Clique-decomposition revisited

David Coudert, Guillaume Ducoffe

Clique-decomposition revisited*

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

[1]Inria, France
[2]Univ. Nice Sophia Antipolis, CNRS, I3S, UMR 7271, 06900 Sophia Antipolis, France

Abstract

The decomposition of graphs by clique-minimal separators is a common algorithmic tool, first introduced by Tarjan. Since it allows to cut a graph into smaller pieces, it can be applied to preprocess the graphs in the computation of many optimization problems. However, the best known clique-decomposition algorithms have respective $\mathcal{O}(nm)$-time and $\mathcal{O}(n^{2.69})$-time complexity, that is prohibitive for large graphs. Here we prove that for every graph G, the decomposition can be computed in $\mathcal{O}\left(T(G) + \min\{n^{2.3729},\ \omega^2 n\}\right)$-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 clique-decomposed in $\tilde{\mathcal{O}}(n^{2.3729})$-time. Based on prior work from Kratsch et al., we prove in addition that computing the clique-decomposition is at least as hard as triangle detection. Therefore, the existence of any $o(n^{2.3729})$-time clique-decomposition algorithm 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 clique-decomposed in linear or quasi-linear time.

Keywords: clique-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*, and the decomposition process is called *clique-decomposition*. We refer to [6] for a survey. The clique-decomposition has been introduced by Tarjan in [32], where it is studied for its algorithmic applications. Indeed, it is often the case that hard problems on graphs (theoretically or in practice) can be solved on each subgraph of the clique-decomposition separately. See for instance [4, 9, 14, 15, 18, 19, 26, 34]. In particular, there are NP-hard problems that can be solved on graphs when the subgraphs obtained with the clique-decomposition are "simple enough" w.r.t. the problem. This was first noted by Gavril in [22] for the so-called clique-separable graphs. Other classes of graphs with a "simple" clique-decomposition comprise the chordal graphs (that can be clique-decomposed into complete subgraphs), the EPT graphs [23] and the P_6-free graphs [12, 13]. 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, 15].

With the exception of an $\mathcal{O}(n^{2.69})$-time algorithm in [28], all the best known algorithms for computing clique-decomposition have an $\mathcal{O}(nm)$-time complexity [2, 30, 32], that is cubic for dense graphs. Therefore,

*This work is partially supported by ANR project Stint under reference ANR-13-BS02-0007 and ANR program "Investments for the Future" under reference ANR-11-LABX-0031-01.

it becomes too prohibitive to run them on large graphs with thousands of vertices and sometimes billions of edges. Following a recent trend in algorithmic [11], we here investigate on the optimal time for computing the clique-decomposition.

Related work. To the best of our knowledge, the time complexity of clique-decomposition has received little attention in the literature. We are only aware of a recent article [7] introducing a generic framework to compute the clique-decomposition of graphs. This framework applies to all the best-known algorithms to compute the clique-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 graphs (using the minimal triangulation);
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 [24] for a survey). So far, the best-known algorithm to compute a minimal triangulation of a graph has an $\tilde{\mathcal{O}}(n^{2.3729})$-time complexity. Note that it is less than $\mathcal{O}(n^{2.69})$, that has been the best-known complexity for computing the clique-decomposition of a graph — until this note.

Furthermore, new clique-decomposition algorithms are proposed in [7] that are provably faster than the classical approach in some cases, that is, they run in $\mathcal{O}(nm_0)$-time for some $m_0 < m$. In order to compare these algorithms with our work, let us note that the authors in [7] claim that bounded-treewidth graphs can be clique-decomposed in *quadratic-time*, whereas we will show that it can be done in quasi *linear-time*. Fast (quadratic-time) algorithms to compute the clique-decomposition can also be found in [3, 8] for some specific graph classes, but the latter algorithms deeply rely upon the structural properties of these graphs.

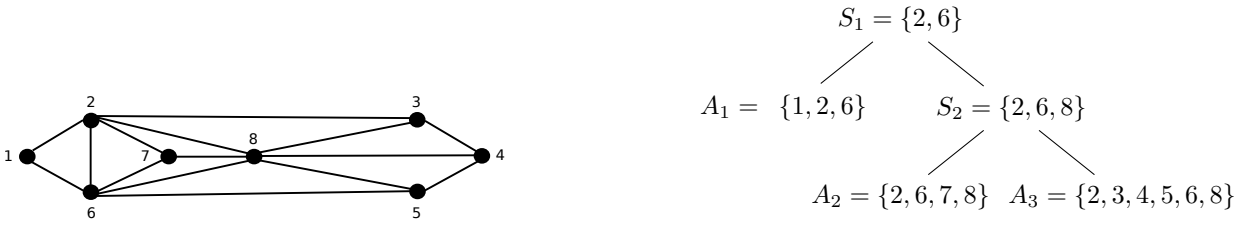
Closest to our work are two papers from Kratsch and Spinrad [27, 28]. In [28], they describe what has been, until this note, the best-known algorithm to compute the clique-decomposition. The latter algorithm has running time $\mathcal{O}(n^{2.69})$, that follows from an algorithm to compute a minimal triangulation of the graph within the same time bounds. We will generalize their result in our work, proving that the clique-decomposition can be computed in $\mathcal{O}(n^{2.3729})$-time if *any* minimal triangulation of the graph is given[1]. Furthermore, lower-bounds on the complexity of computing the clique-decomposition can be deduced from some results in [27]. 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-decomposition algorithms.

Our contributions. On the negative side, we first prove a lower-bound on the complexity of computing the clique-decomposition. More precisely, we will build upon a result in [27] in order to prove that clique-decomposition is at least as hard as triangle detection (Theorem 4).

We next focus on the two last steps of clique-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 $\mathcal{O}(T(G) + \min\{n^{2.3729}, \omega^2 n\})$-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 simple algorithms that respectively run in $\mathcal{O}(T(G) + n^{2.3729})$-time (Proposition 5) and in $\mathcal{O}(T(G) + \omega^2 n)$-time (Proposition 6). Furthermore, whereas the first algorithm (in $\mathcal{O}(T(G) + n^{2.3729})$-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 clique-decomposed within the same time bound $\mathcal{O}(T(G) + \min\{n^{2.3729}, \omega^2 n\})$ (Theorem 7). Since a minimal triangulation can be computed in $T(G) = \tilde{\mathcal{O}}(n^{2.3729})$-time for any graph G, our main result implies that any graph can be clique-decomposed in $\tilde{\mathcal{O}}(n^{2.3729})$-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

[1] It seems to us that the techniques in [28] could also be applied to any minimal triangulation. Nonetheless, we will propose a method that is, to our opinion, slightly simpler than theirs.



(a) G (b) Atom tree of G

Figure 1: A connected graph G (Figure 1a), an atom tree of the graph (Figure 1b).

for interesting graph classes such as planar graphs, bounded-treewidth graphs and bounded-degree graphs (see Section 5.1 for details).

Altogether, this is hint that our $\tilde{\mathcal{O}}(n^{2.3729})$-time clique-decomposition algorithm is optimal up to polylogarithmic factors — due to the well-know equivalence between triangle detection and matrix multiplication [33].

Definitions and useful notations are given in Section 2. Last, we will conclude this paper with an open conjecture in Section 6.

2 Definitions and preliminaries

We will use standard graph terminology from [10]. Graphs in this study are finite, simple (hence without 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 $N[S] = N(S) \cup S$.

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.

Clique-decomposition and atom tree. A graph is *prime* if it does not contain any clique-minimal separator. Examples of prime graphs are the complete graph K_n and the cycle graph C_n. The *clique-decomposition* of a graph G is the family of all inclusionwise maximal subsets A_i such that $G[A_i]$ is prime, and it is unique [30]. The subsets A_i are called the *atoms* of G.

Usually, we represent the clique-decomposition with a binary rooted tree, that is called an *atom tree* and is recursively defined as follows (see Figure 1 for an illustration).

- If G is a prime graph then it has a unique atom tree, that is a single node labeled with V.
- Else, an atom 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 an atom tree of $G[V \setminus C]$.

Informally, an atom tree can be seen as the trace of some execution of a clique-decomposition algorithm (*e.g.*, a decomposition ordering). Note that the atom tree of a graph may not be unique. Furthermore, any atom tree has linear-size (defined as the sum of the label cardinalities) $\mathcal{O}(n + m)$ [5].

Lemma 1 ([30]). *Let $G = (V, E)$ and T be an atom 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.*

Since any atom tree has linear-size, we have by Lemma 1 that $\sum_i |A_i| = \mathcal{O}(n+m)$, where the sets A_i denote the atoms of G. In contrast with the above result, we observe that there may be $\Omega(\omega^2 n)$ edges in 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 2).

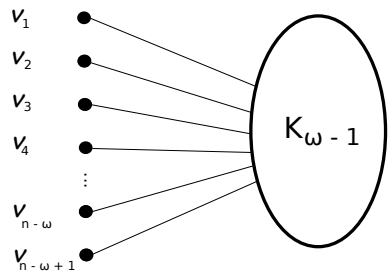


Figure 2: A split graph G with clique-number ω. The atoms of G are the sets $N[v_i]$ for $1 \leq i \leq n-\omega+1$. Hence, there are $\omega(\omega-1)(n-\omega+1)/2$ edges in the subgraphs $G[N[v_i]]$ that are induced by the atoms of G.

Minimal triangulation. A *triangulation* of $G = (V, E)$ is any supergraph $H = (V, E \cup F)$ such that H does not contain any induced cycle of length at least four. 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.

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

Lemma 2 ([6])**.** *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.*

3 Time complexity lower-bound

Let us start proving the hardness of clique-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 3 ([27])**.** *The problem of counting the number of simplicial vertices in a graph with $3n+2$ vertices is at least as hard as determining whether a graph on n vertices has a triangle.*

Theorem 4. *The problem of computing the clique-decomposition of a graph with $3n+2$ vertices is at least as hard as determining whether a graph on n vertices has a triangle.*

Proof. Let $G = (V, E)$ be any graph with $3n+2$ vertices. In order to prove the theorem, by Lemma 3 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-decomposition of G is given (encoded as an atom tree).

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, since $G[N[v]]$ is complete, we have that $G[N[v]]$ is prime, and so, the subset $N[v]$ must be contained in *any* atom A containing v. Furthermore, if it were the case that there exists $u \in A \setminus N[v]$ then the clique $N(v)$ would be a uv-separator, thus contradicting the fact that $G[A]$ is prime. As a result, we have that $A = N[v]$, that proves the claim.

Recall that using an atom tree of G, every atom A_i of G can be written $A_i = C_i \cup S_i$ with S_i being a clique-minimal separator and $S_i \subseteq N(C_i)$. In particular, let $M_i \subseteq C_i$ contain every vertex in the atom that is not contained in any other atom A_j, with $j \neq i$. Note that all the subsets M_i can be computed by visiting the atoms sequentially, which takes $\mathcal{O}(\sum_i |A_i|) = \mathcal{O}(n+m)$-time. Furthermore, we have by the above claim that in order to count the number of simplicial vertices in G, it is sufficient to sum together the cardinalities

$|M_i|$ of the subsets M_i such that the atom A_i is a clique. Here is a way to achieve the goal in linear-time. Since the subsets C_i are pairwise disjoint, let us reorder the vertices in G so that in any adjacency list, it first appears the neighbours in C_1, then those in C_2, and so on. In such case, the atom A_i is a clique if and only if each vertex in C_i has $|A_i| - 1$ neighbours in A_i, that is, $|A_i| - 1$ neighbours that are not contained in any C_j, with $j < i$. The latter can be verified by visiting the subsets C_i sequentially, while removing the vertices in C_i from all adjacency lists at the i^{th} step. Since the adjacency lists have been reordered, it can be done in $\mathcal{O}(m + \sum_i |C_i|) = \mathcal{O}(n + m)$-time. So, overall, finding the atoms A_i that are cliques can be done in $\mathcal{O}(n + m)$-time, which implies that counting the number of simplicial vertices in G can be done within the same time complexity. □

4 Computing the clique-minimal separators

This section is devoted to fast computation of the clique-minimal separators in a graph. We will introduce two methods which both make use of Lemma 2.

Proposition 5. *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 $\mathcal{O}(T(G) + n^{2.3729})$-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 [21], the family Ξ can be computed in $\mathcal{O}(n + m + f) = \mathcal{O}(n^2)$-time. Furthermore, recall that by Lemma 2 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 $1 \leq j \leq n$, 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 $\mathcal{O}(n^{2.3729})$-time by using fast matrix multiplication since $l \leq n$ [29]. Furthermore, for every $1 \leq i \leq n$ and for every $1 \leq j \leq n$, 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 6. *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 [21], the family Ξ can be computed in $\mathcal{O}(n + m + f) = \mathcal{O}(T(G))$-time.

Let $\mathcal{S} = \Xi$. Our aim is to remove separators of H from $\mathcal{S}$ 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 $\mathcal{S}_i \subseteq \mathcal{S}$ contain every $S \in \mathcal{S}$ such that $v_i \in S$. Furthermore, 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 from $\mathcal{S}$. Therefore, once the algorithm has terminated, subsets in $\mathcal{S}$ are exactly the minimal separators of H that are cliques of G. By Lemma 2, 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" $\mathcal{I}_\mathcal{S}$: with vertex set $V \cup \mathcal{S}$ and an edge between every vertex $v_i \in V$ and every separator $S \in \mathcal{S}_i$. Note that $\mathcal{I}_\mathcal{S}$ can be constructed at the initialization step (when $\Xi = \mathcal{S}$) in $\mathcal{O}(|V| + \sum_{S \in \Xi} |S|) = \mathcal{O}(n + m + f)$-time, that is $\mathcal{O}(T(G))$. 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 $S \in \mathcal{S}_i$ from $\mathcal{S}$ 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 $\mathcal{O}(\sum_{S \in \Xi} |S|) = \mathcal{O}(T(G))$-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 from $\mathcal{S}$ 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:

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

□

5 Faster computation of clique-decomposition

In Section 4, 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 $\mathcal{O}(T(G) + \min\{n^{2.3729}, \omega^2 n\})$-time. We now prove that an atom tree of G can be computed within the same time bounds.

Theorem 7. *Let $G = (V, E)$. Suppose that a minimal triangulation of G can be computed in time $T(G)$. Then, the clique-decomposition of G can be computed in $\mathcal{O}(T(G) + \min\{n^{2.3729}, \omega^2 n\})$-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-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 [7]. By Propositions 5 and 6, the clique-minimal separators of G can be computed in $\mathcal{O}(T(G) + \min\{n^{2.3729}, \omega^2 n\})$-time. So, overall it takes $\mathcal{O}(T(G) + \min\{n^{2.3729}, \omega^2 n\})$-time to compute the clique-decomposition of G. □

On the combinatorial side, our approach for computing the clique-decomposition (Theorem 7) 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 $T(G) = \mathcal{O}(nm)$ [31]. Furthermore if G has clique-number ω then it has number of edges $m \geq \omega(\omega-1)/2 = \Omega(\omega^2)$.

Corollary 8. *The clique-decomposition of a graph G can be computed in $\tilde{\mathcal{O}}(n^{2.3729})$-time.*

Proof. Since a minimal triangulation of a graph G can be computed in $\tilde{\mathcal{O}}(n^{2.3729})$-time [25], the result follows from Theorem 7 by replacing $T(G)$ with $\tilde{\mathcal{O}}(n^{2.3729})$. □

5.1 Applications

By Theorem 7, the clique-decomposition of a graph G 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 *tree-width* at most k if there exists a triangulation of G with clique-number at most k. Note that the clique-number ω of G is a lower-bound on its tree-width. Furthermore, if G has tree-width k then a minimal triangulation of G can be computed in $\mathcal{O}(k^7 \cdot n \log n)$-time [20]. Therefore, by Theorem 7 the clique-decomposition of bounded tree-width 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. By Kuratowski Theorem, G is planar if and only if G is $\{K_{3,3}, K_5\}$-minor-free. So, a 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 [16]. As a result, by Theorem 7 the clique-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 Ackerman's function [17]. Hence by Theorem 7 the clique-decomposition of bounded-degree graphs can be computed in $\mathcal{O}(n \cdot \alpha(n))$-time.

6 Conclusion

By Corollary 8, the time complexity of computing the clique-decomposition of an n-vertex graph G is $\tilde{\mathcal{O}}(n^{2.3759})$. It is unlikely that the problem can be solved in $o(n^{2.3759})$-time by Theorem 4 (recall that the two problems of triangle detection and matrix multiplication are equivalent [33]).

Finally, we proved in Theorem 7 that for every graph G with bounded clique-number ω, the clique-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 $\mathcal{O}(\omega^2 n)$-time.

References

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

[2] A. Berry and J.-P. Bordat. Decomposition by clique minimal separators. Technical Report 97213, 1999.

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

[4] 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.

[5] A. Berry, R. Pogorelcnik, and G. Simonet. Efficient clique decomposition of a graph into its atom graph. Technical Report RR-10-07, Mar. 2010.

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

[7] 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.

[8] A. Berry and A. Wagler. Triangulation and clique separator decomposition of claw-free graphs. In M. C. Golumbic, M. Stern, A. Levy, and G. Morgenstern, editors, *Graph-Theoretic Concepts in Computer Science: 38th International Workshop, WG 2012, Jerusalem, Israel, June 26-28, 2012, Revised Selcted Papers*, pages 7–21. Springer Berlin Heidelberg, 2012.

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

[10] J. Bondy and U. Murty. *Graph theory*. Berlin: Springer, 2008.

[11] M. Borassi, P. Crescenzi, and M. Habib. Into the square - on the complexity of quadratic-time solvable problems. *arXiv preprint arXiv:1407.4972*, 2014.

[12] A. Brandstädt and C. Hoàng. On clique separators, nearly chordal graphs, and the maximum weight stable set problem. In *Integer Programming and Combinatorial Optimization*, pages 265–275. Springer, 2005.

[13] A. Brandstädt and R. Mosca. Weighted efficient domination for P_6-free graphs in polynomial time. *arXiv preprint arXiv:1508.07733*, 2015.

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

[15] N. Cohen, D. Coudert, G. Ducoffe, and A. Lancin. Applying clique-decomposition for computing Gromov hyperbolicity. Research Report RR-8535, Inria, June 2014.

[16] E. Dahlhaus. Minimal elimination of planar graphs. In *Algorithm Theory - SWAT '98, 6th Scandinavian Workshop on Algorithm Theory, Stockholm, Sweden, July, 8-10, 1998, Proceedings*, pages 210–221, 1998.

[17] E. Dahlhaus. Minimal elimination ordering for graphs of bounded degree. *Discrete Applied Mathematics*, 116(1-2):127–143, 2002.

[18] M. Didi Biha, B. Kaba, M.-J. Meurs, and E. SanJuan. Graph decomposition approaches for terminology graphs. In *MICAI 2007: Advances in Artificial Intelligence*, volume 4827 of *Lecture Notes in Computer Science*, pages 883–893. Springer, 2007.

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

[20] F. Fomin, D. Lokshtanov, M. Pilipczuk, S. Saurabh, and W. Wrochna. Fully polynomial-time parameterized computations for graphs and matrices of low treewidth. Technical Report abs/1511.01379, arXiv, 2015.

[21] F. Gavril. Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. *SIAM Journal on Computing*, 1(2):180–187, 1972.

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

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

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

[25] P. Heggernes, J. Telle, and Y. Villanger. Computing minimal triangulations in time $O(n^\alpha \log n) = o(n^{2.376})$. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 907–916. Society for Industrial and Applied Mathematics, 2005.

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

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

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

[29] F. Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation*, ISSAC '14, pages 296–303, New York, NY, USA, 2014. ACM.

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

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

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

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

[34] 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.