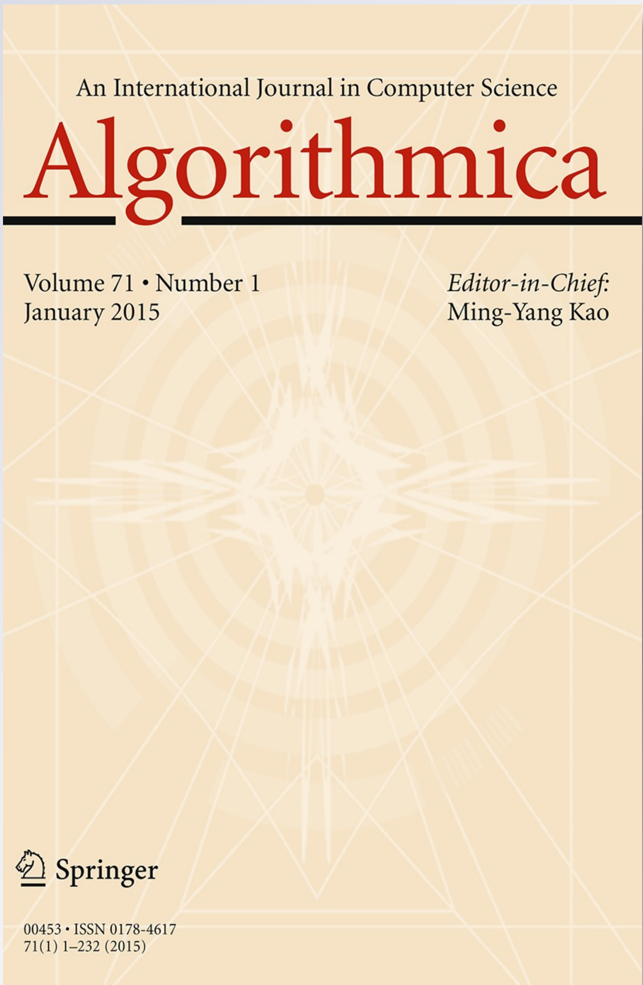
Minimum Fill-in of Sparse Graphs: Kernelization and Approximation

Fedor V. Fomin, Geevarghese Philip & Yngve Villanger

Algorithmica

ISSN 0178-4617
Volume 71
Number 1

Algorithmica (2015) 71:1-20
DOI 10.1007/s00453-013-9776-1



Springer

Algorithmica (2015) 71:1–20
DOI 10.1007/s00453-013-9776-1

Minimum Fill-in of Sparse Graphs: Kernelization and Approximation

Fedor V. Fomin · Geevarghese Philip · Yngve Villanger

Received: 23 May 2012 / Accepted: 30 March 2013 / Published online: 11 April 2013

Abstract The MINIMUM FILL-IN problem is to decide if a graph can be triangulated by adding at most k edges. The problem has important applications in numerical algebra, in particular in sparse matrix computations. We develop kernelization algorithms for the problem on several classes of sparse graphs. We obtain linear kernels on planar graphs, and kernels of size $\mathcal{O}(k^{3/2})$ in graphs excluding some fixed graph as a minor and in graphs of bounded degeneracy. As a byproduct of our results, we obtain approximation algorithms with approximation ratios $\mathcal{O}(\log k)$ on planar graphs and $\mathcal{O}(\sqrt{k}\log k)$ on H-minor-free graphs. These results significantly improve the previously known kernelization and approximation results for MINIMUM FILL-IN on sparse graphs.

Keywords Parameterized complexity · Kernelization · Minimum fill-in · Planar graphs · Linear kernel · d-Degenerate graphs

The research of Fedor V. Fomin was supported by the European Research Council (ERC) grant "Rigorous Theory of Preprocessing", reference 267959.
The research of Yngve Villanger was supported by the Research Council of Norway. The research of Geevarghese Philip was done while he was with The Institute of Mathematical Sciences, Chennai, India.
A preliminary version of this paper appeared in the proceedings of FSTTCS 2011 [14].

F.V. Fomin · Y. Villanger
Department of Informatics, University of Bergen, 5020 Bergen, Norway

F.V. Fomin
e-mail: fedor.fomin@ii.uib.no

Y. Villanger
e-mail: yngve.villanger@ii.uib.no

G. Philip (✉)
Max-Planck-Institut für Informatik, 66123 Saarbrücken, Germany
e-mail: gphilip@mpi-inf.mpg.de

1 Introduction

A graph is *chordal* (or triangulated) if every cycle of length at least four has a chord, i.e. an edge between nonadjacent vertices of the cycle. In the MINIMUM FILL-IN problem (also known as MINIMUM TRIANGULATION and CHORDAL GRAPH COMPLETION) the task is to check if at most k edges can be added to a graph such that the resulting graph is chordal. That is

> MINIMUM FILL-IN
> *Input:* A graph $G = (V, E)$ and a non-negative integer k.
> *Question:* Is there $F \subseteq [V]^2$, $|F| \leq k$, such that graph $H = (V, E \cup F)$ is chordal?

This is a classical computational problem motivated by, and named after, a fundamental issue arising in sparse matrix computations. During Gaussian eliminations of large sparse matrices, new non-zero elements—called *fill*—can replace original zeros, thus increasing storage requirements, the time needed for the elimination, and the time needed to solve the system after the elimination. The problem of finding the right elimination ordering minimizing the amount of fill elements can be expressed as the MINIMUM FILL-IN problem on graphs [27]. Besides sparse matrix computations, applications of MINIMUM FILL-IN can be found in database management, artificial intelligence, and the theory of Bayesian statistics. The survey of Heggernes [19] gives an overview of techniques and applications of minimum and minimal triangulations.

Unfortunately, the problem is notoriously difficult to analyze from the algorithmic perspective. 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 [15] and it was proved to be NP-complete by Yannakakis [31]. Due to its importance the problem has been studied intensively, and many heuristics, without performance guarantees, have been developed [24, 27].

Very few approximation and FPT algorithms for MINIMUM FILL-IN are known. Chung and Mumford [8] proved that every planar, and more generally, H-minor-free, n-vertex graph has a fill-in with $\mathcal{O}(n \log n)$ edges, thus yielding an $\mathcal{O}(n \log n)$-approximation on these classes of graphs. Agrawal et al. [1] gave an algorithm with the approximation ratio $\mathcal{O}(m^{1.25} \log^{3.5} n/k + \sqrt{m} \log^{3.5} n/k^{0.25})$, where m is the number of edges and n the number of vertices in the input graph. For graphs of degree at most d, they obtained a better approximation factor $\mathcal{O}(((nd + k)\sqrt{d} \log^4 n)/k)$. Natanzon et al. [22] provided another type of approximation algorithms for MINIMUM FILL-IN. For an input graph with a minimum fill-in of size k, their algorithm produces a fill-in of size at most $8k^2$, i.e., within a factor of $8k$ of optimal. For graphs with maximum degree d, they gave another approximation algorithm achieving the ratio $\mathcal{O}(d^{2.5} \log^4 (kd))$. Kaplan et al. proved that MINIMUM FILL-IN is fixed parameter tractable (FPT) for the parameter k by giving an algorithm which runs in $\mathcal{O}(k^6 16^k + k^2 mn)$ time [20]. Following this, faster FPT algorithms were devised for the problem, with running times that have smaller constants in the base of the exponent [6, 7]. Very recently, the first and third authors of this paper developed a subexponential FPT algorithm for the problem which runs in $\mathcal{O}(2^{\mathcal{O}(\sqrt{k} \log k)} + k^2 nm)$ time [13].

In this paper we study kernelization algorithms for MINIMUM FILL-IN on different classes of sparse graphs. Kernelization can be regarded as systematic mathematical investigation of preprocessing heuristics within the framework of parameterized complexity. In parameterized complexity each problem instance comes with a parameter k and the parameterized problem is said to admit a *polynomial kernel* if there is a polynomial time algorithm (the degree of the polynomial is independent of k), called a *kernelization* algorithm, that reduces the input instance down to an instance with size bounded by a polynomial $p(k)$ in k, while preserving the answer. This reduced instance is called a $p(k)$ *kernel* for the problem. If $p(k) = \mathcal{O}(k)$, then we call it a *linear kernel*. For example, for the instance (G, k) of PLANAR MINIMUM FILL-IN, where G is a planar graph and k is the parameter, the pair (G', k') is a linear kernel if G' is planar, the size of G', i.e., the number of edges and vertices, is $\mathcal{O}(k)$, and there is a fill-in of G with at most k fill edges if and only if there is a fill-in of G' with at most k' fill edges. Kernelization has been extensively studied, resulting in polynomial kernels for a variety of problems. In particular, it has been shown that many problems have polynomial and linear kernels on planar and other classes of sparse graphs [2, 5, 26].

There are several known polynomial kernels for the MINIMUM FILL-IN problem [20] on general (not sparse) graphs. The best known kernelization algorithm is due to Natanzon et al. [22], 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. Note that not every kernelization algorithm for fill-in in general graphs produces a sparse kernel, even if the input is a sparse graph. For example, the algorithm of Natanzon et al. [22], while reducing the number of vertices in the input graph G, introduces new edges. Thus the resulting kernel G' can be very dense. In order to obtain kernels on classes of sparse graphs, we have to design new kernelization algorithms which preserve the sparsity of the kernel.

Our Results We provide kernelization algorithms for three important and increasingly general classes of graphs. For planar graphs, we obtain an $\mathcal{O}(k)$ kernel, and for graphs excluding a fixed graph as a minor and graphs of bounded degeneracy, kernels of size $\mathcal{O}(k^{3/2})$. Our reduction rules are easy to implement. Small kernels for sparse graphs can be used as an argument explaining the successful behavior of several heuristics for sparse matrix computations. As a byproduct of our results, we obtain an approximation algorithm that, for an input planar graph with minimum fill-in of size k, produces a fill-in of size $\mathcal{O}(k \log k)$, which is within factor $\mathcal{O}(\log k)$ of optimal. For H-minor-free graphs our kernelization yields an approximation algorithm with the ratio $\mathcal{O}(\sqrt{k} \log k)$.

It is natural to ask why such kernelization algorithms, which preserve the graph class, might be worth the trouble involved in developing them. We offer three reasons to justify this effort. Our first reason is purely theoretical. Observe that one can think of a kernelization algorithm as a polynomial-time *encoding* of an arbitrarily-sized input instance to a "small" instance, where the encoding preserves the YES/NO answer. Given a graph problem and, say, a planar instance of the problem, it is not *a priori* clear why there must exist a polynomial-time algorithm which can encode the answer for this instance as a small *planar* graph. It is eminently possible that for certain problems, reducing the total size—in terms of, say, the number of bits required to represent the reduced instance—while preserving the answer necessarily entails creating so many edges that the resulting graph is, in general, non-planar. Our results show that such is *not* the case for the MINIMUM FILL-IN problem, for the three graph classes mentioned above.

A second justification for preserving planarity (or the other two properties) is somewhat more practical. It is well-known that a number of graph problems become significantly easier to solve on planar graphs and their sparse generalizations, as compared to the same problems on general graphs. Thus, many NP-hard graph problems become polynomial-time solvable in these classes, while others become easier to approximate. More pertinently, a host of graph problems have *subexponential* FPT algorithms—which run in time $\mathcal{O}(c^{o(k)}n^{\mathcal{O}(1)})$ for some constant c—on planar, H-minor free, and bounded-degeneracy graphs, while the best known algorithms for these problems on general graphs take time $\mathcal{O}(c^{\mathcal{O}(k)}n^{\mathcal{O}(1)})$ or worse. Reducing to a planar kernel (or a kernel of the other two kinds), even at the expense of some increase in the instance size, could thus be justified, since it may allow us to apply significantly faster algorithms to solve the reduced instance.

As a third justification, we present the approximation algorithms which we obtain using our kernels—see Sect. 6. These algorithms depend critically on the fact that the kernels belong to the respective graph classes—it is not sufficient that the kernel sizes are small.

2 Preliminaries

All graphs in this paper are finite and undirected. In general we follow the graph terminology of Diestel [9]. For a vertex v in graph G, $N_G(v)$ is the set of neighbours of v, and for two non-adjacent vertices u, v, $N_G(u, v) \equiv N_G(u) \cap N_G(v)$. We drop the subscript G where there is no scope for confusion. For $S \subseteq V(G)$, we use $N(S)$ for the set of neighbours in $V(G) \setminus S$ of the vertices in S, and $N[S] \equiv N(S) \cup S$. We also use $G[S]$ to denote the subgraph of G induced by S, and $G \setminus S$ to denote the subgraph $G[V \setminus S]$.

The operation of *contracting* an edge $\{u, v\}$ of a graph consists of replacing its endpoints u, v with a single vertex which is adjacent to all the former neighbours of u and v in G. A graph H is said to be a *contraction* of a graph G if H can be obtained from G by contracting zero or more edges of G. Graph H is a *minor* of G if H is a contraction of some subgraph of G. A family $\mathcal{F}$ of graphs is said to be *H-minor free* if no graph in $\mathcal{F}$ has H as a minor. For $d \in \mathbb{N}$, a graph G is said to be *d-degenerate* if every subgraph of G has a vertex of degree at most d. A family $\mathcal{F}$ of graphs is said to be of *bounded degeneracy* if there is some fixed $d \in \mathbb{N}$ such that every graph in the family is d-degenerate. Note that all graph properties discussed in this paper (being chordal, planar, H-minor free, and d-degenerate) are hereditary, i.e., are closed under taking induced subgraphs.

Minimal Separators Let u, v be two vertices in a graph G. A set S of vertices of G is said to be a *u, v-separator* of G if u and v are in different components in the graph $G \setminus S$. The set S is said to be a *minimal* u, v-separator if no proper subset of S is a

u, v-separator of G. A set S of vertices of G is said to be a *(minimal) separator* of G if there exist two vertices u, v in G such that S is a (minimal) u, v-separator of G.

Let S be a separator of a graph G. A connected component C of $G \setminus S$ is said to be *associated* with S, and is said to be a *full component* if $N(C) = S$.

The following proposition is an exercise in Golumbic's book on perfect graphs [16].

Proposition 1 *A set S of vertices of a graph G is a minimal u, v-separator if and only if u and v are in different full components of $G \setminus S$.*

A set S of vertices of a graph G is said to be a *clique separator* of G if S is a separator of G, and $G[S]$ is a clique.

Minimal and Minimum Fill-in *Chordal* or *triangulated* graphs are 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. Let F be a set of edges which, when added to a graph G, makes the resulting graph chordal. Then F is called a *fill-in* of G, and the edges in F are called *fill edges*. A fill-in F of G is said to be *minimal* if no proper subset of F is a fill-in of G, and F is a *minimum* fill-in if no fill-in of G contains fewer edges. Notice that every minimum fill-in is also minimal, and so to find a minimum fill-in it is sufficient to search the set of minimal fill-ins.

The following proposition relates minimal separators of a certain kind with minimum fill-ins of the graph.

Proposition 2 [6] *Let G be a graph, and let S be a minimal separator of G such that $G[S]$ is a complete graph minus one edge, and there is a vertex v in $V(G) \setminus S$ which is adjacent to every vertex in S. Then there exists a minimum fill-in of G which contains the single missing edge in $G[S]$ as a fill edge.*

The following proposition is folklore; for a proof see, e.g., Bodlaender et al.'s recent article on faster FPT algorithms for the MINIMUM FILL-IN problem [6].

Proposition 3 *Let $\langle v_1, v_2, v_3, v_4, \ldots, v_t \rangle$ be a chordless cycle in a graph G, and let F be a minimal fill-in of G. If $\{v_1, v_3\} \notin F$, then $\{v_2, v\} \in F$ for some $v \in \{v_4, \ldots, v_t\}$.*

The following proposition relates minimal fill-ins of a graph with minimal fill-ins of the components of the graph obtained by deleting a minimal separator.

Proposition 4 [21] *Let S be a minimal separator of G, let G' be the graph obtained by completing S into a clique, and let $E_S = E(G') \setminus E(G)$. Let $C_1, C_2, \ldots, C_r$ be the connected components of $G \setminus S$. Then $E_S \cup F$ is a minimal fill-in 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 fill-in of $G'[N[C_i]]$.*

If S is a minimal *clique* separator of G and F is any minimal fill-in of G, then the above proposition implies that no edge of F has its end points in two distinct components of $G \setminus S$. Since every clique separator S of G contains a *minimal* clique separator S', and since all the vertices of $S \setminus S'$ belong to the same component of G S', we have

Corollary 1 *Let S be a clique separator of G, and let F be a minimal fill-in of G. Then no edge in F has its end vertices in two distinct components of $G \setminus S$.*

Parameterized Complexity Parameterized algorithms [11, 12, 23] constitute one approach towards solving NP-hard problems in "feasible" time. A parameterized problem Π is a subset of $\Gamma^* \times \mathbb{N}$ for some finite alphabet Γ. An instance of a parameterized problem is of the form (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 whose degree is independent of k.

Kernelization A *kernelization algorithm* for a parameterized problem $\Pi \subseteq \Gamma^* \times \mathbb{N}$ takes a pair $(x, k) \in \Gamma^* \times \mathbb{N}$ as input, and runs in time polynomial in $|x| + k$. It outputs a pair $(x', k') \in \Gamma^* \times \mathbb{N}$, called the *kernel*, such that $(x, k) \in \Pi$ if and only if $(x', k') \in \Pi$. Further, there exist computable functions f, g such that $\max\{k', |x'|\} \leq g(k)$ and $k' \leq f(k)$. The function g is referred to as the size of the kernel. If $g(k) = \mathcal{O}(k)$, then we say that Π admits a linear kernel.

The kernels in this paper are obtained by applying a sequence of polynomial time reduction rules. We use the following notational convention: for each reduction rule, (G, k) denotes the instance on which the rule is applied, and (G', k') denotes the resulting instance. We say that a rule is *safe* if (G', k') is a YES instance if and only if (G, k) is a YES instance. We show that each rule is safe. We also show—in most cases—that the resulting graph is in the same class as G.

The remaining part of the paper is organized as follows. Sections 3, 4, and 5 give kernelization algorithms for planar, d-degenerate, and H-minor free graphs, respectively. All the three kernels use Rule 2 of Sect. 3, and Rule 6 of Sect. 4 is used in Sect. 5 as well. The kernels obtained are then used in Sect. 6 to get approximation algorithms for planar and H-minor free graphs. Section 7 shows that the problem remains *NP*-complete on 2-degenerated bipartite graphs. We conclude and state some open problems in Sect. 8.

3 A Linear Kernel for Planar Graphs

In this section we show that the planar minimum fill-in problem has a linear kernel. The kernel is obtained by applying four reduction rules. Rules 1, 2, and 3 are applied exhaustively, while Rule 4 is only applied if none of the other three can be applied. At the end of this process, the algorithm either solves the problem (giving either YES or NO as the answer), or it yields an equivalent instance (G', k'); $k' \leq k$ where G is of size $\mathcal{O}(k)$.

Reduction Rule 1 [29] *Let S be a minimal clique separator in G and let $C_1, \ldots, C_t$ be the connected components of $G \setminus S$. We set G' to be the disjoint union of the graphs $G_1, G_2, \ldots, G_t$, where G_i is isomorphic to $G[N[C_i]]$, $1 \leq i \leq t$, and set $k' \leftarrow k$.*

By Proposition 4, we have the following lemma.

Lemma 1 *Rule 1 is safe.*

Since each connected component of graph G' produced by Rule 1 is an induced subgraph of G, it follows that if G is planar, d-degenerate, or H-minor free, then G' also has the same property.

Our next rule deletes vertices which are not part of any chordless cycle; as we show later (Lemma 3), a vertex v satisfies the conditions of the rule if and only if it is not part of any chordless cycle in the graph. This rule can be inferred from previous work due to Tarjan [29] and Berry et al. [3].

Reduction Rule 2 *For a vertex v of G, let $C_1, C_2, \ldots, C_t$ be the connected components of $G \setminus N[v]$. If for every $1 \le i \le t$, the vertex set $N(C_i)$ is a clique in G, then set $G' \leftarrow G \setminus \{v\}$, $k' \leftarrow k$.*

Lemma 2 *Rule 2 is safe.*

Proof Let H be a chordal graph obtained by adding k edges to G. Chordality is a hereditary property, and thus the graph $H' = H \setminus \{v\}$ is chordal. But H' is a triangulation of $G' = G \setminus \{v\}$, and since it is obtained by adding at most k edges, we have that G' has a fill-in of size at most $k' \le k$.

For the opposite direction, let H' be a *minimal* triangulation obtained from G' by adding the set of fill edges F', where $|F'| \le k'$. Then the graph H obtained by adding F' to G is chordal. Indeed, if H was not chordal, it would contain a chordless cycle A of length at least 4 passing through v. Let w be a vertex of A not adjacent to v and let C be the connected component of $G \setminus N[v]$ containing w. The set $S = N_G(C)$ is a clique minimal separator in G and thus by Corollary 1, we can conclude that in H every path from w to v should go through some vertex of S. Hence the set S contains at least two non-consecutive (in A) vertices a and b of A. But S is a clique in G, and thus is a clique in H. Hence, a and b form a chord in A, which is a contradiction. Therefore, H is chordal. □

In Reduction Rule 2, we only remove a vertex, and thus this rule does not change hereditary properties of graphs, like being H-minor free. We now state some useful properties of graphs on which the above reduction rules cannot be applied.

Lemma 3 *A vertex v in a graph G does not satisfy the conditions of Reduction Rule 2 if and only if v is part of a chordless cycle in G.*

Proof Let v be a vertex in G which does not satisfy the conditions of Reduction Rule 2. Then there exists a connected component C of $G \setminus N[v]$ such that $N(C)$ contains two non adjacent vertices, say $x, y \in N(v)$. Let P be a shortest path from x to y in $G[C \cup \{x, y\}]$. Since x and y are not adjacent, the path P is of length at least two; let $P = \langle x = v_1, v_2, \ldots, v_\ell = y\rangle$. Since P is an induced path, $\langle v, x = v_1, v_2, \ldots, v_\ell = y\rangle$ is a chordless cycle containing v.

Conversely, let $\langle v = v_1, v_2, v_3, \ldots, v_{r-2}, v_{r-1}, v_r = v\rangle$ be a chordless cycle in G containing v, and let C be the connected component of $G \setminus N[v]$ which contains v_3 and v_{r-2}. The vertex set $N(C)$ does not contain the edge $\{v_2, v_{r-1}\}$ and hence is not a clique. □

If Reduction Rule 2 does not apply to a graph, then every vertex in the graph has at least one edge of every fill-in in its neighbourhood, in the following sense.

Lemma 4 *Let G be a graph to which Rule 2 cannot be applied, and let F be an edge set such that $H = (V, E \cup F)$ is chordal. Then for every vertex v in G, there either exists an edge $\{v, x\} \in F$, or an edge $\{u, w\} \in F$, where $u, w \in N(v)$.*

Proof From Lemma 3 it follows that every vertex v in G is part of at least one chordless cycle $\langle v = v_1, v_2, v_3, v_4, \dots, v_t \rangle$. From Proposition 3 it follows—by induction on the length of the chordless cycle—that for every vertex v there is either a fill edge $\{v, v_i\} \in F$ or an edge $\{v_2, v_t\} \in F$, for $i \in \{3, \dots, t-1\}$. □

Our next reduction rule pertains to almost-clique separators.

Reduction Rule 3 [6] *Let (G, k) be an input instance of* MINIMUM FILL-IN. *If G has a minimal separator S such that adding exactly one edge to $G[S]$ turns it into a complete graph, and there exists a vertex v in $V(G) \setminus S$ such that all vertices of S are adjacent to v, then*

1. *Turn $G[S]$ into a complete graph by adding one edge,*
2. *Apply Rule 1 on the resulting minimal clique separator, and*
3. *Reduce k by one.*

The correctness of this rule is evident from Proposition 2 and Lemma 1. We now show that the rule preserves the planarity of the graph. Observe that if the input graph G is planar, then $|S| \le 4$.

Claim 1 *Reduction Rule 3 preserves the planarity of the graph.*

Proof Let G, S be as in the statement of the rule, and let G' be the graph obtained by applying the rule to G. By Proposition 1, there are at least two full components, say C_1, C_2, associated with S in G. Let $\{u, v\}$ be the missing edge in $G[S]$. Notice that for each $i = 1, 2$, there is a uv-path in G with all internal vertices contained in C_i. This implies that each of the connected components of the output graph G' is a minor of planar graph G, and thus is planar. □

Reduction Rule 4 *Let (G, k) be an input instance of* MINIMUM FILL-IN, *where none of the Rules 1, 2, and 3 can be applied. If $|V(G)| > 6k - 4$ then return a trivial* NO *instance and stop.*

Lemma 5 *Reduction Rule 4 is safe.*

Proof Let (G, k) be a YES instance where $G = (V, E)$ is planar and none of the Rules 1, 2, and 3 can be applied. We now argue that $|V| \le 6k - 4$.

Let F be an edge set such that $|F| \le k$ and $H = (V, E \cup F)$ is chordal, and let V_F be the set of at most $2k$ vertices that are incident to the edges in F. We then have:

Claim 2 *Each vertex $v \in V \setminus V_F$ is adjacent to at least three vertices of V_F.*

Proof Since Rule 2 cannot be applied on vertex v it follows that $N[v] \subsetneq V$. Let C be a connected component of $G \setminus N[v]$ and let $S = N(C)$ be the minimal separator of G separating vertices of C from v. Rules 1 and 3 cannot be applied on S, so the graph $G[S]$ is missing at least two edges $\{x_1, y_1\}$ and $\{x_2, y_2\}$. By finding a shortest path P from x_j to y_j in $G[C \cup \{x_j, y_j\}]$ we can create a chordless cycle consisting of P and x_j, v, y_j for $j \in \{1, 2\}$. By Proposition 3 every fill-in of a chordless cycle either adds an edge incident to vertex v on the chordless cycle or adds a fill edge between its two unique neighbours. By definition there is no fill edge in F incident to v, and thus both $\{x_1, y_1\}$ and $\{x_2, y_2\}$ are contained in F. Two edges have to be incident to at least three vertices, and the claim follows. □

We construct a new graph $B = (V, E_B)$ whose edge set E_B is a subset of E, such that $\{u, w\} \in E_B$ if and only if $\{u, w\} \in E$, $u \in V_F$, and $w \notin V_F$. The graph B is planar since it is a subgraph of planar graph G, and is bipartite by construction with the two partite sets being $V_1 = V_F$ and $V_2 = V \setminus V_F$. As noted before, $|V_1| \leq 2k$; we now bound $|V_2|$. Let $\mathcal{F}$ be the set of faces in any fixed planar embedding of B. Let $s = \sum_{f \in \mathcal{F}}$(number of edges on the face f). Since B is bipartite, each face has at least four sides, and so $s \geq 4|\mathcal{F}|$. Since each edge of B lies on at most two faces in the embedding, it is counted at most twice in this process, and so $s \leq 2|E_B|$. Thus $4|\mathcal{F}| \leq 2|E_B|$. From this and the well-known Euler's formula for planar graphs applied to B (namely, $|V| - |E_B| + |\mathcal{F}| \geq 2$; observe that B may be a disconnected graph) we get $|E_B| \leq 2|V| - 4 = 2(|V_1| + |V_2|) - 4$. By Claim 2 each vertex in V_2 has degree at least 3 in B, and so $|E_B| \geq 3|V_2|$. Combining these we get $|V_2| \leq 2|V_1| - 4 \leq 4k - 4$, and so $|V| = |V_1| + |V_2| \leq 6k - 4$. □

We now argue that all executions of the rules can be performed in polynomial time. By Proposition 4, a minimal clique separator is a clique separator in every minimal triangulation of the given graph. A minimal triangulation can be constructed in $\mathcal{O}(nm)$ time [28] and the minimal separators of the triangulation which are also cliques in G can be enumerated in $\mathcal{O}(nm)$ time [4]. As a consequence Rule 1 can be executed in polynomial time. For the remaining three rules it is not hard to see that we can check, find an instance, and execute the rule in polynomial time.

The rules are applied exhaustively in the order they are described. Rule 1 is globally applied at most $n - 1$ times, since all minimal clique separators we split on, even across connected components, are the so called "non-crossing" minimal separators in the initial graph, and a graph on n vertices has at most $n - 1$ pairwise non-crossing minimal separators [25]. Each time Rule 1 is applied, at most n connected components are created, and each of them contains at most n vertices. Thus, Rule 2 is applied at most $\mathcal{O}(n^3)$ times. Rule 3 is applied at most k times as one fill edge is added each time, and finally Rule 4 is applied only once. Thus we get

Theorem 1 MINIMUM FILL-IN *has a planar kernel of size $\mathcal{O}(k)$ in planar graphs.*

4 A Subquadratic Kernel for *d*-Degenerate Graphs

We now describe two reduction rules for d-degenerate graphs. The second among these is in fact an algorithm which specifies how to apply Rule 2 and the first rule of this section in tandem. Given a problem instance (G, k) where G is a d-degenerate graph, the second rule outputs an equivalent instance (G', k') such that $k' \leq k$ and $|V(G')| = \mathcal{O}(k^{3/2})$. However, these rules *do not* guarantee that the resulting graph G' is d-degenerate. We will later show how to obtain an equivalent d-degenerate graph from G' while keeping the size bounded by $\mathcal{O}(k^{3/2})$.

The next reduction rule says that if two non-adjacent vertices in a d-degenerate graph G have many common neighbours, then the missing edge between the two vertices belongs to every small fill-in of G.

Reduction Rule 5 *Let (G, k) be an instance where G is d-degenerate. Let u, w be two non-adjacent vertices in G, and let $b = |N(u, w)|$. If $(b/2)(b-1-2d) > k$, then set $G' \leftarrow (V(G), E(G) \cup \{\{u, w\}\}), k' \leftarrow k - 1$.*

Lemma 6 *Rule 5 is safe.*

Proof Let F be a fill-in of G of size at most k. We claim that $\{u, w\} \in F$. For, if $\{u, w\} \notin F$, then let H be the chordal graph obtained by adding the edges in F to the graph G. Since u and w are non-adjacent in H, there exists an u, w-separator in H, and every minimal u, w-separator in H contains all the vertices in $N(u, w)$. Since H is chordal, every minimal separator in H is a clique [10], and so the vertex set $N(u, w)$ induces a clique in H. Hence the subgraph $H[N(u, w)]$ contains $(b-1)b/2$ edges, where $b = |N(u, w)|$. Since G is d-degenerate, the subgraph $G[N(u, w)]$ contains at most db edges. Thus $|F| \geq (b-1)b/2 - db = (b/2)(b-1-2d) > k$, a contradiction, and so $\{u, w\} \in F$. It immediately follows that $F \setminus \{\{u, w\}\}$ is a fill-in of G' of size at most $k - 1$.

Conversely, if G' has a fill-in F' of size at most $k-1$, then $F' \cup \{\{u, w\}\}$ is a fill-in of G of size at most k. □

Reduction Rule 6 *Let (G, k) be an instance where G is d-degenerate. Set (G', k') to be the instance output by Algorithm 1.*

Lemma 7 *Rule 6 is safe.*

Proof By Rule 2 it is safe to delete vertex u in Line 3. Let $e_1, e_2, \ldots, e_{|F_0'|}$ be the set of edges in F_0'. By Rule 5 it is safe to add edge e_1 to G and decrement k. Let our induction hypothesis be that it is safe to add edges $e_1, e_2, \ldots, e_{i-1}$ to G and reduce k by $i - 1$, and let us argue that it is also safe to add edges $e_1, e_2, \ldots, e_i$ and reduce k by i. Let $k_{i-1} = k - (i - 1)$. Edge $e_i = \{x, y\}$ was added to F_0' because $(b/2)(b - 1 - 2d) > k$ where $b = |N_G(x, y)|$. In the extreme case, all the edges $e_1, e_2, \ldots, e_{i-1}$ are added between vertices in $N_G(x, y)$, but $(b/2)(b - 1 - 2d) - (i - 1) > k - (i - 1) = k_{i-1}$ and thus it is safe to add edge e_i as well and reduce k_{i-1} by 1. We can now conclude that (G', k') in Line 8 is a YES instance if and only

Algorithm 1 Reduction Rule 6 for d-degenerate graphs

1: **procedure** RULE6(G, k) ▷ G is assumed to be d-degenerate.
2: **while** (Rule 2 applies to (G, k) and a vertex $u \in V(G)$) **do**
3: $G \leftarrow G \setminus \{u\}$
4: $F_0' \leftarrow \emptyset$
5: **for** (each nonadjacent pair $x, y \in V(G)$) **do**
6: **if** (Rule 5 applies to (G, k) and the non-adjacent vertices x, y) **then**
7: $F_0' \leftarrow F_0' \cup \{\{x, y\}\}$
8: $G' \leftarrow (V(G), E(G) \cup F_0'), k' \leftarrow k - |F_0'|$
9: $D_0 \leftarrow \emptyset$
10: **while** (Rule 2 applies to (G', k') and a vertex $u \in V(G')$) **do**
11: $G' \leftarrow G' \setminus \{u\}, D_0 = D_0 \cup \{u\}$
12: $F_0 \leftarrow E(G') \cap F_0'$
13: **if** $k' < 0$ **or** $|V(G')| > 2k + k(2\sqrt{k} + 2d + 1)$ **then**
14: **return** a trivial NO instance.
15: **else**
16: **return** (G', k')

if (G, k) is. Finally by the safeness of Rule 2, instance (G', k') at Line 16 is a YES instance if and only if (G, k) is.

It remains to argue that we can safely return a trivial NO instance if $|V(G')| > 2k + k(2\sqrt{k} + 2d + 1)$, where G' is the graph at Line 13. Let us assume that (G', k') is a YES instance and let F be a set of edges such that $H = (V(G'), E(G') \cup F)$ is chordal and $|F| \leq k'$. Let V_F be the set of vertices incident to edges of F, and let V_{F_0} be the set of vertices incident to edges of F_0.

By Line 10 in Algorithm 1, Rule 2 is applied exhaustively, and thus by Lemma 4 every vertex of $V(G') \setminus (V_F \cup V_{F_0})$ is contained in $N_{G'}(x, y)$ for some edge $\{x, y\} \in F$. From the fact that G' is reduced with respect to Rule 5 (See Line 6 of Algorithm 1), we get that $|N_G(x, y)| = b < 2\sqrt{k} + 2d + 1$. To see this, observe that a clique on b vertices contains $b(b - 1)/2$ edges while $G[N_G(x, y)]$ contains at most db edges. Thus if $b \geq 2\sqrt{k} + 2d + 1$ then $b(b - 1)/2 - db = b/2(b - 1 - 2d) \geq ((2\sqrt{k} + 2d + 1)/2)(2\sqrt{k}) > k$ which is a contradiction to the fact that $\{x, y\} \notin F_0'$.

Notice that $|V_F| + |V_{F_0}| \leq 2k$, since (G', k') is a YES instance. In particular, notice that $N_{G'}(x, y) \setminus (V_F \cup V_{F_0}) \subseteq N_G(x, y)$. Summing over all edges in F, we get $|V(G')| \leq |V_F \cup V_{F_0}| + \sum_{\{x,y\} \in F} |N_G(x, y)| \leq 2k + k(2\sqrt{k} + 2d + 1)$. □

Observe that Rule 5—which is applicable only when the input graph is d-degenerate—adds an edge to the graph. The graph resulting from applying Rule 6—which adds the edge set F_0' found by applying Rule 5—is thus *not* necessarily d-degenerate. The graph output by Rule 6 can be modified to become d-degenerate while preserving the bound on its size, and this gives an $\mathcal{O}(k^{3/2})$ kernel for MINIMUM FILL-IN in d-degenerate graphs.

Theorem 2 MINIMUM FILL-IN *has a d-degenerate kernel of size $\mathcal{O}(k^{3/2})$ in d-degenerate graphs.*

Proof Since a 1-degenerate graph is a forest, and every forest has a fill-in of size zero—since the forest is chordal—we can assume without loss of generality that $d \geq 2$. Let (G, k) be an instance of MINIMUM FILL-IN where G is a d-degenerate graph. The kernelization algorithm applies Reduction Rule 6—Algorithm 1—to (G, k) to obtain an equivalent instance (G', k'). If (G', k') is the trivial NO instance returned by Line 14, then it is d-degenerate and its size is a constant, and the kernelization algorithm returns (G', k') itself as the kernel.

Now let (G', k') be a non-trivial instance returned by Line 16. Observe that G' is obtained from G by (i) deleting some vertices—Line 3,—(ii) adding edges F_0'—Line 6,—and (iii) deleting vertices D_0—Line 11. Edge set F_0 is defined in Line 12 as the set of edges in F_0' with both endpoints in $V(G')$: these are the edges added in Line 6 which survive in G'.

The kernelization algorithm constructs a new graph G'' from G' by doing the following for each edge $\{u, v\} \in F_0$: remove the edge $\{u, v\}$, add two new vertices a_{uv}, b_{uv}, and make both these vertices adjacent to both u and v. The algorithm returns (G'', k'') as the kernel, where $k'' = k' + |F_0|$. Let G_1 be the graph G' where edge set F_0 is removed.

To see that (G'', k'') satisfies all the requirements, note that G_1 is d-degenerate by the hereditary property of d-degenerate graphs, and $G' = (V(G'), E(G_1) \cup F_0)$. The graph G'' is d-degenerate since it can be obtained from G_1 by adding a sequence of vertices, each of degree two. Since each edge in F_0 corresponds to two new vertices in G'', $|V(G'')| = |V(G_1)| + 2|F_0| \leq 4k + k(2\sqrt{k} + 2d + 1)$.

It remains to argue that (G', k') is a YES instance if and only if (G'', k'') is. If (G', k') is a YES instance, then let F' be a fill-in of G' of size at most k', and let H' be the chordal graph obtained by adding the edges in F' to G'. Let $F'' = F_0 \cup F'$, and let H'' be the graph obtained by adding the edges in F'' to the graph G''. Observe that H'' can be obtained from the chordal graph H' by adding a sequence of vertices of degree two each, each of which is adjacent to the two end-points of some edge in F_0. It follows that H'' is chordal—any potential chordless cycle in H'' has to contain one of these new vertices, but every cycle passing through such a vertex has the respective edge in F_0 as a chord. Thus F'' is a fill-in of G'' of size at most $|F_0| + k' = k''$.

Conversely, let (G'', k'') be a YES instance. Observe that for each $\{u, v\} \in F_0$, the vertex set $S = \{u, v\}$ satisfies all the conditions of Proposition 2 in G''—S is a minimal a_{uv}, b_{uv}-separator, $G''[S]$ is missing the one edge which will make it a clique, and the vertex $a_{uv} \in V(G'') \setminus S$ is adjacent to every vertex in S. So there exists a *minimum* fill-in F'' of G'' such that $F_0 \subseteq F''$, and $|F''| \leq k''$. Let H'' be the chordal graph obtained by adding the edges in F'' to the graph G'', and let H' be the graph obtained by deleting all the vertices $\{a_{uv}, b_{uv} \mid \{u, v\} \in F_0\}$ from H''. Then H' can be obtained by adding the edges in $F' = F'' \setminus F_0$ to G' , and H' is chordal by the hereditary property of chordality. Thus F' is a fill-in of G' of size at most k'. □

5 A Subquadratic Kernel for H-Minor Free Graphs

It is known [30] that every H-minor free graph is d-degenerate for $d \leq \alpha h\sqrt{\log h}$, where $h = |V(H)|$ and $\alpha > 0$ is a constant. As we have already shown in Sect. 4, the application of Rule 6 on an instance (G, k) of MINIMUM FILL-IN where G is a d-degenerate graph results in an equivalent instance (G', k') where G' has $\mathcal{O}(k^{3/2})$ vertices. However, this G' is not necessarily H-minor free or d-degenerate. In Theorem 2, we show how to transform G' into a d-degenerate graph without significantly increasing its size. In this section, we show how to transform G' to an H-minor free graph, if the starting instance G is H-minor free.

The actual transformation is somewhat involved, and so we first present an informal overview of the procedure. Let (G, k) be an instance of MINIMUM FILL-IN where G is H-minor free, and let (G', k') be the instance obtained by applying Algorithm 1 on (G, k). For simplicity of explanation, we assume that no vertices are deleted from G by Line 3 of Algorithm 1. Since the property of being H-minor free is preserved when vertices are deleted, this assumption is harmless. So the graph G' is obtained from G by (i) adding the set F'_0 of edges, and (ii) deleting the set D_0 of vertices. Of these, the first operation is the only one which could possibly result in G' being *not* H-minor free. Thus, to convert G' into an equivalent H-minor free instance, we only need to take care of the edge set F'_0. More specifically, we only need to take care of the edges set F_0, which is the subset of F'_0 which survives in G' after the deletion of D_0.

A first attempt at this would be to delete the edges in F_0 from G', and to increment the parameter by $|F_0|$. The motivation for this is that since the edges in F_0 are forced in any fill-in of G (See Rule 5 and Line 6 of Algorithm 1), it is sufficient to remember their count. Unfortunately, this does not work since we *delete* vertices from G. Recall that the edges in F_0 are forced in any fill-in of G precisely because the end-points of each such edge has sufficiently many common neighbours. It may so happen that after deleting the set D_0 of vertices, this property no longer holds for some (or all) of the edges in F_0, and so this naive strategy is not safe : we cannot just forget all of F_0 and remember their count instead.

To deal with this, we use a somewhat more sophisticated, multi-step strategy. To simplify the discussion, we use $\tilde{G}$ to denote the graph at any point during this procedure, and $\tilde{k}$ to denote the parameter. We start by setting $\tilde{G} = G' \setminus F_0$, and $\tilde{k} = k'$. Observe that $\tilde{G}$ is H-minor free, but is not necessarily safe for the parameter $\tilde{k}$ (for the reason mentioned above). First, we check if there are edges in F_0 which still have the above property which forces them into any fill-in of $\tilde{G}$. These can be safely "forgotten", and to do this we increment $\tilde{k}$ by the number of such edges. We also remove these edges from F_0. Observe that the graph remains unchanged by this step. After this, we check if we can find a *distinct* vertex in D_0 for each edge in F_0, such that the vertex is a common neighbour of the end-points of the edge in G. If we can find such a "matching" set of vertices in D_0, then we add all the edges in F_0 to $\tilde{G}$ and return $(\tilde{G}, \tilde{k})$ as the kernel. This is clearly safe, and $\tilde{G}$ is H-minor free because each edge in F_0 can be thought of as replacing a path of length two (via a vertex of D_0) which was deleted from G.

If we cannot find such a "matching", then by Hall's Matching Theorem there exists a subset X of edges of the set F_0 which "see" in G a strictly smaller subset—say Y—of vertices from D_0. We find such a pair of sets X, Y. We first add the vertices in Y to $\tilde{G}$. Then we add to $\tilde{G}$ all those edges of G which are incident with vertices in Y whose other end points are present in $\tilde{G}$. Finally, we delete the set X from F_0 and the set Y from D_0, and set $\tilde{k} = \tilde{k} + |X|$. Observe that we restored in $\tilde{G}$ the neighbourhood of the edges in X. Therefore all these edges again become forced in any fill-in of $\tilde{G}$, and so this step is safe. Further, $\tilde{G}$ is a subgraph of G, and so is H-minor free.

We repeat the last two steps—finding a "matching" *or* a small neighbourhood—till all of F_0 is exhausted. This adds at most as many vertices to $\tilde{G}$ as the initial size of F_0. In the end we return $(\tilde{G}, \tilde{k})$ as the kernel.

We now make this more formal.

Theorem 3 *Let H be a fixed graph.* MINIMUM FILL-IN *has an H-minor free kernel of size $\mathcal{O}(k^{3/2})$ in H-minor free graphs.*

Proof Let (G, k) be an instance of MINIMUM FILL-IN where G is H-minor free, and let $(G', k_0 = k')$ be the instance obtained by applying Rule 6 on (G, k). If a trivial NO instance is returned by Rule 6, then we also return NO. In the remaining part we assume that Algorithm 1 returns from Line 16.

Let F_0 be the set of edges defined in Line 12 of Algorithm 1, and let D_0 be the vertex set computed by the algorithm by the time it reaches Line 12. Let c_1 be the number $|F_0'| - |F_0|$. Recall that F_0' is the set of all fill edges which *would be* added to G by Rule 5. D_0 is the set of vertices which become eligibile for deletion as per Rule 2 in the graph obtained by adding these fill edges (Line 8). F_0 is the subset of edges in F_0' which survive after the deletion of the vertices in D_0, and c_1 is the number of edges in F_0' which are deleted along with the vertices of D_0.

Recall that the property of being H-minor free is preserved by the deletion of vertices (or edges). Therefore, we assume without loss of generality that no vertex is deleted in Line 3 of Algorithm 1. Let $M_0 = (V_0, E_0)$ be the graph obtained from G' by removing all the edges in F_0. For $i \geq 1$, we create a sequence of graphs by applying additional reduction rules, such that all the graphs in the sequence are H-minor free and when none of the reduction rules applies, the resulting graph has size $\mathcal{O}(k^{3/2})$ and so it is the desired kernel. We construct the sequence inductively, starting from M_0. To proceed, we prove that for each $i \geq 0$ the tuple (M_i, F_i, D_i, k_i) has the following properties:

1. $M_i = (V_i, E_i)$ is H-minor free;
2. $|F_i| = |F_0| - i$;
3. $|V_i| = \mathcal{O}(k^{3/2})$;
4. $V(G) = V_i \cup D_i$;
5. graph $G_i = (V(G), E(G) \cup E_i)$ is H-minor free;
6. graph $T_i = (V_i, E_i \cup F_i)$ has a fill-in with k_i edges if and only if G has a fill-in of size at most k, and
7. $k = c_1 + |F_i| + k_i$.

Let us first argue that all seven properties hold for $i = 0$, which is the base case. Indeed, 1: Graph M_0 is obtained by deleting vertices D_0 from G, and thus is H-minor free. 2: $|F_0| = |F_0| - 0$. 3: Since V_0 is output by Rule 6 we have that $|V_0| = \mathcal{O}(k^{3/2})$. 4: Because $M_0 = G[V \setminus D_0]$, we have that $V(G) = V_0 \cup D_0$. 5: Since $E_0 \subseteq E(G)$, we get that the graph $G_0 = (V(G), E(G) \cup E_0)$ is H-minor free. 6: Since Rule 6 is safe, it follows that $T_0 = (V_0, E_0 \cup F_0)$ has a fill-in with $k_0 = k'$ edges if and only if G has a fill-in with k edges. 7: Since Rule 6 is safe we have that $c_1 + |F_0| + k_0 = |F_0'| - |F_0| + |F_0| + k_0 = |F_0'| + k_0 = k$.

For the induction step, for $i \geq 0$, we assume that all properties hold for (M_i, F_i, D_i, k_i). We construct a new tuple $(M_{i+1}, F_{i+1}, D_{i+1}, k_{i+1})$ from (M_i, F_i, D_i, k_i) by applying the three reduction rules below in the order they are given. The first rule increments the value of i. The second rule is applied only if the first cannot be applied, and recognizes kernels that can be returned directly. The third rule is applied only if none of the previous two can be applied, and ensures that the first rule can be applied again.

Reduction Rule 7 *For each edge $\{u, w\} \in F_i$ define $b = |N_{M_i}(u, w)|$. Let $\{u, w\}$ be an edge in F_i such that $(b/2)(b - 1 - 2d) > k$, where d is the degeneracy of the input graph G. Then $M_{i+1} = M_i$, $F_{i+1} = F_i \setminus \{\{u, w\}\}$, $k_{i+1} = k_i + 1$, $D_{i+1} = D_i$.*

Claim 3 *Rule 7 is safe; that is, it preserves all the seven properties.*

Proof

1. M_{i+1} is H-minor free, as $M_{i+1} = M_i$;
2. $|F_{i+1}| = |F_0| - (i + 1)$, as $|F_{i+1}| = |F_i| - 1$;
3. $|V_{i+1}|$ is $\mathcal{O}(k^{3/2})$, as $V_{i+1} = V_i$;
4. $V(G) = V_{i+1} \cup D_{i+1}$, as $V_{i+1} = V_i$ and $D_{i+1} = D_i$;
5. graph $G_{i+1} = (V(G), E(G) \cup E_{i+1})$ is H-minor free, as $E_{i+1} = E_i$;
6. graph $T_{i+1} = (V_{i+1}, E_{i+1} \cup F_{i+1})$ can be triangulated by the adding k_{i+1} edges if and only if G can be triangulated by the adding k edges, see the arguments below, and
7. $k = c_1 + |F_{i+1}| + k_{i+1}$ as $|F_{i+1}| = |F_i| - 1$ and $k_{i+1} = k_i + 1$.

The remaining claim (item 6) is that there is a fill-in of graph $T_{i+1} = (V_{i+1}, E_{i+1} \cup F_{i+1})$ with k_{i+1} edges if and only if the fill-in of G is at most k. Notice that $k_{i+1} \leq k$ and by Rule 5 there is no triangulation of T_{i+1} with at most k fill edges that do not add the edge $\{u, w\}$. Thus T_{i+1} can be triangulated by adding k_{i+1} edges if and only if T_i can be triangulated by adding k_i edges and the claim holds. □

Let us assume that Rule 7 cannot be applied to the tuple (M_i, F_i, D_i, k_i). Then for every edge $\{u, w\} \in F_i$ there exists at least one vertex $x \in D_i$ such that $x \in N_G(u) \cap N_G(w)$.

Construct a bipartite graph $B_i = (P_i, Q_i, Z_i)$, where there is a vertex $v_{uw} \in P_i$ for each edge $\{u, w\} \in F_i$, there is a vertex $x \in Q_i$ for each vertex $x \in D_i$, and there is an edge $\{v_{uw}, x\} \in Z_i$ if and only if $u, w \in N_G(x)$.

Reduction Rule 8 *If B_i has a matching saturating P_i, then return the instance $(G_H = (V_i, E_i \cup F_i), k_i)$.*

Claim 4 *Rule 8 is safe. That is, (G_H, k_i) is a* YES *instance if and only if (G, k) is a* YES *instance, graph G_H is H-minor free if G is H-minor free, and $|V(G_H)|$ is $\mathcal{O}(k^{3/2})$.*

Proof Let Y_i be a matching in B_i which saturates P_i. By the induction assumption (item 3), $|V_i|$ is $\mathcal{O}(k^{3/2})$, so $|V(G_H)|$ is $\mathcal{O}(k^{3/2})$. The fact that (G_H, k_i) is a YES instance if and only if (G, k) is a YES instance, follows from the induction assumption (point 6). Finally let us argue that G_H is H-minor free if and only if G is H-minor free.

Let $(u_1, w_1, x_1), (u_2, w_2, x_2), \dots, (u_{|F_i|}, w_{|F_i|}, x_{|F_i|})$ be triples such that $\{u_j, w_j\} \in F_i$ and $\{v_{u_j w_j}, x_j\}$ is an edge of the matching Y_i for $1 \le j \le |F_i|$. By the induction assumption (item 5), the graph $G_i = (V(G), E(G) \cup E_i)$ is H-minor free, where $V(G) = V_i \cup D_i$. Then graph $G_H = (V_i, E_i \cup F_i)$ is also H-minor free if we can argue that G_H is a minor of G_i.

To argue that G_H is a minor of G_i, we delete every vertex in $D_i \setminus \{x_1, x_2, \dots, x_{|F_i|}\}$ from G_i and delete all edges incident to x_j except $\{w_j, x_j\}$ and $\{u_j, x_j\}$ for $1 \le j \le |F_i|$. Then we contract edge $\{u_j, x_j\}$ for $1 \le j \le |F_i|$. The obtained graph $G_H = (V_i, E_i \cup F_i)$ is a minor of G_i. □

By Hall's Theorem [18], the bipartite graph B_i either has a matching saturating P_i, or there is a vertex set $P' \subset P_i$ such that $|N_{B_i}(P')| < |P'|$ in B_i. Furthermore, the set P' can be found in polynomial time. We now apply the following rule, and then apply Reduction Rule 7 to the resulting tuple.

Reduction Rule 9 *If there exists a subset P' of P_i such that $|N_{B_i}(P')| < |P'|$, define X to be the set of vertices in $V(G)$ such that $x \in X$ if and only if $x \in N_{B_i}(P')$. Then create a new tuple $(M_i' = G_i[V_i \cup X], F_i, D_i' = D_i \setminus X, k_i)$.*

Claim 5 *Rule 9 is safe, that is, all seven properties hold for the new tuple (M_i', F_i, D_i', k_i).*

Proof

1. M_i' is H-minor free, as M_i' is an induced subgraph of G_i which is H-minor free;
2. $|F_i| = |F_0| - i$, as F_i is unchanged;
3. $|V(M_i')| = \mathcal{O}(k^{3/2})$, see the argument below;
4. $V(G) = V(M_i') \cup D_i'$, as $V(G) = V(M_i) \cup D_i$ and $V(M_i') = V(M_i) \cup X$ and $D_i' = D_i \setminus X$;
5. graph $G_i = (V(G), E(G) \cup E_i)$ is H-minor free, as $E(G) \cup E_i$ is unchanged;
6. graph $T_i' = (V_i \cup X, E_i \cup F_i)$ can be triangulated by the addition of k_i edges if and only if G can be triangulated by the addition of k edges—see the argument below;
7. and $k = c_1 + |F_i| + k_i$ as F_i and k_i are unchanged.

(item 3) Let $F_{P'}$ be the subset of F_i such that $\{u, w\} \in F_{P'}$ if and only if $v_{uw} \in P'$. By adding vertex set X to M_i' we obtain the property that $N_{M_i'}(u, w) = N_G(u, w)$. The subsequent application of Reduction Rule 7 will thus remove all edges of $F_{P'}$ from F_i. Since $|X| < |P'|$ at most $|F_0|$ vertices are added over all executions of this rule.

(item 6) Let $x_1, x_2, \ldots, x_{|X|}$ be the vertices of $X = V(M_i') \setminus V(M_i)$ in the order they were removed by Rule 2. This means that the graph $(V(M_i'), E(M_i') \cup F_0)$ is an induced subgraph of the graph to which Rule 2 was applied when vertex x_1 was deleted. All edges in $F_0 \setminus F_i$ are contained in every minimum solution by Rule 7. Thus, it is still safe by Rule 2 to delete vertex x_1 from graph M_i'. Using this argument recursively on vertices $x_1, x_2, \ldots, x_{|X|}$ and by deleting the edges of F_0', the graph M_i is obtained again. Thus we can conclude that T_i' can be triangulated by the addition of k_i edges if and only if T_i can be triangulated by the addition of k_i edges. □

Observe that if Rule 9 adds back the vertex set X, then Rule 7 will be applied at most $|P'|$ times where $|X| < |P'|$. When tuple (M_j, F_j, D_j, k_j) where $j = |F_0|$ is reached, then our desired kernel is also obtained, since $F_j = \emptyset$. We can now conclude that a problem instance $(G_H = M_j, k_H = k_j)$ is H-minor free, $k_H \leq k$, $|V(G_H)| = \mathcal{O}(k^{3/2})$, and (G_H, k_H) is a YES instance if and only if (G, k) is, by the conditions of the induction.

Finally, it is easy to check that all the reduction rules can be implemented in polynomial time. □

6 Approximation Algorithms

As a byproduct of our kernelization algorithms, we obtain improved approximation algorithms for the MINIMUM FILL-IN problem on planar and H-minor free graphs. We need the following result of Chung and Mumford [8].

Proposition 5 [8] *Let H be a fixed graph, and let G be an n-vertex graph that is H-minor free. Then there is a triangulation G_T of G such that $|E(G_T)| = \mathcal{O}(n \log n)$, and such a triangulation can be found in polynomial time.*

Together with our improved kernels, this result yields approximate solutions for MINIMUM FILL-IN, with ratio $\mathcal{O}(\sqrt{k} \log k)$ for H-minor-free graphs, and with ratio $\mathcal{O}(\log k)$ for planar graphs.

Theorem 4 *Let k be the minimum size of a fill-in of a graph G. There is a polynomial time algorithm which computes a fill-in of G of size $\mathcal{O}(k \log k)$ if G is planar and of size $\mathcal{O}(k^{3/2} \log k)$ if G is H-minor free for some fixed graph H.*

Proof Let G be a planar graph. For each $k \in \{1, 2, \ldots, n^2\}$, in this order, we run the algorithm of Theorem 1 on (G, k), and compute the maximum value $k^\star$ of the parameter k for which the algorithm gives us a NO answer. This guarantees that there is no fill-in of G of size k^*. We then run the same algorithm on the instance $(G, k^\star + 1)$ to obtain a planar kernel G' on at most $6(k^\star + 1)$ vertices. Using Proposition 5, we obtain a fill-in of G' with at most $c(k^\star + 1) \log (k^\star + 1)$ edges, for some constant c. Using standard backtracking, the solution for G' can be transformed into a fill-in of G with $\mathcal{O}(k \log k)$ fill edges.

The arguments when G is an H-minor free graph are almost identical to the planar case. The only difference is that we use Theorem 3 instead, which provides us with an H-minor free kernel of size $\mathcal{O}(k^{3/2})$. □

It is not clear how to use the above technique to get a better approximation algorithm for d-degenerate graphs. It is known that there are infinite recursively enumerable classes $\mathcal{G}$ of graphs such that for every $G \in \mathcal{G}$, both the treewidth and the number of edges are $\theta(|V(G)|)$: for example, explicit constructions of bounded-degree expanders give such classes [17]. Thus, any minimal triangulation of such a graph would have a clique of size $\theta(|V(G)|)$ and thus the number of fill edges would be $\theta(|V(G)|^2)$ for this graph. Let k be the minimum size of a fill-in of such a d-degenerate graph. Then we can obtain a d-degenerate kernel of size $\mathcal{O}(k^{3/2})$ by Theorem 2. Since the best upper bound we have on the number of edges in a minimum fill-in of this kernel is quadratic, the best solution obtained from this kernel using the ideas in the proof of Theorem 4 would have $\mathcal{O}(k^3)$ edges. But this is worse than the best known approximate solution for general graphs, which has $\mathcal{O}(k^2)$ edges.

7 *NP*-completeness

Before concluding we shows that the Minimum Fill-In problem remains *NP*-complete on bipartite 2-degenerate graphs.

Theorem 5 *The* Minimum Fill-In *problem is NP-complete on bipartite* 2*-degenerate graphs.*

Proof Yannakakis [31] proved that deciding if a graph G can be triangulated by the addition of k edges is *NP*-complete. We will reduce from the Minimum Fill-In problem on general graphs to Minimum Fill-In on bipartite 2-degenerate graphs.

Let us define a new graph G' from G, where G' has one vertex v_w for each vertex $w \in V(G)$ and two vertices a_{uw}, b_{uw} for each edge $\{u, w\} \in E(G)$. For each edge $\{u, w\} \in E(G)$ edges $\{v_w, a_{uw}\}, \{v_w, b_{uw}\}, \{v_u, a_{uw}\}, \{v_u, b_{uw}\}$ are contained in $E(G')$. Let V_1, V_2 be a partitioning of $V(G')$ such that $v_w \in V_1$ for each $w \in V(G)$ and $c_{uw} \in V_2$ for each edge $\{u, w\} \in E(G)$ and each $c \in \{a, b\}$. Notice that graphs $G[V_1]$ and $G[V_2]$ are independent sets, and thus G' is a bipartite graph, where every vertex in V_2 is of degree 2. We can conclude that G' is bipartite 2-degenerate.

We will now argue that G can be triangulated by the addition of k edges if and only if G' can be triangulated by adding at most $k + |E(G)|$ edges.

In the forward direction, let F be an edge set such that $H = (V, E \cup F)$ is chordal and $|F| \leq k$. We will show that G' can be triangulated by the addition of $k + |E(G)|$ fill edges. For each vertex a_{uw} of G', add edge $\{v_u, v_w\}$ as a fill edge. This makes $N(c_{uw})$ into a clique for each $c \in \{a, b\}$, and thus by Rule 2 it is safe to delete a_{uw} and b_{uw}. Finishing this procedure results in graph G after adding $|E(G)|$ fill edges. By definition of edge set F, the graph $H = (V, E \cup F)$ is chordal and thus G' can be triangulated by the addition of $k + |E(G)|$ fill edges.

For the opposite direction we assume that G' can be triangulated by an edge set F' of size $k + |E(G)|$. Observe that v_u, v_w is a minimal a_{uw}, b_{uw}-separator that can be completed into a clique by the addition of a single edge, for each edge $\{u, w\} \in E(G)$. Furthermore $a_{uw}, b_{uw} \in N_{G'}(v_u)$, and thus by Proposition 2 there exists a minimum triangulation of G' where $\{v_u, v_w\}$ is a fill edge. We can safely add the edge $\{v_u, v_w\}$ as a fill edge in G' for each edge $\{u, w\} \in E(G)$. By Rule 2 vertex c_{uw} can safely be deleted for each edge $\{u, w\} \in E(G)$ and $c \in \{a, b\}$ since $N_{G'}(c_{uw})$ is a clique. By Proposition 2 and Rule 2 it follows that G can be triangulated by the addition of $(k + |E(G)|) - |E(G)|$ fill edges if and only if G' can be triangulated by adding $k + |E(G)|$ edges, and the result follows. □

8 Conclusion and Open Questions

In this paper we obtained new algorithms for Minimum Fill-In on several sparse classes of graphs. Specifically, we obtained a linear kernel for the problem on planar graphs and kernels of size $\mathcal{O}(k^{3/2})$ in H-minor free graphs and in graphs of bounded degeneracy. Using these kernels, we obtained approximation algorithms with ratios $\mathcal{O}(\log k)$ for planar graphs, and $\mathcal{O}(\sqrt{k}\log k)$ for H-minor free graphs. These results significantly improve known kernelization and approximation results for this problem. We note that for any $g \in \mathbb{N}$, the same set of reduction rules and essentially the same argument as for the planar case shows that Minimum Fill-In has a kernel of size $\mathcal{O}(k)$ in graphs of genus at most g.

Minimum Fill-in on general graphs is *NP*-complete [31]. However, it is a very old open question if the problem is NP-complete on planar graphs [8]. In Sect. 7 we show that the proof of Yannakakis [31] can be extended to bipartite 2-degenerate graphs.

We conclude with a number of open questions. The complexity of the problem on planar and on H-minor free graphs is still open. From the approximation perspective, we leave the possibility of obtaining an $o(\log k)$-approximation on planar graphs as an open problem.

From the perspective of kernelization, it would be very interesting to find out if there is a linear kernel for Minimum Fill-in on H-minor free graphs. We were not able to find any evidence that the existence of an $\mathcal{O}(k/\log k)$ kernel on planar graphs would contradict any complexity assumption. Can it be that the problem has a sublinear kernel?

Acknowledgements We thank our anonymous reviewers for pointing out a way to reduce the constant factor in Reduction Rule 4 from 22 to 6 with a simpler proof of Lemma 5, and for many other comments which helped in improving the presentation. F.V. Fomin acknowledges the support of the European Research Council (ERC) via grant “Rigorous Theory of Preprocessing”, reference 267959.

References

1. Agrawal, A., Klein, P.N., Ravi, R.: Cutting down on fill using nested dissection: provably good elimination orderings. In: George, A., Gilbert, J., Liu, J.W.H. (eds.) Graph Theory and Sparse Matrix Computation. IMA Volumes in Mathematics and Its Applications, vol. 56, pp. 31–55. Springer, Berlin (1993)
2. Alber, J., Fellows, M.R., Niedermeier, R.: Polynomial-time data reduction for dominating set. J. ACM **51**, 363–384 (2004)

3. Berry, A., Bordat, J.P., Heggernes, P., Simonet, G., Villanger, Y.: A wide-range algorithm for minimal triangulation from an arbitrary ordering. J. Algorithms **58**, 33–66 (2006)
4. Blair, J.R.S., Peyton, B.W.: An introduction to chordal graphs and clique trees. In: George, A., Gilbert, J., Liu, J.W.H. (eds.) Graph Theory and Sparse Matrix Computation. IMA Volumes in Mathematics and Its Applications, vol. 56, pp. 1–30. Springer, Berlin (1993)
5. Bodlaender, H.L., Fomin, F.V., Lokshtanov, D., Penninkx, E., Saurabh, S., Thilikos, D.M.: (Meta) kernelization. In: 50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, Atlanta, Georgia, USA, October 25–27, 2009, pp. 629–638. IEEE Computer Society, New York (2009)
6. Bodlaender, H.L., Heggernes, P., Villanger, Y.: Faster parameterized algorithms for minimum fill-in. Algorithmica **61**, 817–838 (2011)
7. Cai, L.: Fixed-parameter tractability of graph modification problems for hereditary properties. Inf. Process. Lett. **58**, 171–176 (1996)
8. Chung, F.R.K., Mumford, D.: Chordal completions of planar graphs. J. Comb. Theory, Ser. B **62**, 96–106 (1994)
9. Diestel, R.: Graph Theory, 3rd edn. Springer, Heidelberg (2005)
10. Dirac, G.A.: On rigid circuit graphs. Abh. Math. Semin. Univ. Hamb. **25**, 71–76 (1961)
11. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Springer, New York (1999)
12. Flum, J., Grohe, M.: Parameterized Complexity Theory. Texts in Theoretical Computer Science. An EATCS Series. Springer, Berlin (2006)
13. Fomin, F.V., Villanger, Y.: Subexponential parameterized algorithm for minimum fill-in. In: Rabani, Y. (ed.) Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA, Kyoto, Japan, January 17–19, 2012, pp. 1737–1746. SIAM, Philadelphia (2012)
14. Fomin, F.V., Philip, G., Villanger, Y.: Minimum fill-in of sparse graphs: kernelization and approximation. In: LIPIcs, Schloss Dagstuhl—Leibniz-Zentrum fuer Informatik, vol. 13, pp. 164–175 (2011)
15. Garey, M.R., Johnson, D.S.: Computers and Intractability, A Guide to the Theory of NP-Completeness. Freeman, New York (1979)
16. Golumbic, M.C.: Algorithmic Graph Theory and Perfect Graphs. Academic Press, New York (1980)
17. Grohe, M., Marx, D.: On tree width, bramble size, and expansion. J. Comb. Theory, Ser. B **99**, 218–228 (2009)
18. Hall, P.: On representatives of subsets. J. Lond. Math. Soc. **10**, 26–30 (1935)
19. Heggernes, P.: Minimal triangulations of graphs: a survey. Discrete Math. **306**, 297–317 (2006)
20. Kaplan, H., Shamir, R., Tarjan, R.E.: Tractability of parameterized completion problems on chordal, strongly chordal, and proper interval graphs. SIAM J. Comput. **28**, 1906–1922 (1999)
21. Kloks, T., Kratsch, D., Spinrad, J.: On treewidth and minimum fill-in of asteroidal triple-free graphs. Theor. Comput. Sci. **175**, 309–335 (1997)
22. Natanzon, A., Shamir, R., Sharan, R.: A polynomial approximation algorithm for the minimum fill-in problem. SIAM J. Comput. **30**, 1067–1079 (2000)
23. Niedermeier, R.: Invitation to Fixed-Parameter Algorithms. Oxford University Press, Oxford (2006)
24. Ohtsuki, T., Cheung, L.K., Fujisawa, T.: Minimal triangulation of a graph and optimal pivoting ordering in a sparse matrix. J. Math. Anal. Appl. **54**, 622–633 (1976)
25. Parra, A., Scheffler, P.: Characterizations and algorithmic applications of chordal graph embeddings. Discrete Appl. Math. **79**, 171–188 (1997)
26. Philip, G., Raman, V., Sikdar, S.: Solving dominating set in larger classes of graphs: FPT algorithms and polynomial kernels. In: Fiat, A., Sanders, P. (eds.) Algorithms—ESA 2009, 17th Annual European Symposium, Copenhagen, Denmark, September 7–9. Lecture Notes in Computer Science, vol. 5757, pp. 694–705. Springer, Berlin (2009)
27. Rose, D.J.: A graph-theoretic study of the numerical solution of sparse positive definite systems of linear equations. In: Read, R.C. (ed.) Graph Theory and Computing, pp. 183–217. Academic Press, New York (1972)
28. Rose, D.J., Tarjan, R.E., Lueker, G.S.: Algorithmic aspects of vertex elimination on graphs. SIAM J. Comput. **5**, 266–283 (1976)
29. Tarjan, R.E.: Decomposition by clique separators. Discrete Math. **55**, 221–232 (1985)
30. Thomason, A.: The extremal function for complete minors. J. Comb. Theory, Ser. B **81**, 318–338 (2001)
31. Yannakakis, M.: Computing the minimum fill-in is NP-complete. SIAM J. Algebr. Discrete Methods **2**, 77–79 (1981)