Distributed Reconfiguration of Maximal Independent Sets

Keren Censor-Hillel
Department of Computer Science, Technion, Israel
ckeren@cs.technion.ac.il

Mikaël Rabie
IRIF, Université de Paris, France
Aalto University, Finland
mikael.rabie@irif.fr

─── **Abstract** ───

In this paper, we investigate a *distributed maximal independent set (MIS) reconfiguration problem*, in which there are two maximal independent sets for which every node is given its membership status, and the nodes need to communicate with their neighbors in order to find a *reconfiguration schedule* that switches from the first MIS to the second. Such a schedule is a list of *independent sets* that is restricted by forbidding two neighbors to change their membership status at the same step. In addition, these independent sets should provide some covering guarantee.

We show that obtaining an actual MIS (and even a 3-dominating set) in each intermediate step is impossible. However, we provide efficient solutions when the intermediate sets are only required to be independent and 4-dominating, which is almost always possible, as we fully characterize.

Consequently, our goal is to pin down the tradeoff between the possible length of the schedule and the number of communication rounds. We prove that a constant length schedule can be found in $O(\texttt{MIS} + \texttt{R32})$ rounds, where $\texttt{MIS}$ is the complexity of finding an MIS in a worst-case graph and $\texttt{R32}$ is the complexity of finding a $(3, 2)$-ruling set. For bounded degree graphs, this is $O(\log^* n)$ rounds and we show that it is necessary. On the other extreme, we show that with a constant number of rounds we can find a linear length schedule.

1 Introduction

Consider a distributed setting in which each node of a network receives an input from a higher-level application which tells it whether it is *selected* or not, such that the set of selected nodes is a maximal independent set (MIS), which we will denote by α. The reason that the application requires an MIS is because it needs the set of selected nodes to dominate

all nodes for the sake of, say, monitoring the network, but without having violations of two neighbors being in the set, because they may cause conflicting actions. Now, because of changes in the network traffic, the energy consumption, or any one of various conditions that may change, the application needs to change the selected set of nodes. Once a new input MIS, denoted by β, is given to the nodes by the application, the nodes need to *reconfigure* their states to that set while never sacrificing the safety condition of independence. In fact, for compatibility reasons, neighboring nodes cannot change their membership in the set at the same time, so a *sequence* of changes is needed for converging into the new MIS. We call such a sequence a *reconfiguration schedule.*

The length of the schedule is clearly a measure that is required to be minimized. Hence, an extreme solution would be to have all nodes declare themselves as unselected, and then the new set of nodes declare that they are selected. However, this very fast approach suffers from loosing the domination property throughout the reconfiguration schedule. Thus, the structure of the network must be taken into account, but since the topology is unknown, finding a schedule that maintains a good covering at all times necessitates that the nodes communicate. This brings another measure of complexity into question, which is the number of communication rounds that are needed in order to find a short schedule. Our goal in this work is to study the tradeoff between the possible length of the schedule and the number of communication rounds needed for finding it.

Unfortunately, as we show, it is not always possible to find schedules where each set is an MIS. This impossibility holds even if we relax the condition of domination and require only independent 3-dominating sets. Even when 3-domination *is* possible, it may be extremely inefficient (Section 6).

▶ **Theorem 1.** *Requiring* 3-*domination for intermediate steps is costly:*
1. *There exists a class of inputs $G = (V, E)$ with two MIS α and β such that there is no reconfiguration schedule with 3-dominating intermediate steps.*
2. *There exists a class of inputs $G = (V, E)$ with two MIS α and β such that any reconfiguration schedule is of length $\Omega(n)$ and needs $\Theta(n)$ rounds to be found, if intermediate steps must be 3-dominating.*

However, we prove that independence and 4-domination can indeed be obtained. Our main result is the following (Section 3).

▶ **Theorem 2** (informal). *For any graph $G = (V, E)$ of diameter greater than 3 and any input of two MIS α, β, there exists a reconfiguration schedule of constant length* 28*, with independent 4-dominating intermediate steps. Moreover, such a schedule can be found in $O(\texttt{MIS} + \texttt{R32})$ rounds, where* `MIS` *is the complexity of finding an MIS on a worst-case graph and* `R32` *is the complexity of finding a* $(3, 2)$*-ruling set on a worst-case graph.*

Obtaining the above theorem turns out to be an involved task. Our key ingredients are the following. We prove that graphs with a not-too-small diameter always admit a schedule of reconfiguration steps from a given maximal independent set to another. Moreover, full knowledge of the topology of the graph is not necessary in order to be able to locally add an element to the set after having removed its neighbors (to avoid dependence). Rather, only local manipulations are needed for doing so.

The currently known complexities that give $O(\texttt{MIS} + \texttt{R32})$ are discussed in the related work part. Here, we draw attention to the fact that an immediate corollary of Theorem 2 is that for graphs of bounded degree we can compute the constant length schedule within $O(\log^* n)$ rounds. Further, we show that this is a lower bound by reducing the problem of

finding an MIS on a path to obtaining a constant-length schedule for MIS reconfiguration. The following theorem actually holds even if one requires only d-domination, for some constant $d \geq 4$ (Section 6).

▶ **Theorem 3.** *For any fixed k, there exists a class of k-regular inputs $G = (V, E)$ with two MIS α and β such that any reconfiguration schedule of constant length with 4-domination needs $\Theta(\log^* n)$ rounds to be found.*

If one wants to optimize the communication cost of finding a schedule rather than its length, we show that a (rather lengthy) schedule can be obtained within $O(1)$ rounds (Section 4).

▶ **Theorem 4** (informal). *For any graph $G = (V, E)$ and any input of maximal independent sets α, β to the MIS-reconfiguration problem, there exists a reconfiguration schedule of length $\Theta(f(n))$, where $f(n)$ is the largest identifier among the nodes in the graph, which can be found in $O(1)$ rounds.*

The construction generalizes itself on graphs with a distance-k coloring of c colors, with k big enough. It is possible, from this coloring, to compute a schedule of length $O(c)$ after a constant number of communication rounds. Let Δ be the maximal degree of the graph. A distance-k $O(\Delta^{2k})$ coloring can be found in $O(\log^* n)$ rounds [16], and a distance-k $O(\Delta^k)$ coloring can be found in $O(\log^* n + \sqrt{\Delta^k})$ rounds [4]. Hence, with the same respective communication complexities, we can find schedules of lengths $O(\Delta^{2k})$ and $O(\Delta^k)$.

Finally, as can be inferred from Theorem 2, 4-domination suffices for any graph with diameter greater than 3. For graphs with small diameter, we give an exact characterization of the conditions that allow the existence of a reconfiguration schedule (Section 5). This result implies that our algorithm from Theorem 2, combined with a trivial algorithm that collects the entire graph when the diameter is a small constant, produces an efficient reconfiguration schedule in all cases for which it exists.

1.1 Related work

Distributed Reconfiguration.   Questions of distributed reconfiguration were actually not studied before 2018. Then, Bonamy et al. [6] considered distributed reconfiguration of colorings, with the goal of finding which length of schedule can be computed within a given number of communication rounds. The problem being PSPACE complete in the general case, several subcases were explored. Since finding looser restrictions for the transitions is important for making the problem local instead of having to solve a global PSPACE hard problem, the addition of extra colors in the intermediate colorings was allowed. This aided either having a solution, or finding one quickly.

Distributed Constructions.   Our constructions sometimes make use of two fundamental subroutines, which find an MIS or a $(3, 2)$-ruling set in a graph. An (x, y)-ruling set is a set $S \subseteq V$ in which every two nodes are at distance at least x, and every node that is not in S is within distance at most y from S. Thus, an MIS is a $(2, 1)$-ruling set. Finding an MIS is one of the most fundamental problems in distributed computing. The celebrated randomized $O(\log n)$-round algorithms of Luby [19] and Alon et al. [1] have been recently improved by Ghaffari to $O(\log \Delta + 2^{O(\sqrt{\log \log n})})$ rounds, where Δ is the maximal degree in the network [9]. Deterministic solutions are the classic network-decomposition based algorithm of Panconesi and Srinivasan that runs in $2^{O(\sqrt{\log n})}$ rounds [21], and the $O(\Delta + \log^* n)$-round algorithm of Barenboim et al. [5]. The classic lower bound of Linial [17] shows that $\Omega(\log^* n)$ rounds

are necessary, Kuhn et al. gives a higher bound of $\Omega(\log \Delta / \log \log \Delta, \sqrt{\log n / \log \log n})$ [15]. The latest results of Balliu et al. [3] give the new best known lower bounds to find a MIS: There is no deterministic algorithm in $o(\Delta + \frac{\log n}{\log \log n})$ nor randomized algorithm in $o(\Delta + \frac{\log \log n}{\log \log \log n})$. The Figure 1 in [3] sums up all the results on MIS. A $(3, 2)$-ruling set can be computed by computing an MIS over G^2, and more general ruling sets have been studied in [2, 13, 14, 20, 22].

Centralized Reconfiguration of Maximal Independent Sets. Reconfigurations problems on graphs have been widely studied in the centralized setting during the last decade. An excellent survey on reconfiguration problems can be found in [23]. In the centralized setting, the transition rules are different, requiring that any intermediate set must be at least of a certain size. While having their own motivation in that setting, these rules are not the ones that are needed in the distributed setting, as they do not give covering guarantees (moreover, such properties would be costly to obtain in a distributed setting, due to their global nature).

In more detail, three kinds of transitions have been studied for the independent set reconfiguration problem. *Token Addition and Removal* [11], or $TAR(k)$: at each transition, one vertex is removed from or added to the current independent set, as long as there are at least k nodes in the independent set. *Token Jumping* [12]: at each transition, one vertex is removed from the independent set and another one is added somewhere else. *Token Sliding* [10]: at each transition, an edge containing a vertex of the independent set is chosen. This vertex is removed from the set and its neighbor on the other side of that edge is added to the set. The two first versions are actually equivalent when k corresponds to the size of the independent sets minus 1. Reconfiguration problems are in PSPACE, and independent set reconfiguration problems are in general PSPACE complete [10]. Studies over subclasses of graphs exist, and some polynomial algorithm or hardness proofs are given. For example, planar graphs [10], perfect graphs [12], trees [8] and bipartite graphs [18].

2 Preliminaries

We work in the classic LOCAL model of computation, in which n nodes in a synchronous network exchange messages with their neighbors in each round of computation.

Let $G = (V, E, U)$ denote a graph with an assigned subset $U \subseteq V$. An input to the MIS-reconfiguration problem is a pair $G_{input} = (V, E, \alpha), G_{output} = (V, E, \beta)$, where α and β are the initial and final maximal independent sets, respectively. We refer to a node $v \in \alpha$ as an α-*node*, and to a node $v \in \beta$ as a β-*node*. Notice that a node may be both an α-node and a β-node. We refer to node $v \in V \setminus (\alpha \cup \beta)$ as an ϵ-*node*. Throughout the proofs, we say that a node v is *covered* or *4-dominated* by a node u if $d(v, u) \leq 4$.

For a vertex $v \in V$, we denote by $N(v)$ the set of neighbors of v (i.e., $N(v) = \{u \in V : (u, v) \in E\}$), and given a set $U \subseteq V$ we define $N_U(v) = U \cap N(v)$ for the subset of neighbors of v that are in U, and we call this set the U-neighbors of v. For a subset $U \subseteq Y \subseteq V$ and a node $v \in Y$, we denote by $d_Y(v, U)$ the distance of v from U in the subgraph induced by Y.

▶ **Definition 5** (Reconfiguration Schedules). *For a given property P of $G = (V, E, U)$, an (α, β, P)-reconfiguration schedule (or simply a schedule) S of length ℓ is a sequence of subsets of V, $S = (S_0, \ldots, S_\ell)$, such that the following hold:*
1. $S_0 = \alpha$ and $S_\ell = \beta$,
2. *for every $0 < i < \ell$, the graph (V, E, S_i) satisfies P, and*
3. *for every $0 < i \leq \ell$, $S_i \oplus S_{i-1}$ is an independent set of (V, E).*

3 An MIS reconfiguration schedule of constant length

Our main theorem is the following.

▶ **Theorem 2** (formal). *Let P be the property of (V, E, U) that says that U is a $(2,4)$-ruling set. For any graph $G = (V, E)$ of diameter greater than 3 and any input $G_{input} = (V, E, \alpha), G_{output} = (V, E, \beta)$ to the MIS-reconfiguration problem, there exists an (α, β, P)-reconfiguration schedule of constant length 28. Moreover, such a schedule can be found in $O(\texttt{MIS} + \texttt{R32})$ rounds, where $\texttt{MIS}$ is the complexity of finding an MIS on a worst-case graph and $\texttt{R32}$ is the complexity of finding a $(3, 2)$-ruling set on a worst-case graph.*

In particular, Theorem 2 immediately implies a highly efficient solution for bounded degree graphs.

▶ **Corollary 6.** *The constant length schedule of Theorem 2 can be found in $O(\log^* n)$ rounds in graphs of bounded degree.*

We now describe the outline of the algorithm, as follows. Denote by W the set of connected components of $\alpha \cup \beta$. Our main approach is to reconfigure the independent sets according to the components in W. To this end, we first categorize each component in W according to its diameter and whether it is isolated or not: We say that a component $V_i \in W$ is *isolated* if for every ϵ-node u in its neighborhood, $N_\alpha(u)$ and $N_\beta(u)$ are contained in V_i.

Notice that within a constant number of rounds, all α and β-nodes can know whether they are in a component of diameter 0, 1, 2, or at least 3. Moreover, if their diameter is smaller than 3, they can know whether the component is isolated or not.

To avoid excessive notation, we will sometimes say that *we update the component V_i in steps $\{j, j + 1\}$*. This means that we remove $\alpha \cap V_i$ from the independent set in step j and we add $\beta \cap V_i$ to the set in step $j + 1$. Formally, this means that $S_j = S_{j-1} \setminus (\alpha \cap V_i)$ and $S_{j+1} = S_j \cup (\beta \cap V_i)$. Since we will sometimes update multiple components concurrently, we will have $S_j = S_{j-1} \setminus (\alpha \cap Z_j)$ and $S_{j+1} = S_j \cup (\beta \cap Z_j)$, where $Z_j = \bigcup_{i \in I_j} V_i$, with $I_j = \{i : V_i \text{ is being updated in steps } \{j, j+1\}\}$.

The high-level description of our algorithm is as follows. First, for components in W of diameter 0, we do not need to do anything, as such components are comprised only of nodes in $\alpha \cap \beta$. These nodes remain in the independent set S_i for the entire schedule, and we omit these components and all of their ϵ-neighbors from the remaining discussion. Our algorithm then handles non-isolated components and components of diameter ≥ 3, and finally handles the isolated components of diameter ≤ 2.

We begin by claiming that with an overhead of 2 rounds, we may assume that α and β are disjoint. Indeed, if we never remove nodes in $\alpha \cap \beta$ from the independent set, we no longer need to take care about $\alpha \cap \beta$ nor its neighborhood. You can find a more complete explanation of this in the full version of the paper [7].

3.1 Components of diameter ≥ 3

We continue with the following lemma, which is useful for handling components in W whose diameter is not too small. Roughly speaking, the way we handle components of sufficient diameter is by finding a set of α-nodes that are not too close to each other to ensure that β-nodes can be added not too far from them before we remove them from the independent set. This way, we can reconfigure the rest of the component, and then this set of α-nodes and their neighbors. We present the following lemma before the rest of the algorithm because we will need to use it, but notice that it is not the case that we begin the algorithm by reconfiguring components of diameter ≥ 3.

▶ **Lemma 7.** *Let P be the property of (V, E, U) that says that U is a (2,4)-ruling set, and let $G = (V, E, \alpha)$ and $G = (V, E, \beta)$ be an input to the MIS-reconfiguration problem such that $\alpha \cap \beta = \emptyset$, the set $Y = \alpha \cup \beta$ is a single connected component of diameter at least 3, and each ϵ-node is connected to an α-node and to a β-node. Then, there exists an (α, β, P)-reconfiguration schedule of length 8. Moreover, such a schedule can be found in $O(R32)$ rounds.*

Proof. First, assume that the diameter of $Y = \alpha \cup \beta$ is either 3 or 4. Consider a shortest path of length 3 in Y, denoted by (v_1, v_2, v_3, v_4). Either v_1 or v_4 is in α, and is within distance 4 from all other nodes in the component. We denote this node by v, and define $S_1 = \{v\}$ and we have that it 4-dominates the entire component. In addition, it 4-dominates all ϵ-nodes, by the assumption of the lemma that all such nodes are neighbors of β-nodes, because v actually 3-dominates all β-nodes in the component. We then denote $R = \{u \in \beta \mid u \notin N(v)\}$ and define $S_2 = \{v\} \cup R$, and $S_3 = R$, and finally $S_4 = R \cup N(v)$. It is easy to verify that this results in a valid (α, β, P)-reconfiguration schedule. In particular, notice that, without loss of generality, if $v = v_1$, then R contains at least the node v_4, which 4-dominates the entire component.

For a diameter of Y that is at least 5, the high-level idea of the construction is as follows. Consider a $(3, 2)$-ruling set R over the nodes in α, where we imagine an edge between two nodes in α if they are at distance two in the subgraph induced by Y. We reconfigure all β-nodes that are at distance 5 from R in G by removing their α-neighbors first, then by adding them. Then, we do the same for β-nodes that are at distance 3 from R, and finally we repeat this one last time for the β-nodes in the direct neighborhood of R. The choice of a $(3, 2)$-ruling set ensures that all α-nodes in R have a β-node at distance 3 that will be reconfigured in the 4th step. However, while we trust β-nodes at distance 3 from R to cover α-nodes at distance 2 from R while R itself is being reconfigured, we must be careful when handling α-nodes at distance 2 from R that do not have a neighbor at distance 3. We overcome this caveat by taking care of these nodes separately.

Formally, we define a virtual multigraph $\tilde{G} = (\tilde{V}, \tilde{E})$ as follows. The set of virtual nodes $\tilde{V}$ consists of all α-nodes. If v and u in $\tilde{V}$ have a common β-neighbor, we add an edge u, v to $\tilde{E}$. Let R be a $(3, 2)$-ruling set in $\tilde{G}$. It is easy to see that in G, the set of nodes R is a $(6, 5)$-ruling set of Y. We denote R by R_0 and we define R_i for $3 \le i \le 5$ as $R_i = \{v \in Y \mid$ the distance of v from R in the subgraph induced by Y is $i\}$. Then we define $R_1 = \{v \in Y \mid d_Y(v, R) = 1$ and $d_Y(v, R_3) = 2\} \cup \{v \in Y \mid N_Y(v) \subseteq R\}$, which captures all β-neighbors of R that either do not have other α-neighbors, or have other α-neighbors which in turn have β-neighbors that are farther from R. We separate those from the set $R_{-1} = N_Y(R_0) \setminus R_1$. We complete the partition by defining $R_2 = N_Y(R_1) \setminus R_0$ and $R_{-2} = N_Y(R_{-1}) \setminus R_0$. Note that for even i, R_i contains only α-nodes, and for odd i, R_i contains only β-nodes. We have that, for every $-2 \le i \le 5$, $N_Y(R_i) \subseteq R_{i-1} \cup R_{i+1}$ (with $R_{-3} = R_6 = \emptyset$). By construction of R, we have that each node in R has a node at distance 3 in R_3, hence it has a node at distance 2 in R_2 and a node at distance 1 in R_1.

We define $S_0 = \alpha$ and for $i = 0, 1, 2, 3$, we define $S_{2i+1} = S_{2i} \setminus R_{4-2i}$, $S_{2(i+1)} = S_{2i+1} \cup R_{5-2i}$. We claim that $S_0, \ldots, S_8$ is an (α, β, P)-reconfiguration schedule.

First, $S_0 = \alpha$ by definition, and because every β-node is within an odd distance of at most 5 from R and every α-node is within an even distance of at most 4 from R, we have that $S_8 = \beta$. This gives condition (1) of Definition 5.

For condition (2), it is easy to see that $S_i \oplus S_{i-1}$ is an independent set of (V, E) for every $1 \le i \le 8$. For an odd i this holds because to obtain S_i we only remove α-nodes from S_{i-1}, and no two such nodes can be neighbors. For an even i this holds because to obtain S_i we only add β-nodes to S_{i-1}, and no two such nodes can be neighbors.

It remains to show condition (3) of Definition 5. To show that S_i is independent for $i = 2, 4, 6$, notice that β-nodes in R_j (for $j = -1, 1, 3, 5$) are only added to the sequence after all α-nodes in R_k for $k \geq j - 1$ have been removed. By definition, S_0 is also independent. Hence, for $i = 1, 3, 5, 7$, S_i is independent because it is a subset of S_{i-1}.

Next, we need to show that S_i is 4-dominating for every $1 \leq i \leq 7$. Our focus will be for $i = 1, 3, 5, 7$, and for $i = 2, 4, 6$ it then follows because S_i contains S_{i-1}. We first show it on Y, and will prove it for ϵ-nodes afterward. For $i = 1$ this holds because all nodes in R_j for $j \leq 3$ are in or have neighbors in $R_{-2} \cup R_0 \cup R_2$. All nodes in R_j for $j = 4, 5$ are within distance 3 from R_2. Similarly, S_3 is 4-dominating because nodes in R_j for $j \leq 4$ are covered by R_0, nodes in R_5 are in the current independent set. For S_5, recall that for any node in R, there is a node at distance 3 from it in R_3, that node being in the current independent set since S_4. Hence, R_3 covers R_j for $-1 \leq j \leq 5$. R_{-2} is still included in S_5. Finally, for S_7, R_3 still covers R_j for $-1 \leq j \leq 5$. For each node in R_{-2}, there is a node at distance 3 from it in R_1 that has been added in S_6 that covers it.

Now, let u be an ϵ-node that has a node $a \in \alpha$ and $b \in \beta$ in its neighborhood. We show that a or b are always 3-dominated throughout the sequence. In a step where b has no α-neighbor in the independent set, it must be a step right before b gets added to the independent set. If b is in R_5 or in R_3 then when this happens, it is 3-dominated by an α-node in R_2 or R, respectively, and this node is still in the independent set. If $b \in R_{-1}$ then it is 2-dominated by nodes in R_{-2} and then R_1 (with an overlap in S_6, the construction ensures that such node exist at distance at most 3 from b). Finally, If $b \in R_1$ then either there is a β-node in R_3 that 2-dominates it, and this node is already in the independent set, or b is in $\{v \in Y \mid N_Y(v) \subseteq R\}$. Only in the latter case, we must resort to the α-neighbor of u and check that it is 3-dominated by S_5, as we removed R from S_5 and b is added in S_6.

Let i be such that $a \in R_i$. We need to make sure that a is 3-dominated at the step in which we reconfigure R_1. At this step, all of the β-nodes in R_3, R_5 are in the independent set, and hence their α-neighbors in R_2, R_4 are covered by nodes in distance 1, and nodes in R_0 are covered by nodes in distance 3. For α-nodes in R_{-2} they are still in the independent set at this step, and hence are 3-dominated.

This completes the correctness proof. For the round complexity, notice that simulating the $(3, 2)$-ruling set over $\tilde{G}$ can be done in G with a constant overhead. ◄

3.2   Non-isolated components

We first observe that components of diameter ≤ 2 are such that there is a complete bipartite graph between their α-nodes and β-nodes. Let u be an ϵ-node that is a neighbor of several components. Let W_u be the set of all components that are its neighbors, so that in particular, $V_i, V_j \in W_u$. For each pair of distinct components $V_i, V_j \in W_u$, if there is an α node in $N_\alpha(u) \cap V_i$ and a β node in $N_\beta(u) \cap V_j$, then we say that V_j is (u, α)-covered and that V_i is (u, β)-covered (note that this definition allows a single component to satisfy both conditions). As u is an ϵ-node, there must exist a component $V_{u,\alpha} \in W_u$ that is (u, β)-covered and a component $V_{u,\beta} \in W_u$ that is (u, α)-covered.

We say that a component $V_i \in W$ is α-covered (β-covered) if there is an ϵ-node u for which V_i is (u, α)-covered ((u, β)-covered). A component that is both is $\alpha\beta$-covered.

The key insight is that a (u, α)-covered component of diameter ≤ 2 is covered (dominated at distance 4) by some α-node of the component $V_{u,\beta}$ (and similarly with the β-node of $V_{u,\alpha}$). Moreover, any ϵ-node that is connected to an α-node (a β-node) in that component is covered by $V_{u,\beta}$ (or $V_{u,\alpha}$). This implies that an ϵ-node that is connected to two components that are updated in different steps is always covered by the component that is currently not

being updated. However, during the reconfiguration schedule, we need to be careful about ϵ-nodes that are connected to a single component, and ϵ-nodes that are connected to two components that are updated at the same time.

We denote by $C_{\alpha\beta}$ the set of $\alpha\beta$-covered components of diameter ≤ 2, and by C_α and C_β the sets of α-covered and β-covered components of diameter ≤ 2 that are not in $C_{\alpha\beta}$, respectively. Define the component graph $\tilde{G} = (W, \tilde{E})$, where there is an edge between $V_i, V_j \in W$ iff there exists an ϵ-node u such that V_i is (u, α)-covered and V_j is (u, β)-covered, or vice-versa. Notice that in $\tilde{G}$, the sets C_α and C_β are two disjoint independent sets.

We are finally ready to formally provide the algorithm for handling all components that are either non-isolated or have diameter ≥ 3.

▶ **Lemma 8.** *Let P be the property of (V, E, U) that says that U is a (2,4)-ruling set, and let $G = (V, E, \alpha)$ and $G = (V, E, \beta)$ be an input to the MIS-reconfiguration problem such that $\alpha \cap \beta = \emptyset$, and all connected components of $\alpha \cup \beta$ are either non-isolated or have diameter at least 3. Then, there exists an (α, β, P)-reconfiguration schedule of length 18. Moreover, such a schedule can be found in $O(\texttt{MIS} + \texttt{R32})$ rounds.*

Proof. Our reconfiguration schedule works according to the following *parts*.
1. Update components of diameter ≤ 2 in C_α.
   %Let M be an MIS over all nodes in $C_{\alpha\beta}$.
2. Update components of diameter ≤ 2 that are α-covered by a component in M.
3. Reconfigure components of diameter ≥ 3 using the schedule given by Lemma 7.
4. Update components in M.
5. Update components of diameter ≤ 2 in $C_{\alpha\beta}$ that were not previously updated.
6. Update components of diameter ≤ 2 in C_β that were not previously updated.

First, it is easy to see that the schedule has length 18. The part that reconfigures components of diameter ≥ 3 requires 8 steps, by Lemma 7. Each of the other 5 parts takes exactly 2 steps as described in the definition of updating components (removing α-nodes and then adding β-nodes), which sums to 18 reconfiguration steps in the schedule.

It remains to prove correctness. First, condition (1) of Definition 5 trivially holds, as the schedule reconfigures all nodes. Moreover, by Lemma 7 and by the definition of updating a component, it is also immediate that we do not reconfigure two neighbors in a single step, thus the schedule satisfies condition (3) of Definition 5. For condition (2), Lemma 7 and the definition of updating a component also guarantee that each S_i is an independent set. The remainder of the proof shows that each S_i in the schedule is also 4-dominating.

By the order of the reconfiguration steps in the schedule, each component that is being updated is covered by a component that is not concurrently being updated. This also holds for ϵ-nodes that are connected to a component that is not currently being updated. The main condition that must be verified is that ϵ-nodes remain covered even if all of their neighboring components are being concurrently updated in a certain part of the schedule.

Part 1 guarantees that S_1, S_2 are 4-dominating because for each component that is being updated, the α-node covering it is a member of S_1, S_2 and it also covers the required ϵ-nodes that are neighbors of the updated component, as explained earlier. For part 2, let u be an ϵ-node that is connected to two of the components that are being updated and is not connected to any component that is not being updated. One of the components must be connected to u via an α-node. Let V_i be such a component, let u_1 be the α-node connected to u, and let v be the α-node from a component of M that covers V_i. The distance between v and u_1 is 3: v is a distance 2 to a β-node of V_i and, because V_i is of diameter ≤ 2, within V_i all β-nodes are connected to all α-nodes. Hence, u is at distance 4 from v.

For part 3, the 4-domination within the components that are being reconfigured is given by Lemma 7. Notice that any ϵ-node connected to a component of diameter ≥ 3 is connected either to connected components of diameter ≥ 3 through both an α and a β-node, or to a component that is not being updated in those steps. In the first case it is covered by Lemma 7, and in the second it is covered by the component not being concurrently updated.

For part 4, notice that all components that are β-covering components of M have been updated in steps 2 or 3. Hence, as each component of M is in $C_{\alpha\beta}$, there is a β-node in the current set S_{12} that covers it. As M is independent, we do not have ϵ-node between two components that are being updated. For part 5, the ϵ-nodes between two components that are being updated are covered by an argument that is symmetric to the one used for part 2. Finally, for part 6, for each component that is being updated it holds that the β-node covering it is in a component that has already been updated and hence it is already in S_{16}.

Finally, we note that apart from a constant overhead in communication, the number of rounds required for computing the above schedule is proportional to that of finding the MIS M plus solving the diameter ≥ 3 components, which completes in $O(\texttt{MIS} + \texttt{R32})$ rounds, where $\texttt{MIS}$ is the complexity of finding an MIS on a worst-case graph and and $\texttt{R32}$ is the complexity of finding a $(3, 2)$-ruling set on a worst-case graph, as claimed. ◀

3.3 Isolated Components

What remains now is to handle components that are isolated and have diameter at most 2. When we address these components, we will also address all of their ϵ-neighbors. Hence, from this point onwards we will slightly abuse our terminology, and when we refer to such a component we refer to its nodes along with their ϵ-neighbors as the component. This means that now the components that we address might have a diameter that is increased by 1, and thus their diameter can be also 3. Note that the diameter cannot be increased by two as all α-nodes are connected to all β-nodes, and each ϵ-node is connected to an α-node and to a β-node of this component, otherwise the component would not be isolated.

By definition of isolated components, the neighborhood of an ϵ-nodes within such a component, besides containing vertices of the component itself, is only composed of other ϵ-nodes. Moreover, there is at least one additional ϵ-node in this neighborhood, as we consider graphs of diameter at least 4. We distinguish two kinds of isolated components, according to whether their diameter is at most 2, or whether it is 3.

For a component V_i of diameter ≤ 2, suppose u is an ϵ-node that is a neighbor of V_i. This node u has an α-node and a β-node in its neighborhood, that both cover the entire component. Therefore, to update such components, it suffices to make sure that a non-ϵ neighbor of u is in the current independent set during the two reconfiguration steps. By considering connected two of those components that cover each other, we can take an MIS M over those. The schedule of length 4 is: update M, and then update the other components.

Assume now that V_i is a component of diameter 3. It holds that there exists an ϵ-node u, an α-node a and a β-node b such that $(u, a) \notin E$ and $(u, b) \notin E$ (otherwise the diameter would be 2). Here is an informal description of a schedule of 6-steps for this component.

1. Remove $N_\alpha(u)$. The node a stays in the independent set and covers the entire component.
2. Add u in the set.
3. Remove the remaining α-nodes of the component. The node u covers everything.
4. Add b to the set. Note that b covers the component.
5. Remove u.
6. Add the remaining β-nodes of the component.

A caveat is encountered in case there are two such components, V_1 and V_2, whose selected ϵ-nodes, u_1 and u_2, are connected. In such case we cannot do the above 6-step schedule in parallel without violating independence. However, observe that if a single of those two ϵ-nodes is added to the set, it actually covers the second component as well, as it has a diameter of 3. As a consequence, taking an MIS over those ϵ-nodes gives us a selection of nodes that cover all the considered components. Hence, consider the schedule above as being for component V_1 and denote $u = u_1$, then we can add the following to steps 3 and 4 above:

3. Remove the remaining α-nodes of V_1 and all α-nodes of V_2. The node u covers everything.

4. Add b and the β-nodes of V_2 to the set. Note that V_2 is updated and b covers V_1.

We now formalize the above intuition in order to prove the following.[1]

▶ **Lemma 9.** *Let P be the property of (V, E, U) that says that U is a (2,4)-ruling set, and let $G = (V, E, \alpha)$ and $G = (V, E, \beta)$ be an input to the MIS-reconfiguration problem such that $\alpha \cap \beta = \emptyset$, and all connected components of $\alpha \cup \beta$ are isolated and have diameter at most 2. Then, there exists an (α, β, P)-reconfiguration schedule of length 10. Moreover, such a schedule can be found in $O(\mathtt{MIS})$ rounds.*

3.4 Completing the proof

We can now wrap-up all the ingredients and prove Theorem 2.

Proof of Theorem 2. We describe the full (α, β, P)-reconfiguration schedule S. First, each node v in $V_{\alpha,\beta} = \alpha \cap \beta$ sends a message to its neighbors in $N(v)$ and outputs that it is in S_i for all $0 \leq i \leq 28$. Each node that received such a message, sends a message to its neighbors and outputs that it is not in S_i for all $0 \leq i \leq 28$. The nodes that produced an output terminate and any edges incident to them are removed from the graph.

Next, all nodes collect their 4-hop neighborhood to decide whether they are in a component of diameter ≥ 3 or not, and if not then whether they are in an isolated component.

The components of diameter ≥ 3 and the non-isolated components compute the reconfiguration schedule of 18 steps, as given in Lemma 8, which we denote by $S'_0, \ldots, S'_{18}$. The isolated components of diameter ≤ 3 compute the reconfiguration schedule of 10 steps, as given in Lemma 9, which we denote by $S''_0, \ldots, S''_{10}$.

Formally, the (α, β, P)-reconfiguration schedule is now $S_i = S''_0 \cup S'_i \cup V_{\alpha,\beta}$ for $0 \leq i \leq 18$ and $S_i = S''_{i-18} \cup S'_{18} \cup V_{\alpha,\beta}$ for $18 \leq i \leq 28$. It is computed within $O(\mathtt{MIS} + \mathtt{R32})$ rounds. ◀

4 MIS reconfiguration in a constant number of rounds

▶ **Theorem 4** (formal). *Let P be the property of (V, E, U) that says that U is a (2,4)-ruling set. For any graph $G = (V, E)$ and any input $G_{input} = (V, E, \alpha), G_{output} = (V, E, \beta)$ to the MIS-reconfiguration problem, there exists an (α, β, P)-reconfiguration schedule of length $\Theta(f(n))$, where $f(n)$ is the largest identifier among the nodes in the graph, which can be found in $O(1)$ rounds.*

To prove this, we first prove the following lemma, stating that we can always reconfigure locally an independent set to add elements from β without losing any element in $\alpha \cap \beta$.

---

[1] You can find all the missing proofs in the full version of the paper [7].

▶ **Lemma 10.** *Let P be the property of (V, E, U) that says that U is a (2,4)-ruling set. For any graph $G = (V, E)$ of diameter greater than 5, two MIS α and β and $v \in \beta \setminus \alpha$, there exists an MIS γ such that*

1. $(\alpha \cap \beta) \subset \gamma$ *and* $v \in \gamma$, *and*
2. *there exists an (α, γ, P)-reconfiguration schedule of length 6. Moreover, for finding the reconfiguration schedule the nodes only need to know the topology of their 5-hop neighborhood and therefore can be found in $O(1)$ rounds.*

Lemma 10, means that for any element v in β, we can add v to the current MIS in a constant number of steps without losing any element of β already in the MIS. It allows us to prove Theorem 4 as follows.

Proof of Theorem 4. Nodes use their identifiers to know when to start their own reconfiguration. A node with identifier k uses slots $[6k + 1, 6(k + 1)]$ for its schedule. Since a node only needs to know its 5-hop neighborhood, this completes in $O(1)$ rounds. ◀

If the identifiers are guaranteed to be $\{1, \ldots, n\}$ then Theorem 4 gives that a constant number of rounds is sufficient for a linear length schedule. However, we can do even better by using coloring algorithms, as stated in the following corollary.

▶ **Corollary 11.** *Let P be the property of (V, E, U) that says that U is a (2,4)-ruling set. For any graph $G = (V, E)$ and any input $G_{input} = (V, E, \alpha), G_{output} = (V, E, \beta)$ to the MIS-reconfiguration problem, if the nodes are given a k-coloring of G^{10}, then there exists an (α, β, P)-reconfiguration schedule of length $O(k)$, which can be found in $O(1)$ rounds.*

5 A complete characterization for the existence of a reconfiguration schedule with 4-domination

The following gives an exact characterization of inputs for which there exists a reconfiguration schedule with 4-domination. In what follows, we say that two sets of nodes U_1 and U_2 are *fully connected* if every node in U_1 is a neighbor of every node in U_2. If U_1 contains only a single node, then we simply say that this node is fully connected to U_2.

▶ **Theorem 12.** *Let P be the property of (V, E, U) that says that U is a (2,4)-ruling set. For any input $G_{input} = (V, E, \alpha), G_{output} = (V, E, \beta)$ to the MIS-reconfiguration problem, there does not exists an (α, β, P)-reconfiguration schedule if and only if:*

1. *The sets α and β are fully connected.*
2. *Let ϵ_α (resp. ϵ_β) be the set of ϵ-nodes that are fully connected to α (resp. β). Then all the ϵ-nodes are in $\epsilon_\alpha \cup \epsilon_\beta$.*
3. *Let $G' = (V' = \epsilon_\alpha \cup \epsilon_\beta, E' = \overline{E_{V'}})$, where $\overline{E_{V'}}$ is the complementary of E restricted to vertices of V'. Then there is no path from $\epsilon_\alpha \setminus \epsilon_\beta$ to $\epsilon_\beta \setminus \epsilon_\alpha$ in G'.*

6 Impossibility results for MIS reconfiguration

We show here two types of impossibility results. One is the necessity of 4-domination in the sense that 3-domination cannot be obtained, and the other is the necessity of $\Omega(\log^* n)$ rounds with 4-domination, even on bounded degree graphs where it matches the complexity we provide in Corollary 6.

Impossibility of MIS reconfiguration with 3-domination.

▶ **Theorem 1.** *Requiring* 3-*domination for intermediate steps is costly:*

1. *There exists a class of inputs $G = (V, E)$ with two MIS α and β such that there is no reconfiguration schedule with 3-dominating intermediate steps.*

2. *There exists a class of inputs $G = (V, E)$ with two MIS α and β such that any reconfiguration schedule is of length $\Omega(n)$ and needs $\Theta(n)$ rounds to be found, if intermediate steps must be 3-dominating.*
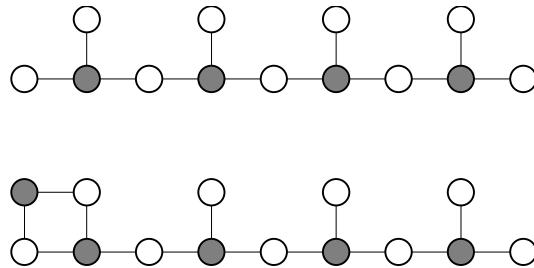
Figure 1 White nodes are α-nodes, and grey nodes are β-nodes. For the graph on the top, there is no schedule with 3-dominating sets. For the graph on the bottom, any schedule with 3-dominating sets must be of linear length and requires a linear number of rounds to be found.

An $\Omega(\log^* n)$ lower bound for MIS reconfiguration with 4-domination.

▶ **Theorem 3.** *For any fixed k, there exists a class of k-regular inputs $G = (V, E)$ with two MIS α and β such that any reconfiguration schedule of constant length with 4-domination needs $\Theta(\log^* n)$ rounds to be found.*
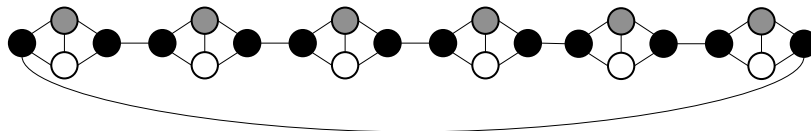
Figure 2 White nodes are α-nodes, grey nodes are β-nodes, and black nodes are ϵ-nodes. Here, $\Omega(\log^* n)$ rounds are needed with 4-domination.

7 Discussion and Open Questions

This paper defines relevant constraints for finding a reconfiguration schedule of maximal independent sets in a distributed setting. For constant-length schedules in bounded-degree graphs we completely settle the required complexity, as we provide an algorithm completing in $\Theta(\log^* n)$ communication rounds, and prove that no lower complexity exists. A main open question that remains is: Can a better complexity be found for general graphs?

Our definition only uses addition and removal of elements to the intermediate independent sets. We propose the following question: Can an efficient distributed reconfiguration schedule be found if the system allows that intermediate steps are 3-dominating and the transitions used can be any combination of addition, removal and Token Sliding?

Finally, we used as a hypothesis that the given independent sets are maximal. Our algorithm still works when the sets are not maximal, as it suffices to complete those. For example, if we are given (2,4)-ruling sets (which is equivalent to the 4-domination condition of P), the problem is solved. An interesting question could be to generalize for other (a, b)-ruling sets. What relation between a and b is needed to ensure that a schedule exists, and that it can be found efficiently with a distributed algorithm?

─────── **References** ───────

1   Noga Alon, László Babai, and Alon Itai. A Fast and Simple Randomized Parallel Algorithm for the Maximal Independent Set Problem. *J. Algorithms*, 7(4):567–583, 1986. `doi:10.1016/0196-6774(86)90019-2`.

2   Baruch Awerbuch, Andrew V. Goldberg, Michael Luby, and Serge A. Plotkin. Network Decomposition and Locality in Distributed Computation. In *30th Annual Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, USA, 30 October - 1 November 1989*, pages 364–369, 1989. `doi:10.1109/SFCS.1989.63504`.

3   Alkida Balliu, Sebastian Brandt, Juho Hirvonen, Dennis Olivetti, Mikaël Rabie, and Jukka Suomela. Lower bounds for maximal matchings and maximal independent sets. *arXiv preprint*, 2019. `arXiv:1901.02441`.

4   Leonid Barenboim, Michael Elkin, and Uri Goldenberg. Locally-Iterative Distributed ($\Delta + 1$): -Coloring below Szegedy-Vishwanathan Barrier, and Applications to Self-Stabilization and to Restricted-Bandwidth Models. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing, PODC 2018, Egham, United Kingdom, July 23-27, 2018*, pages 437–446, 2018. `doi:10.1145/3212734.3212769`.

5   Leonid Barenboim, Michael Elkin, and Fabian Kuhn. Distributed (Delta+1)-Coloring in Linear (in Delta) Time. *SIAM J. Comput.*, 43(1):72–95, 2014. `doi:10.1137/12088848X`.

6   Marthe Bonamy, Paul Ouvrard, Mikaël Rabie, Jukka Suomela, and Jara Uitto. Distributed Recoloring. In Ulrich Schmid and Josef Widder, editors, *32nd International Symposium on Distributed Computing (DISC 2018)*, volume 121 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 12:1–12:17, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. `doi:10.4230/LIPIcs.DISC.2018.12`.

7   Keren Censor-Hillel and Mikaël Rabie. Distributed Reconfiguration of Maximal Independent Sets. *arXiv preprint*, 2018. `arXiv:1810.02106`.

8   Erik D Demaine, Martin L Demaine, Eli Fox-Epstein, Duc A Hoang, Takehiro Ito, Hirotaka Ono, Yota Otachi, Ryuhei Uehara, and Takeshi Yamada. Linear-time algorithm for sliding tokens on trees. *Theoretical Computer Science*, 600:132–142, 2015.

9   Mohsen Ghaffari. An Improved Distributed Algorithm for Maximal Independent Set. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 270–277, 2016. `doi:10.1137/1.9781611974331.ch20`.

10   Robert A Hearn and Erik D Demaine. PSPACE-completeness of sliding-block puzzles and other problems through the nondeterministic constraint logic model of computation. *arXiv preprint*, 2002. `arXiv:cs/0205005`.

11   Takehiro Ito, Erik D Demaine, Nicholas JA Harvey, Christos H Papadimitriou, Martha Sideri, Ryuhei Uehara, and Yushi Uno. On the complexity of reconfiguration problems. *Theoretical Computer Science*, 412(12-14):1054–1065, 2011.

12   Marcin Kamiński, Paul Medvedev, and Martin Milanič. Complexity of independent set reconfigurability problems. *Theoretical computer science*, 439:9–15, 2012.

13   Kishore Kothapalli and Sriram V. Pemmaraju. Super-Fast 3-Ruling Sets. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2012, December 15-17, 2012, Hyderabad, India*, pages 136–147, 2012. `doi:10.4230/LIPIcs.FSTTCS.2012.136`.

14 Fabian Kuhn, Yannic Maus, and Simon Weidner. Deterministic Distributed Ruling Sets of Line Graphs. *CoRR*, abs/1805.07209, 2018. `arXiv:1805.07209`.

15 Fabian Kuhn, Thomas Moscibroda, and Roger Wattenhofer. Local Computation: Lower and Upper Bounds. *J. ACM*, 63(2):17:1–17:44, 2016. `doi:10.1145/2742012`.

16 Nathan Linial. Distributive Graph Algorithms Global Solutions from Local Data. In *Proceedings of the 28th Annual Symposium on Foundations of Computer Science*, SFCS '87, pages 331–335, Washington, DC, USA, 1987. IEEE Computer Society. `doi:10.1109/SFCS.1987.20`.

17 Nathan Linial. Locality in Distributed Graph Algorithms. *SIAM J. Comput.*, 21(1):193–201, 1992. `doi:10.1137/0221015`.

18 Daniel Lokshtanov and Amer E Mouawad. The complexity of independent set reconfiguration on bipartite graphs. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 185–195. SIAM, 2018.

19 Michael Luby. A Simple Parallel Algorithm for the Maximal Independent Set Problem. *SIAM J. Comput.*, 15(4):1036–1053, 1986. `doi:10.1137/0215074`.

20 Shreyas Pai, Gopal Pandurangan, Sriram V. Pemmaraju, Talal Riaz, and Peter Robinson. Symmetry Breaking in the Congest Model: Time- and Message-Efficient Algorithms for Ruling Sets. In *31st International Symposium on Distributed Computing, DISC 2017, October 16-20, 2017, Vienna, Austria*, pages 38:1–38:16, 2017. `doi:10.4230/LIPIcs.DISC.2017.38`.

21 Alessandro Panconesi and Aravind Srinivasan. On the Complexity of Distributed Network Decomposition. *J. Algorithms*, 20(2):356–374, 1996. `doi:10.1006/jagm.1996.0017`.

22 Johannes Schneider, Michael Elkin, and Roger Wattenhofer. Symmetry breaking depending on the chromatic number or the neighborhood growth. *Theor. Comput. Sci.*, 509:40–50, 2013. `doi:10.1016/j.tcs.2012.09.004`.

23 Jan van den Heuvel. The complexity of change. *Surveys in Combinatorics*, 409(2013):127–160, 2013.