Theoretical Computer Science 708 (2018) 18–33

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Bivariate complexity analysis of Almost Forest Deletion

Ashutosh Rai [a,*], Saket Saurabh [a,b,1]

[a] The Institute of Mathematical Sciences, HBNI, Chennai, India
[b] University of Bergen, Bergen, Norway

ARTICLE INFO

Article history:
Received 1 November 2016
Received in revised form 8 October 2017
Accepted 22 October 2017
Available online 24 October 2017
Communicated by F.V. Fomin

Keywords:
Bivariate Analysis
Almost Forest Deletion
Parameterized Complexity

ABSTRACT

In this paper we study a generalization of classic Feedback Vertex Set problem in the realm of multivariate complexity analysis. We say that a graph F is an l-forest if we can delete at most l edges from F to get a forest. That is, F is at most l edges away from being a forest. In this paper we introduce the Almost Forest Deletion problem, where given a graph G and integers k and l, the question is whether there exists a subset of at most k vertices such that its deletion leaves us an l-forest. We show that this problem admits an algorithm with running time $2^{\mathcal{O}(k+l)}n^{\mathcal{O}(1)}$ and a kernel of size $\mathcal{O}(kl(k+l))$. We also show that the problem admits a $2^{\mathcal{O}(\mathbf{tw})}n^{\mathcal{O}(1)}$ algorithm on bounded treewidth graphs, using which we design a subexponential algorithm for the problem on planar graphs.

1. Introduction

In the field of graph algorithms, vertex deletion problems constitute a considerable fraction. In these problems, we need to delete a small number of vertices such that the resulting graph satisfies certain properties. Many well known graph problems like Vertex Cover and Feedback Vertex Set fall under this category. Most of these problems are NP-complete due to a classic result by Lewis and Yannakakis [23]. The field of parameterized complexity tries to provide efficient algorithms for these NP-complete problems by going from the classical view of single-variate measure of the running time to a multivariate one. It aims at getting algorithms of running time $f(k)n^{\mathcal{O}(1)}$, where k is an integer measuring some aspect of the problem. The integer k is called the *parameter*. In most of the cases, the solution size is taken to be the parameter, which means that this approach gives faster algorithms when the solution is of small size. For more background, the reader is referred to the monographs [7,11,24].

Recently, there has been a trend of exploiting other structural properties of the graph other than the solution size [2,4,10,18,22]. For further reading, reader may refer to the recent survey by Fellows et al. [9]. In an earlier work, Guo et al. [17] introduced the notion of *"distance from triviality"* which looked at structural parameterization as a natural way to deal with problems which are polynomial time solvable on some graph classes. They argued that we could ask the same problems on some other (bigger) graph class which is close to the graph class on which the problem is polynomial time solvable, but the parameter is the *closeness* or the *distance* from the original graph class instead of the solution size.

* Corresponding author.
E-mail addresses: ashutosh@imsc.res.in (A. Rai), saket@imsc.res.in (S. Saurabh).

[1] The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement no. 306992. The conference version of this article appeared in the 21st Annual International Computing and Combinatorics Conference (COCOON), 2015.

https://doi.org/10.1016/j.tcs.2017.10.021
0304-3975/

In the same spirit, we introduce the notion of *distance from single parameter tractability*. We know that vertex deletion problems deal with deletion of vertices to get to some graph class. For example, the Vertex Cover problem deals with deleting vertices such that the resulting graph does not have any edge. Similarly, the well known Feedback Vertex Set problem talks about deleting vertices such that the resulting graph is a forest. What if we want to delete vertices so that the resulting graph class is *close* to the earlier graph class, while taking the *measure of closeness* as a parameter? This approach takes us from single variate parameterized algorithms to multivariate ones, which throws some light on the interplay between the parameters concerned. More precisely, they tell us about the trade-offs if we want to go away from the single parameter tractable version of a problem, and hence the term "distance from single parameter tractability".

There is already some work done which can be seen as examples of the notion of parameterizing by distance from single parameter tractability. For example, the Partial Vertex Cover and other related partial cover problems [1,15] come to mind, where after deletion of a small set of vertices, the resulting graph is close to an edgeless graph. In these problems, the measure of closeness is the number of edges. Similarly, work has been done on vertex deletion to get a graph of certain treewidth, which can be looked as deletion of vertices to get a graph close to a forest, where the measure of the closeness is the treewidth of the graph.

The algorithms of the above mentioned kind show the correlation between the solution size and the distance from single parameter tractability. Let the distance from single parameter tractability be the parameter l, and k be the number of vertices to be deleted. Suppose we have an algorithm with running time say $f(l)^{g(k)}n^{\mathcal{O}(1)}$, then is it possible to obtain an algorithm with running time $h_1(l)h_2(k)n^{\mathcal{O}(1)}$? That is, could we disentangle the function depending on both l and k to a product of functions where each function depends only on one of l or k. The answer to this question is *yes* if we take $f(l)^{g(l)}f(k)^{g(k)}n^{\mathcal{O}(1)}$ and the functions f and g are monotone, which is usually the case. However, if we ask for an algorithm with running time $h(l)2^{\mathcal{O}(g(k))}n^{\mathcal{O}(1)}$ then it becomes interesting. This kind of question can be asked for several problems. For an example, it is known that the Treewidth-η-Deletion problem, where the objective is to test whether there exists a vertex subset of size at most k such that its deletion leaves a graph of treewidth at most η. For $\eta = 0$ and $\eta = 1$ this corresponds to the Vertex Cover and Feedback Vertex Set problems, respectively. It is known that Treewidth-η-Deletion admits an algorithm with running time $f(\eta)^k n^{\mathcal{O}(1)}$ [13,19]. However, it is not known whether there exists an algorithm with running time $h(\eta)2^{\mathcal{O}(k)}n^{\mathcal{O}(1)}$. Clearly, algorithms with running time $h(l)2^{\mathcal{O}(g(k))}n^{\mathcal{O}(1)}$ are more desirable.

The Feedback Vertex Set problem has been widely studied in the field of parameterized algorithms. A series of results have improved the running times to $\mathcal{O}^*(3.619^k)$ in deterministic setting [21] and $\mathcal{O}^*(3^k)$ in randomized setting [4], where the $\mathcal{O}^*$ notation hides the polynomial factors. Looking at this problem in the notion of distance from single parameter tractability, number of edges comes to mind as a natural measure of distance. More precisely, we try to address the question of deleting vertices such that the resulting graph is l edges away from being a forest. We call such forests l-forests, and the problem Almost Forest Deletion. The main focus of this paper is to design algorithm for this problem with parameters both l and k.

Our results. We show that Almost Forest Deletion can be solved in time $2^{\mathcal{O}(k+l)}n^{\mathcal{O}(1)}$. We arrive at the result using the iterative compression technique which was introduced in [26] and a non-trivial measure which helps us in getting the desired running time. Then we explore the kernelization complexity of the problem, and show that Almost Forest Deletion admits a polynomial kernel with $\mathcal{O}(kl(k+l))$ edges. For arriving at the result, we first make use of the Expansion Lemma and Gallai's theorem for reducing the maximum degree of the graph, and then we bound the size of the graph. The techniques used for kernelization are inspired by those used by Thomassé [27] to get a quadratic kernel for Feedback Vertex Set. It is easy to see that for a Yes instances (G, k, l) of Almost Forest Deletion, the treewidth of G is bounded by $k + l$. Since we have an algorithm of the form $2^{\mathcal{O}(k+l)}n^{\mathcal{O}(1)}$ on general graphs, the question of finding an $2^{\mathcal{O}(\mathbf{tw})}n^{\mathcal{O}(1)}$ algorithm becomes interesting for bounded treewidth graphs. We answer this question affirmatively by giving an $2^{\mathcal{O}(\mathbf{tw})}n^{\mathcal{O}(1)}$ algorithm for graphs which come with a tree decomposition of width **tw**. This algorithm, along with showing that our problem is minor-bidimensional, immediately gives rise to a subexponential algorithm for Almost Forest Deletion on H-minor free graphs running in time $2^{\mathcal{O}(\sqrt{l+k})}n^{\mathcal{O}(1)}$ by applying the bidimensionality framework given by Demaine et al. [5]. Our methods are based on the known methods to solve the Feedback Vertex Set problem.

2. Preliminaries

For a graph G, we denote the set of vertices of the graph by $V(G)$ and the set of edges of the graph by $E(G)$. For a set $S \subseteq V(G)$, the *subgraph of G induced by S* is denoted by $G[S]$ and it is defined as the subgraph of G with vertex set S and edge set $\{(u, v) \in E(G) : u, v \in S\}$ and the subgraph obtained after deleting S is denoted as $G - S$. If H is a subgraph of G, we write $H \subseteq G$ and for two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, by $G_1 \cup G_2$, we denote the graph $(V_1 \cup V_2, E_1 \cup E_2)$. All vertices adjacent to a vertex v are called *neighbours* of v and the set of all such vertices is called the *neighbourhood* of v. A *k-flower* in a graph is a set of k cycles which are vertex-disjoint except for one vertex v, which is shared by all the cycles in the set. The vertex v is called *centre* of the flower and the cycles are called the *petals* of the flower. A *forest* is a graph which does not contain any cycle. An *l-forest is* a graph which is at most l edges away from being a forest, i.e. the graph can be transformed into a forest by deleting at most l edges. For a connected component C of a graph, we call the quantity $|E(G[C])| - |C| + 1$ the *excess of* C and denote it by $\mathsf{ex}(C)$. It can also be equivalently defined as the minimum number of