Journal of Discrete Algorithms 9 (2011) 231–240

Contents lists available at ScienceDirect

Journal of Discrete Algorithms

www.elsevier.com/locate/jda

Parameterized complexity of even/odd subgraph problems

Leizhen Cai [a,*,1], Boting Yang [b,2]

[a] Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong SAR, China
[b] Department of Computer Science, University of Regina, Regina, Saskatchewan, Canada

ARTICLE INFO

Article history:
Available online 11 March 2011

Keywords:
Parameterized complexity
FPT algorithms
Eulerian graphs
Even graphs
Odd graphs
Color-coding
Random separation

ABSTRACT

We study the parameterized complexity of the problems of determining whether a graph contains a k-edge subgraph (k-vertex induced subgraph) that is a Π-graph for Π-graphs being one of the following four classes of graphs: Eulerian graphs, even graphs, odd graphs, and connected odd graphs. We also consider the parameterized complexity of their parametric dual problems.

For these sixteen problems, we show that eight of them are fixed parameter tractable and four are W[1]-hard. Our main techniques are the color-coding method of Alon, Yuster and Zwick, and the random separation method of Cai, Chan and Chan.

© 2011 Elsevier B.V. All rights reserved.

1. Introduction

The initial motivation of this paper comes from Eulerian graphs, which are often regarded as the cradle of graph theory. How can we efficiently find k edges in a graph to form an Eulerian graph? Unfortunately, like most subgraph problems, the problem is NP-hard. This in turn leads us to study the problem using the framework of parameterized complexity by Downey and Fellows [7]: regard some part of the input I of a problem as a parameter k to form a *parameterized problem* and determine whether the problem can be solved by an *FPT algorithm*, i.e., an algorithm that runs in time $f(k)|I|^{O(1)}$ for some computable function $f(k)$. A parameterized problem is *fixed-parameter tractable* (FPT) if it admits an FPT algorithm.

Subgraph problems (a.k.a. edge-deletion and vertex-deletion problems) have been studied extensively in the literature under the framework of traditional complexity. Among others, a classical result of Lewis and Yannakakis [11] states that for any "interesting" hereditary property[3] Π, it is NP-hard to decide whether a graph G contains at least k vertices that induce a Π-graph, and a useful article of Natanzon, Shamir and Sharan [13] gives complexity classification of a large number of edge-deletion problems.

Parameterized subgraph problems have also been considered for various properties Π in the literature, and here we only mention two general results. Cai [4] has shown that it is FPT to obtain a Π-graph by k vertex/edge additions and deletions whenever Π is a hereditary property that can be characterized by a finite set of forbidden induced subgraphs. Khot and Raman [10] have proved that, for any decidable hereditary property Π, the problem of finding k vertices to induce a Π-graph is FPT if Π includes all trivial graphs and all complete graphs or excludes some trivial graphs and some complete graphs, and W[1]-hard if Π includes all trivial graphs but not all complete graphs or vice versa.

* Corresponding author.
E-mail addresses: lcai@cse.cuhk.edu.hk (L. Cai), boting@cs.uregina.ca (B. Yang).

1 Partially supported by Earmarked Research Grant 410206 of the Research Grants Council of Hong Kong.
2 Partially supported by NSERC.

3 A graph property Π is a collection of graphs, and it is *hereditary* if every induced subgraph of a Π-graph is also a Π-graph.

1570-8667/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.jda.2011.03.004

Table 1
The parameterized complexity of the 16 parameterized subgraph problems. For instance, the k-EDGE EVEN SUBGRAPH problem is FPT as indicated by the entry at row "k-EDGE" and column "EVEN".

	EULERIAN	EVEN	ODD	CONNECTED ODD
k-EDGE	FPT	FPT	FPT	?
$(m-k)$-EDGE	?	FPT	FPT	?
k-VERTEX	?	FPT	FPT	FPT
$(n-k)$-VERTEX	W[1]-hard	W[1]-hard	W[1]-hard	W[1]-hard

In this paper, we study parameterized complexity of subgraph problems motivated by Eulerian graphs. An *even graph* (respectively, *odd graph*) is a graph where each vertex has an even (odd) degree, and an *Eulerian graph* is a connected even graph. Let Π be one of the following four graph classes: *Eulerian graphs, even graphs, odd graphs*, and *connected odd graphs*. For each Π, we define the following subgraph and induced subgraph problems:

Π SUBGRAPH (resp. INDUCED Π SUBGRAPH)
Instance: $G = (V, E)$ and nonnegative integer k.
Question: Does G contain a Π subgraph with k edges (resp. an induced Π subgraph on k vertices)?

We can take k in the above two problems as a parameter to form two parameterized subgraph problems, which also yield their corresponding *parametric dual problems*, where m and n are the numbers of edges and vertices, respectively, in the input graph G:

k-EDGE Π SUBGRAPH (resp. k-VERTEX Π SUBGRAPH)
Instance: $G = (V, E)$ and nonnegative integer k.
Question: Does G contain a Π subgraph with k edges (resp. an induced Π subgraph on k vertices)?
Parameter: k.

$(m-k)$-EDGE Π SUBGRAPH (resp. $(n-k)$-VERTEX Π SUBGRAPH)
Instance: $G = (V, E)$ and nonnegative integer k.
Question: Does G contain a Π subgraph with $m-k$ edges (resp. an induced Π subgraph on $n-k$ vertices)?
Parameter: k.

Our main concern is to determine whether these 16 problems are fixed-parameter tractable. Note that the properties Π we consider here are not hereditary, and therefore the theorems of Cai, and Khot and Raman are not applicable to our problems. Table 1 summarizes our results for the parameterized complexity of the 16 problems we study in the paper: 8 problems are FPT, 4 problems are W[1]-hard and the remaining 4 problems are open. The main techniques for our FPT algorithms are the color-coding method of Alon, Yuster and Zwick [1], and the random separation method of Cai, Chan and Chan [5]. We will also use kernelization and Ramsey numbers in designing our FPT algorithms.

We note that for the eight unparameterized problems corresponding to our parameterized subgraph problems, six of them are NP-complete (Section 4) and the other two are conjectured to be also NP-complete (Section 5).

All graphs in this paper are undirected simple graphs. We use m and n, respectively, for the numbers of edges and vertices of the input graph $G = (V, E)$. For a subset $V' \subseteq V$, $G[V']$ denotes the subgraph induced by V', and for a subset $E' \subseteq E$, $G[E']$ denotes the subgraph formed by E'. For a vertex v, $d_G(v)$ denotes the degree of v, and $N_G(v)$ denotes the neighborhood of v. For a subset $V' \subset V$, we use $N_G(V')$ to denote the open neighborhood $(\bigcup_{v \in V'} N_G(v)) \setminus V'$ of V'. We will drop subscript G in our notation when it will not cause confusion.

The rest of the paper is organized as follows. We present our FPT algorithms in two sections based on the techniques we use: we use color-coding to obtain FPT algorithms for 4 problems in Section 2, and use kernelization and random separation, together with Ramsey numbers, to solve 4 more problems in Section 3. We give W[1]-hardness proofs for another 4 problems in Section 4, where we also establish the NP-completeness of 6 unparameterized problems. We discuss some open problems and propose some conjectures in Section 5.

2. FPT by color-coding

In this section, we use the color-coding method of Alon, Yuster and Zwick [1] to obtain FPT algorithms for the following four problems:

- k-EDGE EULERIAN SUBGRAPH
- k-EDGE EVEN SUBGRAPH
- $(m-k)$-EDGE EVEN SUBGRAPH
- $(m-k)$-EDGE ODD SUBGRAPH.

The key idea of color-coding is to randomly color elements with k colors, and then find a k-element solution with distinct colors in FPT time. Such a randomized algorithm finds a solution, if it exists, with probability at least $k!/k^k$, and can be converted into a deterministic FPT algorithm by using perfect hash functions. An (m,k)-*family of perfect hash functions* is a family $\mathcal{F}$ of functions from a domain of size m to a range of size k such that for every subset S of size k from the domain there is a function in $\mathcal{F}$ that is 1-to-1 on S. Moni Naor (see [1]) has used the construction of Schmidt and Siegal [14] to obtain an (m,k)-family of perfect hash functions of size $2^{O(k)}\log m$ that can be listed in time $2^{O(k)}m\log m$.

To use color-coding to solve our problems, we color each edge randomly by a color from $\{1,\dots,k\}$ and then try to find a k-edge subgraph with certain properties. Before getting into our algorithms, we define a few terms for an edge-colored graph G. A subgraph H of G is *colorful* if its edges have distinct colors. For a colorful H, we may say that it is C'-*colorful* when we want to emphasize the colors C' used for edges of H. For a subset C' of colors, we use $G(C')$ to denote the subgraph of G formed by edges with colors from C'.

We also need connections between trails, Eulerian graphs and even graphs. For a pair (u,v) of vertices, a (u,v)-*trail* is an alternating sequence of vertices and edges $u=v_0,e_1,v_1,\dots,e_t,v_t=v$ such that each edge e_i has v_{i-1} and v_i as endpoints and all edges are distinct. The *length* of a trail is the number of edges in the trail, and a (u,v)-trail is *closed* if $u=v$. An Eulerian graph has a closed trail containing all edges and vertices, and every component of an even subgraph is an Eulerian graph. Therefore trails play a pivotal role in this section, and central to our FPT algorithms is the following result for finding a colorful (u,v)-trail between a pair $\{u,v\}$ of vertices in an edge-colored graph in FPT time.

Lemma 2.1. *Let G be an edge-k-colored graph. It takes $O(k2^km)$ time to find, for any given pair $\{u,v\}$ of vertices, a colorful (u,v)-trail of length k (if it exists), and $O(k2^kmn)$ time to find a colorful k-edge Eulerian subgraph (if it exists).*

Proof. Let $c: E\to\{1,\dots,k\}$ be the edge coloring of G. For a trail T, let $\alpha(T)$ denote the set of colors used for edges of T. For each vertex x of G and each $1\leqslant i\leqslant k$, define

$$C^i(x)=\bigl\{\alpha(T):\ T \text{ is a colorful } (u,x)\text{-trail with } i \text{ edges}\bigr\}.$$

Then G has a colorful (u,v)-trail of length k iff $C^k(v)\neq\emptyset$.

We can easily obtain the following recurrence for $C^i(x)$:

$$C^1(x)=\begin{cases}\{\{c(ux)\}\} & \text{if } x\in N(u),\\ \emptyset & \text{otherwise}\end{cases}$$

and for each $2\leqslant i\leqslant k$,

$$C^i(x)=\bigl\{\bigl\{c(xy)\bigr\}\cup C:\ y\in N(x),\ C\in C^{i-1}(y),\ \text{and } c(xy)\notin C\bigr\}.$$

It is straightforward to compute $C^k(v)$ by dynamic programming, which takes $O(\sum_{i=1}^k i\binom{k}{i}m)=O(k2^km)$ time, and a simple book keeping will enable us to construct a colorful (u,v)-trail with k edges (if it exists) in $O(k2^km)$ time. To find a colorful k-edge Eulerian subgraph, we consider a colorful (v,v)-trail of length k for each vertex v, which altogether takes $O(k2^kmn)$ time. □

With the above lemma in hand, we now proceed to use color-coding to design FPT algorithms.

Theorem 2.2. *k-Edge Eulerian Subgraph is solvable in $2^{O(k)}mn\log n$ time.*

Proof. We use the color-coding method. Randomly color each edge of the input graph G with probability $1/k$ by a color from $\{1,2,\dots,k\}$ to obtain an edge colored graph G'. Then a k-edge Eulerian subgraph in G, if it exists, has probability $k!/k^k$ to become colorful in G'. By Lemma 2.1, we can determine whether G' contains a colorful k-edge Eulerian subgraph in $O(k2^kmn)$ time, and therefore we can determine, with probability at least $k!/k^k$, whether G contains a k-edge Eulerian subgraph in $O(k2^kmn)$ time. We can derandomize the algorithm by using an (m,k)-family of perfect hash functions (for details, see Section 4 in [1]). Note that $\log m=\log n$ as $m=O(n^2)$. This yields a deterministic FPT algorithm that runs in $2^{O(k)}mn\log n$ time. □

Our next three FPT algorithms need to deal with set partitions, and the following bound by Berend and Tassa [2] on the *k-th Bell number* B_k, which denotes the number of partitions of a set of size k, is useful in analyzing the running times of our algorithms: $B_k<(0.792k/\ln(k+1))^k<k^k$.

We can solve k-Edge Even Subgraph by using a close connection between even graphs and Eulerian graphs.

Theorem 2.3. *k-Edge Even Subgraph is solvable in $k^{O(k)}mn\log n$ time.*

Proof. Again we use color-coding. Randomly color each edge of the input graph G with probability $1/k$ by a color from $\{1, 2, \dots, k\}$ to obtain an edge k-colored graph G'.

Since every component of an even subgraph is an Eulerian graph, G' has a colorful k-edge even subgraph iff there is a partition $\{C_1, \dots, C_t\}$ of colors $\{1, \dots, k\}$ such that each subgraph $G'(C_i) = (E_i, V_i)$ has a C_i-colorful Eulerian subgraph. Therefore we only need to check, for every partition $\{C_1, \dots, C_t\}$ of $\{1, 2, \dots, k\}$, whether each $G'(C_i)$ has a C_i-colorful Eulerian subgraph.

For a given partition of $\{1, 2, \dots, k\}$, it takes $O(m + n)$ time to construct all subgraphs $G'(C_i)$, and $O(|C_i| 2^{|C_i|} |E_i||V_i|)$ time (by Lemma 2.1) to find a C_i-colorful Eulerian subgraph in $G'(C_i)$. Since the number of partitions of $\{1, \dots, k\}$ is the k-th Bell number $B_k < k^k$, the total time of our randomized algorithm is $O(B_k k 2^k mn) = k^{O(k)} mn$. After derandomization by an (m, k)-family of perfect hash functions, we obtain a deterministic algorithm that runs in $k^{O(k)} mn \log n$ time. □

We now turn our attention to two parametric dual problems. We cannot use color-coding directly to find colorful $(m - k)$-edge subgraphs in FPT time. However, we can consider k edges not in $(m - k)$-edge subgraphs instead and use the following fact to transform our problems into that of finding colorful trails and even subgraphs.

Lemma 2.4. *Every graph G with $t \geqslant 0$ odd vertices can be decomposed into $t/2$ trails between odd vertices and an even graph, which may be empty.*

Proof. Note that the number of odd vertices is even in a connected graph. If $t > 0$, then there is a trail P between two odd vertices in a connected component of G, implying that the graph $G - E(P)$ has $t - 2$ odd vertices. Thus, we can obtain an even graph by repeating this process until the graph contains no odd vertices. □

We now regard $(m - k)$-Edge Even Subgraph as an edge deletion problem, and use color-coding to get an FPT algorithm.

Theorem 2.5. $(m - k)$-Edge Even Subgraph *is solvable in $k^{O(k)} mn \log n$ time.*

Proof. Clearly the problem is equivalent to whether we can delete k edges E' from the input graph G to obtain an even graph. Since the deletion of an edge can change at most two odd vertices into even ones, G can contain at most $2k$ odd vertices in order for the problem to have a solution, and therefore we only need to consider the case that G has $2t$ odd vertices for some $0 \leqslant t \leqslant k$.

Consider the graph $G' = G[E']$. Every odd vertex in G is also an odd vertex in G' and all other vertices in G' are even vertices. By Lemma 2.4, we can decompose G' into t trails between odd vertices and an even subgraph S (note that S may be an empty graph). Therefore, in order to find E', we only need to find a pairing $\{\{u_i, v_i\}\colon\ 1 \leqslant i \leqslant t\}$ of odd vertices in G such that G contains edge-disjoint (u_i, v_i)-trails T_i with $k' \leqslant k$ edges in total and $G - \bigcup_{i=1}^{t} E_i$, where E_i are edges of T_i, contains an even subgraph S with $k - k'$ edges.

Given a pairing $\{u_i, v_i\}$ of odd vertices, we use color coding to find E' that is consistent with this pairing. Randomly color each edge of G with probability $1/k$ by a color from $\{1, 2, \dots, k\}$ to obtain an edge k-colored graph G^*. Then G^* contains a colorful E' iff there is a partition $\{C_1, \dots, C_{t+1}\}$ of $\{1, \dots, k\}$ such that each $G^*(C_i)$, $1 \leqslant i \leqslant t$, has a C_i-colorful (u_i, v_i)-trail and $G^*(C_{t+1})$ has a C_{t+1}-colorful even subgraph (note that C_{t+1} may be empty). Therefore we consider each possible partition $\{C_1, \dots, C_{t+1}\}$ of $\{1, \dots, k\}$, and for each partition, we check whether each $G^*(C_i)$, $1 \leqslant i \leqslant t$, has a C_i-colorful (u_i, v_i)-trail and $G^*(C_{t+1})$ has a C_{t+1}-colorful even subgraph.

There are at most $(2t - 1)(2t - 3) \cdots 3 \cdot 1 = k^{O(k)}$ different ways to pair $2t$ odd vertices (note $t \leqslant k$) and $B_k < k^k$ different partitions of $\{1, \dots, k\}$. Given a pairing and a partition, it follows from Lemma 2.1 that it takes $O(k 2^k mn)$ time to check whether $G^*(C_i)$ has a C_i-colorful (u_i, v_i)-trail and $G^*(C_{t+1})$ has a C_{t+1}-colorful even subgraph. Therefore our randomized algorithm takes $k^{O(k)} mn$ time, which can be derandomized by an (m, k)-perfect family of perfect hash functions to produce a deterministic algorithm that runs in $k^{O(k)} mn \log n$ time. □

We can use a similar method to obtain an FPT algorithm for $(m - k)$-Edge Odd Subgraph.

Theorem 2.6. $(m - k)$-Edge Odd Subgraph *is solvable in $k^{O(k)} mn \log n$ time.*

Proof. First we observe that the problem is equivalent to whether we can delete k edges E' from the input G to obtain a graph where every nonisolated vertex has an odd degree. Since the deletion of an edge can make at most two even vertices into odd ones, G can contain at most $2k$ even vertices in order for the problem to have a solution, and therefore we only need to consider the case that G has $2t$ even vertices for some $0 \leqslant t \leqslant k$.

Consider the graph $G' = G[E']$. Every even vertex in G is an odd vertex in G' and all other vertices in G' are even vertices. By Lemma 2.4, G' can be decomposed into t trails between odd vertices and an even subgraph S, which may be empty. Therefore, in order to find E', we only need to find a pairing $\{\{u_i, v_i\}\colon\ 1 \leqslant i \leqslant t\}$ of even vertices in G such that G contains edge-disjoint (u_i, v_i)-trails T_i with $k' \leqslant k$ edges in total and $G - \bigcup_{i=1}^{t} E_i$, where E_i are edges of T_i, contains an even subgraph S with $k - k'$ edges.

Given a pairing $\{u_i, v_i\}$ of even vertices, we use color coding, as what we did for $(m-k)$-Edge Even Subgraph, to find E' that is consistent with the pairing. The same analysis gives us a deterministic algorithm that runs in $k^{O(k)}mn\log n$ time. □

3. FPT by kernelization and random separation

In this section, we use kernelization and random separation, together with Ramsey numbers, to obtain FPT algorithms for the following four problems:

- k-Edge Odd Subgraph
- k-Vertex Odd Subgraph
- k-Vertex Connected Odd Subgraph
- k-Vertex Even Subgraph.

The basic idea of *kernelization* is to reduce in polynomial time an instance (I, k) of a parameterized problem to an equivalent instance (I', k') with $|I'| \leqslant g(k)$ for some function $g(k)$ and $k' \leqslant k$. The resulting instance (I', k') is called a *kernel*, which can be easily solved in FPT time by exhaustive search whenever the problem is decidable.

The random separation method of Cai, Chan and Chan [5] is a powerful randomized method for solving parameterized problems, mainly for problems on graphs of bounded degree. Its main idea is to randomly partition the vertex set of a graph G into green and red vertices in a hope to separate a solution from the rest of G into green components (connected components in the graph induced by green vertices). For this purpose, we typically require that all vertices in a solution S are green and all vertices in $N(S)$ are red. Once we have such a partition, we can usually use dynamic programming to select appropriate green components to form a solution in polynomial time. Randomized algorithms from this method can be converted into deterministic FPT algorithms by using universal sets. A collection of binary vectors of length n is (n,t)-*universal* if for every subset of size t of the indices, all 2^t configurations appear. Naor, Schulman and Srinivasan [12] have a deterministic construction for (n,t)-universal sets of size $2^t t^{O(\log t)} \log n$ that can be listed in linear time.

The *Ramsey number* $R(k,k)$ is the smallest integer such that every graph on at least $R(k,k)$ vertices contains either a k clique or an independent set of size k. It is well known that $R(k,k) \leqslant \binom{2k-2}{k-1} = O(4^k/\sqrt{k})$, and we will use Ramsey numbers to bound the number of vertices or vertex degrees, which then enables us to use suitable techniques to obtain our FPT algorithms. We make one technical note here: since there is no efficient way to compute $R(k,k)$, we cannot use it directly in our algorithms. What we really use is the efficiently computable upper bound $\binom{2k-2}{k-1}$ of $R(k,k)$. However, we will write $R(k,k)$ in our proofs for the sake of a clear exposition.

First we show that k-Edge Odd Subgraph is FPT by kernelization.

Theorem 3.1. *k-Edge Odd Subgraph has a kernel with $O(k^2)$ vertices and edges that can be constructed in $O(m+n)$ time.*

Proof. We consider two cases. W.l.o.g., we may assume that G has no isolated vertices.

Case 1. Every vertex has degree at most k.

If G has at least $k(2k-1)$ edges, then we can easily use a greedy method to find a matching with k edges in $O(m+n)$ time, which forms a k-edge odd subgraph. Otherwise, G has less than $k(2k-1)$ edges and thus (G,k) itself is a kernel with $O(k^2)$ vertices and edges.

Case 2. There is a vertex v of degree at least $k+1$.

For odd k, any k edges incident with v form a k-edge odd subgraph. For even k, there is no solution when G is a star. Otherwise, G contains an edge e not incident with v. Since $d(v) \geqslant k+1$, there are $k-1$ edges E' incident with v but not adjacent to e, and hence E' and e together form a k-edge odd subgraph. □

Next we show that k-Vertex Odd Subgraph is also FPT by using Ramsey numbers to obtain a kernel.

Theorem 3.2. *k-Vertex Odd Subgraph has a kernel with $O(4^k\sqrt{k})$ vertices that can be constructed in $O(m+n)$ time.*

Proof. Since every graph has an even number of odd vertices, the problem has no solution for odd k and we will use Ramsey numbers to get a kernel for even k.

If G has a vertex v with at least $R(k-1,k-1)$ neighbors, then $N(v)$ contains $k-1$ vertices V' that induce either a clique or an independent set, and therefore $G[V' \cup \{v\}]$ is an induced odd subgraph with k vertices.

Otherwise, every vertex of G has degree at most d for $d = R(k-1,k-1)-1$. If G has at least dk vertices, then we can easily use a greedy method to find in $O(m+n)$ time an induced matching with $k/2$ edges (note that k is even), which gives us an induced odd subgraph on k vertices. Otherwise, G has less than $dk = O(4^k\sqrt{k})$ vertices and thus (G, k) itself is a kernel. □

For the next two problems, we will use random separation to obtain FPT algorithms. A key step in our algorithms is to use Ramsey numbers to bound the maximum degree of the graph under consideration, which paves the way for us to use the powerful random separation method.

Theorem 3.3. k-VERTEX CONNECTED ODD SUBGRAPH *is FPT.*

Proof. Like k-VERTEX ODD SUBGRAPH, the problem has no solution for odd k. When the input graph G has a vertex v with at least $R(k-1, k-1)$ neighbors, the odd subgraph we have obtained for k-VERTEX ODD SUBGRAPH is connected and thus is also a solution for our problem. Therefore we only need to consider the case that k is even and every vertex of G has degree at most d for $d = R(k-1, k-1) - 1$.

Since the degree of G is bounded by a function of k, we will use random separation to design an FPT algorithm. First we randomly color each vertex of G by either green or red with equal probability. A set V' of k vertices is a "well-colored solution" if

1. $G[V']$ is a connected odd graph, and
2. all vertices in V' are green and all vertices in its neighborhood $N(V')$ are red.

Let V_g be the set of green vertices of G. Then a well-colored solution is a connected component of $G[V_g]$ that is a k-vertex odd graph. Therefore, given a red–green coloring of G, we can easily determine whether there is a well-colored solution in $O(m+n)$ time.

Suppose that G has k vertices V' such that $G[V']$ is a connected odd graph. Then every vertex in V' is adjacent to at least one vertex in V', and thus $|V' \cup N(V')| \leqslant dk$, which implies that a random red–green coloring has probability at least 2^{-dk} to produce a well-colored solution. Therefore, when G has a solution, we can solve the problem with probability at least 2^{-dk} in $O(m+n)$ time.

To derandomize the algorithm, we need a family of 2-colorings (green and red) such that for every 2-coloring C of any dk vertices with k green vertices and $(d-1)k$ red vertices, there is a 2-coloring in the family that is consistent with C. We can use a family of (n, dk)-universal sets for this purpose, where we interpret each n-vector in the family as a 2-coloring for vertices of G. Since such a family can be generated in $2^{dk}dk^{O(\log dk)}n\log n$ time, we obtain a deterministic algorithm that runs in $O(f(k)(m+n)\log n)$ time for

$$f(k) = 2^{dk}(dk)^{O(\log dk)},$$

where $d = R(k-1, k-1) - 1$. □

For k-VERTEX EVEN SUBGRAPH, we will use Ramsey numbers to bound the maximum degree of the complement $\overline{G}$ of G, and then use random separation on $\overline{G}$ to obtain an FPT algorithm. Note that we can also use Theorem 3.2 to obtain an alternative FPT algorithm.

Theorem 3.4. k-VERTEX EVEN SUBGRAPH *is FPT.*

Proof. We consider two cases.

Case 1. k is odd.

If the input graph $G = (V, E)$ has at least $R(k, k)$ vertices, then it contains k vertices V' that form either a clique or an independent set, and hence $G[V']$ is a required even subgraph. Otherwise, G has less than $R(k, k)$ vertices and thus (G, k) itself is a kernel.

Case 2. k is even.

Let $d = R(k-1, k-1)$, and consider vertex degrees in the complement $\overline{G}$ of G. If G has a vertex v with $d_{\overline{G}}(v) \geqslant d$, then $N_{\overline{G}}(v)$ contains $k-1$ vertices V' that form either a clique or an independent set in G, and therefore $V' \cup \{v\}$ induces a k-vertex even subgraph in G.

Otherwise, $d_{\overline{G}}(v) < d$ for every vertex v, and we will use random separation on $\overline{G}$ to solve the problem. Let V' be a set of k vertices. Since k is even, we see that $G[V']$ is an even graph iff $\overline{G}[V']$ is an odd graph. Therefore, we only need to consider the problem of finding k vertices V' such that $\overline{G}[V']$ is an odd graph. Randomly color each vertex of $\overline{G}$ by either green or red with equal probability. We call V' a "well-colored solution" if

1. $\overline{G}[V']$ is an odd graph, and
2. all vertices in V' are green and all vertices in $N_{\overline{G}}(V')$ are red.

Let V_g be the set of green vertices of $\overline{G}$, and call connected components in $\overline{G}[V_g]$ *green components*. Then a well-colored solution consists of a collection of green components of $\overline{G}[V_g]$ such that each green component is an odd subgraph in $\overline{G}$ and the total number of vertices is k.

Given a red–green coloring, we can determine whether there is a well-colored solution in $O(m+kn)$ time as follows: Let $\mathcal{H}$ be the set of green components of $\overline{G}[V_g]$ such that each green component G_i has at most k vertices and is an odd subgraph in $\overline{G}$, and denote the number of vertices of G_i by n_i. What we need is to determine whether $\mathcal{H}$ contains a subset $\mathcal{H}'$ such that $\sum_{G_i \in \mathcal{H}'} n_i = k$, which can be done in $O(kn)$ time by using the standard dynamic programming algorithm for the subset sum problem.

If G has k vertices V' that induce an even graph, then $\overline{G}[V']$ is an odd graph and a random red–green coloring has probability at least 2^{-dk} to produce a well-colored solution as $|V' \cup N_{\overline{G}}(V')| \leqslant dk$. Therefore, when G has a solution, we can solve the problem with probability at least 2^{-dk} in $O(m+kn)$ time.

We can use a family of (n, dk)-universal sets to derandomize our algorithm to obtain a deterministic algorithm that runs in $O(f(k)(m+kn)\log n)$ time for

$$f(k) = 2^{dk}(dk)^{O(\log dk)},$$

where $d = R(k-1, k-1)$. □

4. W[1]-hardness

In this section we show that the following 4 problems are W[1]-hard, which indicates that they are very unlikely to admit FPT algorithms.

- $(n-k)$-Vertex Even Subgraph
- $(n-k)$-Vertex Eulerian Subgraph
- $(n-k)$-Vertex Odd Subgraph and
- $(n-k)$-Vertex Connected Odd Subgraph.

For the 8 unparameterized problems corresponding to our 16 parameterized even/odd subgraph problems, we show that 6 of them are NP-complete, which is another justification for studying our problems under the framework of parameterized complexity.

The above four listed problems are clearly equivalent to the problems of determining whether we can delete k vertices to obtain even (Eulerian, odd, and connected odd, respectively) graphs, and we will use this equivalence in our proofs. To establish their W[1]-hardness, we need the following problem which was shown by Downey et al. [8] to be W[1]-hard and also NP-complete when k is not regarded as a parameter.

k-Exact Odd Set
Instance: Bipartite graph $G = (X, Y; E)$ and positive integer k.
Question: Does G contain an *odd k-set*, i.e., k vertices $X' \subseteq X$ such that every vertex in Y has an odd number of neighbors in X'?
Parameter: k.

We will use fpt-reductions to establish our W[1]-hardness results. An *fpt-reduction* from a parameterized problem Π to another parameterized problem Π' is an FPT algorithm that transforms each instance (I, k) of Π into an instance (I', k') of Π' such that

1. (I, k) is a yes-instance of Π iff (I', k') is a yes-instance of Π', and
2. $k' \leqslant g(k)$ for some computable function $g(k)$.

In our reductions, we will *attach* a special graph H to a vertex v of a graph G to prevent vertex v from being deleted when we delete k vertices to obtain a required graph, where *attaching H to a vertex v of G* means that we identify a vertex of H with vertex v. The following lemma indicates that we can use a $(2k+1)$-cycle as H for our problems on even subgraphs.

Lemma 4.1. *If a $(2k+1)$-cycle C_v is attached to a vertex v of a graph, then the deletion of at most k vertices from C_v will result in at least one odd vertex.*

Proof. When we delete at most k vertices from C_v, we break it into at most k paths and thus at least one path P has at least two vertices. Then one end of P is different from vertex v, and thus is an odd vertex of degree 1 in the resulting graph. □

We are now ready to give our W[1]-hardness proofs. All reductions in our proofs actually take polynomial time, which is easy to see and will not be stated inside our proofs. We start with a reduction from k-Exact Odd Set to establish the following result.

Theorem 4.2. *$(n-k)$-Vertex Even Subgraph is W[1]-hard.*

Proof. We give an fpt-reduction from k-Exact Odd Set. For an arbitrary instance (G,k) of k-Exact Odd Set, we form an instance (G',k') of our problem by setting $k'=k$ and constructing G' from the bipartite graph $G=(X,Y;E)$ as follows:

1. For each vertex $y \in Y$, make a copy y' of y, i.e., create a new vertex y' and add an edge between y' and every vertex in $N_G(y)$. Add edge yy' if $d_G(y)$ is even.
2. Let $Y' = \{y'\colon\ y \in Y\}$. For each vertex $v \in Y \cup Y'$, attach a $(2k+1)$-cycle C_v.

Note that in graph G', X are even vertices and $Y \cup Y'$ are odd vertices.

We claim that G contains an odd k-set iff we can delete k vertices from G' to obtain an even graph. Suppose that X' is an odd k-set of G. Then in graph G', $N_{G'}(X') \subseteq Y \cup Y'$ and each vertex in $Y \cup Y'$ is adjacent to an odd number of vertices in X'. Therefore the deletion of X' will change all odd vertices in G', i.e., vertices in $Y \cup Y'$, into even ones, and thus $G'-X'$ is an even graph. Conversely, suppose that we can delete k vertices X^* from G' to obtain an even graph. Since in G' every vertex $v \in Y \cup Y'$ has a $(2k+1)$-cycle C_v attached to it, we deduce from Lemma 4.1 that $X^* \subseteq X$. For each vertex $y \in Y$, $d_{G'}(y)$ is odd and $d_{G'-X^*}(y)$ is even, and thus $N_{G'}(y)$ contains an odd number of vertices from X^*. Since $X^* \subseteq X$, we deduce that $N_G(y)$ contains an odd number of vertices from X^*, and therefore X^* is an odd k-set of G. □

Because even graphs and Eulerian graphs are closely related, we can obtain the following result by an easy reduction from $(n-k)$-Vertex Even Subgraph.

Theorem 4.3. *$(n-k)$-Vertex Eulerian Subgraph is W[1]-hard.*

Proof. We give an fpt-reduction from $(n-k)$-Vertex Even Subgraph. Given an arbitrary instance (G,k) of $(n-k)$-Vertex Even Subgraph, we construct an instance (G',k') of our problem by setting $k'=k$ and constructing G' from $G=(V,E)$ as follows:

1. Add two new vertices u and v, connect them to all vertices of G by edges, and add edge uv if $|V|-k$ is odd.
2. Attach $(2k+1)$-cycles C_u and C_v to u and v respectively.

We claim that one can delete k vertices from G to obtain an even graph iff one can delete k vertices from G' to obtain an Eulerian graph. Suppose that G has k vertices S such that $G-S$ is an even graph. Then $G'-S$ is a connected even graph (note that both $d_{G'-S}(u)$ and $d_{G'-S}(v)$ are even) and hence an Eulerian graph. Conversely, suppose that G' has k vertices S' such that $G-S'$ is Eulerian. We deduce from Lemma 4.1 that S' should consist of vertices inside G, and therefore $G-S'$ is an even graph. □

We now consider the two parametric dual problems on odd subgraphs. A $(4k+1)$-*wheel* is the graph obtained from a $(4k+1)$-cycle C by adding a new vertex x and edges between x and all vertices of C. The following lemma shows that we can attach a $(4k+1)$-wheel to a vertex v to prevent v from being deleted when we want to obtain an odd subgraph by vertex deletion. Note that all vertices of a $(4k+1)$-wheel are odd vertices, and when we attach it to vertex v, it does not matter which vertex of the wheel is identified with v.

Lemma 4.4. *If a $(4k+1)$-wheel W_v is attached to a vertex v of a graph, then the deletion of at most k vertices from W_v will result in at least one even vertex.*

Proof. When we delete at most k vertices from the $(4k+1)$-cycle of W_v, we will break the cycle into at most k paths and thus at least one path P contains at least 4 vertices. Therefore after the deletion, the two ends of P have degree 2 if the central vertex of W_v is not deleted, otherwise the internal vertices of P have degree 2. At least one of these vertices is different from vertex v, and thus it is an even vertex in the resulting graph. □

With the above lemma at hand, we can establish the W[1]-hardness of the two parametric dual problems on odd subgraphs by a reduction from $(n-k)$-Vertex Even Subgraph.

Theorem 4.5. *Both* $(n-k)$-Vertex Odd Subgraph *and* $(n-k)$-Vertex Connected Odd Subgraph *are* W[1]-*hard.*

Proof. We use the following fpt-reduction from $(n-k)$-Vertex Even Subgraph for both problems.

For an arbitrary instance (G,k) of $(n-k)$-Vertex Even Subgraph, we construct a graph G' from $G=(V,E)$ as follows:

1. Add a new vertex v, and connect it to all vertices of G by edges.
2. Attach to v a $(4k+1)$-wheel if $|V|-k$ is even and two $(4k+1)$-wheels otherwise.

First we show that one can delete k vertices from G to get an even graph iff one can delete k vertices from G' to get an odd graph. Suppose that G has k vertices S such that $G-S$ is an even graph. Then clearly $G'-S$ is an odd graph (note that $d_{G'-S}(v)$ is odd). Conversely, suppose that G' has k vertices S' such that $G'-S'$ is an odd graph. By Lemma 4.4, we see that no vertex of S' comes from the attached $(4k+1)$-wheel(s), and thus S' should consist of vertices of G. It follows that $G-S'$ is an even graph. This proves the W[1]-hardness of $(n-k)$-Vertex Odd Subgraph.

Observe that in the above proof, the odd graph $G'-S$ is connected when $G-S$ is an even graph. Therefore $G-S$ is an even graph iff $G'-S$ is a connected odd graph, which implies the W[1]-hardness of $(n-k)$-Vertex Connected Odd Subgraph. □

We close this section by establishing the NP-completeness of 6, out of 8, unparameterized problems corresponding to our 16 parameterized even/odd subgraph problems.

Theorem 4.6. *The following problems are NP-complete*:

- Induced Even Subgraph
- Induced Eulerian Subgraph
- Induced Odd Subgraph
- Induced Connected Odd Subgraph
- Eulerian Subgraph
- Connected Odd Subgraph.

Proof. These problems are clearly in NP. The NP-hardness of the first four problems follows from the facts that, when k is not regarded as a parameter, k-Exact Odd Set is NP-complete and our fpt-reductions (Theorems 4.2, 4.3 and 4.5) for W[1]-hardness of their corresponding parametric dual problems are actually polynomial reductions.

The NP-hardness of Eulerian Subgraph follows from the facts that an n-vertex cubic graph contains a Hamiltonian cycle iff it has an Eulerian subgraph of size n, and that the Hamiltonian Cycle problem is NP-complete for cubic graphs (see problem [GT37] in [9]).

For Connected Odd Subgraph, we use the following reduction from Hamiltonian Cycle on cubic graphs to establish its NP-hardness. Given an n-vertex cubic graph G, we construct a graph G' by adding, for each vertex v of G, a new vertex v' and edge vv'. Set $k=2n$. Note that G' has n vertices of degree 4 and we need to remove $n/2$ edges from G' to get a connected odd subgraph with $2n$ edges. From the above observation, it is easy to show that G contains a Hamiltonian cycle iff G' has a connected odd subgraph with k edges. □

5. Concluding remarks

We have studied the complexity of parameterized subgraph problems concerning even and odd subgraphs, and settled the fixed-parameter tractability for 12 out of our 16 problems. For the 4 remaining problems, we propose the following conjecture.

Conjecture 5.1. *For the following four problems, the first one is FPT and the other three are* W[1]-*hard*:

- k-Edge Connected Odd Subgraph
- k-Vertex Eulerian Subgraph[4]
- $(m-k)$-Edge Eulerian Subgraph
- $(m-k)$-Edge Connected Odd Subgraph.

Because the main purpose of this paper is to determine the fixed-parameter tractability of our subgraph problems, we have not paid too much attention to the efficiency of our FPT algorithms and we believe that many of our FPT algorithms can be improved.

[4] There is a slightly different definition for Eulerian graphs that allows an Eulerian graph to contain isolated vertices. For this definition, we can make some minor changes to our FPT algorithm for k-Vertex Even Subgraph to get an FPT algorithm for k-Vertex Eulerian Subgraph.

In terms of the tools we have used in designing our FPT algorithms, color-coding seems effective for our k-edge subgraph problems and their duals because of linear orderings of edges in trails, and random separation works very well when we can use Ramsey numbers to bound vertex degrees. We feel that this approach of using random separation will be helpful in obtaining FPT algorithms for other problems on general graphs: seek special (and maybe easy) solutions when the graph has vertices of large degree and use random separation to obtain hard solutions when all vertex degrees are bounded by a function of k. It is also interesting to see if we can come up with FPT algorithms without using Ramsey numbers, which will most likely improve the running times of our algorithms.

For FPT problems, a natural question is whether they admit polynomial kernels. Following the work of Bodlaender et al. [3], we have the following result for 4 subgraph problems concerning connected subgraphs because these problems are easily seen to be compositional and their corresponding unparameterized problems are NP-complete by Theorem 4.6.

Theorem 5.2. *None of the following four problems has a polynomial kernel, unless the polynomial hierarchy collapses to the* 3*rd level*:

- k-Edge Eulerian Subgraph
- k-Edge Connected Odd Subgraph
- k-Vertex Eulerian Subgraph
- k-Vertex Connected Odd Subgraph.

It is certainly interesting to know if our other FPT problems admit polynomial kernels. For the two problems that we have solved by kernelization, we make the following two conjectures.

Conjecture 5.3. k-Edge Odd Subgraph *has a kernel with* $O(k)$ *vertices.*

Conjecture 5.4. k-Vertex Odd Subgraph *has a polynomial kernel.*

We also note that the subgraph problems in the paper have many variations. For instance, instead of subgraphs with k vertices/edges, we can consider subgraphs with at least (or at most) k vertices/edges. These problems may behave quite differently from their corresponding k-vertex/edge subgraph problems: some become trivial, some are easily solvable using FPT algorithms for k-vertex/edge subgraphs, and some seem quite difficult. We will discuss these problems in a forthcoming paper [6]. We may also consider the problems of finding k-vertex/edge subgraphs to maximize the numbers of even vertices/edges, and we leave these problems to the reader to ponder.

Finally we conclude our paper with the following conjecture on two unparameterized problems.

Conjecture 5.5. *Both* Even Subgraph *and* Odd Subgraph *are NP-complete.*

Acknowledgements

The authors would like to thank the anonymous referees for their valuable comments and suggestions.

References

[1] N. Alon, R. Yuster, U. Zwick, Color-coding, J. ACM 42 (1995) 844–856.
[2] D. Berend, T. Tassa, Improved bounds on Bell numbers and on moments of sums of random variables, Probability and Mathematical Statistics 30 (2) (2010) 185–205.
[3] H.L. Bodlaender, R.G. Downey, M.R. Fellows, D. Hermelin, On problems without polynomial kernels, J. Comput. System Sci. 75 (8) (2009) 423–434.
[4] L. Cai, Fixed-parameter tractability of graph modification problems for hereditary properties, Inform. Process. Lett. 58 (4) (1996) 171–176.
[5] L. Cai, S.M. Chan, S.O. Chan, Random separation: a new method for solving fixed-cardinality optimization problems, in: IWPEC 2006, in: LNCS, vol. 4169, 2006, pp. 239–250.
[6] L. Cai, B. Yang, On optimal even/odd subgraph problems, manuscript.
[7] R.G. Downey, M.R. Fellows, Parameterized Complexity, Springer, Heidelberg, 1999.
[8] R.G. Downey, M.R. Fellows, A. Vardy, G. Whittle, The parameterized complexity of some fundamental problems in coding theory, SIAM J. Comput. 29 (1999) 545–570.
[9] M. Garey, D. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, San Francisco, 1979.
[10] S. Khot, V. Raman, Parameterized complexity of finding subgraphs with hereditary properties, Theoret. Comput. Sci. 289 (2) (2002) 997–1008.
[11] J.M. Lewis, M. Yannakakis, The node-deletion problem for hereditary properties is NP-complete, J. Comput. System Sci. 20 (2) (1980) 219–230.
[12] M. Naor, L.J. Schulman, A. Srinivasan, Splitters and near-optimal derandomization, in: Proceedings of the 36th Annual Symposium of Foundations of Computer Science (FOCS), 1995, pp. 182–191.
[13] A. Natanzon, R. Shamir, R. Sharan, Complexity classification of some edge modification problems, Discrete Appl. Math. 113 (1) (2001) 109–128.
[14] J.P. Schmidt, A. Siegel, The spatial complexity of oblivious k-probe hash functions, SIAM J. Comput. 19 (1990) 775–786.