On the Parameterized Complexity of Maximum Degree Contraction Problem

Saket Saurabh
The Institute Of Mathematical Sciences, HBNI, Chennai, India
University of Bergen, Bergen, Norway
saket@imsc.res.in

Prafullkumar Tale
CISPA Helmholtz Center for Information Security, Saarbrücken, Germany
prafullkumar.tale@cispa.saarland

Abstract

In the Maximum Degree Contraction problem, input is a graph G on n vertices, and integers k, d, and the objective is to check whether G can be transformed into a graph of maximum degree at most d, using at most k edge contractions. A simple brute-force algorithm that checks all possible sets of edges for a solution runs in time $n^{\mathcal{O}(k)}$. As our first result, we prove that this algorithm is asymptotically optimal, upto constants in the exponents, under Exponential Time Hypothesis (ETH).

Belmonte, Golovach, van't Hof, and Paulusma studied the problem in the realm of Parameterized Complexity and proved, among other things, that it admits an FPT algorithm running in time $(d + k)^{2k} \cdot n^{\mathcal{O}(1)} = 2^{\mathcal{O}(k \log(k+d))} \cdot n^{\mathcal{O}(1)}$, and remains NP-hard for every constant $d \geq 2$ (Acta Informatica (2014)). We present a different FPT algorithm that runs in time $2^{\mathcal{O}(dk)} \cdot n^{\mathcal{O}(1)}$. In particular, our algorithm runs in time $2^{\mathcal{O}(k)} \cdot n^{\mathcal{O}(1)}$, for every fixed d. In the same article, the authors asked whether the problem admits a polynomial kernel, when parameterized by $k + d$. We answer this question in the negative and prove that it does not admit a polynomial compression unless NP $\subseteq$ coNP$/poly$.

2012 ACM Subject Classification Theory of computation → Fixed parameter tractability

Keywords and phrases Graph Contraction Problems, FPT Algorithm, Lower Bound, ETH, No Polynomial Kernel

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

Funding *Saket Saurabh*: This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 819416), and Swarnajayanti Fellowship (No DST/SJF/MSA01/2017-18).
Prafullkumar Tale: This research is a part of a project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement SYSTEMATICGRAPH (No. 725978).

Acknowledgements Some part of this project was completed when the second author was a Postdoctoral Fellow at Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany.

1 Introduction

For any graph class $\mathcal{H}$, the $\mathcal{H}$-Modification problem takes as input a graph G and an integer k, and asks whether one can make at most k modifications in G such that the resulting graph is in $\mathcal{H}$. These types of modification problems are one of the central problems in graph theory and have received a considerable attention in algorithm design. With appropriate choice of $\mathcal{H}$ and allowed modification operations, $\mathcal{H}$-Modification can encapsulate well studied problems like Vertex Cover, Chordal Completion, Cluster Editing, Hadwinger Number, etc. Some natural and well-studied graph modification operations are vertex deletion, edge deletion, edge addition, and edge contraction. The focus of the vast majority of papers on graph modification problems has been to the first three operations. Consider an example of $\mathcal{H}_{\leq d}$-Modification problem where $\mathcal{H}_{\leq d}$ is the collection of all graphs that has maximum degree at most d. If allowed modification operation is vertex deletion then we know the problem as Bounded Degree Deletion (BDD) and if it is edge contraction then as Maximum Degree Contraction (MDC). The complexity of BDD and several of its variants has been extensively studied [7, 9, 10, 13, 15, 17, 21, 28, 35] whereas, to the best of our knowledge, only [8] addressed MDC. In this article, we enhance our understanding of the second problem and answer an open question stated in [8].

The *contraction* of edge uv in simple graph G deletes vertices u and v from G, and replaces them by a new vertex, which is made adjacent to vertices that were adjacent to either u or v. For a set of edges F in $E(G)$, we denote the graph obtained from G by contracting all edges in F by G/F. In the $\mathcal{H}$-Contraction problem, an input is a graph G and an integer k, and the aim is to decide whether there is a set F of at most k edges in G such that G/F is in $\mathcal{H}$. Early papers by Watanabe et al. [37, 38] and Asano and Hirata [6] showed that $\mathcal{H}$-Contraction is NP-Hard for simple graph classes like trees, paths, stars, etc. Brouwer proved that it is NP-Hard even to decide whether a graph can be contracted to a path of length four [11]. Note that this problem admits a simple polynomial time algorithm if we consider any other modification operation. This has been a recurring theme in graph modification problems. For the same target graph class, edge contraction problem tends to more difficult than their counterparts where modification operation is vertex/edge addition/deletion. This difficulty is evident even in the realm of the Parameterized Complexity and Exact Exponential Algorithms.

In Parameterized Complexity, $\mathcal{H}$-Contraction problems are studied with the number of edges allowed to contract, k, as parameter. Heggernes et al. [27] proved that if $\mathcal{H}$ is the set of acyclic graphs then $\mathcal{H}$-Contraction is FPT but does not admit a polynomial kernel unless $\mathsf{NP} \subseteq \mathsf{coNP}/poly$. The vertex deletion version of the problem, known as Feedback Vertex Set, admits a polynomial kernel. Series of papers studied the parameterized complexity for various graph classes like generalization and restrictions of trees [1, 3], cactus [29], bipartite graphs [24, 26], planar graphs [23], grids [36], cliques [12], split graphs [4], chordal graphs [32], bi-cliques [33], degree constrained graph classes [8, 22], etc. Krithika et al. [30] and Gunda et al. [25] studied $\mathcal{H}$-Contraction problems from the lenses of FPT approximation and lossy kernelization. Agarwal et al. [2] broke the 2^n-barrier for Path Contraction whereas Fomin et al. [19] showed that brute-force algorithms for Hadwinger Number problem and various other $\mathcal{H}$-Contraction problem are optimal under ETH.

Belmonte et al. [8] studied the parameterized complexity of $\mathcal{H}$-Contraction for three different classes $\mathcal{H}$: the class of graphs with maximum degree at most d, the class of d-regular graphs, and the class of d-degenerate graphs. They classified the parameterized complexity of all three problems with respect to the parameters k, d, and $d + k$. The first problem, also known as MDC, is defined as follows.

Maximum Degree Contraction **Parameter:** $k + d$
Input: Graph G, integers k, d
Question: Does there exist a subset F of $E(G)$ of size at most k such that every vertex in G/F has degree at most d?

The authors proved that MDC is FPT when parameterized by $k + d$, W[2]-Hard when parameterized by k (even when restricted to split graphs), and para-NP-Hard when parameterized by d. Note that the problem is trivially solvable in polynomial time when $d \leq 1$ and NP-Hard for every constant $d \geq 2$.

Consider brute-force algorithm for MDC that given an instance (G, k, d), where graph G has n vertices, enumerates all subsets of edges of size at most k in G and for each subset contracts all edges in it to check whether the resulting graph has degree at most d. This algorithm runs in time $n^{\mathcal{O}(k)}$. Our first results states that this algorithm is optimal, up to constants in the exponents, under ETH.

▶ **Theorem 1.** *Unless ETH fails, there is no algorithm that given any instance (G, k, d) of* Maximum Degree Contraction *runs in time $n^{o(k)}$ and correctly determines whether it is a* Yes *instance.*

Belmonte et al. [8] presented an FPT algorithm for MDC that runs in time $(d + k)^{2k} \cdot n^{\mathcal{O}(1)}$. As for any non-trivial instance $d + k$ is smaller than n, we can conclude that there is no algorithm that given any instance (G, k, d) of MDC runs in time $(d + k)^{o(k)} \cdot n^{\mathcal{O}(1)}$ and correctly determines whether it is a Yes instance, unless ETH fails.

We remark that that the lower bound in Theorem 1 does not hold when d is a fixed constant and not a part of input. Hence, it is possible that MDC admits an algorithm that runs in time $k^{o(k)} \cdot n^{\mathcal{O}(1)}$ for a constant value of d. Belmonte et al. [8] proved that MDC problem admits linear vertex kernels on connected graphs when $d = 2$. This linear kernel leads to an FPT algorithm[1] running in time $2^{\mathcal{O}(k)} \cdot n^{\mathcal{O}(1)}$. This hints that it is possible to design a better FPT algorithm for small values of d. Our second result shows that this is indeed the case.

▶ **Theorem 2.** *There is an algorithm that given an instance (G, k, d) of* Maximum Degree Contraction *runs in time $2^{\mathcal{O}(dk)} \cdot n^{\mathcal{O}(1)}$ and correctly determines whether it is a* Yes *instance.*

We note that the reduction used in [8] to prove that MDC is NP-Hard for any constant $d \geq 2$ implies that there is no $2^{o(dk)}$ algorithm for this problem.

Next, we look at the kernelization of MDC. Belmonte et al. [8] left it as an open question to determine whether MDC admits a polynomial kernel when parameterized by $k + d$. Our last result answers this question in negative.

▶ **Theorem 3.** *Unless* NP $\subseteq$ coNP/*poly,* Maximum Degree Contraction, *parameterized by $k + d$, does not admit a polynomial compression.*

It is known that the Bounded Degree Deletion problem admits a kernel with $\mathcal{O}(d^3 k)$ vertices [17]. Hence, $\mathcal{H}_{\leq d}$-Modification is another example for which changing the modification operations from vertex deletion to edge contraction changes the compressibility drastically.

[1] The algorithm colors vertices in the reduced instance with two colors and contracts each connected component in the colored subgraphs.

We organize the remaining paper as follows. In Section 2, we present some preliminaries and observations regarding MDC. In Section 3, we give a parameter preserving reduction from $(k \times k)$-Permutation Independent Set to MDC to rule out $n^{o(k)}$ algorithm for the later problem under ETH. We present an FPT algorithm using universal sets and branching techniques in Section 4. In Section 5, we present a parameter preserving reduction from Red Blue Dominating Set to rule out polynomial compression for MDC problem. We conclude this article with an open question in Section 6.

2 Preliminaries

For a positive integer q, we denote set $\{1, 2, \ldots, q\}$ by $[q]$.

2.1 Graph Theory

In this article, we consider simple graphs with a finite number of vertices. For an undirected graph G, sets $V(G)$ and $E(G)$ denote its set of vertices and edges, respectively. Unless otherwise specified, we use n to denote the number of vertices in the input graph G. We denote an edge with two endpoints u, v as (u, v). Two vertices u, v in $V(G)$ are *adjacent* to each other if there is an edge (u, v) in $E(G)$. The open neighborhood of a vertex v, denoted by $N_G(v)$, is the set of vertices adjacent to v and its degree $\deg_G(v)$ is $|N_G(v)|$. The closed neighborhood of a vertex v, denoted by $N_G[v]$, is the set $N(v) \cup \{v\}$. We omit the subscript in the notation for neighborhood and degree if the graph under consideration is clear. For a subset S of $V(G)$, we define $N[S] = \bigcup_{v \in S} N[v]$ and $N(S) = N[S] \setminus S$. For a subset F of edges, a subset of vertices $V(F)$ denotes the collection of endpoints of edges in F. We say a set of edges F *spans* a set of vertices S if $S \subseteq V(F)$. For a subset S of $V(G)$, we denote the graph obtained by deleting S from G by $G - S$ and the subgraph of G induced on the set S by $G[S]$. For two subsets S_1, S_2 of $V(G)$, edge set $E(S_1, S_2)$ denotes the edges with one endpoint in S_1 and another one in S_2. We say S_1, S_2 are adjacent if $E(S_1, S_2)$ is non empty. For an integer q, a q-coloring of graph G is a function $\phi : V(G) \to [q]$. A *proper coloring* of G is a q-coloring ϕ of $V(G)$ for some integer q such that for any edge (u, v), $\phi(u) \neq \phi(v)$. There is a proper coloring of the graph with $\Delta(G) + 1$ many colors which can found in polynomial time. A set of vertices S is said to be *independent set* if no two vertices in S are adjacent to each other. A set of edges F is called *matching* if no two edges in F share an endpoint. A graph is called *connected* if there is a path between every pair of distinct vertices. A subset S of $V(G)$ is said to be a *connected set* if $G[S]$ is connected. A *spanning tree* of a connected graph is its connected acyclic subgraph, which includes all the vertices of the graph.

2.2 Graph Contraction

The *contraction* of an edge uv in G deletes vertices u and v from G, and adds a new vertex which is adjacent to vertices that were adjacent to either u or v. This process does not introduce self-loops or parallel edges. The resulting graph is denoted by G/e. For a graph G and edge $e = uv$, we formally define G/e in the following way: $V(G/e) = (V(G) \cup \{w\}) \setminus \{u, v\}$ and $E(G/e) = \{xy \mid x, y \in V(G) \setminus \{u, v\}, xy \in E(G)\} \cup \{wx \mid x \in N_G(u) \cup N_G(v)\}$. Here, w is a new vertex. An edge contraction reduces the number of vertices in a graph by exactly one. Several edges might disappear because of one edge contraction. For a subset of edges F in G, graph G/F denotes the graph obtained from G by contracting each connected component in the sub-graph $G' = (V(F), F)$ to a vertex.

We now formally define a contraction of graph G to another graph H.

▶ **Definition 4** (Graph Contraction). *A graph G is said to be* contractible *to graph H if there is a function $\psi : V(G) \rightarrow V(H)$ such that following properties hold.*

1. *For any vertex h in $V(H)$, set $W(h) := \{v \in V(G) \mid \psi(v) = h\}$ is not empty and graph $G[W(h)]$ is connected.*
2. *For any two vertices h, h' in $V(H)$, edge hh' is present in H if and only if $E(W(h), W(h'))$ is not empty.*

We say graph G is contractible to H via mapping ψ. For a vertex h in H, set $W(h)$ is called a *witness set* associated with or corresponding to h. We define the *H-witness structure* of G, denoted by $\mathcal{W}$, as a collection of all witness sets. Formally, $\mathcal{W} = \{W(h) \mid h \in V(H)\}$. A witness structure $\mathcal{W}$ is a partition of vertices in G. If a *witness set* contains more than one vertex, then we call it *big* witness set, otherwise it is *small* witness set.

If graph G has a H-witness structure, then graph H can be obtained from G by a series of edge contractions. For a fixed H-witness structure, let F be the union of spanning trees of all witness sets. By convention, the spanning tree of a singleton set is the empty set. To obtain graph H from G, it is sufficient to contract edges in F. Hence, $H = G/F$. For a G/F-witness structure $\mathcal{W}$ of G, there is a unique function $\psi : V(G) \rightarrow V(G/F)$ corresponding to it. We say graph G is *k-contractible* to H if the cardinality of F is at most k. In other words, H can be obtained from G by at most k edge contractions. The following observations are immediate consequences of definitions.

▶ **Observation 2.1.** *If graph G is k-contractible to graph H via mapping ψ then following statements are true.*

1. *Any H-witness structure of G has at most k big witness sets.*
2. *For a fixed H-witness structure, the number of vertices in G which are contained in big witness sets is at most $2k$.*
3. *For a vertex v in $V(G)$, if $|W(\psi(v))| = 1$ then $\deg_H(\psi(v)) \leq \deg_G(v)$.*
4. *For $U \subseteq V(G)$, define $\psi(U) := \{\psi(u) \mid u \in U\}$. Then, $|U| \leq |\psi(U)| + k$.*

Proof. Let $\mathcal{W}$ be the H-witness structure of G and F be the union of the spanning trees of all witness sets. As G is k-contractible to H, we have $|F| \leq k$.
(1) As any big witness set contains at least one edge in F, the number of big witness set is at most k.
(2) As F spans all vertices in big witness set, the number of vertices in big witness set is at most $2k$.
(3) Let h_i be a vertex in $N_H(\psi(v))$. As $(\psi(v), h_i) \in E(H)$, set $E(W(\psi(v)), W(h_i))$ is a non-empty subset of $E(G)$. As $|W(\psi(v))| = 1$, this implies $E(\{v\}, W(h_i))$ is a non empty. As h_i is an arbitrary neighbor of $\psi(v)$, we can conclude that in graph G, v is adjacent with at least as many vertices as $\deg_H(\psi(v))$. Hence, $\deg_H(\psi(v)) \leq \deg_G(v)$.
(4) Assume that $|U| > |\psi(U)| + k$. Fix an arbitrary order on vertices in U. We define a function $\phi : U \rightarrow \psi(U) \cup \{\perp\}$ as follows: $\phi(u_i) = \phi(u_i)$ if $\phi(u_i) \neq \phi(u_j)$ for $j < i$ otherwise $\phi(u_i) = \perp$. For a vertex $\psi(u_i)$ in $\psi(U)$, the function ϕ selects one vertex amongst the set $\{u | \ \psi(u) = \psi(u_i)\}$. Define $U_0 = \{u \in U | \ \phi(u) = \perp\}$. By our assumption, $|U_0| > k$.

Consider an arbitrary vertex u_i in U_0. By the construction, there is an index $j \in [|U|]$ such that $u_j \in U$, $\psi(u_j) = \psi(u_i)$ and $j < i$. As $\psi(u_i) = \psi(u_j)$, both u_i, u_j are in some big witness set in $\mathcal{W}$. As F is the union of edges of spanning trees of witness sets in $\mathcal{W}$, there is a unique path from u_i to u_j that comprises only edges in F. Consider the edge in this path incident to u_i. We assign vertex u_i to this edge in F. As u_i is an arbitrary vertex in U_0, we can assign an edge in F to every vertex in U_0. Note that we are considering the first edge in the unique path from some vertex in U_0 to some vertex outside U_0. Hence, no two vertices in U_0 can be assigned to same edge in F. This contradicts the fact that $|F| \leq k$. Hence, our assumption is wrong and $|U| \leq |\psi(U)| + k$. ◀

2.3 Maximum Degree Contraction

In this subsection, we prove some observations and a lemma related to MDC. We say a set of edges F is a *solution* to instance (G, k, d) if the number of edges in F is at most k and the maximum degree of graph G/F is at most d. The number of edges that we are allowed to contract, k, is also called *solution size*. We start with the following simple observation that states that contracting an edge in a solution does not produce a No instance.

▶ **Observation 2.2.** *If (G, k, d) is a* Yes *instance of* MDC *and $F \subseteq E(G)$ is a solution to (G, k, d), then for any edge (u, v) in F, instance $(G/\{(u, v)\}, k - 1, d)$ is a* Yes *instance of* MDC.

We bound the maximum degree of graph by $k + d$ in the non-trivial instances of the problem.

▶ **Observation 2.3.** *If there is a vertex of degree $d + k + 1$ or more in G then (G, k, d) is a* No *instance.*

Proof. Suppose there is a vertex, say v, of degree greater than $d+k+1$ in graph G. Assume, for the sake of a contradiction, that (G, k, d) is a Yes-instance. Let (G, k, d) is k-contractible to a graph H, via mapping ψ, such that the maximum degree of vertices in H is at most d. By Observation 2.1 (4), $|N_G[v]| \leq |\psi(N_G[v])| + k$ where $\psi(N_G[v]) = \bigcup_{u \in N_G[v]} \psi(u)$. As $d + k + 1 < |N_G[v]|$, we have $d + 1 < |\psi(N_G[v])|$. As $\psi(N_G[v]) \subseteq N_H(\psi(v))$, vertex $\psi(v)$ is adjacent with $d + 1$ or more vertices in H. This contradicts the fact that the maximum degree of vertices in H is at most d. Hence, our assumption was wrong and (G, k, d) is a No instance. ◀

If every vertex in G has degree at most d, then (G, k, d) is a trivial Yes instance. Hence, there is at least one vertex in G that has degree at least $d + 1$. We prove that the number of such vertices is bounded.

▶ **Observation 2.4.** *Let (G, k, d) be a* Yes *instance of* MDC. *Then, G contains at most $k(d + 2)$ vertices that has degree at least $d + 1$.*

Proof. Let L be the collection of vertices in G which has degree at least $d + 1$. As (G, k, d) is a Yes instance, there is a solution, say F, to it. Let $\mathcal{W}$ be a (G/F)-witness structure of G. By Observation 2.5, every vertex in L is either contained in a big-witness set or at least two of its neighbors are in a big witness set. By Observation 2.1 (2), the number of vertices in big witness sets is at most $2k$. As every vertex in G/F has degree at most d, there are at most dk vertices in G that are adjacent to some vertex in big witness sets. This implies that there are at most $k(2 + d)$ vertices in L. ◀

The following observation specifies how a solution behaves locally.

▶ **Observation 2.5.** *Consider a* Yes *instance (G, k, d) of* MDC *and let v be a vertex of degree at least $d+1$ in G. Then, for any solution F to (G, k, d), there are at least two vertices in $N[v]$ that are in the same witness set in the G/F-witness structure of G.*

Proof. Let G is contractible to a graph G/F, via mapping ψ. Assume, for the sake of contradiction, that no two vertices in $N[v]$ are in the same witness set. This implies $|N[v]| = |\psi(N[v])|$, where $\psi(N_G[v]) = \bigcup_{u \in N_G[v]} \psi(u)$. As $\psi(N_G[v]) \subseteq N_{G/F}(\psi(v))$ and $|N[v]| > d+1$, vertex $\psi(v)$ is adjacent with $d+1$ or more vertices in G/F. This contradicts the fact that the maximum degree of vertices in G/F is at most d. Hence, our assumption was wrong and there are at least two vertices in $N[v]$ that are in some big-witness set in G/F-witness structure of G. ◀

We say that solution F *merges* at least two vertices in $N[v]$. Note that for an edge (u, v) in G, it is possible that $(u, v) \notin F$ but F merges u, v.

The following lemma allows us to conclude that an instance is a No instance if we find a sizeable collection of *large stars* that do not intersect with each other. We present it in the form suitable for the application in the later part of the article.

▶ **Lemma 5.** *For an instance $(G, k-1, d)$, suppose there is subset L° of $V(G)$ that satisfies the following conditions: (i) For every vertex v in L°, $N(v)$ is an independent set of size at least $d+1$. (ii) For any two different vertices u, v in L°, $N(v) \cap N(u) = \emptyset$. (iii) $|L^\circ| \geq k$. Then, $(G, k-1, d)$ is a* No *instance.*

Proof. Assume, for the sake of contradiction, that $(G, k-1, d)$ is a Yes instance. Let F be a solution to (G, k, d) and G is $(k-1)$-contractible to be G/F via ψ. By Observation 2.5, for every vertex v in L°, there are at least two vertices in $N[v]$ which are in same witness set in the G/F-witness structure of G. As $N(v)$ is an independent set, there is no edge whose both endpoints are in $N(v)$. Hence, for every vertex in L°, one of the following two statements must be true: (a) F contains an edge incident to v. (b) F contains at least two edges incident to $N(v)$ but are not incident to $\{v\}$. Let L_1 be the collection of vertices in L° for which the first statement is true. Let F_1 be the subset of F that are incident to some vertex in L_1. Recall that for any two vertices in $u, v \in L_1$, as $N[v] \cap N[u] = \emptyset$. Hence, no edge in F_1 is incident to more than one vertex in L°. Hence, $|F_1| = |L_1|$. For every v in $L \setminus L_1$, there are at least two edges incident to $N(v)$. Note that these edges are in $F \setminus F_1$ as they are not incident to any vertex in L°. As every edge in $F \setminus F_1$ can be incident to the open neighborhood of at most two vertices in $L \setminus L_1$, we can conclude that $2|F \setminus F_1| \geq 2|L \setminus L_1|$. This implies that the number of edges in F is at least $|L_1| + |L \setminus L_1| = |L|$. This contradicts the fact that $|F| \leq k-1$ and $|L| \geq k+1$. Hence our assumption is wrong and (G, k, d) is a No instance. ◀

2.4 Parameterized Complexity

An instance of a parameterized problem Π comprises of an input I, which is an input of the classical instance of the problem and an integer k, which is called as the parameter. A problem Π is said to be *fixed-parameter tractable* or in FPT if given an instance (I, k) of Π, we can decide whether or not (I, k) is a Yes instance of Π in time $f(k) \cdot |I|^{\mathcal{O}(1)}$. Here, $f(\cdot)$ is some computable function whose value depends only on k.

A *compression* of a parameterized problem Π_1 into a (non-parameterized) problem Π_2 is a polynomial algorithm that maps each instance (I_1, k_1) of Π_1 to an instance I of Π_2 such that (1) (I, k) is a Yes instance of Π_1 if and only if I_2 is a Yes instance of Π_2, and (2) size of I_2 is bounded by $g(k)$ for a computable function $g(\cdot)$. The output I_2 is also called a compression. The function g is said to be the size of the compression. A compression is polynomial if g is polynomial. A kernelization algorithm for a parameterized problem Π is a polynomial algorithm that maps each instance (I, k) of Π to an instance (I', k') of Π such that (1) (I, k) is a YES instance of Π if and only if (I', k') is a YES instance of Π, and (2) $|I'| + k'$ is bounded by $g(k)$ for a computable function $g(\cdot)$. Respectively, (I', k') is a kernel and g is its size. A kernel is polynomial if g is polynomial.

It is typical to describe a compression or kernelization algorithm as a series of reduction rules. A reduction rule is a polynomial algorithm that takes as an input an instance of a problem and output another, usually reduced, instance. A reduction rule said to be *applicable* on an instance if the output instances is different than the instance. A reduction rule is *safe* if the input instance is a YES instance if and only if the output instance is a YES instance.

For details on parameterized complexity and related terminologies, we refer the reader to the books of Downey and Fellows [16], Flum and Grohe [18], Niedermeier [34], and the more recent books by Cygan et al. [14] and Fomin et al. [20].

3 A Lower Bound for the Algorithm

In this section, we prove Theorem 1. We present a reduction from $(k \times k)$-PERMUTATION INDEPENDENT SET (PIS) problem to MAXIMUM DEGREE CONTRACTION problem. In the $(k \times k)$-PIS problem we are given a graph H on a vertex set $[k] \times [k]$. In other words, the vertex set is formed by a $k \times k$ table. We denote vertices in the table by $v[i, j]$ for $1 \leq i, j \leq k$. The question is whether there exists an independent set X in H that contains exactly one vertex from each row and each column of the table. In other words, for every $i, j \in [k]$ there is exactly one element of X that has i on the first coordinate and j on the second coordinate. Note that without loss of generality we may assume that each row and each column of the table forms an independent set.[2] The following result is known for this problem.

▶ **Proposition 6** ([31]). *Unless ETH fails,* $(k \times k)$-PERMUTATION INDEPENDENT SET *can not be solved in time* $k^{o(k)}$.

Reduction

The reduction accepts an instance, say (H, k), of $(k \times k)$-PERMUTATION INDEPENDENT SET as an input. Here, H is a graph with vertex set formed by a $k \times k$ table. The reduction modifies a copy of the graph H in the following way.

- It adds a vertex corresponding to each row in the table and makes it adjacent with all vertices in that row. Let $R = \{r_1, r_2, \ldots, r_k\}$ be the set of vertices corresponding to rows.
- It adds a vertex corresponding to each column in the table and makes it adjacent with all vertices in that column. Let $C = \{c_1, c_2, \ldots, c_k\}$ be the set of vertices corresponding to columns.
- It adds set $S = \{s_1, s_2, \ldots, s_k\}$ of k vertices. For every i in $[k]$, it makes s_i adjacent with every vertex in $V(H) \cup C$ and with r_i.
- For every vertex r_i in R, it adds k^2 pendant vertices and makes them adjacent with r_i.
- For every vertex c_j in C, it adds $(k^2 - k + 1)$ pendant vertices and makes them adjacent with c_j.

See Figure 1 for an illustration. Let G be the graph obtained from a copy of graph H with the above modifications. The algorithm returns $(G, k, k^2 + k)$ as instance of MDC.

[2] Since we are looking for an independent set, it is intuitive to add all missing edges in a row or a column of the table. But to simply our reduction, we remove edges that have both their endpoints in the same row or column. It is easy to verify that this operation is safe.

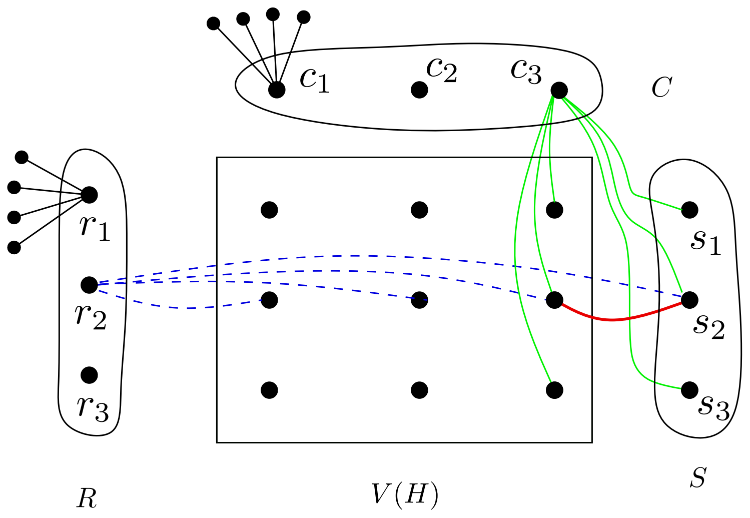

Figure 1 Dotted (blue) lines and thin (green) lines show the adjacency of vertices in R and C, respectively. Contracting the thick (red) edge $(v[2,3], s_2)$ represents selecting vertex $v[2,3]$ into the independent set. For the sake of clarity, we do not depict all edges present in the graph.

We present intuition of the proof of correctness. We describe how a solution, if it exists, to (G, k, d) leads to a solution to (H, k). We hope that this will also provide some intuition as to how a solution to (H, k) leads to a solution to (G, k, d). Note that S, C, R are independent sets in G. Every vertex in $R \cup C \cup S$ has degree $d + 1$ and every vertex in $V(G) \setminus (R \cup C \cup S)$ has degree strictly less than d. We first argue that any solution for (G, k, d) can only contain edges in $E(G)$ that have one endpoint in $V(H)$ and another endpoint in S. Then, we prove that for every $i \in [k]$ a solution must pick an edge incident to some vertex in i^{th} row and on s_i to reduce the degree of vertex r_i. We prove a similar statement for every column. Hence, for every $i \in [k]$, a solution contains an edge of the form $(v[i, j], s_i)$ for some $j \in [k]$. As there are at most k edges in a solution, every edge is of this form. For $i_1, i_2, j_1, j_2 \in [k]$, let $(v[i_1, j_1], s_{i_1})$ and $(v[i_2, j_2], s_{i_2})$ be two edges in a solution. We argue that if $(v[i_1, j_1], v[i_2, j_2])$ is an edge in G (and hence in H) then degrees of vertices obtained by contracting $(v[i_1, j_1], s_{i_1})$ and $(v[i_2, j_2], s_{i_2})$ are more than d. As this is true for any two arbitrary edges in solution, their endpoints in $V(H)$ form an independent set in H. We formalize this intuition in the following two lemmas.

▶ **Lemma 7.** *Suppose the reduction returns (G, k, d) when the input is (H, k). If (H, k) is a* YES *instance of $(k \times k)$-PIS then (G, k, d) is a* YES *instance of* MDC.

Proof. Suppose (H, k) is a YES instance and let X be an independent set in H that contains exactly one vertex from each row and each column of the table. Define a function $\rho : [k] \to [k]$ such that $j = \rho(i)$ if $v[i, j]$ is a vertex in X. By the properties of X, we can conclude that ρ is one-to-one and onto function. We construct solution F to (G, k, d) using independent set X. For every vertex $v[i, \rho(i)]$ in X, add edge $(s_i, v[i, \rho(i)])$ in F. By construction, the cardinality of F is k. We argue that the maximum degree of any vertex in G/F is d. As mentioned before, in graph G, set $R \cup C \cup S$ is the collection of all vertices of degree strictly greater than d. More precisely, every vertex in $R \cup C \cup S$ has degree $d + 1$. We demonstrate that contracting edges in F reduces the degree of each vertex in $R \cup C \cup S$ by one.

Note that edges in F form a matching in G. For every i in $[k]$, let s_i° be the new vertex added while contracting edge $(s_i, v[i, \rho(i)])$. Let $G' = G/F$ and $S^\circ = \{s_1^\circ, s_2^\circ, \dots, s_k^\circ\}$. Note that $V(G')$ can be partitioned into $R, C, S^\circ, V(H) \setminus X$, and pendent vertices which are adjacent with $R \cup C$. Every vertex in $V(H) \setminus X$ is adjacent with at most $k^2 - 2k + k + 2 \le d - 2k + 2 \le d$ vertices in G'. Every vertex in R is adjacent with $k - 1$ vertices in $V(H) \setminus X$, one vertex in S°, and k^2 pendant vertices in G'. Hence, degree of every vertex in R in G' is $k - 1 + 1 + k^2 = d$. For a vertex, say c_j, in C, there exists a vertex $v[i, \rho(i)]$ in X such that $j = \rho(i)$. Hence, in graph G', vertex c_j is adjacent with $k - 1$ vertices in $V(H)$, k vertices in S°, and $k^2 - k + 1$ pendent vertices. Hence, degree of c_j in G' is $k - 1 + k + k^2 - k + 1 = d$. Since c_j is an arbitrary vertex in C, this is true for every vertex in C.

We now argue that S° is an independent set in G'. Consider two vertices, say s_i°, s_j° in S°. By construction, vertices s_i and s_j are not adjacent with each other in G. As X is an independent set in G, vertices $v[i, \rho(i)], v[j, \rho(j)]$ in X are not adjacent with each other. This implies there is no edge with one endpoint in set $\{s_i, v[i, \rho(i)]\}$ and another endpoint in $\{s_j, v[j, \rho(j)]\}$. This implies that vertices s_i° and s_j° are not adjacent with each other in G'. Since this is true for any two vertices in S°, it is an independent set in G'. By construction, for any i in $[k]$, vertex $v[i, \rho(i)]$ is adjacent with only one vertex, viz r_i, in R. Hence, any vertex s_i° in S is adjacent with one vertex vertex in R, k vertices in C and $k^2 - 1$ vertices in $V(H) \setminus X$ in graph G'. This implies that every vertex in S° has degree $1 + k + k^2 - 1 = d$. Hence, the maximum degree of any vertex in G' is at most d. This implies that (G, k, d) is a Yes instance which concludes the proof. ◀

In the remaining section, we prove the following lemma.

▶ **Lemma 8.** *Suppose the reduction returns (G, k, d) when the input is (H, k). If if (G, k, d) is a* Yes *instance of* MDC *then (H, k) is a* Yes *instance of $(k \times k)$*-PIS.

To prove the lemma, we first investigate how a solution to (G, k, d) can intersect with edges in G. Recall that for vertex subset X, Y, we denote the set of all edges with one endpoint in X and another endpoint in Y by $E(X, Y)$. Let P_C and P_R be the collection of pendant vertices that are adjacent with C and R, respectively. By construction, edges of G can be partitioned into following five sets: $E(C \cup R, P_C \cup P_R)$, $E(C \cup R, S)$, $E(C \cup R, V(H))$, $E(V(H), V(H))$, and $E(S, V(H))$. We first prove that any solution to (G, k, d) does not intersect with the first four sets.

Suppose (G, k, d) is a Yes instance and $F \subseteq E(G)$ be a solution to (G, k, d).

▷ Claim 9. $F \cap E(C \cup R, P_C \cup P_R) = \emptyset$.

Proof. Assume that there exist an edge, say (c_i, v), in $F \cap E(C, P_C)$ where vertices c_i, v are in C and P_C, respectively. Note that, instance $(G/\{(c_i, v)\}, k - 1, d)$ and set R satisfy the premise of Lemma 5. Hence, we can conclude that $(G/\{(c_i, v)\}, k - 1, d)$ is a No instance. This contradicts Observation 2.2. Hence our assumption is wrong and $F \cap E(C, P_C)$ is an empty set.

Assume that there exist edge (r_i, v) in $F \cap E(R, P_R)$ where vertices r_i, v are in R and P_R, respectively. Let R' be the set obtained from R by removing r_i and adding the vertex which was introduced while contracting edge (r_i, v). Note that, instance $(G/\{(r_i, v)\}, k - 1, d)$ and set R' satisfy the premise of Lemma 5. Hence, we can conclude that $(G/\{(r_i, v)\}, k - 1, d)$ is a No instance. This contradicts Observation 2.2. Hence our assumption is wrong and $F \cap E(R, P_R)$ is an empty set.

By the construction of G, sets $E(C, P_R)$ and $E(R, P_R)$ are empty. This implies that there is no edge in $F \cap E(C \cup R, P_C \cup P_R)$. ◀

▷ Claim 10. $F \cap E(C \cup R, S) = \emptyset$.

Proof. Assume that there exist an edge, say (c_i, s_j), in $F \cap E(C, S)$ where vertices c_i, s_j are in C and S, respectively. Let w be the new vertex introduced while contracting edge (c_i, s_j). In graph $(G/\{(c_i, s_j)\}$, vertex w is adjacent with every vertex in $(S \setminus \{s_j\}) \cup V(H) \cup (C \setminus \{c_i\})$ and with all pendent vertices which were adjacent with c_i in G. Hence, the degree of w in $(G/\{(c_i, s_j)\})$ is at least $2k^2 + k - 1 \geq (k^2 + k) + (k - 1) + 1 = d + (k - 1) + 1$. By Observation 2.5, $(G/\{(c_i, s_j)\}, k - 1, d)$ is a No instance. This contradicts Observation 2.2. Hence our assumption is wrong and $F \cap E(C, S)$ is an empty set. We can conclude that $F \cap E(C, S)$ is an empty set by a similar argument. This concludes the proof of the claim. ◀

▷ Claim 11. $F \cap E(C, V(H)) = \emptyset$.

Proof. Assume that there exist an edge, say (c_j, x_{ij}), in $F \cap E(C, V(H))$ where vertices c_j, x_{ij} are in C and $V(H)$, respectively. Note that, instance $(G/\{(c_j, x_{ij})\}, k - 1, d)$ and set R satisfy the premise of Lemma 5. Hence, we can conclude that $(G/\{(c_j, x_{ij})\}, k - 1, d)$ is a No instance. This contradicts Observation 2.2. Hence our assumption is wrong and $F \cap E(C, V(H))$ is an empty set. ◀

▷ Claim 12. $F \cap E(R, V(H)) = \emptyset$.

Proof. Assume that is $F \cap E(R, V(H))$ is not empty. As any vertex c_j in C has degree $d + 1$, edges in F merge at least two vertices in $N[c_j]$ (Lemma 5). We argue that if our assumption is correct then there are not enough edges to merge two vertices in each $N[c_j]$.

Let J' be the set of columns such that there is no edge of the form $(x_{ij'}, s)$ in F, where $i, j' \in [k]$ and $s \in S$. Note that set J' is not empty as there are k columns and at most $(k - 1)$ edges in $F \cap E(V(H), S)$. There are at most $|F| - (k - |J'|) = |J'|$ many edges to merge two vertices in $N[c_j']$ for each j' in J'. For any two different vertices $c_j, c_{j'}$ in C, their neighbourhoods outside S do not intersect. Formally, $(N[c_j] \setminus S) \cap (N[c_{j'}] \setminus S) = \emptyset$. Hence, $|J'|$ many edges need to cover $2|J'|$ vertices. This implies that edges in $F \setminus E(V(H), S)$ form a matching in G. For any vertex c_j in C, its neighborhood is an independent set. Hence, the only possible way to merge two vertices in each $N[c_{j'}]$ using edges in matching is to contract an edge incident to $c_{j'}$ and one of its neighbors in $V(H)$. Hence, all the edges in $F \setminus E(V(H), S)$ are in $E(C, V(H))$. This is a contradiction to Claim 10 which states that there is no solution edge in $E(C, V(H)$. Hence our assumption was wrong and $F \cap E(R, V(H)) = \emptyset$. ◀

▷ Claim 13. $F \cap E(V(H), V(H)) = \emptyset$.

Proof. Assume that there exists an edge, say $(x_{ij}, x_{i'j'})$, in $F \cap E(V(H), V(H))$ for some $i, j, i', j' \in [k]$. Consider instance $(G/\{(x_{ij}, x_{i'j'})\}, k - 1, d)$. As any vertex r_i in R has degree $d + 1$, edges in any solution for the instance merge at least two vertices in $N[r_i]$ (Lemma 5). Hence, $F \setminus \{(x_{ij}, x_{i'j'})\}$ merges at least two vertices in $N[r_i]$ for each r_i in R. Let Y be the set of vertices in $G/\{(x_{ij}, x_{i'j'})\}$ such that Y contains at least two vertices in $N[r_i]$ for every r_i in R. Note that the cardinality of Y is at least $2k - 1$. By Claim 4, there is no edge in $F \cap E(R, V(H)$. This implies that $(k - 1)$ in $F \setminus \{(x_{ij}, x_{i'j'})\}$ covers at least $2k - 1$ vertices. This is a contradiction as any edge can cover at most two vertices. Hence our assumption is wrong and $F \cap E(V(H), V(H)) = \emptyset$. ◀

Proof. (of Lemma 8) By Claims 9 to 13, every edge in F is of the form (x_{ij}, s_l) for some $i, j, l \in [k]$ where $x_{ij} \in V(H)$ and $s_l \in S$. Let X' be the collection of vertices in $V(H)$ that are endpoints of some edges in F. The size of X' is at most k. In the remaining part, we argue that X' is an independent set in H and contains one vertex from each row and column.

We first argue that for every vertex in s_l in S, there is exactly one edge in F which is incident to s_l. Note that S is an independent set and every vertex in it has the degree $d+1$. By Lemma 5, edges in F merges at least two vertices in $N[s_l]$ for every s_l. As $|F| \leq k$ and $|S| = k$, there is exactly one edge incident to s_l for every $l \in [k]$.

We now prove that X' contains one vertex from every row and column. Recall that for every $i \in [k]$, the degree of vertex r_i is $d+1$ in G and $N(r_i)$ contains s_i and all the vertices in i^{th} row. By Lemma 5, for every i, edges in F merge at least two vertices in $N[r_i]$. Hence, the other endpoint of the edge incident s_i is some vertex in i^{th} row. This implies there exists a vertex in X' from each row. By similar arguments, we can prove that there exists a vertex in X' from each column.

It remains to argue that X' is an independent set in H. Define function $\phi : S \to V(H)$ as follows: For every $s_l \in S$, assign $\phi(s_l) = x_{ij}$ if $(x_{ij}, s_l) \in F$ for some $i, j \in [k]$. As there is exactly one edge in F which is incident to s_l, function ϕ is well defined. Assume that there exists an edge $(\phi(s_l), \phi(s_{l'}))$ in H. Let s_l° and $s_{l'}^\circ$ be the two new vertices added while contracting edges $(s_l, \phi(s_l))$ and $(s_{l'}, \phi(s_{l'}))$. Note that s_l° is adjacent with $s_{l'}^\circ$, one vertex in R, k vertices in RC, and $k^2 - 1$ vertices in $V(H)$. Hence, degree of s_l° is $1 + 1 + k + k^2 - 1 = d + 1$. This contradicts the fact that every vertex in G/F has degree at most d. Hence our assumption was wrong and there is no edge $(\phi(s_l), \phi(s_{l'}))$ in H. Since, $s_l, s_{l'}$ are any two arbitrary vertices in S, we can conclude that X' is an independent set in H.

Hence, if (G, k, d) is a YES instance than so is (H, k). ◀

We are now in a position to present a proof of Theorem 1 using Proposition 6, Lemma 7, and Lemma 8.

Proof. (of Theorem 1) Assume, for the sake of contradiction, that there is an algorithm, say $\mathcal{A}$, that given any instance (G, k, d) of MDC runs in time $n^{o(k)}$ and correctly determines whether it is a YES instance or not. Using this algorithm as subroutine, we construct an algorithm to solve $(k \times k)$-PIS.

Consider Algorithm $\mathcal{B}$ that given an instance (H, k) of $(k \times k)$-PIS construct an instance (G, k, d) of MDC as described in the reduction. Then, it calls Algorithm $\mathcal{A}$ as subroutine on instance (G, k, d). If Algorithm $\mathcal{A}$ returns YES then Algorithm $\mathcal{B}$ returns YES otherwise it returns NO. The correctness of Algorithm $\mathcal{B}$ follows from the correctness of Algorithm $\mathcal{A}$, Lemma 7, and Lemma 8. We now argue the running time of Algorithm $\mathcal{B}$. By the description of the reduction, it is easy to see that given instance (H, k), the algorithm computes instance (G, k, d) in time polynomial in $|V(H)| \in \mathcal{O}(k^2)$ and $|V(G)| = n \in \mathcal{O}(k^2)$. Hence, the total running time of the algorithm is $n^{o(k)} = k^{o(k)}$.

This implies there is an algorithm to solve $(k \times k)$-PIS in time $k^{o(k)}$. But this contradicts Proposition 6. Hence, our assumption is wrong which concludes the proof. ◀

4 A Different FPT Algorithm

In this section, we present a different FPT algorithm for MAXIMUM DEGREE CONTRACTION. We introduce a variation of the problem called LABELED-MAXIMUM DEGREE CONTRACTION (LABELED-MDC). We present an FPT algorithm for LABELED-MDC and use it as a subroutine to present an FPT algorithm for MDC.

Informally, an instance of LABELED-MDC is an instance of MDC along with a labeling of vertices in the graph. Every vertex has a red or blue label. We are only interest in a solution that satisfies the following properties: (1) every edge has red labelled endpoints, and (2) for any red-labelled maximal connected component, a solution either spans none or all the vertices in that component. We remark that because of the second condition, this problem is *not* a restricted version of MDC. We formally define LABELED-MDC as follows.

LABELED-MDC **Parameter:** $k + d$
Input: Graph G, a partition V_r, V_b of $V(G)$, and integers k, d
Question: Does there exist a subset F of $E(G)$ of size at most k such that (a) every vertex in G/F has degree at most d; (b) $V(F) \subseteq V_r$; and (c) for a connected component C of $G[V_r]$, if $C \cap V(F) \neq \emptyset$ then $C \subseteq V(F)$?

We say a set of edges F is a *solution* to instance $(G, (V_r, V_b), k, d)$ if the number of edges in F is at most k, the maximum degree of graph G/F is at most d, $V(F) \subseteq V_r$ and for a connected component C of $G[V_r]$, if $C \cap V(F) \neq \emptyset$ then $C \subseteq V(F)$.

It is easy to see that if $(G, (V_r, V_b), k, d)$ is a YES instance of LABELED-MDC then (G, k, d) is a YES instance of MDC. Let $\mathcal{U}$ be the family of all subsets of $V(G)$. If (G, k, d) is a YES instance of MDC then $(G, (V_r, V(G) \setminus V_r), k, d)$ is a YES instance of LABELED-MDC for some set V_r in $\mathcal{U}$. We use *universal sets* to construct a 'small' family of subsets of $V(G)$ that suffices for our purpose. We assume that there is a unique integer in $[n]$ for every vertex in $V(G)$. We use a subset of $[n]$ and a corresponding subset of $V(G)$ interchangeably.

▶ **Definition 14** (Universal Sets). *An (n, l)-universal set is a family $\mathcal{U}$ of subsets of $[n]$ such that for any $S \subseteq [n]$ of size l, the family $\{A \cap S \mid A \in \mathcal{U}\}$ contains all subsets of S.*

▶ **Proposition 15** ([5]). *For any $n, l \geq 1$ one can construct an (n, l)-universal set of size $2^{\mathcal{O}(l)} \cdot \log(n)$ in time $2^{\mathcal{O}(l)} \cdot n \log(n)$.*

In the following lemma, we argue that an FPT algorithm for LABELED-MDC leads to an FPT algorithm for MDC.

▶ **Lemma 16.** *Suppose there is an algorithm that given an instance $(G, (V_r, V_b), k, d)$ of* LABELED-MDC *runs in time $f(k, d) \cdot n^{\mathcal{O}(1)}$ and correctly determines whether it is a* YES *instance. Then, there is an algorithm that given an instance (G, k, d) of* MDC *runs in time $2^{\mathcal{O}(dk)} \cdot f(k, d) \cdot n^{\mathcal{O}(1)}$ and correctly determines whether it is a* YES *instance.*

Proof. Let $\mathcal{A}$ be an algorithm that given an instance $(G, (V_r, V_b), k, d)$ of LABELED-MDC runs in time $f(k, d) \cdot n^{\mathcal{O}(1)}$ and correctly determines whether it is a YES instance. We first describe an algorithm for MDC that uses $\mathcal{A}$ as a subroutine. For the input (G, k, d), the algorithm constructs a $(U, 2k + kd)$-universal family $\mathcal{U}$ using Proposition 15. For every set V_r in $\mathcal{U}$, the algorithm runs Algorithm $\mathcal{A}$ with input $(G, (V_r, V(G) \setminus V_r), k, d)$. The algorithm returns YES if Algorithm $\mathcal{A}$ returns YES for one of these inputs otherwise it returns NO. This completes the description of the algorithm. The running time of the algorithm follows from the description and Proposition 15. In the remaining proof, we argue the correctness of the algorithm. More precisely, we prove that (G, k, d) is a YES instance of MDC if and only if there is a subset V_r in $\mathcal{U}$ such that $(G, (V_r, V(G) \setminus V_r), k, d)$ is a YES instance of LABELED-MDC.

Suppose that (G, k, d) is a YES instance of MDC and let F be a solution to it. Note that $|V(F)| \leq 2k$. We first argue that the number of vertices in $N(V(F))$ is at most kd. Let $\mathcal{W}$ be the G/F-witness structure of G and $\psi : V(G) \to V(G/F)$ be the corresponding function. Consider an arbitrary vertex v in $N(V(F))$. As v is not in $V(F)$, $\psi(v)$ corresponds to a small witness set in $\mathcal{W}$. As v is in $N(V(F))$, $\psi(v)$ is adjacent to a vertex in G/F that corresponds to a big witness set in $\mathcal{W}$. By Observation 2.1 (1), there are at most k big witness sets in $\mathcal{W}$. Since the maximum degree of G/F is at most d, there are at most kd small witness sets in $\mathcal{W}$ that are adjacent with some big witness set. Hence, there are at most kd vertices in $N(V(F))$. As $\mathcal{U}$ is a $(n, 2k+dk)$-universal set and $|N[V(F)]| \leq 2k + dk$, there exists a set A in $\mathcal{U}$ such that the family $\{A \cap N[V(F)] \mid A \in \mathcal{U}\}$ contains all subsets of $N[V(F)]$. This implies, there exists a set, say V_r, such that $V_r \cap N[V(F)] = V(F)$. We argued that $(G, (V_r, V(G) \setminus V_r), k, d)$ is a YES instance of LABELED-MDC.

Note that G/F has maximum degree at most d and $V(F) \subseteq V_r$. We need to prove that for a connected component C of $G[V_r]$ if $C \cap V(F) \neq \emptyset$ then $C \subseteq V(F)$. Assume that there exits a connected component C of $G[V_r]$ such that $C \cap V(F) \neq \emptyset$ and $C \setminus V(F) \neq \emptyset$. As C is a connected component and $C \cap V(F) \neq \emptyset$, there exists a vertex v in $C \setminus V(F)$ that is adjacent with some vertex in $V(F)$. Hence, there is a vertex in $N(V(F)) \cap V_r$. This contradicts the fact that $V_r \cap N[V(F)] = V(F)$. Hence, our assumption is wrong and $C \setminus V(F)$ is an empty set. This implies $(G, (V_r, V(G) \setminus V_r), k, d)$ is a YES instance of LABELED-MDC. As mentioned before, it is easy to see that if $(G, (V_r, V_b), k, d)$ is a YES instance of LABELED-MDC then (G, k, d) is a YES instance of MDC. This concludes the proof of the lemma. ◀

In the remaining section, we present a recursive algorithm for LABELED-MDC. We start with the following simple reduction rules.

▶ **Reduction Rule 4.1.** *For an instance $(G, (V_r, V_b), k, d)$, if the maximum degree of vertices in G is at most d and $k \geq 0$ then return a* YES *instance.*

It is easy to see that the first reduction rule is safe. Recall that a set of edges F is called solution to $(G, (V_r, V_b), k, d)$ if the number of edges in F is at most k, the maximum degree of graph G/F is at most d, $V(F) \subseteq V_r$, and for a connected component C of $G[V_r]$, if $C \cap V(F) \neq \emptyset$ then $C \subseteq V(F)$. Consider a connected component C of $G[V_r]$. If $|C| = 1$ then no solution edge can be incident to it. Also, if $|C| \geq 2k + 1$ then because of the last property and the fact that $|V(F)| \leq 2k$, no solution edge can be incident to vertices in C. These simple observations prove that the following reduction rule is safe.

▶ **Reduction Rule 4.2.** *For an instance $(G, (V_r, V_b), k, d)$, if there is a connected component, say C, of $G[V_r]$ such that $|C| = 1$ or $|C| \geq 2k + 1$ then move C from V_r to V_b i.e. return instance $(G, (V_r \setminus C, V_b \cup C), k, d)$.*

By Observation 2.5, vertex v in V_b can be adjacent to at most $d + k$ vertices in V_r. The following reduction rule ensures that the neighbors of v in V_r are not spread across many connected components.

▶ **Reduction Rule 4.3.** *For an instance $(G, (V_r, V_b), k, d)$, if there exists a vertex, say v, in V_b for which $N_G(v)$ intersects with $d + 1$ different connected components of $G[V_r]$ then return a* NO *instance.*

▶ **Lemma 17.** *Reduction Rule 4.3 is safe.*

Proof. Assume that $(G, (V_r, V_b), k, d)$ is a YES instance. Let F be its solution and it contracts G to G/F via mapping ψ. Suppose $C_1, C_2, \ldots, C_{d+1}$ are connected components of $G[V_r]$ such that $C_i \cap N(v) \neq \emptyset$ for $i \in [d + 1]$. For every i, consider a vertex, say u_i, in $C_i \cap N(v)$. Let $U = \{u_1, u_2, \ldots, u_{d+1}\}$. Define $\psi(U) = \bigcup_{u \in U} \psi(u)$. For $i, j \in [d + 1]$, $i \neq j$ implies $\psi(u_i) \neq \psi(u_j)$ as C_i and C_j are two different connected components of $G[V_r]$ and $V(F) \subseteq V_r$. This implies $|\psi(U)| = |U| = d + 1$. As $V(F) \subseteq V_r$ and $v \in V_b$, F does not contain an edge incident to v. Hence, $\psi(v) \neq \psi(u_i)$ for any $i \in [d+1]$. As $\psi(U) \subseteq N_{G/F}(\psi(v))$ and $|\psi(U)| \geq d+1$, vertex $\psi(v)$ is adjacent with $d+1$ or more vertices in G/F. This contradicts the fact that the maximum degree of vertices in G/F is at most d. Hence, our assumption was wrong and $(G, (V_r, V_b), k, d)$ is a No instance. ◀

The algorithm exhaustively applies the reduction rules mentioned above. On a reduced instance, the algorithm creates multiple instances using the following subroutine. For an instance $(G, (V_r, V_b), k, d)$, a subset R of V_r, and a $(d+1)$-coloring of R, the subroutine creates a new instance by contracting each colored component of R into a single vertex, and (re-)label it blue. We need the notion of 'valid coloring' to filter out colorings that will not produce a 'smaller' instance. For graph H, a vertex coloring $\phi : V(H) \to [d+1]$ is said to be a *valid coloring* if every monochromatic connected component is of size at least two. We now describe the subroutine.

Subroutine `Colorwise-Contraction`

This subroutine takes as an input an instance $(G, (V_r, V_b), k, d)$ of LABELED-MDC, a non-empty subset R of V_r, and a valid coloring ϕ of $G[R]$. It returns another instance of LABELED-MDC. It initializes $G' = G$, $V_r' = V_r$, $V_b' = V_b$, and $k' = k$. For a monochromatic connected component C of $G[R]$, the subroutine finds a spanning tree of $G[C]$ and contracts all edges in it. Let v_C be the vertex obtained at the end of this series of edge contractions. It updates $V_r' = V_r \setminus C$, $V_b' = V_b \cup \{v_C\}$ and reduces k by $|C| - 1$. The subroutine repeats this procedure for every monochromatic connected component of $G[R]$. It returns $(G', (V_r', V_b'), k', d)$ as instance of LABELED-MDC. This completes the description of the subroutine.

It is easy to verify that (V_r', V_b') is a partition of $V(G')$. As ϕ is a valid coloring of $G[R]$, a union of spanning trees of all monochromatic connected components of $G[R]$ contains at least $|R|/2$ edges. Hence, the subroutine contracts at least $|R|/2$ edges. This small observation will be helpful to get a bound on the running time of the algorithm.

▶ Remark 18. $k' \leq k - |R|/2$.

Let $\texttt{CC}[(G, (V_r, V_b), k, d); R; \phi]$ denote the instance returned by the subroutine when the input is $(G, (V_r, V_b), k, d)$, R, and ϕ. In the following lemma, we prove if the original instance is a YES instance than at least one of the reduced instances is a YES instance.

▶ **Lemma 19.** *Consider a* YES *instance* $(G, (V_r, V_b), k, d)$ *of* LABELED-MDC. *Let* R *be a union of some connected components of* $G[V_r]$. *Suppose there is solution* F *to* $(G, (V_r, V_b), k, d)$ *such that* $R \subseteq V(F)$. *Then, there is a valid coloring* $\phi : R \to [d+1]$ *of* $G[R]$ *for which* $\texttt{CC}[(G, (V_r, V_b), k, d); R; \phi]$ *is a* YES *instance.*

Proof. Let $H = G/F$. Consider the H-witness structure $\mathcal{W}$ of G and let G be contracted to H via ψ. Define a subset $\mathcal{W}_R$ of $\mathcal{W}$ as the collection of witness sets that intersects R. Formally, $\mathcal{W}_R = \{W \in \mathcal{W} \mid W \cap R \neq \emptyset\}$. Let $\mathcal{W}_R = \{W_1, W_2, \dots, W_q\}$. For every $i \in [q]$, let h_i be the vertex corresponding to W_i. In other words, $W_i = \{v \in V(G) \mid \psi(v) = h_i\}$. Let $R_H = \{h_1, h_2, \dots, h_q\}$.

Let F_1 be the collection of edges in F that are incident to some vertex in R. Hence, $R \subseteq V(F_1)$. As R is the union of connected components in $G[V_r]$ and $V(F_1) \subseteq V(F) \subseteq V_r$, we can conclude that $R = V(F_1) = \bigcup_{i \in [q]} W_i$. Hence, $\{W_1, W_2, \dots, W_q\}$ is a partition of R. As there is a solution edge incident to every vertex in R, every witness set in $\mathcal{W}_R$ is a big witness set. This implies for every $i \in [q]$, there is a subset F_i of F such that $W_i = V(F_i)$. As the maximum degree of vertices in graph H is at most d, there is a proper $(d+1)$-coloring, say γ, of H. For $i, j \in [q]$, if (h_i, h_j) is an edge in H then $\gamma(h_i) \neq \gamma(h_j)$. Define a coloring $\phi : R \to [d+1]$ as follows. For $v \in R$, $\phi(v) = \gamma(h_i)$ where $v \in W_i$. As $\{W_1, W_2, \ldots, W_q\}$ is a partition of R, function ϕ is well defined. Since W_i is a big witness set, ϕ is a valid coloring.

By the construction of ϕ, any witness set in $\mathcal{W}$ is monochromatic. Since γ is a proper coloring of H, any two witness sets adjacent to each other have distinct colors. Hence, every witness set in $\mathcal{W}_R$ is a monochromatic connected component of coloring ϕ. As algorithm constructs every valid coloring of R, it also consider this coloring and create instance $(G', (V'_r, V'_b), k', d) = \texttt{CC}[(G, (V_r, V_b), k, d), R; \phi]$. For every $i \in [q]$, let F°_i be edges in a spanning tree of $G[W_i]$. Define $F^\circ = \bigcup_{i \in [q]} F^\circ_i$. As $W_i = V(F_i)$, graphs G/F_1 and G/F° are identical. Also, $|F^\circ_i| \le |F_i|$ which implies $|F^\circ| \le |F_1|$. Define $F^\star = (F \setminus F_1) \cup F^\circ$. It is easy to verify that $F^\star$ is also a solution to $(G, (V_r, V_b), k, d)$.

We now argue that $F^\star \setminus F^\circ$ is a solution to $(G', (V'_r, V'_b), k', d)$. By the description of the algorithm, $k' = k - |F^\circ|$. As $|F^\star| \le |F| \le k$ and $F^\circ \subseteq F^\star$, we have $|F^\star \setminus F^\circ| \le |F^\star| - |F^\circ| \le k'$. Note that $G/F^\star = (G/F^\circ)/(F^\star \setminus F^\circ) = G'/(F^\star \setminus F^\circ)$ as $G' = G/F^\circ$. This implies the maximum degree of $G'/(F^\star \setminus F^\circ)$ is at most d. The only thing that remains to argue is that $V(F^\star \setminus F^\circ)$ is contained in V'_r. By construction, $F \setminus F_1 = F^\star \setminus F^\circ$. As F_1 is the set of edges in F that were incident to R, we can conclude that no edge in $F^\star \setminus F^\circ$ is incident to R. Recall that $V'_r = V_r \setminus R$. Hence, $V(F^\star \setminus F^\circ) \subseteq V'_r$. This implies that $F^\star \setminus F^\circ$ is a solution to $(G', (V'_r, V_b), k', d)$ and concludes the proof of the lemma. ◀

In the above lemma, instead of considering any arbitrary subset V_r we only consider a subset that is a union of one or more connected components of $G[V_r]$. This suffices for our purpose as the algorithm calls the subroutine only on such subsets of V_r. Also, note that we do not need to know the solution F explicitly to apply the above lemma. It suffices to know that such a solution exists. We are now able to present an algorithm for Labeled-MDC

Algorithm for Labeled-MDC

The algorithm takes as input an instance $(G, (V_r, V_b), k, d)$ of Labeled-MDC and returns Yes or No. If $k < 0$ then the algorithm returns No. If $k = 0$ then it finds the maximum degree of G. If it is at most d then the algorithm returns Yes otherwise it returns No. The algorithm exhaustively applies Reduction Rules 4.1, 4.2, and 4.3. If the reduced instance is a trivial Yes (resp. No) instance then the algorithm returns Yes (resp. No). Otherwise, it creates multiple instances and makes recursive calls on these instances. The algorithm returns Yes if one of the recursive calls returns Yes, otherwise; it returns No.

We now describe the procedure used by the algorithm to create new instances. Let $(G, (V_r, V_b), k, d)$ be the instance on which reduction rules are not applicable. The algorithm finds a vertex, say v, in G such that $\deg_G(v) \ge d + 1$. It considers the following two cases.

1. (Vertex v is in V_r) Let R be the connected component of $G[V_r]$ that contains v. The algorithm constructs all valid coloring $\phi : R \to [d+1]$ of $G[R]$. For each coloring, the algorithm calls subroutine `Colorwise-Contraction` with input $(G, (V_r, V_b), k, d)$, R, and ϕ. The algorithm calls itself with the instances returned by this subroutine as the input.
2. (Vertex v is in V_b) Let $C_1, C_2, \ldots, C_q$ be the connected components of $G[V_r]$ such that $N(v) \cap C_i \neq \emptyset$ for every $i \in [q]$. For a non-empty subset $I \subseteq [q]$, define $R_I := \bigcup_{i \in I} C_i$. For every non-empty subset $I \subseteq [q]$, the algorithm proceeds as follows. If $|R_I| \ge 2k + 1$, the algorithm discards this choice of I and moves to the next one. Otherwise, the algorithm constructs all valid colorings $\phi : R_I \to [d+1]$ of $G[R_I]$. For each coloring, the algorithm calls subroutine `Colorwise-Contraction` with input $(G, (V_r, V_b), k, d)$, R_I, and ϕ. The algorithm calls itself with the instance returned by this subroutine as input.

This completes the description of the algorithm.

In the following lemma, we prove that the algorithm described above is correct and runs in the desired time.

▶ **Lemma 20.** *There is an algorithm that given an instance $(G,(V_r,V_b),k,d)$ of* LABELED-MDC *runs in time $2^{(d+2)k} \cdot (d+1)^{2k} \cdot n^{\mathcal{O}(1)}$ and correctly determines whether it is a* YES *instance.*

Proof. We argue that the algorithm described above solves LABELED-MDC in the desired time. We prove this lemma by the induction on the solution size k.

Consider the base case when the solution size is zero. Here, the algorithm finds a maximum degree of the graph and depending on its value returns YES or NO. It is easy to see that the lemma holds in this case. Assume that the lemma is true when the solution size is at most $k-1$.

We first prove that given a YES instance the algorithm returns YES. Suppose $(G,(V_r,V_b),k,d)$ is a YES instance of LABELED-MDC and let F be its solution. Note that this implies that F is a solution to (G,k,d). If the algorithm returned YES because Reduction Rule 4.1 returned a YES instance then the lemma is vacuously true. By Lemma 17, Reduction Rule 4.3 is not applicable on the input. Consider the instance obtained by the exhaustive application Reduction Rules 4.1 and 4.2 on the input instance. For notational convenience, we denote this reduced instance by $(G,(V_r,V_b),k,d)$. As Reduction Rule 4.1 is not applicable, there is a vertex in G that has degree at least $d+1$. Let v be the vertex of degree at least $d+1$ found by the algorithm. By Observation 2.5, $V(F)$ intersects with $N[v]$.

Consider the case when v is in V_r and let R be the connected component of $G[V_r]$ that contains v. Since $V(F) \subseteq V_r$, we have $R \cap V(F) \neq \emptyset$. As F is a solution to $(G,(V_r,V_b),k,d)$, $R \cap V(F) \neq \emptyset$ implies $R \subseteq V(F)$. Instance $(G,(V_r,V_b),k,d)$, subset R of V_r, and solution F satisfies the premise of Lemma 19. Hence, there is a valid coloring $\phi : R \to [d+1]$ of $G[R]$ such that $\texttt{CC}[(G,(V_r,V_b),k,d),R;\phi]$ is a YES instance. As $R \neq \emptyset$, Remark 18 implies that $k' < k$. By the induction hypothesis, the algorithm correctly returns YES when the input is $(G',(V_r',V_b'),k',d)$. As one of the recursive calls returns YES, the algorithm returns YES when the input is $(G,(V_r,V_b),k,d)$ and v is in V_r.

Consider the case when v is in V_b. Let $C_1,C_2,\ldots,C_q$ be the connected components of $G[V_r]$ such that $N(v) \cap C_i \neq \emptyset$ for every $i \in [q]$. Recall that for a non-empty subset $I \subseteq [q]$, $R_I = \bigcup_{i \in I} C_i$. As $V(F)$ intersects $N[v]$ and $V(F) \subseteq V_r$, there exists a non-empty subset $I' \subseteq [q]$ such that for $i \in [q]$, $C_i \cap N(v) \neq \emptyset$ if and only if $i \in I'$. As F is a solution to $(G,(V_r,V_b),k,d)$, $C_i \cap V(F) \neq \emptyset$ implies $C_i \subseteq V(F)$. Hence, $R_{I'} \subseteq V(F)$. As $|V(F)| \leq 2k$, $|R_{I'}| \leq 2k$. For every non-empty subset $I \subseteq [q]$ for which $|R_I| \leq 2k$, the algorithm constructs all valid coloring $\phi : R_I \to [d+1]$ of $G[R_I]$ and calls `Colorwise-Contraction`. Instance $(G,(V_r,V_b),k,d)$, subset $R_{I'}$ of V_r, and solution F satisfies the premise of Lemma 19. Hence, there is a valid coloring $\phi : R_{I'} \to [d+1]$ of $G[R_{I'}]$ such that $(G',(V_r',V_b'),k',d) = \texttt{CC}[(G,(V_r,V_b),k,d),R_{I'},\phi]$ is a YES instance. As $R \neq \emptyset$, Remark 18 implies that $k' < k$. By the induction hypothesis, the algorithm correctly returns YES when the input is $(G',(V_r',V_b'),k',d)$. As one of the recursive calls returns YES, the algorithm returns YES when the input is $(G,(V_r,V_b),k,d)$ and v is in V_b. This implies that if $(G,(V_r,V_b),k,d)$ is a YES instance then the algorithm returns YES.

We now prove that if the algorithm returns YES on instance $(G,(V_r,V_b),k,d)$ then it is a YES instance of LABELED-MDC. If the algorithm returned YES because Reduction Rule 4.1 returned a YES instance then the lemma is vacuously true. Otherwise, there is a newly created instance, say $(G',(V_r',V_b'),k',d)$, on which the recursive call of the algorithm returned

YES. Let R be the subset of V_r and ϕ be its valid coloring such that `Colorwise-Contraction` returned this instance when input was $(G, (V_r, V_b), k, d)$, R, and ϕ. Let F° be the edges in G contracted by the subroutine to contract G'. In other words, F° is a collection of spanning trees of connected monochromatic components of $G[R]$. Note that $|F^{\circ}| = k - k'$. The algorithm calls `Colorwise-Contraction` only on non-empty subsets R. Hence, by Remark 18, $k' < k$. By the induction hypothesis, $(G', (V'_r, V'_b), k', d)$ is a YES instance of LABELED-MDC. It is easy to see that if F' is a solution to $(G', (V'_r, V'_b), k', d)$ then $F' \cup F^{\circ}$ is a solution to $(G, (V_r, V_b), k, d)$. This concludes the proof of the correctness of the algorithm.

We now bound the running time of the algorithm. The algorithm can apply all the reduction rules in polynomial time. It creates new instances only when none of the reduction rules are applicable. As Reduction Rules 4.2 is not applicable, any connected component of $G[V_r]$ has at least two and at most $2k$ vertices. In Case(1), the algorithm creates at most $(d+1)^{|R|}$ many instances. By Remark 18 and the induction hypothesis, the time taken by the algorithm in this case is

$$(d+1)^{|R|} \cdot 2^{(d+2)(k-|R|/2)} \cdot (d+1)^{2(k-|R|/2)} \cdot n^{\mathcal{O}(1)} \leq 2^{(d+2)k} \cdot (d+1)^{2k} \cdot n^{\mathcal{O}(1)}.$$

As Reduction Rule 4.3 is not applicable, for any vertex v in V_b, there are at most d connected components of $G[V_r]$ that intersects $N(v)$. In Case(2), the algorithm constructs all valid partitions of R_I only when $|R_I| \leq 2k$. Hence, in this case, the algorithm creates $2^d \cdot (d+1)^{|R|}$ many instances. By Remark 18 and the induction hypothesis, the time taken by the algorithm in this case is

$$2^d \cdot (d+1)^{|R|} \cdot 2^{(d+2)(k-|R|/2)} \cdot (d+1)^{2(k-|R|/2)} \cdot n^{\mathcal{O}(1)} \leq 2^{(d+2)k} \cdot (d+1)^{2k} \cdot n^{\mathcal{O}(1)}.$$

As $|R| \geq 2$, we have $2^d \cdot 2^{(d+2)(-|R|/2)} \leq 1$. This completes the proof of the lemma. ◀

The correctness of Theorem 2 immediately follows from Lemma 16 and Lemma 20.

5 No Polynomial Kernel

In this section, we prove that MAXIMUM DEGREE CONTRACTION does not admit a polynomial kernel when parameterized by $k + d$. To show that, we present a reduction from RED BLUE DOMINATING SET (RBDS). In this problem, an input is comprised of a bipartite graph H with a bipartition (R, B) of $V(H)$, and a positive integer l. The question is, does there exist a subset R' of R of size at most l such that $N(R') = B$? Without loss of generality, we can assume that $l + 3 < |B|$ and no vertex in R is adjacent to all but one vertices in B. We know the following result about the compression of the problem. See, for example, Theorem 15.18 in [14].

▶ **Proposition 21.** *Unless* NP $\subseteq$ coNP/*poly*, RBDS, *parameterized by* $|B|$, *does not admit a polynomial compression.*

If $|R| > 2^{|B|}$ then there are at least two different vertices, say r_1, r_2 such that $N(r_1) = N(r_2)$. It is easy to see that it is safe to delete one of these two vertices. In this case, we can ensure, in polynomial time, that $|R| \leq 2^{|B|}$ by repeating the above process. This implies $\log_2 |R| \leq |B|$. Hence, we get the following corollary of Proposition 21.

▶ **Corollary 22.** *Unless* NP $\subseteq$ coNP/*poly*, RBDS, *parameterized by* $|B| + \log_2 |R|$, *does not admit a polynomial compression.*

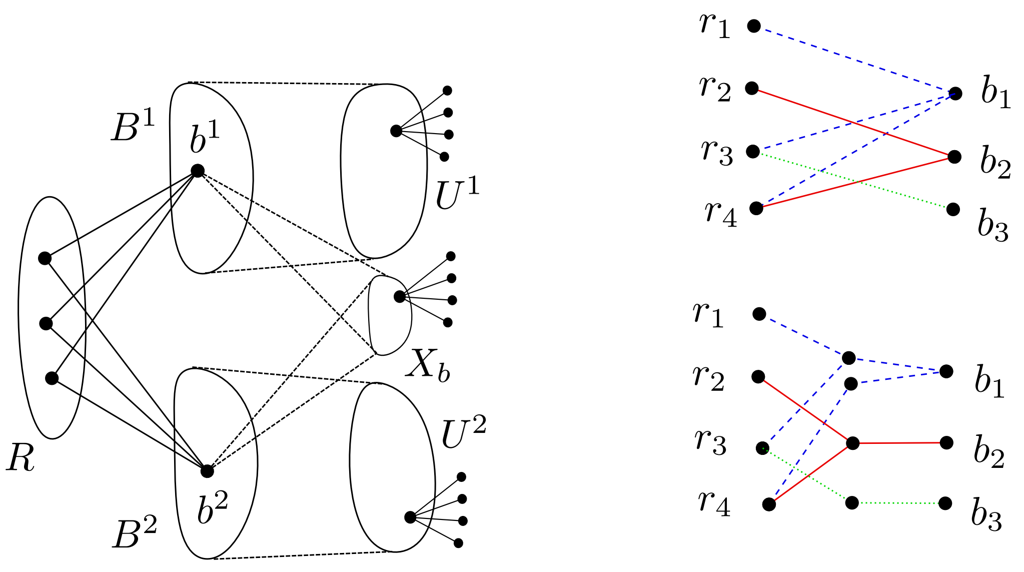

Figure 2 (Left) Overview of the reduction. The doted lines indicate that there is a complete bipartite graph across two sets. (Right) The operation of replacing edges incident to vertex in B by a tree rooted at that vertex.

For the sake of clarity, we use both $|B|$ and $\log_2 |R|$ as parameters instead of replacing $\log_2 |R|$ by the larger parameter $|B|$. For notational convenience, we assume that $\log_2 |R|$ is an integer. If this is not the case, one can add some isolated vertices in R to ensure that $\log_2 |R|$ is an integer. This results in at most doubling of the number of vertices in it.

We first present an overview of the reduction. Consider an instance (H, R, B, l) of RBDS. See Figure 2 for an illustration. The reduction makes a copy of R and two copies of B, say B^1, B^2. For every vertex b in B, we denote its two copies in B^1, B^2 by b^1, b^2, respectively. For every edge (r, b), the reduction adds edges (r, b^1) and (r, b^2). It adds two independent sets U^1, U^2. For every vertex $u \in U^1 \cup U^2$, it adds some pendent vertices adjacent to it. The reduction adds all edges to make a complete bipartite graph with (B^1, U^1) as its bipartition. Similarly, it adds all edges to make a complete bipartite graph with (B^2, U^2) as its bipartition. For every vertex b in B, it adds a set of independent vertices X_b. For every x in X_b, it adds some pendent vertices adjacent to it and adds edges (b^1, x), (b^2, x). We briefly present an intuition behind the construction before presenting the last step. Let G be the graph constructed so far and k, d be two integers whose values depend only on $|B|, \log_2 |R|$. Suppose the reduction returns (G, k, d) as an instance of MDC.

We set the value of d and the number of pendant vertices such that it is ensured that the only vertices in $U^1 \cup U^2 \cup X_b$ have degree more than d in G. We fix k and the sizes of sets U^1, U^2, X_b to ensure that any solution for the reduced instance of MDC satisfy the following properties.

1. It does not include an edge with one of its endpoints in $B^1 \cup B^2$ and another in $U^1 \cup U^2$.
2. For any b in B, it does not include an edge with one of its endpoints in $\{b^1, b^2\}$ and another in X_b.
3. It spans all vertices in $B^1 \cup B^2$. In other words, $B^1 \cup B^2 \subseteq V(F)$.
4. There are at most l witness sets in the G/F-witness structure of G that contain vertices

in B^1 (similarly in B^2).

5. For every b in B, F includes b^1, b^2 in the same witness set.

Property (4) ensures that the degree constraints for the vertices in U^1 (similarly in U^2) are satisfied. Property (5) ensures that for every b in B, the degree constraints for the vertices in X_b are satisfied. Because of Property (1) and (2), only the vertices in R can make a witness set connected. Hence, each witness set should contain at least one vertex from R. We set the budget k such that each witness set contains exactly one vertex from R. To prove connectivity to witness set, this vertex needs to be adjacent to all vertices in that witness set. Hence, the set of endpoints of edges in a solution to (G, k, d) contains at most l vertices in R that dominates B. This naturally leads to a solution to $(H.R, B, l)$.

We now present the last step in the construction. The degree of the vertices formed by contracting a witness set can be larger than d. To avoid this, we replace edges across R, B that are incident to vertex b in B by a binary tree rooted at that vertex. We ensure that for every edge incident b, there is a unique root-to-leaf path in the binary tree rooted at b and vice versa.

Reduction

Given an instance (H, R, B, l) of RBDS as an input, the reduction outputs an instance (G, k, d) of MDC. The reduction creates an intermediate instance $(H^\star, R^\star, B^\star, l)$ of RBDS. It takes a copy of R and two copies of B, namely B^1, B^2, to create vertex set of graph $H^\star$. Formally, $R^\star = R$ and $B^\star = B^1 \cup B^2$. For every edge (r, b) in H such that $r \in R$ and $b \in B$, it adds edges (r, b_1), (r, b_2) to $H^\star$. Here, vertices b^1, b^2 are copies of b in B^1, B^2, respectively. It is easy to see that (H, R, B, l) is a YES instance of RBDS if and only if $(H^\star, R, B^1 \cup B^2, l)$ is a YES instance.

The reduction sets $k = 2|B| \cdot \log_2 |R|$ and $d = 2|B| \cdot (\log_2 |R| + k + 2)$, and constructs graph G by modifying a copy of graph H° in the following way. It repeats the first three steps for $i = 1$ and $i = 2$.

- For every vertex b^i in B^i, it deletes all the edges incident to b^i and constructs a binary tree that satisfies the following conditions: (i) the tree is rooted at b^i, (ii) the height of the binary tree is $\log_2 |R|$, and (ii) its leaves are the vertices in $N_{H^\star}(b^i)$. Note that every edge incident to b^i in $H^\star$ corresponds to an unique root-to-leaf path in this binary tree and vice versa. Let I^i be the collection of all the new vertices added in this step. It is the collection of vertices in binary trees that are not roots or leaves.
- It adds a set U^i of $k+1$ new vertices to $V(G)$. For every vertex b^i in B^i and every vertex u in U^i, it adds edge (b^i, u). It adds all edges to make $G[B^i \cup U^i]$ a complete bipartite graph with B^i, U^i as its bipartition.
- For every vertex u in U^i, it adds $d - l$ pendant vertices adjacent to u.
- For every vertex b in B, it adds a set X_b of $k+1$ new vertices. For every vertex x in X_b, it adds edges (b^1, x) and (b^2, x). It also adds $d-2$ pendant vertices adjacent to x. Let P_X^b be the set of pendent vertices adjacent to some vertex in X_b.

This completes the construction of (G, k, d).

The following lemma identifies the set of vertices in G that has degree more than d.

▶ **Lemma 23.** *Suppose the reduction returns (G, k, d) when the input is (H, R, B, l). Then, $U \cup X$ is the collection of all vertices in G that has degree strictly greater than d. Here, $U = U^1 \cup U^2$ and $X = \bigcup_{b \in B} X_b$.*

Proof. Define $I := I^1 \cup I^2$ and $P := P^1 \cup P^2 \cup \left(\bigcup_{b \in B} P_X^b\right)$. Note that sets R, I, B^1, B^2, U, X, P forms a partition of $V(G)$. Consider a vertex r in R. This vertex is a leaf in the binary trees rooted at every vertex in $N_{H^\star}(r)$. Hence, in graph G, any vertex in R is adjacent to at most $2|B| = |B^1 \cup B^2|$ many vertices in I. Every vertex in I is an internal vertex in a binary tree and hence adjacent to at most three vertices. Every vertex in B^1 is adjacent to at most two vertices from set I, all the vertices in set U^1 and all the vertices in X_b. Hence, vertex in B^1 is adjacent with $2 + |U^1| + |X_b|$ many vertices. As $|U^1| = |X_b| = k + 1$, every vertex in B^1 is adjacent to at most $2k + 4 \leq d$ vertices. By similar arguments, every vertex in B^2 is adjacent to at most d vertices. As P is a collection of pendant vertices, every vertex in it is adjacent with exactly one vertex.

We have proved that every vertex in $V(G) \setminus (U \cup X)$ has degree at most d. It remains to prove that every vertex in $U \cup X$ has degree at least d. Every vertex in U^1 is adjacent with $d - l$ pendant vertices and every vertex in B^1. As $|B| > l$, every vertex in U^1 has degree strictly greater than d. By similar arguments, every vertex in U^2 has degree strictly greater than d. For every b in B, every vertex in X_b is adjacent with $d - 1$ pendant vertices and two vertices in $B^1 \cup B^2$. Hence, every vertex in X is adjacent with at least $d + 1$ vertices. This concludes the proof of the lemma. ◀

In the remaining section, we argue that the reduction is correct. In Lemma 24 and Lemma 25 we prove the forward and backward directions, respectively.

▶ **Lemma 24.** *Suppose the reduction returns (G, k, d) when the input is (H, R, B, l). If (H, R, B, l) is a* Yes *instance of* RBDS *then (G, k, d) is a* Yes *instance of* MDC.

Proof. As mentioned before, (H, R, B, l) is a Yes instance of RBDS if and only if $(H^\star, R, B^1 \cup B^2, l)$ is a Yes instance of RBDS. Let $R' = \{r_1, r_2, \dots, r_{|R'|}\}$ be a subset of R of size at most l such that $B^1 \cup B^2 = N_{H^\star}(R)$. Without loss of generality, we assume that R' is a minimal dominating set. Partition $B^1 \cup B^2$ into $B_1, B_2, \dots, B_{|R'|}$ such that for every $j \in [|R'|]$, r_j dominates vertices in B_j and for every b in B, vertices b^1, b^2 are in same part. Since R' is a minimal dominating set, B_j is a non-empty. For every $j \in [|R'|]$, define F_j as follows: Initialize F_j to an empty set. For every b in B_j, the consider binary tree rooted at b^1 (similarly at b^2). Add the root-to-leaf paths in the trees that correspond to edges (r_j, b^1) and (r_j, b^2) to F_j. Let F be the union of all F_js. Formally, $F = \bigcup_{j \in [|R'|]} F_j$. Since every path is of length $\log_2 |R|$ and any two paths in F are edge-disjoints, $|F| = 2|B| \cdot \log_2 |R|$. Consider the graph G/F and let G is contracted to G/F via function ψ. Define $B^\circ = \{b_1^\circ, b_2^\circ, \dots, b_{|R'|}^\circ\}$, where b_j° is the vertex in G/F which is obtained by contracting all edges in F_j. We argue that the maximum degree of vertices in G/F is at most d.

Vertices in $V(G/F)$ can be partitioned into $R \setminus R' = R \setminus V(F)$, $I \setminus V(F)$, B°, U, X, and P. Here, I, U, X, P are the sets defined in Lemma 23. For every vertex in $(R \cup I \cup R) \setminus V(F)$, we have $|W(\psi(v))| = 1$. By Observation 2.1 (3), $\deg_{G/F}(\psi(v)) \leq \deg_G(v)$. By Lemma 23, every vertex in $R \cup I \cup R$ has degree at most d. Hence, we can conclude that $\deg_{G/F}(\psi(v)) \leq d$. In graph G/F, every vertex v in U is adjacent to every vertex in B° and $d - l$ pendant vertices. As $|B^\circ| = |R'| \leq l$, every vertex in U is adjacent to at most d vertices. As b^1, b^2 are in same witness set for every b in B, every vertex in X is adjacent to one vertex in B° and $d - 1$ pendent vertices. Hence, every vertex in X has degree at most d in G/F.

It remains to argue that the degree of b_j° is at most d in G/F. For every j in $[|R'|]$, set $V(F_j)$ can be partitioned into the following three parts: $(i) V(F_j) \cap R$, $(ii) V(F_j) \cap I$, and $(iii) B_j = V(F_j) \cap (B^1 \cup B^2)$. Consider the first part. By construction, $V(F_j) \cap R = \{r_j\}$. As mentioned in the proof of Lemma 23, in graph G, any vertex in R is adjacent to at most $|B^1 \cup B^2|$ vertices in I. Hence, the vertex in $V(F_j) \cap R$ is adjacent to at most

$|B^1 \cup B^2| - |B_j|$ vertices outside $V(F_j)$. Now consider the second part. Every vertex in $V(F_j) \cap I$ is adjacent to at most one vertex outside $V(F_j)$. As every r_j to b^1 (similarly r_j to b^2) path is of length $\log_2 |R|$, and any two paths in F_j are edge-disjoints, $|V(F_j) \cap I| = |B_j| \cdot \log_2 |R|$. Hence, vertices in $V(F_j) \cap I$ are adjacent to at most $|B_j| \cdot \log_2 |R|$ vertices outside $V(F_j)$. Now consider the third part. Every vertex in $V(F_j) \cap (B^1 \cup B^2)$ is adjacent to at most one vertex in $I \setminus V(F_j)$, every vertex in U and every vertex in X_j. Here, $X_j = \bigcup_{b^1 \in B_j} X_b = \bigcup_{b^2 \in B_j} X_b$. Hence, vertices in $V(F_j) \cap (B^1 \cup B^2)$ are adjacent to at most $|B_j| + (k+1) + |B_j|(k+1)$. This implies that the number of vertices adjacent to $V(F_j)$ is at most $|B^1 \cup B^2| - |B_j| + |B_j| \cdot \log_2 |R| + |B_j| + (k+1) + |B_j| \cdot (k+1) \leq 2|B^1 \cup B^2| \cdot (\log_2 |R| + k + 2) = d$. Here, we use the fact that $1 + |B_j| \leq |B^1 \cup B^2|$. This follows from our assumption that in graph H, no vertex in R is adjacent to all but one vertex in B. Hence, the degree of any vertex in B° is at most d in G/F.

This prove that the maximum degree of any vertex in G/F is at most d. Hence, if (H, R, B, l) is a YES instance, then so is (G, k, d). ◀

We now prove the backward direction. As in Section 3, we prove a series of claims about a solution to reduced instance. We prove the five properties mentioned at the start of this section to prove the following lemma.

▶ **Lemma 25.** *Suppose the reduction returns (G, k, d) when the input is (H, R, B, l). If (G, k, d) is a* YES *instance of* MDC *then (H, R, B, l) is a* YES *instance of* RBDS.

We prove that if (G, k, d) is a YES instance of MDC then $(H^\star, R, B^1 \cup B^2, l)$ is a YES instance of RBDS. Recall that for vertex subset X, Y, we denote the set of all edges with one endpoint in X and another endpoint in Y by $E(X, Y)$. Let P_U^1, P_U^2 and P_X be the collection of pendent vertices adjacent to vertices in U^1, U^2, and X, respectively. By construction, we can partition edges of G into the following four sets: $E(B^1 \cup B^2, U^1 \cup U^2)$, $E(B^1 \cup B^2, X)$, $E(U \cup X, P_U^1 \cup P_U^2 \cup P_X)$, and E'. Here, E' is the collection of edges that are not covered by the first three sets.

Suppose (G, k, d) is a YES instance and F is a solution to (G, k, d).

▷ Claim 26. $F \cap E(B^1 \cup B^2, U^1 \cup U^2) = \emptyset$.

Proof. Assume that there is an edge, say (b^1, u), in $F \cap E(B^1, U^1)$ where vertices b^1, u are in B^1 and U^1, respectively. Let w be the new vertex introduced while contracting edge (b^1, u). In graph $(G/\{(b^1, u)\}$, vertex w is adjacent to every vertex in $B^1 \setminus \{b^1\} \cup X_b \cup (U^1 \setminus \{u\})$ and with all pendent vertices that were adjacent with u in G. Hence, the degree of w in $(G/\{(b^1, u)\})$ is at least $|B| - 1 + |X_b| + |U^1| + d - l > d + (k-1) + 1$. By Observation 2.5, $(G/\{(b^1, u)\}, k-1, d)$ is a NO instance. This contradicts Observation 2.2. Hence our assumption is wrong and $F \cap E(B^1, U^1)$ is an empty set. By similar arguments, $F \cap E(B^2, U^2)$ is an empty set. By construction, sets $E(B^1, U^2)$ and $E(B^2, U^1)$ are empty. This concludes the proof of the claim. ◀

▷ Claim 27. $F \cap E(B^1 \cup B^2, X) = \emptyset$.

Proof. To prove the claim, it suffices to prove that for any b in B, F does not include edge (b^1, x) where b^1 is in B^1 and x is in X_b. For the sake of contradiction, assume such an edge exists. Let w be the new vertex introduced while contracting edge (b^1, x). In graph $(G/\{(b^1, x)\}$, vertex w is adjacent to every vertex in $\{b^2\} \cup (X_b \setminus \{x\}) \cup U^1$ and with all pendent vertices that were adjacent with x in G. Hence, the degree of w in $(G/\{(b^1, u)\})$ is at least $1 + |X_b| + |U^1| + d - 1 > d + (k-1) + 1$. By Observation 2.5, $(G/\{(b^1, x)\}, k-1, d)$

is a No instance. This contradicts Observation 2.2. Hence our assumption is wrong and there is no edge of the form (b^1, x). As every edge in $E(B^1, X)$ is of the form (b^1, x) for some b^1 in B^1 and x in X_b, we can conclude that $E(B^1, X)$ is an empty set. By similar arguments, $E(B^3, X)$ is an empty set. This concludes the proof of the claim. ◀

▷ Claim 28. $(B^1 \cup B^2) \subseteq V(F)$.

Proof. Assume for the sake of contradiction that there is b^1 in $(B^1 \cup B^2) \setminus V(F)$. Recall that every vertex x in X_b, the degree of x is $d+1$. By Observation 2.5, there are at least two vertex in $N[x]$ which are in $V(F)$. As b^1 is not incident to any solution edge, b^1 is not in $N[x] \cap V(F)$. Vertex b^2 can be one of the vertices in $N[x] \cap V(F)$. By Claim 27, edge (b^2, x) is not in any solution. Hence, there is at least one vertex $N[x] \cap V(F)$ which is a pendent vertex adjacent to x or the vertex x itself. This implies for every x in X_b, there is a solution edge incident to pendent vertex adjacent to x. Hence, there are at least $|X_b| = k+1$ edges in F. This contradicts the fact that $|F|$ is at most k. Hence our assumption is wrong and $B^1 \cup B^2 \subseteq V(F)$. ◀

▷ Claim 29. There are at most l witness sets in the G/F-witness structure of G that contains vertices in B^1 (similarly in B^2).

Proof. Recall that for a subset $Z \subseteq V(G)$, we define $\psi(Z) := \{\psi(z) \mid z \in X\}$. To prove the claim, it suffices to prove that the size of $\psi(B^1)$ is at most l. Assume there is an integer $l' \geq 1$ such that $|\psi(B^1)| = l + l'$. For every vertex $u \in U$ in graph G, vertex $\psi(u)$ is adjacent to every vertex in $\psi(B^1)$ in graph G'. The degree of $\psi(u)$ in G/F is at most d. By Claim 26, F does not contain an edge in $E(B^1, U^1)$. Hence, F must contain at least l' many edges incident to u and pendent vertices adjacent to it. As this is true for every vertex in U^1, there are $|U^1| \cdot l'$ many edges in F that are incident to pendant vertices. As $|U^1| = k+1$ and $l' \geq 1$, this contradicts the fact that $|F| \leq k$. Hence, our assumption is wrong and $|\psi(B^1)| \leq l$. This implies the G/F-witness structure of G partitions all vertices in B^1 into at most l witness sets. By similar arguments, we can prove that the G/F-witness structure of G partitions all vertices in B^2 into at most l witness sets. ◀

▷ Claim 30. For every b in B, $\psi(b^1) = \psi(b^2)$.

Proof. Assume for the sake of contradiction that there is b in B, such that $\psi(b^1) \neq \psi(b^2)$. In other words, b^1, b^2 are in two different witness sets in G/F-witness structure of G. Recall that every vertex x in X_b, the degree of x is $d+1$. By Observation 2.5, there are at least two vertex in $N[x]$ which are in sane witness set. By Claim 27, no edge in $E(B^1 \cup B^2, X)$ is a part of any solution. Hence, for every x in X_b, there is a solution edge incident to pendent vertex adjacent to x. As $|X_b| = k+1$, this contradicts the fact that $|F|$ is at most k. Hence our assumption is wrong and for every b in B, $\psi(b^1) = \psi(b^2)$. ◀

We are now able to present a proof of Lemma 25. In the proof, we crucially use the fact that $G[R \cup I \cup B^1 \cup B^2]$ is a union of binary trees rooted at vertices in $B^1 \cup B^2$. Moreover, any two of these binary trees are edge disjoint.

Proof. (of Lemma 25) We prove that $(H^\star, R, B^1 \cup B^2, l)$ is a Yes instance of RBDS. By Claim 28 and 29, there is $l' (\leq l)$ witness sets, say $W_1, W_2, \ldots, W_{l'}$, in the G/F-witness structure of G such that their union contains $B^1 \cup B^2$. For $j \in [l']$, define $B_j^\circ = (B^1 \cup B^2) \cap W_j$. We divide proof of the lemma in two parts. First, we prove that for every b_α in B_j°, there is vertex r in W_j such that r is adjacent with b_α in $H^\star$. This implies that in graph $H^\star$, set

$\bigcup_{j \in [l']}(R \cap W_j)$ dominates $B^1 \cup B^2$. In the second part, we prove that there is at most one vertex in $R \cap W_j$. This proves that the dominating set is of size $l' \leq l$.

Define $\widetilde{E} := E(B^1 \cup B^2, U^1 \cup U^2) \cup E(B^1 \cup B^2, X)$. By Claim 26 and 27, solution F does not contain any edge in $\widetilde{E}$. For any two vertices $b_\alpha, b_\beta \in B^1 \cup B^2$, any b_α to b_β path in $G - \widetilde{E}$ contains a vertex in $N_{H^\star}(b_\alpha)$ and a vertex in $N_{H^\star}(b_\alpha)$. Fix an arbitrary vertex b^1 in B^1. By Claim 30, if b^1 is in W_j then b^2 is also in W_j. Hence, there are at least two vertices in B_j° which is a subset of W_j. As W_j is connected set in $G - \widetilde{E}$, it contains at least one vertex each from $N_{H^\star}(b_\alpha)$ and $N_{H^\star}(b_\beta)$. Hence, for any b_α in $B^1 \cup B^2$, if b_α is in W_j then there exists at least one vertex in W_j which is adjacent to b_α in $H^\star$. This implies that in graph $H^\star$, set $\bigcup_{j \in [l']}(R \cap W_j)$ dominates $B^1 \cup B^2$.

To prove the second part, we need to argue that we can partition F into $|B^1 \cup B^2|$ parts, each corresponding to a vertex in $B^1 \cup B^2$. For every vertex b_α in $B^1 \cup B^2$, let $\rho(b_\alpha)$ be the vertex in R such that (i) b_α and $\rho(b_\alpha)$ are in same witness set, and (ii) $\rho(b_\alpha)$ is the nearest vertex in the witness set that satisfy the first property. The arguments in previous the paragraph ensure that such vertex exits. Let P_α be a path from b_α to $\rho(b_\alpha)$ such that edges in P_α are in F. As $b_\alpha, \rho(b_\alpha)$ are in W_j, which is a witness set, such a path exists. Because of the second property, P_α is a root-to-leaf path in the binary tree rooted at b_α. This implies that the length P_α is $\log_2 |R|$ and for two different vertices b_α, b_β in $B^1 \cup B^2$, paths P_α and P_β are edge-disjoint. As $|F| = |B^1 \cup B^2| \cdot \log_2 |R|$, we can conclude that $\{E(P_\alpha) \mid b_\alpha \in B_1 \cup B_2\}$ is a partition of F.

We now prove the second part. Assume that there is W_j such that j, set $R \cap W_j$ contains two vertices, say r_1, r_2. As r_1, r_2 are in same witness sets, there is a r_1 to r_2 path whose edges are contained in F. By the construction of $H^\star$ and the fact that $F \cap \widetilde{E} = \emptyset$, there is vertex b_α in W_j such that r_1, r_2 are leaves in the binary tree rooted at b_α. Without loss of generality, let $r_1 = \rho(b_\alpha)$. As r_2 is a leaf, there is at least one edge in r_1 to r_2 path which is not contained in path P_α. This edge is not a part of P_β for any $b_\beta \neq b_\alpha \in B^1 \cup B^2$ as binary trees rooted at vertices in $B^1 \cup B^2$ are edge disjoints. This implies there is an edge in F which is not in path P_α for any b_α in $B^1 \cup B^2$. This contradictions the fact that $\{E(P_\alpha) \mid b_\alpha \in B_1 \cup B_2\}$ is a partition of F. Hence our assumption is wrong and for $j \in [l']$, set $R \cap W_j$ contains at most one vertex.

This implies that in graph $H^\star$, set $\bigcup_{j \in [l']}(R \cap W_j)$ is of size at most l and dominates $B^1 \cup B^2$. Hence $(H^\star, R, B^1 \cup B^2, l)$ is a YES instance. As mentioned before, it is easy to see that (H, R, B, l) is a YES if and only if $(H^\star, R, B^1 \cup B^2, l)$ is a YES instance. This concludes the proof of the lemma. ◀

We are now able to present a proof of Theorem 3 using Corollary 22, Lemma 24, and Lemma 25.

Proof. (of Theorem 3) Assume, for the sake of contradiction, MDC, parameterized by $k + d$ admits a polynomial-sized compression. This implies there is an algorithm, say $\mathcal{A}$, that given any instance (G, k, d) of MDC runs in time polynomial time and produces equivalent instance (I', k') of parameterized problem Π such that (i) (G, k, d) is a YES instance of MDC if and only if (I', k') is a YES instance of Π, and (ii) $|I'| + k' \leq (k + d)^c$, where c is a fixed constant. We construct a compression algorithm for RBDS using Algorithm $\mathcal{A}$ as a subroutine.

Consider Algorithm $\mathcal{B}$ that given an instance (H, R, B, l) of RBDS constructs an instance (G, k, d) of MDC as described in the reduction. Then, it calls Algorithm $\mathcal{A}$, as subroutine, on instance (G, k, d). Let (I', k') be the instance of Π returned by Algorithm $\mathcal{A}$. The algorithm returns (I', k') as a compressed instance.

The correctness of the algorithm $\mathcal{A}$, Lemma 24, and Lemma 25 implies that (H, R, B, l) is a YES instance of RBDS if and only if (I', k') is a YES instance of Π. Since (G, k, d) is the instance of MDC constructed by the reduction when input was (H, R, B, l), we have $k = 2|B| \cdot \log_2 |R|$ and $d = 2|B| \cdot (\log_2 |R| + k + 2)$. As, $|I'| + k' \leq (k + d)^c$, we have $|I'| + k' \leq (|B| + \log_2 |R|)^{c_0}$, where c_0 is a fixed constant. By the description of the reduction, it is easy to see that given instance (H, R, B, l), Algorithm $\mathcal{B}$ computes instance (G, k, d) in time polynomial in $|V(H)|$. Hence, the total running time of the algorithm is polynomial in the size of the input.

This implies RBDS, when parameterized by $|B| + \log_2 |R|$, admits a polynomial compression. But this contradicts Corollary 22. Hence, our assumption was wrong, which concludes the proof. ◀

6 Conclusion

In this article, we studied the MAXIMUM DEGREE CONTRACTION problem. We proved that a simple brute force algorithm for this problem is optimal under ETH. This lower bound also implies that the known FPT algorithm with running time $(d+k)^k \cdot n^{\mathcal{O}(1)}$ is also optimal under the same hypothesis. We compliment this result by presenting another FPT algorithm with running time $2^{\mathcal{O}(dk)} \cdot n^{\mathcal{O}(1)}$. While these two FPT algorithms are incomparable, our algorithm runs faster for smaller values of d, for which the problem still remains NP-Hard. We also prove that unless $\mathsf{NP} \subseteq \mathsf{coNP}/poly$, the problem does not admit a polynomial compression when parameterized by $k + d$.

Most of the $\mathcal{H}$-CONTRACTION problems do not admit a polynomial kernel under the same complexity conjecture. For some graph classes like trees, cactus, cliques, splits graphs, such negative results have been complimented by establishing a *lossy kernel* of polynomial size for these problems. There are also examples like CHORDAL CONTRACTION, S-CLUB CONTRACTION (for $s \geq 2$) for which we know that lossy kernel of polynomial size do not exist. We conclude this article with following open question: Does MAXIMUM DEGREE CONTRACTION admit a lossy kernel of polynomial size?

References

1. Akanksha Agarwal, Saket Saurabh, and Prafullkumar Tale. On the parameterized complexity of contraction to generalization of trees. *Theory of Computing Systems*, 63(3):587–614, 2019.
2. Akanksha Agrawal, Fedor V Fomin, Daniel Lokshtanov, Saket Saurabh, and Prafullkumar Tale. Path contraction faster than 2^n. *SIAM Journal on Discrete Mathematics*, 34(2):1302–1325, 2020.
3. Akanksha Agrawal, Lawqueen Kanesh, Saket Saurabh, and Prafullkumar Tale. Paths to trees and cacti. In *International Conference on Algorithms and Complexity*, pages 31–42. Springer, 2017.
4. Akanksha Agrawal, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. Split contraction: The untold story. *ACM Transactions on Computation Theory (TOCT)*, 11(3):1–22, 2019.
5. Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *Journal of the ACM (JACM)*, 42(4):844–856, 1995.
6. Takao Asano and Tomio Hirata. Edge-contraction problems. *Journal of Computer and System Sciences*, 26(2):197–208, 1983.
7. Balabhaskar Balasundaram, Shyam Sundar Chandramouli, and Svyatoslav Trukhanov. Approximation algorithms for finding and partitioning unit-disk graphs into co-k-plexes. *Optimization Letters*, 4(3):311–320, 2010.

8 Rémy Belmonte, Petr A. Golovach, Pim Hof, and Daniël Paulusma. Parameterized complexity of three edge contraction problems with degree constraints. *Acta Informatica*, 51(7):473–497, 2014.

9 Nadja Betzler, Robert Bredereck, Rolf Niedermeier, and Johannes Uhlmann. On bounded-degree vertex deletion parameterized by treewidth. *Discrete Applied Mathematics*, 160(1-2):53–60, 2012.

10 Hans L Bodlaender and Babette van Antwerpen-de Fluiter. Reduction algorithms for graphs of small treewidth. *Information and Computation*, 167(2):86–119, 2001.

11 Andries Evert Brouwer and Henk Jan Veldman. Contractibility and NP-completeness. *Journal of Graph Theory*, 11(1):71–79, 1987.

12 Leizhen Cai and Chengwei Guo. Contracting few edges to remove forbidden induced subgraphs. In *International Symposium on Parameterized and Exact Computation*, pages 97–109. Springer, 2013.

13 Zhi-Zhong Chen, Michael Fellows, Bin Fu, Haitao Jiang, Yang Liu, Lusheng Wang, and Binhai Zhu. A linear kernel for co-path/cycle packing. In *International Conference on Algorithmic Applications in Management*, pages 90–102. Springer, 2010.

14 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

15 Anders Dessmark, Klaus Jansen, and Andrzej Lingas. The maximum k-dependent and f-dependent set problem. In *International Symposium on Algorithms and Computation*, pages 88–97. Springer, 1993.

16 Rod G. Downey and Michael R. Fellows. *Fundamentals of Parameterized complexity*. Springer-Verlag, 2013.

17 Michael R Fellows, Jiong Guo, Hannes Moser, and Rolf Niedermeier. A generalization of nemhauser and trotter's local optimization theorem. *Journal of Computer and System Sciences*, 77(6):1141–1158, 2011.

18 Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006.

19 Fedor V. Fomin, Daniel Lokshtanov, Ivan Mihajlin, Saket Saurabh, and Meirav Zehavi. Computation of Hadwiger Number and Related Contraction Problems: Tight Lower Bounds. In *47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*, pages 49:1–49:18. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

20 Fedor V Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. *Kernelization: theory of parameterized preprocessing*. Cambridge University Press, 2019.

21 Robert Ganian, Fabian Klute, and Sebastian Ordyniak. On structural parameterizations of the bounded-degree vertex deletion problem. In *35th Symposium on Theoretical Aspects of Computer Science (STACS 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

22 Petr A Golovach, Marcin Kaminski, Daniël Paulusma, and Dimitrios M Thilikos. Increasing the minimum degree of a graph by contractions. *Theoretical computer science.*, 481:74–84, 2013.

23 Petr A Golovach, Pim van't Hof, and Daniël Paulusma. Obtaining planarity by contracting few edges. *Theoretical Computer Science*, 476:38–46, 2013.

24 Sylvain Guillemot and Dániel Marx. A faster fpt algorithm for bipartite contraction. *Information Processing Letters*, 113(22-24):906–912, 2013.

25 Spoorthy Gunda, Pallavi Jain, Daniel Lokshtanov, Saket Saurabh, and Prafullkumar Tale. On the parameterized approximability of contraction to classes of chordal graphs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2020)*, 2020 (To Appear).

26 Pinar Heggernes, Pim Van'T Hof, Daniel Lokshtanov, and Christophe Paul. Obtaining a bipartite graph by contracting few edges. *SIAM Journal on Discrete Mathematics*, 27(4):2143–2156, 2013.

27 Pinar Heggernes, Pim Van't Hof, Benjamin Lévêque, Daniel Lokshtanov, and Christophe Paul. Contracting graphs to paths and trees. *Algorithmica*, 68(1):109–132, 2014.

28 Christian Komusiewicz, Falk Hüffner, Hannes Moser, and Rolf Niedermeier. Isolation concepts for efficiently enumerating dense subgraphs. *Theoretical Computer Science*, 410(38-40):3640–3654, 2009.

29 R Krithika, Pranabendu Misra, and Prafullkumar Tale. An fpt algorithm for contraction to cactus. In *International Computing and Combinatorics Conference*, pages 341–352. Springer, 2018.

30 Ramaswamy Krithika, Pranabendu Misra, Ashutosh Rai, and Prafullkumar Tale. Lossy kernels for graph contraction problems. In *36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.

31 Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Slightly superexponential parameterized problems. *SIAM Journal on Computing*, 47(3):675–702, 2018.

32 Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. On the hardness of eliminating small induced subgraphs by contracting edges. In *International Symposium on Parameterized and Exact Computation*, pages 243–254. Springer, 2013.

33 Barnaby Martin and Daniël Paulusma. The computational complexity of disconnected cut and 2k2-partition. *Journal of combinatorial theory, series B*, 111:17–37, 2015.

34 Rolf Niedermeier. *Invitation to fixed-parameter algorithms*. Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press, 2006.

35 Naomi Nishimura, Prabhakar Ragde, and Dimitrios M Thilikos. Fast fixed-parameter tractable algorithms for nontrivial generalizations of vertex cover. *Discrete Applied Mathematics*, 152(1-3):229–245, 2005.

36 Saket Saurabh, Uéverton dos Santos Souza, and Prafullkumar Tale. On the parameterized complexity of grid contraction. In *17th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

37 Toshimasa Watanabe, Tadashi Ae, and Akira Nakamura. On the removal of forbidden graphs by edge-deletion or by edge-contraction. *Discrete Applied Mathematics*, 3(2):151–153, 1981.

38 Toshimasa Watanabe, Tadashi Ae, and Akira Nakamura. On the np-hardness of edge-deletion and-contraction problems. *Discrete Applied Mathematics*, 6(1):63–78, 1983.