Parameterized Complexity of Eulerian Deletion Problems

Marek Cygan[1], Dániel Marx[2], Marcin Pilipczuk[1], Michał Pilipczuk[1], and Ildikó Schlotter[3]⋆

[1] Institute of Informatics, University of Warsaw, Poland⋆⋆
{cygan@,malcin@,mp248287@students.}mimuw.edu.pl
[2] Institut für Informatik, Humboldt-Universität zu Berlin, Germany
dmarx@cs.bme.hu
[3] Department of Computer Science and Information Theory,
Budapest University of Technology and Economics, Hungary
ildi@cs.bme.hu

Abstract. We study a family of problems where the goal is to make a graph Eulerian by a minimum number of deletions. We completely classify the parameterized complexity of various versions: undirected or directed graphs, vertex or edge deletions, with or without the requirement of connectivity, etc. Of particular interest is a randomized FPT algorithm for making an undirected graph Eulerian by deleting the minimum number of edges.

1 Introduction

An undirected graph is Eulerian if it is connected and every vertex has even degree; a directed graph is Eulerian if it is strongly connected and every vertex is balanced (i.e., the indegree equals the outdegree). The class of Eulerian graphs is a well-studied and classical notion in the graph theory. We investigate several algorithmic problems related to the question of how to make a graph Eulerian. We focus on deletion problems, where either vertices or edges can be deleted from the input graph to make it Eulerian, using as few deletions as possible. What makes these problems interesting is the interplay of two different type of constraints: each vertex locally prescribes the constraint that it has to be even/balanced, while retaining connectivity is a global requirement. For comparison, we also investigate the variant of the problem where we have only the local constraints (i.e., the task is to delete the minimum number of edges or nodes to make every vertex even/balanced). As many of the studied problems turn out to be NP-hard, we apply the framework of parameterized complexity to get a more detailed insight.

The investigation of these problems was initiated by Cai and Yang [9] who presented parameterized results for some cases. We complement their work by answering here several open questions raised in [9]. Another motivation for our work comes from an

⋆ Supported by the Hungarian National Research Fund (grant OTKA 67651), and by the European Union and the European Social Fund (grant TÁMOP 4.2.1./B-09/1/KMR-2010-0003).

⋆⋆ Authors from the University of Warsaw are partially supported by the Polish Ministry of Science grant N206 567140.

observation of Cechlárová and Schlotter [10]: computing the deficiency for a certain type of housing market is equivalent to finding the minimum number of arcs whose deletion makes every strongly connected component of the graph balanced. While we are not able to determine the parameterized complexity of this problem, our results shed light on the complexity of several related problems.

Related work. Subgraph problems have been widely studied in the literature. To name a few examples, Lewis and Yannakakis [21] investigated the complexity of the vertex-deletion problem for hereditary properties, Alon et al. [2] examined edge-deletion problems for monotone properties, while Natanzon et al. [28] and Burzyn et al. [6] studied the classical complexity of edge modification problems for various graph classes.

Subgraph problems have also been looked at from the parameterized perspective. The most extensively studied variants are the vertex-deletion problems for hereditary properties: the results by Cai [8], and Khot and Raman [18], yield a complete characterization of the fixed-parameter tractable cases. Apart from hereditary properties, FPT algorithms are known for vertex-deletion problems where the task is to obtain a regular graph [26], a chordal graph [23], a grid [12], etc. Parameterized hardness results have been obtained in numerous cases as well [22, 24]. Recently, researchers focused on the issue of kernelization, yielding both positive [4, 17, 29] and negative results [20].

There is much less known about directed graphs. Raman and Sikdar [32] investigated the parameterized complexity of hereditary vertex-deletion problems in digraphs, while Raman and Saurabh [31] examined feedback set problems in tournaments. The FPT algorithm by Chen et al. for finding a feedback vertex set in a directed graph [11] resolved a long-standing open question.

Work related to the class of Eulerian graphs mainly concentrated on the extension problem, where the task is to add a minimum number of edges or arcs in order to make the given graph Eulerian. FPT algorithms were given for various settings by Dorn et al. [13] and by Sorge [33]. Eulerian deletion problems were studied by Cai and Yang [9].

Our contribution. To settle the classical complexity of the examined problems, first we observe (Thms. 1 and 2) that classical results imply polynomial-time algorithms for the edge-deletion problems where the task is to make the graph even/balanced: in the undirected case, this is essentially a T-join problem, while the directed case can be reduced to a flow problem. These observations answer a question raised by Cai and Yang [9], who observed that the analogous vertex-deletion problems are NP-hard. Moreover, the aforementioned algorithms are used as subroutines in our FPT results.

By contrast to the polynomial time algorithms, we show that the seemingly similar edge- (or arc-) deletion problems where we aim for an Eulerian graph are NP-hard, even in the extremely restricted case when the input is a cubic planar graph and the number of deletions can be arbitrary (Theorem 3). We investigate both the undirected and the directed cases of Eulerian edge-deletion problem thoroughly from the parameterized point of view: we present a fixed-parameter tractable algorithm for both cases where the parameter is the number of deletions allowed (Theorem 8), and prove that these problems do not admit a polynomial-size kernel unless NP $\subseteq$ coNP/poly (Theorem 9), which is known to imply a collapse of the polynomial hierarchy to its third level [34, 7]. The FPT results use a novel argument that might be of independent interest. Intuitively, we need to find a solution S to a T-join problem and a witness (disjoint from

	Undirected even	Undirected Eulerian	Directed balanced	Directed Eulerian
Vertex deletion:	W[1]-hard [9]	W[1]-hard [9]	W[1]-hard Thm. 15	W[1]-hard Thm. 15
Edge deletion:	P Thm. 1	FPT, no poly kernel Thms. 3, 8, 9	P Thm. 2	FPT, no poly kernel Thms. 3, 8, 9

Table 1. Summary of the main results. Parameterized results only appear when the corresponding problem is NP-hard; the parameter considered is the number of deletions allowed.

S) certifying that the graph remains connected after the removal of S. Using a random colouring, we partition the edges into two types: each edge can contribute either to the solution or to the witness of the solution. This partition ensures that the solution and the witness are disjoint. While the use of random colourings is a standard technique for finding a solution consisting of disjoint objects [3], we use this technique to separate the solution from its proof of feasibility.

The undirected vertex-deletion problems, where the task is to obtain an Eulerian or an even graph, were already handled by Cai and Yang [9] who proved their W[1]-hardness. We complemented these results by showing W[1]-hardness for the directed cases as well in Theorem 15. Additionally, we also focus on a slight modification of the node-deletion problems where certain forbidden vertices are not allowed to be deleted. Theorem 16 shows that each of the four node-deletion problems remains W[1]-hard, even if we are only allowed to delete vertices of degree at most 4. This contrasts the easy FPT algorithm applicable if the parameter is not only the number of deletions but also the maximum degree of the graph (Theorem 19).

Table 1 shows a summary of our main results.

Organization of the paper. Section 2 describes our notation, and provides basic concepts of parameterized complexity. Section 3 discusses polynomial-time solvable edge-deletion problems. We deal with the NP-hard Eulerian edge-deletion problems in Section 4, first covering the issue of NP-completeness, and then fixed-parameter tractability and kernelization in Sections 4.1 and 4.2. Node-deletion problems are discussed in Section 5. We summarize our results and draw conclusions in Section 6.

2 Notation and preliminaries

Given a graph G, let $V(G)$ denote its vertex set and $E(G)$ denote its edge set (or, in the directed case, its arc set). The *degree* of a vertex v in an undirected graph G is denoted by $d_G(v)$; we say that v is *even*, if $d_G(v)$ is even. For a vertex v in a directed graph G, we denote by $d_G^{in}(v)$ and $d_G^{out}(v)$ its indegree and its outdegree, respectively. We say that v is *balanced*, if $d_G^{in}(v) = d_G^{out}(v)$. We define the *degree* of v in G (where G is directed), as $d_G(v) = d_G^{in}(v) + d_G^{out}(v)$. If G is clear from the context, we might omit the subscript. A directed graph is *weakly connected* if the underlying undirected graph is connected. An *even (balanced) graph* is an undirected (directed) graph where each vertex is even (balanced). An undirected Eulerian graph is a connected even graph, and a directed Eulerian graph is a strongly connected balanced graph.[4]

Given a path P in a (directed or undirected) graph, the *internal vertices* of P are the vertices lying on P except for the two end-vertices. If $d_G(v) = 2$ holds (meaning $d_G^{in}(v) = d_G^{out}(v) = 1$ in the directed case) for each internal vertex v of P, then we say that the path P is an *unattached path.* In a directed graph, a *pair of twin arcs* is two arcs (a, b) and (b, a).

Given a set X of vertices, edges, or arcs in a graph G, let $G \setminus X$ denote the graph obtained by deleting X from G. When X has only one element x, we might also write $G \setminus x$ instead of $G \setminus \{x\}$.

Parameterized complexity. In the parameterized complexity setting, an instance comes with an integer parameter k — formally, a parameterized problem Q is a subset of $\Sigma^* \times \mathbb{N}$ for some finite alphabet Σ. We say that the problem is *fixed parameter tractable* (*FPT*) if there exists an algorithm solving any instance (x, k) in time $f(k)\mathrm{poly}(|x|)$ for some (usually exponential) computable function f. It is known that a problem is FPT iff it is kernelizable: a kernelization algorithm for a problem Q takes an instance (x, k) and in time polynomial in $|x| + k$ produces an equivalent instance (x', k') (i.e., $(x, k) \in Q$ iff $(x', k') \in Q$) such that $|x'| + k' \leq g(k)$ for some computable function g. The function g is the *size of the kernel*, and if it is polynomial, we say that Q admits a polynomial kernel.

3 Polynomial-time solvable cases

First, we give a simple polynomial time algorithm for the following problem:

UNDIRECTED EVEN EDGE DELETION **Parameter:** k
Input: An undirected graph G and an integer k.
Question: Does there exist a set S of at most k edges in G such that $G \setminus S$ is even?

It turns out that this problem is strongly connected to the concept of a T*-join*. If we define T to be the set of vertices having odd degree, then UNDIRECTED EVEN EDGE DELETION is equivalent with the following classical problem of finding a T-join of minimum size:

MINIMUM T-JOIN
Input: A graph $G = (V, E)$ and a set $T \subseteq V$ of even size.
Task: Find a minimum T-join, i.e., a set $S \subseteq E$ of minimum size such that T is exactly the set of vertices of odd degree in the graph $H = (V, S)$.

Since MINIMUM T-JOIN can be solved in cubic time by the algorithm of Edmonds and Johnson [14], we obtain the following consequence:

Theorem 1. UNDIRECTED EVEN EDGE DELETION *can be solved in $O(n^3)$ time for an n-vertex graph.*

[4] Strictly speaking, the usual definition of being Eulerian requires only that the graph is connected after removing the isolated vertices. However, we feel that requiring connectivity instead leads to more natural and fundamental problems.

Now we turn our attention to the directed version of the problem:

DIRECTED BALANCED EDGE DELETION **Parameter:** k
Input: A directed graph G and an integer k.
Question: Does there exist a set S of at most k arcs in G such that $G \setminus S$ is balanced?

This problem can be formulated as a minimum cost flow problem with unit costs as follows. We create a digraph G' by taking G and adding two vertices s,t (source and sink). Each edge of $E(G)$ has unit capacity and unit cost. For each vertex $v \in V(G)$ such that $d^{in}(v) < d^{out}(v)$ we add to G' an arc (s, v) of capacity $d^{out}(v) - d^{in}(v)$ and cost zero. Similarly, for each vertex $v \in V(G)$ such that $d^{in}(v) > d^{out}(v)$ we add to G' an arc (v, t) of capacity $d^{in}(v) - d^{out}(v)$ and cost zero. Let f^* denote the total capacity of the added arcs (s, v). In a solvable instance we know that $f^* \leq k$.

It is straightforward to see that a flow of size f^* and cost k corresponds to a set S of k arcs for which $G \setminus S$ balanced, and vice versa. Thus, in order to find a solution of minimum size it suffices to find a minimum cost flow of size f^*. As $f^* \leq k$ and each arc has unit cost, this can be done in $O(nm \log n \log \log k)$ time [1], where $n = |V(G)|$ and $m = |E(G)|$. Note that the above argument also handles an annotated case, where we require that $S \subseteq E_a$ for a set $E_a \subseteq E$ given in the input, as we can put zero capacities on $E \setminus E_a$. This yields the following:

Theorem 2. DIRECTED BALANCED EDGE DELETION *can be solved in* $O(nm \log n \log \log k)$ *time for an input graph with* n *vertices and* m *edges, even in an annotated case where some edges are forbidden to delete.*

4 Eulerian edge-deletion problems

In this section we examine the following problems:

UNDIRECTED EULERIAN EDGE DELETION **Parameter:** k
Input: A connected undirected graph G and an integer k.
Question: Does there exist a set S of at most k edges of G such that $G \setminus S$ is Eulerian, i.e., even and connected?

DIRECTED EULERIAN EDGE DELETION **Parameter:** k
Input: A strongly connected directed graph G and an integer k.
Question: Does there exist a set S of at most k arcs of G such that $G \setminus S$ is Eulerian, i.e., balanced and strongly connected?

The undirected problem can be easily seen to be NP-hard by observing that a cubic graph contains a Hamiltonian cycle if and only if it can be made Eulerian by edge deletions. Indeed, if deleting a set of edges from a cubic graph G results in an Eulerian graph G', then each vertex in G' must have degree 2, so G' must be a Hamiltonian cycle of G. Since the HAMILTONIAN CYCLE problem restricted to cubic planar graphs is NP-hard [16] the result follows. The directed version can be treated in a similar way using NP-hardness from [30].

Theorem 3. *The* UNDIRECTED *and* DIRECTED EULERIAN EDGE DELETION *problems are NP-hard, even when restricted to inputs* (G, k) *where* G *is a planar (directed) graph with maximum degree at most 3, and* $k = |E(G)|$.

In Section 4.1, we show that both versions of the problem are FPT and can be solved in time $2^{O(k \log k)} n^{O(1)}$. The algorithm is based on a novel randomized selection argument. In Section 4.2, we sharpen Theorem 3 by showing that the problems do not admit a polynomial kernel. In some sense, the nonexistence of polynomial kernels suggests that randomized selection or a similar technique is inherently required for the problems, as they cannot be solved by simple reduction rules.

4.1 FPT algorithms

We have seen in Section 3 that removing edges to make all the vertices even can be expressed as a T-join problem, where T is the set of odd vertices. Thus UNDIRECTED EULERIAN EDGE DELETION requires us to find a T-join S such that $G \setminus S$ is connected. Observe that if G is connected, and $G \setminus S$ has a connected subgraph W containing the endpoints of every edge in S, then $G \setminus S$ is connected as well. We will call such a subgraph W a *witness* of S. Therefore, the right way to look at the problem is that we need to find a pair (S, W), where is S is a T-join and W is the witness of S. It is clear that the problem has a solution if and only if such a pair exists.

Our approach for finding a pair (S, W) is the following. We randomly colour the edges of the graph red and blue, and try to find a pair (S, W) where S uses only red edges and the subgraph W uses only blue edges. We would like to ensure that if a suitable pair (S, W) exists, then it is correctly coloured red and blue with probability at least $2^{-O(k \log k)}$. However, in general the size of W can be very large (unbounded in k, see Appendix A for an example) and therefore the probability of a correct colouring can be very small. We get around this problem by observing that edges "far" from T can be always coloured blue, and there is a witness W that uses only a bounded number of edges "close" to T. Formally, we say that an edge e is *close* if at least one endpoint of e is at distance at most k from T; otherwise, e is *far*. The following two lemmas contain the crucial combinatorial ideas of the algorithm:

Lemma 4. *If* S *is an optimum solution of size at most* k*, then each edge of* S *is close.*

Proof. As removing a cycle from S would still yield a solution, $H = (V, S)$ has to be a forest for an optimum solution S. Each connected component of H that is not an isolated vertex contains a vertex from T, as each tree contains vertices of odd degree (for example, leaves). Since $|S| \leq k$, each vertex in such a connected component is at distance at most k from T, and thus each edge in S is close. □

Lemma 5. *If* S *is an optimum solution of size at most* k*, then* S *has a witness* W *having at most* $(2k - 1)(2k + 2)$ *close edges.*

Proof. Let X be the set of endpoints of the edges in S. Note that $T \subseteq X$ and $|X| \leq 2|S| \leq 2k$. Let i be the smallest integer such that $G \setminus S$ has a subgraph W containing X, having exactly i connected components and at most $(|X| - i)(2k + 2)$ close edges (such i and W always exist as for $i = |X|$ we can take $W = (X, \emptyset)$). If $i = 1$, then we are done. Otherwise, we can assume that each component of W contains a vertex of X; let P be a shortest path in $G \setminus S$ that connects two different components of W. Denote these components K_1 and K_2.

We claim that only the first $k+1$ and the last $k+1$ edges of P may be close. If this is true, then adding P to W decreases the number of components and increases the number of close edges by at most $2k+2$, contradicting the minimality of i.

Suppose that an edge e is close, but it is not among the first or last $k+1$ edges, i.e., both of its endpoints are at distance greater than k from both K_1 and K_2 on P. As e is close, it has an endpoint v such that there is a path P' of length at most k connecting v and T. As $T \subseteq X$, the path P' connects v to a component K' of W. Assuming without loss of generality that $K' \neq K_1$, the concatenation of P' and the subpath of P from K_1 to v is a walk P'' connecting two different components of W. As the distance of v from K_2 on P is more than k, the walk P'' is shorter than P, contradicting the minimality of P. □

Now, we are ready to state our algorithm, working as follows:

1. Determine which edges are close and which are far.
2. Make each close edge independently with probability $1/k^2$ red; every edge that is not red becomes blue.
3. If there is more than one connected component of the blue edges containing a vertex from T, return NO; otherwise let K_B be this unique component.
4. Solve MINIMUM T-JOIN instance (G_R, T), where G_R is the graph induced by the red edges with both endpoints in K_B. If the solution is of size at most k, return it, otherwise return NO.

Lemma 6. *If the algorithm returns a solution S, then S is a proper solution to* UNDIRECTED EULERIAN EDGE DELETION.

Proof. By the definition of MINIMUM T-JOIN, $G \setminus S$ is even. The component K_B of blue edges ensures that the endpoints of S are in the same component of $G \setminus S$, i.e., $G \setminus S$ is connected. □

Lemma 7. *If the* UNDIRECTED EULERIAN EDGE DELETION *instance (G, k) was a YES-instance, the algorithm returns a solution with probability at least* $1/2^{O(k \log k)}$.

Proof. Let S be an optimum solution to (G, k), and let W be a witness having at most $(2k-1)(2k+2)$ close edges, guaranteed by Lemma 5. In the algorithm:

1. With probability at least $(1/k^2)^k = 1/2^{2k \log k}$ each edge of S becomes red.
2. With probability at least $(1 - 1/k^2)^{(2k-1)(2k+2)} = \Omega(1)$ each close edge of W becomes blue (and hence every edge of W is blue).

The above events are independent, since S and W do not share edges. Furthermore, if both events happen, then W will connect all the endpoints of the edges from S. Therefore, all of these endpoints will be contained in one connected component K_B of the graph induced by blue edges, which in particular connects all the vertices from T. Thus, with probability $1/2^{O(k \log k)}$, every edge of S appears in G_R in the last step of the algorithm and the MINIMUM T-JOIN instance has a solution of size at most k. □

Theorem 8. *Both the* UNDIRECTED *and* DIRECTED EULERIAN EDGE DELETION *problems are fixed-parameter tractable with parameter k.*

Proof. By Lemmas 6 and 7, the presented algorithm for UNDIRECTED EULERIAN EDGE DELETION finds a solution with probability $1/2^{O(k \log k)}$, and never produces a wrong output, that is removal of the returned set of edges always makes the graph Eulerian. Since the algorithm runs in $O(n^3)$ time for an n-vertex graph, we immediately obtain a randomized FPT Monte-Carlo algorithm, running in $2^{O(k \log k)} n^3$ time.

We can derandomize the above algorithm using the standard technique of splitters, which is described in Appendix B.

Regarding DIRECTED EULERIAN EDGE DELETION, we can use a slightly modified version of our randomized algorithm (which then can be derandomized). After defining the set T of terminals to contain the unbalanced vertices, we forget about the orientation of the arcs, and perform Steps $1-3$ of the algorithm. We adjust Step 4 by solving an annotated DIRECTED BALANCED EDGE DELETION instance (G, k) where only red arcs can be deleted. Observe that this algorithm in fact looks for a set of edges S of size at most k such that $G \setminus S$ is balanced and weakly connected. However, every graph that is weakly connected and balanced is Eulerian, thus the algorithm returns the solution to DIRECTED BALANCED EDGE DELETION with high probability, if one exists. $\square$

4.2 Non-existence of a polynomial kernel for UNDIRECTED and DIRECTED EULERIAN EDGE DELETION

The aim of this subsection is to prove the following theorem.

Theorem 9. *If NP $\not\subseteq$ coNP/poly, then there is no polynomial kernel for the* UNDIRECTED *and* DIRECTED EULERIAN EDGE DELETION *problems with parameter k, even if the input graph has maximum degree at most 4.*

We use the cross-composition technique introduced by Bodlaender et al. [5]. Let us recall the crucial definitions.

Definition 10 (Polynomial equivalence relation [5]). *An equivalence relation $\mathcal{R}$ on Σ^* is called a* polynomial equivalence relation *if (1) there is an algorithm that given two strings $x, y \in \Sigma^*$ decides whether $\mathcal{R}(x, y)$ in $(|x| + |y|)^{O(1)}$ time; (2) for any finite set $S \subseteq \Sigma^*$ the equivalence relation $\mathcal{R}$ partitions the elements of S into at most $(\max_{x \in S} |x|)^{O(1)}$ classes.*

Definition 11 (Cross-composition [5]). *Let $L \subseteq \Sigma^*$ and let $Q \subseteq \Sigma^* \times \mathbb{N}$ be a parameterized problem. We say that L* cross-composes *into Q if there is a polynomial equivalence relation $\mathcal{R}$ and an algorithm which, given t strings $x_1, x_2, \ldots x_t$ belonging to the same equivalence class of $\mathcal{R}$, computes an instance $(x^*, k^*) \in \Sigma^* \times \mathbb{N}$ in time polynomial in $\sum_{i=1}^{t} |x_i|$ such that (1) $(x^*, k^*) \in Q$ iff $x_i \in L$ for some $1 \leq i \leq t$; (2) k^* is bounded polynomially in $\max_{i=1}^{t} |x_i| + \log t$.*

Theorem 12 ([5], Theorem 9). *If $L \subseteq \Sigma^*$ is NP-hard under Karp reductions and L cross-composes into the parameterized problem Q that has a polynomial kernel, then NP $\subseteq$ coNP/poly.*

We apply Theorem 12 on the following language L:

Undirected or Directed $s - t$ Path with Forbidden Pairs of Edges
Input: An undirected or directed graph $G = (V, E)$, two vertices $s, t \in V$, and a set $\mathcal{C} \subseteq E \times E$ called *the constraints*.
Task: Does there exist an $s - t$ path $\mathcal{P}$ in G such that from each constraint $(e_1, e_2) \in \mathcal{C}$ at least one edge (arc) does not lie on $\mathcal{P}$?

The undirected version of this problem with forbidden pairs of vertices was proven to be NP-hard by Kolman and Pangrác [19] and their proof can be easily modified to handle our case as well. For sake of completeness, we provide a proof of the lemma below in Appendix B.

Lemma 13. Undirected *and* Directed $s - t$ Path with Forbidden Pairs of Edges *are NP-hard under Karp reductions, even in the case where each vertex has maximum degree three, s and t have degree one, and, in the directed case, each vertex has maximum in- and outdegree two.*

To finish the proof of Theorem 9 we need to show a cross-composition algorithm. This is done in the following lemma.

Lemma 14. Undirected (Directed) $s - t$ Path with Forbidden Pairs of Edges *cross-composes to* Undirected (Directed) Eulerian Edge Deletion. *If the input instances have degrees bounded as in Lemma 13 then the output instance can be made to have maximum degree* 4.

Proof. For the equivalence relation $\mathcal{R}$ we take an almost trivial relation that sorts all malformed instances into one equivalence class and all well-formed into another one. If we are given malformed instances, we simply output a trivial NO-instance. Thus in the rest of the proof we assume we are given a sequence $(G_i, s_i, t_i, \mathcal{C}_i)_{i=1}^t$ of Undirected or Directed $s - t$ Path with Forbidden Pairs of Edges instances.

We now construct an Undirected or Directed Eulerian Edge Deletion instance (G, k). We start by obtaining a graph G_i' for each $1 \leq i \leq t$ as follows. First we subdivide each edge $e \in E(G_i)$ with new vertices x_e^C, one for each constraint $C \in \mathcal{C}_i$ that contains e. Then for each constraint $C = (e_1, e_2) \in \mathcal{C}_i$ we introduce vertices z_1^C and z_2^C and create a (directed) cycle $x_{e_1}^C, z_1^C, x_{e_2}^C, z_2^C$. By $V(G_i)$ we denote the subset of $V(G_i')$ containing vertices different than x_e^C and z_α^C. To construct the graph G, we first take the union of all graphs G_i' and identify all vertices s_i into one vertex s^* and all vertices t_i into one vertex t^*. Let $V^0 = \{s^*, t^*\} \cup \bigcup_{i=1}^t V(G_i) \setminus \{s_i, t_i\}$. Second, we introduce a new vertex r and connect it to the rest of the graph as follows. In the undirected case for each $v \in V^0 \setminus \{s^*, t^*\}$ we connect r and v with one or two unattached paths of length 2, so that in G the vertex v is even. In the directed case, we connect r and v with some positive number of unattached directed paths of length 2, so that in G the vertex v is balanced. We do almost the same construction to connect s^* and t^* to r, but we ensure that the degrees of s^* and t^* are odd (in the undirected case) or that $d_G^{in}(s^*) + 1 = d_G^{out}(s^*)$ and $d_G^{in}(t^*) = d_G^{out}(t^*) + 1$ (in the directed case). Note that r is even (balanced). Finally, we set $k = \max_{i=1}^t |V(G_i')| - 1 = O(\max_{i=1}^t |V(G_i)| + |\mathcal{C}_i|)$.

It is clear that the above construction can be done in polynomial time and that the parameter k is bounded polynomially in the maximum size of the input instances. We verify its correctness and explain the degree reduction in Appendix B. □

5 Node-deletion problems

We first consider the following two node-deletion problems:

DIRECTED BALANCED (or EULERIAN) NODE DELETION **Parameter:** k
Input: A directed graph G and an integer k
Question: Does there exist a set of at most k vertices $S \subseteq V(G)$ such that $G \setminus S$ is balanced (or Eulerian)?

The undirected versions of these problems, namely UNDIRECTED EVEN and UNDIRECTED EULERIAN NODE DELETION, are defined analogously. While these undirected variants were already shown to be W[1]-hard with parameter k by Cai and Yang [9], the complexity of the directed versions has not been studied yet. The following theorem shows that they are intractable as well (see Appendix C).

Theorem 15. DIRECTED BALANCED NODE DELETION *and* DIRECTED EULERIAN NODE DELETION *are NP-hard and W[1]-hard with parameter k.*

As Table 1 shows, the vertex-deletion variant is W[1]-hard in all four cases, while the edge-deletion version is FPT or even polynomial-time solvable. What makes the vertex-deletion versions harder? One obvious difference is that in the edge-deletion problem the answer is trivially no if there are more than $2k$ odd/unbalanced vertices, but the vertex-deletion versions can have a solution even if the number of such nodes is unbounded. This suggests that the higher complexity comes from the ability of affecting the degree of many vertices by a single vertex deletion. Indeed, if every vertex has degree bounded by Δ, then we can solve all of the above defined node-deletion problems in $O((\Delta + 1)^k(|V(G)| + |E(G)|))$ time by a simple branching algorithm which is described in Appendix C. However, this interpretation is not fully correct: as we shall show, the vertex-deletion problems are hard even if we are allowed to delete only vertices of constant degree.

To this end, we define the following variation of the four different node-deletion problems, where α can be UNDIRECTED EVEN, UNDIRECTED EULERIAN, DIRECTED BALANCED, or DIRECTED EULERIAN:

α NODE DELETION WITH FORBIDDEN NODES **Parameter:** k
Input: A graph G, a set $F \subseteq V(G)$ of *forbidden nodes*, and an integer k.
Question: Does there exist a solution $S \subseteq V(G)$ for (G, k) with respect to the corresponding α NODE DELETION problem such that $S \cap F = \emptyset$ and $|S| \leq k$?

In other words, we require the solution to be disjoint from a set of *forbidden vertices*. A vertex is *allowed*, if it is not forbidden. For each of the four node-deletion problems, the above variant is at least as hard as the original problem, and in fact has the same complexity: this variant can easily be reduced to the original version, by attaching long unattached cycles to every forbidden vertex. Furthermore, we show that allowing only the deletion of bounded-degree vertices does not make the problem easier:

Theorem 16. *Each of the problems* α NODE DELETION WITH FORBIDDEN NODES *where* α *is* UNDIRECTED EVEN, UNDIRECTED EULERIAN, DIRECTED BALANCED, *or* DIRECTED EULERIAN *remains W[1]-hard with parameter* k*, even if each allowed vertex has degree at most 4.*

6 Conclusion

We completed the analysis of the complexity of making a graph Eulerian via edge or vertex deletions. There are two open problems that we would like to emphasise here.

First, do there exist FPT algorithms for the edge-deletions problems running in time $c^k n^{O(1)}$? It seems hard to obtain such algorithms using our techniques, mainly due to the fact that the witness subgraph W may contain $\Omega(k^2)$ close edges (see Appendix A).

Second, Cechlárová and Schlotter in [10] asked for the parameterized complexity of a related problem, where the task is to delete at most k arcs from a directed graph to obtain a graph where each strongly connected component is Eulerian. This problem seems to be significantly different than the problems considered in this paper, as for example it includes DIRECTED FEEDBACK VERTEX SET [10], and, to the best of our knowledge, the question of its parameterized complexity still remains open.

References

1. Ahuja, R.K., Goldberg, A.V., Orlin, J.B., Tarjan, R.E.: Finding minimum-cost flows by double scaling. Math. Program. 53, 243–266 (1992)
2. Alon, N., Shapira, A., Sudakov, B.: Additive approximation for edge-deletion problems. In: FOCS. pp. 419–428 (2005)
3. Alon, N., Yuster, R., Zwick, U.: Color-coding. J. ACM 42(4), 844–856 (1995)
4. van Bevern, R., Moser, H., Niedermeier, R.: Approximation and tidying—a problem kernel for s-plex cluster vertex deletion. Algorithmica Published online in February 2011
5. Bodlaender, H.L., Jansen, B.M.P., Kratsch, S.: Cross-composition: A new technique for kernelization lower bounds. CoRR abs/1011.4224 (2010)
6. Burzyn, P., Bonomo, F., Durán, G.: NP-completeness results for edge modification problems. Discrete Appl. Math. 154, 1824–1844 (2006)
7. Cai, J., Chakaravarthy, V.T., Hemaspaandra, L.A., Ogihara, M.: Competing provers yield improved Karp-Lipton collapse results. Inf. Comput. 198(1), 1–23 (2005)
8. Cai, L.: Fixed-parameter tractability of graph modification problems for hereditary properties. Inf. Process. Lett. 58(4), 171–176 (1996)
9. Cai, L., Yang, B.: Parameterized complexity of even/odd subgraph problems. In: CIAC. pp. 85–96 (2010)
10. Cechlárová, K., Schlotter, I.: Computing the deficiency of housing markets with duplicate houses. In: IPEC. pp. 72–83 (2010)
11. Chen, J., Liu, Y., Lu, S., O'Sullivan, B., Razgon, I.: A fixed-parameter algorithm for the directed feedback vertex set problem. J. ACM 55(5), 1–19 (2008)
12. Díaz, J., Thilikos, D.M.: Fast FPT-algorithms for cleaning grids. In: STACS. pp. 361–371 (2006)

13. Dorn, F., Moser, H., Niedermeier, R., Weller, M.: Efficient algorithms for Eulerian extension. In: WG. pp. 100–111 (2010)
14. Edmonds, J., Johnson, E.: Matching, Euler tours and the Chinese postman problem. Math. Program. 5, 88–124 (1973)
15. Fellows, M.R., Hermelin, D., Rosamond, F.A., Vialette, S.: On the parameterized complexity of multiple-interval graph problems. Theor. Comput. Sci. 410, 53–61 (2009)
16. Garey, M.R., Johnson, D.S., Tarjan, R.E.: The planar Hamiltonian circuit problem is NP-complete. SIAM J. on Computing 5, 704–714 (1976)
17. Guo, J.: Problem kernels for NP-complete edge deletion problems: Split and related graphs. In: ISAAC. pp. 915–926 (2007)
18. Khot, S., Raman, V.: Parameterized complexity of finding subgraphs with hereditary properties. Theor. Comput. Sci. 289, 997–1008 (2002)
19. Kolman, P., Pangrác, O.: On the complexity of paths avoiding forbidden pairs. Discrete Applied Mathematics 157(13), 2871–2876 (2009)
20. Kratsch, S., Wahlström, M.: Two edge modification problems without polynomial kernels. In: IWPEC. pp. 264–275 (2009)
21. Lewis, J.M., Yannakakis, M.: The node-deletion problem for hereditary properties is NP-complete. J. Comput. Syst. Sci. 20(2), 219–230 (1980)
22. Lokshtanov, D.: Wheel-free deletion is W[2]-hard. In: IWPEC. pp. 141–147 (2008)
23. Marx, D.: Chordal deletion is fixed-parameter tractable. Algorithmica 57(4), 747–768 (2010)
24. Mathieson, L., Szeider, S.: The parameterized complexity of regular subgraph problems and generalizations. In: CATS. pp. 79–86 (2008)
25. Moser, H., Raman, V., Sikdar, S.: The parameterized complexity of the unique coverage problem. In: ISAAC. pp. 621–631 (2007)
26. Moser, H., Thilikos, D.M.: Parameterized complexity of finding regular induced subgraphs. Journal of Discrete Algorithms 7, 181–190 (2009)
27. Naor, M., Schulman, L.J., Srinivasan, A.: Splitters and near-optimal derandomization. In: FOCS. pp. 182–191 (1995)
28. Natanzon, A., Shamir, R., Sharan, R.: Complexity classification of some edge modification problems. Discrete Appl. Math. 113, 109–128 (2001)
29. Philip, G., Raman, V., Villanger, Y.: A quartic kernel for pathwidth-one vertex deletion. In: WG. pp. 196–207 (2010)
30. Plesník, J.: The NP-completeness of the Hamiltonian cycle problem in planar digraphs with degree bound two. Inf. Process. Lett. 8(4), 199–201 (1979)
31. Raman, V., Saurabh, S.: Parameterized algorithms for feedback set problems and their duals in tournaments. Theor. Comput. Sci. 351, 446–458 (2006)
32. Raman, V., Sikdar, S.: Parameterized complexity of the induced subgraph problem in directed graphs. Inf. Process. Lett. 104, 79–85 (2007)
33. Sorge, M.: On making directed graphs Eulerian. CoRR abs/1101.4283 (2011)
34. Yap, C.K.: Some consequences of non-uniform conditions on uniform classes. Theor. Comput. Sci. 26, 287–300 (1983)

A An example of a large witness set *W*

In this section we give a simple example that the witness graph W, considered in the FPT algorithms in Section 4.1, may be arbitrarily large and may contain $\Omega(k^2)$ close edges. We construct a graph G as follows. First take a cycle of length $2kM$ for some $M > 2k$ and let $v_0, v_1, \ldots, v_{2k-1}$ be a sequence of evenly distributed vertices on this cycle, i.e., $v_i = w_{iM}$, where $w_0, w_1, \ldots, w_{2kM-1}$ are vertices of the cycle, lying in this order. Moreover, for each $0 \leq i < k$ we connect the vertices v_{2i} and v_{2i+1}. Note that $S = \{v_{2i}v_{2i+1} : 0 \leq i < k\}$ is the only feasible solution of size k to the UNDIRECTED EULERIAN EDGE DELETION problem in the graph G, but any witness W of S needs to contain a path of length $(2k-1)M$. Moreover, such a path contains roughly $2k(2k-1) = \Omega(k^2)$ close edges.

Note that in the above construction it is not crucial to start from a long cycle, as any Eulerian graph of large diameter would suffice. In such a graph we simply take the vertices $\{v_i : 0 \leq i < 2k\}$ to be any set of vertices that are pairwise distant.

B Omitted proofs on edge-deletion problems (Section 4)

Proof (of Theorem 8). We can derandomize the described algorithm using the standard technique of splitters. A (m, r, r^2)-*splitter* is a family of functions from $\{1, 2, \ldots, m\}$ to $\{1, 2, \ldots, r^2\}$, such that for any subset $X \subseteq \{1, 2, \ldots, m\}$ of size r, one of the functions in the family is injective on X. Naor et al [27] gave an explicit construction of an (m, r, r^2)-splitter of size $O(r^6 \log r \log m)$.

In Step 3 of the algorithm we want to separate the solution S (of size at most k) from the set of close edges of the witness W (of size at most $\ell = (2k-1)(2k+2)$). Let m be the cardinality of the set of close edges in the graph, we may identify $\{1, 2, \ldots, m\}$ with this set. Instead of the random colouring process, we can try every function f in a $(m, k+\ell, (k+\ell)^2)$-splitter and every set $F \subseteq \{1, 2, \ldots, (k+\ell)^2\}$ of size k. For a particular choice of f and F, we colour red those close edges e for which $f(e) \in F$. By the definition of the splitter, if there exists a solution S with a witness W, there will be a function f that is injective on the set of close edges of $S \cup W$ and a subset F such that $f(e) \in F$ if $e \in S$ and $f(e) \notin F$ if e is a close edge in W. Note that the size of $(m, k+\ell, (k+\ell)^2)$-splitter is bounded polynomially in the input size, whereas there are $2^{O(k \log k)}$ choices for the set F. □

Proof (of Lemma 13). We first provide a Karp reduction from the CLIQUE problem to our problems without the degree condition. Let (H, k) be a CLIQUE instance. We construct an UNDIRECTED or DIRECTED $s-t$ PATH WITH FORBIDDEN PAIRS OF EDGES instance $(G, s, t, \mathcal{C})$ as follows. To construct the graph G, we start by adding vertices s, t and p_i for $0 \leq i \leq k$ and edges sp_0 and $p_k t$ (arcs (s, p_0) and (p_k, t)). Then for each $1 \leq i \leq k$ and $v \in V(H)$ we introduce a vertex x_i^v and edges $p_{i-1}x_i^v$ and $x_i^v p_i$ (arcs (p_{i-1}, x_i^v) and (x_i^v, p_i)). Finally, for each $1 \leq i, j \leq k$ and $u, v \in V(H)$ such that $uv \notin E(H)$ (possibly $u = v$), we introduce constraints $(x_i^u p_i, x_j^v p_j)$ and $(x_j^u p_j, x_i^v p_i)$.

Let us now verify the correctness of the above reduction. If $\{v_1, v_2, \ldots, v_k\} \subseteq V(H)$ is a vertex set of a k-clique in H, then a path consisting of edges (or corresponding arcs) sp_0, $p_k t$ and $p_{i-1}x_i^{v_i}$, $x_i^{v_i} p_i$ for $1 \leq i \leq k$ is a feasible solution to the instance

$(G, s, t, \mathcal{C})$. In the other direction, note that any simple path from s to t visits for each $1 \leq i \leq k$ exactly one vertex from the set $\{x_i^v : v \in V(H)\}$, say $x_i^{v_i}$. We claim that $\{v_1, v_2, \ldots, v_k\}$ induces a k-clique in H. To see this note that the introduced constraints imply that if $i \neq j$ then $v_i \neq v_j$ and $v_i v_j \in E(H)$.

To obtain the degree bounds, note that each vertex $v \in V(G)$ with $d_G(v) \geq 3$ can be replaced with a (directed) cycle of length $d_G(v)$, where each edge (arc) previously incident to v is now connected to a different vertex on the cycle. □

Proof (of Lemma 14). We now verify the correctness of the described construction.

First, let $\mathcal{P}$ be a simple path that is a feasible solution to $(G_j, s_j, t_j, \mathcal{C}_j)$ for some $1 \leq j \leq t$. The path $\mathcal{P}$ naturally defines a simple path $\mathcal{P}'$ in G'_j and in G. We claim that the edge set S of $\mathcal{P}'$ is a feasible solution to the constructed DIRECTED or UNDIRECTED EULERIAN EDGE DELETION instance. As it is contained in G'_j, we have $|S| \leq k$. Since in G the only odd (unbalanced) vertices were s^* and t^*, $G \setminus S$ is even (balanced). We now verify that each vertex $v \in V(G)$ is (weakly) connected to the vertex r in $G \setminus S$. It is clear for each $v \in V^0$, since $E_r \cap S = \emptyset$, where E_r denotes the set of edges in the paths between V^0 and r. For the other vertices, note that if $C = (e_1, e_2) \in \bigcup_{i=1}^{t} \mathcal{C}_i$, then either e_1 or e_2 does not belong to $\mathcal{P}$ (say $e_1 \notin \mathcal{P}$) and the cycle $x_{e_1}^C, z_1^C, x_{e_2}^C, z_2^C$ is connected to r via subdivided edge e_1 and its endpoints. Note that in the directed case it is sufficient to ensure only weak connectivity, as a balanced graph is weakly connected iff it is strongly connected.

In the other direction, let S be a solution to the constructed DIRECTED or UNDIRECTED EULERIAN EDGE DELETION instance. We assume that $|S|$ is minimum possible. It is easy to see that since $G \setminus S$ is even (balanced) and $|S|$ is minimal, S needs to induce a simple path $\mathcal{P}'$ from s^* to t^*. This path cannot contain a neighbour r' of r, since otherwise r' becomes isolated in $G \setminus S$. Thus $\mathcal{P}'$ is contained in graph G'_j for some $1 \leq j \leq t$. Moreover, note that $\mathcal{P}'$ cannot contain any vertex z_α^C, as otherwise z_α^C becomes isolated in $G \setminus S$. Thus $\mathcal{P}'$ naturally defines a path $\mathcal{P}$ in G_j with endpoints s_j and t_j. Observe that if for some $C = (e_1, e_2) \in \mathcal{C}_j$ the path $\mathcal{P}$ contained both e_1 and e_2, then in $G \setminus S$ the cycle $x_{e_1}^C, z_1^C, x_{e_2}^C, z_2^C$ would be unreachable from the rest of the graph G. Thus, $\mathcal{P}$ is a feasible solution to the instance $(G_j, s_j, t_j, \mathcal{C}_j)$.

We now show how to modify the presented construction to obtain an instance with maximum degree 4. First note that if $v \in V(G) \setminus V^0 \setminus \{r\}$ we clearly have $d_G(v) \leq 4$. Moreover, if the degrees in $(G_i, s_i, t_i, \mathcal{C}_i)$ are bounded as in Lemma 13, then for any $v \in V^0 \setminus \{s^*, t^*\}$ the number of unattached paths connecting v and r can be chosen so that v is even (balanced) and we have $d_G(v) \leq 4$ in the undirected case and $d_G^{in}(v), d_G^{out}(v) \leq 2$ in the directed one. Thus, we are left with the vertices s^*, t^* and r.

We first reduce the degree of vertices s^* and t^*. By duplicating some input instances we may ensure that their number is a power of two, $t = 2^\ell$. We replace s^* and t^* with full binary trees T_s and T_t of height ℓ, rooted at s^r and t^r. In the directed case, the edges in the tree T_s are directed towards the leaves, whereas the edges in the tree T_t are directed towards the root t^r. For each instance $(G_j, s_j, t_j, \mathcal{C}_j)$ we identify s_j with one leaf in T_s and t_j with one leaf in T_t, so that each instance is assigned to different leaves in T_s and T_t. Finally, we connect each vertex of the trees T_s and T_t with r using one or two unattached paths of length two, so that $d_G(s^r) = d_G(t^r) = 3$

($d_G^{out}(s^r) = 2 = 1 + d_G^{in}(s^r)$ and $d_G^{in}(t^r) = 2 = 1 + d_G^{out}(t^r)$ in the directed case) and each other vertex in T_s and T_t is of degree 4 and, in the directed case, balanced.

As for the vertex r, we replace it with a (directed) cycle of length $d_G(r)/2$, with each vertex on the cycle adjacent to exactly two edges previously incident to r (one incoming arc and one outgoing arc in the directed case). Finally, we set $k = 2\ell + \max_{i=1}^{t} |V(G'_i)| - 1 = O(\log t + \max_{i=1}^{t} |V(G_i)| + |\mathcal{C}_i|)$. Note that now a minimum solution S to the constructed DIRECTED or UNDIRECTED EULERIAN EDGE DELETION instance needs to induce a simple path $\mathcal{P}'$ from s^r to t^r that first goes down the tree T_s to a leaf s_j (for some $1 \leq j \leq t$), then traverses the graph G'_j, inducing a solution $\mathcal{P}$ to the instance $(G_j, s_j, t_j, \mathcal{C}_j)$, and finally goes up the tree T_t starting at the leaf t_j. □

C Omitted proofs on vertex-deletion problems (Section 5)

Proof (of Theorem 15). **Balanced case.** First, we present an FPT-reduction from the DISJOINT SET COVER problem to DIRECTED BALANCED NODE DELETION. The input of this problem is a triple $(U, \mathcal{F}, k)$ where U is some universe, $\mathcal{F} = \{F_1, \dots, F_n\}$ is a family of subsets of U, and k is an integer. The task is to decide whether there is a collection $\mathcal{H} \subseteq \mathcal{F}$ with $|\mathcal{H}| \leq k$ that covers each element of U exactly once, i.e., such that the sets in $\mathcal{H}$ are pairwise disjoint and their union is U. Given such an input, we are going to construct a directed graph G such that (G, k) is a YES-instance of DIRECTED BALANCED NODE DELETION if and only if $(U, \mathcal{F}, k)$ is a YES-instance of DISJOINT SET COVER. Moreover, the presented reduction will be polynomial-time computable. As DISJOINT SET COVER is NP-hard, and also W[1]-hard with parameter k [25], this suffices to prove the theorem.

Given $(U, \mathcal{F}, k)$, for any $u \in U$, let $n(u)$ denote the number of sets in $\mathcal{F}$ that contain u. For each $u \in U$, we introduce two vertices u^1 and u^2 in G, and connect them by $n(u) - 1$ unattached paths of length $k + 2$, each starting from u^1 and leading to u^2. We denote by D the set of all internal vertices on these paths, each having degree 2 in G. Furthermore, for each $F_i \in \mathcal{F}$ we introduce a vertex f_i, and add the arcs (f_i, u^1) and (u^2, f_i) for each $u \in F_i$. This finishes the construction of G. It is not hard to see that the reduction is indeed polynomial.

Now, suppose that $G \setminus S$ is balanced for some $S \subseteq V(G)$, $|S| \leq k$. Note that if S contains any vertex d on a path of length $k + 2$ leading from some u^1 to u^2 (allowing $d = u^1$ or $d = u^2$), then all $k + 1$ internal vertices of this path should be in S, which contradicts $|S| \leq k$. Thus, we get that $S \subseteq \{f_i \mid F_i \in \mathcal{F}\}$. Observe also that for each $u \in U$, we get $d_G^{in}(u^1) = d_G^{out}(u^1) + 1 = d_{G \setminus S}^{out}(u^1) + 1$, hence the deletion of S must decrease the indegree of each vertex u^1 ($u \in U$) by exactly one. By the definition of G, this means that the sets F_i for $f_i \in S$ form a family of at most k pairwise disjoint sets together covering U, as required.

For the other direction, it is straightforward to see that if $\mathcal{S} \subseteq \mathcal{F}$ is a solution for the DISJOINT SET COVER instance, then deleting the vertex set $\{f_i \mid F_i \in \mathcal{S}\}$ from G results in a balanced graph. This observation relies on the fact that $d_G^{in}(u^1) = d_G^{out}(u^1) + 1$ and $d_G^{in}(u^2) = d_G^{out}(u^2) - 1$ hold for each $u \in U$, and that $d^{in}(v) = d^{out}(v)$ holds for each remaining vertex v. This proves our statement for DIRECTED BALANCED NODE DELETION.

Eulerian case. Now, we give a reduction from DIRECTED BALANCED NODE DELETION problem to DIRECTED EULERIAN NODE DELETION. Given an input (G, k) we construct (G', k) in polynomial time such that there is a set $S \subseteq V(G)$ with $|S| \leq k$ for which $G \setminus S$ is balanced if and only if there is a set $S' \subseteq V(G')$ with $|S'| \leq k$ for which $G' \setminus S'$ is Eulerian.

To construct G', we simply add to G a new vertex r, and connect each vertex to r by a pair of twin arcs. On the one hand, if $G' \setminus S'$ is Eulerian for some $S' \subseteq V(G')$, then $G \setminus (S' \setminus \{r\})$ must be balanced, as deleting r from the balanced graph $G' \setminus S'$ still yields a balanced graph. On the other hand, if $G \setminus S$ is balanced for some $S \subseteq V(G)$, then observe that $G' \setminus S$ is balanced as well. Furthermore, since each vertex in $G' \setminus S$ is connected by a pair of twin arcs to r, $G' \setminus S$ is Eulerian as well, finishing the proof. $\square$

We prove Theorem 16 in two steps. Theorem 17 deals with the cases where we aim for an even or a balanced graph, while Theorem 18 handles the two Eulerian cases. Let Δ_a be the maximum degree taken over all allowed vertices.

Theorem 17. UNDIRECTED EVEN *and* DIRECTED BALANCED NODE DELETION WITH FORBIDDEN NODES *remain W[1]-hard with parameter k, even if* $\Delta_a = 4$.

Proof. **Undirected case.** We present a parameterized reduction from the W[1]-hard MULTICOLOURED CLIQUE problem [15] to UNDIRECTED EVEN NODE DELETION WITH FORBIDDEN NODES with $\Delta_a = 4$. The input of MULTICOLOURED CLIQUE is a graph $G = (V, E)$ and an integer k together with a partition $V_1, V_2, \ldots, V_k$ of V where each V_i is an independent set; the task is to find a clique of size k in G.

W.l.o.g. we can assume that for each vertex v and for each V_j it holds that $|N(v) \cap V_j|$ is even. Indeed, this can be achieved without changing the answer to our instance, by putting a new vertex $a^{i,j}$ into V_j and connecting it with all vertices of V_i having an odd number of neighbours in V_j for each i and j (if there is such a vertex at all), and finally adjusting the parity of the degrees of the auxiliary vertices by connecting pairs $a^{i,j}$ and $a^{j,i}$ where this is necessary. Note that the vertex $a^{i,j}$ cannot be contained in any k-clique in G, as $N(a^{i,j}) \subseteq V_j$.

We are going to construct an instance (G', F, k') of UNDIRECTED EVEN NODE DELETION WITH FORBIDDEN NODES with $k' = k^2 + k$. For each vertex $v \in V_i$, we build a node gadget G_v as follows. We introduce vertices v^j for each $0 \leq j \leq k+1$ and a vertex $e_{v,z}$ for each $z \in N_G(v)$. For each $z \in N_G(v) \cap V_j$ where $j < i$ we connect $e_{v,z}$ with v^j and v^{j+1}, and for each $z \in N_G(v) \cap V_j$ where $j > i$ we connect $e_{v,z}$ with v^{j-1} and v^j. For each relevant j we denote by $A^j(v)$ the vertices connected to both v^j and v^{j+1}. Furthermore, we introduce edges v^0v^1, v^1v^k, and v^kv^{k+1}; this finishes the definition of G_v. Notice that each vertex in $G_v \setminus \{v^0, v^{k+1}\}$ has even degree.

Let G' contain the disjoint union of graphs G_v for $v \in V$, and let us add the vertex sets $P = \{s_i, t_i \mid 1 \leq i \leq k\}$ and $D = \{d_{x,y}, d_{y,x} \mid xy \in E\}$ to G'. We connect each s_i with the vertices v^0 where $v \in V_i$, and similarly, each t_i with the vertices v^{k+1} where $v \in V_i$. In addition, for each edge xy in G we connect the vertices $e_{x,y}, d_{x,y}, e_{y,x}, d_{y,x}$ in this order via a cycle of length 4. To finish the construction of G', we add the edge s_it_i in case s_i (and hence t_i) would have an even degree otherwise. Notice that the odd degree vertices in G' are exactly the vertices of P. Finally, we let the set of forbidden vertices to be $F = \{v^j \mid v \in V, 1 \leq j \leq k\} \cup D \cup P$. As the allowed vertices are only the vertices of the form v^0, v^{k+1}, or $e_{x,y}$, the claimed property $\Delta_a = 4$ indeed holds.

First suppose that $X = \{x_1, \dots, x_k\}$ is a clique in G with each $x_i \in V_i$. We prove that $S = \{x^0, x^{k+1} \mid x \in X\} \cup \{e_{x,y}, e_{y,x} \mid x, y \in X, x \neq y\}$ is a solution for (G', F, k'). Clearly, $|S| = k^2 + k = k'$ and $S \cap F = \emptyset$ hold, so it suffices to show that $G' \setminus S$ is an even graph. As X contains exactly one vertex from each partition V_i, each vertex in P has one neighbour in S, so the vertices of P will become even in $G' - S$. Regarding D, only those vertices $d_{x,y}$ and $d_{y,x}$ have a neighbour in S, for which $x, y \in X$ holds, and these vertices will lose exactly two neighbours, namely $e_{x,y}$ and $e_{y,x}$, when deleting S. A vertex v^j can only have a neighbour in S if $v \in X$, and in such a case it is not hard to verify that v^j will have exactly two neighbours in S. This shows that the first direction of the reduction is sound.

For the other direction, suppose that S is a solution for (G', F, k'). For each i, the vertex s_i must have at least one neighbour in S; suppose that v^0 is such a vertex. Then, since v^1 is connected to v^0 and $N(v^1) \setminus \{v^0\} = A^1(v)$, we know that $|A^1(v) \cap S|$ is odd. Using that $N(v^j) \setminus A^{j-1}(v) = A^j(v)$ for each $2 \leq j \leq k-1$, and that $v^1, \dots, v^k$ are forbidden vertices, we can deduce for each $j = 2, \dots, k-1$ that $|A^j(v) \cap S|$ must be odd as well. Hence, $v^{k+1} \in S$ follows. Therefore, if S contains a vertex v^0, then it must contain altogether at least $k+1$ vertices from the node gadget G_v. By our bound on the size of S, we obtain that S contains vertices from exactly k node gadgets, and if S contains a vertex from G_v, then it contains v^0, v^{k+1}, and one vertex from each of the sets $A^1(v), \dots, A^{k-1}(v)$. Let X contain those vertices x in G for which $x^0 \in S$, and let Y contain those pairs (x, y) for which $e_{x,y} \in S$. By the previous observations, we know $|Y| = k(k-1)$ and $|X| = k$. Suppose that $(x, y) \in Y$. Note that $x \in X$ and $y \in N_G(x)$ are immediate from the definition of G_v. By looking at the forbidden vertices $d_{x,y}$ and $d_{y,x}$, we get $e_{y,x} \in S$, yielding $(y, x) \in Y$. Therefore, $y \in X$ as well, and thus each pair in Y must contain the end-vertices of an edge connecting two vertices of X. By the size of X and Y we get that X must be a clique of size k, finishing the proof for the undirected case.

Directed case. The reduction for the directed case is very similar to the one used in the undirected case, and hence we only describe the differences. First, we direct the edges of each cycle $e_{x,y}, d_{x,y}, e_{y,x}, d_{y,x}$ in a way that they span a directed cycle. Then for each set V_i of the partition and for each $v \in V_i$, we direct every edge e incident to a vertex of G_v, except for the edge v^1v^k, in the direction in which e is traversed when going from s_i to t_i through e. We remove the edge v^1v^k, but for each v^j with $1 \leq j \leq k-1$ we introduce a certain number of parallel arcs from v^{j+1} to v^j in order to ensure each vertex of G_v to be balanced; this means $|A^1(v)| - 1$ arcs from v^2 to v^1, $|A^{k-1}(v)| - 1$ arcs from v^k to v^{k-1}, and $|A^j(v)|$ arcs from v^{j+1} to v^j for each remaining j. As a result, each vertex of $V(G) \setminus P$ becomes balanced. Finally, we add $|V_i| - 1$ arcs from t_i to s_i, for each i. The set of forbidden vertices F and the parameter k' remains unchanged. Also, $\Delta_a = 4$ remains true.

Clearly, each s_i must lose an outgoing arc and each vertex t_i must lose an incoming arc in a solution. Arguing along similar thoughts as above, one can show that the constructed instance is equivalent with the original one.

In case we want to get rid of parallel arcs, we can simply subdivide each arc contained in a set of parallel arcs with a newly introduced forbidden vertex of degree 2. □

Theorem 18. UNDIRECTED *and* DIRECTED EULERIAN NODE DELETION WITH FORBIDDEN NODES *remain W[1]-hard with parameter k, even if* $\Delta_a = 4$.

Proof. Considering the directed version of the problem, the theorem follows from the proof of Theorem 17.

Consider the construction given in the proof of Theorem 17. Observe that each allowed vertex is only connected to forbidden vertices. Thus, if we ensure that the forbidden vertices remain in one connected component of G after deleting an arbitrary set of allowed vertices, then we also ensure that the whole graph remains connected. This can be done by introducing a new forbidden vertex r, and connecting r to each forbidden vertex by a pair of twin arcs. As each allowed vertex has the same degree as originally, we can conclude that $\Delta_a = 4$ will still hold in the transformed instance.

For the undirected version we can use the modified version of the reduction above, by connecting each forbidden vertex to r using a pair of parallel edges (or two unattached paths of length 2) instead of the twin arcs. □

Now we prove that if *every* vertex has bounded degree, then we can solve all of the α NODE DELETION WITH FORBIDDEN NODES problems, even with forbidden nodes, efficiently. The following simple algorithm works:

1. If the parameter k is negative, or we aim for an Eulerian induced subgraph and G is disconnected, then stop and return NO.
2. If the given graph G is already balanced/even/Eulerian, then stop and return the solution collected so far.
3. Otherwise, choose an arbitrary vertex v that is not balanced/even, and branch on removing from G either v or one of its neighbours. In case the deleted vertex x was forbidden, stop and return NO in the corresponding branch. Otherwise, put the deleted vertex x into the solution, decrease the value of the parameter to $k-1$, and go to Step 1.

The above algorithm can clearly be implemented to run in $O((\Delta+1)^k(|V(G)| + |E(G)|))$ time. Its soundness can be proven easily by induction w.r.t. the size of the solution, using the simple observation that if v is a vertex that is not balanced/even, then either v or one of its neighbours must be in the solution. Thus, we can conclude:

Theorem 19. *There is an* $O((\Delta+1)^k(|V(G)| + |E(G)|))$ *time algorithm that solves the* α NODE DELETION WITH FORBIDDEN NODES *problem, where* α *can be* UNDIRECTED EVEN, UNDIRECTED EULERIAN, DIRECTED BALANCED, *or* DIRECTED EULERIAN.