
ELSEVIER
Volume 80 Issue 7 November 2014 ISSN 0022-0000
JOURNAL OF COMPUTER AND SYSTEM SCIENCES
Interim Managing Editor
Michael Segal
Editors
Jin-Yi Cai
Andrew Chi-Chih Yao
Robert L. Constable
Danny Dolev
Ronald Fagin
Shafi Goldwasser
Georg Gottlob
Michael Rabin
Hartley Rogers, Jr.
Micha Sharir
Joseph Sifakis
Mihalis Yannakakis
L. A. Zadeh
Available online at www.sciencedirect.com
ScienceDirect

Journal of Computer and System Sciences 80 (2014) 1374–1383

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

Searching for better fill-in ☆

Fedor V. Fomin *, Yngve Villanger

Department of Informatics, University of Bergen, Bergen, Norway

ARTICLE INFO

Article history:
Received 25 January 2013
Received in revised form 18 February 2014
Accepted 4 April 2014
Available online 23 April 2014

Keywords:
Local search
Minimum fill-in
Parameterized complexity
Triangulation
Chordal graph

ABSTRACT

Minimum Fill-in is a fundamental and classical problem arising in sparse matrix computations. In terms of graphs it can be formulated as a problem of finding a triangulation of a given graph with the minimum number of edges. In this paper, we study the parameterized complexity of local search for the Minimum Fill-in problem in the following form: Given a triangulation H of a graph G, is there a better triangulation, i.e. triangulation with less edges than H, within a given distance from H? We prove that this problem is fixed-parameter tractable (FPT) being parameterized by the distance from the initial triangulation, by providing an algorithm that in time $f(k)|G|^{\mathcal{O}(1)}$ decides if a better triangulation of G can be obtained by swapping at most k edges of H. Our result adds Minimum Fill-in to the list of very few problems for which local search is known to be FPT.

1. Introduction

A graph is *chordal* (or triangulated) if every cycle of length at least four contains a chord, i.e. an edge between non-adjacent vertices of the cycle. The Minimum Fill-in problem (also known as Minimum Triangulation and Chordal Graph Completion) is to turn a given graph into a chordal by adding as few new edges as possible. 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 non-zero elements called *fills* can replace original zeros thus increasing storage requirements and running time needed to solve the system. The problem of finding an optimal elimination ordering minimizing the number of fill elements can be expressed as the Minimum Fill-in problem on graphs [45,46]. See also [9, 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 [3], artificial intelligence, and the theory of Bayesian statistics [8,22,33,51]. The survey of Heggernes [25] 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 [19] and it was proved to be NP-complete by Yannakakis [52]. While different approximation and parameterized algorithms for Minimum Fill-in were studied in the literature [2,5,7,8,17,27,39], in practice, to reduce the fill-in different *heuristic* ordering methods are commonly used. We refer to the recent survey of Duff and Bora [13] on the history and recent developments of fill-in reducing heuristics.

☆ Preliminary results of this paper appeared in the proceedings of STACS'13. The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007–2013)/ERC Grant Agreement No. 2679599.

* Corresponding author.
E-mail addresses: fomin@ii.uib.no (F.V. Fomin), yngvev@ii.uib.no (Y. Villanger).

http://dx.doi.org/10.1016/j.jcss.2014.04.010
0022-0000/

In this paper we study the following local search variant of the problem: Given a fill-in of a graph, is it possible to obtain a better fill-in by changing a small number of edges? An efficient local search algorithm could be used as a generic subroutine of almost every fill-in heuristic.

The idea of local search is to improve a solution by searching for a better solution in a neighborhood of the current solution, that is defined in a problem-specific way. For example, for the classic Traveling Salesman problem, the neighborhood of a tour can be defined as the set of all tours that differ from it in at most k edges, the so-called *k-exchange neighborhood* [34,43]. For inputs of size n, a naïve brute-force search of the k-exchange neighborhood requires $n^{\mathcal{O}(k)}$ time; this is infeasible in practical terms even for relatively small values of k. But is it possible to do better? Is it possible to solve local search problems in, say time $\tau(k) \cdot n^{\mathcal{O}(1)}$, for some function τ of k only? It has been generally assumed, perhaps because of the typical algorithmic structure of local search algorithms: "Look at *all* solutions in the neighborhood of the current solution ...", that finding an improved solution (if there is one) in a k-exchange neighborhood necessarily requires brute-force search of the neighborhood; therefore, verifying optimality in a k-exchange neighborhood requires $\Omega(n^k)$ time (see, e.g. [1, p. 339] or [29, p. 680]).

An appropriate tool to answer these questions is parameterized complexity. In the parameterized framework, for decision problems with input size n and a parameter k, the goal is to design algorithms with runtime $\tau(k) \cdot n^{\mathcal{O}(1)}$, where τ is a function of k alone. Problems having such algorithms are said to be *fixed-parameter tractable* (FPT). There is also a theory of hardness to identify parameterized problems that are probably not amenable to FPT algorithms, based on a complexity hypothesis similar to P $\neq$ NP. For an introduction to the field and more recent developments, see the books [12,15,40].

By making use of developments from parameterized complexity, it appeared that the complexity of local search is much more interesting and involved than it was assumed to be for a long time. While many k-exchange neighborhood search problems, like determining whether there is an improved solution in the k-exchange neighborhood for TSP, are $W[1]$-hard parameterized by k [36], it appears that for some problems FPT algorithms exist. For example, Khuller, Bhatia, and Pless [28] investigated the NP-hard problem of finding a feedback edge set that is incident to the minimum number of vertices. One of the results obtained in [28] is that checking whether it is possible to improve a solution by replacing at most k edges in an n-vertex graph can be done in time $\mathcal{O}(n^2 + n\tau(k))$, i.e., it is FPT parameterized by k. Similar results were obtained for many problems on planar graphs [14] and for the feedback arc set problem in tournaments [16]. Complexity of k-exchange problems for Boolean CSP and SAT was studied in [31,48]. The parameterized complexity of local search of different problems was investigated in [20,24,37,38,42]. However, most of these results exhibit the hardness of local search, and, as it was mentioned by Marx in [35], in most cases, the fixed-parameter tractability results are somewhat unexpected.

Our result. There are various neighborhoods considered in the literature for different problems. Since for the Minimum Fill-in problem the solution is determined by an edge subset, the following definition of the neighborhood comes naturally. For a pair of graphs $G = (V, E)$ and $G' = (V, E')$ on the same vertex set V, let $H(G, G')$ be $|E \triangle E'|$, i.e. the Hamming distance between the edge sets of E and E'. We say that G is a *neighbor of G' with respect to k-exchange neighborhood k-***ExN** if $H(G, G') \leqslant k$. Let $\mathcal{N}_k^{en}(G)$ be the set of neighbors of G with respect to k-**ExN**. For a given triangulation, i.e. a chordal supergraph H of graph G, we ask if there is a better triangulation of G within distance at most k from H. More precisely, we define the following variant of local search.

k-Local Search Fill-in (k-LS-FI)	**Parameter:** k
Input: A graph $G = (V, E)$, its triangulation $H = (V, E \cup F)$ and an integer $k > 0$.	
Question: Decide whether there is a triangulation $H' = (V, E \cup F')$ of G such that $H' \in \mathcal{N}_k^{en}(H)$ and $|F'| < |F|$.	

The main result of the paper is the following theorem.

Theorem 1. *k-LS-FI is FPT.*

The theorem is proved in several steps. Let a graph $G = (V, E)$ and its triangulation $H = (V, E \cup F)$ be an input of k-LS-FI. We refer to a graph $H' = (V, E \cup F') \in \mathcal{N}_k^{en}(H)$ with $|F'| < |F|$ as to a solution of k-LS-FI. We start from a simple criterion to identify edges of F that should be in every solution of k-LS-FI (Lemma 14). Based on this criterion, we can show that if a solution exists, i.e. G and H is a YES-instance of k-LS-FI, then there is a solution $H' = (V, E \cup F')$ such that the edges of $F \triangle F'$ "affect" at most $k(k+1)$ maximal cliques of H. This is done in Lemma 16. The next step is to identify the cliques of H that can be affected by the solution. In a chordal graph, the total number of different families containing at most $k(k+1)$ maximal cliques each, can be $n^{\Omega(k^2)}$. However, we design a procedure to generate at most $n2^{\mathcal{O}(k^5)}$ families of maximal cliques of H, each family containing at most $k(k+1)$ cliques, and such that at least one set of the family is a set of cliques affected by the solution. The procedure generating sets of affected maximal cliques is given in Lemma 19, and this is the most technical part of our algorithm. What remains to show is that for a given set of maximal cliques, we can construct in FPT time a solution of k-LS-FI affecting only these cliques.

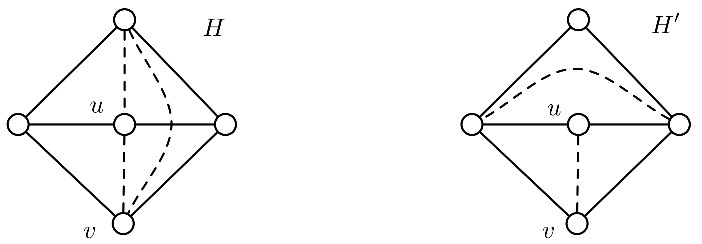

Fig. 1. In the instance of k-LS-FI, $k = 3$, the original edges of $G = (V, E)$ are solid lines, and the fill edges F are dashed lines. Graph $H = (V, E \cup F)$ is one of two minimal triangulations of $G = (V, E)$ and H' on the right side is a solution of the provided 3-LS-FI instance. However, graph H' is not a minimal triangulation of G as $H' \setminus uv$ is chordal and to obtain a minimal triangulation $H' \setminus uv$ from H one has to swap four edges.

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 in G. For a non-empty subset $W \subseteq V$, the subgraph of G induced by W is denoted by $G[W]$. We also use $G \setminus W$ to denote $G[V \setminus W]$. 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 $N[W] = N(W) \cup W$. We say that an edge uv of graph G is *contained* in set $X \subseteq V$, if $u, v \in X$. We refer to Diestel's book [10] for basic definitions from graph theory.

A *walk* is a sequence of vertices $v_1 v_2 \ldots v_\ell$ where $v_i v_{i+1} \in E(G)$ for $1 \leqslant i < \ell$. The walk is called a *path* if the vertices are distinct, and the path is called a *cycle* if $v_1 v_\ell \in E$. The path is referred to as *induced* if $G[\{v_1 v_2 \ldots v_\ell\}]$ only contains the edges of the walk, and the walk is an induced cycle if $v_1 v_\ell$ is the only non-walk edge. A *chord* of a cycle is an edge between two non-consecutive vertices of the cycle, thus induced cycles are chordless.

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.

A *triangulation* of graph $G = (V, E)$ is a chordal supergraph $H = (V, E \cup F)$ of G. For a triangulation $H = (V, E \cup F)$, we refer to edge set F as the set of *fill* edges. A triangulation H of graph G is called *minimal* if $H' = (V, E \cup F')$ is not chordal for any edge set $F' \subset F$, and H is a *minimum* triangulation if $H' = (V, E \cup F')$ is not chordal for every edge set F' such that $|F'| < |F|$. If H is a minimum triangulation of G, then $|F|$ is the minimum fill-in for G.

If chordal graph $H = (V, E \cup F)$ is not a minimal triangulation of $G = (V, E)$, then we can always find an edge $uv \in F$ such that $H \setminus uv$ is chordal. It is possible to check in linear time if the input graph is chordal [49], and thus in time $\mathcal{O}(|F|(|V| + |E \cup F|))$ one can check if H is a minimal triangulation of G. Hence if the input graph H is not a minimal triangulation of G, we can solve k-LS-FI in time $\mathcal{O}(|F|(|V| + |E \cup F|))$. In the remaining part of the paper, we assume that H is a *minimal* triangulation of G.

Even though we can always argue that the input chordal graph H is a minimal triangulation of G, we cannot ensure that every solution H' of the k-LS-FI problem is a minimal triangulation of G, see Fig. 1.

On the other hand, the following lemma ensures that we can seek for a solution which is a minimal triangulation of some supergraph of G and a subgraph of H. Because of the following lemma, we will be able to use nice properties of minimal triangulations in search of a better solution.

Lemma 2. *Let $H' = (V, E \cup F')$ be a solution of k-LS-FI with instance graphs $G = (V, E)$ and $H = (V, E \cup F)$. Then there is a solution $H'' = (V, E \cup F'')$ such that H'' is a minimal triangulation of $H_r = (V, E \cup (F \cap F'))$.*

Proof. Graph H' is chordal and is a supergraph of H_r, hence it is a triangulation of H_r. If H' was not a minimal triangulation of H_r, then removal of a non-empty subset of edges $S \subseteq F' \setminus (F \cap F')$ from H' results in a minimal triangulation $H'' = (V, E \cup F'')$ of H_r. Since $|F \triangle F''| < |F \triangle F'| \leqslant k$, we have that H'' is the required minimal triangulation. □

Vertex eliminations. A vertex of a graph is *simplicial* if its neighborhood is a clique. By the classical result of Fulkerson and Gross [18], a graph H is chordal if and only if it admits a *perfect elimination ordering*, i.e. vertex ordering $\pi : \{1, 2, \ldots, n\} \to V(G)$ such that for every $i \in \{1, 2, \ldots, n\}$, vertex $\pi(i)$ is simplicial in graph $H[\{\pi(i), \ldots, \pi(n)\}]$. Given a vertex ordering π of a graph G, we can construct a triangulation H of G such that π is a perfect elimination ordering for H. Triangulation H is obtained by the following *vertex elimination procedure* (also known as Elimination Game) [18,45]. A vertex elimination procedure takes as an input a vertex ordering π 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 3. *(See [41].) 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 vertex elimination procedure outputs H.*

For a triangulation H of G, the edges of H which are not edges of G are called *fill* edges. We will also need the following description of the fill edges introduced by vertex eliminations.

Proposition 4. *(See [47].) Let H be the chordal graph produced by the vertex elimination procedure from graph G according to an 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 $1 \leqslant i \leqslant \ell$.*

By the arguments used by Fulkerson and Gross [18] in combination with Ohtsuki et al. [41], we can reach the following conclusion.

Proposition 5 *(Folklore). Let H be a minimal triangulation of G and let $X \subseteq V$ be a clique of G. Then there exists ordering π such that vertices of X are the last vertices in π and the corresponding vertex elimination procedure outputs H.*

Minimal separators. Let u and v be two non-adjacent vertices of a graph G. A set of vertices $S \subseteq V$ is a *u, v-separator* if u and v are in different connected components of the graph $G[V \setminus S]$. We say that S is a *minimal u, v-separator* of G if no proper subset of S is a 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 contained in another one. If a minimal separator is a clique, we refer to it as to a *clique minimal separator*. In a chordal graph, each minimal separator is a clique minimal separator [11]. Also a chordal graph on n vertices contains at most n maximal cliques and $n - 1$ minimal separators [11].

A connected component C of $G \setminus S$ is a *full component* associated with S if $N(C) = S$. The following proposition is an exercise in [23].

Proposition 6 *(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 with S. In particular, S is a minimal separator if and only if there are at least two distinct full components associated with S.*

Two separators S and S' are *crossing* if S is a u, v-separator for a pair of vertices $u, v \in S'$, and S' is a u', v'-separator for some $u', v' \in S$.

Proposition 7. *(See [44].) Graph H is a minimal triangulation of G if and only if H can be obtained from G by completing a maximal set of pairwise non-crossing minimal separators into cliques.*

Proposition 8. *(See [30,44].) Let H be a minimal triangulation of G. Then every minimal separator in H is a minimal separator in G.*

For a minimal triangulation $H = (V, E \cup F)$ of G, Proposition 7 implies that for every edge $uv \in F$ there exists a minimal separator S of both G and H such that $u, v \in S$. We also use the following result.

Proposition 9. *(See [30,44].) Let H be a minimal triangulation of G. Then every full component C associated with a minimal separator S in H is also a full component associated with (minimal separator) S in G.*

The following proposition is folklore; see, e.g., [5].

Proposition 10. *(See [5].) Let $H = (V, E \cup F)$ be a minimal triangulation of $G = (V, E)$ and let $v_1v_2\ldots v_\ell$, $\ell \geqslant 4$, be a chordless cycle in G. Then either $v_2v_\ell \in F$, or $v_1v_i \in F$ for some $2 < i < \ell$.*

We also use the following result.

Proposition 11. *(See [30].) Let S be a minimal separator of G, and let G_S be the graph obtained from G by completing S into a clique. Let $C_1, C_2, \ldots, C_r$ be the connected components of $G \setminus S$. Then graph H obtained from G_S 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 $G_S[N[C_i]]$.*

Clique trees and tree decompositions. A *tree decomposition* TD_G of a graph $G = (V, E)$ is a pair (T, χ) consisting of a family χ of vertex subsets of V; the elements of χ are mapped bijectively onto the nodes of T such that $V = \bigcup_{X \in \chi} X$; for every $uv \in E$, $u, v \in X$ for some $X \in \chi$; and for every vertex $v \in V$ the set of elements of χ containing v induces a subtree of T. Often we abuse notation by not distinguishing elements of χ and nodes of T. Tree decompositions are strongly related to chordal graphs due to the following proposition.

Proposition 12. *(See [6,21,50].) Graph G is chordal if and only if there exists a tree decomposition (T, χ) of G such that every $X \in \chi$ is a maximal clique in G.*

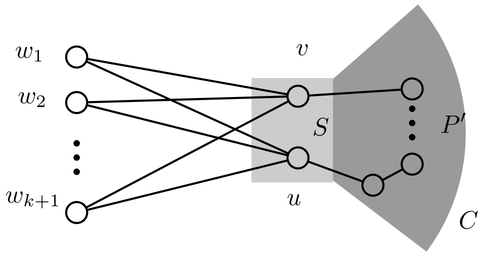

Fig. 2. Construction in the proof of Lemma 14.

Such a tree decomposition is referred to as a *clique tree* of G. It is well known that a clique tree of a chordal graph on n vertices and m edges can be constructed in $O(n+m)$ time [4]. Vertices of the clique tree will be referred to as *nodes* in order to distinguish them from the vertices of the graph. We also need the following result relating edges of a clique tree of a chordal graph and its minimal separators.

Proposition 13. *(See [6,26].) Let (T,χ) be a clique tree of a chordal graph G. Then S is a minimal separator of G if and only if $S = X_i \cap X_j$ for some edge $X_iX_j \in E(T)$.*

By slightly abusing notation, we often do not distinguish between edge X_iX_j of the clique tree T and the vertex set $S = X_i \cap X_j$.

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 [12] for further reading on parameterized complexity.

3. Local search

Immovable edges. Let $G=(V,E)$ be a graph and $H=(V,E\cup F)$ be a minimal triangulation of G. We say that an edge $e \in F$ is *immovable*, if for every triangulation $H'=(V,E\cup F') \in \mathcal{N}_k^{en}(H)$ we have $e \in F'$. In other words, each triangulation H' from the k-neighborhood of H must contain e. We need a sequence of results providing conditions enforcing edges to be immovable.

Lemma 14. *Let S be a minimal separator of a minimal triangulation $H=(V,E\cup F)$ of an n-vertex graph $G=(V,E)$, let C be a full component associated with S in H, and let $u,v \in S$ such that $uv \in F$ and $|(N_H(u)\cap N_H(v)) \setminus (C\cup S)| > k$. Then uv is an immovable edge. Moreover, one can check in time $\mathcal{O}(n^3)$ if an edge $uv \in F$ satisfies the above conditions and thus is immovable.*

Proof. Aiming for a contradiction, let us assume that $X = (N_H(u)\cap N_H(v)) \setminus (S\cup C)$, $|X| \geqslant k+1$, and that uv is not an immovable edge. By Propositions 8 and 9, S is a minimal separator in G and C is a full component associated with S in G. Let P be a path from u to v in G such that all internal vertices of P are in C. Because S is a separator in H, we have that in graph H no internal vertex of P is adjacent to a vertex from X. By our assumption, uv is not immovable and thus there is a solution H' of the k-LS-FI problem not containing uv. Because P is a path in G, the vertices of P induce a connected subgraph in H'. Let P' be a shortest path from u to v in $H'[V(P)]$. Path P' is a chordless path of length at least two. Observe now that for every vertex $w \in X$, adding w to P' creates a cycle of length at least 4, see Fig. 2. By Proposition 10, there exists an edge from w to some vertex of P' because otherwise H' is not chordal. As $|X| \geqslant k+1$, we have that $|E(H')\setminus E(H)| > k$, contradicting the assumption that H' is a solution. Thus, every minimal triangulation from the k-neighborhood of H should contain uv.

Let us now argue for the running time. The problem of verifying if an edge $uv \in F$ is immovable by the lemma, simply reduces to verifying if there exists a minimal separator S of H containing uv and a full component C associated with S such that $|(N_H(u)\cap N_H(v)) \setminus (C\cup S)| > k$. For a given pair S, C, it can be tested in $\mathcal{O}(n)$ time if $|(N_H(u)\cap N_H(v)) \setminus (C\cup S)| > k$. Chordal graph H contains $\mathcal{O}(n)$ minimal separators which can be enumerated in time $\mathcal{O}(n)$ [32]. There are at most n full components associated with each minimal separator, thus the total check can be performed in time $\mathcal{O}(n^3)$. $\square$

Lemma 14 yields the following lemma.

Lemma 15. *Let $H=(V,E\cup F)$ be a minimal triangulation of graph $G=(V,E)$ and let X_1 and X_2 be maximal cliques of H such that $|X_2 \setminus X_1| > k$. Then every edge of F contained in $X_1 \cap X_2$ is immovable.*

Proof. Let T be a clique tree of H and remember that each node of T represents a maximal clique of H. Let X' be the neighbor of X_1 on the unique path from X_1 to X_2 in T. By Proposition 13, $S = X_1 \cap X'$ is a minimal separator in H. Let us remark that $S \supseteq X_1 \cap X_2$. Let C be the full component of $H \setminus S$ associated with S containing $X_1 \setminus S$. For every edge $uv \in F$ such that $u, v \in X_1 \cap X_2$, we have that $u, v \in S$, and because X_2 is a clique, we have that every vertex from $X_2 \setminus (S \cup C)$ is adjacent to both u and v. Finally, $|(N_H(u) \cap N_H(v)) \setminus (C \cup S)| \geqslant |X_2 \setminus (S \cup C)| = |X_2 \setminus X_1| > k$. Now the proof of the lemma follows by Lemma 14. □

Lemma 16. *Let $H = (V, E \cup F)$ and $H' = (V, E \cup F') \in \mathcal{N}_k^{en}(H)$ be minimal triangulations of G. Then H has at most $k(k+1)$ maximal cliques containing both endpoints of some edge from $F \setminus F'$.*

Proof. We start the proof with the following claim.

Claim. *Every edge $uv \in F$ contained in more than $k+1$ maximal cliques of H is immovable.*

Proof. Let $uv \in F$ and let $X_1, X_2, \ldots, X_\ell$, $\ell \geqslant k+2$, be the maximal cliques containing both vertices u and v. In the clique tree T of H, the nodes corresponding to these maximal cliques induce a subtree T_{uv}. Tree T_u has at least $k+2$ nodes and thus should have a leaf. Without loss of generality, let us assume that X_1 is a leaf of T_u and that X_2 is the node adjacent to X_1. Then $S = X_1 \cap X_2$ is a minimal separator containing u and v. Because X_1 is a maximal clique, there is $x_1 \in X_1 \setminus S$ and the connected component of $H \setminus S$ containing x_1 is a full component associated with S. We call this component C. In graph $H \setminus (X_2 \setminus X_1)$, sets $X_2 \setminus X_1, \ldots, X_\ell \setminus X_1$ are maximal cliques containing u and v. Each of these cliques does not intersect C. Also because these cliques are maximal, there are at least $k+1$ vertices that are adjacent to both u and v and not contained in $C \cup S$. By Lemma 14, edge uv is immovable. This concludes the proof of the claim.

We proceed with the proof of the lemma. Because $H' \in \mathcal{N}_k^{en}(H)$, we have that none of the edges from $F \setminus F'$ is immovable. By the claim above, each such edge $e \in F \setminus F'$ is contained in at most $k+1$ maximal cliques of H. Since $|F \setminus F'| \leqslant k$, the lemma follows. □

Generating affected cliques. The following lemmata allow us to reduce the search space. As a result, we are able to generate at most $2^{\mathcal{O}(k^5)}$ sets of cliques, each set of size at most $k(k+1)$, such that if there is a solution to the problem, then there is also a solution that swaps edges only between vertices in one of the sets of maximal cliques.

Lemma 17. *Let $H = (V, E \cup F)$ be a minimal triangulation of G and let $H' = (V, E \cup F')$ be a solution of k-LS-FI. If H has a minimal separator S containing no edges of $F \setminus F'$, then there is a connected component C of $H \setminus S$ and a solution $H'' = (V, E \cup F'')$ of k-LS-FI such that every edge from $(F'' \setminus F) \cup (F \setminus F'')$ is contained in $N_H[C]$.*

Proof. Let $H_r = (V, E \cup (F \cap F'))$. Notice that H is a minimal triangulation of H_r. By Lemma 2, we can assume that H' is a minimal triangulation of H_r too. By Proposition 13, minimal separator S of H is a clique in H. Since S contains no edges of $F \setminus F'$, it is also a clique in H_r. Thus, by Proposition 8, S is a minimal clique separator of H_r, and by Proposition 11, S is also a minimal separator of H'.

Let $C_1, C_2, \ldots, C_p$ be the connected components of $H \setminus S$. By Proposition 11, $C_1, C_2, \ldots, C_p$ are exactly the connected components of $H_r \setminus S$ and of $H' \setminus S$. Thus there is no edge in $F \setminus F'$ having one endpoint in C_i and the other in C_j for $i \neq j$. Hence every edge from $F \setminus F'$ has one endpoint in C_i and the other in $C_i \cup S$, for some $1 \leqslant i \leqslant p$. Because H' is also a minimal triangulation of H_r, with similar arguments, we have that every edge from $F' \setminus F$ also has one endpoint in C_i and the other in $C_i \cup S$, for some $1 \leqslant i \leqslant p$. As $|F| > |F'|$, there exists $i \in \{1, \ldots, p\}$ such that $C_i \cup S$ contains more edges of $F \setminus F'$ than of $F' \setminus F$. Subgraph of H' induced by $C_i \cup S$ is chordal. Because S is a clique separator, this implies that graph H'' obtained from H by replacing $H[C_i]$ with $H'[C_i]$ is also chordal. Hence the required triangulation H'' can be obtained by only changing edges in $(F \setminus F') \cup (F' \setminus F)$ that are contained in $C_i \cup S$. □

By Lemma 17, we obtain the following lemma.

Lemma 18. *Let $H = (V, E \cup F)$ be a minimal triangulation of G and let T be a clique tree of H. If there is a triangulation $H' = (V, E \cup F') \in \mathcal{N}_k^{en}(H)$ with $|F'| < |F|$, then there is a triangulation $H'' = (V, E \cup F'') \in \mathcal{N}_k^{en}(H)$ with $|F''| < |F|$ such that the maximal cliques of H containing edges from $F \setminus F''$ induce a subtree of T.*

Proof. As long as the maximal cliques of H containing edges from $F \setminus F'$ do not induce a subtree of the clique tree T of H, there exists a minimal separator S of H such that no edges of $F \setminus F'$ are contained in S and there exist endpoints of edges in $F \setminus F'$ that are separated by S. By Lemma 17, we can obtain a new solution $H'' = (V, E \cup F'')$, where $|F''| < |F|$ and all endpoints of the edges in $F \setminus F''$ are contained in the same connected component of $H[V \setminus S]$. We repeat this cutting procedure until the maximal cliques of H containing edges from $F \setminus F''$ induce a subtree of the clique tree of H. □

By Lemma 18, if there is a solution of k-LS-FI, then there is also a solution where the maximal cliques of H containing edges deleted from H form a subtree of the clique tree of H. The next lemma gives an algorithm that in FPT time outputs at least one of such subtrees.

Lemma 19. *Let $H = (V, E \cup F)$ be a minimal triangulation of an n-vertex graph G. There is an algorithm that in time $\mathcal{O}(2^{\mathcal{O}(k^5)}n^2 + |F| \cdot n^3)$ outputs families $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_t$, $t \leqslant n2^{\mathcal{O}(k^5)}$, of sets of maximal cliques of H such that*

- *if there is a solution to k-LS-FI, then there exists a solution $H' = (V, E \cup F')$, $|F'| < |F|$, of k-LS-FI and a set $\mathcal{X} \in \{\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_t\}$ such that the maximal cliques of $\mathcal{X}$ induce a subtree of clique tree T of H and these maximal cliques are exactly the cliques containing edges of $F \setminus F'$.*

Proof. Let T be a clique tree of $H = (V, E \cup F)$. By Lemma 18, if H can be improved by k changes, then there is also an improvement $H' = (V, E \cup F')$ such that all maximal cliques of H containing non-changed edges, i.e. edges from $F \setminus F'$, induce a subtree T' of T. By Lemma 16, we can assume that T' contains at most $k(k+1)$ nodes. In what follows, we provide an algorithm listing different subtrees T' such that at least one the subtrees satisfies conditions of the lemma.

For every edge e of F, if the conditions of Lemma 14 hold, we mark e as immovable. For every minimal separator S of H, if all edges of F in $H[S]$ are marked as immovable then we say that S is an *immovable separator*.

We guess a maximal clique Z of H to be a node of T'. Because H has at most n maximal cliques, we make at most n guesses. By Lemma 17, we can assume that for every immovable separator S of the form $S = Z \cap Y$, where Z, Y are maximal cliques of H, at most one from Z and Y is in $\mathcal{X}$. Based on this, we perform the following preprocessing procedure pruning the clique tree T. We remove all edges of T corresponding to minimal separators marked as immovable. For convenience let us assume that T is the connected component containing the maximal clique Z after the pruning procedure.

Let us now search for the tree T' by starting in node Z. We root tree T' in Z and proceed recursively with the children of Z. When the algorithm is called on a node X of T', then we either add to T' some neighbors of X in T or conclude that no more neighbors of X can be added to T'.

We distinguish two cases.

Case 1. Degree of node X in T is at most $k(k+1) - 1$. In this case we simply try each of the $2^{k(k+1)-1}$ possible subsets of neighbors of X as the neighbors of X in T'.

Case 2. Degree of node X in T is at least $k(k+1)$. We select arbitrarily a set $\mathcal{W}$ of $k(k+1)$ children of X in T. Because T' has at most $k(k+1)$ nodes and X is already selected, we conclude that there is a solution $H' = (V, E \cup F')$ of k-LS-FI such that at least one maximal clique $X' \in \mathcal{W}$ does not contain edges from $F \setminus F'$. For each $X' \in \mathcal{W}$, we create a new subproblem by guessing X' to be a clique without edges from $F \setminus F'$ and marking all edges of F in X' as *forced-immovable*. We constrain our search only to solutions where all forced-immovable edges remain in the solution. From now by immovable edges we mean immovable and forced-immovable edges.

Let $S = X \cap X'$ and $W = X \setminus S$. We claim that $|W| \leqslant k$. Indeed, if it is not the case, then by Lemma 15, all edges of F contained in S are immovable, and thus clique X' has to be pruned by the preprocessing.

By our guess of X', no edge from $F \setminus F'$ is contained in S; hence every edge of $F \setminus F'$ contained in X has either both endpoints in W, or one endpoint in S and one endpoint in W. We already know that $|W| \leqslant k$ and thus there are at most $k(k-1)/2$ edges with both endpoints in W. For each subset of edges of F with both endpoints in W, we recursively create a new subproblem corresponding to the guess that all edges of this subset are in $F \setminus F'$. In other words, we branch on at most $2^{\mathcal{O}(k^2)}$ subproblems. For each edge $uv \in F$ with both endpoints in W selected to be contained in $F \setminus F'$ we add to the tree T' all maximal cliques containing both u and v. By the proof of Lemma 16, we know that each edge from $F \setminus F'$ is contained in at most $k+1$ maximal cliques.

In what follows we explain how to identify maximal cliques containing edges from $F \setminus F'$ with one endpoint in S and one endpoint in W. The difficulty here is that the size of S is not bounded by a function of k. By Proposition 7, every edge of F is contained in some minimal separator of H. Then for every unmarked edge $uw \in F$ with $u \in S$ and $w \in W$, there exists a maximal clique X_{uw} adjacent to X in T, $u, w \in X_{uw}$, and a minimal separator $S_{uw} = X \cap X_{uw}$. By Lemma 15, $|X \setminus X_{uw}| \leqslant k$, as otherwise uw would be marked as immovable. Because $S \subseteq X$, we have that for every $uw \in F \setminus F'$, $w \in W$, $u \in S$,

$$|S \setminus S_{uw}| \leqslant |X \setminus S_{uw}| = |X \setminus X_{uw}| \leqslant k. \tag{1}$$

For every vertex $w \in W$, we construct a set $\mathcal{Z}_w$ of maximal cliques of H such that for every $Y \in \mathcal{Z}_w$, there is $u \in S$ such that $X \cap Y$ is a minimal separator containing w and u, and $uw \in F$ is not marked as immovable. This set will include all neighbors of X in T which are not in T' and not separated by an immovable minimal separator from X. If $|\mathcal{Z}_w| \leqslant k$, we try adding each of the at most 2^k different subsets of $\mathcal{Z}_w$ to the tree T' and the to desired set $\mathcal{X}$. Because $|W| \leqslant k$, the total number of cases for all vertices in W is at most 2^{k^2}. Otherwise, if $|\mathcal{Z}_w| > k$, we take the first $k+1$ cliques $\{Z_1, Z_2, \dots, Z_{k+1}\}$ of $\mathcal{Z}_w$, and put $S_j = X \cap Z_j$, $1 \leqslant j \leqslant k+1$. By (1),

$$\left|\bigcap_{j=1}^{k+1} S_j\right| \geqslant |S| - k(k+1).$$

Notice that if $|S| \leqslant k(k+1)$, then $\bigcap_{j=1}^{k+1} S_j$ might be empty. We consider the following partition of S:

$$S_A = S \cap \bigcap_{j=1}^{k+1} S_j \quad \text{and} \quad S_B = S \setminus S_A.$$

Every edge $uw \in F$ with $u \in S_A$ and $w \in W$, is contained in at least $k+1$ maximal cliques $\{Z_1, Z_2, \dots, Z_{k+1}\}$, and thus is immovable by the claim in the proof of Lemma 16. Since $|S_B| \leqslant k(k+1)$ and $|W| \leqslant k$, we have that there are at most $k^2(k+1)$ edges $uw \in F$ with $u \in S_B$ and $w \in W$. Like before, for each subset E_F of edges of F with endpoints in W and S_B, we select all maximal cliques of H containing at least one edge from E_F. If some of the maximal cliques are not in $\mathcal{X}$, we add them to $\mathcal{X}$ and to T'. Otherwise, we conclude that no new neighbor of X can be added to $\mathcal{X}$ and T'. For each $w \in W$, there are at most $2^{k(k+1)}$ possible subsets of edges of F and thus we branch on at most $2^{k^2(k+1)}$ different subproblems. The algorithm stops either when $|\mathcal{X}| > k(k+1)$ or when we cannot add any new node to T'. This completes the description of the algorithm.

On the correctness of the algorithm. The algorithm enumerates all subtrees of T with at most $k(k+1)$ nodes and such that every node, i.e. a maximal clique, contains at least one non-immovable edge and cannot be separated from the root by an immovable separator. By Lemma 18, we know that if G and H is a YES-instance of the problem, then there is a solution such that the removed fill edges of H are covered by at most $k(k+1)$ maximal cliques of H satisfying the properties of the subtrees generated by the algorithm. The number of maximal-clique families generated by the recursive algorithm is proportional to the number of guesses on the root of T', which is n, times the number of leaves in the "branching" tree corresponding to the number of recursive calls. The maximum degree of the branching tree is $2^{\mathcal{O}(k^3)}$, corresponding to the branching on all possible edge subsets in Case 2. The depth of the recursion is at most $2k(k+1)$—at every call we either add a new clique to the set or decide that we cannot add any new neighbor to a node, and thus either increase the number of cliques in $\mathcal{X}$ or the number of "non-extendible" nodes in T'; both numbers are at most $k(k+1)$. Hence we conclude that the number of generated cliques is $n \cdot (2^{\mathcal{O}(k^3)})^{2k(k-1)} = n \cdot 2^{\mathcal{O}(k^5)}$.

Let us now argue for the running time. By Lemma 14, all immovable edges can be marked in time $\mathcal{O}(|F| \cdot n^3)$. Graph H is chordal and thus contains at most $n-1$ minimal separators. For every minimal separator S of H, we check in $\mathcal{O}(|F|)$ time if all edges of F in $H[S]$ are marked as immovable. Thus all immovable separators can be identified in time $\mathcal{O}(|F| \cdot n^3)$. Every chordal graph has at most n maximal cliques, thus with guessing the initial maximal clique of T', we have the following "polynomial" summand $\mathcal{O}(n + |F| \cdot n^3) = \mathcal{O}(|F| \cdot n^3)$ in the running time of the algorithm. For the "exponential" summand, we already observed that the number of instances generated by the algorithm is at most $n2^{\mathcal{O}(k^5)}$. In each of the recursive calls, we need additional time $\mathcal{O}(n)$ in Case 2 to go through all cliques containing a given edge. Thus the total running time of the algorithm is $\mathcal{O}(2^{\mathcal{O}(k^5)} n^2 + |F| \cdot n^3)$. □

Final step.

By Lemma 19, we are able to compute at least one of the subtrees of the clique tree of H that consists of maximal cliques containing all edges of H that will be removed in a better triangulation. We are ready to prove the main result about k-LS-FI, Theorem 1.

Proof of Theorem 1. To prove the theorem, we show that given a minimal triangulation $H = (V, E \cup F)$ of an n-vertex graph $G = (V, E)$, searching for a better triangulation in the k-exchange neighborhood of H can be performed in time $\mathcal{O}(2^{\mathcal{O}(k^5)} n^4 + |F| \cdot n^3)$.

Let T be a clique tree of H. We use Lemma 14 to mark some edges of F as immovable. We also mark minimal separators of H containing only immovable edges from F as immovable. We use the algorithm from Lemma 19 to output at most $n2^{\mathcal{O}(k^5)}$ families $\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_t$ of maximal cliques of $H = (V, E \cup F)$ such that:

- If pair G and H is a YES-instance of k-LS-FI, then there is a triangulation of G, $H' = (V, E \cup F') \in \mathcal{N}_k^{en}(H)$ with $|F'| < |F|$ such that at least one $\mathcal{X}_i$ consists of all cliques containing both endpoints for some edge of $F \setminus F'$;
- Each set $\mathcal{X}_i$ contains at most $k(k+1)$ maximal cliques of H;
- For every set $\mathcal{X}_i$, no two maximal cliques from $\mathcal{X}_i$ can be separated by an immovable separator.

For set $\mathcal{X}_i$, $1 \leqslant i \leqslant t$, we define H_i to be the induced subgraph of H induced by the vertices of cliques from $\mathcal{X}_i$. Let S be a minimal separator of H_i. By Lemma 15, for every pair of intersecting maximal cliques $X_1, X_2 \in \mathcal{X}_i$, we have that $|X_1 \setminus X_2| < k$. Hence, graph H_i contains at most $|S| + k^2(k+1)$ vertices as the whole subtree can be reduced to two maximal cliques whose intersection is S by recursively removing leaf cliques, and each of them having at most $k-1$ private vertices. We also define G_i to be the induced subgraph of G induced by the vertices of cliques from $\mathcal{X}_i$. Then G_i also has at most $|S| + k^2(k+1)$ vertices.

Let $\mathcal{C}$ be the set of all maximal cliques of H. By Lemmata 17 and 19, the search of a solution boils down to the search in the k-exchange neighborhood of H for a better triangulation $H' = (V, E \cup F')$ which satisfies, for some i, $1 \leqslant i \leqslant t$, the following additional condition: no maximal clique $C \in \mathcal{C} \setminus \mathcal{X}_i$ contains any edges from $F \setminus F'$ and any edges from $F' \setminus F$. The latter is trivial as edges of $F' \setminus F$ are not present in H.

Let G'_i be the graph obtained from G_i by adding edges of H_i marked as immovable and all edges of $F \cap E(H_i)$ which are contained in maximal cliques of $\mathcal{C} \setminus \mathcal{X}_i$. We show how to find a better triangulation of G'_i.

By Proposition 3, every minimal triangulation of G'_i corresponds to a minimal elimination ordering of G'_i. In graph G'_i, there are at most $k^2(k+1)$ vertices outside S. Thus in every elimination ordering, there are at most $k^2(k+1)$ vertices preceding the first vertex of S. We try all possible subsets of $V(G'_i) \setminus S$ and their permutations for a possible prefix in this ordering. Thus we try at most $2^{k^2(k+1)}(k^2(k+1))!$ ordered subsets. For every prefix π, we guess also the first vertex $v \in S$ which goes after π. So in total we try at most $n \cdot 2^{k^2(k+1)}(k^2(k+1))!$ ordered subsets. Let Y be the subset of vertices of S which are either adjacent to v or reachable from v through the vertices of the prefix. By Proposition 4, set Y is a clique in any triangulation obtained by an ordering extending πv. Let $Z = S \setminus Y$. If $|Z| > k$, then we made a wrong guess on the prefix π because at least $k+1$ edges incident to v have to be deleted, and this prefix cannot produce a triangulation in the k-exchange neighborhood of H_i.

Hence we assume that $|Z| \leqslant k$. By eliminating vertices of π and v first, it follows by Proposition 5, that there exists a vertex ordering such that the corresponding vertex elimination procedure outputs a minimal triangulation and such that the vertices of Y are the last vertices in this ordering. Thus there is a minimal elimination ordering producing the minimum fill-in of the form $\pi v Z Y$. As we have already shown, there are at most $2^{k^2(k+1)}(k^2(k+1))!$ ways to select the ordered prefix π, and at most n ways to select $v \in S$. As far as π and v are fixed, there is a unique way to define Y and Z. There are at most $k!$ permutations of Z and each of the permutations of Y is fine for us. Thus in total, there are at most $n \cdot 2^{k^2(k+1)}(k^2(k+1))!k! = 2^{\mathcal{O}(k^3 \log k)}n$ permutations. For each of such permutations, the corresponding vertex elimination procedure outputs a chordal supergraph of G'_i. If at least one of these chordal graphs H'_i is in $\mathcal{N}^{en}_k(H_i)$, then we output $H' = (V, E \cup (F \setminus E(H_i)) \cup E(H'_i))$. By Proposition 11, H' is chordal and thus H' is a triangulation of G with less than $|F|$ fill-in edges.

If for every i, $1 \leqslant i \leqslant t$, the minimum triangulation $H'_i \notin \mathcal{N}^{en}_k(H_i)$, then we conclude that the pair G and H is a NO-instance of the problem, and thus there is no better triangulation of G in the k-exchange neighborhood of H.

By Lemma 19, it takes time $\mathcal{O}(2^{\mathcal{O}(k^5)}n^2 + |F| \cdot n^3)$ to generate all subsets of set $\mathcal{X}$ and there are $2^{\mathcal{O}(k^5)}n$ such subsets. For each of the subsets consisting of at most $k(k+1)$ maximal cliques, a separator S can be found in $\mathcal{O}(n^2)$ time. For each set, we try $2^{\mathcal{O}(k^3 \log k)}n$ permutations, resulting in $2^{\mathcal{O}(k^5)}n \cdot 2^{\mathcal{O}(k^3 \log k)}n = 2^{\mathcal{O}(k^5)}n^2$ different elimination orderings. Finally, for each ordering, the corresponding triangulation can be computed in $\mathcal{O}(n^2)$ time. Thus, the total running time is $\mathcal{O}(2^{\mathcal{O}(k^5)}n^4 + |F| \cdot n^3)$. $\square$

4. Conclusion and open problems

The main result of this paper is that k-LS-FI is FPT. Since only a very few search problems are known to be FPT, we find it very interesting to explore what general properties of problems and exchange neighborhoods are responsible for such phenomena. Another natural question is about the running time of the algorithm. The worst case upper bound on the running time of our algorithm makes the result of the paper mainly of theoretical importance. However, the common story about improvements of FPT algorithms is that with more work and new ideas, these algorithms can be made practical.[1] Very recently, it was shown that the parameterized version of Minimum Fill-in is solvable in subexponential $2^{o(k)}n^{\mathcal{O}(1)}$ time. Can it be that k-LS-FI is solvable in time $\mathcal{O}(2^{o(k)}n^c)$ for some small constant c? Combined with other fill-in reducing heuristics, such an algorithm would be of real practical importance.

References

[1] E.H.L. Aarts, J.K. Lenstra, Local Search in Combinatorial Optimization, Princeton University Press, 1997.
[2] A. Agrawal, P.N. Klein, R. Ravi, Cutting down on fill using nested dissection: provably good elimination orderings, Graph Theory and Sparse Matrix Computation 56 (1993) 31–55.
[3] C. Beeri, R. Fagin, D. Maier, M. Yannakakis, On the desirability of acyclic database schemes, J. ACM 30 (1983) 479–513.
[4] J.R.S. Blair, B.W. Peyton, An introduction to chordal graphs and clique trees, in: Graph Theory and Sparse Matrix Computations, in: IMA Vol. Math. Appl., vol. 56, Springer, 1993, pp. 1–30.
[5] H. Bodlaender, P. Heggernes, Y. Villanger, Faster parameterized algorithms for minimum fill-in, Algorithmica 61 (2011) 817–838.
[6] P. Buneman, A characterization of rigid circuit graphs, Discrete Math. 9 (1974) 205–212.
[7] L. Cai, Fixed-parameter tractability of graph modification problems for hereditary properties, Inf. Process. Lett. 58 (1996) 171–176.

[1] Parameterized complexity community wiki contains different examples of running time improvements at http://fpt.wikidot.com/fpt-races.

[8] F.R.K. Chung, D. Mumford, Chordal completions of planar graphs, J. Comb. Theory, Ser. B 62 (1994) 96–106.
[9] T.A. Davis, Direct Methods for Sparse Linear Systems, Fundamentals of Algorithms, vol. 2, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2006.
[10] R. Diestel, Graph Theory, third ed., Grad. Texts Math., vol. 173, Springer-Verlag, Berlin, 2005.
[11] G.A. Dirac, On rigid circuit graphs, Abh. Math. Semin. Univ. Hamb. 25 (1961) 71–76.
[12] R.G. Downey, M.R. Fellows, Parameterized Complexity, Springer-Verlag, New York, 1999.
[13] I.S. Duff, B. Ucar, Combinatorial problems in solving linear systems, in: Combinatorial Scientific Computing, No. 09061, in: Dagstuhl Seminar Proceedings, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Germany, 2009.
[14] M.R. Fellows, F.V. Fomin, D. Lokshtanov, F.A. Rosamond, S. Saurabh, Y. Villanger, Local search: is brute-force avoidable?, J. Comput. Syst. Sci. 78 (2012) 707–719.
[15] J. Flum, M. Grohe, Parameterized Complexity Theory, Springer-Verlag, Berlin, 2006.
[16] F.V. Fomin, D. Lokshtanov, V. Raman, S. Saurabh, Fast local search algorithm for weighted feedback arc set in tournaments, in: Proceedings of the 24th AAAI Conference on Artificial Intelligence, AAAI 2010, AAAI Press, 2010, pp. 65–70.
[17] F.V. Fomin, Y. Villanger, Subexponential parameterized algorithm for minimum fill-in, SIAM J. Comput. 42 (2013) 2197–2216.
[18] D.R. Fulkerson, O.A. Gross, Incidence matrices and interval graphs, Pac. J. Math. 15 (1965) 835–855.
[19] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and Company, New York, 1979.
[20] S. Gaspers, E.J. Kim, S. Ordyniak, S. Saurabh, S. Szeider, Don't be strict in local search!, in: Proceedings of the 26th AAAI Conference on Artificial Intelligence, AAAI-12, AAAI Press, 2012, pp. 486–492.
[21] F. Gavril, The intersection graphs of subtrees in trees are exactly the chordal graphs, J. Comb. Theory, Ser. B 16 (1974) 47–56.
[22] D. Geman, Random fields and inverse problems in imaging, in: École d'été de Probabilités de Saint-Flour XVIII—1988, in: Lect. Notes Math., vol. 1427, Springer, Berlin, 1990, pp. 113–193.
[23] M.C. Golumbic, Algorithmic Graph Theory and Perfect Graphs, Academic Press, New York, 1980.
[24] J. Guo, S. Hartung, R. Niedermeier, O. Suchý, The parameterized complexity of local search for TSP, more refined, Algorithmica 67 (2013) 89–110.
[25] P. Heggernes, Minimal triangulations of graphs: a survey, Discrete Math. 306 (2006) 297–317.
[26] C.-W. Ho, R. Lee, Counting clique trees and computing perfect elimination schemes in parallel, Inf. Process. Lett. 31 (1989) 61–68.
[27] H. Kaplan, R. Shamir, R.E. Tarjan, Tractability of parameterized completion problems on chordal, strongly chordal, and proper interval graphs, SIAM J. Comput. 28 (1999) 1906–1922.
[28] S. Khuller, R. Bhatia, R. Pless, On local search and placement of meters in networks, SIAM J. Comput. 32 (2003) 470–487.
[29] J. Kleinberg, E. Tardos, Algorithm Design, Addison–Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.
[30] T. Kloks, D. Kratsch, J. Spinrad, On treewidth and minimum fill-in of asteroidal triple-free graphs, Theor. Comput. Sci. 175 (1997) 309–335.
[31] A. Krokhin, D. Marx, On the hardness of losing weight, ACM Trans. Algorithms 8 (2012), Article No. 19, 18 pp.
[32] P.S. Kumar, C.E.V. Madhavan, Minimal vertex separators of chordal graphs, Discrete Appl. Math. 89 (1998) 155–168.
[33] S.L. Lauritzen, D.J. Spiegelhalter, Local computations with probabilities on graphical structures and their application to expert systems, J. R. Stat. Soc. B 50 (1988) 157–224.
[34] S. Lin, B.W. Kernighan, An effective heuristic algorithm for traveling-salesman problem, Oper. Res. 21 (1973) 498–516.
[35] D. Marx, Local search, Parameterized Complexity Newsletter 3 (2008) 7–8.
[36] D. Marx, Searching the k-change neighborhood for TSP is W[1]-hard, Oper. Res. Lett. 36 (2008) 31–36.
[37] D. Marx, I. Schlotter, Parameterized complexity and local search approaches for the stable marriage problem with ties, Algorithmica 58 (2010) 170–187.
[38] D. Marx, I. Schlotter, Stable assignment with couples: parameterized complexity and local search, Discrete Optim. 8 (2011) 25–40.
[39] A. Natanzon, R. Shamir, R. Sharan, A polynomial approximation algorithm for the minimum fill-in problem, SIAM J. Comput. 30 (2000) 1067–1079.
[40] R. Niedermeier, Invitation to Fixed-Parameter Algorithms, Oxford University Press, 2006.
[41] T. Ohtsuki, L.K. Cheung, T. Fujisawa, Minimal triangulation of a graph and optimal pivoting ordering in a sparse matrix, J. Math. Anal. Appl. 54 (1976) 622–633.
[42] S. Ordyniak, S. Szeider, Parameterized complexity results for exact Bayesian network structure learning, J. Artif. Intell. Res. 46 (2013) 263–302.
[43] C.H. Papadimitriou, K. Steiglitz, On the complexity of local search for the traveling salesman problem, SIAM J. Comput. 6 (1977) 76–83.
[44] A. Parra, P. Scheffler, Characterizations and algorithmic applications of chordal graph embeddings, Discrete Appl. Math. 79 (1997) 171–188.
[45] S. Parter, The use of linear graphs in Gauss elimination, SIAM Rev. 3 (1961) 119–130.
[46] D.J. Rose, A graph-theoretic study of the numerical solution of sparse positive definite systems of linear equations, in: R.C. Read (Ed.), Graph Theory and Computing, Academic Press, New York, 1972, pp. 183–217.
[47] D.J. Rose, R.E. Tarjan, G.S. Lueker, Algorithmic aspects of vertex elimination on graphs, SIAM J. Comput. 5 (1976) 266–283.
[48] S. Szeider, The parameterized complexity of k-flip local search for SAT and MAX SAT, Discrete Optim. 8 (2011) 139–145.
[49] R.E. Tarjan, M. Yannakakis, Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs, SIAM J. Comput. 13 (1984) 566–579.
[50] J.R. Walter, Representations of rigid cycle graphs, PhD thesis, Wayne State University, 1972.
[51] S. Wong, D. Wu, C. Butz, Triangulation of Bayesian networks: a relational database perspective, in: Rough Sets and Current Trends in Computing, in: Lect. Notes Comput. Sci., vol. 2475, Springer, 2002, pp. 389–396.
[52] M. Yannakakis, Computing the minimum fill-in is NP-complete, SIAM J. Algebr. Discrete Methods 2 (1981) 77–79.