Symposium on Theoretical Aspects of Computer Science 2010 (Nancy, France), pp. 383-394
www.stacs-conf.org

FINDING INDUCED SUBGRAPHS VIA MINIMAL TRIANGULATIONS

FEDOR V. FOMIN [1] AND YNGVE VILLANGER [1]

[1] Department of Informatics,
University of Bergen, Norway
E-mail address: {fedor.fomin|yngve.villanger}@ii.uib.no

Abstract. Potential maximal cliques and minimal separators are combinatorial objects which were introduced and studied in the realm of minimal triangulation problems including Minimum Fill-in and Treewidth. We discover unexpected applications of these notions to the field of moderate exponential algorithms. In particular, we show that given an n-vertex graph G together with its set of potential maximal cliques, and an integer t, it is possible in time the number of potential maximal cliques times $O(n^{O(t)})$ to find a maximum induced subgraph of treewidth t in G and for a given graph F of treewidth t, to decide if G contains an induced subgraph isomorphic to F. Combined with an improved algorithm enumerating all potential maximal cliques in time $O(1.734601^n)$, this yields that both the problems are solvable in time $1.734601^n * n^{O(t)}$.

1. Introduction

One of the most fundamental problems in Graph Algorithms is, for a given graph $G = (V, E)$, to find a maximum or minimum subset S of V that satisfies some property Π. For example, when S is required to be a maximum set of pairwise adjacent vertices this is the Maximum Clique problem. When S is required to be a maximum set of pairwise non-adjacent vertices this is the Maximum Independent Set problem. Its complement, the Minimum Vertex Cover problem, is to find a minimum set S such that the graph $G \setminus S$ is an independent set. Another examples are Maximum Induced Forest, where one is seeking for a set of vertices inducing a forest of maximum size, or its complement Minimal Feedback Vertex Set which is to remove the minimum number of vertices to destroy all cycles.

All these examples are special cases of the problem, where one seeks a maximum subset of vertices that induces a subgraph of G from some given graph class $\mathcal{C}$. If G is an n-vertex graph, and recognition of graphs from $\mathcal{C}$ can be done in polynomial time, then the trivial brute force algorithm solves the problem in time $2^n n^{\mathcal{O}(1)}$. One of the crucial questions in

1998 ACM Subject Classification: Algorithm Analysis, Combinatorics, Data Structures, Graph Algorithms, Graph Theory.

Key words and phrases: Bounded treewidth, minimal triangulation, moderately exponential time algorithms.

Partially supported by the Norwegian Research Council.

27th Symposium on Theoretical Aspects of Computer Science, Nancy, 2010
Editors: Jean-Yves Marion, Thomas Schwentick
Leibniz International Proceedings in Informatics (LIPIcs), Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany
Digital Object Identifier: 10.4230/LIPIcs.STACS.2010.2470

the area of moderate exponential algorithms is if the brute force algorithm can be avoided to solve any hard (NP-hard, $\#P$, PSPACE-hard, etc.) problem. So far we are still very far from answering this question. For some problems we know how to avoid the brute force search, and for some problems, like SAT, it is a big open problem in the area. Similar situation is with the problem of finding a maximum induced subgraph from a given class $\mathcal{C}$. For some simple graph classes $\mathcal{C}$ the trivial 2^n-barrier has been broken. The most well studied case is when $\mathcal{C}$ is the class of graphs without edges, or the class of graphs of treewidth 0. In this case, we are looking for an independent set of maximum size. This is the classical NP-hard problem and it is well studied in the realm of moderate exponential algorithms. The classical result of Moon and Moser [19] (see also Miller and Muller [18]) from the 1960s can be easily turned into algorithms finding a maximum independent set in time $3^{n/3}n^{\mathcal{O}(1)}$. Tarjan and Trojanowski [25] gave a $O(2^{n/3})$ time algorithm. There were several non-trivial steps in improving the running time of the algorithm including the work of Jian [17], Robson [23], and Grandoni et al. [11]. A significant amount of research was also devoted to algorithms for the Maximum Independent Set problem on sparse graphs, some examples are [7, 14, 21]. It is easy to show that a simple branching algorithm can compute a maximum induced path or cycle in time $3^{n/3}n^{\mathcal{O}(1)}$. However, breaking the 2^n-barrier even for the case when the class $\mathcal{C}$ is a forest, i.e. the class of graphs of treewidth 1, was an open problem in the area until very recently. The first exact algorithm breaking the trivial 2^n-barrier is due to Razgon [20]. The running time $\mathcal{O}(1.8899^n)$ of the algorithm from [20] was improved in [9, 10] to $\mathcal{O}(1.7548^n)$. All these algorithms for Maximum Independent Set and Maximum Induced Forest are so-called branching algorithms (a variation of Davis-Putnam-style exponential-time backtracking [8]). There is also a relevant work of Gupta et al. [15] who used branching to show that for every fixed r, there are at most c^n r-regular subgraphs for some $c < 2$. For example, for Maximum Induced Matching and Maximum 2-Regular Induced Subgraph, their results yield algorithms solving these problems in time $\mathcal{O}(1.695733^n)$ and $\mathcal{O}(1.7069^n)$, respectively. However, the results of Gupta et al. strongly depend on the regularity of the maximum subgraphs. To our knowledge, prior to our work no algorithms better than the trivial brute-force $\mathcal{O}(2^n)$ were known for more complicated classes $\mathcal{C}$.

In this work we make a step aside the "branching" path and use a completely different approach for problems on finding induced subgraphs. Our approach is based on a tools from the area of minimal triangulations, namely, potential maximal cliques. Minimal triangulations are the result of adding an inclusion minimal set of edges to produce a triangulation (or chordal graph). The study of minimal triangulations dates back to the 1970s and originated from research on sparse matrices and vertex elimination in graphs. Minimal separators are one of the main tools in the study of minimal triangulations. We refer to the survey of Heggernes [16] for more information on triangulations. Potential maximal cliques were defined by Bouchitté and Todinca [5, 6] and were used in different algorithms for computing the treewidth of a graph [12, 13]. A subset of vertices C of a graph G is a *potential maximal clique* if there is a minimal triangulation TG of G such that C is a maximal clique in TG. At first glance it is not clear, what is the relation between potential maximal cliques and induced subgraphs. Our first main result establishes such a relation.

- Let Π_G be the set of potential maximal cliques in G. A maximum induced subgraph of treewidth t in an n-vertex graph G can be found in time $\mathcal{O}(|\Pi_G|\cdot n^{\mathcal{O}(t)})$ (Section 3).

As we already mentioned, the well studied MAXIMUM INDEPENDENT SET (and its dual MINIMUM VERTEX COVER) and MAXIMUM INDUCED FOREST (and MINIMUM FEEDBACK VERTEX SET) are the special cases for $t = 0$ and $t = 1$, respectively. Our second main result shows that

- All potential maximal cliques can be enumerated in time $\mathcal{O}(1.734601^n)$ (Section 5).

Combining both results, we obtain that a maximum induced subgraph of treewidth t in an n-vertex graph G can be found in time $\mathcal{O}(1.734601^n \cdot n^{\mathcal{O}(t)})$. While for $t = 0$ (the case of MAXIMUM INDEPENDENT SET) the existing branching algorithms are much faster than $\mathcal{O}(1.734601^n)$, already for $t = 1$ (the case of MAXIMUM INDUCED FOREST) our algorithm is already faster than the best known branching algorithm [10]. For fixed $t \geq 2$, no algorithm better than the trivial $\mathcal{O}(2^n n^{\mathcal{O}(1)})$ brute force algorithm was known.

With small modifications, our algorithm can be used for other problems involving induced subgraphs. As an example, we show how to solve the induced subgraph isomorphism problem, which is to decide if G contains an induced subgraph isomorphic to a given graph F (Section 4). We show that when the treewidth of F is at most t, then this problem is solvable in time $1.734601^n \cdot n^{\mathcal{O}(t)}$. In particular, when the treewidth of F is $o(n/\log n)$, for example when F is a planar graph, or a graph excluding some fixed graph as a minor, the running time of our algorithm is $1.734601^{n+o(n)}$. Let us note that no algorithm faster than the trivial brute-force algorithm was known even when F is a tree.

Finally, our new algorithm enumerating potential maximal cliques is not only (slightly) faster than the algorithm from [13] and thus by [12], directly implies faster exact algorithm computing the treewidth of a graph. It is also significantly simpler than the previous algorithms and is easy to implement. Due to space limitations, some proofs are omitted. A full version will appear at some later point.

2. Preliminaries

We denote by $G = (V, E)$ a finite, undirected, and simple graph with $|V| = n$ vertices and $|E| = m$ edges. For any nonempty subset $W \subseteq V$, the subgraph of G induced by W is denoted by $G[W]$. For $S \subseteq V$ we often use $G \setminus S$ to denote $G[V \setminus S]$. The *neighborhood* of a vertex v is $N(v) = \{u \in V : \{u, v\} \in E\}$, $N[v] = N(v) \cup \{v\}$, and for a vertex set $S \subseteq V$ we set $N(S) = \bigcup_{v \in S} N(v) \setminus S$, $N[S] = N(S) \cup S$. A *clique* C of a graph G is a subset of V such that all the vertices of C are pairwise adjacent. By $\omega(G)$ we denote the maximum clique-size of a graph G.

A graph H is *chordal* (or *triangulated*) if every cycle of length at least four has a chord, i.e., an edge between two nonconsecutive vertices of the cycle. A *triangulation* of a graph $G = (V, E)$ is a chordal graph $H = (V, E')$ such that $E \subseteq E'$. Graph H is a *minimal triangulation* of G if for every edge set E'' with $E \subseteq E'' \subset E'$, the graph $F = (V, E'')$ is not chordal.

The notion of treewidth is due to Robertson and Seymour [22]. A *tree decomposition* of a graph $G = (V, E)$, denoted by $TD(G)$, is a pair (X, T) in which $T = (V_T, E_T)$ is a tree and $X = \{X_i \mid i \in V_T\}$ is a family of subsets of V, called *bags*, such that

(i) $\bigcup_{i \in V_T} X_i = V$;
(ii) for each edge $e = \{u, v\} \in E$ there exists an $i \in V_T$ such that both u and v belong to X_i;
(iii) for all $v \in V$, the set of nodes $\{i \in V_T \mid v \in X_i\}$ induces a connected subtree of T.

The maximum of $|X_i|-1$, $i \in V_T$, is called the *width* of the tree decomposition. The *treewidth* of a graph G, denoted by $\mathbf{tw}(G)$, is the minimum width taken over all tree decompositions of G.

Theorem 2.1 (folklore)**.** *For any graph G, $\boldsymbol{tw}(G) \leq k$ if and only if there is a triangulation H of G such that $\omega(H) \leq k+1$.*

Let u and v be two non adjacent vertices of a graph $G = (V, E)$. A set of vertices $S \subseteq V$ is a *u, v-separator* if u and v are in different connected components of the graph $G[V \setminus S]$. A connected component C of $G[V \setminus S]$ is a *full* component associated to S if $N(C) = S$. Separator S is a *minimal u, v-separator* of G if no proper subset of S is a u, v-separator. Notice that a minimal separator can be strictly included in another one. We denote by Δ_G the set of all minimal separators of G.

A set of vertices $\Omega \subseteq V$ of a graph G is called a *potential maximal clique* if there is a minimal triangulation H of G such that Ω is a maximal clique of H. We denote by Π_G the set of all potential maximal cliques of G.

For a minimal separator S and a full connected component C of $G \setminus S$, we say that (S, C) is a *block* associated to S. We sometimes use the notation (S, C) to denote the set of vertices $S \cup C$ of the block. It is easy to see that if $X \subseteq V$ corresponds to the set of vertices of a block, then this block (S, C) is unique: indeed, $S = N(V \setminus X)$ and $C = X \setminus S$.

We also need the following result of Bouchitté and Todinca on the structure of potential maximal cliques.

Theorem 2.2 (Bouchitté and Todinca [5])**.** *Let $K \subseteq V$ be a set of vertices of the graph $G = (V, E)$. Let $\mathcal{C}(K) = \{C_1, \ldots, C_p\}$ be the set of connected components of $G \setminus K$ and let $\mathcal{S}(K) = \{S_1, S_2, \ldots, S_p\}$, where $S_i = N(C_i)$, $i \in \{1, 2, \ldots, p\}$, is the set of those vertices of K which are adjacent to at least one vertex of the component C_i. Then K is a potential maximal clique of G if and only if*

1. *$G \setminus K$ has no full component associated to K, and*
2. *the graph on the vertex set K obtained from $G[K]$ by completing each $S_i \in \mathcal{S}(K)$ into a clique is a complete graph.*

Moreover, if K is a potential maximal clique, then $\mathcal{S}(K)$ is the set of minimal separators of G contained in K.

3. Induced subgraph of bounded treewidth

In this section we prove the first result relating the problems of finding an induced subgraph and enumerating potential maximal cliques. The following lemma is crucial for our algorithm.

Lemma 3.1. *Let $F = (V_F, E_F)$ be an induced subgraph of a graph $G = (V_G, E_G)$. Then for every minimal triangulation TF of F, there is a minimal triangulation TG of G such that for every clique K of TG, the intersection $K \cap V_F$ is either empty, or is a clique of TF.*

Now we are ready to proceed with the main result of this section.

Theorem 3.2. *Let G be a graph on n vertices and m edges given together with the set Π_G of its potential maximal cliques and the set Δ_G of its minimal separators. For any integers $0 \leq t, \ell \leq n$, there is an algorithm that checks in time $\mathcal{O}(n^{t+4}m(|\Pi_G| + |\Delta_G|))$ if G contains an ℓ-vertex induced subgraph of treewidth at most t.*

Proof. Let F be an induced subgraph of treewidth at most t. By Lemma 2.1, there is a minimal triangulation TF of F, such that the size of a maximal clique of TF is at most $t+1$. By Lemma 3.1, there is a minimal triangulation TG of G, such that every clique of TG contains at most $t+1$ vertices of F. If we knew such a minimal triangulation TG, dynamic programming over the clique-tree of TG will provide the answer to our question in time $\mathcal{O}(n^{t+3}m)$. However, we are not given such a triangulation a priori. Thus, the computations require multiplicative factor $n|\Pi_G|$.

We start by enumerating all full blocks and sorting them by their sizes. This can be done by enumerating all minimal separators, and checking for each minimal separator S and each of the connected component of $G \setminus S$ if this is a full component or not. By making use of Theorem 5.6, this step can be performed in time $\mathcal{O}(|\Delta_G| \cdot n^3)$. Sorting blocks can be done in $\mathcal{O}(n|\Delta_G|)$ time using a bucket sort.

For a minimal separator S, a full block (S,C), and a potential maximal clique Ω, we call the triple (S,C,Ω) *good* if $S \subseteq \Omega \subseteq C \cup S$. For each full block we also enumerate all good triples that can be obtained from this block as follows. By Theorem 2.2, if a minimal separators S is a subset of a potential maximal clique Ω, then $S = N(C)$ for some connected component C of $G[V \setminus \Omega]$, and thus, the number of minimal separators contained in Ω is at most n. By Theorem 2.2, $G \setminus \Omega$ has no full component associated to Ω, and thus for every minimal separator $S \subseteq \Omega$, we have that $\Omega \setminus S \neq \emptyset$. Therefore, there exists a vertex $u \in \Omega \setminus S$ and thus Ω is a subset of the full block (S,C) such that $u \in C$. But this yields that every potential maximal clique is contained in at most n good triples, and the total number of good triples is at most $n|\Pi_G|$. Computing for every potential maximal clique all good triples containing it, in time $\mathcal{O}(m|\Pi_G|)$ one can create a data structure that for each full block assigns the set of potential maximal cliques that make a good triple with that block.

After preprocessing blocks and creating good triples, we proceed with dynamic programming. The dynamic programming consists of two step. In the first, most technical step, we compute the sizes of maximal subgraphs in full blocks (S,C) subject to the condition that the minimal separator S contains at most $t+1$ vertices of the subgraph. To compute these values we use deep combinatorial results of Bouchitté and Todinca on the structure of potential maximal cliques. In the second step, we go through all minimal separators, and for each separator we glue solutions found at the first step.

Step 1: Processing full blocks. We need to define several functions. For a full block (S,C), and for every subset $W \subseteq S$, $|W| \leq t+1$, and integer $0 \leq \ell \leq n$, $\alpha(\ell,W,S,C) = 1$ if there exits an induced subgraph $F = (V_F, E_F)$ of $G[C \cup W]$ such that $|V_F| = \ell$, $V_F \cap S = W$, and F has a minimal triangulation TF such that $\omega(TF) \leq t+1$ and W is a clique of TF. Otherwise, $\alpha(\ell,W,S,C) = 0$.

For every inclusion minimal block (S,C), we have that $S \cup C$ is a potential maximal clique. Thus for every inclusion minimal block (S,C), and for every set $W \subseteq S \cup C$, $|W| \leq t+1$, we put

$$\alpha(\ell,W,S,C) = \begin{cases} 1, & \text{if } \ell = |W|, \\ 0, & \text{otherwise.} \end{cases}$$

To compute the values of α for larger blocks, we perform dynamic programming over sets of good triples formed by smaller blocks. For every good triple (S,C,Ω), and for every subset $W \subseteq \Omega$, $|W| \leq t+1$, and integer $0 \leq \ell \leq n$, we want to compute an auxiliary function such that $\beta(\ell,W,S,C,\Omega) = 1$ if there exits an induced subgraphs $F = (V_F, E_F)$ of

$G[C \cup W]$ such that $|V_F| = \ell$, $V_F \cap \Omega = W$, and F has a minimal triangulation TF such that $\omega(TF) \leq t+1$, and W is a clique of TF. Otherwise, $\beta(\ell, W, S, C, \Omega) = 0$.

Let us remark that

$$\alpha(\ell, W, S, C) = 1 \Leftrightarrow \exists \text{ good triple } (S, C, \Omega) \text{ and } W \subseteq W' \subseteq \Omega \text{ s.t. } \beta(\ell, W', S, C, \Omega) = 1.$$

Indeed, if $\beta(\ell, W', S, C, \Omega) = 1$, then there is a minimal triangulation TF of an induced subgraph $F = (V_F, E_F)$ of $G[C \cup W]$ such that $|V_F| = \ell$, $\omega(TF) \leq t+1$, and W is a clique of TF, simply because this is true for W' and $W \subseteq W'$. Then $TF[V_F \setminus (W' \setminus W)]$ is the triangulation of $F[V_F \setminus (W' \setminus W)]$ that certifies $\alpha(\ell, W, S, C) = 1$. For the opposite direction the arguments are similar.

We start computing β from inclusion minimal blocks. For every inclusion minimal block (S, C), and for every set $W \subseteq S \cup C$, $|W| \leq t+1$,

$$\beta(\ell, W, S, C, \Omega) = \begin{cases} 1, & \text{if } \ell = |W|, \\ 0, & \text{otherwise.} \end{cases}$$

To compute $\beta(\ell, W, S, C, \Omega)$ we define an auxiliary function γ as follows. Let $\{C_1, \ldots, C_p\}$ be the vertex sets of the connected components of $G[(S \cup C) \setminus \Omega]$. By Theorem 2.2, the sets $S_i = N(C_i)$, $1 \leq i \leq p$, are minimal separators of G, and moreover, $S_i \subset \Omega$ for $1 \leq i \leq p$. The values of function $\gamma(\ell, j, W, S, C, \Omega)$ are in $\{0, 1\}$. For every good triple (S, C, Ω), and for every subset $W \subset \Omega$, $|W| \leq t+1$, and $0 \leq \ell \leq n$, $\gamma(\ell, j, W, S, C, \Omega) = 1$ if and only if there exits an induced subgraph $F = (V_F, E_F)$ of $G[W \cup \bigcup_{i=1}^{j} C_i]$ such that $|V_F| = \ell$, $V_F \cap \Omega = W$, and F has a minimal triangulation TF such that $\omega(TF) \leq t+1$ and W is a clique in TF. Note that $G[W \cup \bigcup_{i=1}^{p} C_i] = G[W \cup C]$, and by definitions of β and γ, we have that

$$\beta(\ell, W, S, C, \Omega) = \gamma(\ell, p, W, S, C, \Omega).$$

Now for every $\ell \geq 0$,

$$\gamma(\ell, 1, W, S, C, \Omega) = \alpha(\ell - |W \setminus S_1|, W \cap S_1, S_1, C_1).$$

For $j > 1$,

$$\gamma(\ell, j, W, S, C, \Omega) = \begin{cases} 1, & \text{if } \gamma(i, j-1, W, S, C, \Omega) = 1 \wedge \alpha(\ell - i + |W \cap S_j|, \\ & W \cap S_j, S_j, C_j) = 1, \text{for some } i, 1 \leq i \leq \ell, \\ 0, & \text{otherwise.} \end{cases}$$

This is because for every ℓ-vertex subgraph $F = (V_F, E_F)$ of $G[C_1 \cup \cdots C_j \cup W]$ with $V_F \cap \Omega = W$, there is $i \leq \ell$ such that i vertices of F are in $C_1 \cup \cdots C_{j-1} \cup W$ and $\ell - i + |W \cap S_j|$ vertices are in $C_j \cap S_j$.

To compute $\gamma(\ell, j, W, S, C, \Omega)$, we find the blocks (S_j, C_j), $1 \leq j \leq p$, in G, which can be done in time $\mathcal{O}(m)$ and read already computed values $\alpha(\ell - i + |W \cap S_j|, W \cap S_j, S_j, C_j)$ and $\gamma(i, j-1, W, S, C, \Omega)$. Similarly, the values of $\alpha(\ell, W, S, C)$ and $\beta(\ell, W, S, C, \Omega)$ are computable in time $\mathcal{O}(m)$ from the values of the smaller blocks and the values of γ. The total running time required to compute the values of all $\alpha(\ell, W, S, C)$ is $\mathcal{O}(m)$ times the number of different 6-tuple $(\ell, i, W, S, C, \Omega)$ plus the time $\mathcal{O}(n^3(|\Delta_G| + |\Pi_G|))$ required for preprocessing step. The number of good triples (S, C, Ω) is at most $n|\Pi_G|$, and the number of subsets W of size at most $t+1$ is $\mathcal{O}(n^{t+1})$. Thus the total running time required to compute all values $\alpha(\ell, W, S, C)$ is

$$\mathcal{O}(mn^{t+4}(|\Pi_G| + |\Delta_G|)).$$

Now everything is prepared to solve the problem on graph G and to conclude the proof. By Lemma 3.1, if F is an induced subgraph of G of treewidth at most t, there exists a minimal separator S of G, such that $|V_F \cap S| \leq t+1$. We go through all minimal separators, and for each minimal separator S, we try to glue solutions obtained during the first step.

Step 2: Gluing pieces together. Let S be a minimal separator and let $\{C_1, \ldots, C_p\}$ be the vertex sets of the connected components of $G[V \setminus S]$. We put $S_i = N(C_i)$. For every subset $W \subseteq S$ of size at most $t+1$, and integer $0 \leq \ell \leq n$, we define $\delta(\ell, j, W, S) = 1$ if there is an induced ℓ-vertex subgraph $F = (V_F, E_F)$ of $G[W \cup \bigcup_{i=1}^{j} C_i]$ which poses a minimal triangulation TF with $\omega(TF) \leq t+1$, and such that $W = V_F \cap S$ is a clique in TF. If no such graph F exists, we put $\delta(\ell, j, W, S) = 0$. By Lemma 3.1, G has an induced ℓ-vertex subgraph of treewidth at most t if and only if $\delta(\ell, p, W, S) = 1$ for some minimal separator S. Thus computing the value δ for all minimal separators is sufficient for deciding if G has an induced subgraph on ℓ vertices of treewidth at most t.

For every $\ell \geq 0$ and $j = 1$, we have that

$$\delta(\ell, 1, W, S) = \alpha(\ell - |W \setminus S_1|, W \cap S_1, S_1, C_1).$$

For $j > 1$,

$$\delta(\ell, j, W, S) = \begin{cases} 1, & \text{if } \delta(i, j-1, W, S) = 1 \wedge \alpha(\ell - i + |W \cap S_j|, W \cap S_j, S_j, C_j) = 1, \\ & \text{for some } 1 \leq i \leq \ell, \\ 0, & \text{otherwise.} \end{cases}$$

Like in the case with γ, the correctness of the formula above follows from the fact, that for every ℓ-vertex subgraph $F = (V_F, E_F)$ of $G[C_1 \cup \cdots C_j \cup W]$ with $V_F \cap S = W$, there is $i \leq \ell$ such that i vertices of F are in $C_1 \cup \cdots C_{j-1} \cup W$ and $\ell - i + |W \cap S_j|$ vertices are in $C_j \cap S_j$.

Concerning the time required to perform this step. Like in above, in time $\mathcal{O}(m)$ we can find the connected components $\{C_1, \ldots, C_p\}$ of $G[V \setminus S]$, and the corresponding full blocks (S_i, C_i). Thus the running of this step is proportional to m times the number of 4-tuples (ℓ, j, W, S), and we conclude that this step of the algorithm can be performed in time $\mathcal{O}(mn^{t+3} \cdot |\Delta_G|)$. ∎

4. Induced subgraph isomorphism

The technique described in the previous section with slight modifications can be applied for many different problems. In this section we give an important example of such modification.

Theorem 4.1. *Let G be an n-vertex graph given together with the set Π_G of its potential maximal cliques and the set Δ_G of its minimal separators. Let F be a graph of treewidth t. There is an algorithm checking if G contains an induced subgraph isomorphic to F in time $\mathcal{O}(n^{\mathcal{O}(t)}(|\Delta_G| + |\Pi_G|))$.*

Proof. The proof of the theorem follows the lines of Theorem 3.2 with modifications that are similar to the well known Bodlaender's algorithm for solving the graph isomorphism problem on graphs of bounded treewidth [4]. We outline only the most important differences of such a modification.

The treewidth of F is at most t, and we use the algorithm of Arnborg et.al. [2] to construct a minimal triangulation TF of F such that $\omega(TF) \leq t+1$. The running time of this algorithm is in $\mathcal{O}(n^{t+2})$. The number of maximal cliques and minimal separators in an n-vertex chordal graph is $\mathcal{O}(n)$ [24]. Thus the number of full blocks and good triples in TF is $\mathcal{O}(n)$. We list and keep all these blocks and triples. This can be done in polynomial time.

As in the proof of Theorem 3.2, we perform two steps of dynamic programming. First we run computations over full blocks of G, and then use computed values to glue solutions in minimal separators.

For every full block (S, C) of G, every full block (S_F, C_F) of TF, every subset $W \subseteq S$, where $|W| = |S_F| \leq t+1$, and every bijection $\mu\colon S_F \to W$, we define the value $\alpha(S_F, C_F, W, \mu, S, C)$ to be equal to 1 if there is an injection $\lambda\colon S_F \cup C_F \to W \cup C$ such that $F[S_F \cup C_F]$ is isomorphic to $G[\lambda(S_F \cup C_F)]$, and for every $v \in S_F$, $\lambda(v) = \mu(v)$. Otherwise, we put $\alpha(S_F, C_F, W, \mu, S, C) = 0$. In other words, α is equal to 1, when $G[W \cup C]$ contains a subgraph isomorphic to $F[S_F \cup C_F]$, and moreover, the restriction of the corresponding isomorphic mapping on S_F is exactly μ.

As in Theorem 3.2, to compute $\alpha(S_F, C_F, W, \mu, S, C)$ we run through good triples (S, C, Ω), where Ω is a potential maximal clique, $S \subseteq \Omega \subseteq S \cup C$. For every good triple (S, C, Ω) of G and every good triple (S_F, C_F, Ω_F) of F, for every subset $W \subseteq \Omega$, such that $|W| = |\Omega_F| \leq t+1$, and every bijection $\mu\colon \Omega_F \to W$, we define the function $\beta(S_F, C_F, \Omega_F, W, \mu, S, C, \Omega) \in \{0, 1\}$. We put $\beta(S_F, C_F, \Omega_F, W, \mu, S, C, \Omega) = 1$ if and only if there is an injection $\lambda\colon S_F \cup C_F \to W \cup C$ such that $F[S_F \cup C_F]$ is isomorphic to $G[\lambda(S_F \cup C_F)]$, and for every $v \in \Omega_F$, $\lambda(v) = \mu(v)$. Following the lines of Theorem 3.2, it is possible to show that $\alpha(S_F, C_F, W, \mu, S, C) = 1$ if and only if there exist

- Good triple (S, C, Ω) of G and good triple (S_F, C_F, Ω_F) of F;
- Set W', $W \subseteq W' \subseteq \Omega$;
- Bijection $\mu'\colon \Omega_F \to W'$, $\mu'_{|W}(\cdot) = \mu(\cdot)$

such that $\beta(S_F, C_F, \Omega_F, W', \mu', S, C, \Omega) = 1$.

The main difference with the proof of Theorem 3.2 is in the way we compute β. We compute the values of $\beta(S_F, C_F, \Omega_F, W, \mu, S, C, \Omega)$ from the values of smaller blocks contained in $G[S \setminus \Omega]$. This is done by reducing to the problem of finding a maximum matching in some auxiliary bipartite graph. This step is quite similar to the algorithm of Bodlaender [4] for isomorphism of bounded treewidth graphs. Let $F_1, F_2, \ldots, F_p$ be the connected components of the graph $F[C_F \setminus \Omega_F]$. Then the sets $Q_i = N_F(F_i) \subseteq \Omega_F$ are minimal separators and pairs (F_i, Q_i), $1 \leq i \leq p$, are blocks in F. Similarly, for the connected components $G_1, G_2, \ldots, G_q$ of $G[C \setminus \Omega]$, we put $S_i = N_G(G_i)$, and define blocks (G_i, S_i), $1 \leq i \leq q$. We construct an auxiliary bipartite graph B with bipartition $X = \{x_1, x_2, \ldots, x_p\}$ and $Y = \{y_1, y_2, \ldots, y_q\}$. There is an edge $\{x_i, y_j\}$ in B if and only if there is an isomorphic mapping of block (F_i, Q_i) to block (G_j, Q_j) which agrees with μ. But then to decide if blocks (F_i, Q_i) can be mapped to blocked (G_i, S_i) is equivalent to deciding if B has a matching of size p. More formally, $\{x_i, y_j\}$ is an edge in B if and only if there is an injection $\lambda\colon F_i \cup Q_i \to G_j \cup S_j$ such that $F[F_i \cup Q_i]$ is isomorphic to $G[\lambda(F_i \cup Q_i)]$, and for every $v \in Q_i$, $\lambda(v) = \mu(v)$. But such an injection λ exists if and only if $\alpha(F_i, Q_i, W', \mu', G_j, S_j) = 1$, where $W' = \mu(Q_i)$ and $\mu'(\cdot) = \mu_{|Q_i}(\cdot)$. Therefore, to compute the value of β, it is sufficient to run through the already computed values of α of smaller blocks, construct an auxiliary graph and find if this graph contains a matching of specific size.

Finally, as in Theorem 3.2, after all values α are computed, we run through all minimal separators of G and for each minimal separator S, we try to glue solutions obtained for all blocks attached to this separator. Here again, we need only the values of α computed for all such blocks and reduce the problem to bipartite matchings. The running time of the algorithm is up to multiplicative polynomial factor equal to the number of states of the dynamic programming. To compute the values of α and β, we run through all potential maximal cliques, blocks, and good triples of TF and G, which is $n^{\mathcal{O}(1)}|\Pi_G|$. For every pair of blocks or triples, we run through all subsets W of size at most $t+1$, which is $\mathcal{O}(n^{t+1})$, and through all mappings between sets of cardinality at most $t+1$, which is $\mathcal{O}((t+1)^{t+1})$. Finally, we run through all minimal separators. Thus the total running time of the algorithm is $\mathcal{O}(n^{\mathcal{O}(t)}(|\Delta_G|+|\Pi_G|))$. The proof of the correctness of the algorithm follows the lines of Theorem 3.2, and we omit it here. ∎

Let us also remark that with a standard bookkeeping, the algorithm of Theorem 4.1 can also output a subgraph of G isomorphic to F.

5. Enumerating potential maximal cliques

In this section we show that all potential maximal cliques of graph $G=(V,E)$ can be enumerated by making use of connected vertex sets with special restrictions. This approach represents a significant simplification over previous algorithms for listing potential maximal cliques [12, 13]. More precisely, we show that for every potential maximal clique Ω there exists a vertex set $Z \subset V$ and a vertex $z \in Z$ such that

- $|Z|-1 \leq (2/3)(n-|\Omega|)$,
- $G[Z]$ is connected,
- $\Omega = N(Z \setminus \{z\})$ or $\Omega = N(Z) \cup \{z\}$.

As far as we obtain such a classification, the enumeration algorithm is extremely simple: For each vertex $z \in V$ enumerate every connected vertex set Z containing z where $|Z|-1 \leq 2|V \setminus N[Z-\{z\}]|$. (In other words we test for each connected vertex set Z containing z, where at least $\frac{|Z|-1}{2}$ vertices are not contained in $N[Z \setminus \{z\}]$.) For each of these subsets, we run the algorithm of Bouchitté and Todinca from [5] to check if $N(Z\setminus\{z\})$ or $N(Z)\cup\{z\}$ is a potential maximal clique. The algorithm of Bouchitté and Todinca checks in $\mathcal{O}(nm)$ time if a vertex set Ω is a potential maximal clique. This is a significant simplification comparing to previous enumeration algorithms [12, 13] avoiding complications with different treatments of nice and (not) nice potential maximal cliques.

We proceed with a sequence of technical lemmas. For a potential maximal clique Ω and a vertex $x \in \Omega$ we define by D_x the vertex sets of all connected components C of $G[V \setminus \Omega]$ with $x \in N(C)$.

Lemma 5.1. *Let Ω be a potential maximal clique of $G=(V,E)$, and let $\{x,y\}$ be an edge of $G[\Omega]$ such that Ω is not a potential maximal clique in $G \setminus \{x,y\}$. Then there is $Z \subseteq V$ and $z \in Z$, such that*

- $\Omega = N(Z) \cup \{z\}$,
- *$G[Z]$ is connected, and*
- $|Z|-1 \leq (1/2)(n-|\Omega|)$.

Corollary 5.2. *Let Ω be a potential maximal clique of $G = (V, E)$, such that Ω is a potential maximal clique in $G\setminus\{x, y\}$ for every edge $\{x, y\}$ of $G[\Omega]$. Then $N(D_x) = \Omega$ for every vertex $x \in \Omega$.*

Let $\mathcal{C}$ be the set of connected components of $G[V \setminus \Omega]$ with the following two properties: For each connected component $C \in \mathcal{C}$ there exists a pair of vertices $x, y \in \Omega$ such that C is the unique component from $\mathcal{C}$ with $x, y \in N(C)$, and for each pair of vertices $x, y \in \Omega$ there exists a connected component $C \in \mathcal{C}$ such that $x, y \in N(C)$. Let W be the vertex set of $\mathcal{C}$, we refer to the graph $G' = G[\Omega \cup W]$ as to a *reduced graph for* Ω. In other words $\mathcal{C}$ is an inclusion minimal witness for Ω being a potential maximal clique of G, by only using connected components of $G[V \setminus \Omega]$. The set $\mathcal{C}$ can be constructed by the following procedure which is repeated recursively if possible: If there exists a connected component C of $G[V \setminus \Omega]$ such that for each pair $x, y \in N(C)$ there is a connected component $C' \neq C$ in $G[V \setminus \Omega]$ such that $x, y \in N(C')$, then remove C from the graph.

Lemma 5.3. *Let Ω be a potential maximal clique of $G = (V, E)$ such that Ω is also a potential maximal clique in $G\setminus\{x, y\}$ for every edge $\{x, y\}$ of $G[\Omega]$, and where $G' = G[\Omega\cup W]$ contains at least 4 connected components. Then there is $Z \subset V$ and $z \in Z$ such that*

- $\Omega = N(Z \setminus \{z\})$,
- *$G[Z]$ is connected, and*
- $|Z| - 1 \leq (3/5)(n - |\Omega|)$.

The following characterization is used in the new algorithm enumerating potential maximal cliques.

Lemma 5.4. *For every potential maximal clique Ω of $G = (V, E)$, there exists a vertex set $Z \subseteq V$ and $z \in Z$ such that*

- $|Z| - 1 \leq (2/3)(n - |\Omega|)$,
- *$G[Z]$ is connected, and*
- $\Omega = N(Z \setminus \{z\})$ *or* $\Omega = N(Z) \cup \{z\}$.

Let us remark that Lemma 5.4 yields a simple algorithm enumerating potential maximal cliques. We just connected vertex sets Z of bounded size and check if either $N(Z \setminus \{z\})$ or $N(Z) \cup \{z\}$ is a potential maximal clique. The enumeration of such connected vertex sets can be done in time $\mathcal{O}(n^2 \cdot 1.7549^n)$ [13] and checking if a set is a potential maximal clique in $\mathcal{O}(nm)$ time [5].

In what follows we improve (slightly) the running time of the algorithm. The improvement is based on the previous lemmata. The proof gain by exploiting the fact that the most time consuming case is when there are exactly three connected components in the reduced graph.

Theorem 5.5. *All potential maximal cliques of an n-vertex graph can be enumerated in time $\mathcal{O}(1.734601^n)$.*

We need the following results.

Theorem 5.6 (Berry, Bordat, and Cogis [3])**.** *There is an algorithm listing all minimal separators of an input graph G in $\mathcal{O}(n^3|\Delta_G|)$ time.*

Theorem 5.7 (Fomin and Villanger [13])**.** *Every n-vertex graph has $\mathcal{O}(1.6181^n)$ minimal separators.*

Putting together Theorems 3.2, 5.5, 5.6, and 5.7, we arrive at the following corollary.

Corollary 5.8. *For every $t \geq 0$, a maximum induced subgraph of treewidth at most t in an n-vertex graph G can be found in time $\mathcal{O}(1.734601^n \cdot n^{\mathcal{O}(t)})$.*

Similarly, by Theorem 4.1, we have the following corollary.

Corollary 5.9. *For every $t \geq 0$ and graph F of treewidth t, checking if an n-vertex graph G contains an induced subgraph isomorphic to F (and finding one if such exist) can be done in time $\mathcal{O}(1.734601^n \cdot n^{\mathcal{O}(t)})$.*

Let us remark that the treewidth of an n-vertex planar, and more generally, graph excluding some fixed graph as a minor, is $\mathcal{O}(\sqrt{n})$ [1]. Therefore, if F is a graph excluding some fixed graph as a minor, deciding if G has induced subgraph isomorphic to F can be done in time $1.734601^{n+o(n)}$.

6. Conclusion and open questions

In this paper we have shown how the theory of minimal triangulations can be used to obtain moderate exponential algorithms for a number of problems about induced subgraphs. With some modifications our technique can be used for different problems of the same flavor, like finding a maximum connected induced subgraph of small treewidth. It would be interesting to see if Theorem 3.2 can be extended for finding maximum induced subgraphs with other specific properties like being planar or excluding some h-vertex graph H as a minor.

Another very interesting question is, how many potential maximal cliques can be in an n-vertex graph? Theorem 5.5 says that roughly at most 1.734601^n. How tight is this bound? There are graphs with roughly $3^{n/3} \approx 1.442^n$ potential maximal cliques [12]. Let us remind that by the classical result of Moon and Moser [19] (see also Miller and Muller [18]) that the number of maximal cliques in a graph on n vertices is at most $3^{n/3}$. Can it be that the right upper bound on the number of potential maximal cliques is also roughly $3^{n/3}$? By Theorem 3.2, this would yield a dramatic improvement for many moderate exponential algorithms.

References

[1] N. Alon, P. Seymour, and R. Thomas, *A separator theorem for nonplanar graphs*, J. Amer. Math. Soc., 3 (1990), pp. 801–808.

[2] S. Arnborg, D. G. Corneil, and A. Proskurowski, *Complexity of finding embeddings in a k-tree*, SIAM J. Algebraic Discrete Methods, 8 (1987), pp. 277–284.

[3] A. Berry, J. P. Bordat, and O. Cogis, *Generating all the minimal separators of a graph*, Int. J. Found. Comput. Sci., 11 (2000), pp. 397–403.

[4] H. L. Bodlaender, *Polynomial algorithms for graph isomorphism and chromatic index on partial k-trees*, J. Algorithms, 11 (1990), pp. 631–643.

[5] V. Bouchitté and I. Todinca, *Treewidth and minimum fill-in: Grouping the minimal separators*, SIAM J. Comput., 31 (2001), pp. 212–232.

[6] ———, *Listing all potential maximal cliques of a graph*, Theor. Comput. Sci., 276 (2002), pp. 17–32.

[7] J. Chen, I. A. Kanj, and G. Xia, *Labeled search trees and amortized analysis: improved upper bounds for NP-hard problems*, Algorithmica, 43 (2005), pp. 245–273.

[8] M. Davis and H. Putnam, *A computing procedure for quantification theory*, J. Assoc. Comput. Mach., 7 (1960), pp. 201–215.

[9] F. V. Fomin, S. Gaspers, and A. V. Pyatkin, *Finding a minimum feedback vertex set in time* $O(1.7548^n)$, in Proceedings of the 2nd International Workshop on Parameterized and Exact Computation (IWPEC 2006), vol. 4169 of Lecture Notes in Comput. Sci., Springer, Berlin, 2006, pp. 184–191.
[10] F. V. Fomin, S. Gaspers, A. V. Pyatkin, and I. Razgon, *On the minimum feedback vertex set problem: Exact and enumeration algorithms*, Algorithmica, 52 (2008), pp. 293–307.
[11] F. V. Fomin, F. Grandoni, and D. Kratsch, *Measure and conquer: A simple* $O(2^{0.288\,n})$ *independent set algorithm*, in 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2006), New York, 2006, ACM and SIAM, pp. 18–25.
[12] F. V. Fomin, D. Kratsch, I. Todinca, and Y. Villanger, *Exact algorithms for treewidth and minimum fill-in*, SIAM J. Comput., 38 (2008), pp. 1058–1079.
[13] F. V. Fomin and Y. Villanger, *Treewidth computation and extremal combinatorics*, in Proceedings of the 34th International Colloquium on Automata, Languages and Programming (ICALP 2008), LNCS, Springer, 2008, pp. 210–221.
[14] M. Fürer, *A faster algorithm for finding maximum independent sets in sparse graphs*, in Proceedings of the 7th Latin American Theoretical Informatics Symposium (LATIN 2006), vol. 3887 of Lecture Notes in Comput. Sci., Springer-Verlag, Berlin, 2006, pp. 491–501.
[15] S. Gupta, V. Raman, and S. Saurabh, *Fast exponential algorithms for maximum* r*-regular induced subgraph problems*, in Proceedings of the 26th International Conference Foundations of Software Technology and Theoretical Computer Science, (FSTTCS 2006), LNCS, Springer-Verlag, 2006, pp. 139–151.
[16] P. Heggernes, *Minimal triangulations of graphs: A survey*, Discrete Mathematics, 306 (2006), pp. 297–317.
[17] T. Jian, *An* $O(2^{0.304n})$ *algorithm for solving maximum independent set problem*, IEEE Trans. Computers, 35 (1986), pp. 847–851.
[18] R. E. Miller and D. E. Muller, *A problem of maximum consistent subsets*, IBM Research Rep. RC-240, J.T. Watson Research Center, Yorktown Heights, New York, USA, 1960.
[19] J. W. Moon and L. Moser, *On cliques in graphs*, Israel Journal of Mathematics, 3 (1965), pp. 23–28.
[20] I. Razgon, *Exact computation of maximum induced forest*, in Proceedings of the 10th Scandinavian Workshop on Algorithm Theory (SWAT 2006), vol. 4059 of Lecture Notes in Comput. Sci., Berlin, 2006, Springer, pp. 160–171.
[21] ———, *A faster solving of the maximum independent set problem for graphs with maximal degree 3*, in Proceedings of the Second ACiD Workshop (ACiD 2006), vol. 7 of Texts in Algorithmics, King's College, London, 2006, pp. 131–142.
[22] N. Robertson and P. D. Seymour, *Graph minors. II. Algorithmic aspects of tree-width*, Journal of Algorithms, 7 (1986), pp. 309–322.
[23] J. M. Robson, *Algorithms for maximum independent sets*, Journal of Algorithms, 7 (1986), pp. 425–440.
[24] D. J. Rose, R. E. Tarjan, and G. S. Lueker, *Algorithmic aspects of vertex elimination on graphs*, SIAM J. Comput., 5 (1976), pp. 266–283.
[25] R. E. Tarjan and A. E. Trojanowski, *Finding a maximum independent set*, SIAM Journal on Computing, 6 (1977), pp. 537–546.