consider a truth assignment $\sigma$ satisfying the formula. We consider the following exploration of the graph. The agent visits first the clause-gadgets, from the first to the last one. As explained earlier, they can do this while exploring all but one vertex among

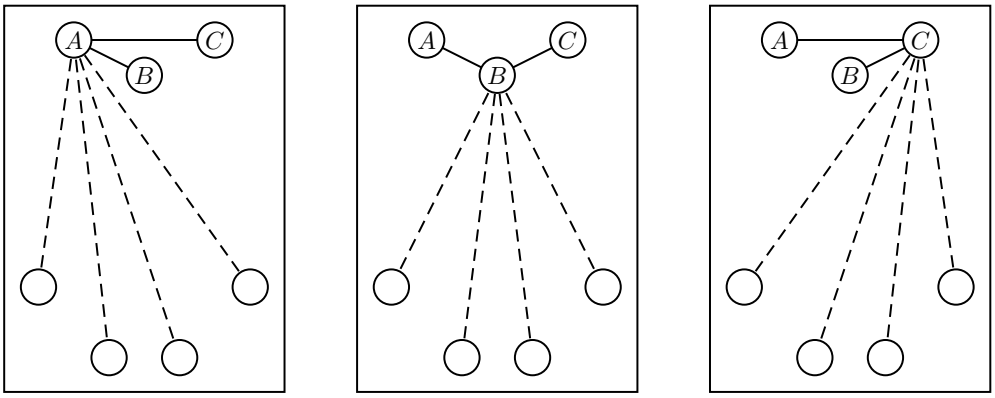


**Figure 4** The three different snapshots of the trap-gadget.

$\{\ell_i^1, \ell_i^2, \ell_i^3\}$ in each clause-gadget. The vertex we choose *not* to visit is one that corresponds to a satisfied literal in $\sigma$.

We are at $v_m^5$ at some time $t \equiv 1 \bmod 3$. We then enter the first variable-gadget at $t+1 \equiv 2 \bmod 3$. Then we visit all the variable-gadgets using path $P_1$ if $y_i$ is set to true, using path $P_2$ if $y_i$ is set to false. After visiting all the variable-gadgets, we go to the trap and visit $A$, $B$ and $C$.

As in the clause-gadget the vertex which was not visited during the exploration of the gadget is set to true, it is visited during the exploration of the corresponding variable-gadget. Thus, all the vertices of the graph are visited. ◀

▶ **Lemma 14.** *If the agent can explore the whole graph then the formula is satisfiable.*

**Proof.** The agent starts at $v_1^1$ at time 1. As noted previously, the trap must be visited at the end of the exploration.

We look at the first vertex $w_1^1$ of the first variable-gadget. The agent is necessarily here at some time $t \equiv 2 \bmod 3$ (from the edge $(v_m^5, w_1^1)$, as the agent cannot be at $w_1^2$ at time $\equiv 2 \bmod 3$, so the edge $(w_1^2, w_1^1)$ cannot be used to reach $w_1^1$). Then, as mentioned earlier, in this variable-gadget they are in $y_1^j$ or $-y_1^j$ only at time $\equiv 1 \bmod 3$. Suppose that they leave the variable-gadget and enters a clause gadget. Then they must take an edge leading to a first vertex $v_i^1$ of the clause gadget, and the agent is stuck there (i.e., they must go to the trap and cannot finish the exploration). So, the agent must follow either path $P_1$ or path $P_2$. They are in $w_1^{10}$, i.e. in the first vertex of the second variable-gadget at some time $\equiv 2 \bmod 3$. More generally, this means that when the agent reaches the first vertex $w_i^1$ of a variable-gadget, then they must stay inside the variable-gadgets, and visit all the subsequent such gadgets, using $P_1$ or $P_2$, till the last vertex $w_n^{10}$ of the last gadget. Then, they must go to the trap.

The exploration starts from vertex $v_1^1$. Suppose now that the exploration leaves some clause-gadget (before $v_m^5$). This must be at some $\ell_i^j$, where they are either at time 2 or 3 mod 3 (coming from $v_i^1$ or some $\ell_i^{j'}$). There is no edge present at time 2 mod 3 incident at some $y_i$-vertex in the variable-gadget.

- If $\ell_i^j$ is a vertex $y_i^1$ or $-y_i^1$, then, to leave the clause-gadget, the agent must reach $w_i^2$ at time 1 mod 3, and is stuck there.
- If $\ell_i^j$ is a vertex $y_i^2$ (resp. $-y_i^2$), then, to leave the clause-gadget, the agent must reach $w_i^4$ (resp. $w_i^6$), then $w_i^8$ at time 2 mod 3, and is stuck here.

In summary, the agent must first visit the clause-gadgets without leaving them. So, they arrive at $v_m^5$ while having visited all the corresponding vertices but one in each clause-gadget. Then, they must follow one path among $P_1$ and $P_2$ in each variable-gadget, and the visit the trap.

We set variable $y_i$ to true (resp. to false) if the agent took path $P_1$ (resp. $P_2$) when visiting the corresponding variable-gadget. As all the vertices have been explored, every $\ell_i^j$ left unexplored in the visit of the clause-gadgets corresponds to a literal set to true in the truth assignment, which concludes the proof.

◀

## 4.2 Case $p \geq 4$

We now show Theorem 3 for $p \geq 4$. The proof is based on a similar reduction from 3-Sat-(2,2). We first extend the clause-gadget and the trap in Section 4.2.1. We then build the variable-gadget in Section 4.2.2. We prove the validity of the reduction in Section 4.2.3.

### 4.2.1 Extending the clause-gadget and the trap

In order to extend the clause-gadget to a larger period, one has to extend its tail. More precisely, we start with the same gadget as for period 3 (up to the fact that edges are present on time modulo $p$ instead of modulo 3). We add $p-3$ vertices $v_i^6, \ldots, v_i^{p+2}$. For $j = 6, \ldots, p+2$, vertex $v_i^{j-1}$ is linked to $v_i^j$ with an edge present on times $k \equiv j-2 \bmod p$. Now, vertex $v_i^{p+2}$ of the gadget of clause $C_i$ is merged with vertex $v_{i+1}^1$ of the gadget of clause $C_{i+1}$.

It is easy to see that the gadget works the same as in the case of period 3: if the agent is at $v_1^1$ at time 1 modulo $p$ and stays inside the clause-gadgets, then they will be at $v_m^{p+2}$ at time 1 modulo $p$, and will have visited every vertex but one $\ell_i^j$ for every $i$.

The trap is generalized in a natural way: it has $p$ vertices, say $f_1, \ldots, f_p$. At time $i$ modulo $p$, $f_i$ is linked to all other vertices of the graph (including the other vertices of the trap). Then, if an agent enters the trap at time $i$ modulo $p$ (in $f_i$), at $i+1$ they must go to $f_{i+1}$, and so on. They can visit all the vertices of the trap but never get out of it.

### 4.2.2 Variable-gadget

The variable-gadget is simpler for $p \geq 4$. As previously, to each variable $y_i$ are already associated 4 vertices in the clause-gadgets, two associated to the literal $y_i$ (denoted $y_i^1$ and $y_i^2$ here) and two associated to the literal $\overline{y_i}$ (denoted $-y_i^1$ and $-y_i^2$ here). The gadget contains $p-1$ vertices $w_i^1, \ldots, w_i^{p-1}$, and the following edges:

- One edge $(w_i^1, w_i^2)$ present on times $k \equiv 2 \bmod p$;
- Two edges $(w_i^2, y_i^1)$ and $(w_i^2, -y_i^1)$ present on times $k \equiv 3 \bmod p$;
- Two edges $(y_i^1, y_i^2)$ and $(-y_i^1, -y_i^2)$ present on times $k \equiv 4 \bmod p$;
- Two edges $(y_i^2, w_i^3)$ and $(-y_i^2, w_i^3)$ present on times $k \equiv 5 \bmod p$;
- If $p \geq 5$, for each $j = 3, \ldots, p-2$, an edge $(w_i^j, w_i^{j+1})$ present on times $k \equiv j+3$.

This construction is illustrated in Figure 5.

Let us call $P_1$ the path $(w_i^1, w_i^2, y_i^1, y_i^2, w_i^3, \ldots, w_i^{p-1})$ and $P_2$ the path $(w_i^1, w_i^2, -y_i^1, -y_i^2, w_i^3, \ldots, w_i^{p-1})$. Note that if an agent is at $w_i^1$ at time 2 mod $p$, then they can follow either $P_1$ or $P_2$, and be at $w_i^{p-1}$ at time 2 mod $p$. If they stay inside the gadget, then these are the only 2 paths that they may follow.

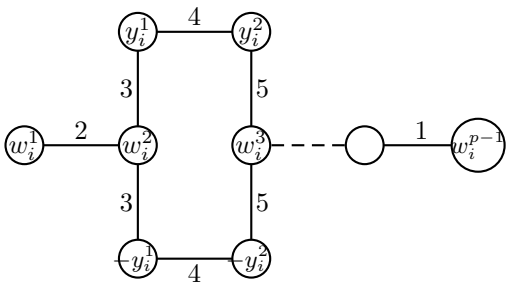


**Figure 5** The variable-gadget. The number above each edge denotes the timeframes where it appears.

To complete the construction, the last vertex $w_i^{p-1}$ of a variable-gadget is merged with the first vertex $w_{i+1}^1$ of the next variable-gadget. Also, there is an edge, present at times 1 mod $p$, between the last vertex $v_m^{p+2}$ of the last clause-gadget and the first vertex $w_1^1$ of the first variable-gadget.

### 4.2.3 Validity of the reduction

We consider the exploration problem starting at vertex $v_i^1$.

▶ **Lemma 15.** *An agent can explore the whole graph if and only if the formula is satisfiable.*

**Proof.** If the formula is satisfiable, let us consider a satisfying assignment $\sigma$. As in the case of period 3, an agent can first visit the clause-gadgets, visiting all vertices but one per clause-gadget which corresponds to a true literal in $\sigma$. They arrive in $v_m^{p+2}$ at time 1 modulo $p$. Then they go to the first vertex of the first variable-gadget, and visit all the variable-gadgets, choosing path $P_1$ if the variable is true in $\sigma$, and $P_2$ otherwise. Finally, they go to the trap and visits it. With the very same argument as previously, we see that they have visited all the vertices of the graph.

Conversely, suppose that an agent can explore all the vertices. Suppose first that they leave a clause-gadget, at some vertex in $\{y_i^1, y_i^2, -y_i^1, -y_i^2\}$. They must be in this vertex at time 2 or 3 modulo $p$.

- If it is $y_i^1$ or $-y_i^1$, they are stuck if they are there at time 2. If they are there at time 3, they can go to $w_i^2$, but then they are stuck there.
- If it is $y_i^2$ or $-y_i^2$, they are immediately stuck.

So, the agent must go through all the clause-gadget first, till $v_m^{p+2}$. Similarly, it is easy to see that an agent cannot leave a variable-gadget:

- they are at vertices $y_i^1$ or $-y_i^1$ at time 4 modulo $p$, and cannot use an edge of the clause-gadget then,
- they are at vertices $y_i^2$ or $-y_i^2$ at time 5 modulo $p$. If $p \geq 5$ they are stuck there, and if $p = 4$, they can go to the first vertex $v_i^1$ of some clause-gadget but then are stuck there.

To conclude, they have to visit first all the clause-gadgets, then all the variable-gadgets, then the trap. We set variable $y_i$ to true (resp. false) if they used path $P_1$ (resp. $P_2$) in the corresponding variable-gadget. As in the case of period 3, from the fact that the agent visits all the vertices of the clause-gadgets, we conclude that every clause has a true literal in the built assignment. ◀

## 5 Exploration time bounds

In this section we focus on the time required to explore the temporal graph (whenever possible). Recall that in the model where the agent is not forced to move, time $O(n^2)$ is sufficient to explore a temporal graph with connected snapshots, and that this bound is tight ($\Omega(n^2)$ steps are necessary for some family of temporal graphs).

We prove here similar results for $p$-periodical temporal graphs under the restlessness 1-constraint.

▶ **Theorem 16.** *Any (restlessly) fully explorable $p$-periodical temporal graph can be explored in at most $pn^2$ steps. Moreover, for every $p \geq 2$ there are families of fully explorable $p$-periodical temporal graphs that require $\Omega(pn^2)$ steps to be fully explored.*

We note that the lower bound $\Omega(pn^2)$ is still valid under the condition that the snapshots are connected (using a trap-structure as in the NP-hardness proof, we can easily make them connected without changing significantly the exploration time bound).

The first part of the theorem, restated in the following lemma, easily follows from Lemma 7.

▶ **Lemma 17.** *Any explorable $p$-periodical temporal graph can be explored restlessly in at most $pn^2$ steps.*

**Proof.** If the graph is explorable, let us consider a journey that visits all vertices. By Lemma 7, we can go from one vertex to the next one in the walk with a walk of at most $pn - 1$ edges. The result follows. ◀

Let us now show the second part of the theorem, restated in the following lemma.

▶ **Lemma 18.** *For every $p \geq 2$ there are families of explorable $p$-periodical temporal graphs that require $\Omega(pn^2)$ steps to be explored.*

**Proof.** We first consider the case where $p$ is even. We build the following graph: first, we consider two even cycles $C_1 = (s, v_2, \ldots, v_{2k}, s)$ and $C_2 = (s, z_2, \ldots, z_{2k}, s)$ of the same size $2k$, sharing a vertex $s$. We choose $k$ (larger than $p$) such that $2k \equiv 2 \bmod p$. In $C_1$, $(s, v_2)$ and every $(v_{2i-1}, v_{2i})$ are present at odd times, and the other edges (including $(v_{2k}, s)$) are present at even times. In $C_2$, starting from $s$ through $z_2$, $z_3$,..., the $i^{\text{th}}$ edge is present at times $\equiv i \bmod p$.

To complete the construction, we split vertex $z_2$ into $2k+1$ identical copies (so each copy is linked to $s$ and to $z_3$). The initial vertex is $s$. Note that the graph has $6k - 1$ vertices.

This consutrction is illustrated in Figure 6

Let us suppose that at some time $t \equiv 1 \bmod p$, the agent is in $s$ and enters $C_2$. Then they must follow the whole cycle, visiting one copy of $z_2$, then $z_3$, ..., $z_{2k}$, being back in $s$ at $t' = t + 2k \equiv 3 \bmod p$. Then, if $p \neq 2$ they must follow $C_1$ (visiting $v_2$, $v_3$,...), being back in $s$ at $t'' = t' + 2k \equiv 5 \bmod p$. In order to be able to go back in $C_2$ again, they must make $p/2 - 1$ tours of $C_1$, so that they are in $s$ at some time $\equiv 1 \bmod p$ (note that this is also true for $p = 2$ as $p/2 - 1 = 0$).

Then, starting at $s$ at time 1, as the agent must go $2k + 1$ times through $C_2$ in order to visit all copies of $z_2$, they must make (at least) $2k$ (complete) tours of $C_2$ and $2k(p/2 - 1)$ (complete) tours of $C_1$, so in total at least $kp$ tours of cycles of length $2k$, leading to an exploration time at least $2k^2p \geq n^2p/18 = \Omega(n^2p)$.

It is easy to see that the graph is indeed explorable with the previous strategy.

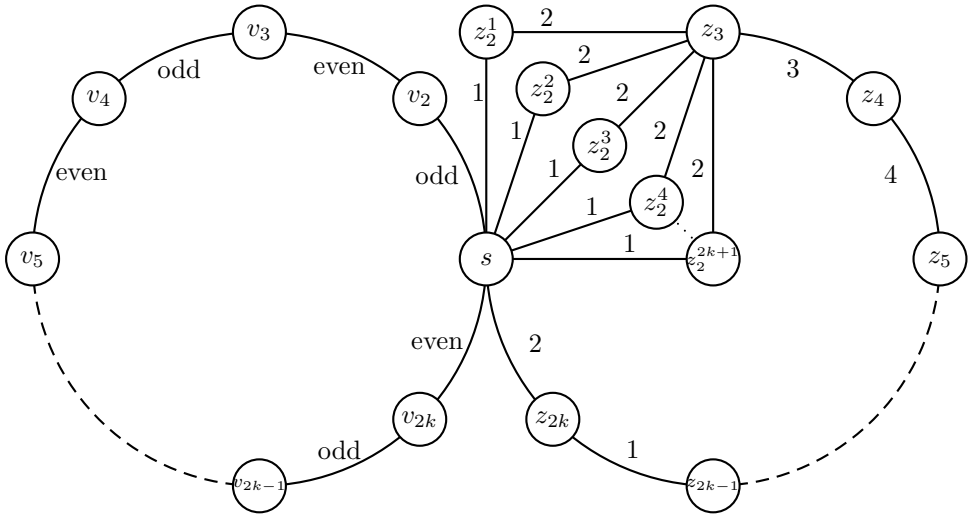


**Figure 6** The graph our construction provides. $C_1$ is on the left, $C_2$ on the right.

If the period is odd, it is no longer possible for an edge to only appear at odd or at even timeframes and our construction requires a little adaptation. We modify $C_1$ as follows:

- Edges $(s, v_2)$ and $(v_{2i-1}, v_{2i})$ are present at times equivalent to $1, 3, \ldots, p-2 \bmod p$, let us call them odd edges;
- Edges $(v_{2i}, v_{2i+1})$ and $(v_{2k}, s)$ are present at times equivalent to $2, 4, \ldots, p-1 \bmod p$, let us call them even edges;
- For each even edge, we add a vertex $w_i$, with an edge $(v_{2i}, w_i)$ present at $p-1 \bmod p$, and an edge $(w_i, v_{2i+1})$ (or $(w_i, s)$) present at $0 \bmod p$.

For example, Figure 7 illustrates the changes we make to the graph of Figure 6 if $p$ is odd.

This way, if the agent is in $C_1$, they will alternate between odd and even edges, up to time $p-1 \bmod p$, where they are at an "odd" vertex $v_{2i}$, has to use the extra vertex $w_i$, leaving $w_i$ at $0 \bmod p$, reaching $v_{2i+1}$ (or $s$ if $i = k$) at time $1 \bmod p$. Then, in $p$ units of times they travel from $v_i$ to $v_{i+p-1}$. We set the length of the cycle $C_1$ (on $s$ and the vertices $v_i$) to be $2k \equiv 2 \bmod (p-1)$.

Then in this configuration, if the agent starts visiting the tour $C_1$ at $s \equiv 3 \bmod p$, it ends the tour (visiting some vertices $w$ during the travel) at $s$ at time $t' \equiv 5 \bmod p$, and so on. After $(p-3)/2$ such tours, it will be at $s$ at $t \equiv 0 \bmod p$, will start a new tour of $C_1$, will be at $v_{2k}$ at time $t' \equiv 0 \bmod p$, and then can use the edge $(v_{2k}, s)$ to be back on $s$ at time $\equiv 1 \bmod p$.

So, to visit all copies of $z_2$, the agent must make at least $2k$ (complete) tours of $C_2$, and $2k(p-1)/2$ tours of $C_1$. As tours have (at least) $2k$ vertices, we get again a lower bound of $2k^2p = \Omega(pn^2)$ to explore the whole graph restlessly.

Here again, the previous strategy shows that the graph is fully explorable. ◀

## 6 Conclusion and perspective

In the previous section, we proved that the maximum amount of time required for the exploration of an explorable $p$-periodic graph of $n$ vertices is in $\Theta(pn^2)$. However, we only proved that this value is somewhere between $\frac{pn^2}{18}$ and $pn^2$ and we leave its exact value as an open question.

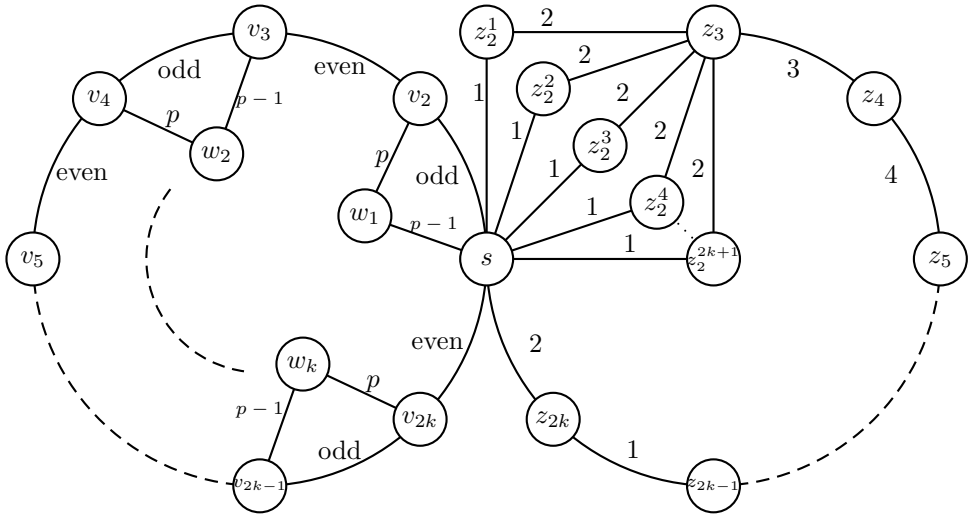


**Figure 7** We add an extra vertex $w_i$ for every odd edge. Note that here, even edges are present on timeframes $2, 4, ..., p-1$ and odd edges on timeframes $1, 3, 5, ..., p-2$ but not $p$, which is dealt with separately.

Another natural extensions of our work could be to investigate the complexity of $\Delta$-restless exploration with $\Delta > 1$ or cases where the agent walking in the graph is allowed to move by more than one edge at each time step but we do not know if those cases would be more interesting than the case $\Delta = 1$.

A problem that we believe would be of great interest is the study of the tractability of the NP-complete cases of the problem for some fixed relevant parameters. This could include structural parameters of the underlying graph (the timeless graph that contains all the edges that appear at any given time) or parameters related to the temporality. In particular, our gadgets were inspired by our previous works on edge-colored graphs and most edges are present in only one snapshot, which makes the graph change drastically from a timeframe to the next. We believe it could be of great interest to study the complexity of cases where the graph cannot change too much in one timeframe.

## References


1 Jørgen Bang-Jensen and Gregory Z. Gutin. *Digraphs – Theory, Algorithms and Applications, Second Edition.* Springer Monographs in Mathematics. Springer, 2009.

2 Piotr Berman, Marek Karpinski, and Alex D. Scott. Approximation hardness of short symmetric instances of MAX-3SAT. *Electron. Colloquium Comput. Complex.*, TR03-049, 2003. `arXiv:TR03-049`.

3 Hans L. Bodlaender and Tom C. van der Zanden. On exploring always-connected temporal graphs of small pathwidth. *Inf. Process. Lett.*, 142:68–71, 2019. `doi:10.1016/j.ipl.2018.10.016`.

4 Arnaud Casteigts, Anne-Sophie Himmel, Hendrik Molter, and Philipp Zschoche. Finding temporal paths under waiting time constraints. *Algorithmica*, 83(9):2754–2802, 2021. `doi:10.1007/s00453-021-00831-w`.

5 Thomas Erlebach, Michael Hoffmann, and Frank Kammer. On temporal graph exploration. *J. Comput. Syst. Sci.*, 119:1–18, 2021. `doi:10.1016/j.jcss.2021.01.005`.

6 Thomas Erlebach, Frank Kammer, Kelin Luo, Andrej Sajenko, and Jakob T. Spooner. Two moves per time step make a difference. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132

of *LIPIcs*, pages 141:1–141:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ICALP.2019.141.

7 Thomas Erlebach and Jakob T. Spooner. Parameterized temporal exploration problems. In James Aspnes and Othon Michail, editors, *1st Symposium on Algorithmic Foundations of Dynamic Networks, SAND 2022, March 28-30, 2022, Virtual Conference*, volume 221 of *LIPIcs*, pages 15:1–15:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.SAND.2022.15.

8 Petter Holme. Temporal network structures controlling disease spreading. *Phys. Rev. E*, 94:022305, August 2016. doi:10.1103/PhysRevE.94.022305.

9 David Ilcinkas, Ralf Klasing, and Ahmed Mouhamadou Wade. Exploration of constantly connected dynamic graphs based on cactuses. In Magnús M. Halldórsson, editor, *Structural Information and Communication Complexity – 21st International Colloquium, SIROCCO 2014, Takayama, Japan, July 23-25, 2014. Proceedings*, volume 8576 of *Lecture Notes in Computer Science*, pages 250–262. Springer, 2014. doi:10.1007/978-3-319-09620-9_20.

10 Othon Michail and Paul G. Spirakis. Traveling salesman problems in temporal graphs. *Theor. Comput. Sci.*, 634:1–23, 2016. doi:10.1016/j.tcs.2016.04.006.

**Revision Notice**

This is a revised version of the eponymous paper appeared in the proceedings of SAND 2023 (LIPIcs, volume 257, https://www.dagstuhl.de/dagpub/978-3-95977-275-4, published in June, 2023), in which the variable-gadget construction of Section 4.1.2 has been changed (in particular the gadget depicted in Figure 3), to correct a flaw in the initial version.

*Dagstuhl Publishing – July 5, 2023.*