



HAL
open science

Complexity of Finding Maximum Locally Irregular Induced Subgraphs

Foivos Fioravantes, Nikolaos Melissinos, Theofilos Triommatis

► **To cite this version:**

Foivos Fioravantes, Nikolaos Melissinos, Theofilos Triommatis. Complexity of Finding Maximum Locally Irregular Induced Subgraphs. Scandinavian Symposium and Workshops on Algorithm Theory (SWAT) 2022, Jun 2022, Torshavn, Faroe Islands. 10.4230/LIPIcs.SWAT.2022.23 . hal-03905056

HAL Id: hal-03905056

<https://hal.science/hal-03905056>

Submitted on 17 Dec 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Complexity of Finding Maximum Locally Irregular Induced Subgraphs

Foivos Fioravantes ✉

Université Côte d’Azur, Inria, CNRS, I3S, Valbonne, France

Nikolaos Melissinos ✉

Université Paris-Dauphine, Université PSL, CNRS, LAMSADE, 75016, Paris, France

Theofilos Triommatis ✉

School of Electrical Engineering, Electronics and Computer Science University of Liverpool,

Liverpool, L69-3BX, UK

Abstract

If a graph G is such that no two adjacent vertices of G have the same degree, we say that G is *locally irregular*. In this work we introduce and study the problem of identifying a largest induced subgraph of a given graph G that is locally irregular. Equivalently, given a graph G , find a subset S of $V(G)$ with minimum order, such that by deleting the vertices of S from G results in a locally irregular graph; we denote with $I(G)$ the order of such a set S . We first examine some easy graph families, namely paths, cycles, trees, complete bipartite and complete graphs. However, we show that the decision version of the introduced problem is \mathcal{NP} -Complete, even for restricted families of graphs, such as subcubic planar bipartite, or cubic bipartite graphs. We then show that we can not even approximate an optimal solution within a ratio of $\mathcal{O}(n^{1-\frac{1}{k}})$, where $k \geq 1$ and n is the order of the graph, unless $\mathcal{P}=\mathcal{NP}$, even when the input graph is bipartite.

Then, looking for more positive results, we turn our attention towards computing $I(G)$ through the lens of parameterised complexity. In particular, we provide two algorithms that compute $I(G)$, each one considering different parameters. The first one considers the size of the solution k and the maximum degree Δ of G with running time $(2\Delta)^k n^{\mathcal{O}(1)}$, while the second one considers the treewidth tw and Δ of G , and has running time $\Delta^{2tw} n^{\mathcal{O}(1)}$. Therefore, we show that the problem is FPT by both k and tw if the graph has bounded maximum degree Δ . Since these algorithms are not FPT for graphs with unbounded maximum degree (unless we consider $\Delta + k$ or $\Delta + tw$ as the parameter), it is natural to wonder if there exists an algorithm that does not include additional parameters (other than k or tw) in its dependency.

We answer negatively, to this question, by showing that our algorithms are essentially optimal. In particular, we prove that there is no algorithm that computes $I(G)$ with dependence $f(k)n^{\mathcal{O}(k)}$ or $f(tw)n^{\mathcal{O}(tw)}$, unless the ETH fails.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms; Theory of Computation → Design and Analysis of Algorithms → Parameterized Complexity and Exact Algorithms; Mathematics of computing → Approximation algorithms

Keywords and phrases Locally irregular, largest induced subgraph, FPT, treewidth, W-hardness, approximability

Digital Object Identifier 10.4230/LIPIcs.SWAT.2022.23

Related Version Full version hosted on HAL.

: <https://hal.archives-ouvertes.fr/hal-03358273>

Funding *Foivos Fioravantes*: Supported by the French government through the UCA JEDI (ANR-15-IDEX-01) and the EUR DS4H (ANR-17-EURE-004) Investments in the Future projects.

Theofilos Triommatis: Supported by EP/S023445/1 EPSRC CDT in Distributed Algorithms, University of Liverpool.



© Foivos Fioravantes, Nikolaos Melissinos and Theofilos Triommatis;

licensed under Creative Commons License CC-BY 4.0

18th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2022).

Editors: Artur Czumaj and Qin Xin; Article No. 23; pp. 23:1–23:20

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

A graph G is said to be *locally irregular*, if every two adjacent vertices of G have different degrees. In this paper, we introduce and study the problem of finding a largest locally irregular induced subgraph of a given graph. This problem is equivalent to identifying what is the minimum number of vertices that must be deleted from G , so that what remains is a locally irregular graph.

Locally irregular graphs. The notion of locally irregular graphs was first introduced in [6]. The most interesting aspect of locally irregular graphs, comes from their connection to the so-called 1-2-3 Conjecture, proposed in [22]. Formally, the 1-2-3 Conjecture states that for almost every graph, we should be able to place weights from $\{1, 2, 3\}$ on the edges of that graph, so that the colouring, that assigns a colour to each vertex equal to the sum of the weights on its adjacent edges, is a proper vertex-colouring of the graph.

As we said earlier, the 1-2-3 Conjecture seems to have some very interesting links to locally irregular graphs. An obvious connection is that this conjecture holds for locally irregular graphs. Indeed, placing weight equal to 1 to all the edges of a locally irregular graph, suffices to produce a proper vertex-colouring, as each vertex receives a colour equal to its degree. Furthermore, there have been some steps towards proving that conjecture, which involve edge-decomposing a graph into a constant number of locally irregular subgraphs, *i.e.*, given G , find an edge-colouring of G using a constant number of colours, such that each colour induces a locally irregular subgraph of G . This is the main motivation behind [6], and it seems to remain interesting enough to attract more attention [8, 25, 30].

Note that the class of locally irregular graphs can be seen as an antonym to that of *regular* graphs, *i.e.*, graphs such that all of their vertices have the same degree. It is important to state here that there exist several alternative such notions. This is mainly due to the very well known fact that there are no non-trivial *irregular* graphs, *i.e.*, graphs that do not contain two vertices (not necessarily adjacent) with the same degree (see [12]). Thus, the literature has plenty of slightly different definitions of irregularity (see for example [2, 12, 13, 20, 29]). One way to deal with the nonexistence of irregular graphs, is to define a notion of *local* irregularity. Intuitively, instead of demanding for all vertices of a graph to have different degrees, we are now considering each vertex v separately, and request that the vertices “around” v to verify some properties of irregularity. For example, the authors of [3] study graphs G such that for every vertex v of G , no two neighbours of v have the same degree. For an overview of other interesting notions of irregularity (local or otherwise), we refer the reader to [4].

Largest induced subgraph verifying specific properties. The problem we introduce belongs in a more general and well studied family of problems, which is about identifying a largest induced subgraph of a given graph that verifies a specific property Π . That is, given a graph $G = (V, E)$ and an integer k , is there a set $V' \subseteq V$ such that $|V'| \leq k$ and $G[V \setminus V']$ has the specified property Π ? In our case, the property Π is “the induced subgraph is locally irregular”. This generalised problem is indeed classic in graph theory, and it is known as the INDUCED SUBGRAPH WITH PROPERTY Π (ISPII for short) problem in [21]. Unfortunately, it was shown in [24], that ISPII is a hard problem for any property Π that is *hereditary*, *i.e.*, all induced subgraphs of G verify Π if G itself verifies that property.

However, the ISPII problem remains interesting (one could say that it actually becomes more interesting) even if the property Π is not hereditary. Recently, the authors of [7] studied the problem for Π being “all vertices of the induced subgraph have odd degree”, which

clearly is not a hereditary property. Nevertheless, they showed that this is an \mathcal{NP} -hard problem, and they gave an FPT algorithm that solves the problem when parameterised by the rank-width. Also, the authors of [1, 5, 28] studied the ISPII problem, where Π is the rather natural property “the induced subgraph is d -regular”, where d is an integer given in the input (recall that a graph is said to be d -regular if all of its vertices have the same degree d). In particular, in [5] it is shown that finding a largest (connected) induced subgraph that is d -regular, is \mathcal{NP} -hard to approximate, even when restricted on bipartite or planar graphs. The authors of [5] also provide a linear-time algorithm to solve this problem for graphs with bounded treewidth. In contrast, the authors of [1] take a more practical approach, as they focus on solving the problem for the particular values of $d = 1$ and $d = 2$, by using bounds from quadratic programming, Lagrangian relaxation and integer programming.

It is quite clear that, in some sense, the property that interests us lies on the opposite side of the one studied in [1, 5, 28]. However, both properties, “the induced subgraph is regular” and “the induced subgraph is locally irregular” are not hereditary. This means that we do not get an \mathcal{NP} -hardness result directly from [24]. Furthermore, the ISPII problem always admits an FPT algorithm, when parameterised by the size of the solution, if Π is a hereditary property (proven in [11, 23]), but for a non-hereditary one, this is not always true. Indeed in [28], the authors proved that when considering Π as “the induced subgraph is regular”, the ISPII problem is W[1]-hard when parameterised by the size of the solution. That is, there should be no $f(k)n^c$ time algorithm for this problem, where c is a constant. For such problems, it is also interesting to see if there exists any algorithm with running time $n^{o(k)}$ or $f(k)n^{o(k)}$. The authors of [14, 15, 16] provide techniques that can be used to strongly indicate the non-existence of such algorithms, by applying them on a variety of W[1]-hard and W[2]-hard problems, such as the INDEPENDENT SET and the DOMINATING SET, parameterised by the size of their solutions. Usually these lower bounds are shown under the assumption of a weaker version of the EXPONENTIAL TIME HYPOTHESIS, which states that SAT can not be solved in time $2^{o(n+m)}$.

Our contribution. We begin in Section 2 by providing the basic notations and definitions that are going to be used throughout this paper. In Section 3, we deal with the complexity of the introduced problem. In particular, we show that the problem belongs in \mathcal{P} if the input graph is a path, cycle, tree, complete bipartite or complete graph. We then prove that finding the maximum induced locally irregular subgraph of a given graph G is \mathcal{NP} -hard, even if G is restricted to being a subcubic planar bipartite, or a cubic bipartite graph.

As the introduced problem seems to be computationally hard even for restricted families of graphs, we then investigate its approximability. Unfortunately, we prove in Section 4 that for any bipartite graph G of order n and $k \geq 1$, there can be no polynomial time algorithm that finds an approximation of $I(G)$ within ratio $\mathcal{O}(n^{1-\frac{1}{k}})$, unless $\mathcal{P}=\mathcal{NP}$. Nevertheless, we do provide a (simple) d -approximation algorithm for d -regular bipartite graphs.

We then decide to look into its parameterised complexity. In Section 5, we present two algorithms that compute $I(G)$, each one considering different parameters. The first considers the size of the solution k and the maximum degree Δ of G , and has running time $(2\Delta)^k n^{\mathcal{O}(1)}$, while the second considers the treewidth tw and Δ of G , and has running time $\Delta^{2tw} n^{\mathcal{O}(1)}$. Unfortunately, these algorithms can be considered as being FPT only if Δ is part of the parameter. In Section 5.1, we present two linear fpt-reductions which prove that the problem is W[2]-hard when parameterised only by the size of the solution and W[1]-hard when parameterised only by the treewidth. These reductions also show that we can not even have an algorithm that computes $I(G)$ in time $f(k)n^{o(k)}$ or $\mathcal{O}^*(f(tw)n^{o(tw)})$, unless the ETH

137 fails. The \mathcal{O}^* notation is used to suppress polynomial factors in regards to n and tw .

138 2 Preliminaries

139 For notions and definitions on graph theory not explained here, we refer the reader to [18].

140 Let $G = (V, E)$ be a graph and $G' = (V', E')$ be a subgraph of G (i.e., created by deleting
141 vertices and/or edges of G). Recall first that the subgraph G' is *induced* if it can be created
142 only by deleting vertices of G . That is, for each edge $uv \in E$, if $u, v \in V'$, then $uv \in E'$. For
143 any vertex $v \in V$, let $N_G(v) = \{u \in V : uv \in E\}$ denote the *neighbourhood* of v in G , and let
144 $d_G(v) = |N_G(v)|$ denote the *degree* of v in G . We also define $N_G[v] = N_G(v) \cup \{v\}$. Finally,
145 for any $X \subseteq V$, we define $N_G[X] = \bigcup_{v \in X} N_G[v]$. Note that, whenever the graph G is clear
146 from the context, we will omit the subscript and simply write $N(v)$, $d(v)$, $N[v]$ and $N[X]$.

147 One way to show that a problem can not be approximated within a certain ratio, is
148 through a *gap reduction*. The goal of such a reduction is to show that it is \mathcal{NP} -hard to
149 differentiate between instances that have a solution of size $\leq \alpha$ and those for which any
150 solution has size $> \beta$. If such is the case, then we know that we cannot approximate the
151 optimal solution within a ratio of $\frac{\beta}{\alpha}$, as otherwise we would get that $\mathcal{P} = \mathcal{NP}$.

152 Finally, recall that a *fixed parameter-tractable* (FPT for short) algorithm, is an algorithm
153 with running time $f(k)n^{\mathcal{O}(1)}$, where f is a computable function and k is the considered
154 parameter. We also make use of what is known as a *linear fpt-reduction*, a type of polynomial
155 reduction such that the size of the parameter of the new problem is linear in regards to the
156 size of the parameter of the original problem. Observe that if we have a linear fpt-reduction
157 from a problem Q with parameter k to a problem Q' with parameter k' and the assumption
158 that Q can not be solved in time $f(k)n_1^{\mathcal{O}(k)}$ (where n_1 is the size of the input of Q), then we
159 can conclude that there is no $f(k')n_2^{\mathcal{O}(k')}$ time algorithm for Q (where n_2 is the size of the
160 input of Q).

161 Let $G = (V, E)$ be a graph. We say that G is *locally irregular* if for every edge $uv \in E$, we
162 have $d(u) \neq d(v)$. Now, let $S \subseteq V$ be such that $G[V \setminus S]$ is a locally irregular graph; any set
163 S that has this property is said to be an *irregulator* of G . For short, we will say that S is an
164 *ir*(G). Moreover, let $I(G)$ be the minimum order that any *ir*(G) can have. We will say that
165 S is a *minimum* irregulator of G , for short S is an *ir**(G), if S is an *ir*(G) and $|S| = I(G)$.

166 We also define the following notion, which generalises *ir*(G). Let $G = (V, E)$ be a graph,
167 $S, X \subseteq V$ and let $G' = G[V \setminus S]$. Now, let $S \subseteq V$ be such that, for each two neighbouring
168 vertices u, v in $X \setminus S$, we have that $d_{G'}(u) \neq d_{G'}(v)$; any set S that has this property is said
169 to be an *irregulator of X in G* , for short *ir*(G, X). We define the notions of *ir**(G, X) and
170 $I(G, X)$ analogously to the previous definitions.

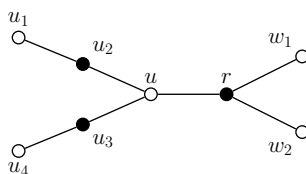
171 We will now provide some lemmas and an observation that will be useful throughout this
172 paper. As the proofs of the following lemmas mainly follow from the definitions, we chose to
173 only include them in the full version of this paper. In the three lemmas below, we investigate
174 the relationship between $I(G)$ and $I(G, X)$.

175 ► **Lemma 1.** *Let $G = (V, E)$ be a graph and let $X \subseteq V$. Then $I(G, X) \leq I(G)$.*

176 ► **Lemma 2.** *Let $G = (V, E)$ be a graph and $S, X \subseteq V$ such that S is an *ir**(G, X). Then,
177 $S \subseteq N[X]$ and $I(G, X) = I(G[N[X]], X)$.*

178 ► **Lemma 3.** *Let $G = (V, E)$ be a graph, and $X_1, \dots, X_n \subseteq V$ such that $N[X_i] \cap N[X_j] = \emptyset$
179 for every $1 \leq i < j \leq n$. Then $\sum_{i=1}^n I(G, X_i) \leq I(G)$.*

180 ► **Lemma 4.** *Let $G = (V, E)$ be a graph, X be a subset of V and S be an *ir*(G). The set
181 $S \cap N[X]$ is an *ir*(G, X) and an *ir*($G[N[X]], X$).*



■ **Figure 1** The gadget used in the proof of Theorem 7. The white and black vertices are used to denote vertices belonging to different bipartitions.

182 The following, almost trivial, observation, will be useful throughout the rest of the paper.

183 ► **Observation 5.** *Let $G = (V, E)$ be a graph and S be an $ir(G)$. Then, for each edge $uv \in E$,*
 184 *if $d(u) = d(v)$, then S contains at least one vertex in $N[\{u, v\}]$. Additionally, for a set*
 185 *$X \subseteq V$, let S^* be an $ir(G[N[X]], X)$. Then for each edge $uv \in E(G[X])$, if $d(u) = d(v)$,*
 186 *then S^* contains at least one vertex in $N[\{u, v\}]$.*

187 3 (Classic) complexity

188 In this section, we deal with the complexity of the problem we introduced. In the following
 189 theorem, we sum up all the families of graphs for which we prove that $I(G)$ is computed in
 190 polynomial time.

191 ► **Theorem 6.** *Let G be a graph. If G is a path, cycle, tree, complete bipartite or a complete*
 192 *graph, then the problem of computing $I(G)$ is in \mathcal{P} .*

193 The result for the case of paths and cycles is proven through induction on the order of
 194 the graph. Then, complete and complete bipartite graphs have a rather trivial structure
 195 in regards to the problem studied here. Finally, the polynomial algorithm for trees follows
 196 directly from upcoming Theorem 14.

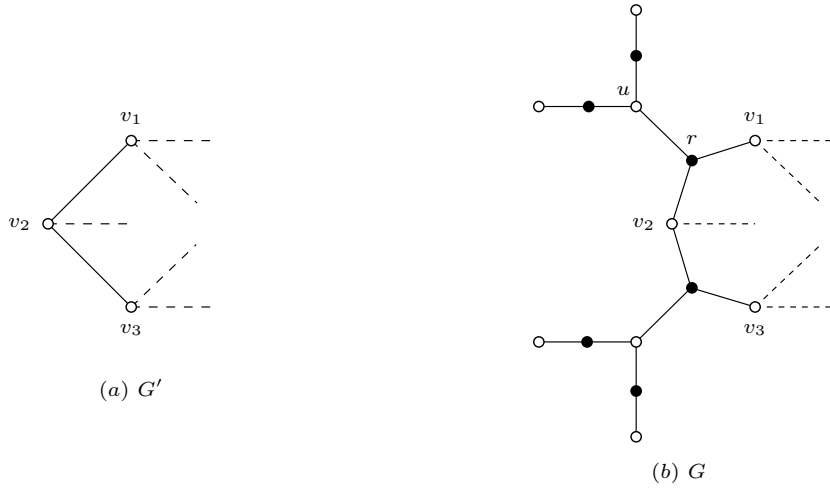
197 3.1 \mathcal{NP} -Hard Cases

198 We now show that finding a minimum irregulator of a graph is \mathcal{NP} -hard. Interestingly,
 199 this remains true even for quite restricted families of graphs, such as cubic (*i.e.*, 3-regular)
 200 bipartite, and subcubic planar bipartite graphs, *i.e.*, planar bipartite graphs of maximum
 201 degree at most 3.

202 ► **Theorem 7.** *Let G be graph and $k \in \mathbb{N}$. Deciding if $I(G) \leq k$ is \mathcal{NP} -complete, even when*
 203 *G is a planar bipartite graph with maximum degree $\Delta \leq 3$.*

204 **Proof.** Since the problem is clearly in \mathcal{NP} , we will focus on proving it is also \mathcal{NP} -hard. The
 205 reduction is from the VERTEX COVER problem, which remains \mathcal{NP} -complete when restricted
 206 to planar cubic graphs [27]. In that problem, a planar cubic graph G and an integer $k \geq 1$
 207 are given as an input. The question is, whether there exists a vertex cover of G of order at
 208 most k . That is, whether there exists a set $VC \subseteq V(G)$ such that for every edge $uv \in E(G)$,
 209 at least one of u and v belongs in VC and $|VC| \leq k$.

210 Let G' be a planar cubic graph and $k \geq 1$ given as input for VERTEX COVER. Let
 211 $|E(G')| = m$. We will construct a planar bipartite graph G as follows; we start with the
 212 graph G' , and modify it by using multiple copies of the gadget, illustrated in Figure 1. Note
 213 that we will be following the naming convention illustrated in Figure 1 whenever we talk
 214 about the vertices of our gadgets. When we say that we *attach* a copy H of the gadget to



■ **Figure 2** The construction in the proof of Theorem 7. The graph G' is the initial planar cubic graph, and G is the graph built during our reduction. In G , the white and black vertices are used to denote vertices belonging to different bipartitions.

215 the vertices v and v' of G' , we mean that we add H to G' , and we identify the vertices w_1
 216 and w_2 to the vertices v and v' respectively. Now, for each edge $vv' \in E(G')$, attach one
 217 copy H of the gadget to the vertices v and v' , and then delete the edge vv' (see Figure 2).
 218 Clearly this construction is achieved in linear time (we have added m copies of the gadget).
 219 Note also that the resulting graph G has $\Delta(G) = 3$ and that the planarity of G' is preserved
 220 since G is constructed by essentially subdividing the edges of G' and adding a tree pending
 221 from each new vertex. Also, G is bipartite. Indeed, observe that after removing the edges
 222 of $E(G')$, the vertices of $V(G')$ form an independent set of G . Furthermore, the gadget is
 223 bipartite, and the vertices w_1, w_2 (that have been identified with vertices of $V(G')$) belong
 224 to the same bipartition (in the gadget). Finally, for any $1 \leq i \leq m$, let H_i be the i^{th}
 225 copy of the gadget attached to vertices of G' . We will also be using the vertices r^i and u^i to denote
 226 the copies of the vertices r and u (respectively) that also belong to H_i .

227 We are now ready to show that the minimum vertex cover of G' has size k' if and only if
 228 $I(G) = k'$.

229 Let VC be a minimum vertex cover of G' and $|VC| = k'$. We will show that the set
 230 $S = VC$ is an $ir(G)$. Let $G^* = G[V(G) \setminus S]$. First, note that S contains only vertices of G' .
 231 Thus, for each i , the vertices of H_i except from r^i , which also remain in G^* , have the same
 232 degree in G' and in G^* . Also note that each vertex of G' is adjacent only to copies of r . It
 233 follows that it suffices to only consider the vertices r^i to show that VC is an $ir(G)$. Now,
 234 for any $1 \leq i \leq m$, consider the vertex r^i . Since VC is a vertex cover of G' , for each edge
 235 $vv' \in E(G')$, VC contains at least one of v and v' . It follows that $d_{G^*}(r^i) \leq 2$. Note also
 236 that $N_{G^*}(r^i)$ contains the vertex $u^i \in V(H_i)$ and possibly one vertex $v \in V(G')$.

237 Also, since we only delete vertices in $V(H_i) \cap V(G')$, we have that $d_{G^*}(u^i) = 3 > d_{G^*}(r^i)$.
 238 In the case where $N_{G^*}(r^i)$ also contains a vertex $v \in V(G')$, the vertex v is adjacent only to
 239 vertices which do not belong in $V(G')$. Thus, $d_{G^*}(v) = d_G(v) = 3 > d_{G^*}(r^i)$. It follows that
 240 r^i has a different degree from all of its neighbours and that VC is an $ir(G)$.

241 Now, we prove that if $I(G) = k'$ then there exists a vertex cover of size at most k' . Assume
 242 that $I(G) = k'$ and let S be an $ir^*(G)$. Observe that since S is an $ir^*(G)$, S contains at least
 243 one vertex of H_i (for each $1 \leq i \leq m$). Let $X_i = V(H_i) \cap V(G')$. To construct a vertex cover

244 VC of G' with $|VC| \leq k'$, we work as follows. For each $1 \leq i \leq m$:

- 245 1. for each vertex $v \in X_i$, if $v \in S$ then put v in VC . Then,
- 246 2. if $S \cap X_i = \emptyset$, put any one of the two vertices of X_i in VC .

247 Observe now that any vertex that is added to VC during step 1. of the above procedure,
 248 also belongs to S and any vertex that is added during step 2. of the above procedure
 249 corresponds to at least one vertex in S . It follows that $|VC| \leq k'$. Also note that VC
 250 contains at least one vertex of X_i , for each i , and that for each $uv \in E(G')$, there exists an i
 251 such that $V(X_i) = \{u, v\}$. Thus VC is indeed a vertex cover of G' .

252 Therefore G' has a minimum vertex cover of size k' if and only if $I(G) = k'$. To complete
 253 the proof note that deciding if $I(G) = k' < k$ for a given k , answers the question whether G'
 254 has a vertex cover of size less than k or not. ◀

255 In the following theorem we show that calculating $I(G)$ is \mathcal{NP} -hard even if G is a cubic
 256 bipartite graph.

257 ▶ **Theorem 8.** *Let G be graph and $k \in \mathbb{N}$. Deciding if $I(G) \leq k$ is \mathcal{NP} -complete even in*
 258 *cubic bipartite graphs.*

259 This theorem is shown through a reduction from the 2-BALANCED 3-SAT, which was
 260 proven to be \mathcal{NP} -complete in [9].

261 4 (In)approximability

262 In the previous section we showed that computing $I(G)$ is \mathcal{NP} -hard, even for graphs G
 263 belonging to quite restricted families of graphs. So the natural question to pose next, which
 264 we investigate in this section, is whether we can approximate $I(G)$. Unfortunately, most of
 265 the results we present below are once again negative.

266 We start with a corollary that follows from the proof of Theorem 7 and the inapproxim-
 267 ability of VERTEX COVER in cubic graphs [17]:

268 ▶ **Corollary 9.** *Given a graph G , it is \mathcal{NP} -hard to approximate $I(G)$ to within a ratio of $\frac{100}{99}$,*
 269 *even if G is bipartite and $\Delta(G) = 3$.*

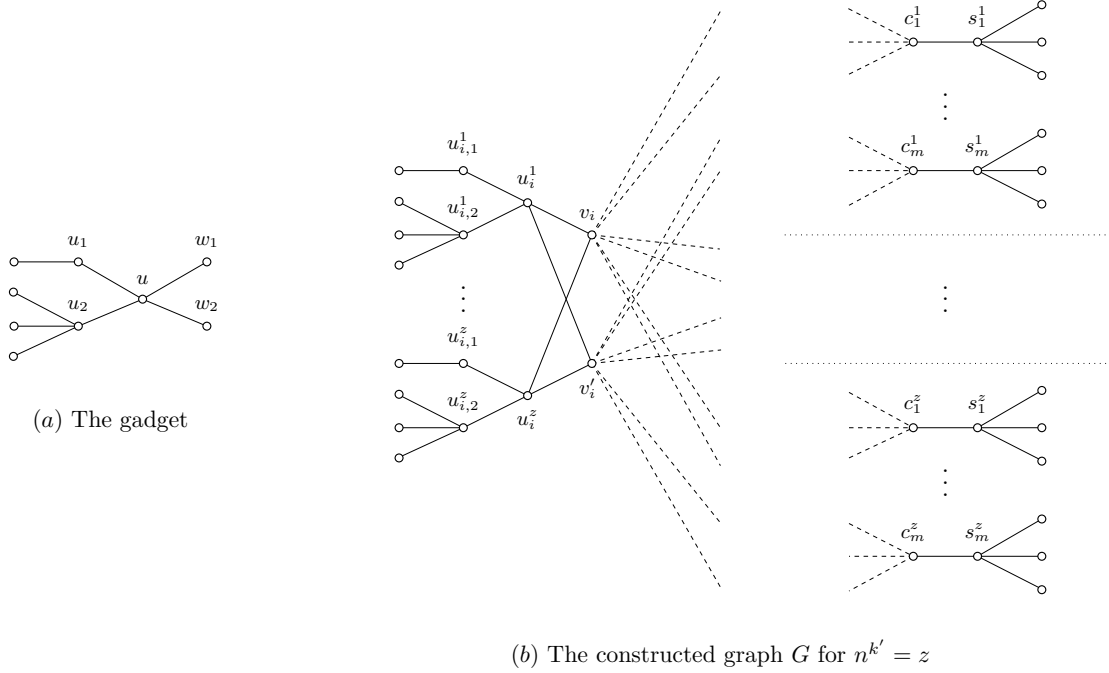
270 Now, we are going to show that there can be no algorithm that approximates $I(G)$ to
 271 within any decent ratio in polynomial time, unless $\mathcal{P} = \mathcal{NP}$, even if G is a bipartite graph
 272 (with no restriction on its maximum degree).

273 ▶ **Theorem 10.** *Let G be a bipartite graph of order N and $k \in \mathbb{N}$ be a constant such that*
 274 *$k \geq 1$. It is \mathcal{NP} -hard to approximate $I(G)$ to within $\mathcal{O}(N^{1-\frac{1}{k}})$.*

275 **Proof.** The proof is by a *gap producing reduction* from 2-BALANCED 3-SAT, which was
 276 proven to be \mathcal{NP} -complete in [9]. In that problem, a 3CNF formula F is given as an input,
 277 comprised by a set C of clauses over a set of Boolean variables X . In particular, we have
 278 that each clause contains exactly 3 literals, and each variable $x \in X$ appears in F exactly
 279 twice as a positive and twice as a negative literal. The question is, whether there exists a
 280 truth assignment to the variables of X satisfying F .

281 Let F be a 3CNF formula with m clauses C_1, \dots, C_m and n variables x_1, \dots, x_n that is
 282 given as input to the 2-BALANCED 3-SAT problem. Let $2k = k' + 1$. Based on the instance
 283 F , we are going to construct a bipartite graph $G = (V, E)$ where $|V| = \mathcal{O}(n^{k'+1})$ and

- 284 ■ $I(G) \leq n$ if F is satisfiable
- 285 ■ $I(G) > n^{k'}$ otherwise.



■ **Figure 3** The construction in the proof of Theorem 10. In subfigure (b), we illustrate how each pair of literal vertices is connected to the rest of the graph. Whenever there is an upper index $1 \leq l \leq n^{k'}$ on a vertex, it is used to denote the l^{th} copy of that vertex. The dashed lines are used to represent the edges between the literal and the clause vertices.

286 To construct $G = (V, E)$, we start with the following graph: for each literal x_i ($\neg x_i$ resp.)
 287 in F , add a *literal vertex* v_i (v'_i resp.) in V , and for each clause C_j of F , add a *clause vertex*
 288 c_j in V . Next, for each $1 \leq j \leq m$, add the edge $v_i c_j$ ($v'_i c_j$ resp.) if the literal x_i ($\neg x_i$ resp.)
 289 appears in C_j according to F . Observe that the resulting graph is bipartite, for each clause
 290 vertex c we have $d(c) = 3$ and for each literal vertex v we have $d(v) = 2$ (since in F , each
 291 variable appears twice as a positive and twice as a negative literal). To finish the construction
 292 of G , we will make use of the gadget shown in Figure 3(a), as well as some copies of S_5 , the
 293 star on 5 vertices. When we say that we *attach* a copy H of the gadget to the vertices v_i
 294 and v'_i (for some $1 \leq i \leq n$), we mean that we add H to G , and we identify the vertices w_1
 295 and w_2 to the vertices v_i and v'_i respectively. Now:

- 296 ■ for each $1 \leq i \leq n$, we attach $n^{k'}$ copies of the gadget to the vertices v_i and v'_i of G .
 297 For convenience, we will give unique names to the vertices corresponding to each gadget
 298 added that way. So, the vertex u_i^l (for $1 \leq l \leq n^{k'}$ and $1 \leq i \leq n$) is used to represent
 299 the vertex u of the l^{th} copy of the gadget attached to v_i and v'_i , and $u_{i,1}^l$ ($u_{i,2}^l$ resp.) is
 300 used to denote the vertex u_1 (u_2 resp.) of that same gadget. Then,
- 301 ■ for each $1 \leq j \leq m$, we add $n^{k'} - 1$ copies of the clause vertex c_j to G , each one of these
 302 copies being adjacent to the same literal vertices as c_j . For $1 \leq l \leq n^{k'}$, the vertex c_j^l is
 303 the l^{th} copy of c_j . Finally,
- 304 ■ for each $1 \leq j \leq m$ and $1 \leq l \leq n^{k'}$, we add a copy of the star on 5 vertices S_5 to G and
 305 identify any degree-1 vertex of S_5 to c_j^l . Let s_j^l be the neighbour of c_j^l that also belongs
 306 to a copy of S_5 .

307 Observe that the resulting graph G (illustrated in Figure 3(b)) remains bipartite and that
 308 this construction is achieved in polynomial time in regards to $n + m$.

309 From the construction of G , we know that for every $1 \leq i \leq n$, $d(v_i) = d(v'_i) = \Theta(n^{k'})$.
 310 So, for sufficiently large n , the only pairs of adjacent vertices of G that have the same degrees
 311 are either the vertices u_i^l and $u_{i,2}^l$, or the vertices c_j^l and s_j^l (for every $1 \leq i \leq n$, $1 \leq l \leq n^{k'}$
 312 and $1 \leq j \leq m$).

313 First, let F be a satisfiable formula and let t be a satisfying assignment of F . Also, let S
 314 be the set of literal vertices v_i (v'_i resp.) such that the corresponding literals x_i ($\neg x_i$ resp.)
 315 are assigned value *true* by t . Clearly $|S| = n$. We will also show that S is an $ir(G)$. Consider
 316 the graph $G' = G[V \setminus S]$. Now, for any $1 \leq i \leq n$, we have that either v_i or v'_i , say v_i ,
 317 belongs to the vertices of G' . Now for every $1 \leq l \leq n^{k'}$, we have that $d_{G'}(u_i^l) = 3$, while
 318 $d_{G'}(u_{i,1}^l) = 2$ and $d_{G'}(u_{i,2}^l) = 4$ (since none of the neighbours of $u_{i,1}^l$ and $u_{i,2}^l$ belongs to S).
 319 Also, for every $1 \leq j \leq m$ and $1 \leq l \leq n^{k'}$, since t is a satisfying assignment of F , $N(c_j^l)$
 320 contains at least one vertex in S . It follows that $d_{G'}(c_j^l) = 3 < 4 = d_{G'}(s_j^l)$. Finally, since S
 321 does not contain any neighbours of v_i , we have that $d_{G'}(v_i) = d_G(v_i) = \mathcal{O}(n^{k'})$. It follows
 322 that S is an $ir(G)$ and thus that $I(G) \leq n$.

323 Now let F be a non-satisfiable formula and assume that there exists an S that is an $ir(G)$
 324 with $|S| \leq n^{k'}$. As usual, let $G' = G[V \setminus S]$. Then:

- 325 1. For every $1 \leq j \leq m$, there exists a literal vertex v such that $v \in N(c_j^l)$ for every
 326 $1 \leq l \leq n^{k'}$. Assume that this is not true for a specific j . Then, since $d_G(c_j^l) = d_G(s_j^l) = 4$,
 327 for every $1 \leq l \leq n^{k'}$, we have that S contains at least one vertex in $N[\{c_j^l, s_j^l\}]$, which
 328 does not belong to the literal vertices. That is, S contains at least one (non-literal) vertex
 329 for each one of the $n^{k'}$ copies of c_j . Observe also that even if this is the case, S would
 330 also have to contain at least one more vertex to, for example, stop $u_{i,2}^1$ and u_i^1 , from
 331 having the same degree in G' . It follows that $|S| > n^{k'}$, which is a contradiction.
- 332 2. For every $1 \leq i \leq n$, S does not contain both v_i and v'_i . Assume this is not true for a
 333 specific i . Then, for every $1 \leq l \leq n^{k'}$, we have that $d_{G'}(u_i^l) = d_{G'}(u_{i,1}^l) = 2$, unless S
 334 also contains an additional vertex of the gadgets attached to v_i and v'_i , for each one of
 335 the $n^{k'}$ such gadgets. It follows that $|S| \geq n^{k'}$. Since we have also assumed that for a
 336 specific i , both v_i and v'_i belong to S , we have that $|S| > n^{k'}$, a contradiction.
- 337 3. For every $1 \leq i \leq n$, S contains at least one of v_i and v'_i . Assume this is not true for
 338 a specific i . Then, for every $1 \leq l \leq n^{k'}$, we have that $d_{G'}(u_i^l) = d_{G'}(u_{i,2}^l) = 4$, unless
 339 S also contains an additional vertex of the gadgets attached to v_i and v'_i , for each one
 340 of the $n^{k'}$ such gadgets. Even if this is the case, S would also have to contain at least
 341 one more vertex to, for example, stop c_1^1 and S_1^1 from having the same degree in G' . It
 342 follows that $|S| > n^{k'}$, which is a contradiction.

343 So from items 2. and 3. above, it follows that for each $1 \leq i \leq n$, S contains exactly one
 344 of v_i and v'_i . Now consider the following truth assignment: we assign the value *true* to every
 345 variable x_i if the corresponding literal vertex v_i belongs in S , and value *false* to every other
 346 variable. Now, from item 1. above, it follows that each clause C_j contains either a positive
 347 literal x_i which has been set to *true*, or a negative literal $\neg x_i$ which has been set to *false*.
 348 Thus F is satisfied, which is a contradiction.

349 Up to this point, we have shown that there exists a graph $G = (V, E)$ with $|V(G)| =$
 350 $N = \mathcal{O}(n^{k'+1})$ where

- 351 ■ $I(G) \leq n$ if F is satisfiable
- 352 ■ $I(G) > n^{k'}$ otherwise.

353 Therefore, we have that $I(G)$ is not $\mathcal{O}(n^{k'-1})$ approximable in polynomial time unless $\mathcal{P} = \mathcal{NP}$.

354 Now, since $N = |V(G)| = \Theta(n^{k'+1})$ and $2k = k' + 1$ we have $\mathcal{O}(n^{k'-1}) = \mathcal{O}(N^{\frac{k'-1}{k'+1}}) =$
 355 $\mathcal{O}(N^{1-\frac{2}{k'+1}}) = \mathcal{O}(N^{1-\frac{1}{k}})$. This ends the proof of this theorem. ◀

23:10 Complexity of Finding Maximum Locally Irregular Induced Subgraphs

356 Now, we consider the case where G is regular bipartite graph. Below we present an
357 upper bound to the size of $I(G)$. This upper bound is then used to obtain a (simple)
358 Δ -approximation of an optimal solution.

359 ► **Theorem 11.** *For any d -regular bipartite graph $G = (L, R, E)$ of order n we have that*
360 $I(G) \geq n/2d$.

361 Now recall that in any bipartite graph G , any bipartition of G is a vertex cover of G .
362 Also observe that any vertex cover of a graph G , is also an irregulator of G . Indeed, deleting
363 the vertices of any vertex cover of G , leaves us with an independent set, which is locally
364 irregular. The next corollary follows from these observations and Theorem 11:

365 ► **Corollary 12.** *For any d -regular bipartite graph $G = (L, R, E)$, any of the sets L and R is*
366 *a d -approximation of $ir^*(G)$.*

367 **5** Parameterised complexity

368 As the problem of computing a minimal irregulator of a given graph G seems to be rather
369 hard to solve, and even to approximate, we focused our efforts towards finding parameterised
370 algorithms that can solve it. First we present an FPT algorithm that calculates $I(G)$ when
371 parameterised by the size of the solution and Δ , the maximum degree of the graph.

372 ► **Theorem 13.** *For a given graph $G = (V, E)$ with $|V| = n$ and maximum degree Δ , and*
373 *for $k \in \mathbb{N}$, there exists an algorithm that decides if $I(G) \leq k$ in time $(2\Delta)^k n^{\mathcal{O}(1)}$.*

374 The main tool we use to show Theorem 13 is Observation 5. Let $G = (V, E)$ be a graph
375 and $k \in \mathbb{N}$. A high level description of our recursive algorithm is as follows: first find an edge
376 $uv \in E$ such that $d(u) = d(v)$. Now, assume that we are making a correct guess of a vertex
377 $w \in N[\{u, v\}] \cap S$ where S is a minimum irregulator. Then, $G_w = G[V \setminus w]$ must have a
378 minimum irregulator of size $|S| - 1$. Note that if we repeat the above process and we make
379 correct guesses, we are going to stop after deleting $|S|$ vertices or when we have deleted k
380 vertices (meaning that $I(G) > k$). Then, by considering all the 2Δ choices for w , we have a
381 running time of $(2\Delta)^k$.

382 We now turn our attention towards graphs that are “close to being trees”, that is graphs
383 of bounded treewidth. In particular, we provide an FPT algorithm that finds a minimum
384 irregulator of G , when parameterised by the treewidth of the input graph and by Δ .

385 ► **Theorem 14.** *For a given a graph $G = (V, E)$ and a nice tree decomposition of G , there*
386 *exists an algorithm that returns $I(G)$ in time $\Delta^{2tw} n^{\mathcal{O}(1)}$, where tw is the treewidth of the*
387 *given decomposition and Δ is the maximum degree of G .*

388 The idea of the proof of Theorem 14, is based on the classic dynamic programming
389 technique on the given nice tree decomposition of G . Let us denote by B_c the bag of vertices
390 of a node c of a nice tree decomposition of G . In essence, for each node c of the tree
391 decomposition, we store the necessary information that allows us to find all the sets that
392 are $ir(G, B_c^\downarrow \setminus B_c)$, where B_c^\downarrow denotes the vertices appearing in a sub-tree rooted at c . Then
393 for the root r of the tree decomposition, we can check which of the stored sets that are
394 $ir(G, B_r^\downarrow \setminus B_r)$, are also $ir(G)$; the minimum such set is an $ir^*(G)$.

395 The running time of our algorithm follows from the size of the tables we keep for these
396 sets. In particular, for each set stored for a node c , for each vertex v of B_c , we keep the
397 degree that we want v to have in the final, locally irregular graph (*i.e.* the graph G after the

398 removal of $ir(G))$ and the degree that v has in $G[B_c^\perp \setminus S]$. This gives us Δ^2 choices for each
 399 vertex of B_c .

400 It is worth noting that the algorithms of Theorem 13 and 14 can be used in order to also
 401 return an $ir^*(G)$.

402 5.1 W-Hardness

403 Observe that both of the algorithms presented above, have to consider Δ as part of the
 404 parameter if they are to be considered as FPT. The natural question to ask at this point is
 405 whether we can have an FPT algorithm, when parameterised only by the size of the solution,
 406 or the treewidth of the input graph. In this section, we give a strong indication towards
 407 the negative answer for both cases, proving that, in some sense, the algorithms provided in
 408 Section 5 are optimal.

409 ► **Theorem 15.** *Let G be a graph and $k \in \mathbb{N}$. Deciding if $I(G) \leq k$ is $W[2]$ -hard, when
 410 parameterised by k .*

411 The proof of Theorem 15 is done through a linear-fpt reduction from the DOMINATING
 412 SET problem, when parameterised by the size of the solution.

413 ► **Theorem 16.** *Let G be a graph with treewidth tw , and $k \in \mathbb{N}$. Deciding if $I(G) = k$ is
 414 $W[1]$ -hard when parameterised by tw .*

415 **Proof.** We will present a reduction from the LIST COLOURING problem: the input consists
 416 of a graph $H = (V, E)$ and a list function $L : V \rightarrow \mathcal{P}(\{1, \dots, k\})$ that specifies the available
 417 colours for each vertex $u \in V$. The goal is to find a proper colouring $c : V \rightarrow \{1, \dots, k\}$ such
 418 that $c(u) \in L(u)$ for all $u \in V$. When such a colouring exists, we say that (H, L) is a *yes-*
 419 *instance* of LIST COLOURING. This problem is known to be $W[1]$ -hard when parameterised
 420 by the treewidth of H [19].

421 Now, starting from an instance (H, L) of LIST COLOURING, we will construct a graph
 422 $G = (V', E')$ (see Figure 4 (a)) such that:

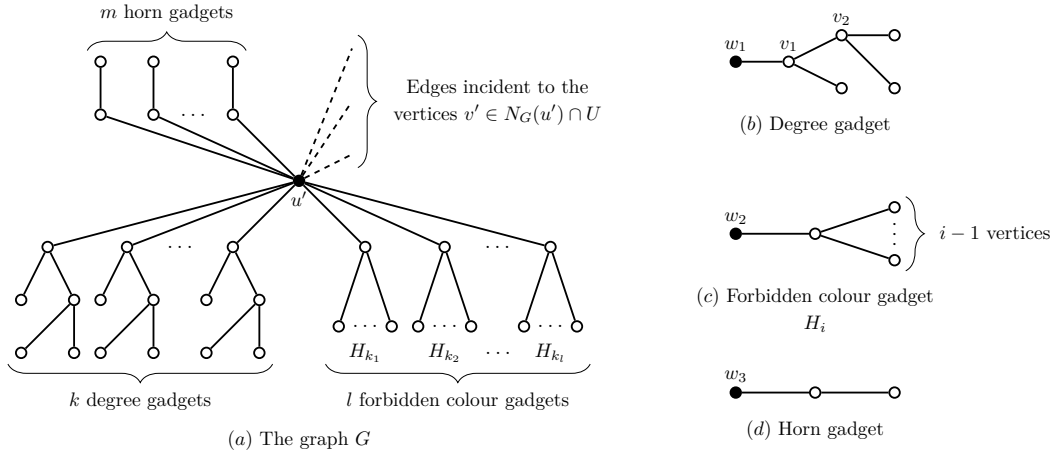
- 423 ■ $|V'| = \mathcal{O}(|V|^6)$,
- 424 ■ $tw(G) = tw(H)$ and
- 425 ■ $I(G) = nk$ if and only if (H, L) is a yes-instance of LIST COLOURING.

426 Before we start with the construction of G , let us give the following observation.

427 ► **Observation 17.** *Let (H, L) be an instance of LIST COLOURING where $H = (V, E)$ and
 428 there exists a vertex $u \in V$ such that $|L(u)| > d(u)$. Then the instance $(H[V \setminus \{u\}], L')$,
 429 where $L'(v) = L(v)$ for all $v \in V \setminus \{u\}$, is a yes-instance of LIST COLOURING if and only if
 430 (H, L) is a yes-instance of LIST COLOURING.*

431 Indeed, observe that for any vertex $u \in V$, by any proper colouring c of H , $c(u)$ only has to
 432 avoid $d(u)$ colours. Since $|L(u)| > d(u)$, we will always have a spare colour to use on u that
 433 belongs in $L(u)$. From the previous observation, we can assume that in our instance, for all
 434 $u \in V$, we have $|L(u)| \leq d(u)$. Furthermore, we can deduce that $k \leq n(n-1)$ as the degree
 435 of any vertex is at most $n-1$. Finally, let us denote by $\bar{L}(u)$ the set $\{0, 1, \dots, k\} \setminus L(u)$. It
 436 is important to note here that for every $u \in V$, the list $L(u)$ contains at least one element
 437 belonging in $\{1, \dots, k\}$. It follows that $\bar{L}(u)$ also contains at least one element, the colour 0.
 438 To sum up, we have that $1 \leq |\bar{L}(u)| \leq k$.

439 Now, we present the three gadgets we are going to use in the construction of G . First,
 440 we have the “forbidden colour gadget” H_i , which is a star with i leaves (see Figure 4(c)).
 441 When we say that we attach a copy of H_i on a vertex v of a graph G , we mean that we



■ **Figure 4** In (a) we illustrate the construction of G , as it is described in the proof of Theorem 16. The black vertex represents every vertex that belongs in U . For the specific vertex u' shown in the figure, we have that $\bar{L}(u) = \{c_1, \dots, c_l\}$ and $k_i = n^3 - c_i$ for all $i = 1, \dots, l$. We also have that $m = 2n^3 - d_G(u) - k - l$.

442 add H_i to G and we identify the vertices v and w_2 (where here and in what follows, we are
 443 using the naming illustrated in Figure 4 when talking about the vertices w_1, w_2, w_3, v_1 and
 444 v_2). The second, will be the “degree gadget”, which is presented in Figure 4(b). Finally, we
 445 have the “horn gadget”, which is a path on three vertices (see Figure 4(d)). We define the
 446 operation of attaching these two gadgets on a vertex v of a graph G similarly to how we
 447 defined this operation for the forbidden colour gadget (each time using the appropriate w_1
 448 or w_3 , according to if it is a degree or a horn gadget respectively).

449 In order to construct G , we start from a copy of H . Let us use $G|_H$ to denote the copy of
 450 H that lies inside of G and, for each vertex $u \in V$, let u' be its copy in V' . We will call the
 451 set of these vertices U . That is, $U = \{v \in V(G|_H)\}$. Then, we are going to attach several
 452 copies of each gadget to u' , for each vertex $u' \in U$. We start by attaching k copies of the
 453 degree gadget to each vertex $u' \in U$. Then, for each $u \in V$ and each $i \in \bar{L}(u)$, we attach one
 454 copy of the forbidden colour gadget H_{2n^3-i} to the vertex u' . Finally, for each $u' \in U$, we
 455 attach to u' as many copies of the horn gadget as are needed, in order to have $d_G(u') = 2n^3$.

456 Before we continue, observe that, for sufficiently large n , we have attached more than n^3
 457 horn gadgets to each vertex of U . Indeed, before attaching the horn gadgets, each vertex
 458 $u' \in U$ has $d_G(u) \leq n - 1$ neighbours in U , k neighbours from the degree gadgets and at
 459 most $k < n^2$ neighbours from the forbidden colour gadgets (recall that $|\bar{L}(u)| \leq k$). We will
 460 now show that $|V'| = \mathcal{O}(n^6)$. For that purpose, let us calculate the number of vertices in
 461 all the gadgets attached to a single vertex $u' \in U$. First, we have $5k < 5n^2$ vertices in the
 462 degree gadgets. Then, we have less than $4n^3$ vertices in the horn gadgets (as we have less
 463 than $2n^3$ such gadgets). Finally, we have at most $k < n^2$ forbidden colour gadgets, each
 464 one of which containing at most $2n^3$ vertices. So, for each vertex $u' \in U$, we have at most
 465 $2n^5 + 4n^3 + 5n^2$ vertices in the gadgets attached to u' . Therefore, we have $|V'| = \mathcal{O}(n^6)$.

466 Before we prove that $I(G) \leq nk$ if and only if (H, L) is a yes-instance of LIST COLOURING,
 467 we need to argue about two things. First, about the treewidth of the graph G and second,
 468 about the minimum value of $I(G)$. Since our construction only attaches trees to each
 469 vertex of $G|_H$ (and recall that a tree has a treewidth of 1 by definition), we know that
 470 $tw(G) = tw(G|_H) = tw(H)$. As for $I(G)$, we will show that it has to be at least equal to nk .

471 For that purpose we have the following two claims.

472 ▷ Claim 18. Let S be an $ir(G)$ and $S \cap U \neq \emptyset$. Then $|S| > n^3$.

473 ▷ Claim 19. Let S be an $ir(G)$ and $S \cap U = \emptyset$. Then $|S| \geq nk$. In particular, S includes at
474 least one vertex from each copy of the degree gadget used in the construction of G .

475 By the previous two claims, we conclude that $I(G) \geq nk$. We are ready to show that, if
476 (H, L) is a yes-instance of LIST COLOURING, then there exists a set $S \subseteq V'$ such that S is
477 an $ir(G)$ and $|S| = nk$. Let c be a proper colouring of H such that $c(u) \in L(u)$ for all $u \in V$.
478 We will construct an $ir(G)$ as follows. For each $u \in V$, we partition (arbitrarily) the k degree
479 gadgets attached to the vertex u' to $c(u)$ “good” and $(k - c(u))$ “bad” degree gadgets. For
480 each good degree gadget, we add the copy of the vertex v_1 of that gadget to S and for each
481 bad degree gadget we add the copy of the vertex v_2 of that gadget to S . This process creates
482 a set S of size nk , as it includes k distinguished vertices for each vertex $u' \in U$.

483 Now we need to show that S is an $ir(G)$. Let $G' = G[V' \setminus S]$; observe that each vertex
484 $u' \in U$ has degree $d_{G'}(u') = 2n^3 - c(u)$. Therefore, u' does not have the same degree as any
485 of its neighbours that do not belong in U . Indeed, for every $v \in N_{G'}(u') \setminus U$, we have that
486 $d_{G'}(v) \in \{1, 2\}$ (if v belongs to a bad degree or a horn gadget) or $d_{G'}(v) \in \{2n^3 - i : i \in \bar{L}(u)\}$
487 (if v belongs to a forbidden colour gadget). Furthermore, since c is a proper colouring of H ,
488 for all $uv \in E$, we have that $c(u) \neq c(v)$. This gives us that for any edge $u'v' \in E'$ with
489 $u', v' \in U$, we have that $d_{G'}(u') = 2n^3 - c(u) \neq 2n^3 - c(v) = d_{G'}(v')$.

490 So, we know that for every vertex $u' \in U$, there is no vertex $w \in N_{G'}(u')$ such that
491 $d_{G'}(u') = d_{G'}(w)$. It remains to show that, in G' , there exist no two vertices belonging to the
492 same gadget, which have the same degrees. First of all, we have that S does not contain any
493 vertex from any of the horn and forbidden colour gadgets, nor from U . Thus any adjacent
494 vertices belonging to these gadgets have different degrees. Last, it remains to check the
495 vertices of the degree gadgets. Observe that for any copy of the degree gadget, S contains
496 either v_1 or v_2 . In both cases, after the deletion of the vertices of S , any adjacent vertices
497 belonging to any degree gadget have different degrees. Therefore, S is an $ir(G)$ of order nk
498 and since $I(G) \geq nk$ we have that $I(G) = nk$.

499 Now, for the opposite direction, assume that there exists a set $S \subseteq V'$ such that S is an
500 $ir^*(G)$ and $|S| = nk$. Let $G' = (V'', E'')$ be the graph $G[V' \setminus S]$. It follows from Claim 18
501 and Claim 19, that $S \cap U = \emptyset$ and that S contains exactly one vertex from each copy of
502 the degree gadget in G and no other vertices. Consider now the colouring c of H defined as
503 $c(u) = 2n^3 - d_{G'}(u')$. We will show that c is a proper colouring for H and that $c(u) \in L(u)$.
504 First, we have that c is a proper colouring of H . Indeed, for any edge $uv \in E$, there exists an
505 edge $u'v' \in E''$ (since $S \cap U = \emptyset$). Since G' is locally irregular we have that $d_{G'}(u') \neq d_{G'}(v')$,
506 and thus $c(u) \neq c(v)$. It remains to show that $c(u) \in L(u)$ for all $u \in V$. First observe that,
507 during the construction of G , we attached exactly k degree gadgets to each $u' \in U$. It follows
508 that $d_{G'}(u') = 2n^3 - j$ and $c(u) = j$ for a $j \in \{0, 1, \dots, k\}$. It is sufficient to show that
509 $j \notin \bar{L}(u)$. Since S contains only vertices from the copies of the degree gadgets, we have that
510 each $u' \in U$ has exactly one neighbour of degree $2n^3 - i$ for each $i \in \bar{L}(u)$ (this neighbour
511 is a vertex of the H_i forbidden colour gadget that was attached to u'). Furthermore, for
512 all $u' \in U$, since G' is locally irregular, we have that $d_{G'}(u') \neq 2n^3 - i$ for all $i \in \bar{L}(u)$.
513 Equivalently, $d_{G'}(u') = 2n^3 - j$ for any $j \in L(u)$. Thus, $c(u) \in L(u)$ for all $u \in V$. ◀

514 Note that the reductions presented in the proofs of Theorem 15 and Theorem 16 are
515 linear fpt-reductions. Additionally we know that

516 ■ there is no algorithm that answers if a graph G of order n has a Dominating Set of size
517 at most k in time $f(k)n^{o(k)}$ unless the ETH fails [26] and

518 ■ there is no algorithm that answers if an instance (G, L) of the LIST COLOURING is a
 519 yes-instance in time $\mathcal{O}^*(f(tw)n^{o(tw)})$ unless the ETH fails [19].
 520 So, the following corollary holds.

521 ► **Corollary 20.** *Let G be a graph of order n and assume the ETH. For $k \in \mathbb{N}$, there is no*
 522 *algorithm that decides if $I(G) \leq k$ in time $f(k)n^{o(k)}$. Furthermore, assuming that G has*
 523 *treewidth tw , there is no algorithm that computes $I(G)$ in time $\mathcal{O}^*(f(tw)n^{o(tw)})$.*

524 6 Conclusion

525 In this work we introduce the problem of identifying the largest locally irregular induced
 526 subgraph of a given graph. There are many interesting directions that could be followed for
 527 further research. An obvious one is to investigate whether the problem of calculating $I(G)$
 528 remains \mathcal{NP} -hard for other, restricted families of graphs. The first candidate for such a family
 529 would be the one of chordal graphs. On the other hand, there are some interesting families,
 530 for which the problem of computing an optimal irregularator could be decided in polynomial
 531 time, such as split graphs. Also, it could be feasible to conceive approximation algorithms for
 532 regular bipartite graphs, which have a better approximation ratio than the (simple) algorithm
 533 we present. The last aspect we find intriguing, is to study the parameterised complexity of
 534 calculating $I(G)$ when considering other parameters, like the size of the minimum vertex
 535 cover of G , with the goal of identifying a parameter that suffices, by itself, in order to have an
 536 FPT algorithm. Finally, it is worth investigating whether calculating $I(G)$ could be done in
 537 FPT time (parameterised by the size of the solution) in the case where G is a planar graph.

538 References

- 539 1 Agostinho Agra, Geir Dahl, Torkel Andreas Haufmann, and Sofia J. Pinheiro. The k -regular
 540 induced subgraph problem. *Discrete Applied Mathematics*, 222:14–30, 2017. doi:10.1016/j.
 541 dam.2017.01.029.
- 542 2 Yousef Alavi, Alfred Boals, Gary Chartrand, Ortrud Oellermann, and Paul Erdős. K -path
 543 irregular graphs. *Congressus Numerantium*, 65, 01 1988.
- 544 3 Yousef Alavi, Gary Chartrand, Fan R. K. Chung, Paul Erdős, Ronald L. Graham, and
 545 Ortrud R. Oellermann. Highly irregular graphs. *Journal of Graph Theory*, 11(2):235–249,
 546 1987. doi:10.1002/jgt.3190110214.
- 547 4 Akhbar Ali, Gary Chartrand, and Ping Zhang. *Irregularity in Graphs*. Springer briefs in
 548 mathematics. Springer, 2021. doi:https://doi.org/10.1007/978-3-030-67993-4.
- 549 5 Yuichi Asahiro, Hiroshi Eto, Takehiro Ito, and Eiji Miyano. Complexity of finding maximum
 550 regular induced subgraphs with prescribed degree. *Theoretical Computer Science*, 550:21–35,
 551 2014. doi:10.1016/j.tcs.2014.07.008.
- 552 6 Olivier Baudon, Julien Bensmail, Jakub Przybyło, and Mariusz Woźniak. On decomposing
 553 regular graphs into locally irregular subgraphs. *European Journal of Combinatorics*, 49:90–104,
 554 2015. doi:10.1016/j.ejc.2015.02.031.
- 555 7 Rémy Belmonte and Ignasi Sau. On the complexity of finding large odd induced subgraphs
 556 and odd colorings. *Algorithmica*, 83(8):2351–2373, 2021. doi:10.1007/s00453-021-00830-x.
- 557 8 Julien Bensmail, Martin Merker, and Carsten Thomassen. Decomposing graphs into a constant
 558 number of locally irregular subgraphs. *European Journal of Combinatorics*, 60:124–134, 2017.
 559 doi:10.1016/j.ejc.2016.09.011.
- 560 9 Piotr Berman, Marek Karpinski, and Alex D. Scott. Approximation hardness of short symmetric
 561 instances of MAX-3SAT. *Electronic Colloquium on Computational Complexity*, (049), 2003.
 562 URL: https://eccc.weizmann.ac.il/eccc-reports/2003/TR03-049/index.html.

- 563 10 Hans L. Bodlaender. A partial k -arboretum of graphs with bounded treewidth. *Theoretical*
564 *Computer Science*, 209(1):1–45, 1998. doi:10.1016/S0304-3975(97)00228-4.
- 565 11 Leizhen Cai. Fixed-parameter tractability of graph modification problems for hereditary
566 properties. *Information Processing Letters*, 58(4):171–176, 1996. doi:10.1016/0020-0190(96)
567 00050-6.
- 568 12 Gary Chartrand, Paul Erdős, and Ortrud Oellermann. How to define an irregular graph. *The*
569 *College Mathematics Journal*, 19, 01 1988. doi:10.2307/2686701.
- 570 13 Gary Chartrand, Michael Jacobon, Jenö Lehel, Ortrud Oellermann, Sergio Ruiz, and Farrokh
571 Saba. Irregular networks. *Congressus Numerantium*, 64, 01 1986.
- 572 14 Jianer Chen, Benny Chor, Mike Fellows, Xiuzhen Huang, David W. Juedes, Iyad A. Kanj,
573 and Ge Xia. Tight lower bounds for certain parameterized np-hard problems. *Information*
574 *and Computation*, 201(2):216–231, 2005. doi:10.1109/CCC.2004.1313826.
- 575 15 Jianer Chen, Xiuzhen Huang, Iyad A. Kanj, and Ge Xia. On the computational hardness
576 based on linear FPT-reductions. *Journal of Combinatorial Optimization*, 11(2):231–247, 2006.
577 doi:10.1007/s10878-006-7137-6.
- 578 16 Jianer Chen, Xiuzhen Huang, Iyad A. Kanj, and Ge Xia. Strong computational lower bounds
579 via parameterized complexity. *Journal of Computer and System Sciences*, 72(8):1346–1367,
580 2006. doi:10.1016/j.jcss.2006.04.007.
- 581 17 Miroslav Chlebík and Janka Chlebíková. Complexity of approximating bounded variants of
582 optimization problems. *Theor. Comput. Sci.*, 354(3):320–338, 2006. doi:https://doi.org/
583 10.1016/j.tcs.2005.11.029.
- 584 18 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*.
585 Springer, 2012. doi:10.1007/978-3-662-53622-3.
- 586 19 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts
587 in Computer Science. Springer, 2013. doi:https://doi.org/10.1007/978-1-4471-5559-1.
- 588 20 Alan M. Frieze, Ronald J. Gould, Michal Karonski, and Florian Pfender. On graph irregularity
589 strength. *Journal of Graph Theory*, 41(2):120–137, 2002. doi:10.1002/jgt.10056.
- 590 21 M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of*
591 *NP-Completeness*. W. H. Freeman, 1979.
- 592 22 Michał Karoński, Tomasz Łuczak, and Andrew Thomason. Edge weights and vertex colors.
593 *Journal of Combinatorial Theory*, 91:151–157, 05 2004. doi:10.1016/j.jctb.2003.12.001.
- 594 23 Subhash Khot and Venkatesh Raman. Parameterized complexity of finding subgraphs with
595 hereditary properties. *Theoretical Computer Science*, 289(2):997–1008, 2002. doi:10.1016/
596 S0304-3975(01)00414-5.
- 597 24 John M. Lewis and Mihalis Yannakakis. The node-deletion problem for hereditary properties
598 is np-complete. *Journal of Computer and System Sciences*, 20(2):219–230, 1980. doi:10.1016/
599 0022-0000(80)90060-4.
- 600 25 Carla Negri Lintzmayer, Guilherme Oliveira Mota, and Maycon Sambinelli. Decomposing
601 split graphs into locally irregular graphs. *Discrete Applied Mathematics*, 292:33–44, 2021.
602 doi:10.1016/j.entcs.2019.08.053.
- 603 26 Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Lower bounds based on the exponential
604 time hypothesis. *Bulletin of the European Association for Theoretical Computer Science*,
605 105:41–72, 2011. URL: <http://eatcs.org/beatcs/index.php/beatcs/article/view/92>.
- 606 27 Bojan Mohar. Face covers and the genus problem for apex graphs. *J. Comb. Theory, Ser. B*,
607 82(1):102–117, 2001. doi:https://doi.org/10.1006/jctb.2000.2026.
- 608 28 Hannes Moser and Dimitrios M. Thilikos. Parameterized complexity of finding regular induced
609 subgraphs. *Journal of Discrete Algorithms*, 7(2):181–190, 2009. doi:10.1016/j.jda.2008.09.
610 005.
- 611 29 Jakub Przybyło. Irregularity strength of regular graphs. *Electronic Journal of Combinatorics*,
612 15, 06 2008. doi:10.37236/806.
- 613 30 Jakub Przybyło. On decomposing graphs of large minimum degree into locally irregular
614 subgraphs. *Electronic Journal of Combinatorics*, 23(2):2–31, 2016. doi:10.37236/5173.

615 **A** Omitted proofs616 **A.1 Proof of Theorem 13**

617 Let us first present the following lemma:

618 **► Lemma 21.** *Let $G = (V, E)$ be a graph such that, G is not locally irregular, and S be an*
 619 *$ir^*(G)$. Furthermore let $G_v = (V', E')$ be the graph $G[V \setminus \{v\}]$ for a vertex $v \in S$. Then*
 620 $I(G_v) = I(G) - 1$.

621 **Proof.** First observe that $S' = S \setminus \{v\}$ must be an $ir(G_v)$ as $G_v[V' \setminus S'] = G[V \setminus S]$. It
 622 follows that $I(G_v) \leq I(G) - 1$. Assume that $I(G_v) < I(G) - 1$. Then there exists an S'' such
 623 that $|S''| < I(G) - 1$ and S'' is an $ir(G_v)$. Since $G_v[V' \setminus S''] = G[V \setminus (S'' \cup \{v\})]$, we have
 624 that $S'' \cup \{v\}$ is an $ir(G)$ and $|S'' \cup \{v\}| = |S''| + 1 < I(G)$. This is a contradiction. ◀

625 Now, we are ready to present the proof of the theorem.

626 **Proof of Theorem 13.** In order to decide if $I(G) \leq k$ we are going to use a recursive
 627 algorithm. The algorithm has input (G, k) , where $G = (V, E)$ is a graph and $k \geq 0$ is an
 628 integer. The basic idea of this algorithm, is to take advantage of Observation 5. We present
 629 the exact procedure in Algorithm 1.

 630 **■ Algorithm 1** [IsIrregular(G, k) decision function]

Input: A graph $G = (V, E)$ and an integer $k \geq 0$.**Output:** Is $I(G) \leq k$ or not?

```

1: if  $G$  is irregular then
2:   return yes
3: else if  $k = 0$  then
4:   return no
5: else ▷  $k > 0$  and  $G$  is not irregular
6:    $ans \leftarrow no$ 
7:   find an edge  $vu \in E$  such that  $d_G(v) = d_G(u)$ 
8:   for all  $w \in N_G[\{u, v\}]$  do
9:     set  $G_w = G[V \setminus \{w\}]$ 
10:    if IsIrregular( $G_w, k - 1$ ) returns yes then
11:       $ans \leftarrow yes$ 
12:   return  $ans$ 

```

630 Now, let us argue about the correctness and the efficiency of this algorithm. We claim
 631 that for any graph $G = (V, E)$ and any integer $k \geq 0$, Algorithm 1 returns yes if $I(G) \leq k$
 632 and no otherwise. Furthermore, the number of steps that the algorithm requires, is $f(k, n) =$
 633 $(2\Delta)^k n^{\mathcal{O}(1)}$, where $n = |V|$. We will prove this by induction on k .

634 **Base of the induction ($k = 0$):** Here, we only need to check if G is locally irregular.
 635 Algorithm 1 does this in line 1 and returns yes if it is (line 2) and no otherwise (line 4).
 636 Furthermore, we can check if G is locally irregular in polynomial time. So, the claim is true
 637 for the base.

638 **Induction hypothesis ($k = k_0 \geq 0$):** We assume that we have a $k_0 \geq 0$ such that
 639 Algorithm 1 can decide if any graph G with n vertices and maximum degree Δ has $I(G) \leq k_0$
 640 in $f(k_0, n) = (k_0 + 1)(2\Delta)^{k_0} n^{\mathcal{O}(1)}$ steps.

641 **Induction step ($k = k_0 + 1$):** Let $G = (V, E)$ be a graph. If G is locally irregular
 642 then $I(G) = 0$ and Algorithm 1 answers correctly (in line 2). Assume that G is not locally

643 irregular; then there exist an edge $vu \in E$ such that $d_G(v) = d_G(u)$. Now, let S be an
 644 $ir^*(G)$. It follows from Observation 5 that S must include at least one vertex $w \in N_G[\{v, u\}]$.
 645 Since Algorithm 1 considers all the vertices in $N_G[\{v, u\}]$, at some point it also considers
 646 the vertex $w \in S \cap N_G[\{v, u\}]$. Now, observe that for any $x \in S$, the set $S_x = S \setminus \{x\}$ is an
 647 $ir^*(G_x)$, where $G_x = G[V \setminus \{x\}]$. Furthermore, by Lemma 21, we have $I(G_x) \leq k - 1 = k_0$
 648 iff $I(G) \leq k$. By the induction hypothesis, we know that the algorithm answers correctly for
 649 all the instances (G_x, k_0) . Thus, if $I(G) \leq k = k_0 + 1$, there must exist one instance (G_w, k_0) ,
 650 where $w \in S \cap N_G[\{v, u\}]$, for which the Algorithm 1 returns yes. Therefore the algorithm
 651 answers for $(G, k_0 + 1)$ correctly. Finally, this process request $n^{\mathcal{O}(1)}$ steps in order to check
 652 if the graph is locally irregular and $2\Delta f(k - 1, n - 1)$ steps (by induction hypothesis) in
 653 order to check if for any graph G_x we have $I(G_x) \leq k - 1 = k_0$ (where $x \in N[\{u, v\}]$). So,
 654 the algorithm decides in $n^{\mathcal{O}(1)} + 2\Delta f(k - 1, n - 1) \leq n^{\mathcal{O}(1)} + 2\Delta k(2\Delta)^{k-1}(n - 1)^{\mathcal{O}(1)} \leq$
 655 $n^{\mathcal{O}(1)} + k(2\Delta)^k n^{\mathcal{O}(1)} \leq (k + 1)(2\Delta)^k n^{\mathcal{O}(1)}$ steps. Finally, note that $k \leq n - 1$, and the result
 656 follows. \blacktriangleleft

657 A.2 Proof of Theorem 14

658 **Proof.** As the techniques we are going to use are standard, we are sketching some of the
 659 introductory details. For more details on tree decompositions (definition and terminology)
 660 see [19]. We are going to perform dynamic programming on the nodes of the given nice tree
 661 decomposition (see [10] for the definition of a nice tree decomposition). For a node t of the
 662 given tree decomposition of G , we denote by B_t the bag of this node and by B_t^\downarrow the set of
 663 vertices of the graph that appears in the bags of the nodes of the subtree with t as a root.
 664 Observe that $B_t \subseteq B_t^\downarrow$.

665 The idea behind our algorithm, is that for each node t we store all the sets $S \subseteq B_t^\downarrow$ such
 666 that S is an $ir(G, B_t^\downarrow \setminus B_t)$. We will also store the necessary ‘‘conditions’’ (explained more
 667 in what follows) such that if there exists a set S' , where $S' \setminus S \subseteq V \setminus B_t^\downarrow$, that meets these
 668 conditions, then S' is an $ir(G, B_t^\downarrow)$. Observe that if we manage to do such a thing for every
 669 node of the tree decomposition, then we can find $I(G)$. To do so, it suffices to check the
 670 size of all the irregularators we stored for the root r of the tree decomposition, which also
 671 meet the conditions we have set. In that way, we can find a set S that is an $ir(G, B_r^\downarrow \setminus B_r)$,
 672 satisfies our conditions and is of minimum order, and since $B_r^\downarrow = V$, this set S is a minimum
 673 irregularator of G and $I(G) = |S|$.

674 Let us now present the actual information we are keeping for each node. Assume that t
 675 is a node of the tree decomposition and $S \subseteq B_t^\downarrow$ is an irregularator of $B_t^\downarrow \setminus B_t$ in G , *i.e.*, S is
 676 an $ir(G, B_t^\downarrow \setminus B_t)$. For this S we want to remember which vertices of B_t belong to S as well
 677 as the degrees of the vertices $v \in B_t \setminus S$ in $G[B_t^\downarrow \setminus B_t]$. This can be done by keeping a table
 678 D of size $tw + 1$ where, if $v \in B_t \setminus S$ we set $D(v) = d_{G[B_t^\downarrow \setminus B_t]}(v)$ and if $v \in B_t \cap S$ we set
 679 $D(v) = \emptyset$ (slightly abusing the notation, by $D(v)$ we mean the position in the table D that
 680 corresponds to the vertex v). Like we have already said, we are going to keep some additional
 681 information about the conditions that could allow these sets to be extended to irregularators
 682 of B_t^\downarrow in G if we add vertices of $V \setminus B_t^\downarrow$. For that reason, we are also going to keep a table
 683 with the ‘‘target degree’’ of each vertex; in this table we assign to each vertex $v \in B_t \setminus S$
 684 a degree d_v such that, if there exists S' where $S' \setminus S \subseteq V \setminus B_t^\downarrow$ and for all $v \in B_t \setminus S$ we
 685 have $d_{G[V \setminus S']}(v) = d_v$, then S is an $ir(G, B_t^\downarrow)$. This can be done by keeping a table T of size
 686 $tw + 1$ where for each $v \in B_t \setminus S$ we set $T(v) = i$, where i is the target degree, and for each
 687 $v \in B_t \cap S$ we set $T(v) = \emptyset$. Such tables T will be called *valid* for S in B_t . Finally, we are
 688 going to keep the set $X = S \cap B_t$ and the value $min = |S|$. Note that the set X does not
 689 gives us any extra information, but we keep it as it will be useful to refer to it directly.

23:18 Complexity of Finding Maximum Locally Irregular Induced Subgraphs

690 To sum up, for each node t of the tree decomposition of G , we keep a set of quadruples
 691 (X, D, T, min) , each quadruple corresponding to a valid combination of a set S that is
 692 an $ir(G, B_t^\downarrow \setminus B_t)$ and the target degrees for the vertices of $B_t \setminus S$. Here it is important
 693 to say that when treating the node B_t , for every two quadruples (X_1, D_1, T_1, min_1) and
 694 (X_2, D_2, T_2, min_2) such that for all $v \in B_t$ we have that $D_1(v) = D_2(v)$ and $T_1(v) = T_2(v)$
 695 (this indicates that $X_1 = X_2$ as well), then we are only going to keep the quadruple with the
 696 minimum value between min_1 and min_2 as we will prove that this is enough in order to find
 697 $I(G)$.

698 \triangleright **Claim 22.** Assume that for a node t , we have two sets S_1 and S_2 that are both
 699 $ir(G, B_t^\downarrow \setminus B_t)$, and that T is a target table that is common to both of them. Furthermore,
 700 assume that $(X_1, D_1, T, |S_1|)$ and $(X_2, D_2, T, |S_2|)$ are the quadruples we have to store for
 701 S_1 and S_2 respectively (both respecting T), with $D_1(v) = D_2(v)$ for every $v \in B_t$. Then
 702 for any set $S \subseteq V \setminus B_t^\downarrow$ such that $d_{G[V \setminus (S_1 \cup S)]}(v) = T(v)$ for all $v \in B_t$, we also have that
 703 $d_{G[V \setminus (S_2 \cup S)]}(v) = T(v)$ for all $v \in B_t$.

704 *Proof.* Assume that we have such an S for S_1 , let v be a vertex in B_t and $H = G[v \cup ((V \setminus$
 705 $B_t^\downarrow) \setminus S)]$ (observe that H does not depend on S_1 or S_2). Since $d_{G[V \setminus (S_1 \cup S)]}(v) = T(v)$, we
 706 know that in the graph H , v has exactly $T(v) - D_1(v)$ neighbours (as $D_1(v) = d_{G[B_t^\downarrow \setminus S_1]}(v)$).
 707 Now, since $D_1(v) = D_2(v) = d_{G[B_t^\downarrow \setminus S_2]}(v)$ we have that $d_{G[V \setminus S_2 \cup S]}(v) = T(v)$. Therefore,
 708 the claim holds. \triangleleft

709 Simply put, Claim 22 states that for any two quadruples $Q_1 = (X, D, T, min_1)$ and
 710 $Q_2 = (X, D, T, min_2)$, any extension S of S_1 is also an extension of S_2 (where S_1 and S_2 are
 711 the two sets that correspond to Q_1 and Q_2 respectively). Therefore, in order to find the
 712 minimum solution, it is sufficient to keep the quadruple that has the minimum value between
 713 min_1 and min_2 .

714 Now we are going to explain how we create all the quadruples (X, D, T, min) for each
 715 type of node in the tree decomposition. First we have to deal with the Leaf Nodes. For a Leaf
 716 node t we know that $B_t = B_t^\downarrow = \emptyset$. Therefore, we have only one quadruple (X, D, T, min) ,
 717 where the size of both D and T is zero (so we do not need to keep any information in them),
 718 $S = \emptyset$ and $min = |S| = 0$.

719 Now let t be an Introduce node; assume that we have all the quadruples (X, D, T, min) for
 720 its child c and let v be the introduced vertex. By construction, we know that v is introduced
 721 in B_t and thus it has no neighbours in $B_t^\downarrow \setminus B_t$. It follows that if $S \subseteq B_c^\downarrow$ is an irregularator
 722 for $B_c^\downarrow \setminus B_c$, then both S and $S \cup \{v\}$ are irregularators for $B_t^\downarrow \setminus B_t$ in G . Furthermore, there
 723 is no set $S \subseteq B_t^\downarrow \setminus \{v\}$ that is an irregularator of $B_t^\downarrow \setminus B_t$ and is not an irregularator of $B_c^\downarrow \setminus B_c$.
 724 So, we only need to consider two cases for the quadruples we have to store for c ; if v belongs
 725 in the under-construction irregularator of $B_t^\downarrow \setminus B_t$ in G or not.

726 *Case 1. (v is in the irregularator):* Observe that for any S that is an $ir(G, B_c^\downarrow \setminus B_c)$,
 727 which is stored in the quadruples of B_c , for every $u \in B_c \setminus S$, we have that $d_{G[B_c^\downarrow \setminus S]}(u) =$
 728 $d_{G[B_t^\downarrow \setminus (S \cup \{v\})]}(u)$. Moreover, for any target table T which is valid for S in c , the target table
 729 T' is valid for $S \cup \{v\}$ in t , where T' is almost the same as T , the only difference being that T'
 730 also contains the information about v , i.e., $T'(v) = \emptyset$. So, for each quadruple (X, D, T, min)
 731 in c , we need to create one quadruple $(X \cup \{v\}, D', T', min + 1)$ for t , where D' is the almost
 732 the same as D , except that it also contains the information about v , i.e., $D'(v) = \emptyset$.

733 *Case 2. (v is not in the irregularator):* Let $q = (X, D, T, min)$ be a stored quadruple of
 734 c and S be the corresponding $ir(G, B_c^\downarrow \setminus B_c)$. We will first explain how to construct D' of
 735 t , based on q . Observe that the only change between $G[B_c^\downarrow \setminus S]$ and $G[B_t^\downarrow \setminus S]$, is that in
 736 the latter there exist some new edges from v to some of the vertices of B_c . Therefore, for

737 each vertex $u \in B_c \setminus X$ we set $D'(u) = D(u) + 1$ if $u \in N[v]$ and $D'(u) = D(u)$ otherwise.
 738 Finally, for the introduced vertex v , we set $D'(v) = |N(v) \cap (B_c \setminus X)|$. We will now treat
 739 the target degrees for t . Observe that the target degrees for each vertex in $B_t \setminus \{v\}$ are the
 740 same as in T , since v only has edges incident to vertices in B_t . Now, we only need to decide
 741 which are the valid targets for v . Since $d_{G[B_t^\downarrow \setminus S]}(v) = D'(v)$, we know that for every target
 742 t' , we have that $D'(v) \leq t' \leq \Delta$. Furthermore, we can not have the target degrees of v to
 743 be the same as the targets of one of its neighbours in B_c (these values are stored in T), as,
 744 otherwise, any valid target table T' of t would lead to adjacent vertices in B_t having the
 745 same degree. Let $\{t_1, \dots, t_k\} \subset \{D(v), \dots, \Delta\}$ be an enumeration of all the valid targets
 746 for v (i.e. $t_i \neq T(u)$ for all $u \in N[v] \cap B_c \setminus X$). Then, for each quadruple (X, D, T, min)
 747 in c , and for each $i = 1, \dots, k$, we need to create the quadruple (X, D', T_i, min) , such that
 748 $T_i(u) = T(u)$ for all $u \in B_c$ and $T_i(v) = t_i$. In total, we have $k \leq \Delta$ such quadruples.

749 Now, let us explain how we deal with the Join nodes. Assume that t is a Join Node with
 750 c_1 and c_2 as its two children in the tree decomposition. Here, it is important to mention that
 751 $B_{c_1} = B_{c_2}$ and $(B_{c_1}^\downarrow \setminus B_{c_1}) \cap (B_{c_2}^\downarrow \setminus B_{c_2}) = \emptyset$. Assume that there exists an irregulator S of
 752 $B_t^\downarrow \setminus B_t$ in G , a valid target table T of S , and let (X, D, T, min) be the quadruple we need to
 753 store in t for this pair (S, T) . Observe that this pair (S, T) is valid for both c_1 and c_2 , so we
 754 must already have stored at least one quadruple in each node. Let $X \subseteq B_t$ and a target table
 755 T such that (X, D_1, T, min_1) and (X, D_2, T, min_2) are stored for c_1 and c_2 respectively. We
 756 create the quadruple (X, D, T, min) for t by setting $D(u) = D_1(u) + D_2(u) - d_{G[B_t \setminus X]}(u)$ for
 757 all $u \in B_t \setminus X$, $D(u) = \emptyset$ for all $u \in X$ and $min = min_1 + min_2 - |X|$. Observe that these are
 758 the correct values for the $D(u)$ and min , as otherwise we would count $d_{G[B_t \setminus X]}(u)$ and $|X|$
 759 twice. Finally, we need to note that we do not store any quadruple (X, D, T, min) we create
 760 for the Join Note such that $D(u) > T(u)$ for a vertex $u \in B_t \setminus X$. This is because for such
 761 quadruples, the degree of vertex u will never be equal to any of the target degrees we have
 762 set, as it can only increase when we consider any of the ancestor (i.e. parent, grandparent
 763 etc.) nodes of t .

764 Finally, we need to treat the Forget nodes. Let t be a Forget node, c be the its child and
 765 v be the forgotten vertex. Assume that we have to store in t a quadruple (X, D, T, min) .
 766 Then, since $X = B_t \cap S$ for an irregulator S of B_t in G , we know that in c we must have
 767 already stored a quadruple (X', D', T', min') such that, $X' = S \cap B_c$, $D'(u) = D(u)$ for all
 768 $u \in B_c$, $T'(u) = T(u)$ for all $u \in B_c$ and $min' = min$. Therefore, starting from the stored
 769 quadruples in c , we can create all the quadruples of t . For each quadruple (X', D', T', min')
 770 in c , we create at most one quadruple (X, D, T, min) for t by considering two cases; the
 771 forgotten vertex v_f belongs to X' or not.

772 *Case 1. (v belongs to X'):* then the quadruple (X, D, T, min) is almost the same as
 773 (X', D', T', min') , with the following differences: $X = X' \setminus \{v\}$, $min = min'$, $D(u) = D'(u)$
 774 and $T(u) = T'(u)$ for all $u \in B_t$ and the tables D and T do not include any information for
 775 v as this vertex does not belong to B_t anymore.

776 *Case 2. (v does not belong to X'):* we will first check if $D'(v_f) = T'(v_f)$ or not. This
 777 is important because the degree of the v will never again be considered by our algorithm,
 778 and thus its degree will remain unchanged. So, if $D'(v_f) = T'(v_f)$, we create the quadruple
 779 (X, D, T, min) where $X = X'$, $min = min'$, $D(u) = D'(u)$ and $T(u) = T'(u)$ for all $u \in B_t$
 780 and the tables D and T do not include any information for v .

781 For the running time, observe that the number of nodes of a nice tree decomposition is
 782 $\mathcal{O}(tw \cdot n)$ and all the other calculations are polynomial in $n + m$. Thus we only need to count
 783 the different quadruples in each node. Now, for each vertex v , we either include it in X or
 784 we have $\Delta + 1$ options for the value $D(u)$ and $\Delta + 1 - i$ for the value $T(u)$ if $D(u) = i$. Also,

23:20 Complexity of Finding Maximum Locally Irregular Induced Subgraphs

785 for sufficiently large Δ , we have that $1 + \sum_{i=0}^{\Delta} (\Delta + 1 - i) < \Delta^2$. Furthermore, the set X
786 and the value min do not increase the number of quadruples because $X = \{u \mid D(u) = \emptyset\}$
787 and from all quadruples (X, D_1, T_1, min_1) , (X, D_2, T_2, min_2) such that $D_1(u) = D_2(u)$ and
788 $T_1(u) = T_2(u)$ for all $u \in B_t$, we only keep one of them (by Claim 22).

789 In total, the number of different quadruples in each node is Δ^{2tw} , and therefore the
790 algorithm decides in $\Delta^{2tw} n^{\mathcal{O}(1)}$ time. ◀