Abstract

Problems of the following kind have been the focus of much recent research in the realm of parameterized complexity: Given an input graph (digraph) on \(n\) vertices and a positive integer parameter \(k\), find if there exist \(k\) edges (arcs) whose deletion results in a graph that satisfies some specified parity constraints. In particular, when the objective is to obtain a connected graph in which all the vertices have even degrees – where the resulting graph is Eulerian the problem is called Undirected Eulerian Edge Deletion. The corresponding problem in digraphs where the resulting graph should be strongly connected and every vertex should have the same in-degree as its out-degree is called Directed Eulerian Edge Deletion. Cygan et al. [Algorithmica, 2014] showed that these problems are fixed parameter tractable (FPT), and gave algorithms with the running time \(2^{O(k \log k)}n^{O(1)}\). They also asked, as an open problem, whether there exist FPT algorithms which solve these problems in time \(2^{O(k)}n^{O(1)}\). It was also posed as an open problem at the School on Parameterized Algorithms and Complexity 2014, Będlewo, Poland.

In this paper we answer their question in the affirmative: using the technique of computing representative families of co-graphic matroids we design algorithms which solve these problems in time \(2^{O(k)}n^{O(1)}\). The crucial insight we bring to these problems is to view the solution as an independent set of a co-graphic matroid. We believe that this viewpoint/approach will be useful in other problems where one of the constraints that need to be satisfied is that of connectivity.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Eulerian Edge Deletion, FPT, Representative Family

1 Introduction

Many well-studied algorithmic problems on graphs can be phrased in the following way: Let \(\mathcal{F}\) be a family of graphs or digraphs. Given as input a graph (digraph) \(G\) and a positive integer \(k\), can we delete \(k\) vertices (or edges or arcs) from \(G\) such that the resulting graph (digraph) belongs to the class \(\mathcal{F}\)? Recent research in parameterized algorithms has focused on problems of this kind where the class \(\mathcal{F}\) consists of all graphs/digraphs whose vertices...
satisfy certain parity constraints [4, 9, 2, 8, 5]. In this paper we obtain significantly faster parameterized algorithms for two such problems, improving the previous best bounds due to Cygan et al. [4]. We also settle the parameterized complexity of a third problem, disproving a conjecture of Cai and Yang [2] and solving an open problem posed by Fomin and Golovach [9]. We obtain our results using recently-developed techniques for the efficient computation of representative sets of matroids.

An undirected graph $G$ is even (respectively, odd) if every vertex of $G$ has even (resp. odd) degree. A directed graph $D$ is balanced if the in-degree of each vertex of $D$ is equal to its out-degree. An undirected graph is Eulerian if it is connected and even; and a directed graph is Eulerian if it is strongly connected and balanced. Cai and Yang [2] initiated the systematic study of parameterized Eulerian subgraph problems. In this work we take up the following edge-deletion problems of this kind:

**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 in $G$ such that $G \setminus S$ is Eulerian?

**Undirected Connected Odd 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 in $G$ such that $G \setminus S$ is odd and connected?

**Directed Eulerian Edge Deletion**
- **Parameter:** $k$
- **Input:** A strongly connected directed graph $D$ and an integer $k$.
- **Question:** Does there exist a set $S$ of at most $k$ arcs in $D$ such that $D \setminus S$ is Eulerian?

Our algorithms for these problems also find such a set $S$ of edges/arcs when it exists; so we slightly abuse the notation and refer to $S$ as a solution to the problem in each case.

**Previous Work.** Cai and Yang [2] listed sixteen odd/even undirected subgraph problems in their pioneering paper, and settled the parameterized complexity of all but four. The first two problems above are among these four; Cai and Yang conjectured that these are both W[1]-hard, and so are unlikely to have fixed-parameter tractable (FPT) algorithms: those with running times of the form $f(k) \cdot n^{O(1)}$ for some computable function $f$ where $n$ is the number of vertices in the input graph. Cygan et al. [4] disproved this conjecture for the first problem: they used a novel and non-trivial application of the colour-coding technique to solve both Undirected Eulerian Edge Deletion and Directed Eulerian Edge Deletion in time $2^{O(k \log k)} n^{O(1)}$. They also posed as open the question whether there exist $2^{O(k)} n^{O(1)}$-time algorithms for these two problems. It was also posed as an open problem at the School on Parameterized Algorithms and Complexity 2014, Będlewo, Poland [3]. Fomin and Golovach [9] settled the parameterized complexity of the other two problems – not defined here – left open by Cai and Yang, but left the status of Undirected Connected Odd Edge Deletion open.

**Our Results and Methods.** We devise deterministic algorithms of running time $2^{O(k)} n^{O(1)}$ for all the three problems defined above. This answers the question of Cygan et al. [4] in the affirmative, solves the problem posed by Fomin and Golovach, and disproves the conjecture of Cai and Yang for Undirected Connected Odd Edge Deletion.

▶ **Theorem 1.1.** Undirected Eulerian Edge Deletion, Undirected Connected Odd Edge Deletion, and Directed Eulerian Edge Deletion can all be solved in
time $O(2^{(2+\omega)k} \cdot n^2m^3k^6) + m^{O(1)}$ where $n = |V(G)|$, $m = |E(G)|$ and $\omega$ is the exponent of matrix multiplication.

Our main conceptual contribution is to view the solution as an independent set of a co-graphic matroid, which we believe will be useful in other problems where one of the constraints that need to be satisfied is that of connectivity.

We now give a high-level overview of our algorithms. Given a subset of vertices $T$ of a graph $G$, a $T$-join of $G$ is a set $S \subseteq E(G)$ of edges such that $T$ is exactly the set of odd degree vertices in the subgraph $H = (V(G), S)$. Observe that $T$-joins exist only for even-sized vertex subsets $T$. The following problem is long known to be solvable in polynomial time [7].

**Min $T$-Join**

**Input:** An undirected graph $G$ and a set of terminals $T \subseteq V(G)$.

**Question:** Find a $T$-join of $G$ of the smallest size.

Consider the two problems we get when we remove the connectivity (resp. strong connectivity) requirement on the graph $G \setminus S$ from **Undirected Eulerian Edge Deletion** and **Directed Eulerian Edge Deletion**; we call these problems **Undirected Even Edge Deletion** and **Directed Balanced Edge Deletion**, respectively. Cygan et al. show that **Undirected Even Edge Deletion** can be reduced to **Min $T$-Join**, and **Directed Balanced Edge Deletion** to a minimum cost flow problem with unit costs, both in polynomial time [4]. Thus it is not the local requirement of even degrees which makes these problems hard, but the simultaneous global requirement of (strong) connectivity.

To handle this situation we turn to a matroid which correctly captures the connectivity requirement. Let $I$ be the family of all subsets $X \subseteq E(G)$ of the edge set of a graph $G$ such that the subgraph $(V(G), E(G) \setminus X)$ is connected. Then the pair $(E(G), I)$ forms a linear matroid called the co-graphic matroid of $G$ (See Section 2 for definitions). Let $T$ be the set of odd-degree vertices of the input graph $G$. Observe that for **Undirected Eulerian Edge Deletion**, the solution $S$ we are after is both a $T$-join and an independent set of the co-graphic matroid of $G$. We exploit this property of $S$ to design a dynamic programming algorithm which finds $S$ by computing “representative sub-families” [10, 12, 14, 15] of certain families of edge subsets in the context of the co-graphic matroid of $G$. We give simple characterizations of solutions which allow us to do dynamic programming, where at every step we only need to keep a representative family of the family of partial solutions where each partial solution is an independent set of the corresponding co-graphic matroid. To find the desired representative family of partial solutions we use the algorithm by Lokshtanov et al. [13]. Our methods also imply that **Undirected Connected Odd Edge Deletion** admits an algorithm with running time $2^{O(k)}n^{O(1)}$.

2 Preliminaries

Throughout the paper we use $\omega$ to denote the exponent in the running time of matrix multiplication, the current best known bound for which is $\omega < 2.373$ [17].

**Graphs and Directed Graphs.** We use “graph” to denote simple graphs without self-loops, directions, or labels, and “directed graph” or “digraph” for simple directed graphs without self-loops or labels. We use standard terminology from the book of Diestel [6] for those graph-related terms which we do not explicitly define. In general we use $G$ to denote a graph and $D$ to denote a digraph. We use $V(G)$ and $E(G)$, respectively, to denote the vertex and edge sets of a graph $G$, and $V(D)$ and $A(D)$, respectively, to denote the vertex and arc
sets of a digraph $D$. For an edge set $E' \subseteq E(G)$, we use (i) $V(E')$ to denote the set of end
vertices of the edges in $E'$, (ii) $G \setminus E'$ to denote the subgraph $G' = (V(G), E(G) \setminus E')$ of
$G$, and (iii) $G(E')$ to denote the subgraph $(V(E'), E')$ of $G$. The terms $V(A')$, $D \setminus A'$, and
$D(A')$ are defined analogously for an arc subset $A' \subseteq A(D)$.

If $P$ is a path from vertex $u$ to vertex $v$ in graph $G$ (or in digraph $D$) then we say that
(i) $P$ connects $u$ and $v$, (ii) $u,v$ are, respectively, the initial vertex and the final vertex of
$P$, and (iii) $u,v$ are the end vertices of path $P$. Let $P_1 = x_1x_2\ldots x_r$ and $P_2 = y_1y_2\ldots y_s$ be
two edge-disjoint paths in graph $G$. If $x_r = y_1$ and $V(P_1) \cap V(P_2) = \{x_r\}$, then we use $P_1P_2$
to denote the path $x_1x_2\ldots x_2y_2\ldots y_s$. A path system $\mathcal{P}$ in graph $G$ (resp., digraph $D$) is
an ordered collection of paths in $G$ (resp. in $D$), and it is edge-disjoint if no two paths in
the system share an edge. We use $V(\mathcal{P})$ and $E(\mathcal{P})$ ($A(\mathcal{P})$ for a path system in digraph) for
the set of vertices and edges, respectively, in a path system $\mathcal{P}$. We say that a path system
$\mathcal{P} = \{P_1, \ldots, P_r\}$ ends at a vertex $u$ if the path $P_r$ ends at $u$, and $u$ is called the final vertex
of $\mathcal{P}$. We use $V^+ (\mathcal{P})$ to denote the set of end vertices of paths in a path system $\mathcal{P}$. For a
path system $\mathcal{P}$ in a digraph $D$, we use $V^D(\mathcal{P})$ and $V^I(\mathcal{P})$, respectively, to denote the set of
initial vertices and the set of final vertices, respectively, of paths in $\mathcal{P}$. For a path system
$\mathcal{P} = \{P_1, \ldots, P_r\}$ and an edge/arc $(u,v)$, we define $\mathcal{P} \circ (u,v)$ as follows.

$$\mathcal{P} \circ (u,v) = \begin{cases} \{P_1, \ldots, P_r, v\} & \text{if } u \text{ is the final vertex of } P_r \text{ and } v \notin V(P_r) \\ \{P_1, \ldots, P_r, u\} & \text{if } u \text{ is not the final vertex of } P_r \end{cases}$$

A directed graph $D$ is *strongly connected* if for any two vertices $u$ and $v$ of $D$, there is
a directed path from $u$ to $v$ and a directed path from $v$ to $u$ in $D$. A digraph $D$ is *weakly
connected* if the underlying undirected graph is connected. The in-neighborhood of a vertex $v$
in $D$ is the set $N^-_D(v) = \{u \mid (u,v) \in A(D)\}$, and the in-degree of $v$ in $D$ is $d^-_D(v) = |N^-_D(v)|$.
The *out-neighborhood* of $v$ is the set $N^+_D(v) = \{w \mid (v,w) \in A(D)\}$, and its out-degree is
$d^+_D(v) = |N^+_D(v)|$.

**Co-Graphic Matroids.** The *co-graphic matroid* of a connected graph $G$ is defined as
$M = (E(G), \mathcal{I})$ where $\mathcal{I} = \{S \subseteq E(G) \mid (G \setminus S) \text{ is connected}\}$. It is a linear matroid and, given a
graph $G$, a representation of the co-graphic matroid of $G$ over the finite field $\mathbb{F}_2$ can be found
in polynomial time [14, 16]. The rank of the cographic matroid of a connected graph $G$ is
$(|E(G)| - |V(G)| + 1)$. We use $M_G$ to denote the co-graphic matroid of a graph $G$. For a
directed graph $D$ we use $M_D$ to denote the co-graphic matroid of the underlying undirected
digraph of $D$.

Let $A$ be a family of path systems in a graph $G$. Let $e = (u,v)$ be an edge in $G$ (or an
arc in $D$), and let $M = (E, \mathcal{I})$ be the co-graphic matroid of graph $G$ (or of digraph $D$). We
use $A \bullet \{e\}$ to denote the family of path systems

$$A \bullet \{e\} = \{\mathcal{P}' \circ e \mid \mathcal{P} \in A, e \notin E(\mathcal{P}), E(\mathcal{P}') \in \mathcal{I} \}.$$ 

**Representative Families of Matroids.** The notion of representative families of matroids
and their fast computation play key roles in our algorithms.

> **Definition 2.1.** [10, 14] Given a matroid $M = (E, \mathcal{I})$, a family $\mathcal{S}$ of subsets of $E$, and
a non-negative integer $q$, we say that a subfamily $\hat{\mathcal{S}} \subseteq \mathcal{S}$ is *$q$-representative* for $\mathcal{S}$ if the
following holds. For every set $Y \subseteq E$ of size at most $q$, if there is a set $X \in \mathcal{S}$ disjoint from $Y$
with $X \cup Y \in \mathcal{I}$, then there is a set $\hat{X} \in \hat{\mathcal{S}}$ disjoint from $Y$ with $\hat{X} \cup Y \in \mathcal{I}$. 

---

FSTTCS 2015
In other words, if some independent set \( X \) in \( S \) can be extended to a larger independent set by a set \( Y \) of at most \( q \) new elements, then there is a set \( \hat{X} \) in \( \hat{S} \) that can be extended by the same set \( Y \). If \( \hat{S} \subseteq S \) is \( q \)-representative for \( S \) we write \( \hat{S} \subseteq_{rep} S \).

In this paper we are interested in linear matroids and in representative families derived from them. The following theorem states the key algorithmic result which we use for the computation of representative families of linear matroids.

**Theorem 2.2** ([13]). Let \( M = (E, I) \) be a linear matroid of rank \( n \) and let \( S = \{S_1, \ldots, S_l\} \) be a family of independent sets, each of size \( b \). Let \( A \) be an \( n \times |E| \) matrix representing \( M \) over a field \( F \), where \( F = \mathbb{F}_p \) or \( F \) is \( \mathbb{Q} \). Then there is deterministic algorithm which computes a representative set \( \hat{S} \subseteq_{rep} S \) of size at most \( nb^{(b+q)Q} \), using \( O\left(\left(b^{(b+q)}Q\right)bn^2 + t\left(b+q\right)^{w-1}(bn)^{w-1}\right) + (n + |E|)^{O(1)} \) operations over the field \( F \).

### 3 Undirected Eulerian Edge Deletion

In this section we describe our \( 2^{O(k)}n^{O(1)} \)-time algorithm for **Undirected Eulerian Edge Deletion**. Let \( (G, k) \) be an instance of the problem. Cygan et al. [4] observed the following characterization.

**Observation 3.1.** A set \( S \subseteq E(G) \); \( |S| \leq k \) of edges of a graph \( G \) is a solution to the instance \( (G, k) \) of **Undirected Eulerian Edge Deletion** if and only if it satisfies the following conditions:

(a) \( G \setminus S \) is a connected graph; and,

(b) \( S \) is a \( T \)-join where \( T \) is the set of all odd degree vertices in \( G \).

For a designated set \( T \subseteq V(G) \) of terminal vertices of graph \( G \), we call a set \( S \subseteq E(G) \) a **co-connected** \( T \)-join of graph \( G \) if (i) \( G \setminus S \) is connected and (ii) \( S \) is a \( T \)-join. From Observation 3.1 we get that the **Undirected Eulerian Edge Deletion** problem is equivalent to checking whether the given graph \( G \) has a co-connected \( T \)-join of size at most \( k \), where \( T \) is the set of all odd-degree vertices in \( G \). We present an algorithm which finds a co-connected \( T \)-join for an arbitrary (even-sized) set of terminals \( T \) within the claimed time-bound. That is, we solve the following more general problem:

<table>
<thead>
<tr>
<th>Co-Connected T-Join</th>
<th>Parameter: ( k )</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> A connected graph ( G ), an even-sized subset ( T \subseteq V(G) ) and an integer ( k ).</td>
<td></td>
</tr>
<tr>
<td><strong>Question:</strong> Does there exist a co-connected ( T )-join of ( G ) of size at most ( k )?</td>
<td></td>
</tr>
</tbody>
</table>

We design a dynamic programming algorithm for this problem where the partial solutions which we store satisfy the first property of co-connected \( T \)-join and “almost satisfy” the second property. To limit the number of partial solutions which we need to store, we compute and store instead, at each step, a representative family of the partial solutions in the corresponding co-graphic matroid. We start with the following characterization of the \( T \)-joins of a graph \( G \).

**Proposition 3.2** ([11, Proposition 1.1]). Let \( T \) be an even-sized subset of vertices of a graph \( G \), and let \( \ell = \frac{|T|}{2} \). A subset \( S \) of edges of \( G \) is a \( T \)-join of \( G \) if and only if \( S \) can be expressed as a union of the edge sets of (i) \( \ell \) paths which connect disjoint pairs of vertices in \( T \), and (ii) zero or more cycles, where the paths and cycles are all pairwise edge-disjoint.

This proposition yields the following useful property of inclusion-minimal co-connected \( T \)-joins (minimal co-connected \( T \)-joins for short) of a graph \( G \).
Lemma 3.3 (♠). Let $T$ be an even-sized subset of vertices of a graph $G$, and let $\ell = \frac{|T|}{2}$. Let $S$ be a minimal co-connected $T$-join of $G$. Then (i) the subgraph $G(S)$ is a forest, and (ii) the set $S$ is a union of the edge-sets of $\ell$ pairwise edge-disjoint paths which connect disjoint pairs of vertices in $T$.

Note that the set of paths described in Lemma 3.3 are just pairwise edge-disjoint. Vertices (including terminals) may appear in more than one path as internal vertices. A partial converse of the above lemma follows directly from Proposition 3.2.

Lemma 3.4 (♠). Let $T$ be an even-sized subset of vertices of a graph $G$, and let $\ell = \frac{|T|}{2}$. Let a subset $S \subseteq E(G)$ of edges of $G$ be such that (i) $G \setminus S$ is connected, and (ii) $S$ is a union of the edge-sets of $\ell$ pairwise edge-disjoint paths which connect disjoint pairs of vertices in $T$. Then $S$ is a co-connected $T$-join.

An immediate corollary of Lemma 3.3 is that for any set $T \subseteq V(G)$, any $T$-join of the graph $G$ has at least $|T|/2$ edges. Hence if $|T| > 2k$ then we can directly return No as the answer for Co-Connected $T$-Join. So from now on we assume that $|T| \leq 2k$. From Lemmas 3.3 and 3.4 we get that to solve Co-Connected $T$-Join it is enough to check for the existence of a pairwise edge-disjoint collection of paths $P = \{P_1, \ldots, P_{|T|/2}\}$ such that (i) the subgraph $(G \setminus E(P))$ is connected, (ii) $|E(P)| \leq k$, and (iii) the paths in $P$ connect disjoint pairs of terminals in $T$. We use dynamic programming to find such a path system.

We first state some notation which we need to describe the dynamic programming table. We use $Q$ to denote the set of all path systems in $G$ which satisfy the above conditions. For $1 \leq i \leq k$ we use $Q^{(i)}$ to denote the set of all potential partial solutions of size $i$; each $Q^{(i)}$ is a collection of path systems $Q^{(i)} = \{Q^{(i)}_{1}, \ldots, Q^{(i)}_{k}\}$ where each path system $Q^{(i)}_{s} = \{P_{1}, \ldots, P_{r}\} \in Q^{(i)}$ has the following properties:

(i) The paths $P_{1}, \ldots, P_{r}$ are pairwise edge-disjoint.

(ii) The end-vertices of the paths $P_{1}, \ldots, P_{r}$ are all terminals and are pairwise disjoint, with one possible exception. One end-vertex (the final vertex) of the path $P_{r}$ may be a non-terminal, or a terminal which appears as an end-vertex of another path as well.

(iii) $|E(P^{(i)}_{s})| = i$, and the subgraph $G \setminus E(P^{(i)}_{s})$ is connected.

Note that the only ways in which a partial solution $P^{(i)}_{s}$ may violate one of the conditions in Lemma 3.4 are: (i) it may contain strictly less than $\frac{|T|}{2}$ paths, and/or (ii) there may be a path $P_{r}$ (and only one such), which has one end-vertex $v$, which is a non-terminal or is a terminal which is an end-vertex of another path as well. For a path system $P = \{P_{1}, \ldots, P_{r}\}$ and $u \in V(G) \cup \{\epsilon\}$, we use $W(P, u)$ to denote the following set.

$$W(P, u) = \begin{cases} V^{e}(P) & \text{if } u = \epsilon \\ (V^{e}(P \setminus \{P_{r}\})) \cup \{v \mid v \text{ is the initial vertex of } P_{r}\} & \text{if } u \neq \epsilon \end{cases}$$

Finally, for each $1 \leq i \leq k$, $T' \subseteq T$, and $v \in (V(G) \cup \{\epsilon\})$ we define

$$Q[i, T', v] = \{P \in Q^{(i)} \mid W(P, v) = T', \text{ and if } v \neq \epsilon \text{ then } v \text{ is the final vertex of } P\}$$

as the set of all potential partial solutions of size $i$ whose set of end vertices is exactly $T' \cup \{v\}$. Observe from this definition that in the case $v = \epsilon$, the last path $P_{r}$ in each path system $P = \{P_{1}, \ldots, P_{r}\} \in Q[i, T', \epsilon]$ ends at a “good” vertex; that is, at a terminal vertex which is different from all the end vertices of the other paths $P_{1}, \ldots, P_{r-1}$ in $P$.

Proof of results labelled with ♠ will appear in the full version of the paper.
Finding Even Subgraphs Even Faster

It is not difficult to see that this definition of $Q[i, T', v]$ is a correct notion of a partial solution for Co-Connected $T$-Join:

\begin{definition}
Let $(G, T, k)$ be a Yes instance of Co-Connected $T$-Join which has a minimal solution of size $k' \leq k$, and let $\ell = \lceil \frac{T}{2} \rceil$. Then for each $1 \leq i \leq k'$ there exist $T' \subseteq T$, $v \in (V(G) \cup \{\epsilon\})$, and path systems $P = \{P_1, P_2, \ldots, P_r\} \subseteq Q[i, T', v]$ and $P' = \{P'_r, P'_{r+1}, \ldots, P'_t\}$ in $G$ (where $E(P'_v) = \emptyset$ if $v = \epsilon$) such that (i) $E(P) \cap E(P') = \emptyset$, (ii) $P, P'_r$ is a path in $G$, and (iii) $P \cup P' = \{P_1, P_2, \ldots, P'_r, P'_{r+1}, \ldots, P'_t\}$ is an edge-disjoint path system whose edge set is a solution to the instance $(G, T, k)$.
\end{definition}

\begin{proof}
Let $\hat{P} = \{\hat{P}_1, \ldots, \hat{P}_t\}$ be a path system in graph $G$ which witnesses -- as per Lemma 3.3 -- the fact that $(G, T, k)$ has a solution of size $k'$. If $i = \sum_{j=1}^{r-1} |E(\hat{P}_j)|$ for some $1 \leq r \leq \ell$ then the path systems $P = \{\hat{P}_1, \hat{P}_2, \ldots, \hat{P}_r\} \subseteq Q[i, T', v]$ and $P' = \emptyset, \hat{P}_{r+1}, \hat{P}_{r+2}, \ldots, \hat{P}_t$ satisfy the claim, where $T' = T \cap V^c(P)$ and $v = \epsilon$.

If $i$ takes another value then let $1 \leq r \leq \ell$ be such that $\sum_{j=i}^{r-1} |E(\hat{P}_j)| < i < \sum_{j=1}^{r} |E(\hat{P}_j)|$. “Split” the path $\hat{P}_r$ as $\hat{P}_r = \hat{P}^1_r \hat{P}^2_r$ such that $\sum_{j=i}^{r-1} |E(\hat{P}^1_j)| + |E(\hat{P}^2_r)| = i$. Now the path systems $P = \{\hat{P}_1, \hat{P}_2, \ldots, \hat{P}_{r-1}, \hat{P}^1_r\} \subseteq Q[i, T', v]$ and $P' = \{\hat{P}_2, \hat{P}_{r+1}, \hat{P}_{r+2}, \ldots, \hat{P}_t\}$ satisfy the claim, where $T' = T \cap V^c(P)$ and $v$ is the final vertex of the path $\hat{P}^1_r$.
\end{proof}

Given this notion of a partial solution the natural dynamic programming approach is to try to compute, in increasing order of $1 \leq i \leq k$, partial solutions $Q[i, T', v]$ for all $T' \subseteq T$, $v \in (V(G) \cup \{\epsilon\})$ at step $i$. But this is not feasible in polynomial time because the sets $Q[i, T', v]$ can potentially grow to sizes exponential in $|V(G)|$. Our way out is to observe that to reach a final solution to the problem we do not need to store every element of a set $Q[i, T', v]$ at each intermediate step. Instead, we only need to store a representative family $R$ of partial solutions corresponding to $Q[i, T', v]$, where $R$ has the following property: If there is a way of extending -- in the sense of Lemma 3.5-- any partial solution $P \in Q[i, T', v]$ to a final solution then there exists a $\hat{P} \in R$ which can be extended the same way to a final solution.

Observe now that our final solution and all partial solutions are independent sets in the co-graphic matroid $M_G$ of the input graph $G$. We use the algorithm of Lokshhtanov et al. [13]—see Theorem 2.2—to compute these representative families of potential partial solutions at each intermediate step. In step $i$ of the dynamic programming we store, in place of the set $Q[i, T', v]$, its $(k-i)$-representative set $Q[i, T', v] \subseteq \bigcap_{v \in P} Q[i, T', v]$ with respect to the co-graphic matroid $M_G$; for the purpose of this computation we think of each element $P$ of $Q[i, T', v]$ as the edge set $E(P)$. Lemma 3.8 below shows that this is a safe step. Whenever we talk about representative families in this section, it is always with respect to the co-graphic matroid $M_G$ associated with $G$; we do not explicitly mention the matroid from now on. We start with the following definitions.

\begin{definition}
Let $1 \leq i \leq k$, $T' \subseteq T$, $\ell = \lceil \frac{T}{2} \rceil$ and $v \in (V(G) \cup \{\epsilon\})$, and let $Q[i, T', v]$ be the corresponding set of partial solutions. Let $P = \{P_1, \ldots, P_r\}$ be a path system in the set $Q[i, T', v]$. Let $P' = \{P'_r, P'_{r+1}, \ldots, P'_t\}$ be a path system in $G$ (where $E(P'_v) = \emptyset$ if $v = \epsilon$) such that (i) $E(P') \leq (k-i)$, (ii) $P, P'_r$ is a path in $G$, (iii) $P \cup P' = \{P_1, P_2, \ldots, P'_r, P'_{r+1}, \ldots, P'_t\}$ is an edge-disjoint path system that connects disjoint pairs of terminals in $T$, (iv) $V^c(P \cup P') = T$ and (v) $G \setminus (E(P) \cup E(P'))$ is connected. Then $P'$ is said to be an extender for $P$.
\end{definition}

\begin{definition}
Let $1 \leq i \leq k$, $T' \subseteq T$ and $v \in (V(G) \cup \{\epsilon\})$, and let $Q[i, T', v]$ be the corresponding set of partial solutions. We say that $J[i, T', v] \subseteq Q[i, T', v]$ is a path-system
and vertices could repeat but not the edges). Now we prove that the "path system"

\begin{itemize}
\item Case 1: \( P \)
\item Case 2: \( \hat{V} = e \)
\end{itemize}

\textbf{Proof.} We first prove the first claim. The second claim of the lemma follows by similar
arguments. Let \( \mathcal{Q}[i, T', v] \subseteq \mathcal{Q}[i, T', v] \), let \( \mathcal{P} = \{ P_1, \ldots, P_r \} \) be a path system in
the set \( \mathcal{Q}[i, T', v] \), and let \( \mathcal{P}' = \{ P'_1, P'_{r+1}, \ldots, P'_r \} \) be a path system in \( G \) (where \( E(P'_r) = \emptyset \)
if \( v = e \) which is an extender for \( \mathcal{P} \). We have to show that there exists a path system
\( \mathcal{P}^* \subseteq \mathcal{Q}[i, T', v] \) such that \( \mathcal{P}' \) is an extender for \( \mathcal{P}^* \) as well. Since \( \mathcal{P}' \) is an extender
for \( \mathcal{P} \) we have, by definition, that (i) \( |E(P'_r)| \leq (k - i) \), (ii) \( P, P'_r \) is a path in \( G \), (iii)
\( \mathcal{P} \cup \mathcal{P}' = \{ P_1, \ldots, P_r, P'_{r+1}, \ldots, P'_r \} \) is an edge-disjoint path system that connects disjoint
pairs of terminals in \( T \), (iv) \( V'(\mathcal{P} \cup \mathcal{P}') = T \) and (v) \( G \backslash (E(\mathcal{P}) \cup E(\mathcal{P}')) \) is connected.

Since (i) \( \mathcal{P} \subseteq \mathcal{Q}[i, T', v] \), (ii) \( E(\mathcal{P}) \cap E(\mathcal{P}') = \emptyset \), (iii) \( G \backslash (E(\mathcal{P}) \cup E(\mathcal{P}')) \) is connected,
and (iv) \( \mathcal{Q}[i, T', v] \subseteq \mathcal{Q}[i, T', v] \), there exists a path system \( \mathcal{P}^* = \{ P'_1, P'_2, \ldots, P'_r \} \) \( i \), \( \mathcal{P} \)
and let \( \mathcal{P}^* \) be a path system which connects disjoint pairs of terminals in \( T \). We show that the "path system" \( \mathcal{P}^* \cup \mathcal{P}' \)
connects disjoint pairs of terminals in \( T \), but for a pair which is connected by an Eulerian
walk. We now consider two cases for the "vertex" \( v \).

\textbf{Case 1:} \( v = \epsilon \). In this case, since \( \mathcal{P} \) and \( \mathcal{P}^* \) both belong to the set \( \mathcal{Q}[i, T', \epsilon] \) we have that \( V'(\mathcal{P}) = V'(\mathcal{P}^*) = T' \). Also \( E(\mathcal{P}') = \emptyset \), and \( \mathcal{P} \cup \mathcal{P}' \) is the path system
\( \{ P_1, \ldots, P_r, P'_{r+1}, P'_{r+2}, \ldots, P'_r \} \) with exactly \( \ell = \frac{|T|}{2} \) paths which connect disjoint pairs of
terminals in \( T \). Since \( V'(\mathcal{P} \cup \mathcal{P}') = T \), \( \mathcal{P} = \{ P_1, \ldots, P_r \} \) and \( V'(\mathcal{P}) = T' \), we get that
\( V'(\mathcal{P}') = T \backslash T' \). Now since \( V'(\mathcal{P}^*) = T' \) it follows that \( \mathcal{P}^* \cup \mathcal{P}' \) is a path system which
connects disjoint pairs of terminals in \( T \).

\textbf{Case 2:} \( v \neq \epsilon \). In this case, since \( \mathcal{P} \) and \( \mathcal{P}^* \) both belong to the set \( \mathcal{Q}[i, T', v] \) we have that \( V'(\mathcal{P}) = V'(\mathcal{P}^*) = T' \cup \{ v \} \), and that the final vertex of each of these two path
systems is \( v \). Also \( \mathcal{P} \cup \mathcal{P}' = \{ P_1, \ldots, P_r, P'_{r+1}, P'_{r+2}, \ldots, P'_r \} \) is a path system with exactly
\( \ell = \frac{|T|}{2} \) paths which connect disjoint pairs of terminals in \( T \). Since (i) \( V'(\mathcal{P} \cup \mathcal{P}') = T \), (ii)
\( \mathcal{P} = \{ P_1, \ldots, P_r \} \), (iii) \( \mathcal{P}' = \{ P'_{r+1}, \ldots, P'_r \} \), (iv) \( V'(\mathcal{P}) = T \cup \{ v \} \), and (v) the final vertex
of the path \( P_1 \) in \( \mathcal{P} \) is \( v \), we get that (i) the initial vertex of the path \( P'_r \) in \( \mathcal{P}' \) is \( v \) and
(ii) \( V'(\mathcal{P}') = (T \backslash T') \cup \{ v \} \). Now since \( V'(\mathcal{P}^*) = T' \cup \{ v \} \) and (ii) the final vertex of
\( \mathcal{P}^* \) is \( v \) it follows that \( \mathcal{P}^* \cup \mathcal{P}' \) is a path system which connects disjoint pairs of terminals in
\( T \), where \( P'_r P'_r \) which could be an Eulerian walk.
Thus, we have shown that $\mathcal{P}^* \cup \mathcal{P}'$ connects disjoint pairs of terminals in $T$ with paths, except for $P_r^* P_r'$ which could be an Eulerian walk. Combining this with Proposition 3.2 and the fact that $G \setminus (E(\mathcal{P}^*) \cup E(\mathcal{P}'))$ is connected, we get that $E(\mathcal{P}^*) \cup E(\mathcal{P}')$ is a co-connected $T$-join of $G$.

Finally, we show that $\mathcal{P}^* \cup \mathcal{P}'$ is a path system. Towards this we only need to show that $P_r^* P_r'$ is not an Eulerian walk but a path. Observe that $|E(\mathcal{P}^*) \cup E(\mathcal{P}')| \leq |E(\mathcal{P}^*)| + |E(\mathcal{P}')| \leq k$. However, $E(\mathcal{P}^*) \cup E(\mathcal{P}')$ is a co-connected $T$-join of $G$ and thus by our assumption, $E(\mathcal{P}^*) \cup E(\mathcal{P}')$ has size exactly $k$ — thus a minimum sized solution. By Lemma 3.3 this implies that $E(\mathcal{P}^*) \cup E(\mathcal{P}')$ is a forest and hence $P_r^* P_r'$ is a path in $G$. This completes the proof.

For our proofs we also need the transitivity property of the relation $\sqsubseteq_{peq}$.

**Lemma 3.9 (4).** The relation $\sqsubseteq_{peq}$ is transitive.

Our algorithm is based on dynamic programming and stores a table $D[i, T', v]$ for all $i \in \{0, \ldots, k\}, T' \subseteq T$ and $v \in V(G) \cup \{\epsilon\}$. The idea is that $D[i, T', v]$ will store a path-system equivalent set to $Q[i, T', v]$. That is, $D[i, T', v] \sqsubseteq_{peq} Q[i, T', v]$. The recurrences for dynamic programming is given by the following.

For $i = 0$, we have the following cases:

$$ D[0, T', v] := \begin{cases} \{\emptyset\} & \text{if } T' = \emptyset \text{ and } v = \epsilon \\ \emptyset & \text{otherwise} \end{cases} \quad (1) $$

For $i \geq 1$, we have the following cases based on whether $v = \epsilon$ or not.

$$ D[i, T', v] := \left( \bigcup_{t \in T'} D[i - 1, T' \setminus \{t\}, \epsilon] \bullet \{(t, v)\} \right) \cup \left( \bigcup_{(u, v) \in E(G)} D[i - 1, T', u] \bullet \{(u, v)\} \right) \quad (2) $$

$$ D[i, T', \epsilon] := \left( \bigcup_{t_1, t_2 \in T'} D[i - 1, T' \setminus \{t_1, t_2\}, \epsilon] \bullet \{(t_1, t_2)\} \right) \cup \left( \bigcup_{(u, t) \in E(G)} D[i - 1, T' \setminus \{t\}, u] \bullet \{(u, t)\} \right) \quad (3) $$

The next lemma will be used in proving the correctness of the algorithm.

**Lemma 3.10.** For all $i \in \{0, \ldots, k\}, T' \subseteq T, v \in V(G) \cup \{\epsilon\}, D[i, T', v] \sqsubseteq_{peq}^{k-i} Q[i, T', v]$.

**Proof.** Let $\mathcal{I}$ denote the family of independent sets in $M_G$, the co-graphic matroid associated with $G$. We prove the lemma using induction on $i$. The base case is $i = 0$. From the definition of $Q[0, T', v]$, we have that $Q[0, T', v] = \{\emptyset\}$ if $T' = \emptyset$ and $v = \epsilon$, and $Q[0, T', v] = \emptyset$ otherwise.

Now we prove that the claim holds for $i \geq 1$. Let us also assume that by induction hypothesis the claim is true for all $i' < i$. Fix a $T' \subseteq T$, and $v \in V(G) \cup \{\epsilon\}$ and let $Q[i, T', v]$ be the corresponding set of partial solutions. Let $P = \{P_1, \ldots, P_r\} \in Q[i, T', v]$ and $P' = \{P_r', P_{r+1}, \ldots, P_k\}$ be a path system such that $P'$ is an extender for $P$. We need to show that there exists a $\mathcal{P}^* \in D[i, T', v]$ such that $\mathcal{P}'$ is also an extender for $\mathcal{P}^*$. 


Case 1: $v \neq e$. Consider the path system $P = \{P_1, \ldots, P_r\} \in Q[i, T', v]$. $P$ has $i$ edges and its set of end-vertices is $T' \cup \{v\}$. Also, its final vertex is $v$. Let $(u, v)$ be the last edge in path $P_r$. Let $P''_r$ be the path obtained by deleting edge $(u, v)$ from $P_r$. More precisely: If $P_r$ has at least two edges then $P''_r$ is the non-empty path obtained by deleting the edge $(u, v)$ and the vertex $v$ from $P_r$, and if $(u, v)$ is the only edge in $P_r$ (in which case $u \in T'$) then $P''_r = \emptyset$. Note that the initial vertex of $P''_r \in P'$ is $v$. Let $uP''_r$ be the path obtained by concatenating the path $uv$ and $P''_r$. Let $P_1 = \{P_1, \ldots, P''_r\}$ and $P'_1 = \{uP''_r, P'_{r+1}, \ldots, P_r\}$. Then $P_1$ has $(i-1)$ edges and $P'_1$ is an extender for $P_1$. Now we consider two cases:

(u, v) is the only edge in $P_r$: Here $P''_r = \emptyset$ and $u \in T'$; let $t = u$. Note that $P_1 = \{P_1, \ldots, P_{r-1}\} \in Q[i-1, T' \setminus \{t\}, e]$. Hence by induction hypothesis there exists $P'_1 \in D[i-1, T' \setminus \{t\}, e]$ such that $P'_1$ is also an extender for $P'_1$. Since $P'_1$ is an extender for $P'_1$, $E(P'_1) \cup E(P'_1) \in I$ (by the definition of extender). This implies that $E(P'_1) \cup \{(t, v)\} \in I$. Since $P'_1 \in D[i-1, T' \setminus \{t\}, e]$, and $(t, v) \in E(G)$, by Equation 2, we get a path system $P^r \in D[i, T', v]$ by adding the new path $P_r = tv$ to $P'_1$. Since $P'_1$ is an extender of $P'_1$, $P^r$ is an extender of $P^r$ as well.

(u, v) is not the only edge in $P_r$: Here $P''_r \neq \emptyset$, and $u$ is the final vertex in $P''_r$. Hence $P_1 = \{P_1, \ldots, P''_r\} \in Q[i-1, T', u]$. Since $P'_1$ is an extender for $P_1$, by induction hypothesis there exists $P'_1 \in D[i-1, T', u]$ such that $P'_1$ is also an extender for $P'_1$. By the definition of extender, we have that $E(P'_1) \cup E(P'_1) \in I$. This implies that $E(P'_1) \cup \{(u, v)\} \in I$. Since $P'_1 \in D[i-1, T', u]$, and $(u, v) \in E(G)$, by Equation 2, we get a path system $P^r \in D[i, T', v]$ by adding the new edge $(u, v)$ to $P'_1$. Since $P'_1$ is an extender of $P'_1$, $P^r$ is an extender of $P^r$ as well.

Case 2: $v = e$. We have that $P = \{P_1, \ldots, P_r\} \in D[i, T', e]$. Then $P$ has $i$ edges, its set of end-vertices is $T'$, and no end-vertex repeats. Let $(u, t)$ be the last edge in path $P_r$. Then $t \in T'$. Let $P''_r$ be the path obtained by deleting edge $(u, t)$ from $P_r$. More precisely: If $P_r$ has at least two edges then $P''_r$ is the non-empty path obtained by deleting the edge $(u, t)$ and the vertex $t$ from $P_r$, and if $(u, t)$ is the only edge in $P_r$ then $P''_r = \emptyset$. Let $P_1 = \{P_1, \ldots, P''_r\}$ and $P'_1 = \{u, P''_r, P'_{r+1}, \ldots, P_r\}$. Then $P_1$ has $(i-1)$ edges and $P'_1$ is an extender for $P_1$. Now we consider two cases:

(u, t) is the only edge in $P_r$: Here $P''_r = \emptyset$, and $\{u, t\} \subseteq T'$. Let $t_1 = u, t_2 = t$. Then $P_1$ is a path system in $Q[i-1, T' \setminus \{t_1, t_2\}, e]$. By induction hypothesis there exists $P'_1 \in D[i-1, T' \setminus \{t_1, t_2\}, e]$ such that $P'_1$ is also an extender for $P'_1$. By the definition of extender, we have that $E(P'_1) \cup E(P'_1) \in I$. This implies that $E(P'_1) \cup \{(t_1, t_2)\} \in I$. Since $P'_1 \in D[i-1, T' \setminus \{t_1, t_2\}, e]$ and $(t_1, t_2) \in E(G)$, by Equation 3, we get a path system $P^r \in D[i, T', v]$ by adding the new path $t_1t_2$ to $P'_1$. Since $P'_1$ is an extender of $P'_1$, $P^r$ is an extender of $P^r$ as well.

(u, t) is not the only edge in $P_r$: Here $P''_r \neq \emptyset$, $u$ is the final vertex in $P''_r$. Then $P_1 \in Q[i-1, (T' \setminus \{t\}), u]$. By induction hypothesis there exists $P'_1 \in D[i-1, (T' \setminus \{t\}), u]$ such that $P'_1$ is also an extender for $P'_1$. By the definition of extender, we have that $E(P'_1) \cup E(P'_1) \in I$. This implies that $E(P'_1) \cup \{(u, t)\} \in I$. Since $P'_1 \in D[i-1, (T' \setminus \{t\}), u]$ and $(u, t) \in E(G)$, by Equation 3, we get a path system $P^r \in D[i, T', \epsilon]$ by adding the new edge $(u, t)$ to $P'_1$. Since $P'_1$ is an extender of $P'_1$, $P^r$ is an extender of $P^r$ as well.

In both cases above we showed that $D[i, T', v] \cup_{p=1}^{i-1} Q[i, T', v]$.\[\]
**Algorithm, Correctness and Running Time.** We now describe the main steps of the algorithm. It finds a smallest sized co-connected $T$-join (of size at most $k$) for $G$. The algorithm iteratively tries to find a solution of size $\frac{m}{n} \leq k' \leq k$ and returns a solution corresponding to the smallest $k'$ for which it succeeds; else it returns No. By Lemma 3.8 it is enough, in the dynamic programming (DP) table, to store the representative set $Q[i, T', v] \subseteq_{\text{peq}} Q[i, T', v]$ instead of the complete set $Q[i, T', v]$, for all $i \in \{1, 2, \ldots, k\}$, $T' \subseteq T$ and $v \in (V(G) \cup \{e\})$. In the algorithm we compute and store the set $Q[i, T', v]$ in the DP table entry $D[i, T', v]$. We follow Equations 1, 2 and 3 and fill the table $D[i, T', v]$. For $i = 0$ we use Equation 1 and fill the table. After this we compute the values of $D[i, T', v]$ in increasing order of $i$ from 1 to $k$. At the $i^{th}$ iteration of the for loop, we compute $D[i, T', v]$ from the DP table entries computed at the previous iteration. Since we need to keep the size of potential partial solutions in check, we compute the representative family $D[i, T', v]$ for each DP table entry $D[i, T', v]$ constructed in the $i^{th}$ iteration and then set $D[i, T', v] \leftarrow D[i, T', v]$. By the definition of $Q[i, T, e]$ and Lemma 3.4, any path system in $D[i, T, e]$ is a solution to the instance $(G, T, k)$; we check for such a solution as the last step. This completes the description of the algorithm.

The correctness of the algorithm follows from the following. By Lemma 3.10 we know that $D[i, T', v] \subseteq_{\text{peq}} Q[i, T', v]$ and by Lemma 3.8 we have that $D[i, T', v] \subseteq_{\text{peq}} D[i, T', v]$. Thus, by transitivity of $\subseteq_{\text{peq}}$ (by Lemma 3.9) we have that $D[i, T', v] \subseteq_{\text{peq}} Q[i, T', v]$. This completes the proof of correctness. We now compute an upper bound on the running time of the algorithm.

**Lemma 3.11.** The above algorithm runs in time $O(2^{2+\omega} \cdot n^2 m^3 k^5) + m^{O(1)}$ where $n = |V(G)|$ and $m = |E(G)|$.

**Proof.** Let $1 \leq i \leq k$ and $T' \subseteq T$ and $v \in (V(G) \cup \{e\})$ be fixed, and let us consider the running time of computing $D[i, T', v]$. That is, the running time to compute $(k-i)$-representative family of $D[i, T', v]$. We know that the co-graphic matroid $M_G$ is representable over $\mathbb{F}_2$ and that its rank is bounded by $m - n + 1$. By Theorem 2.2, the running time of this computation of the $(k-i)$-representative family is bounded by $O\left(\binom{k}{i} \cdot |D[i, T', v]|^2 n^2 + |D[i, T', v]| \cdot \binom{k}{i} \omega^{-1} (i \cdot m)^{\omega-1}\right) + m^{O(1)}$.

The family $D[i, T', v]$ is computed using Equation 2 or Equation 3 from the DP table entries $D[i-1,T'', u]$, computed in the previous iteration and the size of $D[i-1,T'', u]$ is bounded according to Theorem 2.2. Thus the size of the family $D[i, T', v]$ is upper bounded by, $|D[i, T', v]| \leq ((2k)^2 + 2kn) \cdot \left(\max_{T'' \subseteq T', u \in V} D[i-1, T'', u]\right)$. Theorem 2.2 gives bounds on the sizes of these representative families $D[i-1,T'', u]$, from which we get $|D[i, T', v]| \leq 4kn \cdot m \binom{k}{i-1}$. Observe that since the number choices for $(T', v)$ such that $T' \subseteq T$ and $v \in V(G)(e)$ is bounded by $4^k(n + 1)$, and we compute DP table entries for $i = 1$ to $k$, the overall running time can be bounded by $O\left(4^k n \sum_{i=1}^{k} \binom{k}{i} \cdot m \cdot (i \cdot m)^{\omega-1} k n i m^{\omega}\right) + m^{O(1)}$. The running time above simplifies to $O(2^{2+\omega} k \cdot n^2 m^3 k^5) + m^{O(1)}$.

Putting all these together we get

**Theorem 3.12.** CO-CONNECTED $T$-JOIN can be solved in $O(2^{2+\omega} k \cdot n^2 m^3 k^5) + m^{O(1)}$ time where $n = |V(G)|$ and $m = |E(G)|$.

Using Theorem 3.1 and Theorem 3.12 we get
Theorem 3.13. Undirected Eulerian Edge Deletion can be solved in time $O(2^{(2+\omega)k}n^2m^3k^6 + m^{O(1)})$ where $n = |V(G)|$ and $m = |E(G)|$.

4 Directed Eulerian Edge Deletion

In this section we modify the algorithm described for Undirected Eulerian Edge Deletion to solve the directed version of the problem. The main ingredient of the proof is the characterization of “solution” for the directed version of the problem. We begin with a few definitions. For a digraph $D$, we call $S \subseteq A(D)$ a balanced arc deletion set, if $D \setminus S$ is balanced. We call a set $S \subseteq A(D)$ a co-connected balanced arc deletion set if $D \setminus S$ is balanced and weakly connected.

Let $(D, k)$ be an instance to Directed Eulerian Edge Deletion. A solution $S \subseteq A(D)$ of the problem should satisfy the following two properties, (a) $S$ must be a balanced arc deletion set of $D$ and, (b) $D \setminus S$ must be strongly connected. In fact, something stronger is known in the literature.

Proposition 4.1 ([1]). A digraph $D$ is Eulerian if and only if $D$ is weakly connected and balanced.

Due to Proposition 4.1, we can relax the property (b) of the solution $S$ and replace the requirement of having $D \setminus S$ as strongly connected with just requiring $D \setminus S$ to be weakly connected. Now observe that solution $S$ of Directed Eulerian Edge Deletion is in fact a co-connected balanced arc deletion set of the directed graph $D$. Thus our goal is to compute a minimal co-connected balanced arc deletion set of $D$ of size at most $k$.

We start with the following easy property of in-degrees and out-degrees of vertices in $D$. For a digraph $D$, define $T^- = \{v \in V(D) \mid d_D^-(v) > d_D^+(v)\}$, $T^+ = \{v \in V(D) \mid d_D^-(v) = d_D^+(v)\}$ and $T^+ = \{v \in V(D) \mid d_D^-(v) < d_D^+(v)\}$.

Proposition 4.2. In a digraph $D$, $\sum_{v \in T^+} d_D^+(v) - d_D^-(v) = \sum_{v \in T^-} d_D^+(v) - d_D^-(v)$.

The following lemma characterizes the set of arcs which form a minimal solution $S$ of the given instance $(D, k)$. We then use this characterization to design a dynamic-programming algorithm for the problem.

Lemma 4.3. Let $D$ be a digraph, and $\ell = \sum_{v \in T^+} d_D^+(v) - d_D^-(v)$. Let $S \subseteq A(D)$ be a minimal co-connected balanced arc deletion set. Then $S$ is a union of $\ell$ arc disjoint paths $P = \{P_1, \ldots, P_\ell\}$ such that

1. For $i \in \{1, \ldots, \ell\}$, $P_i$ starts at a vertex in $T^+$ and ends at a vertex in $T^-$.
2. The number of paths in $P$ that starts at $v \in T^+$ is equal to $d_D^+(v) - d_D^-(v)$ and the number of paths in $P$ that ends at $u \in T^-$ is equal to $d_D^-(u) - d_D^+(u)$.

Finally, we prove a kind of “converse” of Lemma 4.3.

Lemma 4.4. Let $D$ be a digraph, $\ell = \sum_{v \in T^+} d_D^+(v) - d_D^-(v)$ and let $S \subseteq A(D)$. Furthermore, $S$ is a union of $\ell$ arc disjoint paths $P = \{P_1, \ldots, P_\ell\}$ with the following properties.

1. The digraph $D \setminus S$ is weakly connected.
2. For $i \in \{1, \ldots, \ell\}$, $P_i$ starts at a vertex in $T^+$ and ends at a vertex in $T^-$.
3. The number of paths in $P$ that starts at $v \in T^+$ is equal to $d_D^+(v) - d_D^-(v)$ and the number of paths in $P$ that ends at $u \in T^-$ is equal to $d_D^-(u) - d_D^+(u)$.

Then $S$ is a co-connected balanced arc deletion set.
Now we are ready to describe the algorithm for Directed Eulerian Edge Deletion. Let \((D, k)\) be an instance of the problem. Lemma 4.3 and Lemma 4.4 imply that for a solution we can seek a path system \(P\) with properties mentioned in Lemma 4.4. Let \(T^+_m\) be the multiset of vertices in the graph \(G\) such that each vertex \(v \in T^+\) appears \(d_D^+(v) - d_D^-(v)\) times in \(T^+_m\). Similarly, let \(T^-_m\) be the multiset of vertices in the graph \(D\) such that each vertex \(v \in T^-\) appears \(d_D^+(v) - d_D^-(v)\) times in \(T^-_m\). Due to Proposition 4.2 we know that \(|T^+_m| = |T^-_m|\). Observe that if \(|T^+_m| > k\), then any balanced arc deletion set must contain more than \(k\) arcs and thus the given instance is a No instance. So we assume that \(|T^+_m| \leq k\).

Lemma 4.3 implies that the solution can be thought of as a path system \(P = \{P_1, \ldots, P_r\}\) connecting vertices from \(T^+_m\) to the vertices of \(T^-_m\) such that all the vertices of \(T^+_m \cup T^-_m\) appear as end points exactly once and \(D \setminus A(P)\) is weakly connected. Observe that the solution is a path system with properties which are similar to those in the undirected case of the problem. Indeed, the solution \(S\) corresponds to an independent set in the co-graphic matroid of the underlying (undirected) graph of \(D\). After this the algorithm for Directed Eulerian Edge Deletion is identical to the algorithm for Co-Connected \(T\)-Join. Let \(T = T^+_m \cup T^-_m\). We can define a notion of partial solutions analogous to \(Q[i, T', v]\). The definition of extender remains the same except for the last item, where we now require that \(P \cup P'\) is an arc disjoint path system connecting vertices from \(T^+_m\) to the vertices of \(T^-_m\) such that every vertex of \(T^+_m \cup T^-_m\) is an endpoint of exactly one path. Finally, we can define the recurrences for dynamic programming similar to those defined for \(D[i, T', v]\) in the case of Co-Connected \(T\)-Join. We then use these recurrences along with an algorithm to compute representative families to solve the given instance. The correctness of the algorithm follows via similar arguments as before. And by an analysis similar to the case of Co-Connected \(T\)-Join we can obtain the following bound on the running time of the algorithm.

\[O\left(2^{(2+\omega)k} \cdot n^2 m^3 k^6\right) + m^{O(1)} \text{ where } n = |V(D)| \text{ and } m = |A(D)|.\]

References


