Reconfiguring Independent Sets on Interval Graphs

Marcin Briański
Theoretical Computer Science Department, Faculty of Mathematics and Computer Science, Jagiellonian University, Kraków, Poland

Stefan Felsner
Institut für Mathematik, Technische Universität Berlin, Germany

Jędrzej Hodor
Theoretical Computer Science Department, Faculty of Mathematics and Computer Science, Jagiellonian University, Kraków, Poland

Piotr Micek
Theoretical Computer Science Department, Faculty of Mathematics and Computer Science, Jagiellonian University, Kraków, Poland

Abstract

We study reconfiguration of independent sets in interval graphs under the token sliding rule. We show that if two independent sets of size k are reconfigurable in an n-vertex interval graph, then there is a reconfiguration sequence of length $\mathcal{O}(k \cdot n^2)$. We also provide a construction in which the shortest reconfiguration sequence is of length $\Omega(k^2 \cdot n)$.

As a counterpart to these results, we also establish that Independent Set Reconfiguration is PSPACE-hard on incomparability graphs, of which interval graphs are a special case.

2012 ACM Subject Classification Mathematics of computing → Graph theory

Keywords and phrases reconfiguration, independent sets, interval graphs

Digital Object Identifier 10.4230/LIPIcs.MFCS.2021.23

Funding M. Briański, J. Hodor, P. Micek are partially supported by a Polish National Science Center grant (BEETHOVEN; UMO-2018/31/G/ST1/03718).

1 Introduction

Let G be a graph and let k be a positive integer. The *reconfiguration graph* $R_k(G)$ of independent sets of size k in G has as its vertex set the set of all independent sets of size k in G and two independent sets I, J of size k in G are adjacent in $R_k(G)$ whenever $I \triangle J = \{u, v\}$ and $uv \in E(G)$.

When two sets I and J are in the same component of $R_k(G)$ we say that I and J are *reconfigurable*. In such case, we can perform the following transformation process: (1) start by placing one token on each vertex of I; (2) in each step move one of the tokens to a neighboring vertex in G but always keep the property that the vertices occupied by tokens induce an independent set in G; (3) finish with tokens occupying all vertices of J. Let $(I_0, \ldots, I_m)$ be a path from I to J in $R_k(G)$ so $I_0 = I$, $I_m = J$. For $i \in \{1, \ldots, m\}$, let (u_i, v_i) be a pair of vertices in G such that $I_{i-1} \setminus I_i = \{u_i\}$ and $I_i \setminus I_{i-1} = \{v_i\}$. We say that the sequence $((u_1, v_1), \ldots, (u_m, v_m))$ is a *reconfiguration sequence* from I to J in G.

A large body of research is focused on the computational complexity of the corresponding decision problem – Independent Set Reconfiguration: given a graph G and two independent sets I and J, determine whether I and J are reconfigurable. If we do not assume anything about the input graph this problem is PSPACE-complete, thus it is natural to investigate how its complexity changes when input graphs are restricted to a particular class of graphs. Demaine et al. [4] showed that the problem can be solved in polynomial time on trees. Lokshtanov and Mouawad [8] proved that it remains PSPACE-complete on bipartite graphs.

46th International Symposium on Mathematical Foundations of Computer Science (MFCS 2021).
Editors: Filippo Bonchi and Simon J. Puglisi; Article No. 23; pp. 23:1–23:14
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Hearn and Demaine [6] showed that it is PSPACE-complete on planar graphs. There are polynomial time algorithms for the class of cographs [7] as well as claw-free graphs [2]. For a general survey of problems related to reconfiguration see [9].

Bonamy and Bousquet [1] presented an algorithm that given an interval graph G and two independent sets of size k in G verifies if the independent sets are reconfigurable in $\mathcal{O}(n^3)$ time. Curiously, their proof does not say anything meaningful about the length of the reconfiguration sequence if the two independent sets are reconfigurable. Since there are at most $\binom{n}{k}$ independent sets of size k, we get a trivial bound on the length of the reconfiguration sequence, namely: $\mathcal{O}(n^k)$. The question whether every two reconfigurable independent sets in an interval graph are connected by a reconfiguration sequence of polynomial length of degree independent of k was raised by Bousquet [3], which we answer in the affirmative in Theorem 1.

1.1 Our results

In this paper we will be mainly considering interval graphs. Our main results are stated in Theorems 1 and 2 below.

▶ **Theorem 1.** *Let G be an n-vertex interval graph and k be a positive integer. Then every component of $R_k(G)$ has diameter in $\mathcal{O}(k \cdot n^2)$.*

Moreover, there is a polynomial time algorithm that given G, k, and two independent sets I, J of size k in G decides if I and J are reconfigurable and if so outputs a reconfiguration sequence connecting them of length $\mathcal{O}(k \cdot n^2)$.

A lower bound construction shows that the bound on the length of a reconfiguration sequence given in Theorem 1 is close to tight.

▶ **Theorem 2.** *For all integers $m \geqslant 1$ and $k \geqslant 1$, there is an interval graph $G_{m,k}$ with $|V(G_{m,k})|$ in $\mathcal{O}(m + k)$ and two reconfigurable independent sets I, J of size k in $G_{m,k}$ such that every reconfiguration sequence connecting I and J in G is of length $\Omega(k^2 \cdot m)$.*

In light of Theorem 2, so long as $k \in \Theta(m)$, the bound of Theorem 1 is asymptotically tight. However, if k is small compared to n, the bounds of Theorems 1 and 2 can differ by a factor of $O(n)$. This leads to an interesting problem in its own right. The authors are not aware of any example giving a superlinear lower bound on the length of a reconfiguration sequence when the number of tokens is constant – *even on the class of all graphs.* Specifically, the case $k = 2$ remains open.

Interval graphs are a special case in the more general class of incomparability graphs. As a somewhat expected (and indeed easy) result we have obtained the following theorem.

▶ **Theorem 3.** *Independent Set Reconfiguration is PSPACE-hard on incomparability graphs, even on incomparability graphs of posets of width at most w for some constant $w \in \mathbb{N}$.*

Another interesting specialization of the incomparability graphs are *permutation graphs.* In [5] the authors show a polynomial time algorithm solving Independent Set Reconfiguration on bipartite permutation graphs. We suspect that general permutation graphs should be amenable to methods similar to the tools we use here on interval graphs, however we have not been able to successfully apply them.

▶ **Conjecture 4.** *Independent Set Reconfiguration is solvable in polynomial time when restricted to permutation graphs.*

2 Preliminaries

A *component* of a graph G is a non-empty induced subgraph of G that is connected and is vertex-maximal under these properties. The *length* of a path is the number of edges in the path. The distance of two vertices u, v in a graph G is the minimum length of a path connecting u and v in G. The *diameter* of a connected graph is the maximum distance between any pair of vertices in the graph.

A graph G is an *interval graph* if the vertices of G can be associated with intervals on the real line in such a way that two vertices are adjacent in G if and only if the corresponding intervals intersect. Given an interval graph we will always fix an interval representation and identify the vertices with the corresponding intervals. We will also assume that all the endpoints of intervals in the representation are pairwise distinct. This can be easily achieved by perturbing the endpoints where it is needed.

Let S be a reconfiguration sequence from I to J in G. We sometimes say that we *apply* S to I in G and obtain an independent set $S(I)$. When $S = ((u, v))$ we also simply say that we *apply* a pair (u, v) to I and obtain $S(I) = (I \setminus \{u\}) \cup \{v\}$.

Given a sequence S, we denote by $S\big|_t$ the prefix of S of length t (so $S\big|_0$ is the empty sequence). Given a reconfiguration sequence $S = ((u_1, v_1), \ldots, (u_m, v_m))$ from I to J in G, clearly for each $t \in \{0, \ldots, m\}$, the prefix $S\big|_t$ is a reconfiguration sequence from I to the set $S\big|_t(I)$ in G. Thus, we also have $S\big|_0(I) = I$.

3 Upper bound: The Algorithm

In this section we are going to prove Theorem 1. For brevity, we omit implementation details as well as running time analyses of the algorithms outlined in this section. We note that a straightforward implementation of Algorithm 3 below (which is the procedure whose existence implies Theorem 1) would yield a $\mathcal{O}(n^3)$ algorithm.

Let G be an n-vertex interval graph and let k be a positive integer. We fix an interval representation of G distinguishing all the endpoints. There are two natural linear orderings on the vertices of G: $\leqslant_{\text{left}}$ the order increasing along the left endpoints of the intervals and $\leqslant_{\text{right}}$ the order increasing along the right endpoints of the intervals.

An independent set in G is a set of pairwise disjoint intervals, as such they are naturally ordered on the line. Thus given an independent set $A = \{a_1, \ldots, a_\ell\}$ in G we will treat it as a tuple of intervals $(a_1, \ldots, a_\ell)$ with $a_1 < \cdots < a_\ell$ on the line. We define the projection $\pi_i(A) = a_i$ for each $i \in \{1, \ldots, \ell\}$. Also when we apply (u, v) to an independent set A, we say that (u, v) *moves the i-th token* of A when $u = a_i$.

Let ℓ be a positive integer and let $\mathcal{C}$ be a non-empty family of independent sets each of size ℓ in G. For $j \in \{1, \ldots, \ell\}$ we define

$$\mathrm{ex}_j(\mathcal{C}, \mathrm{left}) = \min_{\leqslant_{\text{right}}} \{\pi_j(A) \;:\; A \in \mathcal{C}\}, \quad \mathrm{ex}_j(\mathcal{C}, \mathrm{right}) = \max_{\leqslant_{\text{left}}} \{\pi_j(A) \;:\; A \in \mathcal{C}\}.$$

For $p \in \{0, \ldots, \ell\}$, we define the *$p$-extreme set* of $\mathcal{C}$ to be

$$\bigcup_{1 \leqslant j \leqslant p} \{\mathrm{ex}_j(\mathcal{C}, \mathrm{left})\} \quad \cup \bigcup_{p+1 \leqslant j \leqslant \ell} \{\mathrm{ex}_j(\mathcal{C}, \mathrm{right})\}.$$

We are going to show (Lemma 7) that every component $\mathcal{C}$ of $R_k(G)$ contains all its p-extreme sets for $p \in \{0, \ldots, k\}$. This gives a foundation for our algorithm: given two independent sets I and J as the input, we are going to devise two reconfiguration sequences, transforming I into the $(k-1)$-extreme set of its component in $R_k(G)$ and J into the $(k-1)$-extreme set of its component in $R_k(G)$ respectively. As the $(k-1)$-extreme set is a function of a connected component, we conclude that I and J are reconfigurable if and only if the obtained $(k-1)$-extreme sets are equal. Thus, the essence of our work is to show that every independent set I can be reconfigured in $\mathcal{O}(k \cdot n^2)$ steps into the $(k-1)$-extreme set of its component in $R_k(G)$.

The technical lemma below is our basic tool used to reason about and manipulate reconfiguration sequences.

▶ **Lemma 5.** *Let $A = (a_1, \ldots, a_\ell)$ and $X = (x_1, \ldots, x_\ell)$ be two independent sets in G and let $S = ((u_1, v_1), \ldots, (u_m, v_m))$ be a reconfiguration sequence from A to X. With $\mathcal{A} = \left\{S\big|_t(A) \;:\; t \in \{0, \ldots, m\}\right\}$ we denote the of all independent sets traversed from A to X along S. Suppose that there are $i, j \in \{0, \ldots, \ell+1\}$ with $i < j$ such that*

$$\begin{aligned} &\mathrm{ex}_i(\mathcal{A}, \mathrm{left}) &&= x_i \quad \text{if } i \geqslant 1, \text{ and} \\ &\mathrm{ex}_j(\mathcal{A}, \mathrm{right}) &&= x_j \quad \text{if } j \leqslant \ell. \end{aligned}$$

Then $A' = (x_1, x_2, \ldots, x_i, a_{i+1}, \ldots, a_{j-1}, x_j, \ldots, x_\ell)$ is also an independent set in G. Moreover, if we let S' be S restricted to those pairs (u_t, v_t) with u_t being at a position p with $i < p < j$ in $S\big|_{t-1}(A)$, then S' is a reconfiguration sequence from A' to X in G.

▶ Remark 6. Since none of the first i tokens nor the last $l+1-j$ tokens are affected by S' we can alternatively conclude that S' transforms $(a_{i+1}, \ldots, a_{j-1})$ into $(x_{i+1}, \ldots, x_{j-1})$ in $G \setminus \bigcup\limits_{p \notin \{i+1, \ldots, j-1\}} N[x_p]$

Proof. Let $B = (b_1, \ldots, b_\ell)$ be a set in $\mathcal{A}$. The *aligned* set of B is defined as $(x_1, \ldots, x_i, b_{i+1}, \ldots, b_{j-1}, x_j, \ldots, x_\ell)$. We claim that the aligned set of B is an independent set in G. Note that some of the three parts $Y_1 = (x_1, \ldots, x_i)$, $Y_2 = (b_{i+1}, \ldots, b_{j-1})$, $Y_3 = (x_j, \ldots, x_\ell)$ might be empty. Since all three parts are contained in an independent set, i.e. X or B, all we need to show is that (1) if Y_1 and Y_2 are non-empty, then x_i is completely to the left of b_{i+1}, and (2) if Y_2 and Y_3 are non-empty, then b_{j-1} is completely to the left of x_j. Thus, suppose that Y_1 and Y_2 are non-empty, so x_i and b_{i+1} exist. By the assumptions of the lemma $x_i = \mathrm{ex}_i(\mathcal{A}, \mathrm{left}) \leqslant_{\mathrm{right}} b_i$ and clearly b_i is completely to the left of b_{i+1}. Therefore, x_i is completely to the left of b_{i+1} as desired. Symmetrically, if Y_2 and Y_3 are non-empty, then $b_{j+1} \leqslant_{\mathrm{left}} x_{j+1}$ and b_j is completely to the left of b_{j+1}. Therefore, b_j is completely to the left of x_{j+1} as desired.

Since A' is the aligned set of A, we conclude that A' is independent in G. Let m' be the length of S' and $S' = ((u'_1, v'_1), \ldots, (u'_{m'}, v'_{m'}))$. Since S' is a subsequence of S, we can fix $\varphi(t)$, for each $t \in \{1, \ldots, m'\}$ such that the pair $(u_{\varphi(t)}, v_{\varphi(t)})$ in S corresponds to (u'_t, v'_t) in S'. Let $A_{\varphi(t)} = S\big|_{\varphi(t)}(A)$ and $A'_t = S'\big|_t(A')$. By construction we have

$$\begin{aligned} &\pi_p(A'_t) = \pi_p(X) && \text{if } p \in \{1, \ldots, i\} \cup \{j, \ldots, \ell\} \\ &\pi_p(A'_t) = \pi_p(A_{\varphi(t)}) && \text{if } p \in \{i+1, \ldots, j-1\}. \end{aligned}$$

Thus for each $t \in \{0, \ldots, m'\}$ we have that $S'\big|_t(A')$ is the aligned set of $A_{\varphi(t)} \in \mathcal{A}$. Therefore $S'\big|_t(A')$ is independent in G which completes the proof that S' is a reconfiguration sequence from A' to X. ◀

▶ **Lemma 7.** *Let H be a non-empty induced subgraph of G, let $\ell \in \{1, \ldots, k\}$, and let $\mathcal{C}$ be a component of $R_\ell(H)$. For every $p \in \{0, \ldots, \ell\}$, the p-extreme set of $\mathcal{C}$ is independent in H and lies in $\mathcal{C}$.*

Proof. Fix $p \in \{0, \dots, \ell\}$. Let $X = (x_1, \dots, x_\ell)$ be the p-extreme set of $\mathcal{C}$. We claim that for every $i \in \{0, \dots, p\}$ and $j \in \{p+1, \dots, \ell+1\}$, there is a set $A_{i,j} \in \mathcal{C}$ such that for all $q \in \{1, \dots, i\} \cup \{j, \dots, \ell\}$, we have $\pi_q(A_{i,j}) = x_q$. We prove this claim by induction on $m = i + (\ell + 1 - j)$. For the base case, when $m = 0$, so $i = 0$ and $j = \ell + 1$, we simply choose $A_{0,\ell+1}$ to be any element in $\mathcal{C}$.

For the inductive step, consider $m > 0$, so $i > 0$ or $j < \ell+1$. The two cases are symmetric, so let us consider only the first one. Thus suppose $i > 0$ and therefore by the induction hypothesis, we get an independent set $A_{i-1,j} \in \mathcal{C}$ of the form

$$(x_1, \dots, x_{i-1}, a_i, \dots, a_{j-1}, x_j, \dots, x_\ell),$$

where $a_i, \dots, a_{j-1}$ are some vertices of H. Let $B \in \mathcal{C}$ such that $\pi_i(B) = \mathrm{ex}_i(\mathcal{C}, \mathrm{left}) = x_i$. Since $B, A_{i-1,j} \in \mathcal{C}$, there is a reconfiguration sequence S from B to $A_{i-1,j}$. By Lemma 5, we obtain that

$$(x_1, \dots, x_{i-1}, x_i, b_{i+1}, \dots, b_{j-1}, x_j, \dots, x_\ell) \in \mathcal{C}.$$

This set witnesses the inductive condition for (i, j) and finishes the inductive step. ◀

When $k = 1$, there is a single token in the graph which moves along a path in the interval graph. This is a rather trivial setting still, we give explicit functions (Algorithm 1) to have a good base for the general strategy.

Algorithm 1 A simple algorithm for moving a single token.

function PushTokenLeft(H, u)
  $w := \min_{\leqslant_{\mathrm{right}}} \{v \in V(H) \;:\; v \text{ is reachable from } u \text{ in } H\}$
  let $(v_0, \dots, v_m)$ be the shortest path from u to w in H
  return $[w, ((v_0, v_1), \dots, (v_{m-1}, v_m))]$
function PushTokenRight(H, u)
  $w := \max_{\leqslant_{\mathrm{left}}} \{v \in V(H) \;:\; v \text{ is reachable from } u \text{ in } H\}$
  let $(v_0, \dots, v_m)$ be the shortest path from u to w in H
  return $[w, ((v_0, v_m), \dots, (v_{m-1}, v_m))]$

▶ **Proposition 8.** *Let H be an interval graph, and u be a vertex in H. Then* PushTokenLeft(H, u) *outputs the $\leqslant_{\mathrm{right}}$-minimum vertex w in the component of u in H and a witnessing u-w path $(v_0, v_1), \dots, (v_{m-1}, v_m)$ in H, where $v_0 = u, v_m = w$ and $v_m <_{\mathrm{right}} \cdots <_{\mathrm{right}} v_2 <_{\mathrm{right}} v_1$ and if $m \geqslant 2$, $v_2 <_{\mathrm{right}} v_0$.*

Symmetrically, PushTokenRight(H, u) *outputs the $\leqslant_{\mathrm{left}}$-maximum vertex w in the component of u in H and a witnessing u-w path $(v_0, v_1), \dots, (v_{m-1}, v_m)$ in H, where $v_0 = u, v_m = w$ and $v_m >_{\mathrm{left}} \cdots >_{\mathrm{left}} v_2 >_{\mathrm{left}} v_1$ and if $m \geqslant 2$, $v_2 >_{\mathrm{left}} v_0$.*

Proof. We prove the first part of the statement about PushTokenLeft(H, u). The second part is symmetric. Consider a shortest path $v_0 v_1 \cdots v_m$ from u to w in H. Note that there is no point on the line that belongs to three intervals in the path as otherwise we could make the path shorter. It is easy to see that v_{i+1} left overlaps v_i, i.e., $v_{i+1} <_{\mathrm{right}} v_i$ and $v_{i+1} <_{\mathrm{left}} v_i$ for all $i \in \{0, \dots, m-1\}$ possibly except two cases: v_m may be contained in v_{m-1} and (if $m \geqslant 2$) v_1 may contain v_0. See Figure 1 for an illustration of such a shortest path in an interval graph. ◀

For $k = 2$, we give an algorithm, see Algorithm 2, that finds a short reconfiguration from a given independent set A in H to the 1-extreme set of the component of A in a reconfiguration graph $R_2(H)$.

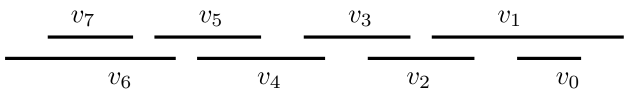

Figure 1 A shortest path from v_0 to v_7 where v_7 is an interval with leftmost right endpoint.

Algorithm 2 A specialized algorithm finding the 1-extreme set of the component of the given set A in $R_2(H)$.

```
function PushApart(H, A)
    a := π₁(A) b := π₂(A)
    S := ()
    do
        [a, S₁] := PushTokenLeft(H\N[b], a)
        [b, S₂] := PushTokenRight(H\N[a], b)
        S := Concat(S, S₁, S₂)
    while Concat(S₁, S₂) ≠ ∅
    return (a, b), S
```

▶ **Proposition 9.** *Let H be an interval graph and A be an independent set of size 2 in H. Then* PushApart(H, A) *outputs the* 1*-extreme set of the component of A in $R_2(H)$ and a reconfiguration sequence from A to the* 1*-extreme set of length at most* $2|V(H)|$.

Proof. Suppose that PushApart(H, A) outputs $(a^\star, b^\star)$ and let (x_1, x_2) be the 1-extreme set of the component $\mathcal{C}$ of A in $R_2(H)$. Suppose to the contrary that $(a^\star, b^\star) \neq (x_1, x_2)$ Thus, either $x_1 = \mathrm{ex}_1(\mathcal{C}, \mathrm{left}) <_{\mathrm{right}} a^\star$ or $b^\star <_{\mathrm{left}} \mathrm{ex}_2(\mathcal{C}, \mathrm{right}) = x_2$.

Consider a path $((a_0, b_0), \ldots, (a_m, b_m))$ from $(a^\star, b^\star)$ to (x_1, x_2) in $R_2(G)$. Let i be the smallest index such that $a_i <_{\mathrm{right}} a_0 = a^\star$ or $b^\star <_{\mathrm{left}} b_i$. This index is well-defined as $i = m$ satisfies the condition and obviously $i > 0$. Now suppose that $a_i <_{\mathrm{right}} a^\star$. The proof of the other case goes symmetrically. By the minimality of i we have

$$a_i <_{\mathrm{right}} a^\star \leqslant_{\mathrm{right}} a_{i-1} \qquad \text{and} \qquad b_i = b_{i-1} \leqslant_{\mathrm{left}} b^\star.$$

The inequalities on the right endpoints of a_i, $a^\star$, and a_{i-1} imply that these three intervals form a connected subgraph of H, because a_i and a_{i-1} must intersect as they are adjacent. Since a_{i-1} and a_i are in the same component of $H \setminus N[b_i]$, we conclude that $a^\star$ is also in that component. Finally, since $b_i \leqslant_{\mathrm{left}} b^\star$ we conclude that a_{i-1}, $a^\star$, a_i are together in the same component of $H \setminus N[b^\star]$. See Figure 2 that illustrates all these inequalities. Consider the last

Figure 2 $a_i <_{\mathrm{right}} a^\star \leqslant_{\mathrm{right}} a_{i-1}$ and $b_i \leqslant_{\mathrm{left}} b^\star$. The intervals a_i, $a^\star$, a_{i-1} are together in a component of $H \setminus N[b_i]$ and so they are together in a component of $H \setminus N[b^\star]$ as well.

iteration of the loop in PushApart(H, A). The variables a and b keep the values $a^\star$ and $b^\star$ in this iteration. In particular PushTokenLeft$(H \setminus N[b^\star], a^\star)$ did not change the value of a. This is a contradiction as by Proposition 8 PushTokenLeft$(H \setminus N[b^\star], a^\star)$ outputs the $\leqslant_{\mathrm{right}}$ minimum vertex in the component of $a^\star$ in $H \setminus N[b^\star]$ but we already know that $a^\star$ is not $\leqslant_{\mathrm{right}}$-minimum there.

Now, let us prove the claim about the length of the output reconfiguration sequence S. Let $(a_0, b_0), \ldots, (a_m, b_m)$ be the path in $R_2(H)$ corresponding to S, that is $(a_t, b_t) = S|_t(A)$ for $t \in \{0, \ldots, m\}$. Suppose that $(a, b) = (a^0, b^0)$ at the beginning of the loop in line 4 and $(a, b) = (a^1, b^1)$ after pushing left and right in lines 5-6. By Proposition 8, we know that $a^1 \leqslant_{\text{right}} a^0$ and $b^0 \leqslant_{\text{left}} b^1$ and every intermediate configuration (a', b') on the path satisfies $a^1 \leqslant_{\text{right}} a' \leqslant_{\text{right}} a^0$ and $b^0 \leqslant_{\text{left}} b' \leqslant_{\text{left}} b^1$ with a possible exception of the first move from a^0 and the first move from b^0. This way we see that the total number of steps without the exceptional moves is at most $|V(H)|$ and each interval in H can be the target of at most one exceptional move. This gives a bound $2|V(H)|$, as desired. ◀

We present now our main reconfiguration algorithm, see Algorithm 3.

Algorithm 3 A general algorithm finding the $(k-1)$-extreme set of the component of the given set A in $R_k(G)$.

1: **function** Reconfigure(G, k, A)
2:     **for** $i \in \{1, 2, \ldots, k\}$ **do**
3:         $a_i := \pi_i[A]$
4:         $\text{lext}_i := a_i$ ▷ lext_k is unused
5:         $\text{rext}_i := a_i$ ▷ rext_1 is unused
6:     $j := 1;\ S = ()$
7:     **while** $j < k$ **do**
8:         $[(a_j, a_{j+1}), S'] :=$ PushApart$(G \setminus \bigcup_{i \neq j, j+1} N[a_i], \{a_j, a_{j+1}\})$
9:         S.Append(S')
10:         $\gamma = 1$
11:         **if** $(a_j, a_{j+1}) \neq (\text{lext}_j, \text{rext}_{j+1})$ **then**
12:             $\text{lext}_j := a_j$
13:             $\text{rext}_{j+1} := a_{j+1}$
14:             **if** $j > 1$ **then** $\gamma = -1$
15:         $j := j + \gamma$
16:     **return** $(a_1, \ldots, a_k)$, S

▶ **Lemma 10.** *Let $k \geqslant 2$ and let A be an independent set of size k in G. Then* Reconfigure(G, k, A) *outputs the $(k-1)$-extreme set of the component of A in $R_k(G)$ and a reconfiguration sequence from A to this set of length $\mathcal{O}(k \cdot n^2)$.*

Proof. We begin our consideration of Algorithm 3 by noting two invariants.

▷ Claim 11. Every time Algorithm 3 reaches line 15, we have

$$(a_1, \ldots, a_k) = (\text{lext}_1, \ldots, \text{lext}_j, \text{rext}_{j+1}, \ldots, \text{rext}_k).$$

Proof. Note that the equation holds after the initialization in lines 4-5. Later on, the values of $(a_1, \ldots, a_k)$ are updated only in line 8 and if so then corresponding values of lext and rext are updated in lines 12-13. ◁

▷ Claim 12. For every $j \in \{2, \ldots, k\}$, the values held by lext_j are nonincreasing with respect to $\leqslant_{\text{right}}$. Symmetrically, for every $j \in \{1, \ldots, k-1\}$, the values held by rext_j are nondecreasing with respect to $\leqslant_{\text{left}}$.

Proof. We prove the statement by induction over the iterations steps in Algorithm 3. Consider the moment when the values of lext_j and rext_{j+1} are updated in lines 12-13. Let $(\ell_1, \ell_2, \ldots, \ell_k)$ and $(r_1, r_2, \ldots, r_k)$ be the values held by vectors lext and rext just after this update. Let $(\ell'_1, \ldots, \ell'_k)$ and $(r'_1, \ldots, r'_k)$ be the values held by vectors lext and rext just after the previous update of lext_j or rext_{j+1} in lines 12-13 or the initial values of these vectors (assigned in lines 4-5) if there was no previous update. All we need to show is that $\ell_j \leqslant_{\mathrm{right}} \ell'_j$ and $r'_{j+1} \leqslant_{\mathrm{left}} r_{j+1}$.

By Claim 11 we know that the algorithm had tokens in $(\ell'_1, \ldots, \ell'_j, r'_{j+1} \ldots, r'_k)$ and after applying some reconfiguration sequence, say $S = ((u_1, v_1), \ldots, (u_m, v_m))$, it reached the configuration $(\ell_1, \ldots, \ell_j, r_{j+1}, \ldots, r_k)$. Let $\mathcal{A} = \{S\big|_t(A) : t \in \{0, \ldots, m\}\}$.

By induction hypothesis the statement holds for all the updates applied so far by Algorithm 3. In particular, ℓ_{j-1} is the $\leqslant_{\mathrm{right}}$-minimal position of the $(j-1)$-th token so far and r_{j+2} it the $\leqslant_{\mathrm{left}}$-maximal position of the $(j+2)$-th token so far (assuming that these tokens exist). Thus,

$$\begin{aligned} \ell_{j-1} &= \mathrm{ex}_{j-1}(\mathcal{A}, \mathrm{left}) && \text{if } j-1 \geqslant 1, \\ r_{j+2} &= \mathrm{ex}_{j+2}(\mathcal{A}, \mathrm{right}) && \text{if } j+2 \leqslant k. \end{aligned}$$

Therefore, we may apply Lemma 5 and Remark 6 and conclude that (ℓ'_j, r'_{j+1}) and (ℓ_j, r_{j+1}) are in the same component of $R_2(G \setminus \bigcup_{i \neq j, j+1} N[a_i])$. By Proposition 9, the execution of PushApart in line 8 outputs the 1-extreme set, namely (ℓ_j, r_{j+1}), of the component of (ℓ'_j, r'_{j+1}) in $R_2(G \setminus \bigcup_{i \neq j, j+1} N[a_i])$. Therefore, $\ell_j \leqslant_{\mathrm{right}} \ell'_j$ and $r_{j+1} \geqslant_{\mathrm{left}} r'_{j+1}$, as desired. ◁

▷ Claim 13. Consider a moment when Algorithm 3 reaches the line 15 and $\gamma = 1$. Let α be the value of variable j prior to the update. Let $(\ell_1, \ldots, \ell_k)$, $(r_1, \ldots, r_k)$ be the values held at this moment by vectors lext and rext, respectively. Then,

$$\begin{aligned} \ell_i &= \mathrm{ex}_i(\mathcal{C}, \mathrm{left}) && \text{for } i \in \{1, \ldots, \alpha\}, \\ r_{\alpha+1} &= \mathrm{ex}_{\alpha+1}(\mathcal{C}, \mathrm{right}), \end{aligned}$$

where $\mathcal{C}$ is the component of $(\ell_1, \ldots, \ell_\alpha, r_{\alpha+1})$ in $R_{\alpha+1}(G \setminus \bigcup_{i > \alpha+1} N[r_i])$. In particular, $(\ell_1, \ldots, \ell_\alpha, r_{\alpha+1})$ is the α-extreme set in $\mathcal{C}$.

Proof. We proceed by induction on α. First we deal with $\alpha = 1$. When Algorithm 3 starts an iteration of the while loop with $j = 1$, then by Claim 11 (and initialization in lines 4-5), we have $(a_3, \ldots, a_k) = (r_3, \ldots, r_k)$. By Proposition 9, $\textsc{PushApart}(G \setminus \bigcup_{i>2} N[r_i], \{a_1, a_2\})$ executed in line 8 outputs the 1-extreme set of the component of $\{a_1, a_2\}$ in $R_2(G \setminus \bigcup_{i>2} N[r_i])$. After the update in lines 12-13, this set is stored in $\{\mathrm{lext}_1, \mathrm{rext}_2\} = \{\ell_1, r_2\}$ when Algorithm 3 reaches the line 15, as desired.

Let us assume that $\alpha > 1$ and that the claim holds for all smaller values of α. Consider an iteration of the while loop with $j = \alpha$ such that Algorithm 3 reaches line 15 with $\gamma = 1$. Let $(\ell_1, \ldots, \ell_k)$, $(r_1, \ldots, r_k)$ be the values held at this moment by vectors lext and rext, respectively. For convenience, we call this iteration the *present* iteration.

Now starting from the present iteration consider the last iteration before with $j = \alpha - 1$. We call this iteration the *past* iteration. Clearly, the past iteration had to conclude with $\gamma = 1$ and all iterations between the past and the present (there could be none) must have the value of variable $j \geqslant \alpha$ and those with $j = \alpha$ must conclude with $\gamma = 1$. This implies that the values of $(\mathrm{lext}_1, \ldots, \mathrm{lext}_\alpha)$ and $(\mathrm{rext}_2, \ldots, \mathrm{rext}_{\alpha+1})$ did not change between the past and the present iterations so they constantly are $(\ell_1, \ldots, \ell_\alpha)$ and $(r_2, \ldots, r_{\alpha+1})$.

Let $\mathcal{D}$ be the connected component of $(\ell_1, \ell_2, \dots, \ell_{\alpha-1}, r_\alpha)$ in $R_\alpha(G \backslash N[r_{\alpha+1}])$. The inductive assumption for the past iteration yields:

$$\begin{aligned} \ell_i &= \mathrm{ex}_i(\mathcal{D}, \mathrm{left}) && \text{for } i \in \{1, \dots, \alpha - 1\}, \text{ and} \\ r_\alpha &= \mathrm{ex}_\alpha(\mathcal{D}, \mathrm{right}), \end{aligned} \tag{$\star$}$$

Note that in the iteration immediately following the past iteration (this was an iteration with $\alpha = j$ and may be the present iteration) the only token movement was the travel of the j-th token from r_α to ℓ_α (the $(j+1)$-th token stays at $r_{\alpha+1}$). In particular, there is a path from r_α to ℓ_α in $G \backslash (N[\ell_{\alpha-1}] \cup N[r_{\alpha+1}])$. Therefore, there is a path connecting $(\ell_1, \dots, \ell_{\alpha-1}, r_\alpha)$ and $(\ell_1, \dots, \ell_\alpha)$ in $R_\alpha(G \backslash N[r_{\alpha+1}])$, so both independent sets are in $\mathcal{D}$.

Now we argue, that

$$\ell_\alpha = \mathrm{ex}_\alpha(\mathcal{D}, \mathrm{left}).$$

Indeed, take any $(v_1, v_2, \dots, v_\alpha) \in \mathcal{D}$ and we aim to show that $\ell_\alpha \leqslant_{\mathrm{right}} v_\alpha$. As $(v_1, \dots, v_\alpha)$ and $(\ell_1, \dots, \ell_\alpha)$ are in one component of the reconfiguration graph $R_\alpha(G \backslash N[r_{\alpha+1}])$ and by ($\star$), we may apply Lemma 5 to conclude that $(\ell_1, \ell_2, \dots, \ell_{\alpha-1}, v_\alpha)$ also lies in $\mathcal{D}$. Moreover by Remark 6, the vertices ℓ_α and v_α are connected by a path in $G \backslash (N[\ell_{\alpha-1}] \cup N[r_{\alpha+1}])$. Thus by Proposition 9, PushApart executed in the present iteration guarantees that $\ell_\alpha \leqslant_{\mathrm{right}} v_\alpha$ as claimed.

Let $\mathcal{C}$ be the component of $(\ell_1, \dots, \ell_\alpha, r_{\alpha+1})$ in $R_{\alpha+1}(G \setminus \bigcup_{i > \alpha+1} N[r_i])$. We proceed to argue that

$$r_{\alpha+1} = \mathrm{ex}_{\alpha+1}(\mathcal{C}, \mathrm{right}).$$

Assume for the sake of contradiction that there is some $A \in \mathcal{C}$, such that $r_{\alpha+1} <_{\mathrm{left}} \pi_{\alpha+1}(A)$ and fix such an A with a shortest possible reconfiguration sequence $S = ((u_1, v_1), \dots, (u_m, v_m))$ from $(\ell_1, \dots, \ell_\alpha, r_{\alpha+1})$ to A in $R_{\alpha+1}(G \setminus \bigcup_{i > \alpha+1} N[r_i])$. Let $A_t = S\big|_t((\ell_1, \dots, \ell_\alpha, r_{\alpha+1}))$, for all $t \in \{0, \dots, m\}$. By the choice of A and S, for every $t \in \{0, \dots, m-1\}$ we have $\pi_{\alpha+1}(A_t) \leqslant_{\mathrm{left}} r_{\alpha+1}$. We apply now Lemma 5 (with $i = 0$, $j = \alpha + 1$) to a path from $A_0 = (\ell_1, \dots, \ell_\alpha, r_{\alpha+1})$ to A_{m-1} and conclude that for each $t \in \{0, \dots, m-1\}$ the set $A'_t = (\pi_1(A_t), \dots, \pi_\alpha(A_t), r_{\alpha+1})$ is an independent set in $\mathcal{C}$. Consider now a path of independent sets of size α formed by dropping the $(\alpha+1)$-th coordinate of each set in the path $(A'_0, \dots, A'_{m-1}, A_m)$. Since $(\pi_1(A_0), \dots, \pi_\alpha(A_0)) = (\ell_1, \dots, \ell_\alpha) \in \mathcal{D}$, the whole path lives in $\mathcal{D}$. Therefore, by ($\star$) we have

$$\ell_i \leqslant_{\mathrm{right}} \pi_i(A_t),$$

for all $t \in \{0, \dots, m\}$ and $i \in \{1, \dots, \alpha\}$. But this in turn allows us to apply Lemma 5 and Remark 6 once more (this time with $i = \alpha$, $j = \alpha + 2$) to a path from $(\ell_1, \dots, \ell_\alpha, r_{\alpha+1})$ to A_m and we conclude that $r_{\alpha+1}$ and $\pi_{\alpha+1}(A_m)$ are in the same component of $G \backslash \left(N[\ell_\alpha] \cup \bigcup_{i > \alpha+1} N[r_i] \right)$. But PushApart executed in the present iteration outputs $(\ell_\alpha, r_{\alpha+1})$ while $r_{\alpha+1} <_{\mathrm{left}} \pi_{\alpha+1}(A_m)$. This contradicts Proposition 9 and completes the proof that $r_{\alpha+1} = \mathrm{ex}_{\alpha+1}(\mathcal{C}, \mathrm{right})$.

It remains to prove that $\ell_i = \mathrm{ex}_i(\mathcal{C}, \mathrm{left})$ for all $i \in \{1, \dots, \alpha\}$. Pick an arbitrary $A = (v_1, v_2, \dots, v_{\alpha+1}) \in \mathcal{C}$. Since we already know that $r_{\alpha+1} = \mathrm{ex}_{\alpha+1}(\mathcal{C}, \mathrm{right})$, we can apply Lemma 5 (with $i = 0$, $j = \alpha + 1$) to a path from $(\ell_1, \dots, \ell_\alpha, r_{\alpha+1})$ to $(v_1, v_2, \dots, v_{\alpha+1})$, and we conclude that there is a reconfiguration sequence transforming $(v_1, \dots, v_\alpha)$ into $(\ell_1, \dots, \ell_\alpha)$ in $G \setminus N[r_{\alpha+1}]$). Thus, this path lies in $\mathcal{D}$ and the desired inequalities $\ell_i \leqslant_{\mathrm{right}} v_i$ for all $i \in \{1, \dots, \alpha\}$ follow by ($\star$). ◁

Clearly, Claims 13 and 11 establish the correctness of the algorithm. Equipped with the invariant given by Claim 12, we can bound the length of the returned reconfiguration sequence. Indeed, observe that in each iteration of the while loop in line 7 either lext_j decreases wrt. $\leqslant_{\mathrm{right}}$, or rext_{j+1} increases wrt. $\leqslant_{\mathrm{left}}$ while j drops by 1, or j increases by 1. Now this implies that the outer loops can iterate at most $4nk + k$, as the quantity

$$j + 2\sum_{i=1}^{k} \mathrm{Index}_{\leqslant_{\mathrm{left}}}(\mathrm{rext}_i) + (n - \mathrm{Index}_{\leqslant_{\mathrm{right}}}(\mathrm{lext}_i) + 1),$$

where $\mathrm{Index}_{\leqslant}(x)$ denotes the position of element x in a given linear order $\leqslant$ on some fixed finite set, increases by at least one in each iteration and it is at most $4nk + k$.

As seen in Proposition 9, each call of the procedure PushApart returns a sequence consisting of at most $2n$ moves. Therefore, the length of reconfiguration sequence returned by Algorithm 3 is at most $8kn^2 + 2kn \in \mathcal{O}(kn^2)$. This completes the proof of Theorem 1. ◀

4 Lower bound: The Example

We present a family of graphs $\{G_{m,k}\}_{m,k\geqslant 1}$, such that $|V(G_{m,k})| = 8k+2m-5$ and $R_k(G_{m,k})$ contains a component of diameter at least $\frac{k^2}{4} \cdot m$. This will prove Theorem 2.

Fix integers $m, k \geqslant 1$. We will describe a family of intervals $\mathcal{I}_{m,k}$. The graph $G_{m,k}$ will be simply the intersection graph of $\mathcal{I}_{m,k}$. We construct the family in three steps. We initialize $\mathcal{I}_{m,k}$ with $(k-1) + (m+2k-1) + k$ pairwise disjoint intervals:

$$a_{k-1}, \ldots, a_1, v_1, \ldots, v_{m+2k-1}, b_1, \ldots, b_k,$$

listed with their natural left to right order on the line. We call these intervals, the *base* intervals. Let $N = m + 2k - 1$. We put into $\mathcal{I}_{m,k}$ further $N - 1$ intervals:

$$v_{1,2}, v_{2,3}, \ldots, v_{N-1,N},$$

where for each $i \in \{1, \ldots, N-1\}$, the interval $v_{i,i+1}$ is an open interval with the left endpoint in the middle of v_i and the right endpoint in the middle of v_{i+1}. We call these intervals the *path* intervals. Finally, we put into $\mathcal{I}_{m,k}$ two groups of *long* intervals:

$$\ell_1, \ldots, \ell_{k-1} \text{ and } r_1, \ldots, r_k,$$

where for each $i \in \{1, \ldots, k-1\}$ the interval ℓ_i is the open interval with the left endpoint coinciding with the left endpoint of a_i and the right endpoint coinciding with the right endpoint of $v_{N-(k-1)-i}$. Symmetrically, for each $i \in \{1, \ldots, k\}$ the interval r_i is the open interval with the left endpoint coinciding with the left endpoint of v_{k-i+1} and the right endpoint coinciding with the right endpoint of b_i. This completes the construction of $\mathcal{I}_{m,k}$. See Figure 3.

Consider two independent sets $I = (v_1, \ldots, v_k)$ and $J = (b_1, \ldots, b_k)$ in $G_{m,k}$.

▶ **Lemma 14.** *The sets I and J are in the same component of $R_k(G_{m,k})$ and every reconfiguration sequence from I to J has length at least $\frac{k^2}{4} \cdot m$.*

Proof. We put most of the effort to prove the second part of the statement, that every reconfiguration sequence from I to J has length at least $\frac{k^2}{4} \cdot m$.

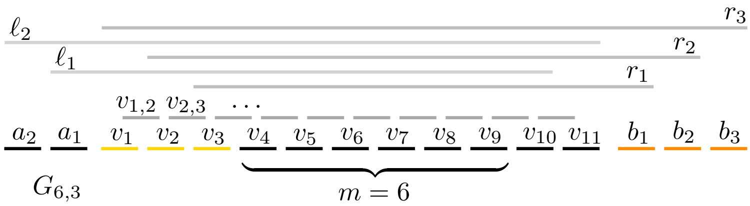

Figure 3 The graph $G_{6,3}$ with two distinguished independent sets $I = \{v_1, v_2, v_3\}$ and $J = \{b_1, b_2, b_3\}$.

We define a sequence of independent sets (see Figure 4):

$$
\begin{aligned}
C_0 &= (v_1, \ldots, v_k) = I,\\
C_1 &= (v_1, \ldots, v_{k-1}, r_1),\\
C_2 &= (\ell_1, v_{N-(k-1)}, \ldots, v_N, b_1),\\
&\vdots\\
C_{2i-1} &= (a_{i-1}, \ldots, a_1, v_1, \ldots, v_{k-1}, r_i),\\
C_{2i} &= (\ell_i, v_{N-(k-i-2)}, \ldots, v_N, b_1, \ldots, b_i),\\
&\vdots\\
C_{2k} &= (b_1, \ldots, b_k) = J.
\end{aligned}
$$

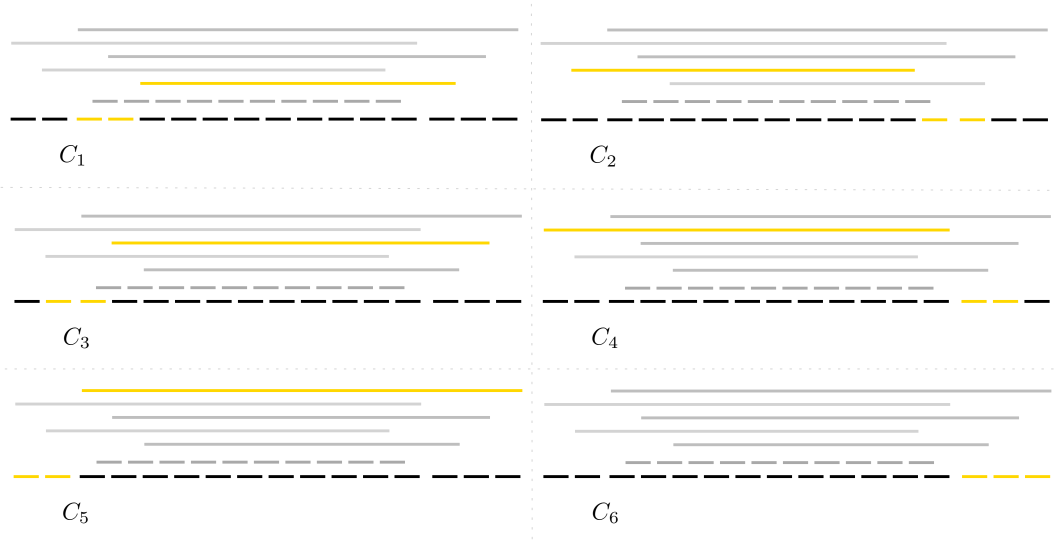

Figure 4 The sets $C_1, \ldots, C_6$ in $G_{6,3}$.

It is easy to construct a path from C_j to C_{j+1} in $R_k(G_{m,k})$ for $j \in \{0, \ldots, 2k-1\}$ which proves that I, J are in the same component of $R_k(G_{m,k})$.

Let $(K_0, \ldots, K_M)$ be a path in $R_k(G_{m,k})$ from I to J. The proof will follow from two claims. The first one is that $(C_0, \ldots, C_{2k})$ is a subsequence of $(K_0, \ldots, K_M)$, and the second one is that for every $i \in \{1, \ldots, k-1\}$ every path from C_{2i-1} to C_{2i} is of length at least $(k - 2i - 1) \cdot m$. A symmetric argument can be used to bound the distance between C_{2i} and C_{2i+1} which we omit here as it would only improve the final lower bound by a constant factor.

Let P be the set of path intervals in $G_{m,k}$. Define for each $i \in \{1, \ldots, k-1\}$ the following graphs:

$$H_{2i-1} = G_{m,k}[\{a_{i-1}, \ldots, a_1, v_1, \ldots, v_N, b_1, \ldots, b_i\} \cup P],$$
$$H_{2i} = G_{m,k}[\{a_i, \ldots, a_1, v_1, \ldots, v_N, b_1, \ldots, b_i\} \cup P].$$

Note that $C_0 = K_0$ and $C_{2k} = K_M$, so C_0 and C_{2k} occurs in $(K_0, \ldots, K_M)$. Fix $j \in \{0, \ldots, 2k-2\}$. Suppose that the independent set C_j occurs in $(K_0, \ldots, K_M)$ and fix such an occurrence. We will argue that C_{j+1} must occur afterwards in the sequence.

Observe that all base intervals from C_j are in H_j. However $b_k \notin H_j$ and $b_k \in K_M$, hence to reconfigure from C_j to K_M eventually a token has to be moved to some base interval not in H_j. Thus, let X_j be the set of base intervals not in H_j, i.e.

$$X_j = \begin{cases} \{a_{k-1} \ldots, a_i\} \cup \{b_{i+1}, \ldots, b_k\} & \text{if } j \text{ is odd,} \\ \{a_{k-1} \ldots, a_{i+1}\} \cup \{b_{i+1}, \ldots, b_k\} & \text{if } j \text{ is even.} \end{cases}$$

Note that the only neighbours of intervals in X_j are long. Let Y be the first independent set in $(K_0, \ldots, K_M)$ that occurs after the fixed occurence of C_j and contains a long interval u_0 neighbouring some element in X_j. We claim that $Y = C_{j+1}$.

First, we show that $u_0 = \ell_i$ if $j = 2i-1$ and $u_0 = r_{i+1}$ if $j = 2i$ respectively. Assume for now that $j = 2i-1$. Observe that for all $p \in \{1, \ldots, i-1\}$ we have $N(\ell_p) \cap X_j = \emptyset$ and consequently $u_0 \neq \ell_p$. On the other hand, for all $p \in \{i+1, \ldots, k-1\}$ we have $\alpha(H_j \backslash N(\ell_p)) < k-1$. Therefore, whenever $u_0 = \ell_p$ there is a token in Y that is not in H_j. This contradicts the minimality of Y. Moreover, for all $p \in \{1, \ldots, i+1\}$ we have $N(r_p) \cap X_j = \emptyset$ in turn implying that $u_0 \neq r_p$. On the other hand, for all $p \in \{i+2, \ldots, k-1\}$ we have $\alpha(H_j \backslash N(r_p)) < k-1$, thus, $u_0 \neq r_p$. This leaves only one possible option of $u_0 = \ell_i$. The case $j = 2i$ follows a symmetric argument. See Figure 5.

Recall that all $k-1$ elements of $Y \backslash \{u_0\}$ must be in H_j. It is easy to see that when $j = 2i-1$ then $H_j \backslash N(\ell_{i+1})$ has exactly one independent set of size $k-1$, namely: $\{v_{N-(k-i-2)}, \ldots, v_N, b_1 \ldots, b_i\}$, symmetrically when $j = 2i$ then $H_j \backslash N(r_{i+1})$ has exactly one independent set of size $k-1$, namely: $\{a_i, \ldots, a_1, v_1, \ldots, v_{k-1}\}$. This proves that $Y = C_{j+1}$.

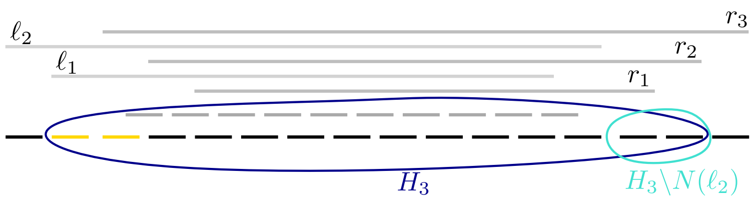

Figure 5 The set H_3 for $G_{6,3}$. We interpret $H_3 \backslash N(u)$ as the space where $k-1$ tokens can "hide". All base neighbours of ℓ_1, r_1, and r_2 are in H_3. Also, $\alpha(H_3 \backslash N(r_3)) = 1 < 2$. This gives $u_0 = \ell_2$.

Let us now prove that for a fixed $i \in \{1, \ldots, k-1\}$ every path from C_{2i-1} to C_{2i} in $R_k(G_{m,k})$ is of length at least $(k-2i-1) \cdot m$.

Fix the shortest reconfiguration sequence from C_{2i-1} to C_{2i}. As tokens do not interchange their relative positions, note that tokens starting at the positions $(v_1, \ldots, v_{k-2i})$ in C_{2i-1} must finish at the positions $(v_{N-(k-2i-1)}, \ldots, v_N)$ in C_{2i}. We call these tokens *heavy*. Their left to right ordinal numbers are $i \ldots, k-i$, and there are exactly $s := k-2i-1$ of them.

We prove that a heavy token cannot use any of the long intervals during the reconfiguration. By the first part of the proof we know that on the shortest path from C_{2i-1} to C_{2i} only base intervals from H_{2i-1} can be used. For each long interval u we define H_u^ℓ as a graph induced by all intervals v in $H_{2i-1} \backslash N(u)$ completely to the left of u. Analogously, define H_u^r as the graph induced by all $v \in H_{2i-1} \backslash N(u)$ completely to the right to u. Finally, put

$$n_\ell(u) = \alpha(H_u^\ell) \text{ and } n_r(u) = \alpha(H_u^r).$$

Assume that a heavy token uses a fixed long interval w on the path from C_{2i-1} to C_{2i}. Armed with the knowledge of the ordinal numbers of the heavy tokens, we see that: $n_\ell(w) \geqslant i-1$ and $n_r(w) \geqslant i$. Elementary computation shows that for every long interval u either $n_\ell(u) < i-1$ or $n_r(u) < i$, which proves that no such long interval w exists.

As heavy tokens cannot use long intervals, each of them has to use base and path intervals forcing it to make at least $2(N-s+1) \geqslant m$ steps. Therefore, we need at least $s \cdot m$ steps in the path.

Summing up all required steps, we conclude, that every path from I to J in $R_k(G_{m,k})$ has length at least $\frac{k^2}{4} \cdot m$. ◀

5 Hardness result for incomparability graphs

In this section, we present a simple reduction showing that Independent Set Reconfiguration is PSPACE-hard on incomparability graphs in general. Note that interval graphs are incomparability graphs of interval orders. The proof exhibits a reduction from H-Word Reachability defined in [10]. For the readers' convenience we state the definition of this problem here. If H is a digraph (possibly with loops) and $a = a_1a_2 \ldots a_n \in V(H)^*$ then a is an *H-word*, if for any $i \in \{1, \ldots, n-1\}$ we have $a_ia_{i+1} \in E(H)$. In the H-Word Reachability we are given two H-words of the same length a and b, and the question is whether one can transform a into b by changing one letter at a time in such a way that each intermediate word is an H-word.

▶ **Theorem 15** ([10], Theorem 3). *There exists a digraph H for which the H-Word Reachability is PSPACE-complete.*

▶ **Theorem 16.** *There exists a constant $w \in \mathbb{N}$, such that Independent Set Reconfiguration is PSPACE-hard on incomparability graphs of posets of width at most w.*

Proof. We demonstrate a reduction from H-Word Reachability for arbitrary H; the result will follow from Theorem 15.

Fix an instance of H-Word Reachability consisting of two H-words a and b of equal length n. We will construct a poset of width at most $2|V(H)|$, and two independent sets A, B in its incomparability graph, such that A is reconfigurable to B if and only if our starting instance is a yes instance of H-Word Reachability. Define the poset $P_n(H)$ as $(V(H) \times \{1, \ldots, n\}, \prec)$ where $\prec$ is defined as follows:

$$(x, i) \prec (y, j) \iff (j = i+1 \text{ and } xy \in E(H)) \text{ or } (j > i+1).$$

By the definition of $\prec$ each set of the form $V(H) \times \{i\}$ is an antichain, thus for any chain C in $P_n(H)$ of cardinality n and any $i \in \{1, \ldots, n\}$, we have $|(V(H) \times \{i\}) \cap C| = 1$. Therefore, any chain C of cardinality n, can be written as $C = \{(x_1, 1), (x_2, 2), \ldots, (x_n, n)\}$. Observe that for each $i \in \{1, \ldots, n-1\}$ we have $(x_i, i) \prec (x_{i+1}, i+1) \iff x_ix_{i+1} \in E(H)$. This implies that the first coordinates $x_1x_2 \ldots x_n$ of the elements of chain C form an H-word. Conversely, given an H-word consisting of n letters $y_1y_2 \ldots y_n$ the set $\{(y_1,1),(y_2,2),\ldots,(y_n,n)\}$ is a chain of cardinality n in $P_n(H)$. It follows that a word $x_1x_2\ldots x_n$ is an H-word if and only if $\{(x_1,1),\ldots,(x_n,n)\}$ is an independent set in the incomparability graph $\mathrm{Inc}(P_n(H))$.

Let $a = a_1a_2\ldots a_n$ and $b = b_1b_2\ldots b_n$ be the two given H-words of length n. We define $A = \{(a_1,1),(a_2,2),\ldots,(a_n,n)\}$ and $B = \{(b_1,1),(b_2,2),\ldots,(b_n,n)\}$. These are two independent sets in $\mathrm{Inc}(P_n)$. Using the fact that for each $i \in \{1,\ldots,n\}$ the set $V(H)\times\{i\}$ is a clique in $\mathrm{Inc}(P_n(H))$, we infer that each edge in $R_n(P_n)$ corresponds to a move of the form $((x,i),(y,i))$ for some $i \in \{1,\ldots,n\}$. Thus A is reconfigurable into B if and only if one can transform a into b one letter at a time keeping each intermediate word an H-word.

All that remains is to observe that we can construct the incomparability graph of $P_n(H)$ together with the sets A and B for a fixed H in logarithmic space, and that the width of $P_n(H)$ is always at most $2|V(H)|$. ◀

References

1 Marthe Bonamy and Nicolas Bousquet. Token sliding on chordal graphs. In *Graph-Theoretic Concepts in Computer Science, Proc. WG 2017*, volume 10520 of *LNCS*, pages 127–139. Springer, 2017.

2 Paul Bonsma, Marcin Kamiński, and Marcin Wrochna. Reconfiguring independent sets in claw-free graphs. In R. Ravi and Inge Li Gørtz, editors, *Algorithm Theory, Proc. SWAT 2014*, volume 8503 of *LNCS*, pages 86–97. Springer, 2014.

3 Nicolas Bousquet. Talk at the graph theory seminar in Bordeaux, October 02, 2020. URL: `https://webconf.u-bordeaux.fr/b/mar-ef4-zed`.

4 Erik D. Demaine, Martin L. Demaine, Eli Fox-Epstein, Duc A. Hoang, Takehiro Ito, Hirotaka Ono, Yota Otachi, Ryuhei Uehara, and Takeshi Yamada. Polynomial-time algorithm for sliding tokens on trees. In Hee-Kap Ahn and Chan-Su Shin, editors, *Algorithms and Computation, Proc. ISAAC 2014*, volume 8889 of *LNCS*, pages 389–400. Springer, 2014.

5 Eli Fox-Epstein, Duc A Hoang, Yota Otachi, and Ryuhei Uehara. Sliding token on bipartite permutation graphs. In *Algorithms and Computation, Proc. ISAAC 2015*, volume 9472 of *LNCS*, pages 237–247. Springer, 2015.

6 Robert A. Hearn and Erik D. Demaine. Pspace-completeness of sliding-block puzzles and other problems through the nondeterministic constraint logic model of computation. *Theoretical Computer Science*, 343(1):72–96, 2005.

7 Marcin Kamiński, Paul Medvedev, and Martin Milanič. Complexity of independent set reconfigurability problems. *Theoretical Computer Science*, 439:9–15, 2012.

8 Daniel Lokshtanov and Amer E. Mouawad. The complexity of independent set reconfiguration on bipartite graphs. *ACM Trans. Algorithms*, 15(1):19 pages, 2018. Article No.: 7.

9 Naomi Nishimura. Introduction to reconfiguration. *Algorithms*, 11(4):52, 2018.

10 Marcin Wrochna. Reconfiguration in bounded bandwidth and tree-depth. *Journal of Computer and System Sciences*, 93:1–10, 2018.