Discrete Mathematics 337 (2014) 119–126

Contents lists available at ScienceDirect

Discrete Mathematics

journal homepage: www.elsevier.com/locate/disc

Decomposition by maxclique separators

Márcia R. Cerioli [a,b], Hugo Nobrega [a], Petrucio Viana [c,*]

[a] COPPE/Engenharia de Sistemas e Computação, Universidade Federal do Rio de Janeiro, Caixa Postal 68511, 21941-972, Rio de Janeiro, Brazil

[b] Instituto de Matemática, Universidade Federal do Rio de Janeiro, Caixa Postal 68530, 21941-909, Rio de Janeiro, Brazil

[c] Instituto de Matemática e Estatística, Universidade Federal Fluminense, Rua Mário Santos Braga s/n, 24020-140, Niterói, Brazil

ARTICLE INFO

Article history:
Received 27 September 2013
Received in revised form 24 July 2014
Accepted 26 July 2014
Available online 18 September 2014

Keywords:
Graph decomposition
Clique separator
Maximal clique
Algorithm

ABSTRACT

We present a minimal counterexample to an $O(|V||E|)$ algorithm for decomposing a graph $G = (V, E)$ by maximal cliques proposed by R. Tarjan. We also present a modification to this algorithm which is correct while retaining its $O(|V||E|)$ complexity.

1. Introduction

Graph decompositions play central roles in many parts of graph theory, both in concrete areas such as algorithmic aspects of the theory and in abstract areas [3,4]. The outline of this general technique for solving a graph problem is:

1. decompose a given graph into several smaller graphs by some specified criterion;
2. iteratively repeat the process for each smaller graph, until all the graphs obtained are *atoms*, i.e., graphs which cannot be further decomposed by that criterion. One can view the result of this process as a rooted tree, with the original graph as the root and the atoms as the leaves. Such a tree is called a *decomposition tree* — see Fig. 1;
3. if the problem can be solved for the leaves of this tree, and if a method is known for *composing* the solutions for the children of an internal node of the tree into a solution for the node itself, then repeatedly compose these solutions until finally a solution for the original graph is obtained.

Note that if polynomial algorithms exist for (a) solving the problem on the leaves of the decomposition tree, and (b) combining solutions upward in the tree, and if furthermore the decomposition tree can be computed in polynomial time, then this clearly indicates a polynomial algorithm to solve the problem on the original graph.

One particular type of graph decomposition which has turned out to have many interesting applications is that of decomposition by *clique separators*. In [9], R. Tarjan proposed an $O(|V|\,|E|)$ algorithm that decomposes a graph $G = (V, E)$ by clique separators and showed how these decompositions can be used to efficiently solve many classical problems such as vertex coloring, maximum independent set, among others, in many graph classes. In [5], H.-G. Leimer extended Tarjan's results with an algorithm that decomposed a graph by clique *minimal* separators, i.e., by cliques which are also minimal separators with respect to inclusion. For a survey of algorithms for decomposition by clique minimal separators, we refer to [1].

* Corresponding author.
E-mail addresses: cerioli@cos.ufrj.br (M.R. Cerioli), hugonobrega@gmail.com (H. Nobrega), petrucio@cos.ufrj.br (P. Viana).

http://dx.doi.org/10.1016/j.disc.2014.07.020
0012-365X/

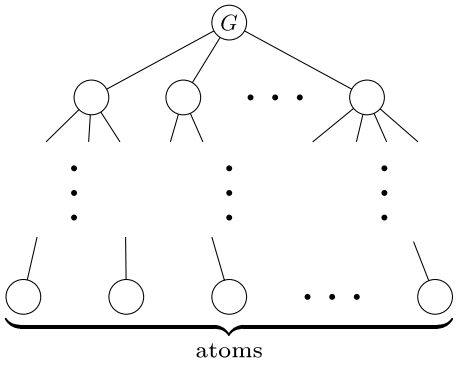

Fig. 1. Scheme of a decomposition tree.

However, in some cases one is interested in decomposing a graph by those separators that are *maximal* cliques (*maxcliques*), which we call *maxclique separators*. For instance, C. Monma and V. Wei used these decompositions to characterize and recognize several related classes of intersection graphs of paths in trees [6]. All graphs in these classes have a polynomial number of maxcliques so that a direct adaptation of a maxclique enumeration algorithm can be used to find such a decomposition. However, it seems desirable to obtain an algorithm which sidesteps maxclique enumeration and that might therefore be more efficient in general. Also, if one is interested in adapting and applying Monma and Wei's techniques to a more general setting, an efficient and general algorithm for decomposing a graph by maxclique separators is crucial.

In order to obtain such a general algorithm, Tarjan added a note at the end of [9] proposing a simple modification of his algorithm to find a decomposition by maxclique separators and pointed out that this modified algorithm retained the same complexity. It has also been noticed [8] that this modified algorithm can indeed be used in recognition algorithms for some of the graph classes studied by Monma and Wei.

In this work, we prove that this proposed algorithm as specified in [9] is incorrect and propose a modification to it in order to obtain a correct algorithm while also retaining the $O(|V|\,|E|)$ complexity. In Section 2, we briefly review Tarjan's original algorithms, providing a (minimal) counterexample to his maxclique decomposition algorithm. In Section 3, we present our alternative algorithm, proving its correctness and $O(|V|\,|E|)$ complexity. By means of this new algorithm, the applications pointed out in [8] can be correctly performed.

2. Definitions and preliminary results

All graphs are assumed to be simple, undirected and finite. Definitions and notations not specified here are standard and can be found in [2]. In particular, given a graph $G = (V, E)$, we denote $|V|$ and $|E|$ by n and m, respectively.

An *elimination ordering* of G is a bijection between $V(G)$ and $\{1, \ldots, n\}$, denoting the position that each vertex of G occupies in the order. Given an elimination ordering π of G, we say $u, v \in V(G)$ are *fillable* w.r.t. π if they are nonadjacent and there exists a path $P = u, x_1, \ldots, x_k, v$ in G such that

$$\pi(x_i) < \min\{\pi(u), \pi(v)\}$$

for all $i \in \{1, \ldots, k\}$. The set F_π of *fill-in edges created by* π is the set of all fillable pairs of vertices of G, and the graph $G_\pi = (V, E \cup F_\pi)$ is then called the *filled-in graph*. An elimination ordering π of G is *minimal* if there exists no elimination ordering π' of G such that $F_{\pi'} \subset F_\pi$. An example of a graph and two of its elimination orderings, along with the fill-in edges they create, can be found in Fig. 2. Note that the ordering presented in Fig. 2(b) is not minimal, while the one in Fig. 2(c) is.

Recall that a graph is *chordal* if it contains no induced cycle of length at least four. The following theorem shows how, in a certain sense, elimination orderings generalize the well-known *perfect elimination orderings* of chordal graphs.

Theorem 1 (*Fulkerson and Gross, 1965; Rose, Tarjan, and Lueker, 1976 cf. [9]*)**.** *A graph $G = (V, E)$ is chordal if, and only if, there exists an elimination ordering π of G such that $F_\pi = \emptyset$. Furthermore, an elimination ordering π of G creates no fill-in edges in G_π.*

Therefore, since G_π is chordal for any π, the problem of determining an elimination ordering with *minimum* (w.r.t. cardinality) set of fill-in edges is at least as hard as the problem of determining the minimum number of edges that must be added to a graph so that the resulting graph is chordal, which is NP-hard (Yannakakis, 1981 cf. [9]). However, finding an elimination ordering with *minimal* (w.r.t. inclusion) set of fill-in edges can be done in $O(nm)$ time by a variation of lexicographic breadth-first search due to Rose, Tarjan, and Lueker [7].

Given an elimination ordering π and a vertex v of G, we define

$$C_\pi(v) = \{u \in V : \pi(u) > \pi(v) \text{ and } uv \in E \cup F_\pi\}.$$

Note that, once F_π is known, determining $C_\pi(v)$ for all v can be done in $O(m + |F_\pi|)$ time since, for each $uv \in E \cup F_\pi$, exactly one of $v \in C_\pi(u)$ or $u \in C_\pi(v)$ is true, so that a simple traversal of the edges of G_π is enough to determine all of these sets.