Discrete Mathematics Letters
www.dmlett.com

Discrete Math. Lett. **18** (2026) 77–84
DOI: 10.47443/dml.2026.159

Research Article

A Note on Directed Clique Token Sliding

Duc A. Hoang*

VNU University of Science, Vietnam National University, Hanoi, Vietnam

(Received: 30 May 2026. Received in revised form: 18 August 2026. Accepted: 28 August 2026. Published online: 3 September 2026.)

Abstract

A *directed clique token sliding (TS) step* in an *oriented graph* D replaces a vertex u of a k-clique of the underlying graph $U(D)$ by an out-neighbor v of u, while requiring the new set to remain a k-clique of $U(D)$. The reachability problem generated by this directed move is studied. As far as the author knows, the combination of clique feasibility with direction-constrained TS moves has not been studied before. It is shown that the problem is PSPACE-complete on oriented graphs, NP-complete on *directed acyclic graphs* (DAGs), and W[1]-hard on DAGs when parameterized by k. The NP-completeness of the bounded-length variant on DAGs is also proved. On the positive side, an explicit polynomial-time reachability criterion for tournaments is derived by specializing known structural results on token digraphs, and polynomial-time algorithms are given for transitive oriented graphs, DAGs of any fixed depth, and oriented graphs D for which $\omega(U(D))$ is bounded by a fixed constant.

Keywords: combinatorial reconfiguration; clique reconfiguration; token sliding; oriented graph; directed acyclic graph.

2020 Mathematics Subject Classification: 05C57, 05C85, 68Q17.

1. Introduction

Recently, *combinatorial reconfiguration* has attracted attention from both theoretical and practical viewpoints. Given two feasible solutions of the same instance, the task is to decide whether one can transform the first solution into the second by local moves while maintaining feasibility. For an overview, we refer readers to the surveys by van den Heuvel [13], Nishimura [12], and Bousquet et al. [2].

As a simple motivating case, consider a *tournament*, an orientation of a complete graph. Every k-subset is a clique in the underlying graph, so the only restriction on a token slide is the direction of the chosen arc. Thus the state space is directed even in the most clique-rich setting, making token sliding (TS) the natural rule for a directed version of CLIQUE RECONFIGURATION.

We now define the reachability problem studied in this paper. An *oriented graph* is a directed graph obtained by orienting the edges of a simple graph. For an oriented graph D, we write $A(D)$ for its *arc set* and $U(D)$ for its *underlying graph*, that is, the graph on $V(D)$ in which uv is an edge if and only if $(u,v) \in A(D)$ or $(v,u) \in A(D)$. A set $C \subseteq V(D)$ is a *k-clique* of $U(D)$ if $|C| = k$ and every two distinct vertices of C are adjacent in $U(D)$.

Definition 1.1. *Let D be an oriented graph, let $k \geq 1$, and let C and C' be two k-cliques of $U(D)$. We say that C' is obtained from C by one* directed clique token sliding step *if there exist $u \in C$ and $v \notin C$ such that $C' = C - u + v$, $(u,v) \in A(D)$, and C' is a k-clique of $U(D)$.*

The definition keeps the feasibility condition undirected and uses directions only in the move relation. A sequence $\langle C_0, C_1, \ldots, C_q \rangle$ is a *directed clique token sliding sequence* if C_{i+1} is obtained from C_i by one directed clique token sliding step for every $i \in \{0, \ldots, q-1\}$. Its length is q, the number of slides. We allow $q = 0$, so every state reaches itself. When the endpoints matter, we say that the sequence is from C_0 to C_q.

Definition 1.2. *Let D be an oriented graph and let $k \geq 1$. The* directed clique TS graph *of (D,k), denoted by $\overrightarrow{\mathcal{C}}_k(D)$, is the directed graph whose vertices are the k-cliques of $U(D)$ and whose arcs are the directed clique token sliding steps.*

We call the following reachability problem the DIRECTED CLIQUE TOKEN SLIDING *problem*.

DIRECTED CLIQUE TOKEN SLIDING **Input:** An oriented graph D, an integer $k \geq 1$, and two k-cliques C_s and C_t of $U(D)$. **Question:** Is there a directed path from C_s to C_t in $\overrightarrow{\mathcal{C}}_k(D)$?

*E-mail address: hoanganhduc@hus.edu.vn

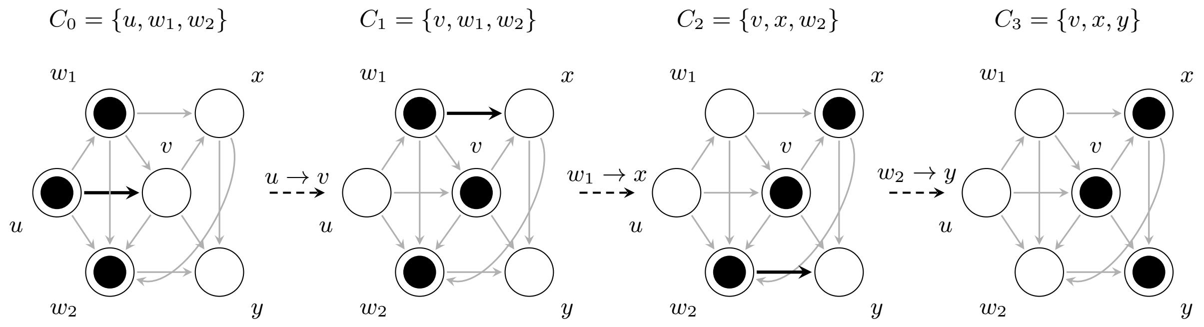

Figure 1.1: A three-step directed clique token sliding sequence. Each panel shows the same oriented graph, all edges are directed, and the filled disks mark the current clique. The arrows between panels show the three single-token slides.

The *bounded-length variant* additionally takes a nonnegative integer L, encoded in binary, and asks whether such a directed clique token sliding sequence can be chosen with length at most L.

We focus on TS because it is the only standard clique-reconfiguration rule with a canonical directed interpretation. Directed analogues of *token addition and removal* (TAR) or *token jumping* (TJ) would require extra conventions about how direction constrains an addition, a deletion, or an unrestricted jump.

Undirected Clique Reconfiguration was studied by Ito, Ono, and Otachi [10]. They proved that TAR, TJ, and TS are equivalent from the viewpoint of polynomial-time solvability; in particular, the problem is PSPACE-complete even for perfect graphs, while several graph classes admit polynomial-time algorithms.

Directed token sliding has also been studied for independent sets [1, 5, 9]. That model is closest to ours at the move level, but it does not imply the clique results below. At the level of feasible sets, a clique of $U(D)$ is an independent set of $\overline{U(D)}$, and its complement is a vertex cover of $\overline{U(D)}$. However, TS also constrains the moved pair: a clique slide uses an edge of $U(D)$, whereas the corresponding independent-set or vertex-cover slide in the complement would need an edge of $\overline{U(D)}$. In the directed setting, the moved pair must also follow an arc, so complement dualities for independent sets and vertex covers do not transfer to directed clique TS.

Token graphs and *token digraphs* study unconstrained movements of k indistinguishable tokens [6, 7]. They coincide with our state graph whenever every k-subset is a clique; in particular, this holds if $U(D)$ is complete. The special case $k = 1$ also coincides for every oriented graph.

In summary, prior work studies undirected clique reconfiguration [10], directed independent-set TS [9], or token digraphs without a clique constraint [7]. To our knowledge, none combines directed TS moves with clique feasibility.

In this paper, we initiate the study of Directed Clique Token Sliding. The problem keeps clique feasibility undirected and uses arc directions only in the move relation. Our first results are as follows.

- We prove that Directed Clique Token Sliding is PSPACE-complete on oriented graphs (Theorem 3.1), NP-complete on directed acyclic graphs (DAGs) (Theorem 3.2), and W[1]-hard on DAGs when parameterized by k (Corollary 3.3). We also prove that the bounded-length variant is NP-complete on DAGs (Corollary 3.2).
- Using the token-digraph structure established by Fernandes et al. [7], we derive an explicit polynomial-time reachability criterion for tournaments (Theorem 4.1). We give polynomial-time algorithms for DAGs of fixed depth (Proposition 4.1), oriented graphs D for which $\omega(U(D))$ is bounded by a fixed constant (Proposition 4.2), and transitive oriented graphs (Theorem 4.2).

Ito et al. [9] established the same PSPACE-, NP-, and W[1]-hardness pattern for directed independent-set TS. We consider clique-feasible configurations rather than independent sets and use reductions tailored to the clique constraint; we also establish the bounded-length result. Their oriented-graph reduction is from undirected Token Sliding, whereas ours is from H-Word Reachability; on DAGs, they reduce from Multicolored Independent Set using $k + 1$ tokens, whereas ours reduces from Clique using exactly k tokens.

The rest of this paper is organized as follows. Section 2 fixes notation and recalls the source problem used in the general hardness proof. Section 3 proves the hardness results. Section 4 gives the polynomial-time algorithms, and Section 5 concludes.

2. Preliminaries

We use $V(G)$ and $E(G)$ to denote the sets of vertices and edges of a (simple, undirected) graph G, respectively. For an oriented graph D, we use $V(D)$ and $A(D)$ to denote the sets of vertices and arcs of D, respectively. For a vertex set $X \subseteq V(G)$, we write $G[X]$ for the *subgraph of G induced by X*. The *neighborhood* of a vertex v in G, denoted by $N_G(v)$, is the set $\{w \in V(G) : vw \in E(G)\}$. The *closed neighborhood* of v in G, denoted by $N_G[v]$, is simply the set $N_G(v) \cup \{v\}$. Standard graph-theoretic notation not defined in this paper follows the terminology of Diestel [4].

For a positive integer r, we write $[r] = \{1, \dots, r\}$. For two sets A, B, we sometimes use $A - B$ and $A + B$ to indicate $A \setminus B$ and $A \cup B$, respectively. We also use $A - x$ and $A + x$ as shorthand for $A - \{x\}$ and $A + \{x\}$, respectively.

An *oriented graph* is a directed graph obtained by orienting the edges of a simple graph. In particular, it has no loops and no pair of opposite arcs. A *directed acyclic graph* (DAG) is a directed graph with no directed cycle. A *topological ordering* of a DAG D is an ordering $v_1, \dots, v_n$ of $V(D)$ such that $(v_i, v_j) \in A(D)$ implies $i < j$. The *depth* of a DAG is the number of vertices in a longest directed path. A *tournament* is an oriented graph whose underlying graph is complete. A *strongly connected component* (SCC) of a directed graph is a maximal vertex set in which every ordered pair of vertices is mutually reachable by directed paths. The *condensation* of a directed graph is the DAG obtained by contracting each SCC to one vertex.

For an oriented graph D, the *underlying graph* of D, denoted by $U(D)$, is the graph on $V(D)$ in which $uv \in E(U(D))$ if and only if $(u, v) \in A(D)$ or $(v, u) \in A(D)$. A *clique of D* means a clique of $U(D)$.

A *complete multipartite graph* is a graph whose vertex set can be partitioned into independent sets, called *parts*, such that two vertices are adjacent if and only if they lie in different parts. A *transitive oriented graph* is an oriented graph D such that $(u, v) \in A(D)$ and $(v, w) \in A(D)$ imply $(u, w) \in A(D)$ for every triple $u, v, w \in V(D)$.

Let $k \geq 1$ be an integer. A *k-clique* of a graph G is a set $C \subseteq V(G)$ with $|C| = k$ such that $G[C]$ is complete. We use $\omega(G)$ to denote the *clique number* of G.

The *k-token digraph* of a directed graph D, denoted by $F_k(D)$, is the directed graph whose vertices are the k-subsets $S \subseteq V(D)$ and whose arcs are the ordered pairs $(S, S - u + v)$ such that $u \in S$, $v \notin S$, and $(u, v) \in A(D)$.

We also recall the source problem used in the general hardness proof. Let $H = (\Sigma, R)$ be a fixed *possibly looped digraph*, equivalently a fixed binary relation $R \subseteq \Sigma^2$. A word $w = w_1 w_2 \cdots w_n$ over Σ is an *H-word* if $(w_i, w_{i+1}) \in R$ for every $i \in \{1, \dots, n-1\}$. The H-Word Reachability problem, also called H-Word Reconfiguration in the literature, asks whether two given H-words of the same length can be transformed into one another by changing one symbol at a time while keeping every intermediate word an H-word. Wrochna [14] proved that there is a fixed possibly looped digraph H for which H-Word Reachability is PSPACE-complete.

3. Hardness Results

We begin with a local structural observation.

Lemma 3.1. *Let D be an oriented graph, let $k \geq 1$, and let C and C' be two k-cliques of $U(D)$. If C' is obtained from C by one directed clique token sliding step, then $C \cup C'$ is a $(k+1)$-clique of $U(D)$.*

Proof. By definition, $C' = C - u + v$ for some $u \in C$ and $v \notin C$ with $(u, v) \in A(D)$. Since C is a clique, the vertex u is adjacent to every vertex of $C - \{u\}$. Since C' is a clique, the vertex v is adjacent to every vertex of $C - \{u\}$. Finally, $(u, v) \in A(D)$ implies that u and v are adjacent in $U(D)$. Therefore every pair of vertices in $C \cup C'$ is adjacent in $U(D)$. Moreover, $C \cup C' = C + v$, so $|C \cup C'| = k + 1$. □

Corollary 3.1. *Let D be an oriented graph. If $k = \omega(U(D))$, then every k-clique of $U(D)$ is isolated in $\overrightarrow{\mathcal{C}}_k(D)$.*

Proof. If a k-clique had an outgoing or incoming TS step, then Lemma 3.1 would yield a $(k+1)$-clique of $U(D)$, contradicting $k = \omega(U(D))$. □

Corollary 3.1 delineates a degenerate boundary: when $k = \omega(U(D))$, deciding reachability reduces to checking whether the two endpoint cliques are identical, and the tractable cases studied below are nontrivial only for $k < \omega(U(D))$.

Proposition 3.1. Directed Clique Token Sliding *is in* PSPACE.

Proof. We first verify that the input endpoints C_s and C_t are k-cliques of $U(D)$; otherwise we reject. The number of k-cliques of $U(D)$ is at most $\binom{|V(D)|}{k} \leq 2^{|V(D)|}$. Each state can be stored explicitly in polynomial space, and from a state C one can enumerate all directed TS steps in polynomial time by trying all ordered pairs (u, v) with $u \in C$ and $v \notin C$. If C_t is reachable from C_s, then there is a simple directed path from C_s to C_t in $\overrightarrow{\mathcal{C}}_k(D)$ of length at most $\binom{|V(D)|}{k} - 1$. A nondeterministic machine guesses such a path one successor at a time, storing only the current state and an $O(|V(D)|)$-bit step counter. The procedure uses polynomial space, so the problem is in NPSPACE $=$ PSPACE. □

Theorem 3.1. Directed Clique Token Sliding *is* PSPACE-*complete.*

Proof. Membership follows from Proposition 3.1. It is therefore sufficient to construct a polynomial-time reduction from H-Word Reachability for the fixed relation $H = (\Sigma, R)$ of Wrochna [14].

Let w^s and w^t be two H-words of length n. We may assume that $n \geq 1$, since zero-length instances are trivial. We construct an oriented graph D and two n-cliques C_s and C_t.

For every position $i \in \{1, \ldots, n\}$ and every symbol $a \in \Sigma$, create a *stable vertex* x_i^a. For every position i and every ordered pair of distinct symbols $a, b \in \Sigma$, create a *transition vertex* $p_i^{a,b}$. The *layer* of x_i^a and $p_i^{a,b}$ is i.

We add arcs as follows. For stable vertices in different layers, if $i < j$, then add (x_i^a, x_j^b) whenever either $j > i+1$, or $j = i+1$ and $(a, b) \in R$. No stable–stable arc is added inside one layer. For every transition vertex $p_i^{a,b}$, add the two arcs $(x_i^a, p_i^{a,b})$ and $(p_i^{a,b}, x_i^b)$. It remains to define the adjacencies between transition vertices and stable vertices in other layers. If $j < i$, then add $(x_j^c, p_i^{a,b})$ whenever either $j < i-1$, or $j = i-1$ and both (c, a) and (c, b) belong to R. If $j > i$, then add $(p_i^{a,b}, x_j^c)$ whenever either $j > i+1$, or $j = i+1$ and both (a, c) and (b, c) belong to R. There are no other arcs. From the construction, no pair of vertices receives two opposite arcs. Thus D is an oriented graph. Since H is fixed, the number of stable and transition vertices is $O(n)$ and the number of arcs is $O(n^2)$; the construction is polynomial time.

Figure 3.1 shows the local part of the construction used when the symbol in position i changes from a to b.

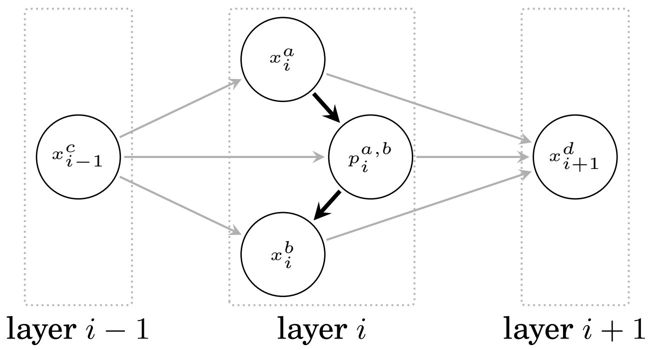

Figure 3.1: The layer gadget in the reduction from H-Word Reachability. The very thick arcs simulate changing the symbol at position i from a to b; the gray arcs indicate the adjacencies to the neighboring stable layers that preserve the H-word constraints. Arcs not relevant to the change at position i, such as $x_{i-1}^c \to x_{i+1}^d$, are omitted.

Let

$$C_s = \{x_i^{w_i^s} : i \in \{1, \ldots, n\}\} \quad \text{and} \quad C_t = \{x_i^{w_i^t} : i \in \{1, \ldots, n\}\}.$$

Since w^s and w^t are H-words, both C_s and C_t are n-cliques of $U(D)$.

($\Rightarrow$) Suppose that w^s can be transformed into w^t by changing one symbol at a time through H-words. Consider one such change, where a word w changes at position i from a to b. Let w' be the resulting word. In the clique corresponding to w, slide the token on x_i^a to $p_i^{a,b}$, and then slide it to x_i^b. The first slide is valid because $p_i^{a,b}$ is adjacent to every selected stable vertex in another layer. For the two neighboring layers, when they exist, this follows from the fact that both w and w' are H-words. For all other layers, the adjacencies were added unconditionally. The second slide is valid because the clique corresponding to w' is feasible. Repeating this two-step simulation gives a directed clique TS sequence from C_s to C_t.

($\Leftarrow$) Suppose that there is a directed clique TS sequence from C_s to C_t. For a vertex z of D, let $\lambda(z)$ be its layer, and for a clique C define

$$\Phi(C) = \sum_{z \in C} \lambda(z).$$

Every arc (z, z') of D satisfies $\lambda(z) \leq \lambda(z')$. Moreover, the inequality is strict for every arc joining two different layers. Therefore the layer potential Φ never decreases along a directed clique TS sequence, and it strictly increases whenever a token slides between two different layers. Since C_s and C_t contain exactly one vertex in each layer, they have the same value of Φ. Hence no step of the sequence from C_s to C_t uses an arc between two different layers.

It follows that every used slide stays inside one layer. Since C_s contains exactly one vertex in each layer, every state in the sequence contains exactly one vertex from each layer, where a transition vertex also counts as a vertex of its layer. By construction, no arc was added between two transition vertices, so $U(D)$ has no edge between two transition vertices. Hence no state contains two transition vertices. Within one layer, the only intra-layer arc into $p_i^{a,b}$ is $(x_i^a, p_i^{a,b})$, the only intra-layer arc out of $p_i^{a,b}$ is $(p_i^{a,b}, x_i^b)$, and there are no intra-layer stable–stable arcs. Therefore, once a state contains $p_i^{a,b}$, the next slide must be $p_i^{a,b} \to x_i^b$: any slide in another layer would either lack an intra-layer arc or introduce a second transition vertex.

Thus every transition state is part of a two-step fragment $x_i^a \to p_i^{a,b} \to x_i^b$. Whenever the first slide in such a fragment is valid, the transition vertex $p_i^{a,b}$ is adjacent to the stable vertices chosen in the neighboring layers. From the construction of these adjacencies, replacing a by b at position i keeps the word an H-word. After deleting the transition states from the clique TS sequence, every remaining state is a stable clique with one vertex in each layer. Such a stable clique encodes an H-word, because consecutive stable layers are adjacent exactly according to R. Hence we obtain a sequence of H-words from w^s to w^t in which consecutive words differ in one symbol. This completes the proof. □

Proposition 3.2. *If the input oriented graph D is a DAG, then* Directed Clique Token Sliding *belongs to* NP. *Moreover, every yes-instance admits a reconfiguration sequence of length at most $k(n-k)$, where $n = |V(D)|$.*

Proof. Let $v_1, v_2, \dots, v_n$ be a topological ordering of D, and define $\pi(v_i) = i$ for every vertex v_i. For a k-clique C, define

$$\Phi(C) = \sum_{x \in C} \pi(x).$$

Suppose that $C' = C - u + v$ is one directed TS step from C. Since $(u,v) \in A(D)$ and the ordering is topological, we have $\pi(u) < \pi(v)$. Hence $\Phi(C') > \Phi(C)$. Therefore every directed TS sequence strictly increases the potential.

Let the topological positions of the vertices of an arbitrary k-clique C be $p_1 < p_2 < \cdots < p_k$. Then $p_i \geq i$ and $p_i \leq n-k+i$ for every $i \in [k]$. Hence every k-clique C satisfies $1+2+\cdots+k \leq \Phi(C) \leq (n-k+1)+\cdots+n$. Since each step increases Φ by at least one, every directed path has length at most $((n-k+1)+\cdots+n) - (1+2+\cdots+k) = k(n-k)$. A yes-certificate is a sequence of at most $k(n-k)$ valid steps, which can be verified in polynomial time. □

Theorem 3.2. Directed Clique Token Sliding *is* NP-*complete even when the input oriented graph is a DAG.*

Proof. Membership follows from Proposition 3.2. We reduce from Clique [11]. Let (G,k) be an instance of Clique. We may assume that $1 \leq k \leq |V(G)|$; instances outside this range are trivial and can be handled separately. We construct a DAG D and two k-cliques C_s and C_t.

Let $S = \{s_1, \dots, s_k\}$ and $T = \{t_1, \dots, t_k\}$ be two new sets of vertices, and let $X = V(G)$. Fix an arbitrary ordering $x_1, \dots, x_n$ of X. The vertex set of D is $S \cup X \cup T$. The arc set is defined as follows.

- For every $1 \leq i < j \leq k$, add (s_i, s_j) and (t_i, t_j).
- For every $s_i \in S$ and every $x \in X$, add (s_i, x).
- For every $x \in X$ and every $t_i \in T$, add (x, t_i).
- For every edge $x_i x_j \in E(G)$ with $i < j$, add (x_i, x_j).

Figure 3.2 summarizes the construction. There are no arcs between S and T, and there are no arcs between two vertices of X that are nonadjacent in G.

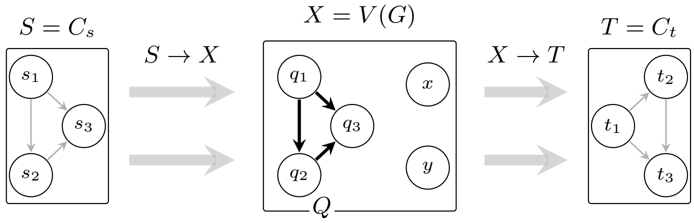

Figure 3.2: The DAG construction, shown for $k = 3$ with a possible witness triangle highlighted. The gray bundled arrows represent all arcs from S to X and all arcs from X to T; there are no adjacencies between S and T. The thick triangle represents a clique $Q = \{q_1, q_2, q_3\}$ in X when the input graph has one. For general k, the construction uses k vertices in each of S and T, and any witness clique in X has k vertices.

The ordering $s_1, \dots, s_k, x_1, \dots, x_n, t_1, \dots, t_k$ is a topological ordering of D. Thus D is a DAG. The construction has $|V(G)| + 2k$ vertices and polynomially many arcs. Let $C_s = S$ and $C_t = T$. Both sets are k-cliques of $U(D)$.

($\Rightarrow$) Suppose first that G contains a clique $Q = \{q_1, \dots, q_k\}$ of size k. Starting from S, slide the tokens one by one from vertices of S to vertices of Q. Each slide uses an arc from S to X. During this phase, the current set remains a clique because S is a clique, Q is a clique, and every vertex of S is adjacent to every vertex of X. Then slide the tokens one by one from vertices of Q to vertices of T. Each slide uses an arc from X to T. During this phase, the current set remains a clique because Q is a clique, T is a clique, and every vertex of X is adjacent to every vertex of T. Therefore (D, k, C_s, C_t) is a yes-instance of Directed Clique Token Sliding.

($\Leftarrow$) Conversely, suppose that there is a directed clique TS sequence from S to T. Since no vertex of S is adjacent to any vertex of T, no feasible state contains both a vertex of S and a vertex of T.

Consider the first step after which a vertex of T appears. Immediately before this step, there is no vertex of T. Immediately after this step, there is no vertex of S; otherwise the resulting set would contain a nonadjacent pair with one endpoint in S and one endpoint in T. The moved token cannot slide directly from S to T, since there is no arc from S to T. Therefore the state immediately before the first appearance of a vertex of T consists of exactly k vertices of X. Since this state is a clique of $U(D)$, these k vertices form a clique of G. This completes the proof. □

Corollary 3.2. *The bounded-length variant of* Directed Clique Token Sliding *is* NP*-complete even when the input oriented graph is a DAG.*

Proof. Membership in NP on DAGs follows from Proposition 3.2. For a bounded-length yes-instance, let ℓ be the length of a shortest directed clique token sliding sequence from C_s to C_t. By the definition of the bounded-length variant, $\ell \leq L$. Moreover, by Proposition 3.2, $\ell \leq k(n-k)$, where $n = |V(D)|$. Therefore $\ell \leq \min\{L, k(n-k)\} \leq k(n-k)$, so a witnessing sequence has polynomial length even when L is encoded in binary. A verifier checks that the guessed sequence has length at most L and that each step is valid.

For hardness, start with an input (G, k) of Clique, use the reduction in the proof of Theorem 3.2, and set $L = 2k$. If G has a k-clique Q, then the forward direction of the proof gives a sequence from S to T of length exactly $2k$: slide the k tokens from S to Q, and then slide the k tokens from Q to T. Conversely, any sequence from S to T of length at most $2k$ is in particular a reconfiguration sequence from S to T. The reverse direction of Theorem 3.2 then implies that G contains a k-clique. Thus the bounded-length variant is NP-hard on DAGs. □

Corollary 3.3. Directed Clique Token Sliding *on DAGs is* W[1]*-hard when parameterized by the number k of tokens.*

Proof. The reduction in the proof of Theorem 3.2 maps an instance (G, k) of k-Clique to an instance of Directed Clique Token Sliding with exactly k tokens. Since k-Clique is W[1]-hard [3, Chapter 13], the claim follows. □

4. Polynomial-Time Algorithms

Proposition 4.1. *For every fixed integer $d \geq 1$,* Directed Clique Token Sliding *can be solved in polynomial time on DAGs of depth at most d.*

Proof. Let D be a DAG of depth at most d, and let C be a clique of $U(D)$. The orientation of $U(D)[C]$ is an acyclic tournament. If $v_1, \dots, v_{|C|}$ is the order of C induced by a topological ordering of D, then (v_i, v_j) is an arc of D for every $i < j$. Thus $v_1, v_2, \dots, v_{|C|}$ is a directed path in D. Hence $|C| \leq d$. It follows that $\omega(U(D)) \leq d$. Since d is fixed, Proposition 4.2 gives a polynomial-time algorithm. □

Proposition 4.2. *Let $c \geq 1$ be a fixed constant. If the input oriented graph D satisfies $\omega(U(D)) \leq c$, then* Directed Clique Token Sliding *can be solved in polynomial time.*

Proof. First verify that C_s and C_t are k-cliques of $U(D)$; if not, reject. If $k > c$, then no input satisfying $\omega(U(D)) \leq c$ has a feasible k-clique, so reject. Hence $k \leq c$ in the remaining case. Every feasible state is a k-clique of $U(D)$. Since $k \leq \omega(U(D)) \leq c$, the number of feasible states is at most $\sum_{i=0}^{c} \binom{|V(D)|}{i} = O(|V(D)|^c)$. We enumerate all k-cliques by brute force and build the directed state graph explicitly. For every clique C and every ordered pair (u, v) with $u \in C$ and $v \notin C$, we test whether $(u, v) \in A(D)$ and whether $C - u + v$ is a k-clique. The resulting graph has polynomial size, and reachability from C_s to C_t can be tested by breadth-first search or depth-first search. □

Proposition 4.3. *Let D be an oriented graph such that $U(D)$ is complete. If $1 \leq k \leq |V(D)| - 1$, then $\vec{\mathcal{C}}_k(D) \cong F_k(D)$. If $k = |V(D)|$, then $\vec{\mathcal{C}}_k(D)$ consists of one isolated vertex.*

Proof. First assume that $1 \leq k \leq |V(D)| - 1$. If $U(D)$ is complete, then D is a tournament and every k-subset of $V(D)$ is a k-clique. Therefore the vertices of $\vec{\mathcal{C}}_k(D)$ are all k-subsets of $V(D)$. By Definition 1.1, there is an arc from C to $C - u + v$ exactly when $(u, v) \in A(D)$. Thus $\vec{\mathcal{C}}_k(D) \cong F_k(D)$ [7]. If $k = |V(D)|$, then the only feasible state is $V(D)$, and no token can move. Hence $\vec{\mathcal{C}}_k(D)$ consists of one isolated vertex. □

By Proposition 4.3, for every tournament T and every $1 \le k \le |V(T)|-1$, $\overrightarrow{\mathcal{C}}_k(T) \cong F_k(T)$. Fernandes et al. [7, Lemma 3.2 and Theorem 3.3] describe the strongly connected components and the strong component digraph of $F_k(D)$ for every digraph D. Applying their results with $D = T$ gives the following tournament specialization. We retain a constructive proof of the resulting prefix criterion.

Theorem 4.1. *Let T be a tournament, let $1 \le k \le |V(T)|$, and let A and B be two k-subsets of $V(T)$. Let $Q_1, \ldots, Q_r$ be the strongly connected components of T, ordered so that for every $i < j$, every vertex of Q_i has an arc to every vertex of Q_j. For $i \in \{1, \ldots, r\}$, let $P_i = Q_1 \cup \cdots \cup Q_i$. Then B is reachable from A in $\overrightarrow{\mathcal{C}}_k(T)$ if and only if $|B \cap P_i| \le |A \cap P_i|$ for every $i \in \{1, \ldots, r-1\}$. Consequently,* Directed Clique Token Sliding *can be solved in polynomial time on tournaments.*

Proof. For a tournament, the stated SCC ordering exists because, if two SCCs had arcs in both directions between them, their internal strong connectivity would merge them into one SCC. Hence all arcs between two SCCs go one way, and the condensation is an acyclic tournament, so it has the displayed linear order.

If $k = |V(T)|$, then $A = B = V(T)$ and the claim is immediate. Assume that $1 \le k \le |V(T)|-1$. By Proposition 4.3, $\overrightarrow{\mathcal{C}}_k(T) \cong F_k(T)$.

For a k-subset C of $V(T)$, call $(|C \cap Q_1|, \ldots, |C \cap Q_r|)$ its *occupancy vector*. By Fernandes et al. [7, Lemma 3.2], two configurations with the same occupancy vector lie in the same strongly connected component of $F_k(T)$. Moreover, their description of its strong component digraph [7, Theorem 3.3] shows, in the tournament specialization, that an arc between occupancy vectors transfers one token from Q_p to Q_q whenever $p < q$.

First suppose that B is reachable from A. Fix $i \in \{1, \ldots, r-1\}$. No arc enters P_i from $V(T) - P_i$. Hence a token slide cannot increase the number of tokens in P_i. Therefore $|B \cap P_i| \le |A \cap P_i|$.

Conversely, suppose that the inequalities hold, and set $a_i = |A \cap Q_i|$ and $b_i = |B \cap Q_i|$ for every $i \in [r]$. For each component Q_i with $a_i > b_i$, choose $a_i - b_i$ vertices of $A \cap Q_i - B$ as sources. For each component Q_i with $b_i > a_i$, choose $b_i - a_i$ vertices of $B \cap Q_i - A$ as targets. These choices are possible because $|A \cap Q_i - B| \ge a_i - b_i$ and $|B \cap Q_i - A| \ge b_i - a_i$.

The prefix inequalities imply that every target can be assigned to a source in an earlier component: after scanning the first i components, the number of chosen sources is at least the number of chosen targets because $\sum_{j=1}^{i} a_j \ge \sum_{j=1}^{i} b_j$. At the end of the scan, the two totals are equal. Thus a greedy scan from Q_1 to Q_r works as follows. Keep a queue of unmatched sources seen so far; when scanning Q_i, first add the chosen sources in Q_i, then match every chosen target in Q_i to an earlier queued source. The prefix surplus invariant says the queue never becomes negative, and because $b_i > a_i$ for every target component Q_i, every matched source lies in some component strictly before its matched target. Hence every matched source-target pair (x, y) has $x \in Q_p$ and $y \in Q_q$ with $p < q$.

For every such matched pair, the SCC ordering of T implies that every vertex of Q_p has an arc to every vertex of Q_q, so $(x, y) \in A(T)$. We perform the matched slides in any order. The chosen sources lie in $A - B$ and the chosen targets in $B - A$, so the source set and target set are disjoint; in particular, no slide's source is another slide's target, and each slide replaces one source by one target without affecting the other matched sources or targets. Since T is a tournament, every k-subset of $V(T)$ is a clique of $U(T)$, so every intermediate state is a k-clique. Hence these inter-component moves are valid. After performing them, each component Q_i contains exactly b_i tokens.

Let C be the configuration obtained after these inter-component slides. Then $|C \cap Q_i| = b_i = |B \cap Q_i|$ for every i, so C and B have the same occupancy vector. By [7, Lemma 3.2], they lie in the same strongly connected component of $F_k(T)$; hence B is reachable from C.

Combining the inter-component moves with a path from C to B gives a directed clique token sliding sequence from A to B. The characterization can be checked after computing the strongly connected components of T, and hence the problem is in P on tournaments. This completes the proof. ☐

Transitive Oriented Graphs

We next treat transitive oriented graphs. Recall that every transitive oriented graph is acyclic. Its arc relation defines a *strict partial order*: write $x \prec y$ when $(x, y) \in A(D)$.

Theorem 4.2. *Let D be a transitive oriented graph, and let A and B be two k-cliques of $U(D)$. Write $A = \{a_1 \prec a_2 \prec \cdots \prec a_k\}$ and $B = \{b_1 \prec b_2 \prec \cdots \prec b_k\}$, where the two chains are listed in their unique increasing orders. Then B is reachable from A if and only if $a_i = b_i$ or $(a_i, b_i) \in A(D)$ for every $i \in \{1, \ldots, k\}$. Consequently,* Directed Clique Token Sliding *can be solved in polynomial time on transitive oriented graphs.*

Proof. We write $x \preceq y$ if $x = y$ or $(x, y) \in A(D)$. Recall that a *chain* is a set of pairwise comparable elements. Since D is transitive, the cliques of $U(D)$ are exactly the chains of this partial order.

First suppose that B is reachable from A. It is enough to show that one slide cannot decrease any coordinate of the sorted chain. Let $C = \{c_1 \prec c_2 \prec \cdots \prec c_k\}$ be a clique, and suppose a slide replaces c_j by a vertex v with $c_j \prec v$. Let $C' = \{c'_1 \prec c'_2 \prec \cdots \prec c'_k\}$ be the resulting clique. Removing one element of a chain and inserting a larger element cannot decrease any coordinate: since $c_j \prec v$ and the totally ordered chain $C - c_j + v$ is sorted, the new sorted positions satisfy $c_i \preceq c'_i$ for every i. Applying this argument to every slide gives $a_i \preceq b_i$ for every i.

Conversely, suppose that $a_i \preceq b_i$ for every i. We process the indices $k, k-1, \dots, 1$. Before processing i, the current clique is $\{a_1, \dots, a_i, b_{i+1}, \dots, b_k\}$. If $a_i = b_i$, then no move is needed. Otherwise, we slide the token from a_i to b_i. The arc (a_i, b_i) exists by assumption. The vertex b_i is unoccupied: it is distinct from $a_1, \dots, a_{i-1}$ because $a_j \prec a_i \prec b_i$ for every $j < i$, and it is distinct from $b_{i+1}, \dots, b_k$ because B is a set. The new set remains a chain because $a_1 \prec \cdots \prec a_{i-1} \preceq b_i \preceq b_{i+1} \prec \cdots \prec b_k$, where the endpoint inequalities are omitted when necessary. After all indices are processed, the current clique is B. The condition can be checked after sorting the two chains, so the algorithm runs in polynomial time. □

Following Gurski et al. [8, Definition 3], we use the term *oriented co-graph* for a digraph recursively obtained from one-vertex digraphs by disjoint union and order composition, where order composition adds all arcs from the first operand to the second. By [8, Theorem 5], every oriented co-graph is transitive. The following result follows from Theorem 4.2.

Corollary 4.1. Directed Clique Token Sliding *can be solved in polynomial time on oriented co-graphs.*

5. Conclusion

In this paper, we have introduced Directed Clique Token Sliding, proved PSPACE-completeness on oriented graphs, proved NP-completeness on directed acyclic graphs, and proved W[1]-hardness on directed acyclic graphs when parameterized by k. We also proved that the bounded-length variant is NP-complete on DAGs. On the positive side, we have derived from the token-digraph results of Fernandes et al. [7] an explicit polynomial-time reachability criterion for tournaments, and we have given polynomial-time algorithms for transitive oriented graphs, fixed-depth DAGs, and oriented graphs D for which $\omega(U(D))$ is bounded by a fixed constant.

The main open direction is to classify natural subclasses of oriented graphs: which restrictions preserve the PSPACE-complete behavior of the general case, and which force polynomial-time reachability? In particular, although Corollary 4.1 handles oriented co-graphs, it remains open whether arbitrary orientations of undirected cographs or complete multipartite graphs admit polynomial-time characterizations or sharper hardness results, and which parameterizations beyond $\omega(U(D))$ admit fixed-parameter algorithms.

Acknowledgments

This research was supported by the Vietnam National University, Hanoi under the project QG.25.07 “A study on reconfiguration problems from algorithmic and graph-theoretic perspectives”. The author thanks the anonymous reviewers for their useful comments and suggestions.

References

[1] N. Banerjee, C. Engels, D. A. Hoang, Directed token sliding, *arXiv*:2411.16149v2 [cs.DS], (2026).
[2] N. Bousquet, A. E. Mouawad, N. Nishimura, S. Siebertz, A survey on the parameterized complexity of reconfiguration problems, *Comput. Sci. Rev.* **53** (2024) #100663.
[3] M. Cygan, F. V. Fomin, Ł. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, S. Saurabh, *Parameterized Algorithms*, Springer, Cham, 2015.
[4] R. Diestel, *Graph Theory*, 5th Edition, Springer, Berlin, 2017.
[5] J. Dirks, A. Vigny, Token sliding reconfiguration on DAGs, In: P. Gawrychowski, F. Mazowiecki, M. Skrzypczak (Eds.), *50th International Symposium on Mathematical Foundations of Computer Science (MFCS 2025)*, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2025, 41:1–41:17.
[6] R. Fabila-Monroy, D. Flores-Peñaloza, C. Huemer, F. Hurtado, J. Urrutia, D. R. Wood, Token graphs, *Graphs Combin.* **28** (2012) 365–380.
[7] C. G. Fernandes, C. N. Lintzmayer, J. P. Peña, G. Santos, A. Trujillo-Negrete, J. Zamora, A study on token digraphs, *Discrete Math.* **349** (2026) 114951.
[8] F. Gurski, D. Komander, C. Rehs, On characterizations for subclasses of directed co-graphs, *J. Comb. Optim.* **41** (2021) 234–266.
[9] T. Ito, Y. Iwamasa, Y. Kobayashi, Y. Nakahata, Y. Otachi, M. Takahashi, K. Wasa, Independent set reconfiguration on directed graphs, *SIAM J. Discrete Math.* **40** (2026) 82–101.
[10] T. Ito, H. Ono, Y. Otachi, Reconfiguration of cliques in a graph, *Discrete Appl. Math.* **333** (2023) 43–58.
[11] R. M. Karp, Reducibility among combinatorial problems, In: R. E. Miller, J. W. Thatcher, J. D. Bohlinger (Eds.), *Complexity of Computer Computations*, Plenum Press, New York, 1972, 85–103.
[12] N. Nishimura, Introduction to reconfiguration, *Algorithms* **11** (2018) #52.
[13] J. van den Heuvel, The complexity of change, In: S. R. Blackburn, S. Gerke, M. Wildon (Eds.), *Surveys in Combinatorics 2013*, Cambridge University Press, Cambridge, 2013, 127–160.
[14] M. Wrochna, Reconfiguration in bounded bandwidth and tree-depth, *J. Comput. System Sci.* **93** (2018) 1–10.