Parameterized Complexity of Eulerian Deletion Problems

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

Institute of Informatics, University of Warsaw, Poland

Institut für Informatik, Humboldt-Universität zu Berlin, Germany

Department of Computer Science and Information Theory,
Budapest University of Technology and Economics, Hungary

21th June 2011,
WG 2011, Teplá

The subject

The subject

We study tractability of a family of problems where the goal is to make a (directed) graph Eulerian by a minimum number of edge (vertex) deletions.

Definitions

An undirected graph is:

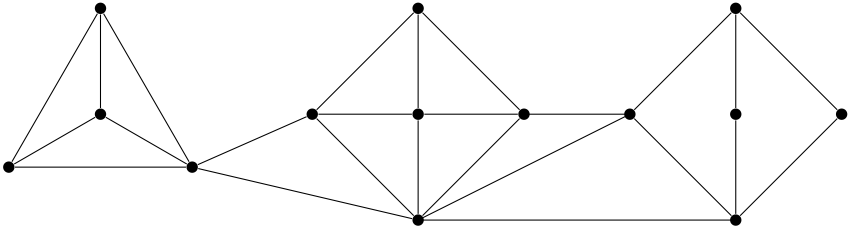
- even iff $\forall_v \deg(v)$ is even,
- Eulerian iff even and connected,

A directed graph is:

- balanced iff $\forall_v \deg^{\text{in}}(v) = \deg^{\text{out}}(v)$,
- Eulerian iff strongly connected and balanced.

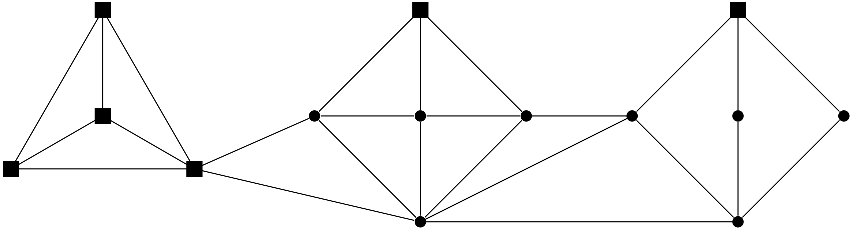
Example

Can we make the graph Eulerian by removing at most 5 edges?



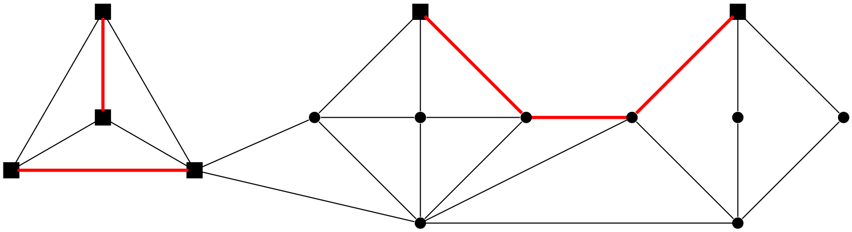
Example

Can we make the graph Eulerian by removing at most 5 edges?



Example

Can we make the graph Eulerian by removing at most 5 edges?



Yes, we can.

NP-hardness

Observation

A cubic graph can be made Eulerian by removing $n/2$ edges iff it has a Hamiltonian Cycle.

Hence the problem of removing the minimum number of edges to make an undirected graph Eulerian is NP-hard.

Parameterized complexity

- Not all NP-complete problems are the same.
- Fixed Parameter Tractability tries to distinguish between NP-complete problems.
- Each instance comes with a parameter k – the hardness measure.
- A problem is Fixed Parameter Tractable (FPT) if it admits an algorithm of $f(k)|G|^{O(1)}$ time complexity.
- A problem has a kernel of size $f(k)$ if it is possible to reduce the instance size to $f(k)$ in polynomial time.

Previous results

Eulerian deletion problems study initiated by Cai and Yang.

	Undirected Eulerian	Directed Eulerian
Vertex deletion:	W[1]-hard Cai and Yang	?
Edge deletion	?	?

Our results

	Undirected Eulerian	Directed Eulerian
Vertex deletion:	W[1]-hard Cai and Yang	W[1]-hard this paper
Edge	FPT, no poly kernel this paper	FPT, no poly kernel this paper

Outline

During the talk we consider edge deletion versions only.

1. Making an undirected graph even – known facts.
2. Connectivity witness notion.
3. Making an undirected graph Eulerian – FPT algorithm.
4. Making a directed graph Eulerian – sketch of an FPT algorithm.
5. Open problems.

Making a graph even

Problem definition

Can we make an undirected graph $G = (V, E)$ even by removing at most k edges?

- Let V_{odd} be the set of vertices of odd degree.
- If $|V_{\mathrm{odd}}| > 2k$ then NO.
- If for $S \subseteq E$ the graph $G \setminus S$ is even then v is odd in G iff it is odd in $G \setminus S$.
- It is possible to partition S into $|V_{\mathrm{odd}}|/2$ edge disjoint paths between vertices of V_{odd} in G.

Making a graph even

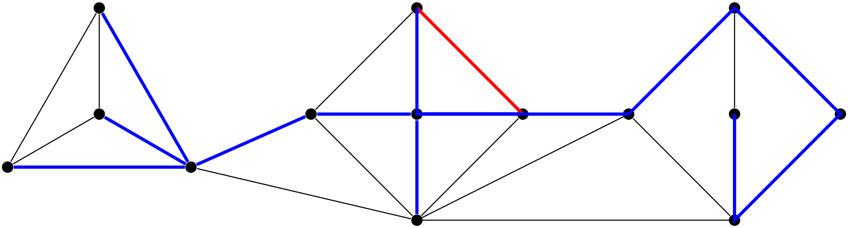
Two consequences:

- problem is poly time solvable by a reduction to min weight perfect matching (V_{odd} with distance in G as a weight function),
- S contains no edges at distance more than k from V_{odd}.

We will use both these observations in our FPT algorithm.

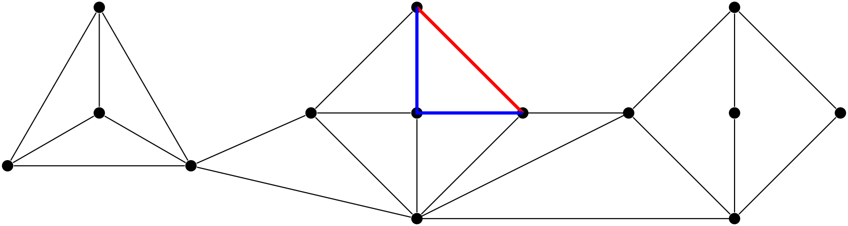
Witness of connectivity

- Assume that $G = (V, E)$ is connected.
- How can we prove that $G \setminus e$ is connected?
- We can show a spanning tree of $(V, E \setminus \{e\})$ – correct but too big.



Witness of connectivity

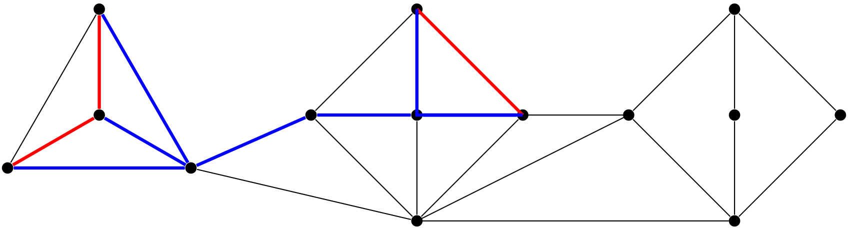
- Assume that $G = (V, E)$ is connected.
- How can we prove that $G \setminus e$ is connected?
- We can show a path in G disjoint with e.



Witness of connectivity

Definition

For a set of edges $S \subseteq E$ we call $W \subseteq (E \setminus S)$ a witness of connectivity of $G \setminus S$ iff W spans all vertices of $V(S)$.



Problem definition

Making an undirected graph Eulerian

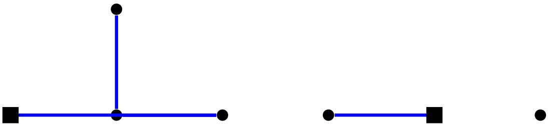
For a given connected graph G we want to find a pair of disjoint sets of edges (S, W) such that:

- $|S| \leqslant k$,
- $G \setminus S$ is even,
- W is a witness of connectivity of $G \setminus S$.

We want to solve this problem in FPT time with k as a parameter.

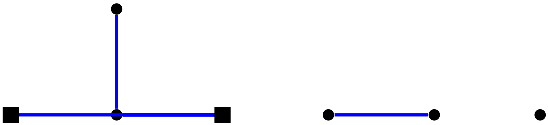
FPT algorithm

- Assume that we are given a partition $E = E_s \cup E_w$, $E_s \cap E_w = \emptyset$.
- We want to check whether there exists a solution (S, W), such that $S \subseteq E_s$ and $W \subseteq E_w$.
- If two vertices from V_{odd} are in different connected components of (V, E_w) – return NO.



FPT algorithm

- Assume that we are given a partition $E = E_s \cup E_w$, $E_s \cap E_w = \emptyset$.
- We want to check whether there exists a solution (S, W), such that $S \subseteq E_s$ and $W \subseteq E_w$.
- If two vertices from V_{odd} are in different connected components of (V, E_w) – return NO.
- Otherwise take the connected component of (V, E_w) containing V_{odd} and solve the problem in polynomial time in the subgraph of (V, E_s) induced by this component.

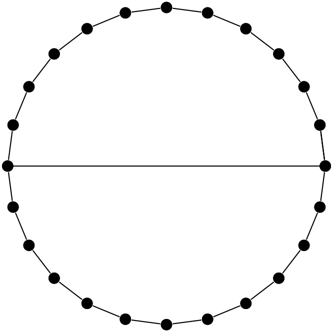


FPT algorithm

- But how can we get a good partition?
- We can guess it.
- If there exists a solution (S, W) where $|W| \leqslant f(k)$ then a random partition will be good with probability at least $2^{-(k+f(k))}$.

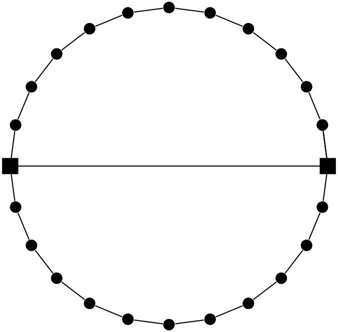
FPT algorithm

Unfortunately sometimes small witness does not exists.



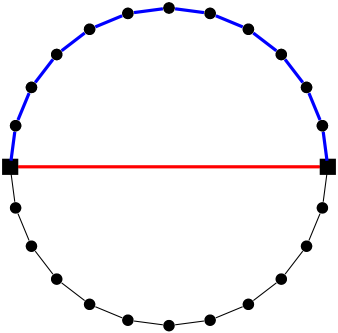
FPT algorithm

Unfortunately sometimes small witness does not exists.



FPT algorithm

Unfortunately sometimes small witness does not exists.



FPT algorithm

Recall that edges far from V_{odd} are never used in a solution, so we can make them always blue!

Lemma

If G is a YES instance than there exists a witness W containing $O(k^2)$ edges at distance at most k from V_{odd}.

FPT algorithm

1. Determine which edges are close (of distance at most k from V_{odd}) and which are far.
2. Make each close edge independently with probability $1/k^2$ red ($\in E_s$); every edge that is not red becomes blue ($\in E_w$).
3. If there is more than one connected component of the blue edges containing a vertex from V_{odd}, return NO; otherwise let K_B be this unique component.
4. Solve the 'making a graph even' problem in the subgraph of (V, E_s) induced by K_B.

FPT algorithm

- By the previous lemma if there exists a solution then there exists (S, W) with $|S| \leqslant k$ and $|W \cap \mathrm{close}(E)| = O(k^2)$.
- With probability at least $(1/k^2)^k = 1/2^{2k \log k}$ each edge of S becomes red.
- 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).
- At least $1/2^{O(k \log k)}$ success probability.
- We can derandomize the above algorithm using the standard technique of splitters.

Directed case

- A directed graph Eulerian iff strongly connected and balanced.
- Equivalently: **weakly connected** and balanced.
- Use the notion of connectivity witness for weak connectivity.
- Solve the 'making a graph balanced' problem by a reduction to min cost max flow.

Open problems

1. FPT algorithm for the edge-deletions problems running in $c^k n^{O(1)}$ time?
2. Cechlárová and Schlotter [IPEC'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.

Thank you for your attention

Questions?