Journal of Computer and System Sciences 74 (2008) 1188–1198

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

Improved algorithms for feedback vertex set problems ☆

Jianer Chen [a], Fedor V. Fomin [b], Yang Liu [a], Songjian Lu [a], Yngve Villanger [b,*]

[a] *Department of Computer Science, Texas A&M University, College Station, TX 77843-3112, USA*
[b] *Department of Informatics, University of Bergen, N-5020 Bergen, Norway*

ARTICLE INFO

Article history:
Received 13 December 2007
Received in revised form 24 April 2008
Available online 13 June 2008

Keywords:
Parameterized complexity
Feedback vertex set

ABSTRACT

We present improved parameterized algorithms for the FEEDBACK VERTEX SET problem on both unweighted and weighted graphs. Both algorithms run in time $\mathcal{O}(5^k kn^2)$. The algorithms construct a feedback vertex set of size at most k (in the weighted case this set is of minimum weight among the feedback vertex sets of size at most k) in a given graph G of n vertices, or report that no such feedback vertex set exists in G.

© 2008 Elsevier Inc. All rights reserved.

1. Introduction

Let G be an undirected graph. A *feedback vertex set* (FVS) F in G is a set of vertices in G whose removal results in an acyclic graph (or equivalently, every cycle in G contains at least one vertex in F). The problem of finding a minimum feedback vertex set in a graph is one of the classic NP-complete problems [14] and has many applications. The history of the problem can be traced back to the early '60s. For several decades, many different algorithmic approaches were tried on this problem, including approximation algorithms, linear programming, local search, polyhedral combinatorics, and probabilistic algorithms (see the survey of Festa et al. [10]). There are also exact algorithms that find a minimum FVS in a graph of n vertices in time $\mathcal{O}(1.8899^n)$ [17] and in time $\mathcal{O}(1.7548^n)$ [11].

An important application of the FVS problem is *deadlock recovery* in operating systems [19], in which a deadlock is presented by a cycle in a *system resource-allocation graph* G. In order to recover from deadlocks, we need to abort a set of processes in the system, i.e., to remove a set of vertices in the graph G, so that all cycles in G are broken. Equivalently, we need to find an FVS in G. The problem also has a version on weighted graphs, where the weight of a vertex can be interpreted as the cost of aborting the corresponding process. In this case, we are looking for an FVS in G whose weight is minimized.

In a practical system resource-allocation graph G, it can be expected that the size k of the minimum FVS in G, i.e., the number of vertices in the FVS, is fairly small. This motivated the study of *parameterized algorithms* for the FVS problem that find an FVS of k vertices in a graph of n vertices (where the weight of the FVS is minimized, in the case of weighted graphs), and run in time $f(k)n^{\mathcal{O}(1)}$ for a fixed function f (thus, the algorithms become practically efficient when the value k is small).

This line of research has received considerable attention, mostly on the unweighted version of the problem. The first group of parameterized algorithms of running time $f(k)n^{\mathcal{O}(1)}$ for the FVS problem on unweighted graphs was given by

☆ This work was supported in part by the National Science Foundation of USA under the Grant CCF-0430683 and by the Research Council of Norway. The extended abstract of this paper was presented at WADS 2007.

* Corresponding author.

E-mail addresses: chen@cs.tamu.edu (J. Chen), fomin@ii.uib.no (F.V. Fomin), yangliu@cs.tamu.edu (Y. Liu), sjlu@cs.tamu.edu (S. Lu), yngvev@ii.uib.no (Y. Villanger).

0022-0000/$ – see front matter © 2008 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcss.2008.05.002

Bodlaender, Fellows [3,8]	$\mathcal{O}(17(k^4)!n^{\mathcal{O}(1)})$
Downey and Fellows [9]	$\mathcal{O}((2k+1)^k n^2)$
Raman et al. [16]	$\mathcal{O}(\max\{12^k, (4\log k)^k\} n^{2.376})$
Kanj et al. [13]	$\mathcal{O}((2\log k + 2\log\log k + 18)^k n^2)$
Raman et al. [15]	$\mathcal{O}((12\log k/\log\log k + 6)^k n^{2.376})$
Guo et al. [12]	$\mathcal{O}((37.7)^k n^2)$
Dehne et al. [7]	$\mathcal{O}((10.6)^k n^3)$

Fig. 1. The history of parameterized algorithms for the unweighted FVS problem.

Bodlaender [3] and by Downey and Fellows [8]. Since then a chain of dramatic improvements was obtained by different researchers (see Fig. 1 for references).

Randomized parameterized algorithms have also been studied in the literature for the FVS problem, for both unweighted and weighted graphs. The best known randomized parameterized algorithms for the FVS problems are due to Becker et al. [2], who developed a randomized algorithm of running time $\mathcal{O}(4^k kn^2)$ for the FVS problem on unweighted graphs, and a randomized algorithm of running time $\mathcal{O}(6^k kn^2)$ for the FVS problem on weighted graphs. To our knowledge, no deterministic algorithm of running time $f(k)n^{O(1)}$ for any function f was known prior to our results for the weighted FVS problem.

1.1. Our results

The main result of this paper is an algorithm that for a given integer k and a weighted graph G, either finds a minimum weight FVS in G of at most k vertices, or correctly reports that G contains no FVS of at most k vertices. The running time of our algorithm is $\mathcal{O}(5^k kn^2)$. This improves and generalizes a long chain of results in parameterized algorithms. Let us remark that the running time of our (deterministic) algorithm comes close to that of the best randomized algorithm for the FVS problem on unweighted graphs and is better than the running time of the previous best randomized algorithm for the FVS problem on weighted graphs.

The general approach of our algorithm is based on the *iterative compression* method [18], which has been successfully used recently for improved algorithms for the FVS and other problems [7,12,18]. The method starts with an FVS of k vertices for a small subgraph of the given graph, and iteratively grows the small subgraph while keeping an FVS of k vertices in the grown subgraph until the subgraph becomes the original input graph. This method makes it possible to reduce the original FVS problem on general graphs to the FVS problem on graphs with a special decomposition structure. The main contribution of the current paper is the development of a general algorithmic technique that identifies a dual parameter in problem instances that limits the number of times where the original parameter k cannot be effectively reduced during a branch and search process. In particular, for the FVS problem on graphs of the above special decomposition structure, a measure is introduced that nicely combines the original parameter and the dual parameter and bounds effectively the running time of a branch and search algorithm for the FVS problem. This technique leads to a simpler but significantly more efficient parameterized algorithm for the FVS problem on unweighted graphs. Moreover, the introduction of the measure greatly simplifies the processing of degree-2 vertices in a weighted graph, and enables us to solve the FVS problem on weighted graphs in the same complexity as that for the problem on unweighted graphs. Note that this is significant because no previous algorithms for the FVS problem on unweighted graphs can be extended to weighted graphs mainly because of the lack of an effective method for handling degree-2 vertices. Finally, the technique of dual parameters seems to be of general usefulness for the development of parameterized algorithms, and has been used more recently in solving other parameterized problems [4,5].

The remaining part of this paper is organized as follows. In Section 2, we provide in full details a simpler algorithm and its analysis for unweighted graphs. This is done for clearer demonstration of our approach. We also indicate why this simpler algorithm does not work for weighted graphs. In Section 3, we obtain the main result of the paper, the algorithm for the weighted FVS problem. This generalization of the results from Section 2 is not straightforward and requires a number of new structures and techniques.

2. On feedback vertex sets in unweighted graphs

In this section, we consider the FVS problem on unweighted graphs. We start with some terminologies. A *forest* is a graph that contains no cycles. A *tree* is a forest that is connected (therefore, a forest can be equivalently defined as a collection of disjoint trees). Let W be a subset of vertices in a graph $G = (V, E)$. We will denote by $G[W]$ the subgraph of G that is induced by the vertex set W. For simplicity we will use the notation $G - w$ and $G - W$ for respectively $G[V \setminus \{w\}]$ and $G[V \setminus W]$ where $w \in V$ and $W \subseteq V$. A pair (V_1, V_2) of vertex subsets in a graph $G = (V, E)$ is a *forest bipartition* of G if $V_1 \cup V_2 = V$, $V_1 \cap V_2 = \emptyset$, and both induced subgraphs $G[V_1]$ and $G[V_2]$ are forests. For a vertex $u \in V$ the degree of u will be the number of edges incident to u.

Let G be a graph and let F be a subset of vertices in G. The set F is a *feedback vertex set* (shortly, FVS) of G if $G - F$ is a forest. The *size* of an FVS F is the number of vertices in F.