Restless Exploration and Token Dissemination in Vertex-Permuted Temporal Graphs

Kamran Ayoubi
Department of Computer Science and Software Engineering, Concordia University, Montreal, Canada

Lata Narayanan
Department of Computer Science and Software Engineering, Concordia University, Montreal, Canada

Abstract

In the temporal graph exploration problem, an agent wishes to visit all the vertices of a temporal graph, moving at most one edge in every time step. In *restless exploration*, the agent is required to move in every step, and cannot wait at a vertex. We study the problem of restless exploration in vertex-permuted graphs, which are a class of temporal graphs in which the topology of the graph stays the same in every time step. In other words, in a vertex permuted temporal graph, in every step i, the graph G_i is isomorphic to the same base graph G. We give a precise characterization of graphs G such that restless exploration is possible in *every* vertex-permuted graph with base graph G. Our technique is based on an a characterization of networks in which there is an online distributed algorithm for restless token dissemination. Finally we describe some families of graphs in which restless exploration is always possible, and some in which it is not.

2012 ACM Subject Classification Theory of computation; Theory of computation → Dynamic graph algorithms

Keywords and phrases Temporal graphs, Vertex permuted graphs, Restless exploration, Periodic graphs, Token dissemination

Digital Object Identifier 10.4230/LIPIcs.SAND.2025.12

1 Introduction

Many networks are now understood to be dynamic rather than static, with nodes and edges changing over time. This dynamic nature of networks applies to many real-world scenarios, such as communication networks, transportation networks, social networks, chemical reaction networks, and more. For instance, in wireless networks formed by mobile nodes, such as IoT devices, buses, satellites, or autonomous mobile robots, links are created or broken because of the mobility of nodes. In wireless sensor networks, where sensor nodes are resource-constrained, nodes turn off their batteries to conserve energy, once again causing edges (and nodes) to appear and disappear. In road networks, traffic changes impact the travel time of roads which in turn changes the weights of the links. The dynamic nature of such networks greatly increases the challenge of solving problems that are well-understood in the static setting.

Many models of dynamic networks [25] have been studied in the last two decades, which have been variously referred to as time-varying graphs [7, 16], fixed schedule dynamic networks [27], evolving graphs [15], and temporal graphs [20]. For our purposes, a temporal graph is defined as a sequence of graphs $G_1, G_2, \ldots, G_L$, where each $G_i = (V, E_i)$ is a static undirected graph, and all graphs in the sequence have the same vertex set V, but possibly different edge sets. The number L is called the *lifetime* of the temporal graph. A *temporal walk* is a time-respecting walk in a temporal graph, that is a sequence of time-vertex pairs $(v_1, t_1), \ldots, (v_k, t_k), (v_{k+1}, t_{k+1})$ such that for each $i \in [k]$, the edge $\{v_i, v_{i+1}\}$ is in the graph

G_{t_i} and $1 \leq t_i < t_{i+1} \leq L+1$ [11]. An *exploration schedule* is a temporal walk that visits all vertices of the graph [11, 22]. A temporal graph is called *explorable* if it admits an exploration schedule starting at *any* vertex; it is unexplorable otherwise.

The process of exploration can be described as an *agent*, starting at a source vertex, and traveling at most one edge in the temporal graph in each time step, and visiting every vertex in the graph. The term "agent" is used to refer to a mobile agent or a process or a packet that travels through the graph.

The notion of *restless* exploration was introduced in [5]: in k-restless exploration, the agent must leave a vertex within at most k steps after arriving, that is, $t_i < t_{i+1} \leq t_i + k$. In 1-restless exploration, the agent cannot wait at a vertex, and must leave in the step after it arrives. Such a restriction has also been studied in the context of communication networks and is motivated by constraints on buffer space. In particular, in *hot-potato routing* or *deflection routing* [4, 6, 14, 23], an intermediate node has to forward a packet to a neighboring node as soon as it receives it, and cannot store it in order to wait for the best link to become available.

The notion of *restless* exploration was introduced in [5]: in k-restless exploration, the agent must leave a vertex within at most k steps after arriving, that is, $t_i < t_{i+1} \leq t_i + k$. In 1-restless exploration, the agent cannot wait at a vertex, and must leave in the step after it arrives. Such a restriction has also been studied in the context of communication networks and is motivated by constraints on buffer space. In particular, in *hot-potato routing* or *deflection routing* [4, 6, 14, 23], an intermediate node has to forward a packet to a neighboring node as soon as it receives it, and cannot store it in order to wait for the best link to become available.

In [22], it was shown that any temporal graph that is connected in every step, and has a lifetime of at least $|V|^2$ is explorable. However, not all such graphs are 1-restless explorable. For example, consider a 2-periodic temporal graph with 3 vertices $\{1, 2, 3\}$; in odd steps, the edge set is $\{\{1, 2\}, \{2, 3\}\}$, and in even steps, the edge set is $\{\{1, 2\}, \{1, 3\}\}$. Starting at vertex 1, in odd steps, due to the restless constraint, the agent must move from vertex 1 to vertex 2, and in even steps, it must move from vertex 2 to vertex 1, and vertex 3 is never visited. Bellito et al [5] showed that 1-restless explorability is NP-hard to determine in a 3-periodic temporal graph, while it can be determined in polynomial time in a 2-periodic graph.

In this paper, we study 1-restless exploration in a specific type of temporal graph, referred to as a *Vertex Permuted Graph (VPG)*, introduced in [2]. A VPG is a temporal graph in which the structure of the graph remains fixed at each time step throughout the graph's lifetime, though the vertices may assume different positions in the graph. For example, birds flying in formation are known to swap positions to share 'lift' while maintaining the same structure of the formation. Similarly, elite cyclists who ride in pelotons, swap their positions after specific intervals to share the fatigue of being in front of the formation. In the context of sensor networks, different nodes may periodically assume the role of clusterhead, with the cluster having a star topology, in order to balance energy consumption across cluster members. All the above are examples of vertex-permuted graphs. While only ring topologies are studied in [2], we study VPGs with arbitrary connected topologies.

1.1 Our Contributions

We give a precise characterization of graphs G such that every vertex-permuted graph with base graph G is 1-restlessly explorable. If the base graph has more than one connected component, then clearly the vertex-permuted graph where the permutation remains the same

in every step is not explorable. So it is necessary that the base graph is connected. We show that *every* vertex-permuted graph with base graph $G = (V, E)$ is 1-restlessly explorable if and only if G does not have a so-called *blocking set* of size less than $|V|/2$. Our result relies on a distributed restless algorithm for token dissemination that we prove is successful on any temporal graph based on a graph G that does not have a blocking set. On the other hand, if the graph G does have a blocking set, then there exists a vertex-permuted graph based on G in which 1-restless token dissemination is impossible.

We then consider 1-restless explorability of specific families of graphs. We show that all VPGs with base graphs that are regular graphs and grids of even length are 1-restlessly explorable, while graphs containing a vertex of degree 1, and some bipartite graphs, including grids of odd length, are not 1-restlessly explorable.

1.2 Related Work

The exploration of temporal graphs, introduced by Michail and Spirakis [22], addresses the problem of designing temporal walks that visit all vertices of a dynamic graph. They established that any temporal graph with n vertices that is connected in every step, can be explored in $\mathcal{O}(n^2)$ time steps, a bound later confirmed to be tight by Erlebach et al. [11]. Improved bounds have been given for specific graph classes, such as planar graphs, cycles, grids, and graphs with bounded treewidth [11]. Several studies have explored different facets of temporal graph exploration, including periodically varying graphs [16], parameterized and non-strict exploration problems [12, 13], delay-robust routing under unreliable connections [17], and exploration of temporal graphs with a certain structural property [9, 19].

Bellitto et al. [5] introduced the problem of *k-restless exploration*, where an agent cannot remain at any vertex for more than k time steps. They characterized the complexity of determining if 1-restless exploration is possible in p-periodic graphs; proving that it is solvable in polynomial time for $p = 2$ but NP-hard for $p \geq 3$. The restless property has also been studied in the context of temporal reachability and path-finding problems [8, 26, 28], where the objective is to find time-respecting paths under bounded waiting times.

In the problem of k-token dissemination, or k-gossip, there are k tokens initially present at some of the nodes of the network, and the problem is to disseminate the tokens to all nodes in the network. Token dissemination in dynamic networks was first proposed in [21], where a simple flooding algorithm was shown to work in $O(kn)$ steps, under the constraint that the network is always connected. Subsequent work has focused on upper and lower bounds on the number of rounds required for token dissemination for both deterministic and randomized algorithms, under different connectivity constraints, different adversarial models, and the ability/inability to combine tokens [3, 1, 18, 24, 10]. As far as we are aware, there is no prior work on token dissemination under restless constraints. We study token dissemination *under restless constraints* on vertex-permuted temporal graphs.

In the problem of k-token dissemination, or k-gossip, there are k tokens initially present at some of the nodes of the network, and the problem is to disseminate the tokens to all nodes in the network. Token dissemination in dynamic networks was first proposed in [21], where a simple flooding algorithm was shown to work in $O(kn)$ steps, under the constraint that the network is always connected. Subsequent work has focused on upper and lower bounds on the number of rounds required for token dissemination for both deterministic and randomized algorithms, under different connectivity constraints, different adversarial models, and the ability/inability to combine tokens [3, 1, 18, 24, 10]. As far as we are aware, there is no prior work on token dissemination under restless constraints. In this paper, we show the connection between restless token dissemination and restless graph exploration in vertex-permuted temporal graphs.

In [2], the notion of vertex permutation dynamism was introduced, and the authors gave an algorithm for dispersion of mobile agents in vertex-permuted rings. Vertex-permuted graphs of other topologies were not studied. We study restless exploration in vertex-permuted graphs of *arbitrary* topologies.

1.3 Organization of the paper

In Section 2, we describe the notation we use and give some preliminary definitions. Section 3 contains our main result: a characterization of vertex permuted graphs that are restlessly explorable. In Section 4, we use the characterization to determine the explorability of vertex-permuted graphs based on many well-known classes of graphs. We discuss future directions in Section 5.

2 Preliminaries and Definitions

For our purposes, a *temporal graph* is a sequence of graphs $G_1 = (V, E_1), G_2 = (V, E_2), \ldots, G_L = (V, E_L)$, where each $G_i = (V, E_i)$ is a static undirected graph, and all G_i have the same vertex set V, but possibly different edge sets. The number L is called the *lifetime* of the temporal graph. Each graph G_i corresponds to the state of the temporal graph at time step i, with an edge $e \in E_i$ representing a connection between two vertices that is available at time i. The sequence of graphs models the evolution of a network over time [11]. A temporal graph $\mathcal{G}$ with lifetime L is called *periodic* with a fixed period p if $G_i = G_{i+p}$ for every $1 \leq i \leq L - p$.

Given a temporal graph $G_1, G_2, \ldots, G_L$, with $G_i = (V, E_i)$, a *temporal walk* in the graph is an alternating sequence of time-vertex pairs $(v_1, t_1), \ldots, (v_k, t_k), (v_{k+1}, t_{k+1})$ such that for each $i \in [k]$, the edge $\{v_i, v_{i+1}\}$ is in the graph G_{t_i} and $1 \leq t_i < t_{i+1} \leq L + 1$ [11]. In this paper, we only consider *strict* temporal walks, in which at most one edge can be traversed in a time step. A temporal walk is k-restless if for every i, we have $t_{i+1} \leq t_i + k$. In this paper, we focus exclusively on 1-restless walks, that is, $t_{i+1} = t_i + 1$, and in the sequel, we use restless to mean 1-restless.

Given a source vertex v, an *exploration* schedule is a temporal walk that starts with v and visits every vertex in the graph. Note that a vertex may be visited more than once. A restless exploration schedule is a restless temporal walk that visits every vertex in the graph. We say a graph is *restlessly explorable* if there exists a restless exploration schedule starting from every vertex in the graph.

Vertex-Permuted Graphs (VPGs) are a class of temporal graphs where the vertices are permuted at each time step, but the underlying structure of the graph remains unchanged. Formally:

▶ **Definition 1** (Vertex Permuted Graph). *Given a graph $G = (V, E)$ and a permutation sequence $\Pi = \pi_1, \pi_2, \ldots, \pi_k$ where each π_i is a permutation of V, the vertex permuted graph $\mathcal{G}^\Pi$ is a temporal graph $\mathcal{G} = (G_1, G_2, \ldots G_L)$ in which $G_i = (V, E_i)$, where $E_i = \{(\pi_i(u), \pi_i(v)) \mid (u, v) \in E\}$, that is, for all $i \in [L]$, the graph G_i is isomorphic to graph G where the isomorphism is defined by the permutation π_i. The graph G is called the base graph of $\mathcal{G}^\Pi$; we also say that $\mathcal{G}^\Pi$ is based on G.*

A **Vertex Permuted Cycle (VPC)** with n vertices is a vertex permuted graph whose base graph is C_n. Figure 1 illustrates six consecutive time steps of a VPC. Observe that in contrast to the model of a temporal cycle in which a different edge may be unavailable in each step, but all other connections between nodes remain the same [11], in a VPC, the connections between nodes can vary arbitrarily, while still maintaining a cycle topology.

In this paper, we study the restless explorability of vertex-permuted graphs. As already mentioned, we restrict ourselves to base graphs that are connected. We first describe an online and distributed *token dissemination procedure*, and then introduce the concepts of *tokenizable* and *untokenizable* graphs.

Restless Token Dissemination Procedure

Given a VPG $\mathcal{G}^{\Pi}$ with n vertices, and a starting node $s \in V$, the token dissemination procedure proceeds as follows:

- At time step 1, the starting vertex s is assigned a token (henceforth, a vertex with a token is referred to as a *token holder*).
- For every $t \geq 1$, any vertex that has the token sends it to all its neighbors (it does not keep the token). In other words, a vertex becomes a token holder at time $t+1$ if it is adjacent to at least one token holder at time step t.

Observe that the above token dissemination procedure is simply a flooding algorithm under the restless constraint, unlike the 1-token dissemination algorithms in [21], in which a vertex repeatedly broadcasts the token, once it receives it. It is online in the sense that a vertex has no knowledge of future graphs in the sequence in order to determine the action at time step t. Note also that a vertex can both send and receive tokens in the same time step, in which case it remains a token-holder in the following time step. To illustrate, consider our token dissemination procedure applied to the *VPC* in Figure 1. The propagation of tokens in every time step is shown in Figure 1. Starting from vertex 1, tokens propagate based on the adjacency of token holders at each time step. As the vertices permute at each step, tokens continue to spread across the graph. At time step $t = 2$, vertices 2 and 8 become token holders, as they were the neighbors of vertex 1 in time step 1, which was the only token holder at that time. At time step $t = 3$, vertices 3, 6, 4, and 7 become token holders, as they were neighbors of the holders in time step $t = 2$. By continuing this process, all vertices become token holders by time step $t = 5$.

For the token dissemination procedure initiated at time step 1 from the starting vertex s in the graph G^{Π}, we use the following notations throughout this paper:

- $TH(G^{\Pi}, s, t)$ refers to the set of token holders at time step t.
- $NoT(G^{\Pi}, s, t)$ denotes the number of token-holders at time step t, that is, $NoT(G^{\Pi}, s, t) = |TH(G^{\Pi}, s, t)|$.
- $HS(G^{\Pi}, s, t)$ is the set of all vertices in G that have held a token at at some time step i with $1 \leq i \leq t$, that is, $HS(G^{\Pi}, s, t) = \bigcup_{i=1}^{t} TH(G^{\Pi}, s, i)$.

For example, for the token dissemination procedure shown in Figure 1, we have $TH(G^{\Pi}, 1, 3) = \{3, 4, 6, 7\}$, $NoT(G^{\Pi}, 1, 3) = 4$, and $|HS(G^{\Pi}, 1, 3)| = 7$. Also for every vertex permuted graph G^{Π}, and any start vertex s, we have $TH(G^{\Pi}, s, 1) = HS(G^{\Pi}, s, 1) = \{s\}$ and $NoT(G^{\Pi}, s, 1) = 1$.

We now present the definition of a *tokenizable graph*.

▶ **Definition 2** (Tokenizable graph). *A graph $G = (V, E)$ with n vertices is called* tokenizable *if, for every starting vertex $s \in V$ and every permutation sequence Π, we have $HS(G^{\Pi}, s, n) = V$. Furthermore, if G is tokenizable and there exists a starting vertex $s \in V$ and a permutation sequence Π such that $TH(G^{\Pi}, s, n) \neq V$, then G is* partially tokenizable*; otherwise, it is* fully tokenizable.

A graph G is called untokenizable, if it is not tokenizable. Formally:

▶ **Definition 3** (Untokenizable graph). *A graph $G(V, E)$ is called untokenizable if there exists a series of permutations Π and a start vertex $s \in V$ such that $HS(G^{\Pi}, s, n) \neq V$.*

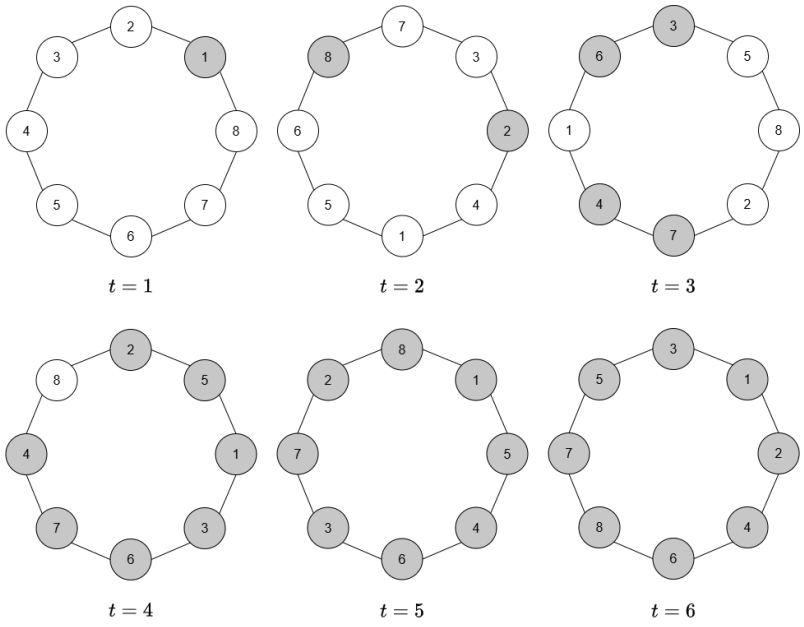

Figure 1 Token dissemination procedure initiated from vertex 1; by time step 5, all vertices have become token holders.

We will show later that it is enough to restrict to the first n graphs in G_Π: if there exists a vertex that does not receive the token before time n, then it can never receive the token in G^Π. Figure 1 showed a VPC with 8 vertices in which all vertices become token holders at time step 5 and will remain token holders afterwards. However, as shown in Figure 2, there exist VPCs with 8 vertices such that there is no time step in which all vertices simultaneously hold a token, even though every vertex has received a token at or before time step 4. Furthermore, the permutation sequence for this VPC is 2-periodic. This shows that a cycle of 8 vertices is not fully tokenizable. We will show in the upcoming section that all cycles are either fully or partially tokenizable.

Figure 3 gives an example of a VPG and the token dissemination procedure starting at vertex 1. We will show in Theorem 12 that the base graph shown in Figure 3 is fully tokenizable; that is, under all permutation sequences and all possible start vertices, there will always be 5 token holders after at most 5 steps.

Note that if a graph is disconnected, then it is untokenizable, as under the identity permutation sequence, tokens cannot be sent between two different connected components in the VPG G^Π. In Figure 4, we present an example of a vertex-permuted graph based on an untokenizable graph G. When starting the token dissemination procedure from vertex 5, vertex 1 will never become a token holder under the given permutation sequence. Notably, the sequence shown is 2-periodic, and satisfies $\Pi_5 = \Pi_3 = \Pi_1$ and $\Pi_4 = \Pi_2$, and even if this pattern continues indefinitely, node 1 will remain without a token.

For any subset $V' \subseteq V$, let $N(V')$ denote the set of neighbors of vertices in V'.

▶ **Definition 4** (Blocking Set (BS) and Minimum Blocking Set (MBS)). *Given a graph $G = (V, E)$, a* blocking set (BS) *of G is an independent set S of vertices such that $|S| = |N(S)|$. Furthermore, if S is the smallest such set, then it is called a* minimum blocking set (MBS) *of G.*

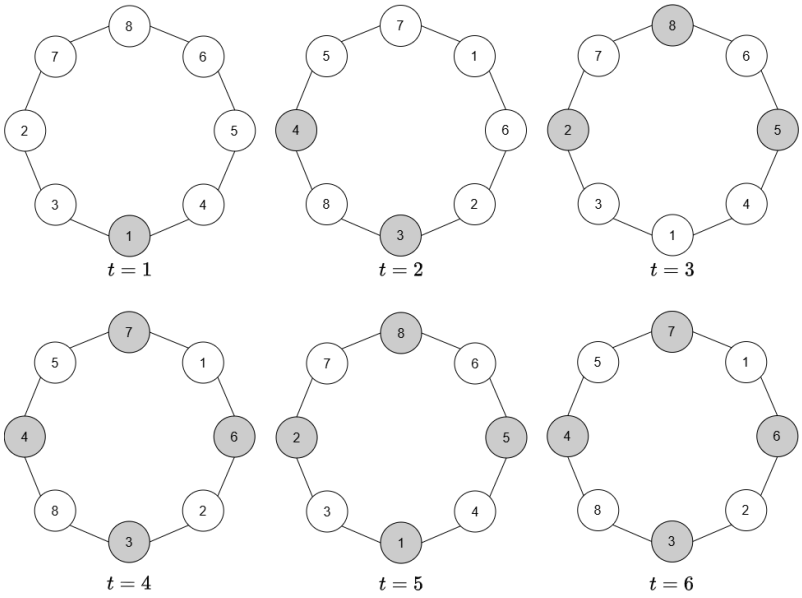

Figure 2 A 2-periodic vertex-permuted cycle in which all vertices receive the token by time 4, but there is no time step in which all vertices simultaneously hold the token.

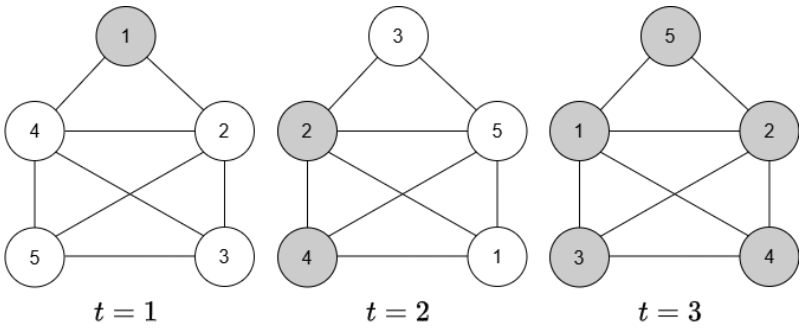

Figure 3 Token dissemination procedure in a vertex permuted graph based on a fully tokenizable graph.

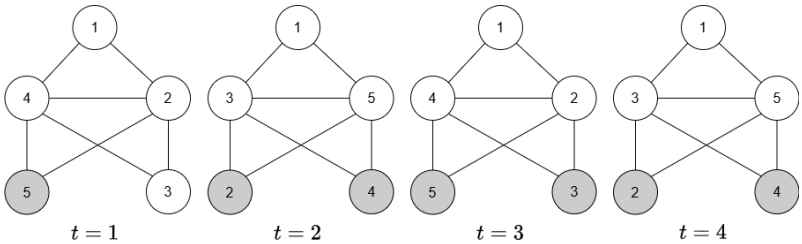

Figure 4 First four steps of the token dissemination procedure in a 2-periodic VPG, where vertex 1 will never hold a token.

Note that for a graph with n vertices, any blocking set can be of size at most $\lfloor n/2 \rfloor$. In Figure 5, the graph in part (b) contains 4 blocking sets: $\{1,5\}$, $\{1,3\}$, $\{3,5\}$, and $\{1\}$. Among these, the minimum blocking set is $\{1\}$. In contrast, in part (a) the graph has three blocking sets $\{1,5\}$, $\{1,3\}$, $\{3,5\}$, all of which are *MBS* .

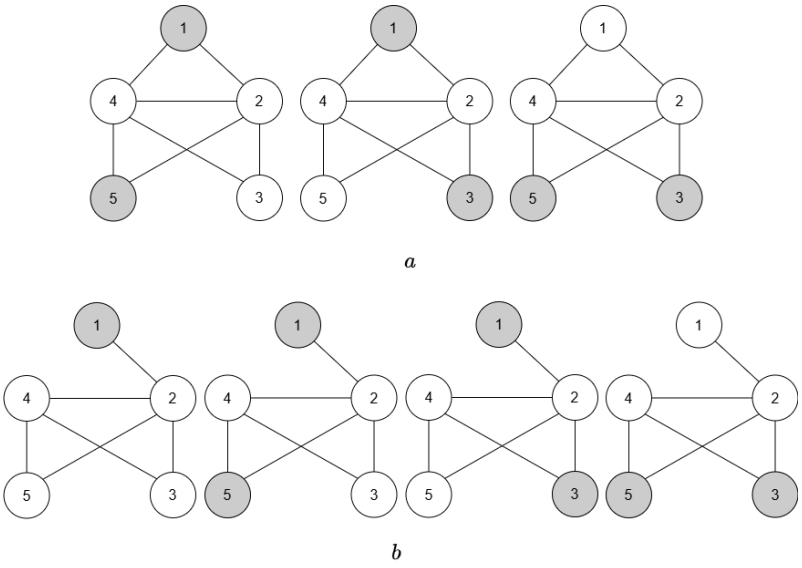

Figure 5 Possible minimum blocking sets for two different graphs.

3 Restless explorability in *vertex permuted graphs*

This section addresses the restless explorability problem in vertex-permuted graphs. We investigate the connection between tokenizability and the existence of a blocking set and provide a characterization of the structure of graphs that can be explored under restless constraints and vertex permutations.

3.1 A characterization of tokenizable graphs

In this subsection, we give a structural characterization of tokenizable graphs by showing the relationship of tokenizability to the existence of a blocking set. First, we establish several lemmas that will support the proofs of the main results. Recall that $N(V')$ is the set of neighbors of vertices in V' for any $V' \subseteq V$.

▶ **Lemma 5.** *If a connected graph $G = (V, E)$ contains a subset $I \subset V$ that is an independent set such that $|I| \geq |N(I)|$, then there exists a blocking set $S \subseteq I$ with $|S| \leq |N(I)|$.*

Proof. If $|I| = |N(I)|$, then the lemma follows trivially, since I is the blocking set. If $|I| > |N(I)|$, we prove the claim by giving a simple iterative procedure to find such a subset.

While $|I| > |N(I)|$, remove an arbitrary vertex from I and update $N(I)$. At each step, the size of I decreases by 1, while the size of $N(I)$ does not increase. Since G is connected, $N(I)$ is never empty, ensuring that the process terminates with $|I| = |N(I)| > 0$, where I is the required blocking set. This concludes the proof. ◀

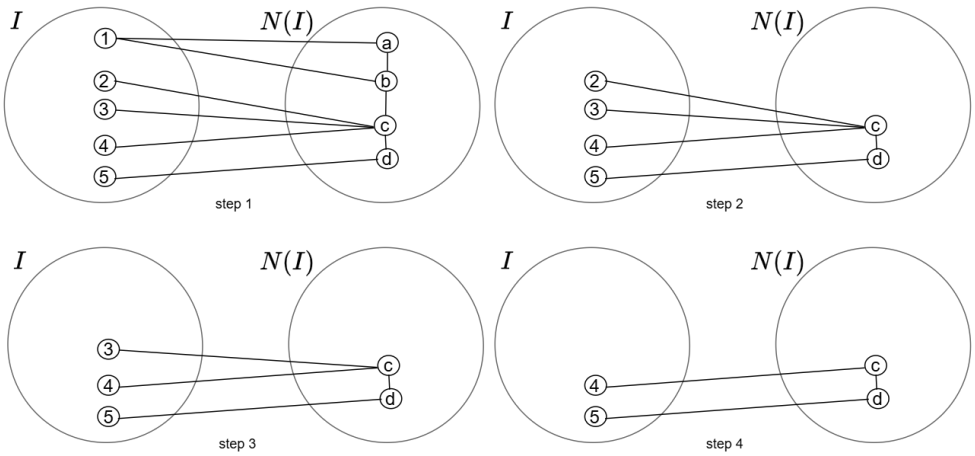

Figure 6 An illustration of the iterative process for finding a blocking set.

Figure 6 illustrates an example of this procedure: I is an independent set with 5 vertices, and $N(I)$ is set of neighbors of vertices in I. Since $|I| > |N(I)|$, we start by removing vertex 1 from I, which causes the removal of vertices a and b from $N(I)$. The updated sets are $I = \{2, 3, 4, 5\}$ and $N = \{c, d\}$. In the next step, we remove vertex 2 from I but $N(I)$ remains unchanged since c and d have other neighbors in I. Similarly, removing vertex 3 does not alter $N(I)$. At this point, the process terminates with $|I| = |N(I)|$, forming a blocking set. Note that selecting different vertices during the steps may lead to different blocking sets.

▶ **Lemma 6.** *Suppose for a connected graph $G = (V, E)$ there exists a sequence of permutations Π, a start vertex s, and some time step $t < n$ such that $NoT(G^{\Pi}, s, t+1) \leq NoT(G^{\Pi}, s, t) < |V|$. Then at time t, there exists a non-empty independent set of token holders H'_t such that $|H'_t| \geq |N(H'_t)| \geq 1$.*

Proof. Let $k = NoT(G^{\Pi}, s, t+1) \leq NoT(G^{\Pi}, s, t+1) < |V|$. Denote $TH(G^{\Pi}, s, t)$ as H_t and let $N_t = N(H_t)$. So the set of vertices that are holders as well as neighbors of holders is $H_t \cap N_t$. Denote $H'_t = H_t - N_t$ and N'_t to be the set of neighbors of vertices in H'_t, that is $N'_t = N(H'_T)$. See Figure 7 for an illustration.

Observe that H'_t is an independent set. We claim that H'_t is not empty. If instead, every vertex in H_t is also a neighbor of a vertex in H_t, all these vertices will also be token holders in time step $t+1$. Since the graph is connected, and since $NoT(t) < |V|$, at least one more vertex will also be token holder at time $t+1$, implying that $NoT(t+1) > NoT(t)$, a contradiction. This proves the claim.

Next, consider N'_t, the set of neighbors of H'_t. Observe that N'_t is not empty, if so, H'_t would be disconnected from the rest of the graph. Also clearly $N'_t \cap H_t = \emptyset$. We claim that $|H'_t| \geq |N'_t|$. Suppose instead that $|H'_t| < |N'_t|$. Consider the token holders at time $t+1$. Every node (if any) in $H_t \cap N_t$ will be a token holder at time $t+1$. Also all nodes in N'_t will be token holders at time $t+1$. Therefore

$$|H_{t+1}| \geq |H_t \cap N_t| + |N'_t| > |H_t \cap N_t| + |H'_t| = |H_t|$$

a contradiction to $NoT(G^{\Pi}, s, t+1) \leq NoT(G^{\Pi}, s, t)$.

Therefore $|H'_t| \geq |N'_t| \geq 1$ as claimed. ◀

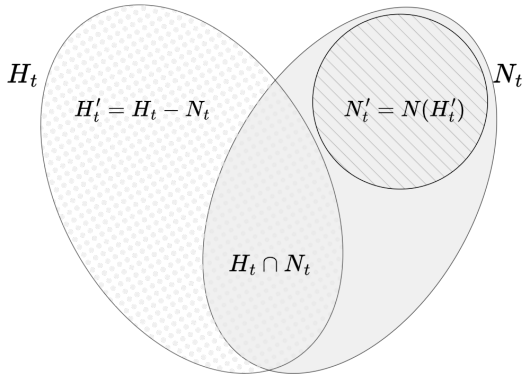

Figure 7 Token holders and their neighbors at time step t. Here, N_t and N'_t denote respectively the set of all neighbors of the vertices in H_t and H'_t.

In the token dissemination procedure for any VPG based on $G = (V, E)$, unless there is a blocking set in G , the number of vertices that have received a token *increase* in every time step until the number of token holders is $|V|$. Lemma 7 formalizes this observation.

▶ **Lemma 7.** *Suppose for a connected graph $G = (V, E)$ there exists a sequence of permutations Π, a start vertex s, and some time step $t < n$ such that $NoT(G^\Pi, s, t+1) \leq NoT(G^\Pi, s, t) < |V|$. Then G has a blocking set S. Moreover $|S| \leq NoT(G^\Pi, s, t+1)$.*

Proof. Follows from Lemmas 5 and 6. ◀

▶ **Lemma 8.** *If a connected graph $G = (V, E)$ doesn't have a* BS*, then it is fully tokenizable.*

Proof. From Lemma 7, we can deduce that if the graph $G(V, E)$ does not have a blocking set, then the number of token holders keeps increasing in every time step until it equals $|V|$. So G is fully tokenizable. ◀

An important observation in the token dissemination procedure for a graph $G = (V, E)$ with a *MBS* S is as follows: the number of token holders continues to increase in every step until it reaches $|S|$; after this time step, the number of token holders will remain at least $|S|$ in all subsequent time steps. This behavior is formalized and proven in the following lemma.

▶ **Lemma 9.** *Suppose a connected graph $G = (V, E)$ has a* MBS *S of size k, then for every sequence of permutations Π, and start vertex s and time step t, if $NoT(G^\Pi, s, t) < k$ then $NoT(G^\Pi, s, t + 1) > NoT(G^\Pi, s, t)$. Moreover, if for some time step t $NoT(G^\Pi, s, t) \geq k$, then $NoT(G^\Pi, s, t + 1) \geq k$.*

Proof. For the first part, suppose as a contradictory assumption that there exists some Π, s, and t such that, $NoT(G^\Pi, s, t + 1) \leq NoT(G^\Pi, s, t) < k$, then from Lemma 7, there must exist a blocking set S' such that $|S'| \leq NoT(G^\Pi, s, t + 1) \leq NoT(G^\Pi, s, t) < k$, which contradicts the assumption that the size of the minimum blocking set is k. Therefore, for every π, s, t, if $NoT(G^\Pi, s, t) < k$, it follows that $NoT(G^\Pi, s, t + 1) > NoT(G^\Pi, s, t)$.

For the second part, suppose $NoT(G^\Pi, s, t) \geq k$ and $NoT(G^\Pi, s, t+1) < k$. Once again, by Lemma 7, this would imply the existence of an *MBS* S' such that $|S'| \leq NoT(G^\Pi, s, t+1) < k$, which contradicts the assumption that the size of the *MBS* is k. ◀

Building on the lemmas and observations established so far, we now present a characterization of untokenizable, partially tokenizable, and fully tokenizable graphs. The following theorems precisely describe the conditions under which a graph is untokenizable or partially tokenizable based on the properties of its *MBS*.

▶ **Theorem 10.** *A connected graph* G *with* n *vertices is untokenizable if and only if it contains a blocking set* S *such that* $|S| < \frac{n}{2}$.

Proof. First we prove that if there is a *MBS* S such that $|S| < \frac{n}{2}$, then $G(V, E)$ is untokenizable. Let $N = N(S)$, and let $U = V - S - N$. We describe a sequence of permutations Π and a start vertex such that vertices in $U \subset V$ will never hold a token in G^{Π}. Since S is a *MBS* of G and $|S| = k$, we have $|N| = k$ and $|U| = n - 2k$. Wlog let $S = [0, k-1]$, $N = [k, 2k-1]$, and $U = [2k, n-1]$. Our temporal graph will be a 2-periodic graph such that in the odd time steps, we have the original graph G and in the even steps we use the following permutation:

$$\Pi(i) = \begin{cases} i + k \mod 2k & \text{if } i < 2k \\ i & \text{if } i \geq 2k \end{cases}$$

With this permutation Π, observe that the subgraph induced by U is not connected to S in odd steps, and is not connected to vertices in N in even steps. To prevent U from ever obtaining a token, it is sufficient to start the token procedure at some vertex s in S. Then in time step 2, a subset of the vertices in N will hold the token, and none of the vertices in U have the token. Since the subgraph induced by U is not connected to any vertex in N in time step 2, there is no way for the vertices in U to become token holders in the following step. Inductively, it is straightforward to see that $TH(G^{\Pi}, s, t) \subseteq S$ for odd t and $TH(G^{\Pi}, s, t) \subseteq N$ for even t. This ensures that none of the vertices in U will ever hold a token, making the graph G untokenizable.

Next we prove the other direction. Suppose that $G(V, E)$ with n vertices is untokenizable. This means there exists an initial vertex s and a sequence of permutations Π such that $HS(G^{\Pi}, s, n) \neq V$. This implies that the maximum number of token holders at any step must be some $k < n$, and hence, there must exist some time step $t < n$ in which $NoT(G^{\Pi}, s, t) = k$ and $NoT(G^{\Pi}, s, t+1) \leq NoT(G^{\Pi}, s, t)$. By Lemma 6, at time t, there there exists a non-empty independent set of token holders H'_t such that $|H'_t| \geq |N(H'_t)| \geq 1$.

Furthermore, $|H'_t| + |N(H'_t)| < n$ since G is untokenizable, and therefore $|N(H'_t)| < n/2$. By Lemma 5, G has a blocking set of size $\leq |N(H'_t)| < n/2$ as desired. ◀

▶ **Theorem 11.** *A connected graph* G *with* n *vertices is partially tokenizable if and only if it contains a* MBS S *such that* $|S| = \frac{n}{2}$.

Proof. First, we will prove that if there exists a *MBS* S with $|S| = \frac{n}{2}$, then the graph is partially tokenizable. Since $|S| = \frac{n}{2}$, the graph is tokenizable by Theorem 10.

Now, let S be the *MBS* , wlog let $S = [0, n/2-1]$. Our temporal graph will be a 2-periodic graph; in odd steps, we have the original graph G, and in even steps, we use the permutation $\pi(i) = (i + n/2) \bmod n$. Let $NoT(i)$ be the number of tokens in step i. By Lemma 9, $NoT(G^{\Pi}, s, t)$ keeps increasing until for some time step t we have $NoT(G^{\Pi}, s, t) = |S|$. After this step, all vertices in the set $[0, n/2-1]$ are holders in odd steps, and all remaining vertices are holders in even steps and $NoT(G^{\Pi}, s, n) = \frac{n}{2}$. This proves that G is partially tokenizable.

Conversely, suppose for graph G, there does not exist an *MBS* S with $|S| = \frac{n}{2}$. Then either G does not have a blocking set at all, or it has an *MBS* of size $< \frac{n}{2}$ (since any blocking set has to be of size $\leq \frac{n}{2}$). In the first case, by Theorem 8, G is fully tokenizable, and in the second case, by Theorem 10, it is untokenizable. ◀

▶ **Theorem 12.** *A connected graph G with n vertices is fully tokenizable if and only if does not contain any* BS.

Proof. If G does not contain any *BS*, then by Lemma 8, G is fully tokenizable. For the converse, assume for the purpose of contradiction that G is fully tokenizable, and that G contains a *BS*. Let S be a *MBS* of G. Then either $|S| = \frac{n}{2}$ or $|S| < \frac{n}{2}$.
- If $|S| = \frac{n}{2}$, then by Theorem 11, G is partially tokenizable.
- If $|S| < \frac{n}{2}$, then by Theorem 10, G is untokenizable.

In both cases, G is not fully tokenizable, a contradiction to the assumption. Therefore, G does not contain a *BS*. ◀

3.2 Restless token dissemination

In this section, we present a straightforward consequence of our characterization of tokenizable graphs.

▶ **Theorem 13.** *There is a distributed algorithm for restless token dissemination in every vertex permuted graph G^{Π} based on a connected graph $G = (V, E)$ in $|V|$ steps, if and only if G has no* MBS *of size $< |V|/2$.*

Proof. Suppose G has no *MBS* of size $< |V|/2$. Then Theorems 12 and 11 imply that under any permutation sequence Π, and from any start vertex, the token dissemination procedure will ensure that all vertices in G^{Π} become token holders in $|V|$ steps, ensuring restless token dissemination in G^{Π}. Conversely, suppose there is a distributed algorithm for token dissemination in every vertex-permuted graph G^{Π} based on G. Fix a vertex-permuted graph G^{Π} based on G, and suppose there exists a schedule for restless token dissemination in G^{Π}. It must be a subset of the schedule implied by our token dissemination procedure, in which every vertex sends the token to all its neighbors in the step immediately after it receives it. Therefore our token dissemination procedure will also ensure that tokens are restlessly disseminated to all vertices in G^{Π}. Since this is true for every G^{Π} based on G, it follows that G is tokenizable, and by Theorem 10, G has no *MBS* of size $< |V|/2$. ◀

We note that if G has an *MBS* of size exactly $|V|/2$, there are VPGs based on G, in which the token will never be held simultaneously by all nodes in the restless setting, whereas if G has no blocking set, then for every VPG G^{Π} based on G, the token dissemination procedure described in Section 3.1 results in every node simultaneously holding the token.

3.3 Restless exploration

Next, we outline the relationship between tokenizability of a graph G and explorability of vertex-permuted graphs based on G.

▶ **Theorem 14.** *A connected graph $G = (V, E)$ is tokenizable if and only if every temporal graph G^{Π} with base graph G and lifetime $L \geq n(n-1)$ is restless explorable, where $n = |V|$.*

Proof. Suppose G is tokenizable. Let G^{Π} be an arbitrary VPG based on the graph G, and of lifetime $L \geq n(n-1)$. By the definition of tokenizability, for every $s \in V$, $HS(G^{\Pi}, s, n) = V$. That is, for any vertex u, we have $u \in HS(G^{\Pi}, s, n)$. This implies that there exists a restless temporal walk from s to u in at most n time steps. Using the standard technique of concatenating temporal walks between pairs of vertices into an exploration schedule, it follows that there is a restless exploration schedule of length at most $n(n-1)$ in G^{Π} starting at any vertex s.

Conversely, suppose G is untokenizable. Then by Theorem 10, G has an MBS S of size $|S| < n/2$. Similarly to the proof of Theorem 10, we define a 2-periodic VPG (of infinite lifetime) based on G, in which $G_i = G$ for odd i, and in even time steps, the vertices in S change places with the vertices in $N(S)$. Let v be an arbitrary vertex in S and let u be an arbitrary vertex in $V - \{S \cup N(S)\}$. Since $|S| = |N(S)| < n/2$, there exists at least one such u. It is clear that there is no temporal walk (of any length) from v to u in this graph. Therefore G^Π is unexplorable. ◀

We note that the exploration schedule mentioned above can be constructed in an offline and centralized manner, and is not necessarily an optimal schedule; we only claim that a schedule of length at most $n(n-1)$ exists.

4 Restless explorability of some graph families

In this section, we study the explorability of some specific families of graphs. Observe that cliques are clearly restlessly explorable: a vertex-permuted clique is actually a static graph, since in every step, the graph is the same: every vertex is connected to every other vertex.

▶ **Theorem 15.** *Every VPG whose base graph is a connected regular graph is restlessly explorable.*

Proof. Assume for the purpose of contradiction that there exists an unexplorable VPG G^Π with base graph a δ-regular graph G. By Theorem 14 and 10, we conclude that G contains a *MBS* S such that $|S| < \frac{|V|}{2}$. Note that $|S| = |N(S)|$ and since $|S \cup N(S)| = |S| + |N(S)| < 2\frac{|V|}{2} = |V|$, there is at least one vertex, say u, in $V - (S \cup N(S))$.

Let E' be the set of edges incident on S. Since G is δ-regular, we have $|E'| = \delta|S| = \delta|N(S)|$. Since the edges in E' are also incident on $N(S)$, we conclude that the only neighbors of $N(S)$ are vertices in S. However, this means the subgraph induced by $S \cup N(S)$ is not connected to u, a contradiction to G being connected. Thus, G^Π must be explorable. ◀

The following corollary is immediate.

▶ **Corollary 16.** *Every VPG whose base graph is a cycle, d-dimensional torus, or a connected regular bipartite graph, is restlessly explorable.*

We note that cycles and tori with odd numbers of vertices are fully tokenizable, while those with an even number of vertices are partially tokenizable.

Next, we consider some base graphs that are not regular.

▶ **Theorem 17.** *Let G^Π be a vertex-permuted grid with n vertices where n is even. Then G^Π is restlessly explorable.*

Proof. Assume without loss of generality that the base grid G has m columns where m is even. We show that G has no blocking set with $< n/2$ vertices by showing that $|N(S)| > |S|$ for any independent set S with $|S| < n/2$. In particular, we show that there is a matching M of size $|S|$ between S and $N(S)$ and show that there is at least one element of $N(S)$ that is unmatched.

Consider an independent set S with $|S| < n/2$, and denote the subset of elements of S in the i^{th} row by S_i. Clearly, since S_i is an independent set and the row has an even number of vertices, there is always a matching between S_i and the elements of $N(S)$ in row i. Since $|S| < n/2$, there must be a row i such that $|S_i| < m/2$. First, suppose there exists a row that has *no* elements from S at all; in this case, we take a row S_i such that $|S_i| = 0$, and it is adjacent to a row S_j with $|S_j| \neq 0$. Such a row S_i must exist, and clearly, there is at least

one neighbor of a vertex in S_j in the row i that is not matched, and we are done. Therefore in what remains, we assume that *every* row has at least one element of S, and we consider a row S_i such that $0 < |S_i| < m/2$.

Let s_i denote the 0-1 string representing the vertices in row i, where the j-th symbol is 1 if the j-th vertex is in S and 0 otherwise. Note that the number of 0's in s_i must be at least two more than the number of 1's, since $|S_i| \leq m/2 - 1$. We first consider the following three cases:

s_i has a prefix in $(01)^+00$: in this case, for each pair (01) in the prefix, we match the corresponding vertices to each other, and find an arbitrary matching in the corresponding suffix (which must have at least as many 0s as 1s). Let u be the vertex corresponding to the second-last symbol of the prefix. Then $u \in N(S)$ and is unmatched.

s_i has a suffix in $00(10)^+$: We match each pair as in the previous case, and let u be the vertex corresponding to the second symbol in the suffix; we note that $u \in N(S)$ is unmatched.

s_i has a substring in $00(10)^+0$: The argument is identical to the previous case; the vertex u corresponding to the second symbol in the substring is in $N(S)$ and is unmatched.

Thus in all the above cases, we have the required matching and unmatched element of $N(S)$ as needed. It remains to consider the situation when for every row S_i with $|S_i| < m/2$, none of the three conditions listed above is satisfied. Let S_i be such a row and s_i the corresponding string. Since s_i does not have a prefix in $(01)^+00$, it either starts with 00 or starts with 1. If it starts with 1, it must be either of the form $(10)^+(00)^+$ (Type A) or of the form $(10)^+(00)^+(01)^+$ (Type B), since it does not satisfy the second and third conditions. If instead it starts with 00, it must be of the form $(00)^+(01)^+$ (Type C). For any of these types, we match the vertices corresponding to each (01) pair. We now identify an unmatched vertex u for each type. Let S_j be a row adjacent to S_i.

Type A. $s_i \in (10)^+(00)^+(01)^+$: If $|S_j| = m/2$ or if $|S_j|$ is of types A, B, or C, then s_j must end with or start with a 1, which implies that two elements of S are adjacent, a contradiction.

Type B. $s_i \in (10)^+(00)^+$: Then as in the previous case, it cannot be that S_j is of Type A or B. So S_j must be of type C, or $|S_j| = m/2$ and $s_j \in (01)^+$. Let u be the last vertex in row i; it is adjacent to the last vertex in row j which is in S; note that u is unmatched.

Type C. $s_i \in (00)^+(01)^+$: Symmetric to the previous case, S_j must be of type B, or $|S_j| = m/2$ and $s_j \in (10)^+$. Then let u be the first vertex in row i; it is adjacent to the first vertex in row j, which is in S; note that u is unmatched.

The above analysis is exhaustive: we showed that if $|S| < m/2$, there exists a matching of size $|S|$ between S and $N(S)$ and a vertex $u \in N(S)$ that is unmatched. That is, $|N(S)| > |S|$, which implies that S is not a blocking set. Therefore, by Theorems 10 and 14, G^Π is restlessly explorable. ◀

Finally, we present some examples of untokenizable graphs. Recall that for an untokenizable graph G, there may be some VPGs with base graph G that are explorable, and others that are not explorable; see for example Figure 8 part (b).

▶ **Theorem 18.** *Any connected graph $G = (V, E)$ with $|V| \geq 3$ that contains a vertex of degree 1 is untokenizable. Therefore, there exist VPGs based on G that are not restlessly explorable.*

Proof. Let $v \in V$ be the vertex with degree 1, and let u be its only neighbor. Since v has no other connections, it forms an independent set of size 1, with $N(v) = \{u\}$. This implies that v constitutes a *MBS* of size 1, satisfying $|v| < \frac{|V|}{2}$. By Theorem 10, G is untokenizable, and by Theorem 14, there exist VPGs based on G that are not restlessly explorable. ◀

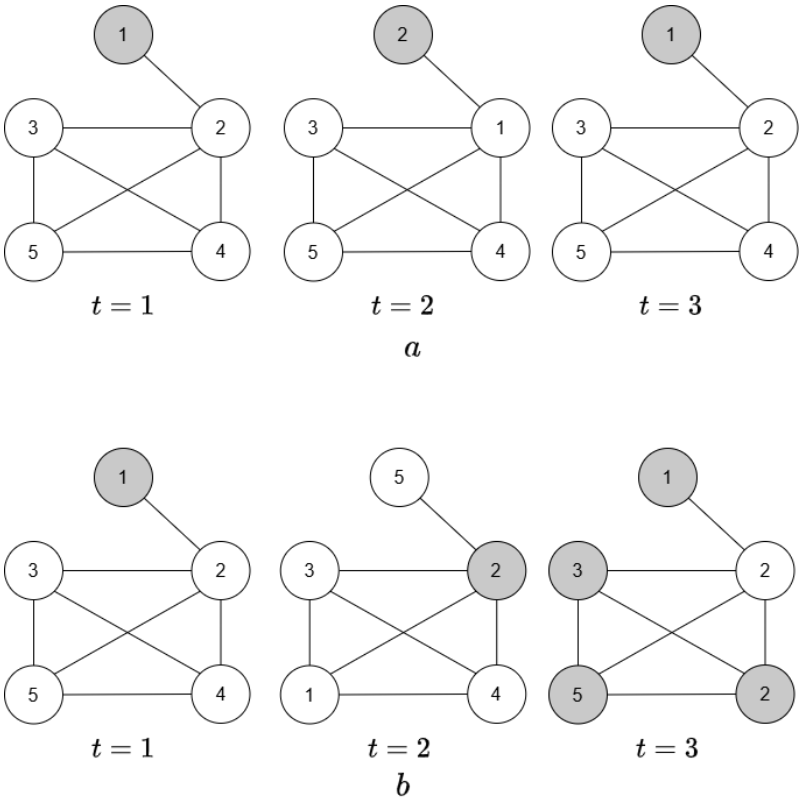

Figure 8 Illustration of 2 vertex-permuted graphs with the same base graph G with a vertex of degree 1. In the VPG shown in part (a), the token alternates between vertex 1 and its only neighbor 2, leaving the other vertices unexplored. In contrast, part (b) demonstrates a VPG where all vertices are visited by the token within 3 time steps.

The following corollary is an immediate consequence of Theorem 18 and 14.

▶ **Corollary 19.** *There exist VPGs with base graph a path or a star-graph that are not restlessly explorable.*

Figure 8 illustrates two VPGs with the same base graph G. Note that by Theorem 18, the base graph G is untokenizable. In part (a), the first three time steps of a 2-periodic VPG based on G are shown; in odd and even time steps the vertices 1 and 2 swap places. Consequently, $TH(G^{\Pi}, 1, i) = \{1\}$ for odd i, and $TH(G^{\Pi}, 1, i) = \{2\}$ for even i. For all $t \geq 2$, $HS(G^{\Pi}, 1, t) = \{1, 2\} \neq V$, and it is clear that restless token dissemination or exploration are not possible in this VPG. In contrast, in part (b), a different 2-periodic VPG based on the same base graph G is shown, in which the token reaches all vertices in 3 time steps.

▶ **Theorem 20.** *For every bipartite graph $G = (M, N, E)$ such that $|M| \neq |N|$, there exist VPGs based on G that are not restlessly explorable.*

Proof. Without loss of generality, assume $|M| > |N|$. By Lemma 5, there exists a blocking set $S \subseteq M$ such that $|S| < |N|$. Since $|M| > |N|$, it follows that $|S| < \frac{|V|}{2}$. By Theorem 10, G is untokenizable, and by Theorem 14, there exist VPGs based on G that are not restlessly explorable. ◀

As a direct consequence of Theorem 20, we have the following:

▶ **Corollary 21.** *For every grid G with an odd number of vertices, there exists VPGs based on G that are not restlessly explorable.*

5 Open Problems

In this paper, we studied restless exploration in vertex-permuted graphs, and showed that unless a graph $G = (V, E)$ has a blocking set of size $< |V|/2$, all vertex-permuted graphs with base graph G are explorable. The complexity of determining whether such a blocking set exists in a given graph G remains an open computational problem. While a graph may be untokenizable, it is possible that a specific VPG G^{Π} is still 1-restlessly explorable. The complexity of determining if a specific VPG is 1-restlessly explorable is an interesting open question. Extending our results to the k-restless setting is another direction of research.

References

1 Sebastian Abshoff, Markus Benter, Andreas Cord-Landwehr, Manuel Malatyali, and Friedhelm Meyer auf der Heide. Token dissemination in geometric dynamic networks. In *International Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics*, pages 22–34. Springer, 2013. doi:10.1007/978-3-642-45346-5_3.

2 Ankush Agarwalla, John Augustine, William K Moses Jr, Sankar K Madhav, and Arvind Krishna Sridhar. Deterministic dispersion of mobile robots in dynamic rings. In *Proceedings of the 19th International Conference on Distributed Computing and Networking*, pages 1–4, 2018. doi:10.1145/3154273.3154294.

3 Mohamad Ahmadi, Fabian Kuhn, Shay Kutten, Anisur Rahaman Molla, and Gopal Pandurangan. The communication cost of information spreading in dynamic networks. In *IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pages 368–378, 2019. doi:10.1109/ICDCS.2019.00044.

4 Amotz Bar-Noy, Prabhakar Raghavan, Baruch Schieber, and Hisao Tamaki. Fast deflection routing for packets and worms. In *Proceedings of the Twelfth Annual ACM Symposium on Principles of Distributed Computing*, pages 75–86, 1993.

5 Thomas Bellitto, Cyril Conchon-Kerjan, and Bruno Escoffier. Restless exploration of periodic temporal graphs. In *2nd Symposium on Algorithmic Foundations of Dynamic Networks (SAND 2023)*, pages 13:1–13:15, 2023. doi:10.4230/LIPICS.SAND.2023.13.

6 Costas Busch, Maurice Herlihy, and Rogert Wattenhofer. Routing without flow control. In *Proceedings of the Thirteenth Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 11–20, 2001. doi:10.1145/378580.378582.

7 Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, and Nicola Santoro. Time-varying graphs and dynamic networks. *International Journal of Parallel, Emergent and Distributed Systems*, pages 387–408, 2012. doi:10.1080/17445760.2012.668546.

8 Arnaud Casteigts, Anne-Sophie Himmel, Hendrik Molter, and Philipp Zschoche. Finding temporal paths under waiting time constraints. *Algorithmica*, pages 2754–2802, 2021. doi:10.1007/S00453-021-00831-W.

9 Konstantinos Dogeas, Thomas Erlebach, Frank Kammer, Johannes Meintrup, and William K Moses Jr. Exploiting automorphisms of temporal graphs for fast exploration and rendezvous. In *51st International Colloquium on Automata, Languages, and Programming, ICALP*, pages 55:1–55:18, 2024. doi:10.4230/LIPICS.ICALP.2024.55.

10 Chinmoy Dutta, Gopal Pandurangan, Rajmohan Rajaraman, Zhifeng Sun, and Emanuele Viola. On the complexity of information spreading in dynamic networks. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 717–736, 2013. doi:10.1137/1.9781611973105.52.

11 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.

12 Thomas Erlebach and Jakob T Spooner. Non-strict temporal exploration. In *International Colloquium on Structural Information and Communication Complexity*, pages 129–145, 2020. doi:10.1007/978-3-030-54921-3_8.

13 Thomas Erlebach and Jakob T Spooner. Parameterised temporal exploration problems. *Journal of Computer and System Sciences*, 135:73–88, 2023. doi:10.1016/J.JCSS.2023.01.003.

14 U. Feige and P. Raghavan. Exact analysis of hot-potato routing. In *Proceedings, 33rd Annual Symposium on Foundations of Computer Science*, pages 553–562, 1992.

15 Afonso Ferreira. *Building a Reference Combinatorial Modelfor Dynamic Networks: Initial Results in Evolving Graphs*. PhD thesis, INRIA, 2003.

16 Paola Flocchini, Bernard Mans, and Nicola Santoro. Exploration of periodically varying graphs. In *Algorithms and Computation: 20th International Symposium, ISAAC 2009*, pages 534–543, 2009. doi:10.1007/978-3-642-10631-6_55.

17 Eugen Füchsle, Hendrik Molter, Rolf Niedermeier, and Malte Renken. Delay robust routes in temporal graphs. In *39th International Symposium on Theoretical Aspects of Computer Science (STACS 2022)*, pages 30:1–30:15, 2022. doi:10.4230/LIPICS.STACS.2022.30.

18 Bernhard Haeupler. Analyzing network coding gossip made easy. In *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing*, pages 293–302, 2011. doi:10.1145/1993636.1993676.

19 David Ilcinkas, Ralf Klasing, and Ahmed Mouhamadou Wade. Exploration of constantly connected dynamic graphs based on cactuses. In *International Colloquium on Structural Information and Communication Complexity*, pages 250–262. Springer, 2014. doi:10.1007/978-3-319-09620-9_20.

20 David Kempe, Jon Kleinberg, and Amit Kumar. Connectivity and inference problems for temporal networks. *Journal of Computer and System Sciences*, pages 820–842, 2002. doi:10.1006/JCSS.2002.1829.

21 Fabian Kuhn, Nancy Lynch, and Rotem Oshman. Distributed computation in dynamic networks. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, pages 513–522, 2010. doi:10.1145/1806689.1806760.

22 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.

23 Thomas Moscibroda and Onur Mutlu. A case for bufferless routing in on-chip networks. In *Proceedings of the 36th Annual International Symposium on Computer Architecture*, pages 196–207, 2009. doi:10.1145/1555754.1555781.

24 Regina O'Dell and Rogert Wattenhofer. Information dissemination in highly dynamic graphs. In *Proceedings of the 2005 Joint Workshop on Foundations of Mobile Computing*, pages 104–110, 2005. doi:10.1145/1080810.1080828.

25 Christian Scheideler. Models and techniques for communication in dynamic networks. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 27–49, 2002. doi:10.1007/3-540-45841-7_2.

26 Suhas Thejaswi, Juho Lauri, and Aristides Gionis. Restless reachability problems in temporal graphs. *Knowledge and Information Systems*, pages 1–47, 2025.

27 B Bui Xuan, Afonso Ferreira, and Aubin Jarry. Computing shortest, fastest, and foremost journeys in dynamic networks. *International Journal of Foundations of Computer Science*, pages 267–285, 2003.

28 Philipp Zschoche. Restless Temporal Path Parameterized Above Lower Bounds. In *40th International Symposium on Theoretical Aspects of Computer Science (STACS 2023)*, pages 55:1–55:16, 2025.