On approximating the d-girth of a graph

David Peleg[1], Ignasi Sau[2], and Mordechai Shalom[3]

[1] Department of Computer Science, Weizmann Institute of Science, Rehovot, Israel.
david.peleg@weizmann.ac.il
[2] AlGCo project-team, CNRS, Laboratoire d'Informatique de Robotique et de Microélectronique de Montpellier (LIRMM), Montpellier, France.
ignasi.sau@gmail.com
[3] TelHai Academic College, Upper Galilee, 12210, Israel.
cmshalom@telhai.ac.il

Abstract. For a finite, simple, undirected graph G and an integer $d \geq 1$, a *mindeg-d subgraph* is a subgraph of G of minimum degree at least d. The *d-girth* of G, denoted $g_d(G)$, is the minimum size of a mindeg-d subgraph of G. It is a natural generalization of the usual girth, which coincides with the 2-girth. The notion of d-girth was proposed by Erdős *et al.* [13, 14] and Bollobás and Brightwell [7] over 20 years ago, and studied from a purely combinatorial point of view. Since then, no new insights have appeared in the literature. Recently, first algorithmic studies of the problem have been carried out [2,4]. The current article further explores the complexity of finding a small mindeg-d subgraph of a given graph (that is, approximating its d-girth), by providing new hardness results and the first approximation algorithms in general graphs, as well as analyzing the case where G is planar.

Keywords: generalized girth, minimum degree, approximation algorithm, hardness of approximation, randomized algorithm, planar graph.

1 Introduction

Degree-constrained subgraph problems have attracted considerable attention in the last decades, resulting in a large body of literature (see e.g. [1,2,4,8,13–15,17,21,22]). Beyond the theoretical importance of these problems, the reasons for such intensive study are mainly rooted in their wide applicability in the areas of interconnection networks and routing algorithms, among others. This article studies the computational complexity of one such problem, presented next.

For a finite, simple, undirected graph G and an integer $d \geq 1$, a *mindeg-d subgraph* is a subgraph of G of minimum degree at least d. The *d-girth* of G, denoted $g_d(G)$, is the minimum size of a mindeg-d subgraph of G. The notion of d-girth was proposed and studied by Erdős *et al.* [13,14] and Bollobás and Brightwell [7] (using different terminology). Combinatorial bounds on the d-girth of a graph can also be found in [6, 18]. For $d = 2$, $g_2(G)$ coincides with the *girth* of G, hence the d-girth can be seen as a natural generalization of the usual girth. Our interest is in the corresponding optimization problem of finding a minimum-size mindeg-d subgraph of a given graph (namely, one of size $g_d(G)$). For $d = 1$, this problem is trivial, as any edge constitutes an optimal solution. For $d = 2$, the problem corresponds exactly to finding the shortest cycle in G (as every subgraph of minimum degree at least 2 contains a cycle), and thus can be solved in polynomial time. For a fixed integer $d \geq 1$, our optimization problem is formally defined as follows.

The d-GIRTH Problem
Input: A simple undirected graph $G = (V, E)$.
Output: A minimum-size subset $S \subseteq V$ such that $G[S]$ has minimum degree at least d.

Note that $\mathrm{OPT}_{d\text{-GIRTH}}(G) = g_d(G)$. Until very recently, the computational complexity of the d-GIRTH problem had not been studied in the literature. It has been proved in [2] that for any fixed $d \geq 3$, the d-girth of a graph cannot be approximated within any constant factor, unless P = NP. Concerning approximation algorithms, the only positive result is an $\mathcal{O}(n/\log n)$-approximation algorithm for minor-free graphs [2]; approximation algorithms for the d-GIRTH problem in general graphs were missing in the literature. On the other hand, the problem has been recently studied in [4] from the *parameterized complexity* point of view [12], taking as the parameter the number of vertices in a solution. It was shown that the problem is $W[1]$-hard in general graphs, and admits FPT algorithms in minor-free families of graphs [4].

It is worth mentioning that the d-GIRTH problem is closely related to the *traffic grooming* problem, which is fundamental in modern optical networks. Loosely speaking, an important particular case of the traffic grooming problem can be stated, in graph-theoretical terms, as partitioning the edges of a given graph into subgraphs with bounded number of edges, while minimizing the total number of vertices in the partition. Traffic grooming has been proved to be a computationally hard problem [3, 9], and good approximation algorithms for the d-GIRTH problem would directly translate into efficient approximation algorithms for traffic grooming. See [4, 23] for more details about this relation.

Our results. Section 2 focuses on hardness results. The hardness results of [2] are substantially improved by proving that for any $d \geq 3$ and any $\varepsilon > 0$, there is no polynomial-time algorithm for the d-GIRTH problem with approximation ratio $2^{\mathcal{O}(\log^{1-\varepsilon} n)}$ unless $\mathrm{NP} \subseteq \mathrm{DTIME}\left(2^{\mathcal{O}(\log^{1/\varepsilon} n)}\right)$. These hardness results hold even in graphs with degrees d or $d+1$. Section 3 provides the first approximation algorithms for the d-GIRTH problem in general graphs. We first present a randomized algorithm with approximation ratio $n/\log n$ in Section 3.1. We then present another randomized algorithm with better performance in high-degree graphs (Section 3.2), and a deterministic algorithm for low-degree graphs (Section 3.3). In Section 4 we turn to the case where the input graph is planar. We prove that the d-GIRTH problem is NP-hard in planar graphs for $d \in \{3, 4, 5\}$ (Section 4.1), present a deterministic approximation algorithm (Section 4.2), and show that the problem can be solved exactly in *subexponential* time (Section 4.3). A concluding discussion appears in Section 5.

We would like to point out that in view of our results, the d-GIRTH problem appears to be rather difficult. Although the approximation ratios obtained are in some sense weak, the performance of our algorithms is not far from the best approximation algorithms for other very hard graph optimization problems like MAXIMUM CLIQUE, CHROMATIC NUMBER, or LONGEST PATH. Our work will hopefully trigger further research on the d-GIRTH problem.

Notation. All the graphs considered in this paper are finite, simple, and undirected. We use standard graph terminology, see for instance [10]. Unless stated otherwise, we denote the number of vertices of the input graph G by n. We use $\deg_G(v)$, $\delta(G)$, and $\Delta(G)$ to denote the *degree* of a vertex v in G, the *minimum degree* of G, and the *maximum degree* of G, respectively. We use $H \subseteq G$ to denote the fact that H is a *subgraph* of G. Given a subset $S \subseteq V(G)$, $G[S]$ denotes the subgraph of G *induced* by the vertices in S. For convenience, we use 'log' to denote the natural logarithm.

2 Hardness results for general graphs

It is proved in [2] that for any $d \geq 3$, the d-GIRTH problem is not in APX unless P = NP. Theorem 2 given in this section improves the hardness results of [2], relying on a slightly stronger complexity assumption. The ideas are inspired mainly by [17], and the proof builds upon the reductions and the constructions presented in [2]. Before proceeding to the improved hardness in Section 2.2, we first describe in Section 2.1 the families of graphs constructed in [2].

2.1 Preliminaries: some families of graphs

For the sake of intuition, it is helpful throughout this section to think about the case $d = 3$. Given a fixed integer $d \geq 3$, we proceed to construct a class of graphs $\mathcal{G}_d$ starting from the class of d-regular graphs whose number of edges is $d \cdot (d-1)^{\ell}$ for some positive integer ℓ. Given such a d-regular graph H, with $|V(H)| = n$, we construct a graph $G = f(H) \in \mathcal{G}_d$ as follows. By assumption, we have that $|E(H)| = nd/2 = d \cdot (d-1)^{\ell}$ for some integer ℓ. Let T be the complete d-ary rooted tree (that is, internal vertices have degree d) with root r and height $\ell+1$, which has $d \cdot (d-1)^{\ell}$ leaves and $1+d \cdot \left((d-1)^{\ell+1}-1\right)/(d-2)$ vertices overall. We identify the leaves of T with the elements in $E(H)$, and denote this set – slightly abusing notation– by E (that is, $E \subseteq V(T)$). We add another copy of E, called F, and $d-1$ edge-disjoint perfect matchings on $E \cup F$, inducing a bipartite graph with partition classes E and F. We also identify the vertices of F with the elements in $E(H)$. Now we add a set A of $|V(H)|$ new vertices identified with the elements in $V(H)$, and join them to the vertices in F according to the incidence relations in H: we add an edge between a vertex in F corresponding to $e \in E(H)$ and a vertex in A corresponding to $u \in V(H)$ if and only if e contains u. This completes the construction of G. Note that the vertices in E have degree d, and those in F have degree $d+1$. An illustration of such a graph G for $d = 3$ can be found in Fig. 2 in Appendix A. The important property of these graphs is that any solution to the d-GIRTH problem contains all vertices of G, except possibly some vertices in set A (see [2] for more details).

We now define a *graph squaring* operation for graphs in the family $\mathcal{G}_d$. Given a graph $G \in \mathcal{G}_d$, we describe the construction of G^2, and repeating inductively k times the same construction defines the graph G^{2^k}, a typical element of the class $\mathcal{G}_d^{2^k} = \{G^{2^k} \mid G \in \mathcal{G}_d\}$, for any $k \geq 0$. For every vertex v in G, construct a graph G_v as follows: first, take a copy of G, and choose $d_v = \deg_G(v)$ other arbitrary vertices $x_1, \ldots, x_{d_v}$ of degree d in $T \subseteq G$. Then, replace each of these vertices x_i by the following:

- if $d \geq 3$ is odd: a graph obtained from K_{d+1} by removing a perfect matching (e.g., a C_4 for $d = 3$).
- if $d \geq 4$ is even: a graph obtained from K_{d+2} by removing a cycle going through $d+1$ vertices. Let v^* be the vertex of degree $d+1$ in this graph.

Next, join d of the vertices of this new graph (different from v^*) to the d neighbors of x_i, $i = 1, \ldots, d_v$. Let G_v be the graph obtained in this way. Note that G_v has exactly d_v vertices of degree $d-1$. Now, take a copy of G, and replace each vertex v by G_v. Then, join each of the d_v neighbors of v in G to one of the d_v vertices of degree $d-1$ in G_v. This completes the construction of the graph G^2. We have that $|V(G^2)| = |V(G)|^2 + o(|V(G)|^2)$, because each vertex of G gets replaced by a copy of G where some of the vertices were replaced by a graph of size $d+1$ or $d+2$. An illustration of G^2 for $d = 3$ can be found in Fig. 3 in Appendix B.

Theorem 1 ([2]). *For any fixed $d \geq 3$, finding a constant-factor polynomial-time approximation algorithm for the d-GIRTH problem in the class of graphs $\bigcup_{k \geq 0} \mathcal{G}_d^{2^k}$ is* NP-*hard.*

2.2 Improved hardness results

The following technical lemma is a consequence of the constructions of Section 2.1.

Lemma 1. *For any $d \geq 3$, let G be a graph of the class $\mathcal{G}_d$ constructed in Section 2.1, and let G^2 be the graph constructed from G by the graph squaring operation.*

(i) *If $g_d(G) = \ell$, then $g_d(G^2) \leq 2\ell^2$; and*
(ii) *given a solution in G^2 containing m vertices, we can obtain in polynomial time a solution in G containing at most $\sqrt{m}$ vertices.*

Proof: The first claim follows from the fact that, given a solution $S \subseteq V(G)$ to the d-GIRTH problem in G, a feasible solution S_2 to the d-GIRTH problem in the square graph G^2 can be obtained by choosing the copies of G corresponding to vertices in S, and by choosing again in each such copy the vertices defined by S. From the construction of G^2 it follows that $|S_2| \leq |S| \cdot (|S| + (d+1)^2) \leq 2|S|^2$, as the degree of any vertex $v \in V(G)$ is at most $d+1$, and in the copy of G in G^2 corresponding to vertex v, $\deg(v)$ vertices are replaced by graphs on at most $d+2$ vertices.

In order to prove the second claim, let $S_2 \subseteq V(G^2)$ be a solution to the d-GIRTH problem in G^2, with $|S_2| = m$. We distinguish two cases. First, if S_2 contains vertices from fewer than $\sqrt{m}$ copies of G, then the solution in G defined by the vertices corresponding to these copies has size at most $\sqrt{m}$. Otherwise, there exists a copy G_v of G intersecting S_2 in which at most $\sqrt{m}$ vertices belong to S_2. Then, the solution in G defined by the vertices in G_v belonging to S_2 contains at most $\sqrt{m}$ vertices. □

Theorem 2. *For any $d \geq 3$ and any $\varepsilon > 0$, there is no polynomial-time algorithm for the d-GIRTH problem with approximation ratio $2^{\mathcal{O}(\log^{1-\varepsilon} n)}$ unless $\mathrm{NP} \subseteq \mathrm{DTIME}\left(2^{\mathcal{O}(\log^{1/\varepsilon} n)}\right)$. The theorem holds even for the class of graphs with minimum degree d and maximum degree $d+1$.*

Proof: Let $d \geq 3$ and $\varepsilon > 0$ be fixed, and suppose that there exists a polynomial-time approximation algorithm $\mathcal{A}$ that approximates the d-GIRTH problem within a ratio $2^{\mathcal{O}(\log^{1-\varepsilon} n)}$. Let $G = (V, E)$ be an instance to the d-GIRTH problem belonging to the class of graphs $\mathcal{G}_d$ defined in Section 2.1, with $|V| = n$ and $g_d(G) = \ell$. For a positive integer k, let G^{2^k} be the graph obtained from G by applying k times the graph squaring operation defined in Section 2.1. Note that for any $k \geq 0$, the vertices of G^{2^k} have degree d or $d+1$, and that $|V(G^{2^k})| = n^{2^k} + o(n^{2^k}) = \Theta(n^{2^k})$. Let p be the smallest integer such that $N = |V(G^{2^p})| \geq 2^{\log^{1/\varepsilon} n}$. Note that $N = \Theta(n^{2^p})$, so $2^p = \Theta(\log N / \log n) = \Theta(\log^{1-\varepsilon} N)$. Consequently, $2^{\mathcal{O}\left(\frac{\log^{1-\varepsilon} N}{2^p}\right)} = \mathcal{O}(1)$. By repeatedly applying Lemma 1(i), it follows that $g_d(G^{2^p}) \leq 2^{2^p - 1} \cdot \ell^{2^p}$. Then, algorithm $\mathcal{A}$ finds in time polynomial in N a solution to the d-GIRTH problem in G^{2^p} of size at most $2^{2^p-1} \cdot \ell^{2^p} \cdot 2^{\mathcal{O}(\log^{1-\varepsilon} N)}$. Then, by repeatedly applying Lemma 1(ii), we can find a solution to the d-GIRTH problem in G of size at most

$$\left(2^{2^p-1} \cdot \ell^{2^p} \cdot 2^{\mathcal{O}(\log^{1-\varepsilon} N)}\right)^{1/2^p} \leq \ell \cdot 2^{\mathcal{O}\left(\frac{\log^{1-\varepsilon} N}{2^p}\right)} = \mathcal{O}(\ell).$$

This implies that the d-GIRTH problem can be approximated in the class $\bigcup_{k \geq 0} \mathcal{G}_d^{2^k}$ within a constant factor in time polynomial in N, that is, in time $2^{\mathcal{O}(\log^{1/\varepsilon} n)}$. But since finding a constant-factor approximation algorithm for the d-GIRTH problem in $\bigcup_{k \geq 0} \mathcal{G}_d^{2^k}$ is NP-hard by Theorem 1, it follows that $\mathrm{NP} \subseteq \mathrm{DTIME}\left(2^{\mathcal{O}(\log^{1/\varepsilon} n)}\right)$. □

3 Approximation algorithms for general graphs

Section 3.1 presents a randomized approximation algorithm in general graphs and a detailed analysis of its approximation ratio. Section 3.2 proposes another randomized algorithm for graphs with high minimum degree, and discusses the relation of this algorithm with a combinatorial result of Erdős *et al.* [14]. Finally, Section 3.3 presents a deterministic approximation algorithm for graphs with low maximum degree.

3.1 A randomized $(n/\log n)$-approximation

Theorem 3. *For any $d \geq 3$, the d-GIRTH problem admits a polynomial-time randomized approximation algorithm with ratio $n/\log n$.*

Proof: A graph G is said to be *valid* if $\delta(G) \geq d$ and $V(G) \neq \emptyset$. Consider the subroutine REDUCE(G, v), that given a valid graph G and $v \in V(G)$, finds the maximum (not necessarily proper) induced subgraph of $G \setminus \{v\}$ with $\delta(G') \geq d$.

```
Procedure REDUCE(G, v)
   G' = G
   remove v and all its incident edges from G'
   while(δ(G') < d and V(G') ≠ ∅)
      { choose an arbitrary node v' ∈ V(G') with degree less than d
        remove v' and all its incident edges from G' }
   return G'.
```

Clearly the graph returned by Procedure REDUCE is either empty or valid. We now consider the following randomized algorithm.

```
Algorithm RANDOMREDUCE(G)
   while(G ≠ ∅)
      { RR ← G
        pick a node v of G uniformly at random
        G ← REDUCE(G, v) }
   return RR.
```

Clearly the algorithm returns a valid subgraph RR. We now analyze its performance. Assume the algorithm performs k iterations. Let G_i be the graph after iteration i, and $n_i = |V(G_i)|$. Clearly, $n_{\text{RR}} = |V(\text{RR})| = n_k < n_{k-1} < \ldots < n_1 < n_0 = n$. Let OPT be some minimum size valid subgraph of G, i.e., an optimal solution to the d-GIRTH problem in G, and let $\rho(n)$ be the approximation ratio of the algorithm (to be fixed later). Let $n_{\text{OPT}} = |V(\text{OPT})| = g_d(G)$. Consider the event that the algorithm is successful in finding a valid subgraph of the desired size, and the sub-event that the subgraph found by the algorithm happens to contain the optimal solution OPT, namely,

$$\begin{aligned} \text{SUCC} &= (n_{\text{RR}} \leq \rho(n) \cdot n_{\text{OPT}}), \\ \text{SUCC}^{+} &= \text{SUCC} \wedge (V(\text{OPT}) \subseteq V(\text{RR})). \end{aligned}$$

Then

$$\begin{aligned} \mathbf{Pr}[\text{SUCC}] \geq \mathbf{Pr}[\text{SUCC}^{+}] &= \mathbf{Pr}[\, n_{\text{RR}} \leq \rho(n) \cdot n_{\text{OPT}} \wedge V(\text{OPT}) \subseteq V(\text{RR}) \,] \\ &= \prod_{i=0}^{k} \left(\frac{n_i - n_{\text{OPT}}}{n_i} \right). \end{aligned}$$

The last equality holds because $V(\mathrm{OPT}) \subseteq V(G_i)$ for every $i \leq k$. If $V(\mathrm{OPT}) \subseteq V(G_i)$, then $V(\mathrm{OPT}) \subseteq V(G_{i+1})$ if and only if the node v chosen in iteration i is not in $V(\mathrm{OPT})$, which happens with probability $(n_i - n_{\mathrm{OPT}})/n_i$.

Note that at each step at least one node is removed, thus $k \leq n - n_{\mathrm{RR}}$. For fixed RR, the minimum of the last expression is attained when $k = n - n_{\mathrm{RR}}$, which implies $n_i = n - i$. Therefore,

$$\mathbf{Pr}[\textsc{Succ}] \;\geq\; \prod_{i=0}^{n-n_{\mathrm{RR}}} \left(\frac{n-i-n_{\mathrm{OPT}}}{n-i}\right) \;=\; \frac{n-n_{\mathrm{OPT}}}{n} \cdots \frac{n_{\mathrm{RR}}-n_{\mathrm{OPT}}}{n_{\mathrm{RR}}} \;.$$

If $n_{\mathrm{OPT}} = \Omega(n)$, then any solution is a constant-factor approximation, so we assume $n_{\mathrm{OPT}} = o(n)$, implying $n_{\mathrm{RR}} = o(n)$, and therefore $n - n_{\mathrm{RR}} \geq 2n_{\mathrm{OPT}}$. Then we have

$$\begin{aligned}\mathbf{Pr}[\textsc{Succ}] \;&\geq\; \frac{n_{\mathrm{RR}}-1}{n} \cdots \frac{n_{\mathrm{RR}}-n_{\mathrm{OPT}}}{n-n_{\mathrm{OPT}}+1} \;\geq\; \left(\frac{n_{\mathrm{RR}}-n_{\mathrm{OPT}}}{n-n_{\mathrm{OPT}}+1}\right)^{n_{\mathrm{OPT}}} \\ &=\; \left(\frac{n_{\mathrm{RR}}}{n} - o(1)\right)^{n_{\mathrm{OPT}}} \;=\; \left(\frac{\rho(n)\cdot n_{\mathrm{OPT}}}{n}\right)^{n_{\mathrm{OPT}}} - o(1) \;. \qquad (1)\end{aligned}$$

Hereafter we neglect the $o(1)$ term. Let $f(n) = \log n / n_{\mathrm{OPT}}$. Then $\log n / n \leq f(n) \leq \log n$. By taking $\rho(n) = (n/\log n)\cdot(f(n)/e^{c\cdot f(n)})$ for some constant c, and substituting n_{OPT} and $\rho(n)$ in Equation (1), we get

$$\mathbf{Pr}[\textsc{Succ}] \;\geq\; \left(1/e^{c\cdot f(n)}\right)^{\frac{\log n}{f(n)}} \;=\; e^{-c\cdot \log n} \;=\; n^{-c} \;.$$

For any $\varepsilon > 0$, if we run Algorithm RandomReduce $\log(1/\varepsilon)\cdot n^c$ times and choose the best solution, the probability of success is amplified to at least

$$1 - \left(1 - \frac{1}{n^c}\right)^{\log(1/\varepsilon)\cdot n^c} \;=\; 1 - (1/e)^{\log 1/\varepsilon} \;=\; 1 - \varepsilon \;,$$

and the approximation ratio is at least $\rho(n) = (n/\log n)\cdot(f(n)/e^{c\cdot f(n)})$. Note that $\rho(n) \leq n/\log n$. Indeed, this is achieved with equality (up to a constant factor) when $f(n) = 1$, i.e., $n_{\mathrm{OPT}} = \log n$. Otherwise, when $f(n) > 1$ we have $f(n)/e^{c\cdot f(n)} < 1$, and when $f(n) < 1$ we have $f(n)/e^{c\cdot f(n)} \leq f(n) < 1$. □

3.2 A better algorithm for high-degree graphs

In this section we provide another randomized algorithm for graphs with high minimum degree, and make a connection with known combinatorial results concerning subgraphs with given minimum degree.

Proposition 1. *For any $d \geq 3$ and any function $f(n)$ such that $\log n \leq f(n) \leq n$, there exists a polynomial-time randomized approximation algorithm for the d-GIRTH problem in the class of graphs with minimum degree at least $d \cdot f(n)$, with approximation ratio $\frac{16n\cdot\log n}{d\cdot f(n)}$.*

Proof: Let G be a graph with minimum degree at least $d \cdot f(n)$. The algorithm is very simple: it chooses each vertex independently with probability $8\log n/f(n)$. Let H be the graph induced in G by the vertices chosen by the algorithm, and let $n_0 = |V(H)|$. The variable n_0 is the sum of n independent Boolean random variables $B_1, \ldots, B_n$, and its expected value is $8n \cdot \log n/f(n)$. Therefore, applying the Chernoff-Hoeffding bound we get

$$\mathbf{Pr}\left[n_0 > 2 \cdot \frac{8n \cdot \log n}{f(n)}\right] \leq e^{-\frac{8n \cdot \log n}{3f(n)}} = n^{-\frac{8n}{3f(n)}} \leq \frac{1}{n^2} ,$$

so $|V(H)| \leq 16n \cdot \log n/f(n)$ with high probability. Let us now argue about the degree of a vertex $v \in V(H)$. Since $\deg_G(v) \geq d \cdot f(n)$, the expected value of $\deg_H(v)$ is at least $d \cdot f(n) \cdot 8 \log n/f(n) = 8d \cdot \log n$. Applying the Chernoff-Hoeffding bound again we get

$$\mathbf{Pr}\left[\deg_H(v) < d\right] \leq \mathbf{Pr}\left[\deg_H(v) < \frac{8d \cdot \log n}{2}\right] \leq e^{\frac{-8d \cdot \log n}{8}} = \frac{1}{n^d} < \frac{1}{n^2} ,$$

relying on the fact that $d \geq 3$. Therefore, using the union bound we get

$$\mathbf{Pr}\left[\delta(H) < d\right] \leq |V(H)| \cdot \frac{1}{n^2} \leq \frac{n}{n^2} = \frac{1}{n} .$$

Hence H is a valid solution to the d-GIRTH problem with probability at least $1 - 1/n$. Finally, the approximation ratio follows from the fact that any solution has at least $d+1$ vertices. □

The proof of Proposition 1 implies that for a graph G with $\delta(G) \geq d \cdot k$, one can find w.h.p. a subgraph H with $\delta(H) \geq d$ and $|V(H)| \leq \frac{16n \cdot \log n}{k}$. This can be thought of as a weaker but *constructive* version of the following result about subgraphs with minimum degree at least d.

Theorem 4 (Erdős *et al.* [14]). *Let $d \geq 2$ and $k > 1$ be given. Every n-vertex graph G with at least $\lceil d \cdot k \cdot n \rceil$ edges has a subgraph H with $\delta(H) \geq d$ and $|V(H)| \leq \lceil n/k \rceil$.*

Note that the combinatorial result of Theorem 4 is stronger than the one that follows from the proof of Proposition 1 in two ways. First, the required premise concerns only the number of edges of G, instead of its minimum degree. Second, the size of the subgraph obtained in the proof of Proposition 1 is greater by a factor $16 \log n$ than the one given by Theorem 4. However, the proof of Theorem 4 is *non-constructive* (at least, in polynomial time).

3.3 A deterministic algorithm for low-degree graphs

Note that the hardness results of Section 2 hold even if the degrees of the input graph are either d or $d+1$. In the following proposition we provide a *deterministic* approximation algorithm in the more general case where the input graph has degree bounded by an appropriate function of the size of the input graph.

Proposition 2. *For any integer $d \geq 3$, there exists a deterministic polynomial-time $\mathcal{O}\left(\frac{n \cdot \log \log n}{\log n}\right)$-approximation algorithm for the d-GIRTH problem in the class of n-vertex graphs with maximum degree $\mathcal{O}\left(\frac{\log n}{\log \log n}\right)$.*

Proof: The algorithm consists in doing an exhaustive search to try to build a solution to the d-GIRTH problem of appropriate size, by trying all possibilities of obtaining such a solution starting from each vertex of G. The algorithm stops at some point in order to keep the running time polynomial, and if no solution has been found so far, it outputs the whole graph.

Let the maximum degree of G be at most b, and let k be the maximum size of a solution that the algorithm can find (both b and k will be specified later). The algorithm proceeds as follows. Starting from a given vertex, it tries to build a feasible solution $S \subseteq V(G)$ by adding vertices one by one to S. At a given moment, if some vertex $v \in S$ has strictly less than d neighbors in S, it chooses a neighbor of v in $V(G) \setminus S$, and adds it to S. This process continues until either all vertices of G have degree at least d in $G[S]$, or $|S| = k$.

For an integer ℓ, with $0 \leq \ell \leq k$, we define $f(\ell)$ to be the *remaining* running time of the algorithm assuming that all possible solutions of size at most ℓ have been already considered. Therefore, by definition $f(0)$ is the overall running time of the algorithm. Once ℓ vertices have been already chosen in S, with $\ell \geq 1$, the number of choices for a neighbor of each vertex of S in $V(G) \setminus S$ is at most b, so it holds

$$f(\ell) \;\leq\; \ell \cdot b \cdot f(\ell + 1). \tag{2}$$

On the other hand, at the beginning the algorithm chooses an arbitrary vertex of G, so

$$f(0) \;\leq\; n \cdot f(1). \tag{3}$$

Starting from Equation (3), using Equation (2) recursively, and assuming that the algorithm stops when $|S| = k$, we get

$$f(0) \;\leq\; n \cdot b^k \cdot k! \cdot f(k + 1). \tag{4}$$

As the running time must be polynomial in n, from Equation (4) it follows that $n \cdot b^k \cdot k! = n^{\mathcal{O}(1)}$, that is, $b^k \cdot k! \leq n^c$ for some positive integer c. In other words,

$$k \cdot \log b + k \cdot \log k \;\leq\; c \cdot \log n. \tag{5}$$

If we further impose that $b \leq k$, a sufficient condition for Equation (5) to be satisfied is that $k = \mathcal{O}\left(\frac{\log n}{\log \log n}\right)$. That is, for graphs with maximum degree $\mathcal{O}\left(\frac{\log n}{\log \log n}\right)$, the above procedure constitutes a polynomial-time $\mathcal{O}\left(\frac{n \cdot \log \log n}{\log n}\right)$-approximation algorithm. □

4 Planar graphs

In this section we focus on the case where the input graph is restricted to be a planar graph. The first important observation is that as any planar graph has a vertex of degree at most 5 [10], in a planar graph there exist feasible solutions to the d-GIRTH problem only for $d \leq 5$. Note that the hardness results of [2], and therefore also those of Section 2, do not apply to planar graphs, as the constructed graphs are highly nonplanar. In Section 4.1 we prove that the d-GIRTH problem is NP-hard in planar graphs for $d \in \{3, 4, 5\}$. We then discuss approximation algorithms in Section 4.2, and present a subexponential exact algorithm in Section 4.3.

4.1 Hardness results

Theorem 5. *For $d \in \{3, 4, 5\}$, the d-GIRTH problem is NP-hard in planar graphs with maximum degree at most $3d$.*

Proof: The reduction is from MINIMUM VERTEX COVER (VC for short) in planar graphs with maximum degree at most 3, which is known to be NP-hard [16]. Note that MINIMUM VERTEX COVER admits a PTAS in planar graphs [5]. For the sake of

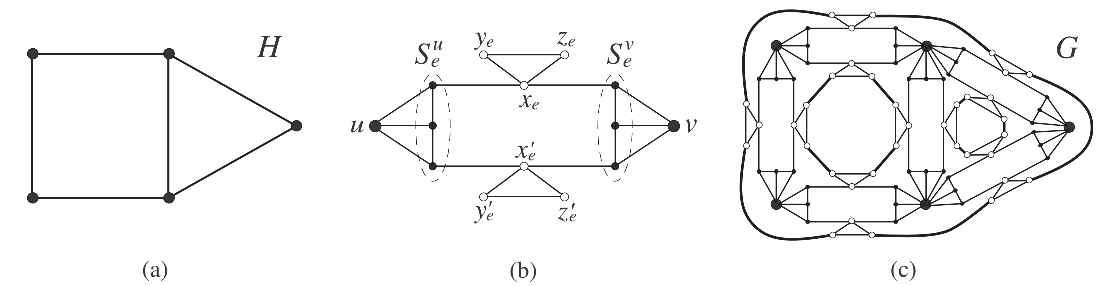

Fig. 1. Reduction in the proof of Theorem 5 for $d = 3$: (a) Instance H to Minimum Vertex Cover. (b) Gadget corresponding to an edge $e = \{u, v\} \in E(H)$. (c) Instance G to the 3-girth problem built from H.

presentation, we first state the reduction for $d = 3$, and then we show how to modify the gadgets for $d \in \{4, 5\}$.

Let H be a planar graph with $\Delta(H) \leq 3$, an instance to the Minimum Vertex Cover problem, which we can assume to be connected (see Fig. 1(a) for an example). To build G from H, we first replace each edge $e = \{u, v\} \in E(H)$ by the gadget depicted in Fig. 1(b), containing u, v, and 12 new vertices. Among these vertices, let S_e^u (resp. S_e^v) be the three vertices adjacent to u (resp. v). Vertices $x_e, y_e, z_e, x'_e, y'_e, z'_e$ are colored *white* and vertices in S_e^u and S_e^v are colored *black* (see Fig. 1(b)). Now, for each face f of H (including the external one) consisting of edges $e_0, e_2, \ldots, e_{k-1}$ such that e_i is incident to e_{i+1} for $i = 0, \ldots, k-1$ (indices taken modulo k), we add the following edges. Assume without loss of generality that all the white vertices corresponding to f are of the form x_e, y_e, z_e. For $i = 0, \ldots, k-1$, add an edge between vertex z_{e_i} and vertex $y_{e_{i+1}}$, the indices being taken modulo k. These edges between white vertices corresponding to different edges are called *face edges*. This completes the construction of G, which is illustrated in Fig. 1(c). Note that G is a planar graph with maximum degree at most 9.

Consider a solution $S \subseteq V(G)$ to the 3-girth problem in G. By construction, S cannot contain just black vertices or vertices corresponding to vertices in H, so at least one white vertex belongs to S, say x_e for an edge $e = \{u, v\} \in E(H)$. Due to the degree constraints and since x_e is adjacent to only 2 white vertices, vertex x_e forces either all the vertices in S_e^u or all the vertices in S_e^v to belong to S, which in turn also forces vertex x'_e to be in S. Due to the face edges, once a vertex x_e is in S, so are all the white vertices corresponding to edges in the same face as edge $e \in E(H)$. Thus, white vertices inductively force *all* $6 \cdot |E(H)|$ white vertices to be in S. Now recall that for a pair of white vertices x_e, x'_e to have degree at least 3 in $G[S]$, either all the vertices in S_e^u or in S_e^v must belong to S, so any optimal solution to the 3-girth in G contains exactly $3 \cdot |E(H)|$ black vertices. Finally, note that if the vertices in S_e^u (resp. S_e^v) belong to S, they force vertex u (resp. v) to belong to S. As for each edge $e = \{u, v\} \in E(H)$, either u or v must belong to S, we conclude that there is a bijection between vertex covers of H and feasible solutions to the 3-girth in G.

The above discussion implies that $g_3(G) = 9 \cdot |E(H)| + \mathrm{OPT}_{\mathrm{VC}}(H)$. (For instance, in the example of Fig. 1, any optimal vertex cover of H of size 3 defines an optimal subgraph of G on 57 vertices with minimum degree at least 3.) As Minimum Vertex Cover is NP-hard, so is the 3-girth problem.

For $d \in \{4,5\}$, one just needs to modify the gadget of Fig. 1(b) with respect to the reduction for $d = 3$; the corresponding gadgets for $d \in \{4,5\}$ can be found in Appendix C.□

4.2 Approximation algorithms

In this section we derive the existence of a deterministic approximation algorithm with ratio $n/f(n)$ and running time $2^{\mathcal{O}(f(n))} \cdot n$ when the input graph G is restricted to be planar. In particular, this provides an alternative to the polynomial-time $n/\log n$-approximation algorithm provided in [2] for minor-free graphs.

Proposition 3. *For any function $f(n) \leq n$, there exists a deterministic approximation algorithm for the d-GIRTH problem in n-vertex planar graphs with approximation ratio $n/f(n)$ and running time $2^{\mathcal{O}(f(n))} \cdot n$.*

Proof: It is well-known that the number of non-isomorphic planar graphs on k vertices is $2^{\mathcal{O}(k)}$ [24]. In addition, this set of graphs can be generated in time proportional to its size using the algorithm in [19]. Given a planar graph G on n vertices and an arbitrary function $f(n) \leq n$, we generate all non-isomorphic planar graphs on $f(n)$ vertices in time $2^{\mathcal{O}(f(n))}$. We remove from the list the graphs with minimum degree less than d. Then, for each graph H in this list, we test whether G contains a subgraph isomorphic to H using the recent results for planar subgraph isomorphism [11], in time $2^{\mathcal{O}(f(n))} \cdot n$. If none of these subgraphs is found, we output the whole graph G. This procedure clearly yields a $(n/f(n))$-approximation algorithm running in time $2^{\mathcal{O}(f(n))} \cdot n$. □

4.3 Exact algorithms

In this section we show that the problem can be solved in *subexponential* time when the input is restricted to planar graphs. Recall that there exist valid solutions only for $d \leq 5$.

Theorem 6. *For any $d \geq 3$, the d-GIRTH problem can be solved exactly in planar graphs in time $2^{\mathcal{O}(\sqrt{n} \cdot \log n)}$.*

Proof: We use the classical divide-and-conquer approach. By the planar separator Theorem [20], every n-vertex planar graph has a vertex separator W of size at most $c\sqrt{n}$, for some small constant $c \leq 2\sqrt{2}$, such that after the removal of W the graph is partitioned into two disconnected subgraphs on vertex sets Z_1 and Z_2, each of cardinality at most $2n/3$. In addition, such separator W can be found in time $\mathcal{O}(n)$.

Our algorithm proceeds recursively as follows. The separator given by [20] divides the problem into two or more smaller problems. In order to build a feasible solution to the d-GIRTH problem, exhaustively check every subset of vertices in the separator, and then for each subset check every possible set of up to d neighbors in Z_1 and Z_2. As usual, the subproblems are solved by applying the method recursively, and the solutions to the subproblems are combined to give a solution to the original problem in the input graph G.

To analyze the running time, define the function $f(\ell)$ to be the time required by the algorithm to process a graph on ℓ vertices. Hence $f(n)$ is the overall running time of our algorithm, and we can assume that $f(1) = 1$. As the number of choices for a subset of the separator W is $2^{|W|} \leq 2^{c\sqrt{n}}$, and the number of choices in Z_1 or Z_2 for a set of at most d neighbors of each vertex in the separator is at most $\binom{|Z_i|}{d}$, $i = 1, 2$,

$$\begin{aligned} f(n) &\leq 2^{|W|} \cdot \left(\binom{|Z_1|}{d} \cdot \binom{|Z_2|}{d} \right)^{|W|} \cdot 2 \cdot f(2n/3) \leq 2^{c\sqrt{n}} \cdot (2n/3)^{2dc\sqrt{n}} \cdot 2 \cdot f(2n/3) \\ &= 2^{c_1 \sqrt{n} \cdot \log n} \cdot f(2n/3) \leq 2^{c_1 \left(\sqrt{n} \cdot \log n + \sqrt{2n/3} \cdot \log(2n/3) + \ldots \right)} \cdot f(1) \\ &\leq 2^{c_1 \log n \cdot \left(\sqrt{n} + \sqrt{2n/3} + \sqrt{4n/9} + \ldots \right)} \leq 2^{c_1 \sqrt{n} \cdot \log n \cdot \left(\sum_{i=0}^{\infty} (2/3)^i \right)} = 2^{c_2 \sqrt{n} \cdot \log n} , \end{aligned}$$

where c_1, c_2 are suitable constants defined by the above equations, depending on c and d. □

5 Concluding remarks

This article studies the problem of approximating the d-girth of a graph, the order of a smallest subgraph with minimum degree at least d, for a fixed integer $d \geq 3$, and makes first steps towards understanding the computational complexity of this apparently hard problem. We now summarize our results and present several possible lines for further research.

We proved that for any $d \geq 3$ and $\varepsilon > 0$, there is no polynomial-time algorithm for the d-GIRTH problem with approximation ratio $2^{\mathcal{O}(\log^{1-\varepsilon} n)}$ in graphs with degrees d or $d+1$ unless NP $\subseteq$ DTIME $\left(2^{\mathcal{O}(\log^{1/\varepsilon} n)}\right)$. We suspect that the problem is even harder than this. In the spirit of [17] for the LONGEST PATH problem, we present the following conjecture.

Conjecture 1. For every fixed $d \geq 3$, there is no polynomial-time approximation algorithm for the d-GIRTH problem with ratio $n^{1-\delta}$, for some constant $\delta > 0$ unless P = NP.

We provided the first approximation algorithms for the d-GIRTH problem in general graphs. Specifically, we presented a randomized algorithm with approximation ratio $n/\log n$ in any graph, another randomized algorithm with better performance in high-degree graphs, and a deterministic algorithm for low-degree graphs. These approximation ratios could hopefully be improved, although it looks like a challenging task. Our latter two approaches for high- and low-degree graphs complement each other in some sense, so it would be interesting to try to combine them into a better algorithm.

We also studied the case where the input graph is planar. We proved that the d-GIRTH problem is NP-hard in planar graphs for $d \in \{3, 4, 5\}$, presented a deterministic approximation algorithm (with the same ratio as the algorithm for general graphs) based on a recent result for subgraph isomorphism [11], and showed that the problem can be solved exactly in subexponential time. This latter result may provide some clue to the possible existence of a PTAS in planar graphs, which remains wide open. So far, none of the many approaches to obtaining a PTAS in planar graphs seems to fit the d-GIRTH problem.

We point out that some of our results do not strongly use specific properties of the d-GIRTH problem, and can be applied to the class of problems of the form "minimum subgraph with property P", in particular to the problem of finding a *d-regular* subgraph of minimum size. It would be interesting also to study other variants of the problem, like minimizing the number of edges of a subgraph with minimum degree at least d (in planar graphs, this version is equivalent to the original one, modulo a constant factor), or considering the vertex-weighted version.

Finally, the reader is also referred to several nice combinatorial conjectures of Erdős *et al.* [13, 14] about the existence of small subgraphs with given minimum degree.

References

1. L. Addario-Berry, K. Dalal, and B. A. Reed. Degree constrained subgraphs. *Discrete Applied Mathematics*, 156(7):1168–1174, 2008.
2. O. Amini, D. Peleg, S. Pérennes, I. Sau, and S. Saurabh. Degree-Constrained Subgraph Problems: Hardness and Approximation Results. In *Proc. of Workshop on Approximation and On-line Algorithms (ALGO/WAOA)*, volume 5426 of *LNCS*, pages 29–42, 2008. Full version available at `www.cs.technion.ac.il/~ignasi/Pubs/APP+10.pdf`.
3. O. Amini, S. Pérennes, and I. Sau. Hardness and Approximation of Traffic Grooming. *Theoretical Computer Science*, 410(38-40):3751–3760, 2009.
4. O. Amini, I. Sau, and S. Saurabh. Parameterized Complexity of the Smallest Degree-Constrained Subgraph Problem. In *Proc. of the International Workshop on Parameterized and Exact Computation (IWPEC)*, volume 5008 of *LNCS*, pages 13–29, 2008. Full version available at `www.cs.technion.ac.il/~ignasi/Pubs/ASS10.pdf`.
5. B. S. Baker. Approximation Algorithms for NP-Complete Problems on Planar Graphs. *Journal of the ACM*, 41(1):153–180, 1994.
6. J.-C. Bermond and C. Peyrat. Induced Subgraphs of the Power of a Cycle. *SIAM Journal on Discrete Mathematics*, 2(4):452–455, 1989.
7. B. Bollobás and G. Brightwell. Long Cycles in Graphs with no Subgraphs of Minimal Degree 3. *Discrete Mathematics*, 75:47–53, 1989.
8. J. Cheriyan and R. Thurimella. Approximating Minimum-Size k-Connected Spanning Subgraphs via Matching (extended abstract). In *Proc. of the 37th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 292–301, 1996.
9. T. Chow and P. Lin. The ring grooming problem. *Networks*, 44:194–202, 2004.
10. R. Diestel. *Graph Theory*, volume 173. Springer-Verlag, 2005.
11. F. Dorn. Planar Subgraph Isomorphism Revisited. In *Proc. of the 27th International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 5 of *LIPIcs*, pages 263–274, 2010.
12. R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer-Verlag, 1999.
13. P. Erdős, R. J. Faudree, A. Gyárfás, and R. H. Schelp. Cycles in Graphs Without Proper Subgraphs of Minimum Degree 3. *Ars Combinatorica*, 25(B):195–201, 1988.
14. P. Erdős, R. J. Faudree, C. C. Rousseau, and R. H. Schelp. Subgraphs of Minimal Degree k. *Discrete Mathematics*, 85(1):53–58, 1990.
15. H. N. Gabow. An Efficient Reduction Technique for Degree-Constrained Subgraph and Bidirected Network Flow Problems. In *Proc. of the 15th Annual ACM Symposium on Theory of Computing (STOC)*, pages 448–456, 1983.
16. M. R. Garey and D. S. Johnson. The Rectilinear Steiner Tree Problem is NP-Complete. *SIAM Journal on Applied Mathematics*, 32(4):826–834, 1977.
17. D. Karger, R. Motwani, and G. Ramkumar. On Approximating the Longest Path in a Graph. *Algorithmica*, 18(1):82–98, 1997.
18. A. Kézdy. *Studies in Connectivity*. PhD thesis, Univ. of Illinois, Urbana-Champaign, 1991.
19. Z. Li and S.-I. Nakano. Efficient generation of plane triangulations without repetitions. In *Proc. of the 28th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 2076 of *LNCS*, pages 433–443, 2001.
20. R. J. Lipton and R. E. Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36:177189, 1979.
21. Z. Nutov. Approximating Directed Weighted-Degree Constrained Networks. In *Proc. of the 11th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX-RANDOM)*, volume 5171 of *LNCS*, pages 219–232, 2008.
22. M. Safari and M. R. Salavatipour. A Constant Factor Approximation for Minimum λ-Edge-Connected k-Subgraph with Metric Costs. In *Proc. of the 11th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX-RANDOM)*, volume 5171 of *LNCS*, pages 233–246, 2008.
23. I. Sau. *Optimization in Graphs under Degree Constraints. Application to Telecommunication Networks*. PhD thesis, Université de Nice-Sophia Antipolis (UNS) and Universitat Politècnica de Catalunya (UPC), 2009.
24. W. T. Tutte. A census of planar triangulations. *Canadian Journal of Mathematics*, 14:21–38, 1962.

Appendices

A Fig. 2: the class of graphs $\mathcal{G}_d$ (Section 2.1)

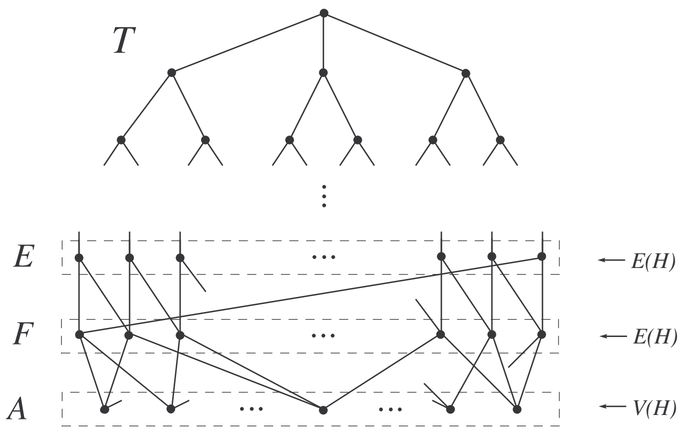

Fig. 2. Example of a graph $G \in \mathcal{G}_3$ built in Section 2.1.

B Fig. 3: the class of graphs $\mathcal{G}_d^{2^k}$ (Section 2.1)

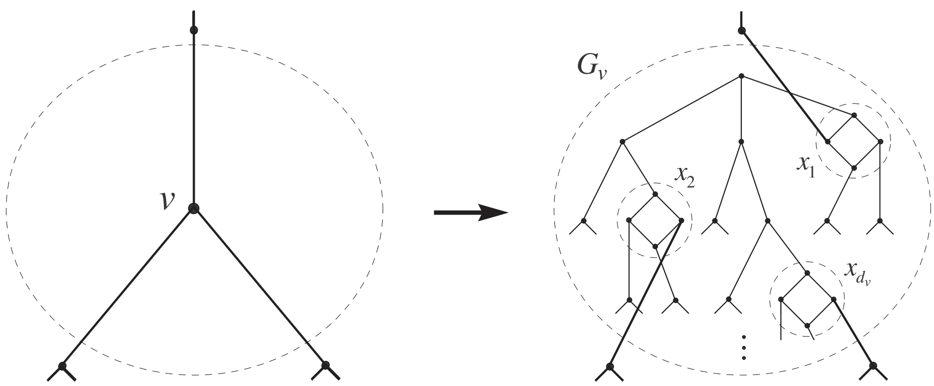

Fig. 3. Example of a graph $G^2 \in \mathcal{G}_3^2$ built in Section 2.1.

C Gadgets in the reduction of Theorem 5 for $d \in \{4, 5\}$

For $d \in \{4, 5\}$, the proof is essentially the same as in the case $d = 3$, just by replacing the gadget of Fig. 1(b) by the gadgets depicted in Fig. 4 (smaller gadgets may exist). Note that since $\Delta(H) \leq 3$, it holds $\Delta(G) \leq 3d$.

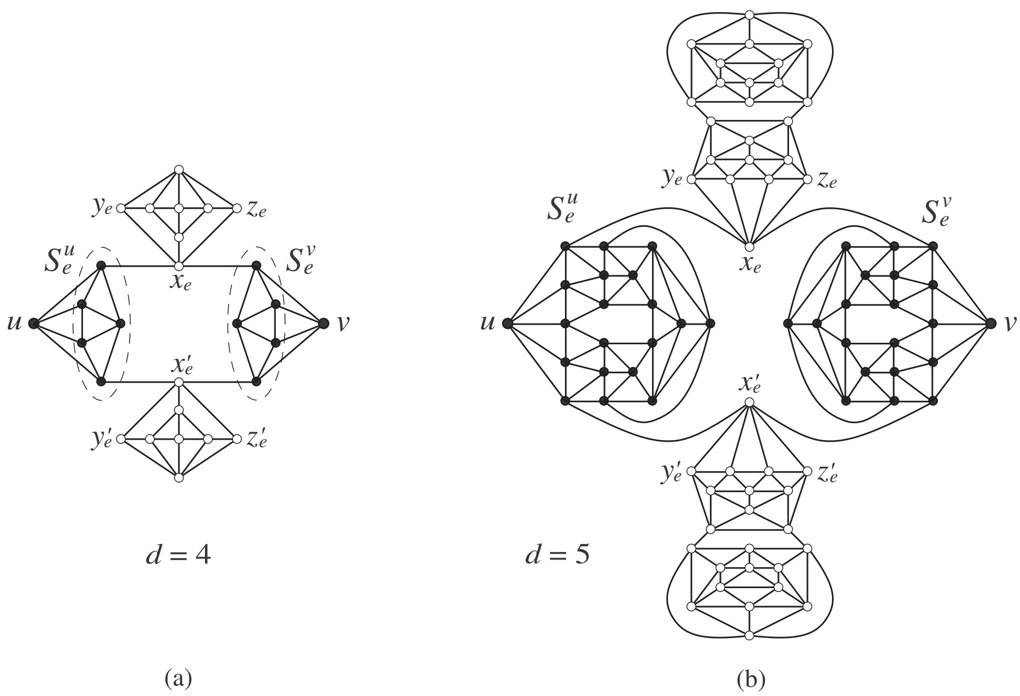

Fig. 4. Gadgets in the proof of Theorem 5: (a) for $d = 4$, and (b) for $d = 5$.

Again, for $d = 4$ (resp. $d = 5$), the face edges connect the white vertices that have degree 3 (resp. 4) in Fig. 4(a) (resp. Fig. 4(b)). Therefore, any solution S in G contains all the white vertices, and any optimal solution contains, for each edge $e = \{u, v\} \in E(H)$, either all the vertices in S_e^u or in S_e^v. One can check that for $d = 4$ it holds that $g_4(G) = 21 \cdot |E(H)| + \mathrm{OPT}_{\mathrm{VC}}(H)$, and that for $d = 5$ it holds that $g_5(G) = 65 \cdot |E(H)| + \mathrm{OPT}_{\mathrm{VC}}(H)$. The proof is now completed.