Theoretical Computer Science 657 (2017) 86–97

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Exact algorithms for the maximum dissociation set and minimum 3-path vertex cover problems

Mingyu Xiao, Shaowei Kou

School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China

ARTICLE INFO

Article history:
Received 13 October 2015
Received in revised form 7 April 2016
Accepted 13 April 2016
Available online 7 June 2016

Keywords:
Exact algorithm
Graph algorithm
Dissociation number
3-path vertex cover
Dynamic programming

ABSTRACT

A dissociation set in a graph $G = (V, E)$ is a vertex subset D such that the subgraph $G[D]$ induced on D has vertex degree at most 1. A 3-path vertex cover in a graph is a vertex subset C such that every path of three vertices contains at least one vertex from C. A vertex set D is a dissociation set if and only if $V \setminus D$ is a 3-path vertex cover. There are many applications for dissociation sets and 3-path vertex covers. However, it is NP-hard to compute a dissociation set of maximum size or a 3-path vertex cover of minimum size in graphs. Several exact algorithms have been proposed for these two problems and they can be solved in $O^*(1.4658^n)$ time in n-vertex graphs. In this paper, we reveal some interesting structural properties of the two problems, which allow us to solve them in $O^*(1.4656^n)$ time and polynomial space or $O^*(1.3659^n)$ time and exponential space.

1. Introduction

A subset of vertices in a graph is called a *dissociation set* if it induces a subgraph with vertex degree at most 1. The maximum size of a dissociation set is called the *dissociation number* of the graph. To compute a dissociation set of maximum size or the dissociation number is NP-hard even in bipartite graphs or planar graphs [26]. The complexity of this problem in more restricted graph classes has been studied. It remains NP-hard even in C_4-free bipartite graphs with vertex degree at most 3 [3]. But it is polynomially solvable in trees and some other graph classes [1–3,5,6,9,12,16–18]. Computing the dissociation number can be helpful in finding a lower bound for the 1-improper chromatic number of a graph; see [11]. In fact, dissociation set generalizes two other important concepts in graphs: independent set [23] and induced matching [25]. The Maximum Dissociation Set problem, to find a maximum dissociation set is also a special case of the Maximum Bounded-Degree-d problem [7], in which we are finding a maximum induced subgraph with degree bounded by d. The dual problem of Maximum Dissociation Set is known as the Minimum 3-path Vertex Cover problem. A vertex subset C is called a 3-path vertex cover if every path of three vertices in a graph contains at least one vertex from C and Minimum 3-path Vertex Cover is to find a 3-path vertex cover of minimum size. There are also some applications for Minimum 3-path Vertex Cover [5,13]. It remains NP-hard to compute a special 3-path vertex cover C such that the degree of the induced graph $G[C]$ is bounded by any constant $d_0 \geq 0$ [24]. A more general problem, to find a minimum p-path vertex cover has been considered in the literature [4,5].

Maximum Dissociation Set and Minimum 3-path Vertex Cover have been studied in approximation algorithms, parameterized algorithms and exact algorithms. For Minimum 3-path Vertex Cover, there is a randomized approximation algorithm with an expected approximation ratio of $\frac{23}{11}$ [13]. For the problem parameterized by the size k of 3-path vertex cover, it is

E-mail addresses: myxiao@uestc.edu.cn (M. Xiao), shaoweikou1993@163.com (S. Kou).

http://dx.doi.org/10.1016/j.tcs.2016.04.043

fixed parameter tractable. The running time bound has been improved by several times recently [21,22,14]. The current best result is $O^*(1.8172^k)$ by Katrenič [14]. On the other hand, it is hard to compute a dissociation set of size at least k in approximation and parameterized algorithms. No approximation algorithms with constant ratio exist under some assumption [17]. It is W[1]-hard to find a 2-plex of size k in a graph [15], which implies the W[1]-hardness of our problem parameterized by the size k of the dissociation set. In terms of exact algorithms, it does not make sense to distinguish these two problems. Kardoš et al. [13] gave an $O^*(1.5171^n)$-time algorithm to compute a maximum dissociation set in an n-vertex graph. Chang et al. [7] improved the result to $O^*(1.4658^n)$. Their algorithm was analyzed by the measure-and-conquer method. Although many fastest exact algorithms are obtained via the measure-and-conquer method, this paper will not use this technique and turn back to a normal measure. The reason is that if we use the measure-and-conquer method by setting different weights to vertices, we may not be able to use dynamic programming to further improve the time complexity to $O^*(1.3659^n)$. It is also surprising that our polynomial-space algorithm using normal measure runs in $O^*(1.4656^n)$ time, even faster than the $O^*(1.4658^n)$-time algorithm analyzed by the measure-and-conquer method [7]. Our improvement relies on new structural properties developed in this paper.

The organization of this paper is as follows: Section 2 collects some technical preliminaries and some basic definitions that will be used in this paper. Section 3 introduces some structure properties. Section 4 discusses the main algorithm and analyzes its running time bound. Section 5 discusses how to reduce the time complexity via dynamic programming. In the end of this paper, some concluding remarks are given.

2. Preliminaries

We let $G=(V,E)$ denote a simple and undirected graph with $n=|V|$ vertices and $m=|E|$ edges. A singleton $\{v\}$ may be simply denoted by v. The vertex set and edge set of a graph G' are denoted by $V(G')$ and $E(G')$, respectively. For a subgraph (resp., a vertex subset) X, the subgraph induced by $V(X)$ (resp., X) is simply denoted by $G[X]$, and $G[V \setminus V(X)]$ (resp., $G[V \setminus X]$) is also written as $G \setminus X$. A vertex in a subgraph or a vertex subset X is also called a *X-vertex*. For a vertex subset X, let $N(X)$ denote the set of *open neighbors* of X, i.e., the vertices in $V \setminus X$ adjacent to some vertex in X, and $N[X]$ denote the set of *closed neighbors* of X, i.e., $N(X) \cup X$. Let $N_2(v)$ denote the set of vertices with distance exactly 2 from v. The *degree* of a vertex v in a graph G, denoted by $d(v)$, is defined to be the number of neighbors of v in G. We also use $d_X(v)$ to denote the number of neighbors of v in a subgraph X. A vertex v is *dominated* by a neighbor u of it if v is adjacent to all neighbors of u. A vertex $s \in N_2(v)$ is called a *satellite* of v if there is a neighbor p_s of v such that $N[p_s]-N[v]=\{s\}$. The vertex p_s is also called a *parent* of the satellite s at v. If there is a neighbor u of v such that $|N[u]-N[v]|=2$, then any vertex in $N[u]-N[v]$ is a *tw-satellite* of v, the two tw-satellites in $N[u]-N[v]$ are *twins*, and u is a *parent* of the tw-satellites at v. The set of all tw-satellites of a vertex v is denoted by T_v. A vertex subset V' is called a *dissociation set* of a graph if the induced graph $G[V']$ has maximum degree 1. In fact, in this paper, we will consider a general version of Maximum Dissociation Set, in which a specified vertex subset S is given and we are going to find a maximum dissociation set containing S. See the following definition.

Generalized Maximum Dissociation Set (MDS)
Input: A undirected graph $G=(V,E)$ and a vertex subset $S \subset V$.
Output: A vertex set $D \supseteq S$ of maximum cardinality such that D is a dissociation set of G.

Our algorithm is a branch-and-search algorithm. In this kind of algorithms, we recursively branch on the current instance into several smaller instances to search for a solution. Assume we use w as the measure to evaluate the size of an instance, where w can be the number of vertices in a graph for graph problems. Let $C(w)$ denote the maximum number of leaves in the search tree generated by the algorithm for any instance with measure at most w. If a branch generates l branches and the measure in the i-th branch decreases by at least a_i, then the branch creates a recurrence

$$C(w) \le C(w-a_1)+C(w-a_2)+\cdots+C(w-a_l).$$

The largest root of the function $f(x)=1-\sum_{i=1}^{l} x^{-a_i}$ is called the *branching factor* of the recurrence. The running time of the algorithm can be bounded by $O^*(\gamma^n)$, where γ is the maximum branching factor among all branching factors in the algorithm. More details about the analysis can be found in the monograph [8]. Note that we will use a modified O-notation that suppresses all polynomially bounded factors. For two functions f and g, we write $f(n)=O^*(g(n))$ if $f(n)=O(g(n)ply(n))$ for some polynomial function $ply(n)$.

The simplest branching rule in our algorithm is

(B1): *Branching on a vertex $v \in V \setminus S$ to generate two instances by either including v to S or deleting v from the graph directly.*

This rule is not often used, because for most cases it is not effective. Indeed, some of previous papers [13,14] use the following branching rule