SIAM J. COMPUT.
Vol. 42, No. 6, pp. 2197–2216

SUBEXPONENTIAL PARAMETERIZED ALGORITHM FOR MINIMUM FILL-IN*

FEDOR V. FOMIN† AND YNGVE VILLANGER†

Abstract. The Minimum Fill-in problem is used to decide if a graph can be triangulated by adding at most k edges. In 1994, Kaplan, Shamir, and Tarjan showed that the problem is solvable in time $\mathcal{O}(2^{\mathcal{O}(k)} + k^2nm)$ on graphs with n vertices and m edges and thus is fixed parameter tractable. Here, we give the first subexponential parameterized algorithm solving Minimum Fill-in in time $\mathcal{O}(2^{\mathcal{O}(\sqrt{k}\log k)} + k^2nm)$. This substantially lowers the complexity of the problem. Techniques developed for Minimum Fill-in can be used to obtain subexponential parameterized algorithms for several related problems, including Minimum Chain Completion, Chordal Graph Sandwich, and Triangulating Colored Graph.

Key words. chordal graph, parameterized complexity, subexponential algorithm

AMS subject classifications. 68Q17, 68Q25, 68W40

DOI. 10.1137/11085390X

1. Introduction.

A graph is *chordal* (or triangulated) if every cycle of length at least four contains a chord, i.e., an edge between nonadjacent vertices of the cycle. The Minimum Fill-in problem (also known as Minimum Triangulation and Chordal Graph Completion) is as follows:

Minimum Fill-in
Input: A graph $G = (V, E)$ and a nonnegative integer k.
Question: Is there $F \subseteq [V]^2$, $\lvert F\rvert \leq k$, such that graph $H = (V, E \cup F)$ is chordal?

The name "fill-in" is due to the fundamental problem arising in sparse matrix computations which was studied intensively in the past. During Gaussian eliminations of large sparse matrices new nonzero elements called fill can replace original zeros, thus increasing storage requirements and running time needed to solve the system. The problem of finding the right elimination ordering minimizing the number of fill elements can be expressed as the Minimum Fill-in problem on graphs [47, 49]. See also [15, Chapter 7] for a more recent overview of related problems and techniques. Besides sparse matrix computations, applications of Minimum Fill-in can be found in database management [2], artificial intelligence, and the theory of Bayesian statistics [13, 28, 41, 53]. The survey of Heggernes [31] gives an overview of techniques and applications of minimum and minimal triangulations.

Minimum Fill-in (under the name Chordal Graph Completion) was one of the 12 open problems presented at the end of the first edition of Garey and Johnson's book [27] and it was proved to be NP-complete by Yannakakis [54]. Kaplan, Shamir, and Tarjan proved that Minimum Fill-in is fixed parameter tractable by giving an algorithm of running time $\mathcal{O}(m16^k)$ in [37] and improved the running time to

*Received by the editors November 3, 2011; accepted for publication (in revised form) September 11, 2012; published electronically December 5, 2013. The preliminary version of this paper appears in *Proceedings of ACM-SIAM Symposium on Discrete Algorithms*, Kyoto, Japan, 2012.
http://www.siam.org/journals/sicomp/42-6/85390.html

†Department of Informatics, University of Bergen, Bergen, Norway (fedor.fomin@ii.uib.no, yngve.villanger@ii.uib.no). The first author is supported by the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013)/ERC grant agreement 267959. The second author is supported by the Research Council of Norway.

$\mathcal{O}(k^6 16^k + k^2 mn)$ in [38], where m is the number of edges and n is the number of vertices of the input graph. There was a chain of algorithmic improvements resulting in decreasing the constant in the base of the exponents. In 1996, Cai [11], reduced the running time to $\mathcal{O}((n+m)\frac{4^k}{k+1})$. The fastest parameterized algorithm known prior to our work is the recent algorithm of Bodlaender, Heggernes, and Villanger with running time $\mathcal{O}(2.36^k + k^2 mn)$ [4].

In this paper we give the first subexponential parameterized algorithm for Minimum Fill-in. The last chapter of Flum and Grohe's book [21, Chapter 16] concerns subexponential fixed parameter tractability (FPT), the complexity class SUBEPT, which, loosely speaking—we skip here some technical conditions—is the class of problems solvable in time $2^{o(k)} n^{\mathcal{O}(1)}$, where n is the input length and k is the parameter. Subexponential FPT is intimately linked with the theory of exact exponential algorithms for hard problems, which are better than the trivial exhaustive search, though still exponential [22]. Based on the fundamental results of Impagliazzo, Paturi, and Zane [34], Flum and Grohe established that most of the natural parameterized problems are not in SUBEPT unless the Exponential Time Hypothesis (ETH) fails. Until recently, the only notable exceptions of problems in SUBEPT were problems on planar graphs and, more generally, on graphs excluding some fixed graph as a minor [16]. In 2009, Alon, Lokshtanov, and Saurabh et al. [1] used a novel application of color coding to show that parameterized Feedback Arc Set in Tournaments is in SUBEPT. Minimum Fill-in is the first problem on general graphs which appeared to be in SUBEPT.

General overview of our approach. The important tools in obtaining our subexponential algorithm are the techniques based on the properties of minimal triangulations and potential maximal cliques of Bouchitté and Todinca [8]. These technics were deployed in the context of computing the treewidth of special graph classes and were used later in exact exponential algorithms [23, 24, 25]. The novel application of potential maximal cliques in subexponential algorithms is based on new algorithmic and combinatorial results about these objects.

A set of vertices Ω of a graph G is a *potential maximal clique* if there is a minimal triangulation such that Ω is a maximal clique in this triangulation. Let Π be the set of all potential maximal cliques in graph G. The importance of potential maximal cliques is that if we are given the set Π, then by using the machinery from [8, 23], it is possible to compute an optimum triangulation in time up to polynomial factor proportional to $|\Pi|$. Let G be an n-vertex graph and k be the parameter. If (G,k) is a YES instance of the Minimum Fill-in problem, then every maximal clique of every optimum triangulation is obtained from some potential maximal clique of G by adding at most k fill edges. We call potential maximal cliques missing at most k edges *vital*. To give a general overview of our algorithm, we start with an approach that does not work directly, and then explain what has to be changed to succeed. The algorithm consists of three main steps:

Step A. Apply a kernelization algorithm that in time $n^{\mathcal{O}(1)}$ reduces the problem instance to an instance of size polynomial in k.

Step B. Enumerate all vital potential maximal cliques of an n-vertex graph in time $n^{o(k/\log k)}$. By Step A, $n = k^{\mathcal{O}(1)}$. Thus the running time of the enumeration algorithm and the number of vital potential maximal cliques is $2^{o(k)}$.

Step C. Use dynamic programming to solve the problem in time proportional to the number of vital potential maximal cliques, which is $2^{o(k)}$.

Step A, kernelization for Minimum Fill-in, was known prior to our work. In 1994, Kaplan et al. gave a kernel with $\mathcal{O}(k^5)$ vertices. Later the kernelization was improved to $\mathcal{O}(k^3)$ in [38] and then to $2k^2+4k$ in [45]. Step C, with some modifications, is similar to the algorithm from [8, 23]. Step B does not work directly or at least we do not know how to make it work. Instead of enumerating vital potential maximal cliques we make a "detour." We use a branching (recursive) algorithm that in subexponential time outputs a subexponential number of graphs avoiding a specific combinatorial structure, the nonreducible graphs. (We postpone the definition of these graphs to section 3.) In nonreducible graphs we are able to enumerate vital potential maximal cliques. Thus Step B is replaced with the following:

Step B1. Apply a branching algorithm to generate $n^{\mathcal{O}(\sqrt{k})}$ nonreducible instances such that the original instance is a YES instance if and only if at least one of the generated nonreducible instances is a YES instance.

Step B2. Show that if G is nonreducible, then all vital potential maximal cliques of G can be enumerated in time $n^{\mathcal{O}(\sqrt{k})}$.

Putting together Steps A, B1, B2, and C, we obtain the subexponential algorithm.

Our algorithmic techniques can be used to show that several other problems belong to SUBEPT:

- A *chain* graph is a bipartite graph where the sets of neighbors of vertices form an inclusion chain. In the MINIMUM CHAIN COMPLETION problem, we are asked if a bipartite graph can be turned into a chain graph by adding at most k edges. The problem was introduced by Golumbic [29] and Yannakakis [54]. The concept of chain graph has surprising applications in ecology [43, 48]. Feder, Mannila, and Terzi in [20] gave approximation algorithms for this problem. We show that MINIMUM CHAIN COMPLETION is solvable within time $\mathcal{O}(2^{\mathcal{O}(\sqrt{k}\log k)} + k^2nm)$.
- The TRIANGULATING COLORED GRAPH problem is a generalization of MINIMUM FILL-IN. The instance is a graph with some of its vertices colored; the task is to add at most k fill edges such that the resulting graph is chordal and no fill edge is monochromatic. We postpone the formal definition of the problem to section 7. The problem was studied intensively because of its close relation to the PERFECT PHYLOGENY PROBLEM—a fundamental and long-standing problem for numerical taxonomists [7, 10, 36]. The TRIANGULATING COLORED GRAPH problem is NP-complete [6] and $W[t]$-hard for any t when parameterized by the number of colors [5]. However, it is not hard to see that a fixed parameter tractable algorithm when parameterized by the number of fill edges can be obtained by adapting the minimum fill-in algorithm of Cai [11]. By our results, TRIANGULATING COLORED GRAPH is solvable in time $\mathcal{O}(2^{\mathcal{O}(\sqrt{k}\log k)}+n^{\mathcal{O}(1)})$.
- In CHORDAL GRAPH SANDWICH we are given two graphs G_1 and G_2 on the same vertex set, and the question is if there is a chordal graph G which is a supergraph of G_1 and a subgraph of G_2. The problem is a generalization of TRIANGULATING COLORED GRAPH. We refer to the paper of Golumbic, Kaplan, and Shamir [30] for a general overview of graph sandwich problems. We show that deciding if a sandwiched chordal graph G can be obtained from G_1 by adding at most k fill edges is possible in time $\mathcal{O}(2^{\mathcal{O}(\sqrt{k}\log k)} + n^{\mathcal{O}(1)})$.

The remaining part of the paper is organized as follows. Section 2 contains definitions and preliminary results. In section 3, we provide a branching algorithm simplifying the instance corresponding to Step B1 of the description above. Section 4 provides an algorithm enumerating vital potential maximal cliques in nonreducible graphs, i.e., Step B2. This is the most important part of our algorithm. It is based on new insight into the combinatorial structure of potential maximal cliques. In section 5, we show how to adapt the algorithm from [8, 23] to implement Step C. The main algorithm is given in section 6. In section 7, we show how the ideas used for MINIMUM FILL-IN can be used to obtain subexponential algorithms for other problems. We conclude with open problems in section 8.

2. Preliminaries. We denote by $G = (V, E)$ a finite, undirected, and simple graph with vertex set $V(G) = V$ and edge set $E(G) = E$. We also use n to denote the number of vertices and m the number of edges in G. For a nonempty subset $W \subseteq V$, the subgraph of G induced by W is denoted by $G[W]$. We say that a vertex set $W \subseteq V$ is *connected* if $G[W]$ is connected. The *open neighborhood* of a vertex v is $N(v) = \{u \in V : uv \in E\}$ and the *closed neighborhood* is $N[v] = N(v) \cup \{v\}$. For a vertex set $W \subseteq V$ we put $N(W) = \bigcup_{v \in W} N(v) \setminus W$ and $N[W] = N(W) \cup W$. Also for $W \subset V$ we define $\mathbf{fill}_G(W)$, or simple $\mathbf{fill}(W)$, to be the number of nonedges of W, i.e., the number of pairs $u \neq v \in W$ such that $uv \notin E(G)$. We use G_W to denote the graph obtained from graph G by completing its vertex subset W into a clique. We say that a path P in graph G is *chordless* if its vertex set induces a path. In other words, any two nonconsecutive vertices of P are not adjacent in G. We refer to Diestel's book [17] for basic definitions of graph theory.

Chordal graphs and minimal triangulations. *Chordal* or *triangulated* graphs form the class of graphs containing no induced cycles of length more than three. In other words, every cycle of length at least four in a chordal graph contains a chord. Graph $H = (V, E \cup F)$ is said to be a *triangulation* of $G = (V, E)$ if H is chordal. The triangulation H is called *minimal* if $H' = (V, E \cup F')$ is not chordal for every edge subset $F' \subset F$ and H is a *minimum* triangulation if $H' = (V, E \cup F')$ is not chordal for every edge set F' such that $|F'| < |F|$. The edge set F for the chordal graph H is called the *fill* of H, and if H is a minimum triangulation of G, then $|F|$ is the minimum fill-in for G.

Minimal triangulations can be described in terms of vertex eliminations (also known as the elimination game) [26, 47]. A vertex elimination procedure takes as input a vertex ordering $\pi\colon \{1, 2, \ldots, n\} \to V(G)$ of graph G and outputs a chordal graph $H = H_n$. We put $H_0 = G$ and define H_i to be the graph obtained from H_{i-1} by completing all neighbors v of $\pi(i)$ in H_{i-1} with $\pi^{-1}(v) > i$ into a clique. An elimination ordering π is called *minimal* if the corresponding vertex elimination procedure outputs a minimal triangulation of G.

PROPOSITION 2.1 (see [46]). *Graph H is a minimal triangulation of G if and only if there exists a minimal elimination ordering π of G such that the corresponding procedure outputs H.*

We will also need the following description of the fill edges introduced by vertex eleminations.

PROPOSITION 2.2 (see [50]). *Let H be the chordal graph produced by vertex elimination of graph G according to ordering π. Then $uv \notin E(G)$ is a fill edge of H if and only if there exists a path $P = uw_1w_2 \ldots w_\ell v$ such that $\pi^{-1}(w_i) < \min(\pi^{-1}(u), \pi^{-1}(v))$ for each $1 \leq i \leq \ell$.*

Minimal separators. Let u and v be two nonadjacent vertices of a graph G. A set of vertices $S \subseteq V$, $u, v \notin S$, is a *u, v-separator* if u and v are in different connected components of the graph $G[V \setminus S]$. We say that S is a *minimal u, v-separator* of G if no proper subset of S is an u, v-separator and that S is a *minimal separator* of G if there are two vertices u and v such that S is a minimal u, v-separator. Notice that a minimal separator can be a proper vertex subset of another minimal separator.

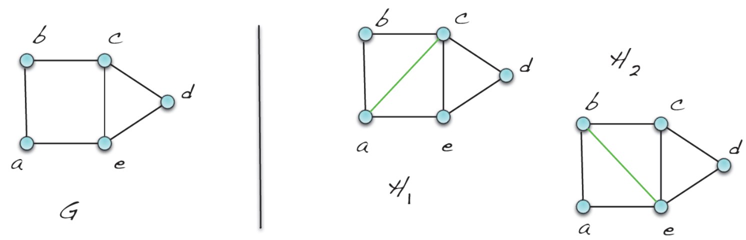

FIG. 2.1. *Graph G has two minimal triangulations, H_1 and H_2. The maximal cliques in H_1 are $\{a, b, c\}$, $\{a, c, e\}$, and $\{c, d, e\}$. In H_2 the maximal cliques are $\{a, b, e\}$, $\{b, c, e\}$, and $\{c, d, e\}$. Potential maximal cliques of G are $\{a, b, c\}$, $\{a, c, e\}$, $\{c, d, e\}$, $\{a, b, e\}$, and $\{b, c, e\}$. Graph G has two minimal a, d-separators, namely, $\{c, e\}$ and $\{b, e\}$.*

If a minimal separator is a clique, we refer to it as a *clique minimal separator*. A connected component C of $G[V \setminus S]$ is a *full* component associated to S if $N(C) = S$. The following proposition is an exercise in [29].

PROPOSITION 2.3 (folklore). *A set S of vertices of G is a minimal a, b-separator if and only if a and b are in different full components associated to S. In particular, S is a minimal separator if and only if there are at least two distinct full components associated to S.*

Potential maximal cliques are combinatorial objects whose properties are crucial for our algorithm. A vertex set Ω is defined as a *potential maximal clique* in graph G if there is some minimal triangulation H of G such that Ω is a maximal clique of H; see Figure 2.1. Potential maximal cliques were defined by Bouchitté and Todinca in [8, 9].

The following proposition was proved by Kloks, Kratsch, and Spinrad for minimal separators [39] and by Bouchitté and Todinca for potential maximal cliques [8].

PROPOSITION 2.4 (see [8, 39]). *Let X be either a potential maximal clique or a minimal separator of G, and let G_X be the graph obtained from G by completing X into a clique. Let $C_1, C_2, \ldots, C_r$ be the connected components of $G \setminus X$. Then graph H obtained from G_X by adding a set of fill edges F is a minimal triangulation of G if and only if $F = \bigcup_{i=1}^{r} F_i$, where F_i is the set of fill edges in a minimal triangulation of $G_X[N[C_i]]$.* See the example in Figure 2.2. Let us remark that by Proposition 2.4, we have that for every minimal triangulation H of a graph G,

- if S is a clique minimal separator of G, then S is a minimal separator of H;
- if Ω is a potential maximal clique of G and a clique in G, then Ω is a maximal clique and potential maximal clique of H.

The following result about the structure of potential maximal cliques is due to Bouchitté and Todinca.

PROPOSITION 2.5 (see [8]). *Let $\Omega \subseteq V$ be a set of vertices of the graph G. Let $\{C_1, C_2, \ldots, C_p\}$ be the set of the connected components of $G \setminus \Omega$ and let $S_i = N(C_i)$, $i \in \{1, 2, \ldots, p\}$. Then Ω is a potential maximal clique of G if and only if*

1. *$G \setminus \Omega$ has no full component associated to Ω, i.e., $S_i \subset \Omega$ for every $i \in \{1, 2, \ldots, p\}$, and*
2. *the graph on the vertex set Ω obtained from $G[\Omega]$ by completing each S_i, $i \in \{1, 2, \ldots, p\}$, into a clique is a complete graph. In other words, every pair of nonadjacent vertices of Ω is in some S_i.*

Moreover, if Ω is a potential maximal clique, then $\{S_1, S_2, \ldots, S_p\}$ is the set of minimal separators of G contained in Ω.

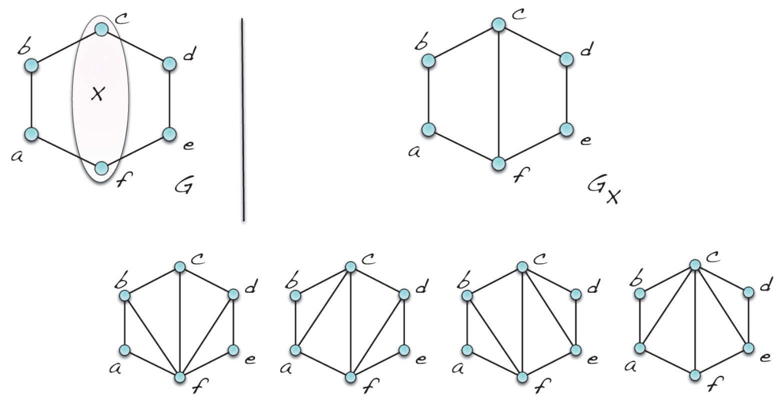

Fig. 2.2. *In graph G vertices c and f form a minimal separator X. Graph $G\backslash X$ has two connected components, $C_1 = \{a, b\}$ and $C_2 = \{d, e\}$. There are four minimal triangulations of G that can be obtained by triangulating G_X. Each of these minimal triangulations is a "combination" of a minimal triangulation of graph $G_X[N[C_1]]$, which is a cycle $abcf$, and $G_X[N[C_2]]$, which is cycle $cfde$. No minimal triangulation of G can be obtained from G_X by adding an edge "crossing" cf.*

Let us remark that if Ω is a potential maximal clique, and we want to check if it remains so after adding new edges, then it is sufficient to verify only the first condition of Proposition 2.5 since the second condition will hold automatically.

A naive approach of deciding if a given vertex subset is a potential maximal clique would be to try all possible minimal triangulations. Proposition 2.5 brings us to a faster recognition algorithm.

Proposition 2.6 (see [8]). *There is an algorithm that, given a graph $G = (V, E)$ and a set of vertices $\Omega \subseteq V$, verifies if Ω is a potential maximal clique of G in time $\mathcal{O}(nm)$.*

We need also the following proposition from [23].

Proposition 2.7 (see [23]). *Let Ω be a potential maximal clique of G. Then for every $y \in \Omega$, $\Omega = N_G(Y) \cup \{y\}$, where Y is the connected component of $G\backslash(\Omega\backslash\{y\})$ containing y.* For example, in graph G depicted in Figure 2.1, vertices $\{b, c, e\}$ form a potential maximal clique Ω. Say, for vertex $c \in \Omega$, the component C of $G\backslash(\Omega\backslash\{c\})$ containing c consists of vertices c and d. The neighborhood of this component consists of b and e and $\Omega = N_G(C) \cup \{c\} = \{b, c, e\}$.

Parameterized complexity. A parameterized problem Π is a subset of $\Gamma^* \times \mathbb{N}$ for some finite alphabet Γ. An instance of a parameterized problem consists of (x, k), where k is called the parameter. A central notion in parameterized complexity is fixed parameter tractability (FPT) which means, for a given instance (x, k), solvability in time $f(k) \cdot p(|x|)$, where f is an arbitrary function of k and p is a polynomial in the input size. We refer to the book of Downey and Fellows [19] for further reading on parameterized complexity. For Minimum Fill-in, the natural parameterization is by the number of fill edges and the parameterized version of the problem is as follows:

Minimum Fill-in
Input: A graph $G = (V, E)$ and a nonnegative integer k.
Parameter: k.
Question: Is there $F \subseteq [V]^2$, $|F| \leq k$, such that graph $H = (V, E \cup F)$ is chordal?

Kernelization. A *kernelization algorithm* for a parameterized problem $\Pi \subseteq \Gamma^* \times \mathbb{N}$ is an algorithm that given $(x, k) \in \Gamma^* \times \mathbb{N}$ outputs in time polynomial in $|x| + k$

a pair $(x', k') \in \Gamma^* \times \mathbb{N}$, called a *kernel* such that $(x, k) \in \Pi$ if and only if $(x', k') \in \Pi$, $|x'| \leq g(k)$, and $k' \leq k$, where g is some computable function. The function g is referred to as the size of the kernel. If $g(k) = k^{\mathcal{O}(1)}$, then we say that Π admits a polynomial kernel.

There are several known polynomial kernels for the MINIMUM FILL-IN problem [37, 38]. The best known kernelization algorithm is due to Natanzon, Shamir, and Sharan [44, 45], which for a given instance (G, k) outputs in time $\mathcal{O}(k^2nm)$ an instance (G', k') such that $k' \leq k$, $|V(G')| \leq 2k^2+4k$, and (G, k) is a YES instance if and only if (G', k') is.

PROPOSITION 2.8 (see [44, 45]). MINIMUM FILL-IN *has a kernel with vertex set of size* $\mathcal{O}(k^2)$*. The running time of the kernelization algorithm is* $\mathcal{O}(k^2nm)$*.*

3. Branching. Rule 1 is a branching procedure identifying a set of subproblems on which we call the algorithm recursively. The procedure is based on a notion of an obscure visibility. Loosely speaking, the visibility of a set of vertices X is h-obscured from a chordless path P if for every internal vertex v of P at least h vertices of X are not adjacent to v. Thus from every internal vertex of P, at least h vertices cannot be seen.

More formally, let X be a vertex set of G. We say that the *visibility of* X *from a chordless path* $P = uw_1w_2 \dots w_\ell v$ *is* h*-obscured* if $|X \setminus N(w_i)| \geq h$ for every $i \in \{1, \dots, \ell\}$. See Figure 3.1 for an example.

In the branching procedure, we use the notion of obscure visibility only for special sets and paths defined by a pair of nonadjacent vertices u and v. Let $X = N(u) \cap N(v)$ be the common neighborhood of u and v, and let $P = uw_1w_2 \dots w_\ell v$ be a chordless path such that the visibility of X from P is h-obscured. The idea behind the branching is that every fill-in edge set of G should contain either uv or at least h edges between vertices of X and some internal vertex of P. The proof of this fact is based on Proposition 2.2 and is given in Lemma 3.1.

RULE 1 (branching rule). *If instance* $(G = (V, E), k)$ *of* MINIMUM FILL-IN *contains a pair of nonadjacent vertices* $u, v \in V$ *and a chordless* uv*-path* $P = uw_1w_2 \dots w_\ell v$ *such that visibility of* $X = N(u) \cap N(v)$ *from* P *is* h*-obscured, then branch into* $\ell + 1$ *instances* $(G_0, k_0), (G_1, k_1), \dots, (G_\ell, k_\ell)$*. Here*

- $G_0 = (V, E \cup F_0)$*,* $k_0 = k - 1$*, where* $F_0 = \{uv\}$*;*
- *for* $i \in \{1, \dots, \ell\}$*,* $G_i = (V, E \cup F_i)$*,* $k_i = k - |F_i|$*, where* $F_i = \{w_ix | x \in X \wedge w_ix \notin E\}$*.*

LEMMA 3.1. *Rule* 1 *is sound, i.e.,* (G, k) *is a YES instance if and only if* (G_i, k_i) *is a YES instance for some* $i \in \{0, \dots, \ell\}$*.*

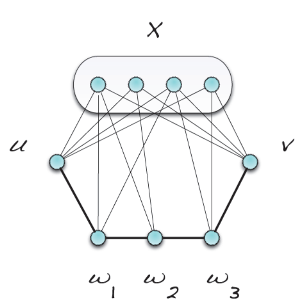

FIG. 3.1. *A vertex set* $X = N(u) \cap N(v)$ *of size* 4 *and a path* $P = uw_1w_2w_3v$ *such that the visibility of* X *from* P *is* 2*-obscured.*

Proof. If for some $i \in \{0, \dots, \ell\}$, (G_i, k_i) is a YES instance, then G can be turned into a chordal graph by adding at most $k_i + |F_i| = k$ edges, and thus (G, k) is a YES instance.

Let (G, k) be a YES instance, and let $F \subseteq [V]^2$ be such that graph $H = (V, E \cup F)$ is a minimal triangulation of G and $|F| \leq k$. By Proposition 2.1, there exists an ordering π of V such that the elimination game algorithm on G and π outputs H. Without loss of generality, we can assume that $\pi^{-1}(u) < \pi^{-1}(v)$. If for some $x \in X$, $\pi^{-1}(x) < \pi^{-1}(u)$, then by Proposition 2.2, $uv \in F$. Also by Proposition 2.2, if $\pi^{-1}(w_i) < \pi^{-1}(u)$ for each $i \in \{1, \dots, \ell\}$, then again $uv \in F$. In both cases (G_0, k_0) is a YES instance.

The only remaining case is when $\pi^{-1}(u) < \pi^{-1}(x)$ for all $x \in X$, and there is at least one vertex of P placed after u in ordering π. Let $i \geq 1$ be the smallest index such that $\pi^{-1}(u) < \pi^{-1}(w_i)$. Thus for every $x \in X$, in the path $xuw_1w_2 \dots w_i$ all internal vertices are ordered by π before x and w_i. By Proposition 2.2, this implies that w_i is adjacent to all vertices of X, and hence (G_i, k_i) is a YES instance. ∎

The following lemma shows that every branching step of Rule 1 can be performed in polynomial time.

LEMMA 3.2. *Let (G, k) be an instance of* MINIMUM FILL-IN *and let h be an integer. It can be identified in time $\mathcal{O}(n^4)$ if there is a pair $u, v \in V(G)$ satisfying the conditions of Rule* 1*. Moreover, if the conditions of Rule* 1 *hold, then a pair u, v of two nonadjacent vertices and a chordless uv-path P such that the visibility of $N(u) \cap N(v)$ from P is h-obscured can be found in time $\mathcal{O}(n^4)$.*

Proof. For each pair of nonadjacent vertices u, v, we compute $X = N(u) \cap N(v)$. We compute the set of all vertices $W \subseteq V(G) \setminus \{u, v\}$ such that every vertex of W is nonadjacent to at least h vertices of X. Then conditions of Rule 1 do not hold for u and v if in the subgraph G_{uv} induced by $W \cup \{u, v\}$, u and v are in different connected components. If u and v are in the same connected component of G_{uv}, then a shortest (in G_{uv}) uv-path P is a chordless path and the visibility of X from P is h-obscured. Clearly, all these procedures can be performed in time $\mathcal{O}(n^4)$. ∎

To obtained the claimed bound on the running time of our algorithm, we use Rule 1 only for the case when $h \geq \sqrt{k}$. We say that instance (G, k) is *nonreducible* if Rule 1 cannot be applied to an $\sqrt{k}$-obscured path. Thus for every pair of nonadjacent vertices u, v of a nonreducible graph G, there is no uv-path with $\sqrt{k}$-obscured visibility of $N(u) \cap N(v)$.

LEMMA 3.3. *Let $t(n, k)$ be the maximum number of nonreducible problem instances resulting from recursive application of Rule* 1 *starting from instance (G, k) with $|V(G)| = n$ and $h = \sqrt{k}$. Then $t(n, k) = n^{\mathcal{O}(\sqrt{k})}$ and all generated nonreducible instances can be enumerated within the same time bound.*

Proof. Let us assume that we branch on the instances corresponding to a pair u, v and path $P = uw_1w_2 \dots w_\ell v$ such that the visibility of $N(u) \cap N(w)$ is obscure from P. Then the value of $t(n, k)$ is at most $\sum_{i=0}^{\ell} t(n, k_i)$. Here $k_0 = k - 1$ and for all $i \geq 1$, $k_i = k - |F_i| \leq k - \sqrt{k}$. Since the number of vertices in P does not exceed n, $t(n, k) \leq t(n, k-1) + n \cdot t(n, k - \sqrt{k})$. By making use of standard arguments on the number of leaves in branching trees (see, for example, [35, Theorem 8.1]) it follows that $t(n, k) = n^{\mathcal{O}(\sqrt{k})}$. By Lemma 3.2, every recursive call of the branching algorithm can be done in time $\mathcal{O}(n^4)$, and thus all nonreducible instances are generated in time $\mathcal{O}(n^{\mathcal{O}(\sqrt{k})} \cdot n^4) = n^{\mathcal{O}(\sqrt{k})}$. ∎

4. Listing vital potential maximal cliques. Let (G, k) be a YES instance of Minimum Fill-in. This means that G can be turned into a chordal graph H by adding at most k edges. Every maximal clique in H corresponds to a potential maximal clique of G. The observation here is that if a potential maximal clique Ω needs more than k edges to be added to become a clique, then no solution H can contain Ω as a maximal clique.

A potential maximal clique Ω is *vital* if the number of edges in $G[\Omega]$ is at least $|\Omega|(|\Omega|-1)/2-k$. In other words, the subgraph induced by vital potential maximal clique can be turned into a complete graph by adding at most k edges. In this section we show that all vital potential maximal cliques of an n-vertex nonreducible graph can be enumerated in time $n^{\mathcal{O}(\sqrt{k})}$. In section 5 we prove that the only potential maximal cliques that are essential for a fill-in with at most k edges are the ones that miss at most k edges from a clique.

First we show how to enumerate potential maximal cliques which are, in some sense, almost cliques. This enumeration algorithm will be used as a subroutine to enumerate vital potential maximal cliques. A potential maximal clique Ω is a *quasi clique* if there is a set $Z \subseteq \Omega$ of size at most $5\sqrt{k}$ such that $\Omega \backslash Z$ is a clique. In particular, if $|\Omega| \leq 5\sqrt{k}$, then Ω is a quasi clique. The following lemma gives an algorithm enumerating all quasi cliques.

Lemma 4.1. *Let (G, k) be a problem instance on n vertices. Then all quasi cliques in G can be enumerated within time $n^{\mathcal{O}(\sqrt{k})}$.*

Proof. We will prove that while a quasi clique can be very large, it can be reconstructed in polynomial time from a small set of $\mathcal{O}(\sqrt{k})$ vertices. Hence all quasi cliques can be generated by enumerating vertex subsets of size $\mathcal{O}(\sqrt{k})$. Because the number of vertex subsets of size $\mathcal{O}(\sqrt{k})$ is $n^{\mathcal{O}(\sqrt{k})}$, this will prove the lemma.

Let $\Omega = X \cup Z$ be a potential maximal clique which is a quasi clique, where $Z \subseteq \Omega$ is a set of size at most $5\sqrt{k}$ such that $X = \Omega \backslash Z$ is a clique. Depending on the number of full components associated to X in $G \backslash \Omega$, we consider three cases: there are at least two full components, there is exactly one, and there is no full component.

Consider first the case when X has at least two full components, say, C_1 and C_2. In this case, by Proposition 2.3, X is a minimal clique separator of $G \backslash Z$. Let H be some *minimal* triangulation of $G \backslash Z$. By Proposition 2.4, X is a minimal separator in every minimal triangulation of $G \backslash Z$. Therefore, X remains a minimal separator in H. It is well known that every chordal graph has at most $n-1$ minimal separators and that they can be enumerated in linear time [12]. To enumerate quasi cliques we implement the following algorithm. We construct a minimal triangulation H of $G \backslash Z$. A minimal triangulation can be constructed in time $\mathcal{O}(nm)$ or $\mathcal{O}(n^{\omega} \log n)$, where $\omega < 2.373$ is the exponent of matrix multiplication and m is the number of edges in G [33, 50, 52]. For every minimal separator S of H, where $G[S]$ is a clique, we check if $S \cup Z$ is a potential maximal clique in G. This can be done in $\mathcal{O}(nm)$ time by Proposition 2.6. Therefore, in this case, the time required to enumerate all quasi cliques Ω of the form $X \cup Z$, up to a polynomial multiplicative factor, is proportional to the number of sets Z of size at most $5\sqrt{k}$. The total running time to enumerate quasi cliques of this type is $n^{\mathcal{O}(\sqrt{k})}$.

Now we consider the case when no full component in $G \backslash \Omega$ is associated to X. This means that for every connected component C of $G \backslash \Omega = G \backslash (Z \cup X)$, there is a vertex $x \in X \backslash N(C)$. By Proposition 2.5, X is also a potential maximal clique in $G \backslash Z$. We construct a minimal triangulation H of $G \backslash Z$. Because X is a clique and a potential maximal clique in $G \backslash Z$, by Proposition 2.4, we have that X is also a potential maximal clique in H. By the classical result of Dirac [18], chordal graph H contains at most n maximal cliques and all maximal cliques of H can be enumerated in linear time [3]. For every maximal clique K of H such that K is also a clique in G, we check if $K \cup Z$ is a potential maximal clique in G, which can be done in $\mathcal{O}(nm)$ time by Proposition 2.6. As in the previous case, the enumeration of all such quasi cliques boils down to enumerating sets Z, which takes time $n^{\mathcal{O}(\sqrt{k})}$.

In the last case, vertex set X has a unique full component C_r in $G \backslash \Omega$ associated to X. Let $C_1, C_2, \ldots, C_r$ be the connected components of $G \backslash \Omega = G \backslash (Z \cup X)$. We claim that for every $i \in \{1, \ldots, r-1\}$,

- $S_i = N_{G \backslash Z}(C_i)$ is a clique minimal separator in $G \backslash Z$.

Indeed, C_i is a connected component of $G \backslash \Omega$ and therefore $N(C_i) \subseteq \Omega$. Therefore, $S_i = N_{G \backslash Z}(C_i) = N_G(C_i) \backslash Z \subseteq \Omega \backslash Z = X$ and thus S_i is a clique. To prove that S_i is a minimal separator in $G \backslash Z$, we show that S_i has at least two full components in $G \backslash Z$. By Proposition 2.3, this will imply that S_i is a minimal separator. By definition, C_i is a full component associated to S_i in graph $G \backslash Z$. Moreover, in graph $G \backslash Z$, there is a connected component C' of $(G \backslash Z) \backslash S_i$ containing $X \backslash S_i$. Because X is a clique, we have that $N_{G \backslash Z}(C') = S_i$, and thus C' is another full component associated to S_i. This conclude the proof of the claim.

Let H be a minimal triangulation of $G \backslash Z$. Because X is a clique in $G \backslash Z$, X is also a clique in H. Let K be a maximal clique of H containing X. By the claim above, each S_i, $i \in \{1, \ldots, r-1\}$, is a clique minimal separator in $G \backslash Z$ and hence, by Proposition 2.4, is also a minimal separator in H. Therefore H has no fill edges between vertices separated by S_i in $G \backslash Z$. This implies that K is disjoint from $C_1, C_2, \ldots, C_{r-1}$. Indeed, C_r is the unique full component associated to X, and thus for every C_i, $i \in \{1, \ldots, r-1\}$, there is $x \in X \subseteq K$ such that $x \notin N_{G \backslash Z}(C_i) = N_H(C_i) = S_i$. In $G \backslash Z$, every vertex of C_i is separated from x by S_i. Thus if there was a vertex $y \in C_i \cap K$, then because K is a clique in H, this would imply the existence of a fill edge xy between vertices separated by S_i. But as we already observed, there are no such fill edges, and hence $K \cap C_i = \emptyset$.

Because Ω is a potential maximal clique in G, by Proposition 2.5, there is $y \in \Omega$ such that $y \notin N_G(C_r)$. Since C_r is a full component for X, it follows that $y \in Z$. Moreover, because for every connected component $C \neq C_r$ of $G \backslash \Omega$, we have that $K \cap C = \emptyset$, it follows that C is also a connected component of $H \backslash K$. Thus every connected component of $H \backslash K$ containing a neighbor of y in G is also a connected component of $G \backslash \Omega$ containing a neighbor of y.

Let $B_1, B_2, \ldots, B_\ell$ be the set of connected components in $G \backslash (K \cup Z)$ with $y \in N_G(B_i)$. We define

$$Y = \bigcup_{1 \leq i \leq \ell} B_i \cup \{y\}.$$

By Proposition 2.7, $\Omega = N_G(Y) \cup \{y\}$ and in this case the potential maximal clique is characterized by y and Y.

To summarize, to enumerate all quasi cliques corresponding to the last case we do the following. For every set Z of size at most $5\sqrt{k}$, we construct a minimal triangulation H of $G \backslash Z$. Chordal graph H has at most n maximal cliques. For every maximal clique K of H and for every $y \in Z$, we compute the set Y. We use Proposition 2.6 to check if $N_G(Y) \cup \{y\}$ is a potential maximal clique. The total running time to enumerate quasi cliques in this case is bounded, up to polynomial factor, by the number of subsets of size $\mathcal{O}(\sqrt{k})$ in G, which is $n^{\mathcal{O}(\sqrt{k})}$. ∎

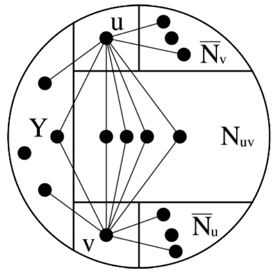

FIG. 4.1. *Partitioning of potential maximal clique Ω into sets $\overline{N}_u, \overline{N}_v, N_{uv}, \{u\}, \{v\}$, and Y.*

Now we are ready to prove the result about vital potential maximal cliques in nonreducible graphs.

LEMMA 4.2. *Let (G, k) be a nonreducible instance of the problem. All vital potential maximal cliques in G can be enumerated within time $n^{\mathcal{O}(\sqrt{k})}$, where n is the number of vertices in G.*

Proof. We start by enumerating all vertex subsets of G of size at most $5\sqrt{k} + 2$ and apply Proposition 2.6 to check if each such set is a vital potential maximal clique or not.

Let Ω be a vital potential maximal clique with at least $5\sqrt{k} + 3$ vertices and let $Y \subseteq \Omega$ be the set of vertices of Ω such that each vertex of Y is adjacent in G to at most $|\Omega| - 1 - \sqrt{k}$ vertices of Ω. To turn Ω into a complete graph, for each vertex $v \in Y$, we have to add at least $\sqrt{k}$ fill edges incident to v. Hence $|Y| \leq 2\sqrt{k}$. If $\Omega \backslash Y$ is a clique, then Ω is a quasi clique. By Lemma 4.1, all quasi cliques can be enumerated in time $n^{\mathcal{O}(\sqrt{k})}$.

If $\Omega \backslash Y$ is not a clique, there is at least one pair of nonadjacent vertices $u, v \in \Omega \backslash Y$. By Proposition 2.5, there is a connected component C of $G \backslash \Omega$ such that $u, v \in N(C)$.

CLAIM 1. *There is $w \in C$ such that $|\Omega \backslash N(w)| \leq 5\sqrt{k} + 2$.*

Proof. Aiming toward a contradiction, we assume that the claim does not hold. We define the following subsets of $\Omega \backslash Y$:

- $\overline{N}_u \subseteq \Omega \backslash Y$ is the set of vertices which are not adjacent to u,
- $\overline{N}_v \subseteq \Omega \backslash Y$ is the set of vertices which are not adjacent to v, and
- $N_{uv} = \Omega \backslash (Y \cup \overline{N}_u \cup \overline{N}_v)$ is the set of vertices adjacent to u and to v.

See Figure 4.1 for an illustration. Let us note that

$$\Omega = \overline{N}_u \cup \overline{N}_v \cup N_{uv} \cup \{u\} \cup \{v\} \cup Y.$$

Since $u, v \notin Y$, we have that there is less than $\sqrt{k}$ fill edges incident to u or v, and thus $\max\{|\overline{N}_u|, |\overline{N}_v|\} \leq \sqrt{k}$.

We claim that $|N_{uv}| \leq \sqrt{k}$. Aiming toward a contradiction, let us assume that $|N_{uv}| > \sqrt{k}$. By our assumption, every vertex $w \in C$ is not adjacent to at least $5\sqrt{k}+2$ vertices of Ω. Since $|Y \cup \overline{N}_u \cup \overline{N}_v \cup \{u\} \cup \{v\}| \leq 2\sqrt{k} + \sqrt{k} + \sqrt{k} + 2 = 4\sqrt{k} + 2$, we see that each vertex of C is nonadjacent to at least $\sqrt{k}$ vertices of N_{uv}. We take a shortest uv-path P with all internal vertices in C. Because C is a connected component and $u, v \in N(C)$, such a path exists. Every internal vertex of P is nonadjacent to at least $\sqrt{k}$ vertices of $N_{uv} \subseteq N(u) \cap N(v)$, and thus the visibility of N_{uv} from P is $\sqrt{k}$-obscured. But this is a contradiction to the assumption that (G, k) is nonreducible. Hence $|N_{uv}| \leq \sqrt{k}$.

Thus if the claim does not hold, we have

$$|\Omega| = |\overline{N}_u \cup \overline{N}_v \cup N_{uv} \cup \{u\} \cup \{v\} \cup Y| \le 5\sqrt{k} + 2,$$

but this contradicts the assumption that $|\Omega| \ge 5\sqrt{k}+3$. This concludes the proof of the claim. □

We have shown that for every vital potential maximal clique Ω of size at least $5\sqrt{k}+3$, there is a connected component C and $w \in C$ such that $|\Omega \backslash N(w)| \le 5\sqrt{k}+2$. Let H be the graph obtained from G by completing $N(w)$ into a clique.

CLAIM 2. *Ω is a quasi clique in H.*

Proof. The graph $H[\Omega]$ consist of a clique plus at most $5\sqrt{k}+2$ vertices. Thus to show that Ω is a quasi clique in H, it is sufficient to argue that Ω is a potential maximal clique in H. Vertex set Ω is a potential maximal clique in G, and thus by Proposition 2.5, there is no full component associated to Ω in $G\backslash\Omega$. Because $N(w) \cap \Omega \subseteq N(C) \subset \Omega$, there is no full component associated to Ω in H. Then by Proposition 2.5, Ω is a potential maximal clique in H as well. Hence Ω is a quasi clique in H, which concludes the proof of the claim. □

To conclude, we use the following strategy to enumerate all vital potential maximal cliques:

- We enumerate first all quasi cliques in G in time $n^{\mathcal{O}(\sqrt{k})}$ by making use of Lemma 4.1, and for each quasi clique we use Proposition 2.6 to check if it is a vital potential maximal clique.
- We also try all vertex subsets of size at most $5\sqrt{k}+2$ and use Proposition 2.6 to check if each such set is a vital potential maximal clique.
- As we have shown, all vital potential maximal cliques which are not enumerated prior to this moment should satisfy the condition of Claim 1. By Claim 2, each such vital potential maximal clique is a quasi clique in the graph H obtained from G by selecting some vertex w and turning $N_G(w)$ into clique. Thus for every vertex w of G, we construct graph H by completing $N(w)$ into a clique and then use Lemma 4.1 to enumerate all quasi cliques in H. For each quasi clique of H, we use Proposition 2.6 to check if it is a vital potential maximal clique in G.

The total running time of this procedure is $n^{\mathcal{O}(\sqrt{k})}$. □

5. Exploring the remaining solution space. For an instance (G, k) of MINIMUM FILL-IN, let Π_k be the set of all vital potential maximal cliques. In this section, we give an algorithm of running time $\mathcal{O}(nm|\Pi_k|)$, where n is the number of vertices and m the number of edges in G. The algorithm receives (G, k) and Π_k as an input and decides if (G, k) is a YES instance. The algorithm is a modification of the algorithm from [23]. The only difference is that the algorithm from [23] computes an optimum triangulation from the set of all potential maximal cliques, while here we have to work only with vital potential maximal cliques. For the reader's convenience we provide the full proof, but first we need the following lemma.

LEMMA 5.1. *Let S be a minimal separator in G and let C be a full connected component of $G\backslash S$ associated to S. Then every minimal triangulation H of G_S contains a maximal clique K such that $S \subset K \subseteq S \cup C$.*

Proof. By Proposition 2.4, H is a minimal triangulation of G_S if and only if $H[S \cup C]$ is a minimal triangulation of $G_S[S \cup C]$. Because S is a clique in G_S, S is a subset of some maximal clique K of $H[S \cup C]$. By definition, K is a potential maximal clique in $G_S[S \cup C]$, and by Proposition 2.5, K is a potential maximal clique in G. Since $G_S[S \cup C] \backslash S$ has a full component associated to S, by making use of Proposition 2.5 we conclude that S is not a potential maximal clique in $G_S[S \cup C]$ and thus $S \subset K$. $\square$

LEMMA 5.2. *Given a set of all vital potential maximal cliques Π_k of G, it can be decided in time $\mathcal{O}(nm|\Pi_k|)$ if (G,k) is a YES instance of* MINIMUM FILL-IN.

Proof. Let $\mathbf{mfi}(G)$ be the minimum number of fill edges needed to triangulate G. Let us recall that by $\mathbf{fill}_G(\Omega)$ we denote the number of nonedges in $G[\Omega]$ and by G_Ω the graph obtained from G by completing Ω into a clique. If $\mathbf{mfi}(G) \leq k$, then by Proposition 2.4, we have

$$\mathbf{mfi}(G) = \min_{\Omega \in \Pi_k} \left[\mathbf{fill}_G(\Omega) + \sum_{C \text{ is a component of } G\backslash\Omega} \mathbf{mfi}(G_\Omega[C \cup N_G(C)]) \right]. \tag{5.1}$$

Formula (5.1) can be used to compute $\mathbf{mfi}(G)$; however, by making use of this formula we are not able to obtain the claimed running time. To implement the algorithm in time $\mathcal{O}(nm|\Pi_k|)$, we compute $\mathbf{mfi}(G_\Omega[C \cup N_G(C)])$ by dynamic programming.

By Proposition 2.5, for every connected component C of $G\backslash\Omega$, where $\Omega \in \Pi_k$, $S = N_G(C) \subset \Omega$ is a minimal separator. We define the set Δ_k as the set of all minimal separators S such that $S = N(C)$ for some connected component C in $G\backslash\Omega$ for some $\Omega \in \Pi_k$. Since for every $\Omega \in \Pi_k$ the number of components in $G\backslash\Omega$ is at most n, we have $|\Delta_k| \leq n|\Pi_k|$.

For $S \in \Delta_k$ and a full connected component C of $G\backslash S$ associated to S, we define $\Pi_{S,C}$ as the set of potential maximal cliques $\Omega \in \Pi_k$ such that $S \subset \Omega \subseteq S \cup C$. The triple (S, C, Ω) was called a good triple in [23].

For every $\Omega \in \Pi_k$, connected component C of $G\backslash\Omega$, and $S = N(C)$, we compute $\mathbf{mfi}(F)$, where $F = G_\Omega[C \cup S]$. We start dynamic programming by computing the values for all sets (S, C) such that $\Omega' = C \cup S$ is an inclusion-minimal potential maximal clique. In this case we put $\mathbf{mfi}(F) = \mathbf{fill}(C \cup S)$. Observe that $G_S[C \cup S] = G_\Omega[C \cup S]$. Hence by Lemma 5.1, for every minimal triangulation H of G_S, there exists a potential maximal clique Ω in G such that Ω is a maximal clique in H and $S \subset \Omega \subseteq S \cup C$. Thus $\Omega \in \Pi_{S,C}$. Using this observation, we write the following formula for dynamic programming:

$$\mathbf{mfi}(F) = \min_{\Omega' \in \Pi_{S,C}} \left[\mathbf{fill}_F(\Omega') + \sum_{C' \text{ is a component of } F\backslash\Omega'} \mathbf{mfi}(F_{\Omega'}[C' \cup N(C')]) \right]. \tag{5.2}$$

The fact $S \subset \Omega'$ ensures that the solution in (5.2) can be reconstructed from instances with $|S \cup C|$ of smaller sizes. By (5.1) and (5.2), we can decide if there exists a triangulation of G using at most k fill edges. It remains to argue for the running time.

Finding connected components in $G\backslash\Omega$ and computing $\mathbf{fill}(\Omega)$ can easily be done in $\mathcal{O}(n+m)$ time. Furthermore, (5.1) is applied $|\Pi_k|$ times in total. The running time of dynamic programming using (5.2) is proportional to the number of states of the dynamic programming, which is

$$\sum_{S \in \Delta_k} \sum_{C \in G\backslash S} |\Pi_{S,C}|.$$

The graph $G\backslash\Omega$ contains at most n connected components and thus for every minimal separator, each potential maximal clique is counted at most n times, and thus the number of elements in the sum does not exceed $n|\Pi_k|$. The total running time is $\mathcal{O}(nm|\Pi_k|)$. ◻

Let us remark that the algorithm from Lemma 5.2 can be easily modified to construct the triangulation.

6. Putting things together. Now we are in position to prove the main result of this paper.

THEOREM 6.1. *The* MINIMUM FILL-IN *problem is solvable in time* $\mathcal{O}(2^{\mathcal{O}(\sqrt{k}\log k)} + k^2nm)$.

Proof. Step A. Given instance (G,k) of the MINIMUM FILL-IN problem, we use Proposition 2.8 to obtain a kernel (G',k') on $\mathcal{O}(k^2)$ vertices and with $k' \le k$. Let us note that (G,k) is a YES instance if and only if (G',k') is a YES instance. This step is performed in time $\mathcal{O}(k^2nm)$.

*Step B*1. We use Branching Rule 1 on instance (G',k'). Since the number of vertices in G' is $\mathcal{O}(k^2)$, by Lemma 3.3, the result of this procedure is the set of $(k^2)^{\mathcal{O}(\sqrt{k})} = 2^{\mathcal{O}(\sqrt{k}\log k)}$ nonreducible instances $(G_1,k_1),\dots,(G_p,k_p)$. For each $i \in \{1,2,\dots,p\}$, graph G_i has $\mathcal{O}(k^2)$ vertices and $k_i \le k$. Moreover, by Lemma 3.1, we have that (G',k'), and thus (G,k), is a YES instance if and only if at least one (G_i,k_i) is a YES instance. By Lemma 3.3, the running time of this step is $2^{\mathcal{O}(\sqrt{k}\log k)}$.

*Step B*2. For each $i \in \{1,2,\dots,p\}$, we list all vital potential maximal cliques of graph G_i. By Lemma 4.2, the number of all vital potential maximal cliques in nonreducible graph G_i is $2^{\mathcal{O}(\sqrt{k}\log k)}$ and they can be listed within the same running time.

Step C. At this step for each $i \in \{1,2,\dots,p\}$, we are given instance (G_i,k_i) together with the set Π_{k_i} of vital potential maximal cliques of G_i computed in Step B2. We use Lemma 5.2 to solve the MINIMUM FILL-IN problem for instance (G_i,k_i) in time $\mathcal{O}(k^6|\Pi_{k_i}|) = 2^{\mathcal{O}(\sqrt{k}\log k)}$. If at least one of the instances (G_i,k_i) is a YES instance, then by Lemma 3.1, (G,k) is a YES instance. If all instances (G_i,k_i) are NO instances, we conclude that (G,k) is a NO instance. Since $p = 2^{\mathcal{O}(\sqrt{k}\log k)}$, we have that Step C can be performed in time $2^{\mathcal{O}(\sqrt{k}\log k)}$. The total running time required to perform all steps of the algorithm is $\mathcal{O}(2^{\mathcal{O}(\sqrt{k}\log k)} + k^2nm)$. ◻

Let us remark that our decision algorithm can be easily adapted to output the optimum fill-in of size at most k.

7. Applications to other problems. The algorithmic techniques developed in the previous sections can be modified to solve several related problems. Problems considered in this section are MINIMUM CHAIN COMPLETION, CHORDAL GRAPH SANDWICH, and TRIANGULATING COLORED GRAPH.

Minimum chain completion. A bipartite graph $G = (V_1,V_2,E)$ is a chain graph if the neighborhoods of the nodes in V_1 form a chain, that is, there is an ordering $v_1,v_2,\dots,v_{|V_1|}$ of the vertices in V_1 such that $N(v_1) \subseteq N(v_2) \subseteq \cdots \subseteq N(v_{|V_1|})$.

MINIMUM CHAIN COMPLETION
Input: A bipartite graph $G = (V_1,V_2,E)$ and integer $k \ge 0$.
Parameter: k.
Question: Is there $F \subseteq V_1 \times V_2$, $|F| \le k$, such that graph $H = (V_1,V_2,E \cup F)$ is a chain graph?

In his NP-completeness proof of MINIMUM FILL-IN, Yannakakis [54] used the following observation. Let G be a bipartite graph with bipartitions V_1 and V_2, and let G' be cobipartite (the complement of bipartite) graph formed by turning V_1 and V_2 into cliques. Then G can be transformed into a chain graph by adding k edges if and only if G' can be triangulated with k edges. By Theorem 6.1, MINIMUM CHAIN COMPLETION is solvable in $\mathcal{O}(2^{\mathcal{O}(\sqrt{k}\log k)} + k^2nm)$ time.

Chordal graph sandwich. In the chordal graph sandwich problem we are given two graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ on the same vertex set V and with $E_1 \subset E_2$. The CHORDAL GRAPH SANDWICH problem asks if there exists a chordal graph $H = (V, E_1 \cup F)$ sandwiched between G_1 and G_2, that is, $E_1 \cup F \subseteq E_2$.

CHORDAL GRAPH SANDWICH
Input: Two graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ such that $E_1 \subset E_2$, and an integer k.
Parameter: k.
Question: Is there $F \subseteq E_2$ such that $|F| \leq k$ and graph $H = (V, E_1 \cup F)$ is a triangulation of G_1?

Let us remark that the CHORDAL GRAPH SANDWICH problem is equivalent to asking if there is a minimal triangulation of G_1 sandwiched between G_1 and G_2. To solve CHORDAL GRAPH SANDWICH we cannot use the algorithm from Theorem 6.1 directly. The reason is that we are only allowed to add edges from E_2 as fill edges. We need a kernelization algorithm for this problem as well. This algorithm is very similar to the kernelization algorithm for the fill-in problem, and we provide it here for completeness.

LEMMA 7.1. CHORDAL GRAPH SANDWICH *admits a kernel with $\mathcal{O}(k^2)$ vertices.*

Proof. To simplify notation, we denote an instance of the CHORDAL GRAPH SANDWICH problem by (G, E', k), where $G_1 = G = (V, E)$, $E' \cap E = \emptyset$, and $G_2 = (V, E \cup E')$.

We define two reduction rules and prove their correctness. The first rule eliminates vertices which do not participate in any induced cycle of length at least four.

Vertex-in-cycle rule. If instance (G, E', k) has vertex $u \in V$ such that for each connected component C of $G \setminus N[u]$, set $N_G(C)$ is a clique in G, then replace instance (G, E', k) with instance $(G \setminus \{u\}, E', k)$.

Let us remark that if the vertex-in-cycle rule cannot be applied to an instance (G, E', k), then every vertex of G belongs to a chordless cycle. Indeed, if the vertex-in-cycle rule cannot be applied to a vertex v, then there is a component C of $G \setminus N[v]$ such that $N(C)$ contains a pair of nonadjacent vertices x, y. But then the union of a shortest xy-path in C with v is a chordless cycle containing v.

CLAIM 3. *The vertex-in-cycle rule is sound, i.e., $(G \setminus \{u\}, E', k)$ is a YES instance if and only if (G, E', k) is a YES instance.*

Proof. Let $G_u = G \setminus \{u\}$. Chordality is a hereditary property; hence if $H = (V, E \cup F)$ is a triangulation of G with $|F| \leq k$, then $H \setminus \{u\}$ is a triangulation of G_u.

For the opposite direction, assume that $H_u = (V(G_u), E(G_u) \cup F_u)$ is a minimal triangulation of G_u, where $|F_u| \leq k$. Our objective now is to argue that $H = (V, E \cup F_u)$ is a triangulation of G. Aiming toward a contradiction, let us assume that there is a chordless cycle W of length at least four in H. Then u should be a vertex of W as $H_u = H \setminus \{u\}$ is chordal. Thus $W = uaw_1w_2 \ldots w_\ell bu$. Vertices a and b of W are adjacent to u in G_u because F_u has no edges incident with u. By the same arguments, vertices $w_1w_2 \ldots w_\ell$ are not adjacent to u in G, and by Proposition 2.3 and the fact that $N_G(C)$ is a clique for each connected component C of $G \setminus N[u]$, we have that vertices $w_1w_2 \ldots w_\ell$ are contained in a connected component C of $G \setminus N[u]$.

Cycle W is chordless, and thus a and b are not adjacent. We arrive to the fact that nonadjacent neighbors a, b of u are contained in $N(C)$, which is a clique by the condition of applying the vertex-in-cycle rule. This is a contradiction. $\square$

For every pair of nonadjacent vertices $x, y \in V$, we define A_{xy} as the set of vertices $w \in N_G(x) \cap N_G(y)$ such that there is an xy-path P avoiding $N[w]$, i.e., no inner vertex of P is in $N[w]$. Let us note that nonadjacent vertices x, y, together with $w \in A_{xy}$ and with a shortest xy-path avoiding $N[w]$, induce a cycle of length at least four. Our second rule exploits this property.

Safe-edge rule. If $|A_{xy}| > 2k$ for some pair of nonadjacent vertices x, y in a problem instance (G, E', k), then

- if $xy \notin E'$, then (G, E', k) is a NO instance, and in this case we replace (G, E', k) with a trivial NO instance;
- if $xy \in E'$, then make a new instance $(G = (V, E \cup \{xy\}), E' \setminus \{xy\}, k - 1)$.

CLAIM 4. *The safe-edge rule is sound.*

Proof. By the definition of A_{xy}, there exists an induced cycle of length at least four consisting of x, w, y and a shortest induced path from x to y in $G[(V \setminus N[w]) \cup \{x, y\}]$. Then either every triangulation of G has xy as a fill edge or there exists a fill edge incident to w. Thus in every minimal triangulation not using xy as a fill edge, every vertex of A_{xy} is an endpoint of some fill edge. But since $|A_{xy}| > 2k$, this is impossible for a triangulation using k edges. Hence (G, E', k) is a NO instance if $xy \notin E'$. Otherwise, $xy \in F$ for every edge set $F \subseteq E'$ such that $H = (V, E \cup F)$ is chordal. $\square$

We apply both rules exhaustively. It is clear that application of each rule can be done in polynomial time. We claim that if a nonreducible instance, i.e., an instance such that none of the reduction rules is applicable on this instance, contains more than $2k + 2k^2$ vertices, then this is a NO instance.

Let (G, E', k) be a nonreducible instance and let F be a set of at most k fill edges such that $F \subset E'$. Let V_F be the set of vertices of G incident with F.

CLAIM 5. *The number of vertices in* $V \setminus V_F$ *is at most* $2k^2$.

Proof. As graph G is nonreducible, for each vertex $u \in V \setminus V_F$ we cannot apply the vertex-in-cycle rule. Thus there is a connected component C of $G \setminus N[u]$ such that a pair of vertices $x, y \in N_G(C)$ are not adjacent in G. The union of a shortest xy-path in C with v induces a chordless cycle in G. Since no edge of F is incident with u and F is the set of fill edges, we conclude that $xy \in F$. As the component C contains a chordless path from x to y avoiding $N[u]$, we conclude that $u \in A_{xy}$. We cannot apply the safe-edge rule on G, hence $|A_{xy}| \leq 2k$ for each $xy \in F$. Because every vertex of $V \setminus V_F$ is contained in A_{xy} for some edge $xy \in F$, the claim follows. $\square$

We are ready to complete the proof of the lemma. The number of vertices in G is $|V| = |V_F| + |V \setminus V_F|$. Because the number of fill edges $|F| \leq k$, we have that $|V_F| \leq 2k$. By Claim 5, we have that $V \setminus V_F \leq 2k^2$, hence $|V| \leq 2k + 2k^2$. $\square$

Let us remark that the kernelization algorithm in Lemma 7.1 can be implemented to run in time $\mathcal{O}(kn^2m)$. By following almost the same procedure as in the work of Natanzon, Shamir, and Sharan [44, 45] for the fill-in problem (and with much more work and a more carefully selected set of reduction rules), it is possible to improve the running time of the algorithm from Lemma 7.1 to $\mathcal{O}(k^2nm)$. Since such an improvement is not strongly relevant to our main results and since it follows almost exactly the lines of the the work of [44, 45], we do not provide it here.

THEOREM 7.2. CHORDAL SANDWICH PROBLEM *is solvable in time* $\mathcal{O}(2^{\mathcal{O}(\sqrt{k}\log k)} + n^{\mathcal{O}(1)})$.

Proof. Let (G_1, G_2, k) be an instance of the problem. We sketch the proof by following the steps of the proof of Theorem 6.1 and commenting on the differences.

Step A. We use the kernelization algorithm of Lemma 7.1 to obtain in polynomial time an equivalent instance (G_1', G_2', k') such that $|V(G_1')| = \mathcal{O}(k^2)$ and $k' \leq k$.

Step B1. On the new instance we use Branching Rule 1 exhaustively with the adaptation that every instance is defined by fill edge set F_i where $F_i \not\subseteq E_2$ is discarded. Thus we obtain $2^{\mathcal{O}(\sqrt{k}\log k)}$ nonreducible instances.

Step B2. For each nonreducible instance (G_1^i, G_2^i, k_i), we enumerate vital potential maximal cliques of G_1^i but discard all potential maximal cliques that are not cliques in G_2^i.

Step C. Solve the remaining problem in time proportional to the number of vital potential maximal cliques in G_1' that are also cliques in G_2^i. This step is almost identical to Step C of Theorem 6.1. ∎

Triangulating colored graph. In the TRIANGULATING COLORED GRAPH problem we are given a graph $G = (V, E)$ with a partitioning of V into sets $V_1, V_2, \ldots, V_c$, a coloring of the vertices. Let us remark that this coloring is not necessarily a proper coloring of G. The question is if G can be triangulated without adding edges between vertices in the same set (color).

TRIANGULATING COLORED GRAPH *Input:* A graph $G = (V, E)$, a partitioning of V into sets $V_1, V_2, \ldots, V_c$, and an integer k. *Parameter:* k. *Question:* Is there $F \subseteq [V]^2$, $\lvert F\rvert \leq k$, such that for each $uv \in F$, $\lvert\{u, v\} \cap V_i\rvert \leq 1$, $1 \leq i \leq c$, and graph $H = (V, E_1 \cup F)$ is a triangulation of G?

TRIANGULATING COLORED GRAPH can be trivially reduced to CHORDAL GRAPH SANDWICH by defining $G_1 = G$, and the edge set of graph G_2 as the edge set of G_1 plus the set of all vertex pairs of different colors. Thus by Theorem 7.2, TRIANGULATING COLORED GRAPH is solvable in time $\mathcal{O}(2^{\mathcal{O}(\sqrt{k}\log k)} + n^{\mathcal{O}(1)})$.

8. Conclusions and open problems. In this paper we gave the first parameterized subexponential time algorithm solving MINIMUM FILL-IN in time $\mathcal{O}(2^{\mathcal{O}(\sqrt{k}\log k)} + k^2nm)$. It would be interesting to find out how tight the exponential dependence is. We would be surprised to hear about a $2^{o(\sqrt{k})}n^{\mathcal{O}(1)}$ time algorithm solving MINIMUM FILL-IN. For example, because with a natural assumption $k \leq \binom{n}{2}$, such an algorithm would be able to solve the problem in time $2^{o(n)}$. However, the only results we are aware of in this direction is that MINIMUM FILL-IN cannot be solved in time $2^{o(k^{1/6})}n^{\mathcal{O}(1)}$ unless the ETH, which posits that no subexponential time algorithms for k-CNF-SAT or CNF-SAT exist, fails [14]. See [34, 42] for more information on the ETH. Similar uncertainty occurs with a number of other graph problems expressible in terms of vertex orderings. Is it possible to prove that unless the ETH fails, there are no $2^{o(n)}$ algorithms for TREEWIDTH, MINIMUM INTERVAL COMPLETION, and OPTIMUM LINEAR ARRANGEMENT? Here the big gap between what we suspect and what we know is frustrating.

On the other hand, for the TRIANGULATED COLORED GRAPH problem, which we are able to solve in time $\mathcal{O}(2^{\mathcal{O}(\sqrt{k}\log k)} + n^{\mathcal{O}(1)})$, Bodlaender, Fellows, and Warnow [6] gave a polynomial time reduction that from a 3-SAT formula on p variables and q

clauses constructs an instance of Triangulated Colored Graph. This instance has $2 + 2p + 6q$ vertices and a triangulation of the instance respecting its coloring can be obtained by adding at most $(p + 3q) + (p + 3q)^2 + 3pq$ edges. Thus up to ETH, Triangulated Colored Graph and Chordal Graph Sandwich cannot be solved in time $2^{o(\sqrt{k})}n^{\mathcal{O}(1)}$.

The possibility of improving the nm factor in the running time $\mathcal{O}(2^{\mathcal{O}(\sqrt{k}\log k)} + k^2nm)$ of the algorithm is another interesting open question. The factor nm appears from the running time required by the kernelization algorithm to identify simplicial vertices. Identification of simplicial vertices can be done in time $\mathcal{O}(\min\{mn, n^{\omega}\log n\})$, where $\omega < 2.373$ is the exponent of matrix multiplication [33, 40, 52]. Is the running time required to obtain a polynomial kernel for Minimum Fill-in at least the time required to identify a simplicial vertex in a graph, and can search of a simplicial vertex be done faster than finding a triangle in a graph?

A combinatorial problem related to our work is to bound the number of vital potential maximal cliques that can be in an n-vertex graph. Are there graphs containing $n^{\Omega(k/\log k)}$ vital potential maximal cliques?

Finally, there are various problems in graph algorithms, where the task is to find a minimum number of edges or vertices to be changed such that the resulting graph belongs to some graph class. For example, the problems of completion to interval and proper interval graphs are fixed parameter tractable [32, 37, 38, 51]. Can these problems be solved by subexponential parameterized algorithms? Are there any generic arguments explaining why some FPT graph modification problems can be solved in subexponential time and some can't?

Acknowledgments. We are grateful to Marek Cygan and Saket Saurabh for discussions and useful suggestions.

REFERENCES

[1] N. Alon, D. Lokshtanov, and S. Saurabh, *Fast FAST*, Automata, Languages, and Programming, Lecture Notes in Comput. Sci. 5555, Springer, New York, 2009, pp. 49–58.

[2] C. Beeri, R. Fagin, D. Maier, and M. Yannakakis, *On the desirability of acyclic database schemes*, J. ACM, 30 (1983), pp. 479–513.

[3] J. R. S. Blair and B. W. Peyton, *An introduction to chordal graphs and clique trees*, in Graph Theory and Sparse Matrix Computations, IMA Vol. Math. Appl. 56, Springer, New York, 1993, pp. 1–30.

[4] H. Bodlaender, P. Heggernes, and Y. Villanger, *Faster parameterized algorithms for minimum fill-in*, Algorithmica, 61 2011, pp. 817–838.

[5] H. L. Bodlaender, M. R. Fellows, M. T. Hallett, T. Wareham, and T. Warnow, *The hardness of perfect phylogeny, feasible register assignment and other problems on thin colored graphs*, Theoret. Comput. Sci., 244 (2000), pp. 167–188.

[6] H. L. Bodlaender, M. R. Fellows, and T. Warnow, *Two strikes against perfect phylogeny*, in Automata, Languages, and Programming, Lecture Notes in Comput. Sci. 623, Springer, New York, 1992, pp. 273–283.

[7] M. L. Bonet, C. A. Phillips, T. Warnow, and S. Yooseph, *Constructing evolutionary trees in the presence of polymorphic characters*, in Proceedings of STOC, ACM, 1996, pp. 220–229.

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

[9] V. Bouchitté and I. Todinca, *Listing all potential maximal cliques of a graph*, Theoret. Comput. Sci., 276 (2002), pp. 17–32.

[10] P. Buneman, *A characterisation of rigid circuit graphs*, Discrete Math., 9 (1974), pp. 205–212.

[11] L. Cai, *Fixed-parameter tractability of graph modification problems for hereditary properties*, Inform. Process. Lett., 58 (1996), pp. 171–176.

[12] L. S. Chandran and F. Grandoni, *A linear time algorithm to list the minimal separators of chordal graphs*, Discrete Math., 306 (2006), pp. 351–358.

[13] F. R. K. Chung and D. Mumford, *Chordal completions of planar graphs*, J. Combin. Theory Ser. B, 62 (1994), pp. 96–106.

[14] M. Cygan, *private communication*, 2011.

[15] T. A. Davis, *Direct Methods for Sparse Linear Systems*, Fundam. Algorithms 2, SIAM, Philadelphia, 2006.

[16] E. D. Demaine, F. V. Fomin, M. Hajiaghayi, and D. M. Thilikos, *Subexponential parameterized algorithms on graphs of bounded genus and H-minor-free graphs*, J. Assoc. Comput. Mach., 52 (2005), pp. 866–893.

[17] R. Diestel, *Graph Theory*, 3rd ed., Grad. Texts in Math. 173, Springer, Berlin, 2005.

[18] G. A. Dirac, *On rigid circuit graphs*, Abh. Math. Sem. Univ. Hamburg, 25 (1961), pp. 71–76.

[19] R. G. Downey and M. R. Fellows, *Parameterized complexity*, Springer, New York, 1999.

[20] T. Feder, H. Mannila, and E. Terzi, *Approximating the minimum chain completion problem*, Inform. Process. Lett., 109 (2009), pp. 980–985.

[21] J. Flum and M. Grohe, *Parameterized Complexity Theory*, Texts Theoret. Comput. Sci. EATCS Ser., Springer, Berlin, 2006.

[22] F. V. Fomin and D. Kratsch, *Exact Exponential Algorithms*, Texts Theoret. Comput. Sci. EATCS Ser., Springer, Berlin, 2010.

[23] 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.

[24] F. V. Fomin and Y. Villanger, *Treewidth computation and extremal combinatorics*, in Automata, Languages, and Programming, Lecture Notes in Comput. Sci. 5125, Springer, New York, 2008, pp. 210–221.

[25] F. V. Fomin and Y. Villanger, *Finding induced subgraphs via minimal triangulations*, in STACS, LIPIcs 5, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2010, pp. 383–394.

[26] D. R. Fulkerson and O. A. Gross, *Incidence matrices and interval graphs*, Pacific J. Math., 15 (1965), pp. 835–855.

[27] M. R. Garey and D. S. Johnson, *Computers and Intractability, A Guide to the Theory of NP-Completeness*, W.H. Freeman, New York, 1979.

[28] D. Geman, *Random fields and inverse problems in imaging*, in École d'été de Probabilités de Saint-Flour XVIII—1988, Lecture Notes in Math. 1427, Springer, Berlin, 1990, pp. 113–193.

[29] M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, 1980.

[30] M. C. Golumbic, H. Kaplan, and R. Shamir, *Graph sandwich problems*, J. Algorithms, 19 (1995), pp. 449–473.

[31] P. Heggernes, *Minimal triangulations of graphs: A survey*, Discrete Math., 306 (2006), pp. 297–317.

[32] P. Heggernes, C. Paul, J. A. Telle, and Y. Villanger, *Interval completion with few edges*, in Proceedings of STOC, 2007, pp. 374–381.

[33] 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 (2005), pp. 900–913.

[34] R. Impagliazzo, R. Paturi, and F. Zane, *Which problems have strongly exponential complexity?*, J. Comput. System Sci., 63 (2001), pp. 512–530.

[35] M. Jurdzinski, M. Paterson, and U. Zwick, *A deterministic subexponential algorithm for solving parity games*, SIAM J. Comput., 38 (2008), pp. 1519–1532.

[36] S. Kannan and T. Warnow, *Triangulating* 3*-colored graphs*, SIAM J. Discrete Math., 5 (1992), pp. 249–258.

[37] H. Kaplan, R. Shamir, and R. E. Tarjan, *Tractability of parameterized completion problems on chordal and interval graphs: Minimum fill-in and physical mapping*, in Proceedings of FOCS, IEEE, 1994, pp. 780–791.

[38] H. Kaplan, R. Shamir, and R. E. Tarjan, *Tractability of parameterized completion problems on chordal, strongly chordal, and proper interval graphs*, SIAM J. Comput., 28 (1999), pp. 1906–1922.

[39] T. Kloks, D. Kratsch, and J. Spinrad, *On treewidth and minimum fill-in of asteroidal triple-free graphs*, Theoret. Comput. Sci., 175 (1997), pp. 309–335.

[40] D. Kratsch and J. Spinrad, *Between $O(nm)$ and $O(n^{\alpha})$*, SIAM J. Comput., 36 (2006), pp. 310–325.

[41] S. L. Lauritzen and D. J. Spiegelhalter, *Local computations with probabilities on graphical structures and their application to expert systems*, J. Royal Statist. Soc. B, 50 (1988), pp. 157–224.

[42] D. Lokshtanov, D. Marx, and S. Saurabh, *Lower bounds based on the Exponential Time Hypothesis*, Bull. EATCS 105, 2011, pp. 41–72.

[43] H. Mannila and E. Terzi, *Nestedness and segmented nestedness*, in Proceedings of KDD '07, ACM, New York, 2007, pp. 480–489.

[44] A. Natanzon, R. Shamir, and R. Sharan, *A polynomial approximation algorithm for the minimum fill-in problem*, in Proceedings of STOC, 1998, pp. 41–47.

[45] A. Natanzon, R. Shamir, and R. Sharan, *A polynomial approximation algorithm for the minimum fill-in problem*, SIAM J. Comput., 30 (2000), pp. 1067–1079.

[46] T. Ohtsuki, L. K. Cheung, and T. Fujisawa, *Minimal triangulation of a graph and optimal pivoting ordering in a sparse matrix*, J. Math. Anal. Appl., 54 (1976), pp. 622–633.

[47] S. Parter, *The use of linear graphs in Gauss elimination*, SIAM Review, 3 (1961), pp. 119–130.

[48] B. Patterson and W. Atmar, *Nested subsets and the structure of insular mammalian faunas and archipelagos*, Biological J. Linnean Society, 28 (1986).

[49] D. J. Rose, *A graph-theoretic study of the numerical solution of sparse positive definite systems of linear equations*, in Graph Theory and Computing, R. C. Read, ed., Academic Press, New York, 1972, pp. 183–217.

[50] 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.

[51] Y. Villanger, P. Heggernes, C. Paul, and J. A. Telle, *Interval completion is fixed parameter tractable*, SIAM J. Comput., 38 (2009), pp. 2007–2020.

[52] V. Vassilevska Williams, *Multiplying matrices faster than Coppersmith-Winograd*, in Proceedings of STOC, ACM, 2012, pp. 887–898.

[53] S. Wong, D. Wu, and C. Butz, *Triangulation of bayesian networks: A relational database perspective*, in Rough Sets and Current Trends in Computing, Lecture Notes in Comput. Sci. 2475, Springer, New York, 2002, pp. 950–950.

[54] M. Yannakakis, *Computing the minimum fill-in is NP-complete*, SIAM J. Algebraic Discrete Methods, 2 (1981), pp. 77–79.