On Generating Triangle-Free Graphs

Daniel Brügmann [1] Christian Komusiewicz [2] Hannes Moser [3]

Institut für Informatik, Friedrich-Schiller-Universität Jena,
Ernst-Abbe-Platz 2, D-07743 Jena, Germany
daniel.bruegmann@uni-jena.de, {ckomus,moser}@minet.uni-jena.de

Abstract

We show that the problem to decide whether a graph can be made triangle-free with at most k edge deletions remains NP-complete even when restricted to planar graphs of maximum degree seven. In addition, we provide polynomial-time data reduction rules for this problem and obtain problem kernels consisting of $6k$ vertices for general graphs and $11k/3$ vertices for planar graphs.

Keywords: NP-complete problem, parameterized algorithmics, kernelization.

1 Introduction

The problem of destroying all triangles of a graph by edge deletions (also see [2,5]) can be formulated as follows.

Triangle Edge Deletion
Input: An undirected graph G, a nonnegative integer k.
Question: Can we transform G, by deleting $\leq k$ edges, into a triangle-free graph?

[1] Student of the Carl-Zeiss-Gymnasium Jena, Erich-Kuithan-Str. 7, D-07743 Jena.
[2] Supported by a PhD fellowship of the Carl-Zeiss-Stiftung.
[3] Supported by the Deutsche Forschungsgemeinschaft, project AREG, NI 369/9.

Unfortunately, Triangle Edge Deletion is NP-complete [9], destroying hope for polynomial-time algorithms. We show that it remains NP-complete even when restricted to planar graphs of maximum degree seven (Section 2). This motivates our subsequent study of fixed-parameter algorithms [7] for this problem (Section 3). In particular, we focus on kernelization [4,7], that is, we devise efficient (polynomial-time) and effective (yielding an instance whose size is bounded by the parameter) data reduction rules. For general graphs, we show that in $O(m\sqrt{m})$ time we can create an equivalent instance consisting of at most $6k$ vertices. For planar graphs, we strengthen this result by showing that we can create an instance comprising at most $11k/3$ vertices, although our algorithm requires $O(k \cdot n\sqrt{n})$ time

Related work. As observed by Gramm et al. [3], one can solve Triangle Edge Deletion by reducing it to the 3-Hitting Set problem. This reduction can be performed in $O(m\sqrt{m})$ time, which is the time needed for listing all triangles of a graph [2]. In combination with the current fastest 3-Hitting Set algorithm [8], this approach leads to a running time of $O(2.076^k + m\sqrt{m})$. It also follows from this reduction that Triangle Edge Deletion admits a problem kernel of $O(k^2)$ vertices, since we can apply a kernelization algorithm for 3-Hitting Set [1] and then reduce the resulting "kernelized" instance back to Triangle Edge Deletion.[4]

Due to the lack of space, some proofs are omitted.

2 NP-Completeness on Planar Graphs

In this section, we strengthen the hardness result for Triangle Edge Deletion [9], showing that the problem remains NP-hard even when restricted to planar graphs of maximum degree seven. To this end, we describe a polynomial-time many-one reduction from the NP-complete Vertex Cover problem:

Input: An undirected graph $G = (V, E)$, a nonnegative integer k.
Question: Is there a vertex set $S \subseteq V$ of size $\leq k$ such that $G[V \setminus S]$ has no edges?

Vertex Cover remains NP-complete even if the input graph is planar and cubic.Using this, we prove the NP-hardness of Triangle Edge Deletion on planar graphs, describing a reduction from Vertex Cover on planar

[4] In general, we cannot achieve a parameter-preserving reduction from 3-Hitting Set to Triangle Edge Deletion. However, the used kernelization produces an induced kernel [1], yielding an induced subgraph of the original instance of Triangle Edge Deletion.

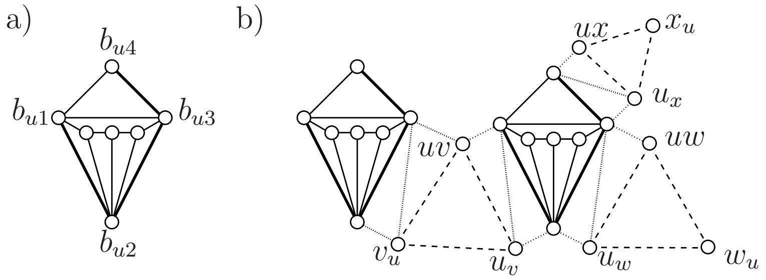

Fig. 1. The gadget graphs used in the reduction from Vertex Cover to Triangle Edge Deletion. a) Vertex gadget B_u for some vertex $u \in V$. Bold lines are docking edges. b) Left: Vertex gadget for $v \in V$. Right: Vertex gadget graph for $u \in V$ with $N(u) = \{v, w, x\}$ connected to the edge gadgets corresponding to its incident edges. Dashed lines are edges inside edge gadgets, dotted lines are edges connecting vertex gadget and edge gadgets.

cubic graphs to Triangle Edge Deletion in planar graphs.

Given an instance (G, k) of Vertex Cover, where $G = (V, E)$ is a planar cubic graph, we construct an instance (G', k') of Triangle Edge Deletion as follows. For each $u \in V$, create a planar gadget graph B_u as shown in Figure 1a. This gadget graph has 3 "docking" edges, namely $\{b_{u1}, b_{u2}\}$, $\{b_{u2}, b_{u3}\}$, and $\{b_{u3}, b_{u4}\}$, which are used for attaching the edge gadgets. Then, for each edge $\{u, v\} \in E$, create a triangle consisting of the vertex set $T_{uv} = \{uv, u_v, v_u\}$. This triangle is then attached to the vertex gadgets B_u and B_v as follows. For triangle T_{uv} and vertex gadget B_u, add the edges $\{uv, b_{ui}\}$,$\{u_v, b_{uj}\}$ and $\{u_v, b_{uj}\}$ to E', where $\{b_{ui}, b_{uj}\}$ is a docking edge that has not been used before. Vertex gadget B_v is attached analogously. Since G is cubic, the three docking edges that each vertex gadget provides suffice and each docking edge is used. Note that one can ensure planarity, by using the docking edges of u according to the relative order of neighbors of u given by an embedding of G. Since all gadgets are planar, this yields a planar graph.

The idea behind this construction is the following. Each edge $\{u, v\}$ of the original graph G must have at least one of its endpoints in the vertex cover. Correspondingly, for each triangle T_{uv} at least one edge must be deleted. Consider the graph C_u induced by the vertex set $V(B_u) \cup \{ux, u_x, uw, u_w, uv, u_v\}$. Note that the minimum number of edge deletions to make C_u triangle-free is six. However, if one of the "outer" edges $\{ux, u_x\}, \{uw, u_w\}, \{uv, u_v\}$ is deleted, it is possible to delete the other two outer edges while only deleting seven edges. Note that this is the minimum number of edge deletions to make C_u triangle-free under the constraint of having to use one of the outer edges. If we do so, we destroy all triangles in edge gadgets for edges incident to u.

Conversely, if there is a solution for the constructed instance of TRIANGLE EDGE DELETION, there always is an optimal solution for TRIANGLE EDGE DELETION which does not contain the "third" edge $\{u_v, v_u\}$ and consequently "activates" C_u or C_v, making the deletion of all the outer edges of one of these two graphs possible. There are at most k vertex gadgets corresponding to members of the vertex cover, hence we set $k' := 7k + 6(|V| - k) = 6|V| + k$. Observe that the maximum degree in G' is seven. Obviously, TRIANGLE EDGE DELETION is in NP.

Theorem 2.1 TRIANGLE EDGE DELETION *is NP-complete even when restricted to planar graphs of maximum degree seven.*

3 Problem Kernelization

In this section, we describe two kernelization algorithms, one for general graphs, which produces a kernel consisting of $6k$ vertices, and one for planar graphs, which produces a kernel consisting of only $11k/3$ vertices. The kernelization for general graphs is based on ideas for kernelizing HITTING SET [1] and VERTEX-DISJOINT TRIANGLE PACKING [6]. First, we apply the following simple data reduction rule, which is obviously correct.

Reduction Rule 1 *Remove all vertices and edges that are not contained in any triangle in G.*

In the following, assume that G is reduced with respect to Reduction Rule 1. The general strategy of our kernelization algorithm is as follows. First, greedily compute a maximal set $\mathcal{T}$ of edge-disjoint triangles in G. If $|\mathcal{T}| > k$, then the input graph is a No-instance, because each edge in a solution can destroy at most one triangle in $\mathcal{T}$. Hence, in the following we assume that $|\mathcal{T}| \leq k$. Obviously, there can be at most $3k$ vertices that are part of a triangle in $\mathcal{T}$. The task is now to bound the number of vertices that are not part of any triangle in $\mathcal{T}$. Let $I := V \setminus V(\mathcal{T})$ be the set of these vertices. It holds that

- I is an independent set and
- every triangle containing a vertex $v \in I$ shares exactly one edge with a triangle in $\mathcal{T}$.

The two properties follow from the fact that $\mathcal{T}$ is a *maximal* set of edge-disjoint triangles and can be easily verified.

To give an intuition for our next data reduction rule that bounds the size of I, we start with a simple example. Suppose that there are two vertices $u, v \in I$, each of which is contained in exactly one triangle, such that both triangles T_u

and T_v share the same edge e with a triangle in $\mathcal{T}$. It is easy to verify that it is always optimal to delete e in order to destroy the two triangles T_u and T_v. Moreover, this is still true even if we remove one of the vertices u or v (thereby decreasing the size of I). This idea can be generalized as follows. Suppose one can find a set $I' \subseteq I$ and an edge subset $E' \subseteq E(\mathcal{T})$ such that

(C1) the triangles containing a vertex in I' only share edges in E' with triangles in $\mathcal{T}$, and

(C2) one can assign a unique vertex $v_e \in I'$ to each edge $e \in E'$, such that $e \cup \{v_e\}$ induces a triangle in G.

If so, we have identified a set of $|E'|$ edge-disjoint triangles, one for each edge $e \in E'$. Since each triangle containing a vertex in I' contains an edge of E', we know that deleting all edges in E' is always optimal. This argument still holds if we remove all vertices in I'', where $I'' \subseteq I'$ is the set of all unassigned vertices. We will see that if I is too big, then there always exist such sets I' and E' and we can identify in polynomial time some vertices I'' that can be removed from the graph. In the following, we give the formal proof, which is based on matching techniques.

Define an auxiliary bipartite graph B as follows. The vertex set consists of I as one partite set and $J := \{v_e \mid e \in E(\mathcal{T})\}$ as the other, and B contains an edge $\{u, v_e\}$ if $\{u\} \cup e$ induces a triangle in G.

Reduction Rule 2 *Compute a maximum matching in B. Remove all unmatched vertices in I from G.*

Lemma 3.1 *Reduction Rule 2 is correct, that is, G is a Yes-instance if and only if the graph resulting by removing all unmatched vertices in I from G is a Yes-instance.*

Proof. Let M be the computed maximum matching in B and let I'' be all unmatched vertices in I. Since M is maximum, the graph B contains no M-augmenting path. Intuitively, we will prove that if I'' is not empty, assuming that (C1) and (C2) cannot be fulfilled leads to an M-augmenting path in B.

Let I' be the set of vertices in I that are contained in some M-alternating path starting at some vertex in I'' (including zero-length paths, that is, $I'' \subseteq I'$). Due to the definition of I'', each vertex in $I' \setminus I''$ must be matched by M. Let $E' \subseteq J$ be the matching endpoints of the vertices in $I' \setminus I''$ with respect to M.

We claim that there is no edge from some vertex in I' to some vertex in $J \setminus E'$. Suppose that there is an edge $\{u, v\}$ connecting $u \in I'$ and $v \in J \setminus E'$. Clearly, $\{u, v\} \notin M$, thus if v is not matched, then we obtain an M-

augmenting path, a contradiction, and if v is matched, then its other matching endpoint $w \in I \setminus I'$ would be contained in an M-alternating path beginning at some vertex in I'', and is therefore contained in I', again a contradiction, showing the claim.

Hence, every vertex in I' has only neighbors in E', and every vertex in E' is matched. In G, this directly corresponds to a vertex set I' and an edge set E' fulfilling Conditions (C1) and (C2). Therefore, all unmatched vertices I'' can be safely removed from the graph. □

Using Reduction Rule 1 and Reduction Rule 2, we obtain our first main result.

Theorem 3.2 Triangle Edge Deletion *admits a problem kernel with* $6k$ *vertices, which can be computed in* $O(m\sqrt{m})$ *time.*

The above results can be further improved when the input is restricted to planar graphs. In addition to Reduction Rule 1, we apply the following Reduction Rule 3. Observe that although we use Reduction Rule 3 to obtain a linear-size problem kernel for Triangle Edge Deletion on planar graphs, it is correct in non-planar graphs, too. All reduction rules presented in this paper preserve planarity because they construct a subgraph of G.

Reduction Rule 3 *If a triangle in* G *contains only one edge* e *contained in another triangle, delete* e*, and set* $k' := k - 1$*. If a triangle* Δ *in* G *does not contain an edge contained in another triangle, delete an arbitrary edge of* Δ*, and set* $k' := k - 1$*.*

The following argument shows the correctness of Reduction Rule 3. One edge of the three edges e, f, g of any triangle has to be deleted. Reduction Rule 3 always chooses an edge e, which covers all triangles covered by f or g.

The following lemma identifies a structure which cannot be found arbitrarily often in planar graphs.

Definition 3.3 Let v be a vertex of G. Two edges $\{\{w, b\}, \{w, c\}\} \subseteq E(G)$ are a *base* of v if and only if $\{w, b, c\} \subseteq N(v)$ and $\{w, b, c, v\}$ are four distinct vertices. The vertex w contained in both edges of a base is called the *base vertex* of v and *base vertex* of the base $\{\{w, b\}, \{w, c\}\}$.

Lemma 3.4 *Let* $G = (V, E)$ *be a graph and* $S \subseteq E$ *with a vertex* $w \in M := \bigcup_{e \in S} e$*. Further, let* $\deg_{G[S]}(w) \geq 2$*. If* $L = V \setminus M$ *contains more than* $2\deg_{G[S]}(w) - 2$ *vertices with a base* $B \subseteq S$ *with* w *as the base vertex, then* G *is not planar.*

The omitted proof of Lemma 3.4 uses induction on $\deg_{G[S]}(w)$.

Theorem 3.5 Triangle Edge Deletion *on planar graphs admits a problem kernel comprising* $\leq 11k/3$ *vertices, which can be computed in* $O(k \cdot n\sqrt{n})$ *time.*

Proof. Our kernelization consists of two steps. The first step is the exhaustive application of Reduction Rule 1 and Reduction Rule 3. This can be done by enumerating all triangles in $O(n\sqrt{n})$ time and determining for each edge if it is contained in zero, one or at least two triangles. This needs to be done at most k times, because every time Reduction Rule 3 is applied, the number of edges decreases by one. Let $I = (G, k)$ be a instance of Triangle Edge Deletion on planar graphs reduced with respect to Reduction Rule 1 and Reduction Rule 3. Let $S \subseteq E(G)$ be a solution of I. Let M be the set of vertices incident to an edge from S and $L := V(G) \setminus M$ be the set of vertices which are not incident to an edge from S. The second step recognizes instances with solutions consisting of just one edge. We check in $O(n\sqrt{n})$ time whether it is possible to delete at most one edge in G and obtain a triangle-free graph by enumerating all triangles in $O(n\sqrt{n})$ time.[5] If this is the case, we output "Yes". Otherwise, if $k \leq 1$, then we output "No". Afterwards, we can assume that $|S| \geq 2$, implying $|M| \geq 3$. It follows from the definition of M that $|M| \leq 2k$. Now we bound $|L|$ from above by observing that for all $v \in L$ there is a base $B \subseteq S$ of v, because otherwise Reduction Rule 1 or Reduction Rule 3 could be applied.

Let w be any vertex from $M = V(G[S])$ and let $d = \deg_{G[S]}(w)$. There are not more than $2d - 2$ vertices $v \in N$ which have $w \in M$ as the base vertex corresponding to a base $B \subseteq S$ of v: Cases $d = 0$ and $d = 1$ are trivial. See Lemma 3.4 for $d \geq 2$.

Thus there are at most $2d - 2$ neighbors of w, which are contained in L and which have a base $B \subseteq S$ with w as the base vertex. As already shown, every vertex $v \in L$ has a base $B \subseteq S$, implying that the corresponding base vertex is in M. It follows that

$$|L| \leq \sum_{w \in M} \big(2 \deg_{G[S]}(w) - 2\big) = 2 \sum_{w \in M} \deg_{G[S]}(w) - \sum_{w \in M} 2 \leq 4|S| - 2|M|.$$

[5] Observe that the first two triangles have an edge e in common, and e needs to be in every other triangle if there is to be a solution comprising only one edge.

We used the fact that the sum over all $\deg_{G[S]}(w)$ counts every edge in S exactly twice. The vertex set V is partitioned into $\{M, L\}$, hence

$$|V| \leq |M| + |L| \leq 4|S| - |M|.$$

Euler's formula applied to the planar graph $G[S]$ (G is planar) states that $|S| \leq 3|M| - 6$ for $|M| \geq 3$. Note that $|M| \geq 3$ holds (see above). Using Euler's formula yields $|M| \geq \frac{|S|}{3} + 2$. Using this, we obtain an upper bound for $|V|$:

$$|V| \leq 4|S| - \frac{|S|}{3} - 2 \leq \frac{11}{3}|S| - 2 \leq \frac{11}{3}k - 2 \leq \frac{11}{3}k$$

□

References

[1] F. N. Abu-Khzam. Kernelization algorithms for d-hitting set problems. In *Proc. 10th WADS*, volume 4619 of *LNCS*, pages 434–445. Springer, 2007.

[2] N. Alon, R. Yuster, and U. Zwick. Finding and counting given length cycles. *Algorithmica*, 17(3):209–223, 1997.

[3] J. Gramm, J. Guo, F. Hüffner, and R. Niedermeier. Automated generation of search tree algorithms for hard graph modification problems. *Algorithmica*, 39(4):321–347, 2004.

[4] J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *ACM SIGACT News*, 38(1):31–45, 2007.

[5] G. Kortsarz, M. Langberg, and Z. Nutov. Approximating maximum subgraphs without short cycles. In *Proc. 11th APPROX*, volume 5171 of *LNCS*, pages 118–131. Springer, 2008.

[6] H. Moser. Problem kernelization for graph packing. In *Proc. 35th SOFSEM*, LNCS. Springer, 2009. To appear.

[7] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms.* Oxford University Press, 2006.

[8] M. Wahlström. *Algorithms, Measures, and Upper Bounds for Satisfiability and Related Problems.* PhD thesis, Department of Computer and Information Science, Linköpings universitet, Sweden, 2007.

[9] M. Yannakakis. Edge-deletion problems. *SIAM J. Comput.*, 10(2):297–309, 1981.