Revisiting Decomposition by Clique Separators

David Coudert, Guillaume Ducoffe

Revisiting decomposition by clique separators*

David Coudert[1] and Guillaume Ducoffe[2,3]

[1]Université Côte d'Azur, Inria, CNRS, I3S, France
[2]ICI – National Institute for Research and Development in Informatics, Romania
[3]The Research Institute of the University of Bucharest ICUB

Abstract

We study the complexity of decomposing a graph by means of clique separators. This common algorithmic tool, first introduced by Tarjan, allows to cut a graph into smaller pieces, and so, it can be applied to preprocess the graph in the computation of optimization problems. However, the best-known algorithms for computing a decomposition have respective $\mathcal{O}(nm)$-time and $\mathcal{O}(n^{(3+\alpha)/2}) = o(n^{2.69})$-time complexity, with $\alpha < 2.3729$ being the exponent for matrix multiplication. Such running times are prohibitive for large graphs. Here we prove that for every graph G, a decomposition can be computed in $\mathcal{O}(T(G) + \min\{n^\alpha, \omega^2 n\})$-time with $T(G)$ and ω being respectively the time needed to compute a minimal triangulation of G and the clique-number of G. In particular, it implies that every graph can be decomposed by clique separators in $\mathcal{O}(n^\alpha \log n)$-time. Based on prior work from Kratsch et al., we prove in addition that decomposing a graph by clique-separators is as least as hard as triangle detection. Therefore, the existence of any $o(n^\alpha)$-time algorithm for this problem would be a significant breakthrough in the field of algorithmic. Finally, our main result implies that planar graphs, bounded-treewidth graphs and bounded-degree graphs can be decomposed by clique separators in linear or quasi-linear time.

Keyword: clique minimal separator decomposition; minimal triangulation; clique-number; treewidth; planar graphs; bounded-degree graphs.

1 Introduction

Our purpose in this work is to study the complexity of separating a graph with all its minimal separators that are cliques. In the literature, such minimal separators are called clique-minimal separators while the decomposition process is sometimes called "clique minimal separator decomposition" [5]. We refer to [5] for a survey and for the terminology used in this paper (technical definitions are postponed to Section 2).

Clique separators have attracted some interest in algorithmic graph theory since the seminal work of Gavril [27] where he showed that they can be applied to preprocess the graphs in the computation of some optimization problems. More precisely, there are hard problems on graphs (theoretically or in practice) that can be solved on each subgraph of the clique minimal separator decomposition separately. See for instance [3, 10, 15, 16, 22, 24, 32, 45]. Furthermore, although

*This work has been partially supported by ANR project Stint under reference ANR-13-BS02-0007, ANR program "Investments for the Future" under reference ANR-11-LABX-0031-01, and the Inria associated team AlDyNet.

this decomposition is strictly less powerful than other well-known graph decompositions – such as, *e.g.*, tree decompositions and rank decompositions – there do exist NP-hard problems that can be solved on graphs when the subgraphs obtained with the clique minimal separator decomposition are "simple enough" w.r.t. the problem. This was first noted by Gavril in [27] for the so-called clique-separable graphs. Other classes of graphs with such a "simple" decomposition comprise the chordal graphs (that can be decomposed by clique separators into complete subgraphs), the EPT graphs [28] and the P_6-free graphs [14]. Note that general graphs may fail to contain a clique-minimal separator (we will call them prime graphs in the following), however in practice the biological networks, the graph of the autonomous systems of the Internet and some other complex networks do contain clique-minimal separators – as supported by some experimentations [1, 16, 17].

Motivated by these applications, we will propose in this work new lower and upper bounds on the time complexity $C(n, m)$ for computing the clique minimal separator decomposition of n-vertex m-edge graphs[1].

Related work Historically, the first polynomial upper bound on this complexity was established by Whitesides in [44], where she proposed an $\mathcal{O}(nm)$-time algorithm for computing a clique separator in a graph (if any); as noted by Tarjan, the latter implies that $C(n, m) = \mathcal{O}(n^2m)$. Later on, Tarjan has introduced the first $\mathcal{O}(nm)$-time algorithm for decomposing a graph by means of its clique-separators [40]. Since then, and in spite of improvements on his work (especially by Leimer) [6, 21, 36], the state of the art for this problem has remained $C(n, m) = \mathcal{O}(nm)$. The only exception to this we are aware of is an $o(n^{2.69})$-time algorithm in [34], that improves the running time for dense graphs. On the practical side, such running times are too prohibitive on large graphs with thousands of vertices and sometimes billions of edges (see for instance [18]). Therefore, it is interesting to obtain a finer-grained complexity analysis for this problem. We here investigate on the *optimal* time for computing the clique minimal separator decomposition.

Our work is an example of hardness results in P, that is a growing area of research (*e.g.*, see [12]). Roughly, finer-grained notions of reduction between polynomial-time solvable problems are used in order to prove that if A can be reduced to B and B can be solved in $\tilde{\mathcal{O}}(n^{q-\varepsilon})$-time[2] then A can also be solved in $\tilde{\mathcal{O}}(n^{q-\varepsilon})$-time [43]. This combined with standard complexity assumptions – such as the Strong Exponential-Time Hypothesis [31] – allows one to prove *conditional* time complexity lower bounds for many problems in P, and unsuspected relationships between these problems. In our case we will assume, like in [42], the computational equivalence between Triangle Detection and Matrix Multiplication. This equivalence has been proved in [43] for combinatorial algorithms (*i.e.*, not using algebraic methods).

To the best of our knowledge, the time complexity of clique minimal separator decomposition has received little attention in the literature. We are only aware of a recent article [6] introducing a generic framework to compute this decomposition. Interestingly, this framework applies to all the best-known algorithms for computing the clique minimal separator decomposition. Indeed, all these algorithms follow the same three steps:

1. Compute a minimal triangulation of the graph;

2. Find the clique-minimal separators of the graph (using the minimal triangulation);

[1] As usual, the dependency in m will be discarded for sparse graphs (with $m = \mathcal{O}(n)$ edges) and for the time bounds in $\mathcal{O}(n^{2+\varepsilon} + m)$.

[2] The $\tilde{\mathcal{O}}(.)$ notation suppresses polylog factors.

3. Finally, recursively disconnect the graph with its clique-minimal separators.

We emphasize that the first step: computing a minimal triangulation, has been extensively studied (see [29] for a survey). So far, the best-known algorithm to compute a minimal triangulation of a graph has an $\tilde{\mathcal{O}}(n^\alpha) = \tilde{o}(n^{2.3729})$-time complexity with $\alpha < 2.3729$ being the exponent for matrix multiplication. Note that it is less than $\mathcal{O}(n^{(3+\alpha)/2}) = o(n^{2.69})$, that has been the best-known complexity for computing the clique minimal separator decomposition of a graph – until this note.

Furthermore, new clique minimal separator decomposition algorithms are proposed in [6] that are provably faster than the classical approach in some cases, that is, they run in $\mathcal{O}(nm_0)$-time for some $m_0 \leq m$. In order to compare these algorithms with our work, let us note that the authors in [6] claim that bounded-treewidth graphs can be decomposed by clique-separators in *quadratic-time*, whereas we will show that it can be done in quasi *linear-time*. Faster algorithms to compute the clique minimal separator decomposition can also be found in [2, 7] for some specific graph classes, but the latter algorithms deeply rely upon the structural properties of these graphs.

Closest to our work are two papers from Kratsch and Spinrad [33, 34]. In [34], they describe what has been, until this note, the best-known algorithm to compute the clique minimal separator decomposition for dense graphs. The latter algorithm has running time $\mathcal{O}(n^{(3+\alpha)/2})$, that follows from an algorithm to compute a minimal triangulation of the graph within the same time bounds. We will generalize their result in our work, proving that the clique minimal separator decomposition can be computed in $\mathcal{O}(n^\alpha)$-time if *any* minimal triangulation of the graph is given[3]. Furthermore, lower bounds on the complexity $C(n, m)$ of computing the clique minimal separator decomposition can be deduced from some results in [33]. In particular, they show that finding a clique-minimal separator in a graph is at least as hard as finding a simplicial vertex, *even if a minimal elimination ordering is given as part of the input*. The latter implies that computing a minimal triangulation is not the only complexity bottleneck of clique minimal separator decomposition algorithms.

Our contributions There are different versions of clique minimal separator decomposition studied in the literature, that only slightly differ in their output [6, 36, 40]. Our results in what follows apply to any of them. Indeed, we first prove as a warm-up that they are computationally equivalent (Theorem 4).

On the negative side, we then prove a conditional lower bound on the complexity of computing the clique minimal separator decomposition. More precisely, we will build upon a result in [33] in order to prove that this decomposition is at least as hard as TRIANGLE DETECTION (Theorem 6).

We next focus on the two last steps of clique minimal separator decomposition algorithms; that is, we ignore the first step of computing a minimal triangulation. Our main result is that the clique-minimal separators of a graph G can be computed in $\mathcal{O}(T(G) + \min\{n^\alpha, \omega^2 n\})$-time, with $T(G)$ and ω being respectively the time needed to compute a minimal triangulation of G and the *clique-number* of G (let us remind that the clique-number of G is the size of a largest clique in G). The latter result follows from two algorithms that respectively run in $\mathcal{O}(T(G) + n^\alpha)$-time (Proposition 7) and in $\mathcal{O}(T(G) + \omega^2 n)$-time (Proposition 8). Furthermore, whereas the first algorithm (in $\mathcal{O}(T(G) + n^\alpha)$-time) relies upon fast matrix multiplication, the second one is purely combinatorial and can be easily implemented.

We finally notice that any graph G can be decomposed by clique-separators within the same time bound $\mathcal{O}(T(G) + \min\{n^\alpha, \omega^2 n\})$-time (Theorem 9). Since a minimal triangulation can be

[3]It seems to us that the techniques in [34] could also be applied to any minimal triangulation. Nonetheless, we will propose a method that is, to our opinion, slightly simpler than theirs.

computed in $T(G) = \tilde{\mathcal{O}}(n^\alpha)$-time for any graph G [30], our main result implies that any graph can be decomposed by clique-separators in $\tilde{\mathcal{O}}(n^\alpha)$-time. Furthermore, faster and practical algorithms can be obtained in some cases – whenever the graphs have bounded clique-number and a minimal triangulation can be computed efficiently. We will show it is the case for interesting graph classes such as planar graphs, bounded-treewidth graphs and bounded-degree graphs (see Section 6.1 for details).

Altogether, this is hint that our $\tilde{\mathcal{O}}(n^\alpha)$-time clique minimal separator decomposition algorithm is optimal up to polylogarithmic factors – assuming the computational equivalence between TRIANGLE DETECTION and MATRIX MULTIPLICATION [43, 42].

Organization Definitions and useful notations are given in Section 2. Last, we will conclude this paper with an open conjecture in Section 7.

2 Definitions and preliminaries

We will use standard graph terminology from [11]. Graphs in this study are finite, simple (hence with neither loops nor multiple edges) connected and unweighted, unless stated otherwise. Given a graph $G = (V, E)$ and a set $S \subseteq V$, we will denote by $G[S]$ the subgraph of G that is induced by S. The open neighbourhood of S, denoted by $N(S)$, is the set of all vertices in $G[V \setminus S]$ that are adjacent to at least one vertex in S. The closed neighbourhood of S is denoted by $N[S] = N(S) \cup S$.

From the algorithmic point of view, we make the standard assumption that graphs are encoded as adjacency lists. For every vertex, we can enumerate its neighbourhood in constant-time per neighbour. However, we make no assumption on the complexity of testing whether two vertices are adjacent (*i.e.*, it can take super-constant time).

Before formally introducing the clique minimal separator decomposition in Section 2.2, we will survey in Section 2.1 some useful results on related (hyper-)graph classes.

2.1 Some basics on triangulations

Chordal graphs Let $G = (V, E)$ be a graph and let $\mathcal{C}_G$ be the set of its (inclusionwise) maximal cliques. A *clique-tree* of G is a tree $T = (\mathcal{C}_G, F)$ whose nodes are the maximal cliques of G such that for every vertex $v \in V$, the set of maximal cliques containing v induces a subtree of T. A graph $G = (V, E)$ is called *chordal* if and only if it has a clique-tree [26]. Equivalently, a chordal graph is a graph with no induced cycle of length at least four, hence the terminology of "triangulated graphs" is sometimes preferred [38]. Chordal graphs enjoy many algorithmic properties that can be generalized – to some extent – to clique-minimal separator decomposition [8]. We will use some of them in what follows.

Dual hypertrees and their join trees A hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{E})$ is called a *dual hypertree* if the underlying graph $G_\mathcal{H} = (\mathcal{V}, E')$, obtained by adding an edge between every two vertices in a common hyperedge of $\mathcal{H}$, is chordal and it has all its maximal cliques contained in $\mathcal{E}$ [13]. Furthermore, it is a *reduced* hypergraph if no hyperedge of $\mathcal{E}$ is properly contained in another one. Every hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{E})$ can be tranformed to the reduced hypergraph $\mathcal{H}' = (\mathcal{V}, \mathcal{E}')$, whose hyperedges are the maximal elements of $\mathcal{E}$. For general hypergraphs, this transformation can be

done in quadratic-time and this is optimal under the Strong Exponential-Time Hypothesis [12]. However, for dual hypertrees it can be done in linear-time [41].

Given a dual hypertree $\mathcal{H} = (\mathcal{V}, \mathcal{E})$, a *join tree* of $\mathcal{H}$ is a tree $T = (\mathcal{E}, F)$ whose nodes are the hyperedges of $\mathcal{H}$ such that for every vertex $v \in \mathcal{V}$, the set of hyperedges containing v induces a subtree of T. Note that if $\mathcal{H}$ is reduced then the clique-trees of its underlying graph $G_{\mathcal{H}}$ are exactly the join trees of $\mathcal{H}$.

Minimal triangulation Finally, a *triangulation* of $G = (V, E)$ is any chordal supergraph $H = (V, E \cup F)$ of G. In particular, H is a *minimal triangulation* of G if for any strict subset $F' \subset F$, the supergraph $H' = (V, E \cup F')$ is not a triangulation of G. Minimal triangulations have many applications in the study of optimization problems on graphs and sparse matrices. We refer to [29] for a survey.

Computing a minimal triangulation of a given graph $G = (V, E)$ can be done in $\mathcal{O}(\min\{nm, n^\alpha \log n\})$-time [30, 39]. However, this has been improved for various graph classes such as planar graphs [19], bounded-degree graphs [20], etc. We will come back to this point in Section 6.1.

2.2 Decomposition by clique separators

Let us now formally introduce clique-separators and clique minimal separator decomposition.

Clique-minimal separators A set $S \subseteq V$ is a *separator* in G if there are at least two connected components in $G[V \setminus S]$. In particular, a *full component* in $G[V \setminus S]$ is any connected component C in $G[V \setminus S]$ satisfying that $N(C) = S$ (note that a full component might fail to exist). The set S is called a *minimal separator* in G if it is a separator and there are at least two full components in $G[V \setminus S]$. In particular, S is a *clique-minimal separator* if it is a minimal separator and $G[S]$ is a complete subgraph.

There exist strong relationships between minimal triangulations and clique-minimal separators. Namely, we will use the following lemma.

Lemma 1 ([5])**.** *For any minimal triangulation H of a graph G, the clique-minimal separators in G are exactly the minimal separators in H that induce complete subgraphs of G.*

Derived systems and atoms A graph is called *prime* if it does not contain any clique-minimal separator, otherwise it is called *reducible*. Examples of prime graphs are the complete graph K_n and the cycle graph C_n. Examples of reducible graphs are all the chordal graphs that are not a complete graph [23].

If a graph $G = (V, E)$ is reducible then its vertex-set can be partitioned into $V = A \cup S \cup B$ with S being a clique-separator and A and B being a partition of the connected components of $G \setminus S$ into two nonempty subsets. Furthermore, the gotten subgraphs $G_A = G[A \cup S]$ and $G_B = G[B \cup S]$ can be similarly decomposed until all the derived subgraphs are prime. Such a family of prime subgraphs is called a *derived system* of G.

Clique minimal separator decomposition is a particular example of derived systems. Formally, "a" clique minimal separator decomposition of a graph $G = (V, E)$ is a derived system that is obtained by decomposing G with its clique-*minimal* separators. Leimer has proved that the clique minimal separator decomposition of G is unique [36]. More precisely, the *atoms* of G are the

inclusionwise maximal subsets A_i such that $G[A_i]$ is prime. The following relationship holds between the atoms of a graph and its derived systems:

Lemma 2 ([36]). *Every derived system of a graph $G = (V, E)$ contains its atoms. Furthermore, the clique minimal separator decomposition of G is the family of all its atoms, and so, it is the* unique minimal derived system *of G.*

In what follows, we study the complexity of the following graph problem:

Problem 1 (Clique Minimal Separator Decomposition)**.**

Input: *A graph $G = (V, E)$.*

Output: *The atoms of G.*

Note that the output of the clique minimal separator decomposition has size $\sum_i |A_i|$ in $\mathcal{O}(n+m)$, where the sets A_i denote the atoms of G [4]. In contrast with the above result, we observe that there may be $\Omega(\omega^2 n)$ edges in total cumulated on the subgraphs that are induced by the atoms of G, with ω being the clique-number of G, that is, the size of a largest complete subgraph in G (*e.g.*, see Figure 1).

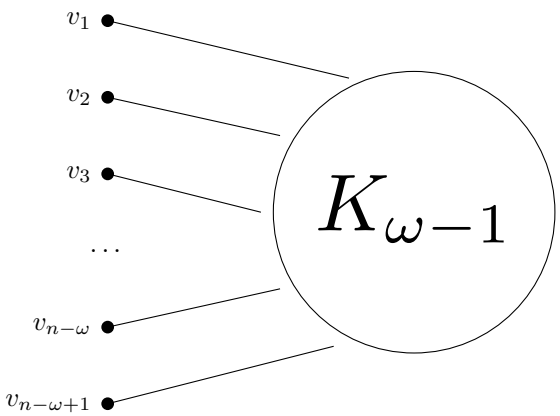

Figure 1: An n-vertex split graph with clique-number ω. The vertices are bipartitioned in a clique $K_{\omega-1}$ with $\omega - 1$ vertices and an independent set with $n - \omega + 1$ vertices. Furthermore, each vertex in the independent set is adjacent to all vertices in the clique. The atoms of the graph are exactly the closed neighbourhoods $N[v_i]$, $1 \leq i \leq n - \omega + 1$. Therefore, there are $\omega(\omega - 1)(n - \omega + 1)/2$ edges in total in the subgraphs induced by the atoms.

Alternative encodings Most clique-minimal separator decomposition algorithms output a tree representation of the atoms, that is sometimes called a *binary decomposition tree* [40] and is recursively defined as follows.

- If G is a prime graph then it has a unique binary decomposition tree, that is a single node labeled with V.

- Else, a binary decomposition tree of G is any binary rooted tree such that: its root is labeled with a clique-minimal separator S in G, the left child of the root is a leaf-node that is labelled with $A = S \cup C$ where C is a full component of $G[V \setminus S]$ and $G[A]$ is prime, furthermore the subtree that is rooted at the right child of the root is a binary decomposition tree of $G[V \setminus C]$.

Informally, a binary decomposition tree can be seen as the trace of some execution of a clique minimal separator decomposition algorithm (*e.g.*, a decomposition ordering). Note that a binary decomposition tree of a graph may not be unique.

Lemma 3 ([36]). *Let $G = (V, E)$ and T be a binary decomposition tree of G. Each leaf-node of T is labeled with an atom of G, and each atom of G appears exactly once as a leaf-node label in T.*

An *atom tree* is another tree-like representation of the clique minimal separator decomposition, defined in [6] as follows. For any graph $G = (V, E)$, the graph G^* obtained by saturating all its atoms is chordal and its maximal cliques are exactly the atoms of G [36]. The atom trees of G are exactly the clique-trees of G^*.

Note that any binary decomposition tree and any atom tree have size in $\mathcal{O}(\sum_i |A_i|)$, where the sets A_i denote the atoms of G, that is in $\mathcal{O}(n + m)$. We will show in Section 3 that the two above tree-like representations can be obtained "for free" from clique minimal separator decomposition.

3 Computational equivalence between the representations

When decomposing a graph by clique-separators, there are different ways to encode the result (*i.e.*, as a derived system, a binary decomposition tree or an atom tree). Hence, it is interesting to ask whether one encoding is easier to compute than the others. We will show that they all can be computed in linear-time from the clique minimal separator decomposition (as defined in Problem 1).

Theorem 4. *For every graph $G = (V, E)$, the following problems are computationally equivalent:*

1. *Computing a derived system of G;*
2. *Computing the clique minimal separator decomposition of G;*
3. *Computing an atom tree of G;*
4. *Computing a binary decomposition tree of G.*

Proof. The four equivalence are trivial when G is prime, so, we will only consider the case when G is reducible. It is enough to prove the four following (cyclic) implications.

1 $\Longrightarrow$ 2 Let $\mathcal{S}$ be a derived system of G. Since all the subgraphs in $\mathcal{S}$ are prime, and since by Lemma 2 all the atoms of G are in $\mathcal{S}$, by maximality of the atoms the inclusionwise maximal elements in $\mathcal{S}$ are exactly the atoms of G. Furthermore, let $G_{\mathcal{S}}$ be obtained from G by saturating all the subsets in $\mathcal{S}$. Note that $G_{\mathcal{S}}$ is exactly G^*, obtained from G by saturating all its atoms. Therefore, $G_{\mathcal{S}}$ is chordal and all its maximal cliques are contained in $\mathcal{S}$ [36]. The latter implies that the hypergraph $\mathcal{H}_{\mathcal{S}} = (V, \mathcal{S})$ – whose underlying graph is $G_{\mathcal{S}}$ – is a dual hypertree (cf. Section 2.1). Then, reducing H to $H' = (V, \mathcal{C})$ – whose hyperedges are the maximal elements in $\mathcal{S}$ – will lead to the clique minimal separator decomposition $\mathcal{C}$ of G. For a dual hypertree, it can be done in linear-time [41].

2 $\Longrightarrow$ 3 Let $\mathcal{C}$ be the clique minimal separator decomposition of G, that is the family of all atoms A_i of G. Recall that G^*, obtained from G by saturating all its atoms, is chordal and its maximal cliques are exactly the atoms of G [36]. Therefore the hypergraph $\mathcal{H} = (V, \mathcal{C})$ is a dual hypertree (cf. Section 2.1). In particular, since G^* is the underlying graph of $\mathcal{H}$ and $\mathcal{H}$ is reduced by maximality of the atoms, the atom trees of G are exactly the join trees of $\mathcal{H}$. A join tree of a given dual hypertree can be computed in linear-time [41].

3 $\Longrightarrow$ 4 Let us fix an atom tree $T^*_G = (\mathcal{C}, F)$ of G. Let A_i be any leaf-node in the atom tree (an atom whose corresponding node in the tree has degree at most one). Since G is reducible, the node corresponding to A_i is adjacent to a unique other node in the atom tree, whose label is denoted by A_j. Furthermore, let $C_i = A_i \setminus A_j$. By [6, Proposition 3.6], we have that: $S = A_i \cap A_j$ is a clique-minimal separator, C_i is a full component in $G[V \setminus S]$ and $T^*_{G \setminus C_i} = (\mathcal{C} \setminus A_i, F \setminus \{A_i, A_j\})$ is an atom tree of $G \setminus C_i$. Therefore, we can start a binary decomposition tree of G whose root is labeled by S, whose left-child is labeled by A_i, and whose right child is obtained by applying recursively the same operation to the atom tree $T^*_{G \setminus C_i}$ of $G \setminus C_i$.

Overall, computing the intersections between every two adjacent atoms in T_G can be done in time $\mathcal{O}(\sum_i |A_i|)$ as follows. We arbitrarily root T_G. For every atom A_j, let $A_{i_1}, A_{i_2}, \ldots, A_{i_k}$ be its children. We first mark all the vertices of A_j. Then we scan $A_{i_1}, A_{i_2}, \ldots, A_{i_k}$, and we so compute the sets $A_j \cap A_{i_1}, A_j \cap A_{i_2}, \ldots, A_j \cap A_{i_k}$. Finally, the vertices of A_j are unmarked. Doing so, every atom is scanned at most three times (*i.e.*, twice to mark then unmark its vertices, and at most one more time when we visit its parent). Hence, the running time is $\mathcal{O}(\sum_i |A_i|)$, that is in $\mathcal{O}(n + m)$-time [4].

4 $\Longrightarrow$ 1 Let T be any binary decomposition tree of G. Since by Lemma 3 the leaf-node labels of T are exactly the atoms of G, the atoms of G can be extracted from T in time $\mathcal{O}(\sum_i |A_i|)$, that is in $\mathcal{O}(n + m)$-time [4]. The family of all atoms is a derived system of G by Lemma 2. □

4 Conditional time complexity lower bound

Let us start proving the hardness of computing clique minimal separator decomposition by reducing this problem from TRIANGLE DETECTION. In the following, recall that a simplicial vertex is one whose closed neighbourhood induces a complete subgraph. We will need the following lemma.

Lemma 5 ([33])**.** TRIANGLE DETECTION *for n-vertex graphs can be reduced in $\mathcal{O}(n^2)$-time to counting the number of simplicial vertices in a graph with $3n + 2$ vertices.*

Theorem 6. *The problem of detecting a triangle in an n-vertex graph reduces in quadratic time to the problem of computing the clique minimal separator decomposition of a graph with $3n + 2$ vertices.*

Proof. Let $G = (V, E)$ be any graph with $3n + 2$ vertices. In order to prove the theorem, by Lemma 5 it is sufficient to prove that counting the number of simplicial vertices in G can be done in $\mathcal{O}(n + m)$-time if the clique minimal separator decomposition of G is given.

We claim that for every simplicial vertex $v \in V$, its closed neighbourhood $N[v]$ is an atom, and in particular it is the unique atom containing v. Indeed, suppose for the sake of contradiction

that there exists $u \notin N[v]$ such that u and v lie on a same atom A. Then, $N(v) \cap A$ is an uv-separator in the subgraph $G[A]$. Since $N(v) \cap A$ is a clique, the latter contradicts that $G[A]$ has no clique-separator. Therefore, every atom containing v is a subset of $N[v]$. Finally, since $G[N[v]]$ is complete, we have that $G[N[v]]$ has no clique-separator, and so, by inclusionwise maximality of the atoms, $N[v]$ is the unique atom containing v, that proves the claim.

In particular, it follows from this above claim that a vertex is simplicial if and only if it is contained in a unique atom and this atom is a clique. Indeed, if a vertex is simplicial then by the above claim it satisfies the desired property. Conversely, if a vertex v is uniquely contained in an atom A and A is a clique then v is trivially simplicial with its neighbourhood being equal to $N[v] = A$.

Let us take advantage of this above characterization of simplicial vertices in order to count them in G. Let $A_1, A_2, \ldots, A_k$ be the atoms of G. We will use in the following analysis that $\sum_{i=1}^{k} |A_i|$ is in $\mathcal{O}(n + m)$ [4].

We first compute an atom tree of G. By Theorem 4, it can be done in $\mathcal{O}(n + m)$-time. Then, let A_i be any leaf-node in the atom tree (an atom whose corresponding node in the tree has degree at most one). Since the intersection of two atoms is a clique [36], we have that A_i is a clique if and only if every vertex that is uniquely contained in A_i has degree $|A_i| - 1$. Furthermore, we have by [6, Proposition 3.6] that if we name C_i the set of vertices that are uniquely contained in A_i and we discard A_i from the atom tree, one obtains an atom tree of $G \setminus C_i$. Therefore, we can repeat the above process in order to list all the atoms of G that are cliques. As explained for Theorem 4, computing the intersections between every two adjacent atoms in T_G can be done in time $\mathcal{O}(\sum_i |A_i|)$ during a tree traversal of T_G. So, overall, it takes time $\mathcal{O}(\sum_{v \in V} |N(v)| + \sum_{i=1}^{k} |A_i|)$, that is in $\mathcal{O}(n + m)$-time.

Finally, let $A_{i_1}, \ldots, A_{i_l}$ be the atoms of G that are cliques. We can count all the vertices that are only contained in A_{i_j}, for some $1 \leq j \leq l$, simply by scanning all the atoms in time $\mathcal{O}(\sum_{i=1}^{k} |A_i|)$, that is in $\mathcal{O}(n + m)$-time. Since we proved that these are exactly the simplicial vertices of G, the latter achieves proving that counting the number of simplicial vertices can be done in $\mathcal{O}(n+m)$-time if the atoms are given. □

5 Computing the clique minimal separators

This section is devoted to the fast computation of the clique-minimal separators in a graph. We will introduce two methods which both make use of Lemma 1.

Proposition 7. *Let $G = (V, E)$. Suppose that a minimal triangulation of G can be computed in time $T(G)$. Then, the clique-minimal separators of G can be computed in $\mathcal{O}(T(G) + n^{\alpha})$-time.*

Proof. Let $H = (V, E \cup F)$ be a minimal triangulation of G, with $f = |F|$ fill edges. By the hypothesis it can be computed in time $T(G)$. Let $\Xi = (S_1, S_2, \ldots, S_l)$ be the minimal separators of H, with $l \leq n$. By [26], the family Ξ can be computed in $\mathcal{O}(n+m+f) = \mathcal{O}(n^2)$-time. Furthermore, recall that by Lemma 1 the clique-minimal separators of G are exactly the separators in Ξ that are cliques of G. In order to compute them, let $V = (v_1, v_2, \ldots, v_n)$ be totally ordered. Let $\mathcal{A}_G$ be the adjacency matrix of G, and let $\mathcal{B}_H$ be the *clique matrix* of H (of dimensions $n \times l$) defined as follows. For every $1 \leq i \leq n$ and for every $1 \leq j \leq l$, we have $b_{ij} = 1$ if $v_i \in S_j$ and $b_{ij} = 0$ otherwise. Then, $\mathcal{C} = \mathcal{A}_G \mathcal{B}_H$ is a matrix of dimensions $n \times l$. It can be computed in $\mathcal{O}(n^{\alpha})$-time by

using fast matrix multiplication since $l \leq n$ [35]. Furthermore, for every $1 \leq i \leq n$ and for every $1 \leq j \leq l$, we have $c_{ij} = |N_G(v_i) \cap S_j|$. Therefore, $S_j \in \Xi$ is a clique-minimal separator of G if and only if we have $c_{ij} = |S_j| - 1$ for every $v_i \in S_j$. As a result, the clique-minimal separators of G are obtained from the matrix $\mathcal{C}$ in time $\mathcal{O}(\sum_{j=1}^{l} |S_j|)$, that is $\mathcal{O}(n + m + f) = \mathcal{O}(n^2)$. □

Proposition 8. *Let $G = (V, E)$. Suppose that a minimal triangulation of G can be computed in time $T(G)$. Then, the clique-minimal separators of G can be computed in $\mathcal{O}(T(G) + \omega^2 n)$-time.*

Proof. Let $H = (V, E \cup F)$ be a minimal triangulation of G, with $f = |F|$ fill edges. By the hypothesis it can be computed in time $T(G)$. Let us compute the set Ξ of all minimal separators of H. By [26], the family Ξ can be computed in $\mathcal{O}(n + m + f) = \mathcal{O}(T(G))$-time.

Let $\Xi^{(0)} = \Xi$. Our aim is to remove separators of H from Ξ^0 until it only contains the clique-minimal separators of G. In order to achieve the result, let $V = (v_1, v_2, \ldots, v_n)$ be totally ordered. We consider the vertices sequentially. For every $1 \leq i \leq n$, let $\Xi^{(i-1)} \subseteq \Xi$ be the minimal separators of H that have not been discarded at a previous step $j < i$. Furthermore, let $\mathcal{S}_i \subseteq \Xi^{(i-1)}$ contain every $S \in \Xi^{(i-1)}$ such that $v_i \in S$, and let $S_{<i} = S \cap \{v_1, \ldots, v_{i-1}\}$ for every $S \in \mathcal{S}_i$. If $S_{<i} \not\subseteq N_G(v_i)$ then S is not a clique and it is discarded. Therefore, once the algorithm has terminated, subsets in $\Xi^{(n)}$ are exactly the minimal separators of H that are cliques of G. By Lemma 1, these are exactly the clique-minimal separators of G. Hence the above algorithm is correct.

Let us focus on the time complexity. Assume for ease of computation that we maintain an "incidence graph" $\mathcal{I}_\mathcal{S}$: such that for every $1 \leq i \leq n$, the vertex set of $\mathcal{I}_\mathcal{S}$ is $V \cup \Xi^{(i-1)}$ and there is an edge between every vertex $v \in V$ and every separator $S \in \Xi^{(i-1)}$ containing v. Note that $\mathcal{I}_\mathcal{S}$ can be constructed at the initialization step (when $\Xi^{(0)} = \Xi$) in time $\mathcal{O}(|V| + \sum_{S \in \Xi} |S|)$, that is in $\mathcal{O}(n + m + f)$-time, and so, in $\mathcal{O}(T(G))$-time. Furthermore, for every $1 \leq i \leq n$ the separators in $\mathcal{S}_i$ are exactly the neighbours of vertex v_i in $\mathcal{I}_\mathcal{S}$, hence it takes $\mathcal{O}(|\mathcal{S}_i|)$-time to access to each of the separators in $\mathcal{S}_i$. Discarding a separator $S \in \mathcal{S}_i$ is equivalent to deleting the vertex corresponding to S in $\mathcal{I}_\mathcal{S}$, which can be done in $\mathcal{O}(|S|)$-time. Overall, these two types of operations (accessing and discarding) take $\mathcal{O}(\sum_{i=1}^{n} |\mathcal{S}_i| + \sum_{S \in \Xi} |S|)$-time, that is in $\mathcal{O}(\sum_{S \in \Xi} |S|) = \mathcal{O}(T(G))$-time.

Finally, deciding whether $S_{<i} \not\subseteq N_G(v_i)$ for every $S \in \mathcal{S}_i$ takes time $\mathcal{O}(|N_G(v_i)| + \sum_{S \in \mathcal{S}_i} |S_{<i}|)$. Furthermore, since the vertices are considered sequentially, we have that $S_{<i}$ is a clique for every $S \in \mathcal{S}_i$ (or else, S would have been discarded at some step $j < i$ of the algorithm). This implies that $\sum_{i | S \in \mathcal{S}_i} |S_{<i}| \leq \sum_{j=1}^{\omega} j = \omega(\omega + 1)/2 = \mathcal{O}(\omega^2)$ for every $S \in \Xi$. Hence, since H is triangulated, and so, $|\Xi| \leq n$, we have:

$$\sum_{i=1}^{n} \sum_{S \in \mathcal{S}_i} |S_{<i}| = \sum_{S \in \Xi} \sum_{i | S \in \mathcal{S}_i} |S_{<i}| = \mathcal{O}(\omega^2 n).$$

□

6 Faster computation of clique minimal separator decomposition

In Section 5, we proved that if a minimal triangulation of a graph G can be computed in time $T(G)$ then the clique-minimal separators of G can be computed in $\mathcal{O}(T(G) + \min\{n^\alpha, \omega^2 n\})$-time. We now prove that a clique-minimal separator decomposition of G can be computed within the same time bounds.

Theorem 9. *Let $G = (V, E)$. Suppose that a minimal triangulation of G can be computed in time $T(G)$. Then, the clique-minimal separator decomposition of G can be computed in $\mathcal{O}(T(G) + \min\{n^\alpha, \omega^2 n\})$-time.*

Proof. Let $H = (V, E \cup F)$ be a minimal triangulation of G, with $f = |F|$ fill edges. By the hypothesis it can be computed in time $T(G)$. Furthermore, the clique-minimal separator decomposition of G can be computed in $\mathcal{O}(n + m + f) = \mathcal{O}(T(G))$-time if H and the clique-minimal separators of G are given [6]. By Propositions 7 and 8, the clique-minimal separators of G can be computed in $\mathcal{O}(T(G) + \min\{n^\alpha, \omega^2 n\})$-time. So, overall it takes $\mathcal{O}(T(G) + \min\{n^\alpha, \omega^2 n\})$-time to compute the clique-minimal separator decomposition of G. □

On the combinatorial side, our approach for computing the clique-minimal separator decomposition (Theorem 9) is at least as good as the state-of-the-art $\mathcal{O}(nm)$-time algorithm. Indeed, for any graph G, a minimal triangulation of G can be computed in time $T(G) = \mathcal{O}(nm)$ [39]. Furthermore if G has clique-number ω then it has number of edges $m \geq \omega(\omega - 1)/2 = \Omega(\omega^2)$.

Corollary 10. *The clique-minimal separator decomposition of a given graph $G = (V, E)$ can be computed in $\mathcal{O}(n^\alpha \log n) = \tilde{\mathcal{O}}(n^{2.3729})$-time.*

Proof. Since a minimal triangulation of a graph G can be computed in $\mathcal{O}(n^\alpha \log n)$-time [30], the result follows from Theorem 9 by replacing $T(G)$ with $\mathcal{O}(n^\alpha \log n)$. □

6.1 Applications

By Theorem 9, the clique-minimal separator decomposition of a given graph $G = (V, E)$ can be computed in quasi linear-time if `i)` G has bounded clique-number and `ii)` a minimal triangulation of G can be computed efficiently. Below, we list a few graph classes for which it is the case.

- A graph G has *treewidth* at most k if there exists a triangulation of G with clique-number at most $k + 1$. See [9] for a survey on this class of graphs. Note that the clique-number ω of G is a lower-bound on its treewidth. Furthermore, if G has treewidth k then a minimal triangulation of G can be computed in $\mathcal{O}(k^7 \cdot n \log n)$-time [25]. Therefore, by Theorem 9 the clique-minimal separator decomposition of bounded treewidth graphs can be computed in $\mathcal{O}(n \log n)$-time.

- A graph G is *planar* if it can be drawn in the Euclidean plane so that edges may only intersect at their endpoints. We refer to [37] for a survey on this class of graphs. In particular, by Kuratowski Theorem G is planar if and only if G is $\{K_{3,3}, K_5\}$-minor-free. So, any planar graph G has bounded clique-number $\omega \leq 4$. Furthermore, if G is planar then a minimal triangulation of G can be computed in $\mathcal{O}(n)$-time [19]. As a result, by Theorem 9 the clique-minimal separator decomposition of planar graphs can be computed in linear-time.

- Finally, let us consider bounded-degree graphs. Indeed, for every graph G, $\omega \leq \Delta + 1$ with ω and Δ being respectively the clique-number and the maximum degree of G. Therefore, bounded-degree graphs have bounded clique-number. Furthermore, if G has maximum degree Δ then a minimal triangulation of G can be computed in $\mathcal{O}(n \cdot (\Delta^3 + \alpha(n)))$-time where $\alpha(n)$ here denotes the inverse of Ackermann's function [20]. Hence by Theorem 9 the clique-minimal separator decomposition of bounded-degree graphs can be computed in $\mathcal{O}(n \cdot \alpha(n))$-time.

7 Conclusion

By Corollary 10 the time complexity of computing the clique minimal separator decomposition of an n-vertex graph G is $\tilde{\mathcal{O}}(n^{\alpha}) = \tilde{\mathcal{O}}(n^{2.3729})$. It is unlikely that the problem can be solved in $o(n^{\alpha})$-time by Theorem 6 (recall that the two problems of TRIANGLE DETECTION and MATRIX MULTIPLICATION are assumed to be equivalent [43, 42]).

Finally, we proved in Theorem 9 that for every graph G with bounded clique-number ω, the clique minimal separator decomposition of G can be computed in $\mathcal{O}(T(G)+\omega^2 n)$-time where $T(G)$ here denotes the time needed to compute a minimal triangulation of G. We conjecture that in fact, it can be computed in $\mathcal{O}(\omega^{\mathcal{O}(1)}(n+m))$-time. More precisely, can any graph be decomposed by means of clique separators in, say $\mathcal{O}(\omega^2 n + m)$-time ?

Acknowledgments

We wish to thank the referees for their careful reading of the first version of this manuscript, and their useful comments. Their remarks and suggestions have improved the presentation of this paper significantly.

References

[1] M. Abu-Ata and F. F. Dragan. Metric tree-like structures in real-world networks: an empirical study. *Networks*, 67(1):49–68, 2016.

[2] A. Berry, A. Brandstädt, V. Giakoumakis, and F. Maffray. Efficiently decomposing, recognizing and triangulating hole-free graphs without diamonds. *Discrete Applied Mathematics*, 184:50–61, 2015.

[3] A. Berry, R. Pogorelcnik, and A. Sigayret. Vertical decomposition of a lattice using clique separators. In *Proceedings of The Eighth International Conference on Concept Lattices and Their Applications, Nancy, France, October 17-20, 2011*, pages 15–29, 2011.

[4] A. Berry, R. Pogorelcnik, and G. Simonet. Efficient clique decomposition of a graph into its atom graph. Technical Report hal-00678702, HAL, Mar. 2010.

[5] A. Berry, R. Pogorelcnik, and G. Simonet. An introduction to clique minimal separator decomposition. *Algorithms*, 3(2):197–215, 2010.

[6] A. Berry, R. Pogorelcnik, and G. Simonet. Organizing the atoms of the clique separator decomposition into an atom tree. *Discrete Applied Mathematics*, 177:1–13, 2014.

[7] A. Berry and A. Wagler. Triangulation and clique separator decomposition of claw-free graphs. In *International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 7–21. Springer, 2012.

[8] J. Blair and B. Peyton. An introduction to chordal graphs and clique trees. In *Graph theory and sparse matrix computation*, pages 1–29. Springer, 1993.

[9] H. L. Bodlaender. Treewidth: Characterizations, applications, and computations. In *WG 2006, Bergen, Norway*, pages 1–14, 2006.

[10] H. L. Bodlaender and A. M. C. A. Koster. Safe separators for treewidth. *Discrete Mathematics*, 306(3):337–350, 2006.

[11] J. A. Bondy and U. S. R. Murty. *Graph theory*. Grad. Texts in Math., 2008.

[12] M. Borassi, P. Crescenzi, and M. Habib. Into the square: On the complexity of some quadratic-time solvable problems. *Electronic Notes in Theoretical Computer Science*, 322:51–67, 2016.

[13] A. Brandstädt, F. F. Dragan, V. Chepoi, and V. Voloshin. Dually chordal graphs. *SIAM Journal on Discrete Mathematics*, 11(3):437–455, 1998.

[14] A. Brandstädt and R. Mosca. Weighted efficient domination for P_5-free and P_6-free graphs. *SIAM Journal on Discrete Mathematics*, 30(4):2288–2303, 2016.

[15] M. Changat and J. Mathew. On triangle path convexity in graphs. *Discrete Mathematics*, 206(1-3):91–95, 1999.

[16] N. Cohen, D. Coudert, G. Ducoffe, and A. Lancin. Applying clique-decomposition for computing Gromov hyperbolicity. *Theoretical Computer Science*, 2017.

[17] N. Cohen, D. Coudert, and A. Lancin. On computing the Gromov hyperbolicity. *ACM Journal on Experimental Algorithmics*, 20(1):18, 2015.

[18] P. Crescenzi, R. Grossi, M. Habib, L. Lanzi, and A. Marino. On computing the diameter of real-world undirected graphs. *Theoretical Computer Science*, 514:84–95, 2013.

[19] E. Dahlhaus. Minimal elimination of planar graphs. In *Scandinavian Workshop on Algorithm Theory*, pages 210–221. Springer, 1998.

[20] E. Dahlhaus. Minimal elimination ordering for graphs of bounded degree. *Discrete applied mathematics*, 116(1):127–143, 2002.

[21] E. Dahlhaus, M. Karpinski, and N. B. Novick. Fast parallel algorithms for the clique separator decomposition. In *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms, 22-24 January 1990, San Francisco, California.*, pages 244–251, 1990.

[22] M. Didi Biha, B. Kaba, M.-J. Meurs, and E. SanJuan. Graph decomposition approaches for terminology graphs. In *Mexican International Conference on Artificial Intelligence*, pages 883–893. Springer, 2007.

[23] G. Dirac. On rigid circuit graphs. In *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, volume 25, pages 71–76. Springer, 1961.

[24] Y. Dourisboure and C. Gavoille. Tree-decompositions with bags of small diameter. *Discrete Mathematics*, 307(16):2008–2029, 2007.

[25] F. V. Fomin, D. Lokshtanov, M. Pilipczuk, S. Saurabh, and M. Wrochna. Fully polynomial-time parameterized computations for graphs and matrices of low treewidth. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1419–1432. SIAM, 2017.

[26] F. Gavril. The intersection graphs of subtrees in trees are exactly the chordal graphs. *Journal of Combinatorial Theory, Series B*, 16(1):47–56, 1974.

[27] F. Gavril. Algorithms on clique separable graphs. *Discrete Mathematics*, 19(2):159–165, 1977.

[28] M. C. Golumbic and R. E. Jamison. The edge intersection graphs of paths in a tree. *Journal of Combinatorial Theory, Series B*, 38(1):8–22, 1985.

[29] P. Heggernes. Minimal triangulations of graphs: A survey. *Discrete Mathematics*, 306(3):297–317, 2006.

[30] P. Heggernes, J. A. Telle, and Y. Villanger. Computing minimal triangulations in time $O(n^{\alpha} \log n) = o(n^{2.376})$. *SIAM J. Discrete Math.*, 19(4):900–913, 2005.

[31] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? In *Foundations of Computer Science, 1998. Proceedings. 39th Annual Symposium on*, pages 653–662. IEEE, 1998.

[32] B. Kaba, N. Pinet, G. Lelandais, A. Sigayret, and A. Berry. Clustering gene expression data using graph separators. *Silico Biology*, 7(4, 5):433–452, 2007.

[33] D. Kratsch and J. Spinrad. Between $O(nm)$ and $O(n^{\alpha})$. *SIAM Journal on Computing*, 36(2):310–325, 2006.

[34] D. Kratsch and J. Spinrad. Minimal fill in $o(n^{2.69})$ time. *Discrete mathematics*, 306(3):366–371, 2006.

[35] F. Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th international symposium on symbolic and algebraic computation*, pages 296–303. ACM, 2014.

[36] H.-G. Leimer. Optimal decomposition by clique separators. *Discrete mathematics*, 113(1-3):99–123, 1993.

[37] B. Mohar and C. Thomassen. *Graphs on surfaces*, volume 2. Johns Hopkins University Press, 2001.

[38] D. J. Rose. Triangulated graphs and the elimination process. *Journal of Mathematical Analysis and Applications*, 32(3):597–609, 1970.

[39] D. J. Rose, R. E. Tarjan, and G. S. Lueker. Algorithmic aspects of vertex elimination on graphs. *SIAM Journal on computing*, 5(2):266–283, 1976.

[40] R. E. Tarjan. Decomposition by clique separators. *Discrete Mathematics*, 55(2):221 – 232, 1985.

[41] R. E. Tarjan and M. Yannakakis. Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal on computing*, 13(3):566–579, 1984.

[42] V. Vassilevska Williams, J. R. Wang, R. Williams, and H. Yu. Finding four-node subgraphs in triangle time. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1671–1680. SIAM, 2015.

[43] V. Vassilevska Williams and R. Williams. Subcubic equivalences between path, matrix and triangle problems. In *51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 645–654. IEEE, 2010.

[44] S. H. Whitesides. An algorithm for finding clique cut-sets. *Information Processing Letters*, 12(1):31–32, 1981.

[45] H. Yaghi and H. Krim. Probabilistic graph matching by canonical decomposition. In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pages 2368–2371. IEEE, 2008.