



**HAL**  
open science

## Local certification of graph decompositions and applications to minor-free classes

Nicolas Bousquet, Laurent Feuilloley, Théo Pierron

► **To cite this version:**

Nicolas Bousquet, Laurent Feuilloley, Théo Pierron. Local certification of graph decompositions and applications to minor-free classes. 25th International Conference on Principles of Distributed Systems, OPODIS 2021, Dec 2021, Starsbourg, France. pp.22:1–22:17, 10.4230/LIPIcs.OPODIS.2021.22 . hal-03772974

**HAL Id: hal-03772974**

**<https://hal.science/hal-03772974>**

Submitted on 8 Sep 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.

# Local certification of graph decompositions and applications to minor-free classes

Nicolas Bousquet 

Univ Lyon, CNRS, INSA Lyon, UCBL, LIRIS, UMR5205, F-69622 Villeurbanne, France  
nicolas.bousquet@univ-lyon1.fr

Laurent Feuilloley 

Univ Lyon, CNRS, INSA Lyon, UCBL, LIRIS, UMR5205, F-69622 Villeurbanne, France  
laurent.feuilleley@univ-lyon1.fr

Théo Pierron 

Univ Lyon, UCBL, CNRS, INSA Lyon, LIRIS, UMR5205, F-69622 Villeurbanne, France  
theo.pierron@univ-lyon1.fr

---

## Abstract

Local certification consists in assigning labels to the nodes of a network to certify that some given property is satisfied, in such a way that the labels can be checked locally. In the last few years, certification of graph classes received a considerable attention. The goal is to certify that a graph  $G$  belongs to a given graph class  $\mathcal{G}$ . Such certifications with labels of size  $O(\log n)$  (where  $n$  is the size of the network) exist for trees, planar graphs and graphs embedded on surfaces. Feuilloley et al. ask if this can be extended to any class of graphs defined by a finite set of forbidden minors.

In this work, we develop new decomposition tools for graph certification, and apply them to show that for every small enough minor  $H$ ,  $H$ -minor-free graphs can indeed be certified with labels of size  $O(\log n)$ . We also show matching lower bounds using a new proof technique.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Design and analysis of algorithms  $\rightarrow$  Distributed algorithms

**Keywords and phrases** Local certification, proof-labeling schemes, locally checkable proofs, graph decompositions, minor-free graphs

**Funding** This work was supported by ANR project GrR (ANR-18-CE40-0032).

## 1 Introduction

Local certification is an active field of research in the theory of distributed computing. On a high level it consists in certifying global properties in such a way that the verification can be done locally. More precisely, for a given property, a local certification consists of a labeling (called a *certificate assignment*), and of a local verification algorithm. If the configuration of the network is correct, then there should exist a labeling of the nodes that is accepted by the verification algorithm, whereas if the configuration is incorrect no labeling should make the verification algorithm accept.

Local certification originates from self-stabilization, and was first concerned with certifying that a solution to an algorithmic problem is correct. However, it is also important to understand how to certify properties of the network itself, that is, to find locally checkable proofs that the network belongs to some graph class. There are several reasons for that. First, because certifying some solutions can be hard in general graphs, while they become simpler on more restricted classes. To make use of this fact, it is important to be able to certify that the network does belong to the restricted class. Second, because some distributed algorithms work only on some specific graph classes, and we need a way to ensure that the network does belong to the class, before running the algorithm. Third, the distinction between certifying solutions and network properties is rather weak, in the sense that the techniques are basically the same. So we should take advantage of the fact that a lot is known about graph classes to learn more about certification.

45 In the domain of graph classes certification, there have been several results on various classes  
 46 such as trees [33], bipartite graphs [29] or graphs of bounded diameter [7], but until two years ago  
 47 little was known about essential classes, such as planar graphs. Recently, it has been shown that  
 48 planar graphs and graphs of bounded genus can be certified with  $O(\log n)$ -bit labels [15, 21, 22].  
 49 This size,  $O(\log n)$ , is the gold standard of certification, in the sense that little can be achieved with  
 50  $o(\log n)$  bits, thus  $O(\log n)$  is often the best we can hope for.

51 Planar and bounded-genus graphs are classic examples of graphs classes defined by forbidden  
 52 minors, a type of characterization that has become essential in graph theory since the Graph minor  
 53 series of Robertson and Seymour [39]. Remember that a graph  $H$  is a minor of a graph  $G$ , is  
 54 it possible to obtain  $H$  from  $G$  by deleting vertices, deleting edges, contracting edges. At this  
 55 point, the natural research direction is to try to get the big picture of graph classes certification,  
 56 by understanding all classes defined by forbidden minors. In particular, we want to answer the  
 57 following concrete question.

58 ► **Question 1** ([18, 21]). *Can any graph class defined by a finite set of forbidden minors be certified*  
 59 *with  $O(\log n)$ -bit certificates?*

60 This open question is quite challenging: there are as many good reasons to believe that the  
 61 answer is positive as negative.

62 First, the literature provides some reasons to believe that the conjecture is true. Properties  
 63 that are known to be hard to certify, that is, that are known to require large certificates, are very  
 64 different from minor-freeness. Specifically, all these properties (*e.g.* small diameter [7], non-3-  
 65 colorability [29], having a non-trivial automorphism [29]) are non-hereditary. That is, removing a  
 66 node or an edge may yield a graph that is not in the class. Intuitively, hereditary properties might be  
 67 easier to certify in the sense that one does not need to encode information about every single edge  
 68 or node, as the class is stable by removal of edges and nodes. Minor-freeness is a typical example of  
 69 hereditary property. Moreover, this property, that has been intensively studied in the last decades,  
 70 is known to carry a lot of structure, which is an argument in favor of the existence of a compact  
 71 certification (that is a certification with  $O(\log n)$ -bit labels).

72 On the other hand, from a graph theory perspective, it might be surprising that a general  
 73 compact certification existed for minor-free graphs. Indeed, for the known results, obtaining a  
 74 compact certification is tightly linked to the existence of a precise constructive characterization  
 75 of the class (*e.g.* a planar embedding for planar graphs [15, 22], or a canonical path to the root for  
 76 trees [33]). Intuitively, this is because forbidden minor characterizations are about structures that  
 77 are absent from the graphs, and local certification is often about certifying the existence of some  
 78 structures. While such a characterization is known for some restricted minor-closed classes, we  
 79 are far from having such a characterization for every minor-closed class. Note that there are a lot  
 80 of combinatorial and algorithmic results on  $H$ -minor free graphs, but they actually follow from  
 81 properties satisfied by  $H$ -minor free graphs, not from exact characterizations of such graphs. For  
 82 certification, we need to rule out the graphs that do not belong to the class, hence a characterization  
 83 is somehow necessary.

## 84 1.1 Our results

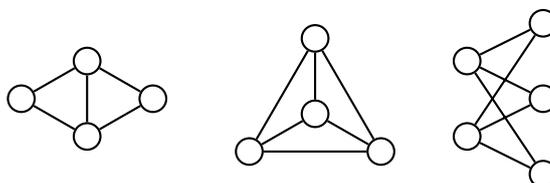
85 Answering Question 1 seems unfortunately out of reach, at the current state of our knowledge. We  
 86 have explained above about why designing compact certification is hard for classes that do not have  
 87 a constructive characterization. We will later give some intuition about why lower bounds seem  
 88 equally difficult to get. In this paper, we intend to build the foundations needed to tackle Question 1.  
 89 More precisely, we have four types of contributions.

90 First, we show how to certify some graph decompositions. Such decompositions state how to  
 91 build a class based on a few elementary graphs and a few simple operations. They are essential in  
 92 structural graph theory, and more specifically in the study of minor-closed classes. Amongst the  
 93 most famous examples of these theorems is the proof of the 4-Color Theorem [2] or the Strong  
 94 Perfect Graph Theorem [10].

95 Second, we show that by directly applying these tools, we can design compact certification for  
 96 several  $H$ -minor free classes, for which a precise characterization is known. See Fig. 1 and 2. That  
 97 is, we answer positively Question 1, for several small minors, and show that our decomposition  
 98 tools can easily be used.

Class	Optimal size	Result
$K_3$ -minor free	$\Theta(\log n)$	Equivalent to acyclicity [29, 33].
Diamond-minor-free	$\Theta(\log n)$	Corollary 29
$K_4$ -minor-free	$\Theta(\log n)$	Corollary 29
$K_{2,3}$ -minor-free	$\Theta(\log n)$	Corollary 29
$(K_{2,3}, K_4)$ -minor-free (i.e. outerplanar)	$\Theta(\log n)$	Corollary 29
$K_{2,4}$ -minor-free	$\Theta(\log n)$	Lemma 36

■ **Figure 1** Our main results for the certification of minor-closed classes.



■ **Figure 2** From left to right: the diamond, the clique on 4 vertices  $K_4$ , and the complete bipartite graph  $K_{2,3}$ .

99 Third, we do a systematic study of small minors to identify which is the first one that we cannot  
 100 tackle. We first prove the following theorem.

101 ► **Theorem 2.**  $H$ -minor-free classes can be certified in  $O(\log n)$  bits when  $H$  has at most 4 vertices.

102 Then, we extend this theorem to minors on five vertices with a specific shape, proving along  
 103 the way new purely graph-theoretic characterizations for the associated classes. After this study,  
 104 we can conclude that the next challenge is to understand  $K_5$ -minor free graphs.

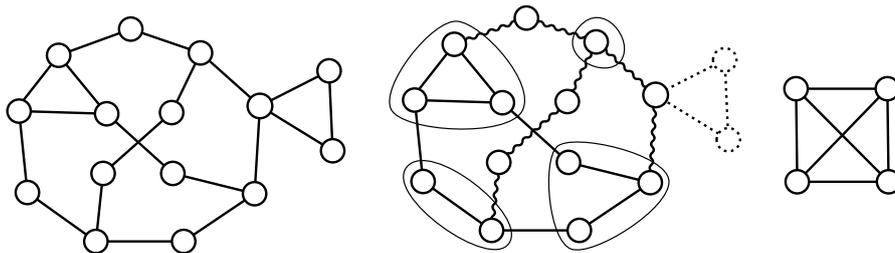
105 Finally, we prove a general  $\Omega(\log n)$  lower bounds for  $H$ -minor-freeness for all 2-connected  
 106 graphs  $H$ . This generalizes and simplifies the lower bounds of [22] which apply only to  $K_k$  and  
 107  $K_{p,q}$ -minor-free graphs, and use ad-hoc and more complicated techniques.

108 At the end of the paper, we discuss why the current tools we have, both in terms of upper and  
 109 lower bounds, do not allow settling Question 1. We list a few key questions that we need to answer  
 110 before we can fully understand the certification of minor-closed classes, from the certification of  
 111 classes with no tree minors to the certification  $k$ -connectivity, for arbitrary  $k$ .

112 **1.2 Our techniques**

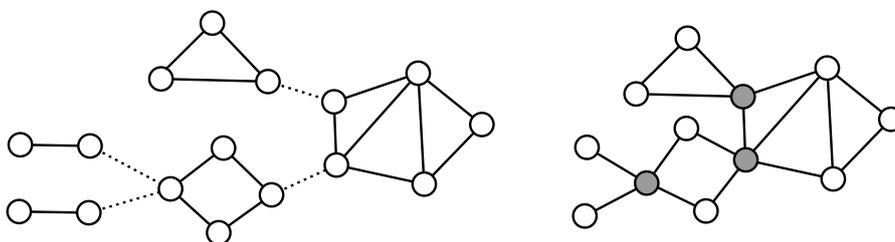
113 **General approach and challenges**

114 To give some intuition about our techniques, let us focus on a concrete example:  $K_4$ -minor-free  
 115 graphs. Remember that a graph has  $K_4$ -minor if we can get a  $K_4$  by deleting vertices and edges,  
 116 and contracting edges. An alternative definition is that a graph has a  $K_4$ -minor, if it is possible  
 117 to find four disjoint sets of vertices, called *bags*, such that: each bag is connected, there is a path  
 118 between each pair of bags, these paths and bags are all vertex-disjoint (except for the endpoints of  
 119 the paths that coincide with vertices of the bags). See Figure 3.



■ **Figure 3** The graph on the left has a  $K_4$  minor. Indeed, the bags of the second definition are depicted in the picture in the middle, and it is easy to find the six disjoint paths that link them. Alternatively, one can get a  $K_4$  like the one of the right-most picture by contracting all the edges inside the bags, contracting the wavy paths between bags into edges, and deleting the dotted vertices and edges.

120 An important observation is that, if we take a collection  $F_1, \dots, F_k$  of  $K_4$ -minor-free graphs, and  
 121 organize them into a tree, by identifying pairs of vertices like in Figure 4, we get a  $K_4$ -minor-free  
 122 graph.



■ **Figure 4** The five graphs with plain edges on the left picture are  $K_4$ -minor free. Organizing them into a tree by identifying the nodes linked by dotted edges makes a larger  $K_4$ -minor-free graph.

123 To see that, suppose that the graph we created has a  $K_4$ -minor. Then there exist bags and paths  
 124 as described above. If the bags and paths are all contained in the same former  $F_i$ , then this  $F_i$  would  
 125 not be  $K_4$ -minor-free, which is a contradiction. If it is not the case, then the bags and paths use  
 126 vertices that belong to different subgraphs  $F_i$  and  $F_j$ . And because of connectivity, they should use  
 127 a vertex  $v$  that connects two such subgraphs (grey vertices in Figure 4). Then the bags and paths  
 128 cannot be vertex-disjoint as required, because at least two of them should use the vertex  $v$ .

129 As a consequence of the observation above, a classic way to study  $K_4$ -minor-free graphs (as well  
 130 as other classes) is to decompose the graph into maximal 2-connected components organized into a  
 131 tree. This is called the *block-cut tree* of the graph, where every maximal 2-connected component  
 132 is called a *block*. (Figure 4 actually show the block-cut structure of the right-most graph.) This is  
 133 relevant here because 2-connected  $K_4$ -minor-free graphs have a specific structure; we will come  
 134 back to this later.

135 Now, from the certification point of view, there is a natural strategy: first certify the structure of  
 136 the block-cut tree, and then certify the special structure of each block. There are several challenges  
 137 to face with this approach. First, to certify the block-cut tree, it is essential to be able to certify the  
 138 connectivity of the blocks. Second, we need to avoid what we call certificate congestion, which is  
 139 the issue of having too large certificates because we use too many layers of certification on some  
 140 nodes. We now detail these two aspects, starting with the latter.

### 141 Avoiding certificate congestion

142 In the block-cut tree of a graph, the blocks are attached to each other by shared vertices, the *cut*  
 143 *vertices*. There is no bound on the number of blocks that are attached to a given cut vertex, and  
 144 this is problematic for certification. Indeed, we cannot give to every node the list of the blocks  
 145 it belongs to, as we aim for  $O(\log n)$  certificates, and such a list could contain  $\Omega(n)$  blocks. And  
 146 even if we could fix the certification of the block-cut tree, the same problem would appear with the  
 147 certification of the specific structure of each block: the cut vertices would have to hold a piece of  
 148 certification for each block.

149 We basically have two tools to deal with this problem. The first one is not new, it is a degeneracy  
 150 argument that already appeared in [21, 22]. A graph is  $k$ -degenerate if in every subgraph there exists  
 151 a vertex that has degree at most  $k$ . Intuitively (and a bit incorrectly), this means that when we need  
 152 to put a large certificate on a vertex, we can spread it on its some of its neighbors that have lower  
 153 degree. A more precise statement is that, for  $k$ -degenerate graphs, we can transform a certification  
 154 with  $O(f(n))$  labels *on the edges of the graphs*, into a classic certification with  $O(k \cdot f(n))$  labels on  
 155 the vertices. This is relevant for our problem, as a priori there is less congestion on the edges, and  
 156 minor-free classes have bounded degeneracy. Unfortunately, this is not enough for our purpose.  
 157 We then build a second, more versatile tool. It consists in proving that it is possible to transform  
 158 in mechanical way any certification of a graph or subgraph, into a certification that would put an  
 159 empty certificate on some given vertex. Once we have this tool, we can adapt the certification of  
 160 the blocks to work well in the block-cut tree: build the block-cut tree by adding blocks iteratively,  
 161 making sure that the connecting node has an empty label in the certification of the newly added  
 162 block.

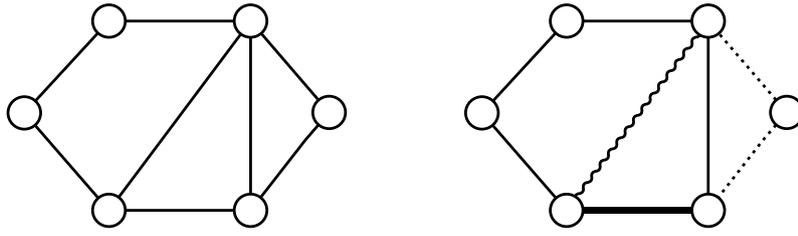
163 See Section 3 for the details on this topic.

### 164 Certifying connectivity properties

165 Connectivity properties have been studied before in distributed certification. Specifically, certifying  
 166 that for two given vertices  $s$  and  $t$ , the  $st$ -connectivity is at least  $k$  has been studied in [33] and [29].  
 167 But here we are interested in the connectivity of the graph itself, or in other words, in the  $st$ -  
 168 connectivity between any pair of vertices. Clearly, proving  $st$ -connectivity for any pair using the  
 169 schemes of the literature would lead to huge certificates. Instead, we use the characterizations of  
 170  $k$ -connected graphs that are known for small values of  $k$ . There are various such characterizations,  
 171 but they are all based on the same idea of *ear decomposition*.

172 To explain ear decompositions, consider a graph that we can build the following way (see  
 173 Figure 5). Start from an edge, and iteratively apply the following process: take two different nodes  
 174 of the current graph and link them by a path whose internal nodes are new nodes of the graph. It is  
 175 not hard to see that such a graph is always 2-connected. Remarkably, the converse is also true: any  
 176 2-connected graph can be built (or decomposed this way). This is called an open ear decomposition,  
 177 and similar constructions characterize 2-edge connected graphs and 3-vertex-connected graphs.

178 The good thing about these constructions is that we can certify them, by describing and certifying  
 179 every step. This requires some care, as when certifying a new path, we could increase the size of



■ **Figure 5** Illustration of an open ear decomposition. The graph on the left can be built with the ear decomposition described on the right. First, put the bold edge. Then add the path of plain edges. Finally, add the dotted path, and the wavy path, which is just one edge.

180 the certificates of the endpoints, that are already in the graph. Fortunately, the tools developed to  
 181 avoid certificate congestions allow us to control the certificate size.

182 The details about the connectivity certification can be found in Section 5.

### 183 Putting things together

184 Combining these techniques, we can prove the following theorem.

185 ► **Theorem 3.** *For any 2-connected graph  $H$ , if the 2-connected  $H$ -minor-free graphs can be certified  
 186 with  $f(n)$  bits, then the  $H$ -minor-free graphs can be certified with  $O(f(n) + \log n)$  bits.*

187 Going back to our example,  $K_4$ -minor-free graphs, given Theorem 3, we are left with certifying  
 188 the 2-connected  $K_4$ -minor-free graphs. As said above, these have a specific shape. More precisely,  
 189 2-connected  $K_4$ -minor-free graphs have a nested ear decomposition, which is yet another type of  
 190 ear decomposition, this time with additional constraints related to outerplanarity. We can certify  
 191 this structure by adapting a construction from [22] for outerplanar graphs.

192 More generally the 2-connected graphs corresponding to most of the classes of Figure 1 have  
 193 specific shapes that we can certify quite easily, which imply our compact certification schemes.  
 194 We do this in Section 6. A special case is  $K_{2,4}$ , that has a more complicated structure, requiring to  
 195 consider 3-connected components, and some more complicated substructures. We study this case in  
 196 Section 7.

197 Finally, in Section 8, we study all the minors on at most 4 vertices, and in Section 9 all the minors  
 198 on 5 vertices of some simple form. For these, we do not need new techniques on the certification  
 199 side, but we need to work on the graph theory side to establish new characterizations, as for these  
 200 minors the literature does not help. The work we do in Section 9 might be of independent interest  
 201 as we study the natural notion of  $H$ -minimal graph, which are the graph that have  $H$  as a minor,  
 202 but for which any vertex deletion would remove this property.

### 203 Lower bounds

204 Towards the end of the paper, we show that  $\Omega(\log n)$ -bit labels are necessary to certify (2-connected)  
 205 minor-free graph classes. When it comes to  $\Omega(\log n)$  lower bounds in our model, there are basically  
 206 two complementary techniques (called *cut-and-plug techniques* in [18]). Both techniques basically  
 207 show that paths cannot be differentiated from cycles, if the certificates use  $o(\log n)$  bits. First,  
 208 in [29], the idea is to use many correct path instances, and to prove that we can plug them into  
 209 an incorrect cycle instance, thanks to a combinatorial result from extremal graph theory. Second,  
 210 in [20], the idea is to consider a path, to cut it into small pieces, and to show via Sterling formula,  
 211 that there exists a shuffle of these pieces that can be closed into a cycle.

212 Previous lower bounds for minor-free graphs in [22] followed the same kind of strategies as [29]  
 213 and [20], with the same type of counting arguments, more complicated constructions, and tackled  
 214 only minors that were cliques or bicliques.

215 In this paper, we are able to do a black-box reduction between the path/cycle problem and the  
 216  $H$ -minor-freeness for any 2-connected  $H$ . This way we avoid explicit counting arguments, and get  
 217 a more general result with a simpler proof.

### 218 1.3 Related work

219 Local certification first appeared under the name of *proof-labeling schemes* in [33], inspired by works  
 220 on self-stabilizing algorithms (see [11] for a book on self-stabilization). It has then been generalized  
 221 under the name of *locally checkable proofs* in [29], and the field has been very active since these  
 222 seminal papers. In the following, we will focus on the papers about local certification of graph  
 223 classes, but we refer to [18] and [19] for an introduction and a survey of local certification in general.

224 As said earlier, certification was first mostly about checking that the solution to an algorithmic  
 225 problem was correct, a typical example being the verification of a spanning tree [33]. Some graph  
 226 properties have also been studied, for example symmetry in [29], or bounded diameter in [7]. Very  
 227 recently, classes that are more central in graph theory have attracted attention. It was first proved  
 228 in [37], as an application of a more general method, that planar graphs can be certified with  $O(\log n)$   
 229 bits in the more general model of distributed interactive proofs. Then it was proved in [22] that these  
 230 graphs can actually be certified with  $O(\log n)$  bits in the classic model, that is, without interaction.  
 231 This result was extended to bounded-genus graphs in [21]. Later, [15] provided a simpler proof of  
 232 both results via different techniques. It was also proved in [32, 36] that cographs, distance-hereditary  
 233 graphs, and some intersection graphs have compact distributed interactive proofs.

234 After the publication of the first version of this paper, some progress has been done on Question 1.  
 235 On the one hand, a positive answer has been given for an approximate version of the question.  
 236 More precisely, by allowing mistakes for graphs that are close to being  $H$ -minor-free (in the spirit  
 237 of property testing) one can define a compact certification [16] (follows from Theorem 6). (An  
 238 approximate certification for bounded degree planar graphs with constant size labels had been  
 239 established before, in [12].) On the other hand, two papers have established meta-theorems that  
 240 answer the question for specific minor shapes. More precisely, by proving that monadic second  
 241 order properties can be certified with  $O(\log n)$  when the treedepth is bounded [23], and  $O(\log^2 n)$   
 242 bits when treewidth is bounded [26], these papers prove as corollaries that the same sizes suffice for  
 243 path minors and planar minors, respectively.

244 Still in distributed computing, but outside local certification, the networks with some forbidden  
 245 structures have attracted a lot of attention recently. A popular topic is the distributed detection  
 246 of some subgraph  $H$ , which consists, in the CONGEST (or CONGEST-CLIQUE) model to decide  
 247 whether the graph contains  $H$  as a subgraph or not (see [6] and the references therein). A related  
 248 task is  $H$ -freeness testing, which is the similar but easier task consisting in deciding whether the  
 249 graph is  $H$ -free or far from being  $H$ -free (in terms of the number of edges to modify to get a  $H$ -free  
 250 graph). This line of work was formalized by [5] after the seminal work of [4] (see [25] and the  
 251 references therein). To our knowledge, no detection/testing algorithm or lower bounds have been  
 252 designed for  $H$ -minor-freeness.

253 Finally, we have mentioned in the introduction that certifying that the graph belongs to some  
 254 given class is important because some algorithms are specially designed to work on some specific  
 255 classes. For example, there is a large and growing literature on approximation algorithms for *e.g.*  
 256 planar, bounded-genus, minor-free graphs. We refer to [17] for a bibliography of this area. There  
 257 are also interesting works for exact problems in the CONGEST model, *e.g.* in planar graphs [27],  
 258 graphs of bounded treewidth or genus [30] and minor-free graphs [31]. In particular the authors

of [31] justify the focus on minor-free graphs by the fact that this class allows for significantly better results than general graphs, while being large enough to capture many interesting networks. Very recently, [28] proved general tight results on low-congestion short-cuts (an essential tool for algorithms in the CONGEST model) for graphs excluding a dense minor.

## 2 Preliminaries

In this section, we define formally the notions we use and describe some useful known certification building blocks.

### 2.1 Graphs and minors

Let  $G = (V, E)$  be a graph. Let  $X \subseteq V$ . The *subgraph of  $G$  induced by  $X$*  is the graph with vertex set  $X$  and edge set  $E \cap X^2$ . The graph  $G \setminus X$  is the subgraph of  $G$  induced by  $V \setminus X$ . A graph  $G'$  is a *subgraph of  $G$*  if  $V' \subseteq V$  and  $E' \subseteq E$ . For every  $v \in V$ ,  $N(v)$  denotes the *neighborhood of  $v$*  that is the set of vertices adjacent to  $v$ . The graph  $G$  is  *$d$ -degenerate* if there exists an ordering  $v_1, \dots, v_n$  of the vertices such that, for every  $i$ ,  $N(v_i) \cap \{v_{i+1}, \dots, v_n\}$  has size at most  $d$ . It refines the notion of maximum degree since any graph of maximum degree  $\Delta$  are indeed  $\Delta$ -degenerate (but the gap between  $\Delta$  and the degeneracy can be arbitrarily large). Let  $u, v \in V$ , a *path* from  $u$  to  $v$  is a sequence of vertices  $v_0 = u, v_1, \dots, v_\ell = v$  such that for every  $i \leq \ell - 1$ ,  $v_i v_{i+1}$  is an edge. It is a *cycle* if  $v_\ell v_0$  also exists.

A graph  $G$  is *connected* if there exists a path from  $u$  to  $v$  for every pair  $u, v \in V$ . All along the paper, we only consider connected graphs. Indeed, in certification, the nodes can only communicate with their neighbors, so no node can communicate with nodes of another connected component.

A vertex  $v$  is a *cut-vertex* if  $G \setminus \{v\}$  is not connected. If  $G$  does not contain any cut-vertex,  $G$  is *2-(vertex)-connected*. If the removal of any edge does not disconnect the graph, we say that  $G$  is *2-edge-connected*. A graph is  *$k$ -(vertex)-connected* if there does not exist any set  $X$  of size  $k - 1$  such that  $G \setminus X$  is not connected. To avoid cumbersome notations, we will simply write  *$k$ -connected* for  *$k$ -vertex-connected*.

A graph  $H$  is a *minor of  $G$*  if  $H$  can be obtained from  $G$  by deleting vertices, deleting edges and contracting edges. Equivalently, it means that, if  $G$  is connected, there exists a partition of  $V$  into connected sets  $V_1, \dots, V_{|H|}$  such that there is (at least) an edge between  $V_i$  and  $V_j$  if  $h_i h_j$  is an edge of  $H$ . We say that  $V_1, \dots, V_{|H|}$  is a *model of  $H$* . The graph  $G$  is  *$H$ -minor-free* if it does not contain  $H$  as a minor.

### 2.2 Local computation and certification

We assume that the graph is equipped with unique identifiers in polynomial range  $[1, n^k]$ , thus these identifiers can be encoded on  $O(\log n)$  bits.

Local certification is a mechanism for verifying properties of labeled or unlabeled graphs. In this paper we will use a local certification at distance 1, which is basically the model called *proof-labeling schemes* [33]. A convenient way to describe a local certification is with a prover and a verifier. The *prover* is an external entity that assigns to every node  $v$  a certificate  $c(v)$ . The *verifier* is a distributed algorithm, in which every node  $v$  acts as follows:  $v$  collects the identifiers and the certificates of its neighbor and itself, and outputs a decision *accept* or *reject*. A local certification certifies a graph class  $\mathcal{C}$  if the following two conditions are verified:

1. For every graph of  $\mathcal{C}$ , the prover can find a certificate assignment such that the verifier accepts, that is, all nodes output *accept*.

2. For every graph not in  $\mathcal{C}$ , there is no certificate assignment that makes the verifier accept, that is for every assignment, there is at least one node that rejects.

The size of the certificate of  $\mathcal{C}$  is the largest size of a certificate assigned to a node of a graph of  $\mathcal{C}$ .

Note that to describe a local certification, the only essential part is the verifier algorithm, the prover is just a way to facilitate the description of a scheme.

In this paper, we are going to use a variant of the model above, called *edge certification*, where the certificates can be assigned on both the nodes and the edges. See Subsection 3.1.

### 2.3 Known building blocks for graph certification

There are few known certification schemes that we are going to use intensively as building blocks in the paper.

► **Lemma 4** ([1, 33]). *Acyclicity can be certified in  $O(\log n)$  bits.*

The classic way to certify that the graph is acyclic, is for the prover to choose a root node, and then to give to every node as its certificate its distance to the root. The nodes can simply check that the distances are consistent.

The same idea can be used to certify a *spanning tree* of the graph, encoded locally at each node by the pointer to its parent, which is simply the ID of this parent. The scheme is the same, except that the prover, in addition to the distances, gives the ID of the root, and the verification algorithm checks that all nodes have been given the same root-ID, and only takes into account the edges that correspond to pointers (also the root checks that its ID is the root-ID). A spanning tree is a very useful tool to broadcast the *existence of a vertex satisfying a locally checkable property*: simply choose a spanning tree rooted at the special vertex, encode it locally with pointers and certify it. Then the root can check that indeed it has the right property, and all the other vertices know that such a vertex exists.

Finally, with the same ideas, one can easily deduce  $O(\log n)$  certification for paths. We just add to the acyclicity scheme the verification that the degree of every node is at most 2. Note that cycles do not need certificates to be verified: every node just checks that it has degree exactly 2.

Let us now define a graph class that will appear in several decompositions.

► **Definition 5.** *A path-outerplanar graph is a graph that admits a path  $P$  that can be drawn on a horizontal line, such that all the edges that do not belong to  $P$  can be drawn above that line without crossings. The edges are said to be nested.*

We are going to use the following result as a black box.

► **Lemma 6** ([22]). *Path-outerplanar graphs can be certified with  $O(\log n)$ -bit certificates.*

The following classic result will also be useful at some point of the paper.

► **Lemma 7** ([33]). *Every graph class can be certified with  $O(n^2)$  bits.*

The idea of the scheme is that the prover gives to every node  $v$  the map of the graph, e.g. as an adjacency matrix, along with the position of  $v$  in this map. Then every node can check that it has been given the same map as its neighbors, and that the map is consistent with its neighborhood in the network.

## 3 Avoiding certificate congestion

One can obtain many structured graph classes like minor free graphs with "gluing" operations, for instance, by identifying vertices of two graphs of the class. If we have a certification for both

342 graphs, we would like to simply take both certificate assignments to certify the new graph. However,  
 343 for the vertex on which the two graphs are glued, the size of the certificate might have doubled.  
 344 While it is not a problem for bounded degree graphs, it can become problematic if many gluing  
 345 operations occur around the same vertex, since this vertex would get an additional certificate from  
 346 each operation. In this section, we present two ways to tackle these issues, that will be used in the  
 347 forthcoming sections.

348 The first one consists in shifting the certification on edges instead of vertices, which helps in  
 349 the sense that when gluing on vertices the edge certificate can remain unchanged. As we will see,  
 350 the edge setting is equivalent to the usual vertex certification for nice enough classes. The second  
 351 option uses that one can (almost) freely assume that a given vertex has an empty label in a correct  
 352 certification.

### 353 3.1 Edge certification and degeneracy

354 Transforming a node certification into an edge certification can always be done without additional  
 355 asymptotic costs: just copy on every edge the certificate of the two endpoints, and adapt the  
 356 verification algorithm accordingly. Transforming an edge certification into a node certification is  
 357 also always possible, by giving a copy of the edge label to each of its endpoint. But this transformation  
 358 can drastically increase the certificate size: if an edge certification uses  $\Omega(f(n))$ -bit labels, the  
 359 associated node certification might use  $\Omega(n \cdot f(n))$ -bit if the maximum degree of the graph is  
 360 linear. The following theorem ensures that in degenerate graph classes there is a more efficient  
 361 transformation that permits to drastically reduce the size of the certificate.

362 ► **Theorem 8** ([21]). *Consider an edge certification of a graph class  $\mathcal{C}$  where the edges are labeled*  
 363 *with  $f(n)$ -bit certificates. If  $\mathcal{C}$  is  $d$ -degenerate, then there exists a (node) certification with  $d \cdot f(n)$ -bit*  
 364 *certificates.*

365 Note that  $H$ -minor free graphs have degeneracy  $O(h\sqrt{\log h})$  where  $h = |V(H)|$  [34, 41].  
 366 Therefore, we can freely put labels on edges when certifying classes defined by forbidden minors.

### 367 3.2 Certification with one empty label

368 In this part, our goal is to erase the certificate of a node. To this end, we first consider certification  
 369 of spanning trees and strengthen both Lemma 4 and the discussion that followed in Subsection 2.3.  
 370 We then extend this intermediate step to every graph class in Lemma 10.

371 ► **Lemma 9.** *Let  $T$  be a spanning tree of  $G$ . There exists a certification of  $T$  that does not assign a*  
 372 *label to the root, and uses the same certificate as the classic tree certification (cf. Subsection 2.3) on the*  
 373 *other nodes.*

374 **Proof.** On yes-instances, the prover assigns the labels as in the classic scheme, and removes the  
 375 label of the root. Then the verification proceeds like in the classic scheme except for a node that  
 376 has no label or a node that has a neighbor with no label. If two adjacent nodes have been given an  
 377 empty label, then they reject. If a node with no label sees that two of its neighbors have been given  
 378 different root-ID, then it rejects. Otherwise, every node simulates the computation where the node  
 379 with empty label has been given distance 0, and the same root-ID as its neighbors. Because of the  
 380 previous checks, the labels used in the simulation are consistent, and on correct instance are the  
 381 same as the one used in the classic certification. Thus, the correctness follows from the correctness  
 382 of the classic scheme. ◀

383 A *pointed graph* is a graph with one selected node. Given a class, one can build its pointed  
 384 version by taking for each graph all the pointed versions of it.

385 ► **Lemma 10.** *Consider a class  $\mathcal{C}$  that can be certified with certificates of size  $f(n)$ . One can certify*  
 386 *the pointed class of  $\mathcal{C}$  with  $O(f(n) + \log n)$  bits, without having to put certificates on the selected node.*

387 **Proof.** First, to certify that exactly one node is pointed, we can simply find a spanning tree rooted  
 388 on the pointed vertex and assign to each node the spanning tree certification of Lemma 9 which  
 389 uses  $O(\log n)$  bits. For the rest of the certification, on a *yes*-instance, the prover first assigns the  
 390 certificates following the original certification. Then it removes the certificate of the selected node  
 391 and appends copies of it to the certificates of its neighbors.

392 Every node  $v$  runs the following verification. If  $v$  is not the selected node, nor one of its neighbors,  
 393 then it does the same verification as before. If  $v$  is the selected node, it checks that its neighbors  
 394 have been given the same label as "label of the selected node", and then takes this label as its own,  
 395 and runs the previous verification algorithm. If  $v$  is a neighbor of the selected node, it runs the same  
 396 verification algorithm as before, but simulating that the selected node has been given the certificate  
 397 that was appended to its own certificate.

398 All nodes are simulating the computation in the graph where the selected node would have  
 399 been given its certificates, thus the correctness of this new certification follows from the correctness  
 400 of the original certification. ◀

401 Observe that the previous results can be easily iterated: one can always remove the labels  
 402 of  $k$  nodes (as long as they are pairwise non-adjacent) to the cost of a factor  $k$  in the size of the  
 403 certificates. Therefore, the result extends to the case of  $k$ -independent pointed classes (i.e. where an  
 404 independent set of size at most  $k$  is selected instead of only one vertex).

405 ► **Corollary 11.** *Consider a class that can be certified with certificates of size  $f(n)$ . One can certify*  
 406 *the  $k$ -independent pointed class with  $O(kf(n) + k \log n)$  bits, without having to put certificates on*  
 407 *the selected nodes.*

408 Moreover, with more constraints on the structure of the set of pointed vertices (for instance if  
 409 they are all at distance at least 3), one could even obtain certificate of size  $O(f(n) + k \log n)$  (since  
 410 every node receives the certificate of at most one selected node).

## 411 **4 Compositions of certifications**

412 In this section, we show how to combine certification algorithms for several classes to certify larger  
 413 ones, and we illustrate this idea on two constructions. The first one considers classes defined by the  
 414 existence of some subgraph: we settle the intuition stating that it is often easier to test the existence  
 415 of a structure rather than its absence, since we can pinpoint which nodes/edges lie in the structure.

416 The second construction mimics a natural operation on graphs, consisting in replacing some  
 417 vertex/edge by another graph. This operation occurs quite often in the literature: many classes,  
 418 especially the ones defined by forbidden minors, get a characterization using this operation.

419 Some results of this section will not be used to certify minor-free classes later in the paper. They  
 420 are proved here for completeness.

### 421 **4.1 Subgraphs**

422 ► **Proposition 12.** *Let  $\mathcal{C}$  be a graph class that can be certified with  $f(n)$ -bit labels. Let  $\mathcal{C}'$  be the class*  
 423 *of the graphs that contain a graph of  $\mathcal{C}$  as subgraph. Then  $\mathcal{C}'$  can be certified with certificates of size*  
 424  *$O(f(n) + \log n)$  on the nodes and  $O(1)$  on the edges.*

425 **Proof.** On a *yes*-instance  $G$ , the prover assigns the certificates on nodes and edges in the following  
 426 way. First, it chooses a subgraph  $H$  that belongs to  $\mathcal{C}$  and assigns the certificates that certify that

427  $H$  is in  $\mathcal{C}$ , as if the rest of the graph did not exist. This takes at most  $f(n)$  bits. Second to every  
 428 node and edge that belongs to  $H$ , the prover assigns a special label. Third, the prover describes and  
 429 certifies a spanning tree pointing to a node that has the special label.

430 The verification algorithm is the following. The nodes that have the special label, run the  
 431 verification algorithm for  $\mathcal{C}$ , taking into account only the nodes and edges that have the special  
 432 label. The nodes also check the spanning tree structure, and the root of the tree checks that it does  
 433 have the special label.

434 Because of the spanning tree, there must exist a node with the special label, thus there are nodes  
 435 that run the verification algorithm for  $\mathcal{C}$ , and if they succeed it means that a graph of  $\mathcal{C}$  appears as a  
 436 subgraph in the graph  $G$ . ◀

437 The edge certificates in Proposition 12 can be inconvenient if we want a classic certification  
 438 (without edge certificates) and if the graph is not assumed to be degenerate, which prevents us from  
 439 using Theorem 8. However, observe that we give non-empty certificates only to the edges of the  
 440 subgraph, hence we can obtain a vertex-certification when the class  $\mathcal{C}$  is degenerate.

441 ► **Corollary 13.** *Let  $\mathcal{C}$  be a  $d$ -degenerate graph class that can be certified with  $f(n)$ -bit labels. Let  
 442  $\mathcal{C}'$  be the class of the graphs that contain a graph of  $\mathcal{C}$  as subgraph. Then  $\mathcal{C}'$  can be certified with  
 443 certificates of size  $O(f(n) + d \log n)$  on the nodes.*

444 Observe also that when considering induced subgraphs, we only have to specify which vertices  
 445 are special, hence we do not need edge certificates either. Note that, since we do not need to label  
 446 edges, we do not need the class  $\mathcal{C}$  to be degenerate.

447 ► **Corollary 14.** *Let  $\mathcal{C}$  be a graph class that can be certified with  $f(n)$ -bit labels. Let  $\mathcal{C}'$  be the class of  
 448 the graphs that contain a graph of  $\mathcal{C}$  as an induced subgraph. Then  $\mathcal{C}'$  can be certified with certificates  
 449 of size  $O(f(n) + \log n)$  on the nodes.*

## 450 4.2 Expansions

451 Two common operations in characterizations of graph classes are what we call node and edge  
 452 expansions.

453 ► **Definition 15.** *Consider two graph classes  $\mathcal{C}_1$  and  $\mathcal{C}_2$ .*

454 ■ *The node expansion of  $\mathcal{C}_1$  by  $\mathcal{C}_2$  is the class of graphs obtained by the following operation. Take  
 455 a graph  $G$  in  $\mathcal{C}_1$  and replace every node  $v$  by a graph  $H(v)$  in  $\mathcal{C}_2$ , in such a way that for every  
 456 edge  $uv \in E(G)$ , there is (at least) one edge between  $H(u)$  and  $H(v)$  in  $G$  (and no such edge if  
 457  $uv \notin E(G)$ ).*

458 ■ *The edge expansion of  $\mathcal{C}_1$  by  $\mathcal{C}_2$  is the class of graphs obtained by the following operation. Take  
 459 a graph  $G$  in  $\mathcal{C}_1$  and replace every edge  $uv$  by a graph  $H(u, v)$  from  $\mathcal{C}_2$ , in such a way that the  
 460 nodes of the original graph that are contained in  $H(u, v)$  are exactly  $u$  and  $v$ .*

461 We would like to have results of the form: if  $\mathcal{C}_1$  and  $\mathcal{C}_2$  can be certified with  $f(n)$  and  $g(n)$ -bit  
 462 labels respectively, then the expansion can be certified with  $O(f(n) + g(n))$ -bit labels. While  
 463 the natural approach (almost) works for edge-expansion, it does not give such a result for node-  
 464 expansion. However, we can actually make it work with a bound that takes into account the  
 465 maximum degree of the expanded graph.

466 ► **Proposition 16.** *Consider two graph classes  $\mathcal{C}_1$  and  $\mathcal{C}_2$  that can be certified with  $f(n)$ -bit and  $g(n)$ -  
 467 bit labels respectively, where all the graphs of  $\mathcal{C}_1$  have maximum degree  $\Delta$ . Then the node-expansion  
 468 of  $\mathcal{C}_1$  by  $\mathcal{C}_2$  can be certified with  $O(\Delta \cdot f(n) + g(n) + \Delta \log n)$ -bit certificates.*

469 **Proof.** Consider a graph  $G \in \mathcal{C}_1$  on  $\ell$  nodes  $v_1, \dots, v_\ell$  of maximum degree  $\Delta$ , and let  $H_1, \dots, H_\ell$   
 470 be graphs of  $\mathcal{C}_2$ . We consider the node expansion of  $G$  where every  $v_i$  is replaced by  $H_i$ .

471 On a *yes*-instance, the prover assigns the certificates the following way. First it assigns to every  
 472 node the index  $i$  corresponding to the graph  $H_i$  it belongs to and the certification of the fact that  
 473  $H_i$  belongs to  $\mathcal{C}_2$  (without taking into accounts the other nodes and edges). This takes at most  
 474  $g(n) + \log n$  bits per node. Second, the prover gives to each vertex of  $H_i$  the original certificate of  
 475  $v_i$  that  $G$  belongs to  $\mathcal{C}_1$  as well as the original certificate of all the vertices in  $N(v_i)$  in  $G$  together  
 476 with their names, which takes  $O(\Delta f(n))$  bits. Finally, for every  $v_j \in N(v_i)$ , the prover chooses a  
 477 vertex  $w_j$  in  $H_i$  adjacent to a vertex in  $H_j$ , and certifies a spanning tree of  $H_i$  rooted at  $w_j$ . This  
 478 takes  $O(\Delta \log n)$  bits.

479 The verification algorithm is the following. Every node (labeled as) in  $H_i$  checks that the number  
 480 of trees corresponds to the degree of  $v_i$  in  $G$ . Every node checks the correctness of the different trees.  
 481 Moreover, every root  $v$  of a spanning tree in  $H_i$  checks that it has a neighbor in the corresponding  
 482  $H_j$ . All the nodes of  $H_i$  check that their neighbors are in  $H_i$  or in some  $H_j$  with  $v_j$  incident to  $v_i$   
 483 in  $G$ . Every node of each  $H_i$  runs the verification algorithm to check that  $H_i$  does belong to  $\mathcal{C}_2$ .  
 484 Finally, every node of  $H_i$  simulates the verification of the original node  $v_i$ , which is possible since  
 485 every vertex of  $H_i$  receives the certificate of  $v_i$  and all its neighbors in  $G$ . And every vertex  $w_i \in H_i$   
 486 incident to  $w_j \in H_j$  checks that  $v_j \in N_G(v_i)$  and that the certificate of  $w_j$  indeed contains the  
 487 certificates of  $v_i$  and  $v_j$  given for  $G$ . ◀

488 ▶ **Proposition 17.** *Consider two graph classes  $\mathcal{C}_1$  and  $\mathcal{C}_2$  that can be certified with  $f(n)$ -bit and  $g(n)$ -  
 489 bit labels respectively. Then the edge-expansion of  $\mathcal{C}_1$  by  $\mathcal{C}_2$  can be certified with  $O(f(n) + g(n) + \log n)$ -  
 490 bit certificates on the edges.*

491 **Proof.** We use a similar reasoning as for the proof of Proposition 16, except that we first transform  
 492 the node certifications of  $\mathcal{C}_1$  and  $\mathcal{C}_2$  into edge certifications (by putting the label of a node on all the  
 493 edges incident with it).

494 Consider a graph  $G \in \mathcal{C}_1$ . We consider the edge expansion of  $G$  where every  $uv$  is replaced by  
 495  $H(u, v)$ . Each edge  $e$  from  $H(u, v)$  receives the labels of  $u$  and  $v$ , the certificate of  $uv$  in  $G$  for  $\mathcal{C}_1$ ,  
 496 and the certificate of  $e$  in  $H(u, v)$  for  $\mathcal{C}_2$ . Therefore, the certificates have size  $O(f(n) + g(n) + \log n)$ .

497 Now each vertex can check that all the edges labeled in some  $H(u, v)$  share the same certificate  
 498 for  $uv$ . There are two kinds of nodes: some where all incident edges are labeled as in the same  
 499  $H(u, v)$ , and the others (the original vertices of  $G$ ). All of them run the verification algorithm for  $\mathcal{C}_2$   
 500 by considering each group of incident edges labeled as in the same  $H(u, v)$ . The latter also recover  
 501 the certificates of their neighbors in  $G$  from the edge labeling, and run the verification algorithm  
 502 for  $\mathcal{C}_1$ . ◀

503 Before giving deeper applications of these results in future sections, let us prove that the *existence*  
 504 of a minor in the graph is easy to certify. This was already mentioned in previous papers without  
 505 formal proofs [21, 22]. We prove it here to show a simple application of our techniques, and we  
 506 think it is a meaningful illustration of the fact that certifying that a structure is present or absent  
 507 are two very different tasks in our model.

508 ▶ **Corollary 18.** *Given a graph  $H$ , one can certify that a graph has  $H$  as a minor in  $O(\log n)$  bits.*

509 **Proof.** As we already observed, a graph  $G$  has  $H$  as minor if and only if  $V(G)$  can be partitioned  
 510 into  $|H|$  connected sets such that there is an edge between  $V_i$  and  $V_j$  when the corresponding  
 511 vertices in  $H$  are connected. Free to delete edges, we can assume that each  $V_i$  is actually a spanning  
 512 tree and there is a unique edge from  $V_i$  to  $V_j$  if and only if the corresponding vertices are connected  
 513 in  $H$ . In other words,  $G$  has a subgraph that is a node expansion of  $H$  by trees. Moreover, we can  
 514 choose such a subgraph with degree at most  $|H| - 1 = O(1)$  since  $H$  is fixed.

515 Let us start from a certification of  $H$  and build a certification of  $G$ . The structure of  $H$  can be  
 516 certified in a brute-force way, by providing to every node the complete map of the graph which  
 517 takes constant space (since  $H$  is fixed). Then, since trees can be certified in  $O(\log n)$  bits, thanks to  
 518 Proposition 16, any node-expansion of  $H$  by trees can be certified with  $O(\log n)$  certificates.

519 We finally get a node certification with certificates of size  $O(\log n)$  using Corollary 13. ◀

## 520 **5 Connectivity and connectivity decompositions**

521 In this section, we explain how to certify connectivity properties and connectivity decompositions,  
 522 in particular the block-cut tree mentioned in the introduction.

523 An *ear decomposition* is a way to build a graph by iteratively adding paths, the so-called *ears*.  
 524 Ear decompositions are central tools for decades in structural graph theory and are used in many  
 525 decomposition or algorithmic results. There exists various variants of this process, that characterize  
 526 different classes and properties. For certification, these decompositions happen to be easier to  
 527 manipulate than some other types of characterizations since they are based on iterative construction  
 528 of the graph, and use paths, which are easy to certify. These paths are convenient since we can  
 529 "propagate" some quantity of information on them as long as every vertex belongs to a bounded  
 530 number of paths. In this section, we remind several such decompositions, and use them to certify  
 531 various connectivity properties and decompositions.

### 532 **5.1 Connectivity properties**

533 Let us start with 2-connectivity. A graph  $G$  has an *open ear decomposition* if  $G$  can be built, by  
 534 starting from a single edge, and iteratively applying the following process: take two different nodes  
 535 of the current graph and link them by a path whose internal nodes are new nodes of the graph  
 536 (such a path is called an *ear*). Note that this path can be a single edge, and then there is no new  
 537 node. Let an *inner node* of an ear be a vertex that is created with this ear, and let a *long ear* be an ear  
 538 with at least one inner node.

539 ▶ **Theorem 19** ([43] (reformulated)). *A graph is 2-connected if and only if it has an open ear*  
 540 *decomposition.*

541 We use this characterization to certify 2-connectivity.

542 ▶ **Lemma 20.** *2-connected graphs can be certified with  $O(\log n)$  bits.*

543 **Proof.** First observe that one can obtain a long-ear decomposition from an ear decomposition (and  
 544 vice versa) by removing/adding short ears, i.e. edges. Therefore, having an open ear decomposition  
 545 is equivalent to having a subgraph with an open long-ear decomposition. Note that if a graph  $G$   
 546 has an open long-ear decomposition, then it is 2-degenerate. Indeed, the vertices of the last added  
 547 long-ear have degree two and their removal is still a graph with an open long-ear decomposition.  
 548 So in order to get the conclusion, Corollary 13 ensures that we only have to certify open long-ear  
 549 decomposition with  $O(\log n)$  bits per vertex.

550 The certification works as follows. First the prover gives to every node the identifiers of the very  
 551 first edge, and describes and certifies a spanning tree pointing to one of the endpoints of this edge.  
 552 The nodes of this edge are given an *index* 0. Second, the prover gives to every node the information  
 553 related to the step when it has been added, and only about this step. That is, the prover gives the  
 554 *index* of the addition (that is the number of the ear in which the vertex is created), along with two  
 555 oriented paths spanning the path and pointing to the two extremities of the ear. By Corollary 11,  
 556 these paths can be certified without certificates on the extremities of the paths.

557 Every node checks the correctness of the spanning tree pointing to the first edge, and the fact  
 558 that only these nodes have index 0. Then, every node also checks that the spanning paths it has  
 559 been given are correct, that is: (1) the distances and root-ID are consistent (2) all nodes have the  
 560 same index, and that (3) the declared endpoints are different. Also, the two nodes that are adjacent  
 561 to the endpoints of the paths check that the endpoints have a smaller index than their own.

562 Let us now prove the correctness of the certification. Because of the spanning tree, the original  
 563 edge exists, is unique, and is the only set of nodes with index 0. Because of the certified paths  
 564 spanning the ears, one can also recover the path structure and the fact that a path is added after its  
 565 endpoints. Note that in an instance where all nodes accept, there might be two different paths with  
 566 the same index  $i$ , but this is not a problem: the only important feature is the precedence order. ◀

567 With similar construction we can certify the edge connectivity instead of the vertex connectivity.

568 ▶ **Corollary 21.** *2-edge-connected graphs can be certified with  $O(\log n)$  bits.*

569 **Proof.** Robbins proved in [38] that a graph  $G$  is 2-edge connected if and only if  $G$  has an ear  
 570 decomposition. An ear decomposition is the same as an open ear decomposition, except that it  
 571 starts from a cycle and that the two endpoints of an ear do not need to be different. The proof above  
 572 can thus be adapted to this class.

573 The only difference is that vertices with index 0 form a cycle (which can be certified). Then  
 574 during the verification procedure we simply do not have to check that the extremities of the path of  
 575 the ear decomposition are distinct, in other words we do not have to check (3). ◀

576 A more refined type of ear decomposition characterizes the 3-vertex-connected graphs.

577 ▶ **Definition 22** ([8, 35, 40]). *Let  $ru$  and  $rt$  be two edges of a graph  $G$ . A Mondshein sequence  
 578 through  $rt$ , avoiding  $u$  is an open ear decomposition of  $G$  such that:*

- 579 1.  *$rt$  is in the first ear.*
- 580 2. *the ear that creates node  $u$  is the last long ear,  $u$  is its only inner vertex, and it does not contain  $ru$ .*
- 581 3. *the ear decomposition is non-separating, that is, for every long ear except the last one, every inner  
 582 node has a neighbor that is created in a later ear.*

583 ▶ **Theorem 23** ([8, 40]). *Let  $ru$  and  $rt$  be two edges of a graph  $G$ . The graph  $G$  is 3-vertex-connected  
 584 if and only if it has a Mondshein sequence through  $rt$  avoiding  $u$ , and there are three internally  
 585 vertex-disjoint path between  $t$  and  $u$ .*

586 ▶ **Corollary 24.** *3-connected graphs can be certified with  $O(\log n)$  bits on vertices and  $O(1)$  bits on  
 587 edges.*

588 **Proof.** On yes-instances the prover chooses an arbitrary edge  $ru$  and certifies the ear decomposition  
 589 as in Lemma 20. The prover also adds a spanning tree pointing to the edges  $ru$  and  $rt$ , and gives to  
 590 every vertex the index of the last long ear created. These new pieces of information allow the nodes  
 591 to check that the ear decomposition is a Mondshein sequence. The prover also encode the three  
 592 vertex disjoint paths, by pointer on the nodes of these paths, and number them 1, 2 and 3, to allow  
 593 the nodes to check disjointness. ◀

## 594 5.2 Block-cut tree

595 Now that we can certify connectivity properties, we introduce a way to certify decomposition of  
 596 graphs into parts of higher connectivity. Let us start with a few definitions.

597 A *2-connected component* of a graph  $G$  is a connected subgraph  $H$  maximal by inclusion such  
 598 that the removal of one node does not disconnect  $H$ . Observe that a 2-connected component can  
 599 consist of just one edge in the case of a bridge, i.e. an edge whose removal disconnects the graph.

600 The intersection of any pair  $C, C'$  of 2-connected components has size at most one. Indeed, if  
 601 it had size at least two, then we could merge these into a larger 2-connected component, which  
 602 would contradict the maximality. So we can define an auxiliary graph from  $G$  where every node  
 603 corresponds to a 2-connected component and there is an edge between two components if and only  
 604 if they intersect on exactly one node. This graph is a tree, because a cycle would again create a  
 605 larger 2-connected component, contradicting maximality. This tree is called the *block-cut tree*.

606 Let  $T$  be a block-cut tree of  $G$ , and  $D$  a maximal 2-connected component chosen to be the root  
 607 of this tree. (Note that if  $G$  is 2-connected then the graph is reduced to this component). Let  $C$  be  
 608 a component that is not the root of the tree. The *connecting node* of a component  $C$  is the node  
 609 lying both in  $C$  and in its parent component. The *interior* of  $C$  is the set of nodes of  $C$  minus the  
 610 connecting node of  $C$ . Note that the interior of a component is always non-empty.

611 This section is devoted to proving the following result and apply it for certification:

612 ► **Theorem 3.** *For any 2-connected graph  $H$ , if the 2-connected  $H$ -minor-free graphs can be certified*  
 613 *with  $f(n)$  bits, then the  $H$ -minor-free graphs can be certified with  $O(f(n) + \log n)$  bits.*

614 **Proof of Theorem 3.** Since  $H$  is 2-connected, a graph  $G$  is  $H$ -minor-free if and only if each of  
 615 its 2-connected components is. (This is basically the observation we made at the beginning of  
 616 Subsection 1.2.) This is the property we certify. On a *yes*-instance, the prover will assign the  
 617 certificates the following way. It first computes the block-cut tree and root it on some node  $C$ . It  
 618 then does the following:

- 619 1. For each 2-connected component, the prover chooses a node from the interior of the component  
 620 to be the *leader* of this component. Every node of the interior of a component  $C$  is given the  
 621 identifier of the leader of  $C$  as well as a spanning tree of  $C$  pointing towards it. Since the  
 622 component is 2-connected, the component minus the leader of the component is connected and  
 623 such a tree exists.
- 624 2. Every node is given a label stating whether it is a connecting node or not.
- 625 3. Every node is given the identifier of the connecting node of its component closest from the root  
 626 in the block-cut tree (called the *component* of the node), as well as a spanning tree pointing to it,  
 627 using the certification of Lemma 9 that uses an empty certificate on the root.
- 628 4. In order to check acyclicity of the block-cut tree, every node is given the distance of its component  
 629 to the root-component (in terms of number of components).
- 630 5. The prover certifies the 2-connectivity of each component using the certification of Lemma 20  
 631 and the fact that it is  $H$ -minor free using the certification with  $f(n)$  bits of the theorem. By  
 632 Lemma 10 this can be done by only assigning labels to the interior nodes of the component.

633 Before we move on to the verification and the correctness of the scheme, note that every node is  
 634 given a certificate of size  $O(f(n) + \log n)$ . Indeed, each piece of information we have given to the  
 635 node is of size  $O(\log n)$  or  $f(n)$ , and we have given a constant number of those to every node. In  
 636 particular, a connecting node in the interior of a component  $C$ , received only labels that are related  
 637  $C$ , and not labels related to other components it belongs to (since we consider pointed components).

638 Now, every node does the following verification. Every node checks that the spanning tree  
 639 pointing to the leader is correct. If this step succeeds, we have a partition of the nodes in components.  
 640 Every node also checks the correctness of the spanning tree pointing to the connecting node.

641 Every node  $v$  checks that, if it has an edge to a connecting node  $w$  with a different leader,  
 642 then  $w$  is the connecting node of its own component. Every connecting node  $v$  checks that it is  
 643 connected to a single node in its parent component and that it is the claimed neighbor in that  
 644 component. If this step succeeds, we have a decomposition into components linked by connecting  
 645 nodes. The consistency of the component distances are also checked by the nodes: this distance  
 646 should be decremented at each connecting node, and only there. This ensures the acyclicity of the

647 component structure. Finally, every node checks that the 2-connectivity and the  $H$ -minor-freeness  
 648 of its component. Globally this verification ensures that the graph is  $H$ -minor-free. ◀

## 649 **6 Application to $C_4$ , Diamond, $K_4$ and $K_{2,3}$ minor-free graphs**

650 This section is devoted to the certification of  $C_4$ -minor-free, diamond-minor-free graphs,  $K_4$ -minor-  
 651 free graphs and  $K_{2,3}$ -minor-free graphs. All the proofs will follow the same structure: prove that  
 652 the 2-connected components, which are more structured, can be certified with small labels, and  
 653 then use Theorem 3 to conclude for the general case.

654 Before going to this proof let us describe how to certify *series-parallel graphs*, which in addition  
 655 to be interesting network topologies [24], are closely related to  $K_4$ -minor-free graphs.

656 ▶ **Definition 25.** A (2-terminal) series-parallel graph is a graph with two labeled vertices called the  
 657 source and the sink that can be built recursively as follows. A single edge is a series-parallel graph  
 658 where one endpoint is the source and the other is the sink. Let  $G_1, G_2$  be two series-parallel graphs. The  
 659 series of  $G_1$  and  $G_2$  which consists in merging the sink of  $G_1$  and the source of  $G_2$  is a series-parallel  
 660 graph. The parallel of  $G_1$  and  $G_2$ , which consists in merging the sources of  $G_1$  and  $G_2$  together and  
 661 merging the sinks of  $G_1$  and  $G_2$  together, is a series-parallel graph.

662 A nested ear decomposition is an open ear decomposition that starts from a path, with two  
 663 properties: (1) both ends of an ear have to be connected to the same ear, and (2) for every ear, the  
 664 ears that are plugged onto it are nested. Eppstein proved the following in [14] about series-parallel  
 665 graphs.

666 ▶ **Theorem 26** ([14]). A 2-connected graph is series-parallel if and only if it has a nested ear  
 667 decomposition.

668 We will use this decomposition theorem for our certification.

669 ▶ **Theorem 27.** 2-connected series-parallel graphs can be certified with  $O(\log n)$ -bit labels.

670 **Proof.** The prover certifies the decomposition of Theorem 26. We have already described how to  
 671 certify an open ear decomposition in the proof of Lemma 20. We can easily adapt it so that it starts  
 672 from a path instead of an edge: there is a spanning tree pointing to one of the endpoints of the  
 673 paths, and the path itself is certified with distances, the usual way.

674 It is also easy to certify that each ear  $e$  has both of its endpoints on the same older ear  $e'$ : just  
 675 give to each vertex of  $e$  the identifiers of the endpoints of  $e$  and  $e'$ . The endpoints of an ear can  
 676 check the consistency of these announced identifiers with the identifiers of their paths. A more  
 677 tricky part is to certify that the ears are nested. Remember that Lemma 6 states that a path with  
 678 nested edges (a path-outerplanar graph) can be certified with  $O(\log n)$ -bit labels. This is exactly  
 679 what we need except that we would like to have nested paths instead of nested edges. But then we  
 680 can transfer the information from one endpoint of the paths to the other endpoint. ◀

681 ▶ **Lemma 28.** 2-connected  $C_5$ -minor free graphs are either graphs of size at most 4 or  $K_{2,p}$  or  $K'_{2,p}$   
 682 which is the complete bipartite graph  $K_{2,p}$  plus an edge between the two vertices on the set of size 2.

683 **Proof.** Since  $G$  is 2-connected, by Menger's theorem, for every pair  $x, y$  of vertices, there exist at  
 684 least two vertex disjoint  $xy$ -paths. Since  $G$  is  $C_5$ -minor free, these paths have size at most 2, in  
 685 particular  $x, y$  are at distance at most 2.

686 Let  $u, v$  be two non-adjacent vertices. Then the removal of  $N(u) \cap N(v)$  disconnects  $u$  from  $v$   
 687 since otherwise we can find two vertex disjoint  $uv$ -paths, one being of size at least 3, which provides  
 688 a  $C_5$ . In particular, it implies that  $|N(u) \cap N(v)| \geq 2$  since  $G$  is 2-connected.

689 Let  $x \in N(u) \setminus (N(v) \cup \{v\})$ . Since  $x, v$  are non-adjacent, there must be an edge between  $x$  and  
 690  $N(v)$ . But this creates a  $C_5$  since  $|N(u) \cap N(v)| \geq 2$ . Therefore non-adjacent vertices are twins.

691 Let  $I$  be a maximum independent set in  $G$ . Note that all the vertices of  $I$  are twins. Therefore,  
 692 by maximality, if  $u \notin I$  then  $u$  is complete to  $I$ . Now either vertices of  $I$  have degree at least 3, and  
 693  $G$  contains  $K_{3,3}$  hence a  $C_5$ -minor, or vertices of  $I$  have degree 2 and  $G$  is  $K_{2,p}$  or  $K'_{2,p}$ . ◀

694 We can now prove easily the claimed certifications.

695 ► **Corollary 29.** *The following classes of graphs can be certified with  $O(\log n)$  bit certificates:*  
 696  *$C_4$ -minor-free graphs,  $C_5$ -minor free graphs, diamond-minor-free graphs, house-minor free graphs<sup>1</sup>,*  
 697 *outerplanar graphs (that is  $(K_{2,3}, K_4)$ -minor-free graphs),  $K_{2,3}$ -minor-free and  $K_4$ -minor-free graphs.*

698 **Proof.** By Theorem 3, if we can certify the 2-connected graphs of these classes we obtain the  
 699 conclusion. So we simply have to prove that for each class we can certify the 2-connected graph of  
 700 the class.

- 701 ■ 2-connected  $C_4$ -minor-free graphs are  $K_2$  and  $K_3$  [9], which can be certified with  $O(1)$  bits.
- 702 ■ 2-connected  $C_5$ -minor-free graphs are either graphs of size at most 4 or a complete bipartite  
 703 graph  $K_{2,p}$  (with a potential edge between the two vertices in the set of size 2 by Lemma 28.  
 704 Since such graphs can be certified with  $O(\log n)$  bits, the conclusion follows.
- 705 ■ 2-connected diamond-minor-free graphs are induced cycles. Cycles can be certified with  $O(1)$   
 706 bits (see the discussion after Lemma 4).
- 707 ■ 2-connected house-minor-free graphs are either induced cycles or graphs of size at least four.  
 708 Indeed, assume that there is a cycle of length at least 5. Then it should be induced, since  
 709 otherwise it contains a house as a minor. Moreover, it should contain all the vertices of the  
 710 graph otherwise there is an ear starting from this cycle and the cycle plus the ear provides a  
 711 house. Since induced cycles can be easily certified with  $O(\log n)$  bits, the conclusion follows.
- 712 ■ 2-connected outerplanar graphs are exactly path-outerplanar graphs with an edge between the  
 713 first and the last node. Indeed, by 2-connectivity, the outer face must be a cycle, and removing  
 714 any edge from it yields a path-outerplanar graph. One can then certify the existence and  
 715 uniqueness of this edge using a spanning tree, and then certify that the rest of the graph is  
 716 path-outerplanar. This yields a  $O(\log n)$ -bit certification by Theorem 6.
- 717 ■ Let  $G$  be a 2-connected  $K_{2,3}$ -minor-free graph. If  $G$  does not contain  $K_4$ , then it is a 2-connected  
 718 outerplanar graph and the result follows from the previous item. Otherwise, if  $G$  is not restricted  
 719 to  $K_4$ , then it contains a fifth vertex  $u$ . Since  $G$  is connected, there is a shortest path from  $u$  to  
 720 the  $K_4$  ending at  $v$ . Since  $v$  is not a cut-vertex, there should be another path between  $u$  and the  
 721  $K_4$  avoiding  $v$ , but this creates a  $K_{2,3}$  minor. Therefore,  $G$  is  $K_4$ , which can be certified easily.
- 722 ■ The 2-connected  $K_4$ -minor-free graphs are exactly the 2-connected series-parallel graphs. Then  
 723 the results follow directly from Theorem 27. ◀

## 724 **7** Application to $K_{2,4}$ -minor free graphs

725 When the size of the minors are increasing (and for most of the decomposition theorems known in  
 726 structural graph theory), 2-connectivity is not enough. In this example we will illustrate how to use  
 727 the certificate of 3-connectivity to conclude.

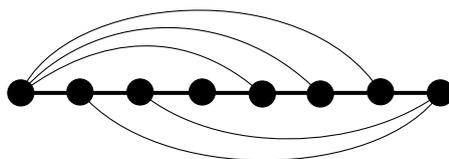
728 Let us illustrate it for this section on the characterization of  $K_{2,4}$ -minor-free graphs from [13].  
 729 It is more involved than the other characterizations we have seen so far. We will follow the  
 730 structure of [13], restricting first to 3-connected graphs, then to 2-connected graphs, and finally all  
 731  $K_{2,4}$ -minor-free graphs.

<sup>1</sup> The house being a  $C_4$  plus a vertex connected to two consecutive vertices of the  $C_4$ .

## 7.1 3-connected case

Let us start with the definition of a graph class. We use notations similar to [13] (Section 2.1). See Figure 6.

**Definition 30.** Let  $n, r, s$  be three integers, and  $p$  a Boolean. The graph  $G_{n,r,s,p}$  consists of a path  $v_1, \dots, v_n$ , the edges  $v_1v_{n-i}$  for  $1 \leq i \leq r$ , the edges  $v_nv_{1+j}$  for  $1 \leq j \leq s$ , and the edge  $v_1v_n$  if  $p = 1$ . For a function  $f(n, r, s, p)$  that associate a Boolean with each combination of parameters, let  $\mathcal{G}[f]$  be the set of graphs  $G_{n,r,s,p}$  such that  $f(n, r, s, p) = 1$ .



**Figure 6** Example for Definition 30. This graph is  $G_{8,3,2,0}$ : it has 8 nodes, edges  $v_1v_{n-i}$  for  $i \in \{1, 2, 3\}$ , edges  $v_nv_{1+j}$  for  $j \in \{1, 2\}$ , and it does not have the edge  $v_1v_n$ .

**Theorem 31** (Theorem 2.12 in [13] (adapted)). *There exists an  $f$  such that the set of 3-connected  $K_{2,4}$ -minor-free graphs is  $\mathcal{G}[f]$ , plus nine graphs on at most 8 vertices.*

In [13], the authors give an explicit description of  $f$  but we can avoid going into details here because of the following general lemma.

**Lemma 32.** *For all  $f$ ,  $\mathcal{G}[f]$  can be certified with  $O(\log n)$ -bit labels.*

**Proof.** On a *yes*-instance, the prover certifies the spanning paths with root  $v_1$ , with last node  $v_n$ , and writes in each certificate the values  $n, r, s$  and  $p$ . The nodes check the structure of the path and the fact that  $n, r, s$  and  $p$  are the same on all nodes. Second  $v_1$  checks the structure of its neighborhood, and in particular the values  $r$  and  $p$ . Similarly,  $v_n$  checks the structure of its neighborhood, and in particular the values  $s$ , and the fact that its distance to the root is indeed  $n$ . Finally, all nodes check that  $f(n, r, s, p) = 1$ . The correctness of the scheme is straightforward. ◀

This directly yields the following lemma.

**Lemma 33.** *3-connected  $K_{2,4}$ -minor-free graphs can be certified with  $O(\log n)$ -bit labels.*

**Proof.** Consider a *yes*-instance. By Theorem 31, either it is one of the nine small graphs of Theorem 31, and then we can use a constant size certification, or it is a graph of  $\mathcal{G}[f]$  for the specific  $f$  of Theorem 31, and then we can use Lemma 32. ◀

For the 2-connected case, one of the types of graphs that we want to certify is of the following form: a 3-connected graph, where a set of edges with a special property is expanded with another graph class. To be able to certify this, we will need the nodes to check that the set of edges that has been expanded has the special property. To capture the notion of special property, without going into the intricate details of what this property is exactly, let us define an *edge-set decider*. A function  $h$  is an edge-set decider if it takes as input a 3-connected  $K_{2,4}$ -minor-free graph whose edges are either unlabeled, or labeled with a special label, and outputs a Boolean.

762 ► **Lemma 34.** *Let  $h$  be an edge-set decider, such that for every graph  $G$ , there is at most  $O(n)$  different*  
 763 *sets of edges  $S$  such that  $h(G, S) = 1$ . The set of 3-connected  $K_{2,4}$ -free graphs  $G$  with labelled edges,*  
 764 *where  $h(G)$  is true, can be certified with  $O(\log n)$  bits.*

765 **Proof.** First, for every graph  $G$ , we fix an indexing of edge sets  $S$  such that  $h(G, S) = 1$ . The  
 766 prover first uses the same certificates as in Lemma 33 for the certification of unlabeled 3-connected  
 767  $K_{2,4}$ -minor-free graphs. Then it gives to all nodes the index of the set of labeled edges. Following  
 768 the certification of the proof of Theorem 31, every node knows in which graph it lives and what is  
 769 its position in that graph. Then, every node just checks that the labeled edges in its neighborhood  
 770 correspond to the index announced by the prover. The labels have size  $O(\log n)$  because of Lemma 33  
 771 and because there are at most  $O(n)$  different sets of edges  $S$  such that  $h(G, S) = 1$ . ◀

## 772 7.2 2-connected case

773 We now state the characterization theorem of the 2-connected case.

774 ► **Theorem 35** (Theorem 3.5 in [13]). *There exists a function  $h$  as in Lemma 34 such that the following*  
 775 *holds. A graph  $G$  is 2-connected  $K_{2,4}$ -minor-free graph if and only if one of the following holds:*

- 776 1.  $G$  is outerplanar.
- 777 2.  $G$  is the union of three path-outerplanar graphs  $H_1, H_2, H_3$  with the same path endpoints  $x$  and  
 778  $y$ , and possibly the edge  $(x, y)$ , where  $|V(H_i)| \geq 3$ , for each  $i$  and  $V(H_i) \cap V(H_j) = x, y$  for  
 779  $i \neq j$ .
- 780 3.  $G$  is obtained from a 3-connected  $K_{2,4}$ -minor-free graph  $G_0$  by choosing a subset  $S$  such that  
 781  $h(G_0, S) = 1$ , and replacing each edges  $(x_i, y_i)$  of  $S$  by a path-outerplanar graphs  $H_i$  with  
 782 endpoints  $(x_i, y_i)$ , where  $V(H_i) \cap V(G_0) = x_i, y_i$  for each  $i$ , and  $V(H_i) \cap V(H_j) \subset V(G_0)$  for  
 783  $i \neq j$ .

784 In [13],  $h$  is called the set of subdividable edges, and is fully characterized. Our proof works for  
 785 any  $h$ , as long as it satisfies the properties of Lemma 33, and it is the case for the  $h$  of [13].

786 ► **Lemma 36.** *2-connected  $K_{2,4}$ -minor-free graphs can be certified with  $O(\log n)$ -bit labels.*

787 **Proof.** We show that each of the three cases can be certified with  $O(\log n)$  bits.

- 788 1. Outerplanar graphs can be certified with  $O(\log n)$  bits (Corollary 29).
- 789 2. This case basically consists in an edge expansion of a multigraph with three edges between  
 790 two nodes by path outerplanar graphs. Note that the proof of Proposition 17 works here even  
 791 if the original graph is a multigraph. Proposition 17 gives us an  $O(\log n)$  edge certification  
 792 because path-outerplanar graphs can be certified with  $O(\log n)$  certificates, and the condition  
 793 on the number of nodes can also be certified with  $O(\log n)$  bits with a spanning tree counting  
 794 the number of nodes (see e.g. in [18]). This edge certification can be transferred to a node  
 795 certification with the same certificate size asymptotically because of Theorem 8, and because  
 796  $H$ -minor-free graphs have bounded degeneracy.
- 797 3. Again, this item basically corresponds to an edge-expansion: the edge expansion of a 3-connected  
 798  $K_{2,4}$ -minor-free graph by path-outerplanar graphs. We know by Lemma 33 and Lemma 6 that  
 799 both these classes can be certified on  $O(\log n)$  bits, so the vanilla edge-expansion can also be  
 800 certified with  $O(\log n)$  bits (using the degeneracy like in the previous item). The only issue left  
 801 is the fact that the only edges of  $G_0$  that are allowed to be expanded by something different  
 802 from an edge need to belong to an  $S$  such that  $h(G_0, S) = 1$ . But this is easy with Lemma 34:  
 803 the edges that have a path-outerplanar expansion are the one that are considered to have a  
 804 special label. ◀

## 8 Certifying $H$ -free graphs with $|H| \leq 4$

In previous sections, we have proven that certifying  $H$ -minor free graphs can be done with  $O(\log n)$  bits for some graphs  $H$ . The graphs we have treated in previous sections are somehow amongst the hardest graphs of small size. When the connectivity of the graph  $H$  increases, the class of  $H$ -minor free graph contains more and more graphs, and then is (morally speaking) harder to certify. Let us prove that the other graphs on 4 vertices (which have fewer edges, and then are less connected) can also be certified, with arguments either simpler than or similar to what has been done in previous sections, to establish the following theorem.

► **Theorem 2.**  *$H$ -minor-free classes can be certified in  $O(\log n)$  bits when  $H$  has at most 4 vertices.*

We consider two cases depending on whether  $H$  contains a cycle.

► **Lemma 37.** *If  $|H| \leq 4$ , and  $H$  contains a cycle, then  $H$ -free graphs can be certified with  $O(\log n)$  bits.*

**Proof.** Since  $H$  contains a cycle, either it is  $C_4$ , and the result follows from Corollary 29, or it contains a triangle. Let us distinguish the cases depending on how the fourth vertex is connected to the triangle. If it is connected to two or three vertices, then  $H$  is either  $K_4$ , or a diamond, and then  $H$ -minor-free graphs can be certified with  $O(\log n)$  bits by Corollary 29.

So we can assume that  $H$  is a triangle plus one vertex attached to at most one vertex of the triangle. If  $G$  contains a cycle, let  $C$  be a shortest cycle in  $G$ , that is a cycle that contains the minimum number of vertices. Then  $C$  must contain all the vertices of the graph. Indeed, otherwise, since  $G$  is connected, there exists a node  $v$  attached to  $C$ , and  $v \cup C$  contains  $H$  as a minor. Therefore,  $G$  is either a cycle or a tree, which can be both certified in  $O(\log n)$  bits, see Subsection 2.3. ◀

► **Lemma 38.** *If  $|H| \leq 4$  and  $H$  is acyclic, then we can certify  $H$ -free graphs with  $O(\log n)$  bits.*

**Proof.** If  $H$  has an isolated node then any graph  $G$  contain  $H$  has a minor as long as  $G$  contains a (non necessarily induced) path on three nodes and an isolated vertex. Since this property holds for every connected graph on 4 vertices, the conclusion follows.

So we can assume that  $H$  is connected. There are only two acyclic connected graphs on 4 vertices: the star  $S_{1,3}$  with 3 leaves, and the path  $P_4$  on four vertices. If  $G$  does not contain a star with 3 leaves as a minor, it means that  $G$  is either a path or a cycle which can be easily certified. If  $G$  does not contain a path on four nodes as a minor, it means that  $G$  is a star which, can be certified the following way. Give the identifier of the center to all nodes, and let the nodes check that they have been given the same ID, and that the non-center nodes have exactly one neighbor, and that this neighbor has this ID. ◀

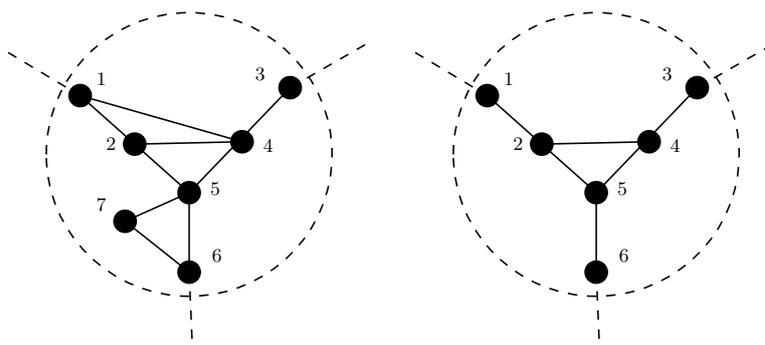
This completes the picture for graphs  $H$  on at most 4 vertices.

## 9 Graphs on at most 5 vertices

Let us now focus on graphs  $H$  with at most 5 vertices. We were not able to deal with all of them, the most problematic one being  $K_5$ , as we will discuss later on. However, we proved that  $H$ -minor freeness can be certified for some dense graphs like  $K_{2,3}$ . The goal of this section is to provide evidence that again, the hardest case will be the case where  $H$  is dense. Before entering into the details of the proof, let us study some necessary conditions on the graph to be minimally not  $H$ -minor-free.

## 9.1 $H$ -minimal graphs

A graph  $G$  is  $H$ -minimal if  $G$  admits a  $H$ -minor but, for any vertex  $v$ ,  $G \setminus v$  does not admit any  $H$ -minor. Consider such a model  $V_1, \dots, V_{|H|}$  of  $H$  in a  $H$ -minimal graph  $G$ . Intuitively, for all  $i$ , the important part of the subgraph induced by  $V_i$  is a spanning tree that makes it connected, and connected to the neighboring  $V_j$ 's. For example, if a  $V_i$  contains a node that is only connected to other nodes of  $V_i$ , and whose removal does not disconnect the subgraph of  $V_i$ , then this node is unessential. In other words, such a node would not appear in a  $H$ -minimal graph, because we could remove it, and still have a model of  $H$ . Nevertheless, it is not true that the subgraph induced by every  $V_i$  is a tree (see Figure 7).



■ **Figure 7** The two pictures represent some set  $V_i$  in a  $H$ -model. The dashed edges represent connections to other nodes of the model. In the first picture, the graph cannot be  $H$ -minimal, indeed we can remove the nodes 7 and 2, and still have a proper model. In the second picture, no node can be removed without disconnecting the subgraph induced by  $V_i$ .

We now describe what the  $V_i$ 's subgraphs precisely look like in a  $H$ -minimal graph. Let  $T$  be a graph, and  $S_1, \dots, S_r$  be some prescribed subsets of vertices of  $T$ . A *Steiner tree* of  $T$  with respect to the  $S_i$ 's is a tree in  $T$  containing at least one element of each  $S_i$ . We say that  $T$  is an *almost tree* for the  $S_i$ 's if any Steiner tree with respect to the  $S_i$ 's contains all the vertices of  $T$ . Now, given a model  $V_1, \dots, V_{|H|}$  of  $H$ , and  $v_i \in H$ , the prescribed sets we are going to consider for  $V_i$  are the subsets  $S_j \subseteq V_i$  containing all the vertices connected to  $V_j$ , for every  $j$  such that  $v_i v_j$  is an edge of  $H$ . A Steiner tree of  $V_i$  for the model  $V_1, \dots, V_{|H|}$  of  $H$  is a Steiner tree containing at least one vertex of each prescribed set. When the model is clear from context, we simply say a Steiner tree of  $V_i$ .

With these notions, let us describe some properties of  $H$ -minimal graphs:

► **Lemma 39.** *Let  $H$  be a graph and let  $G$  a  $H$ -minimal graph. For every  $H$ -model of  $G$ , each  $V_i$  is an almost tree.*

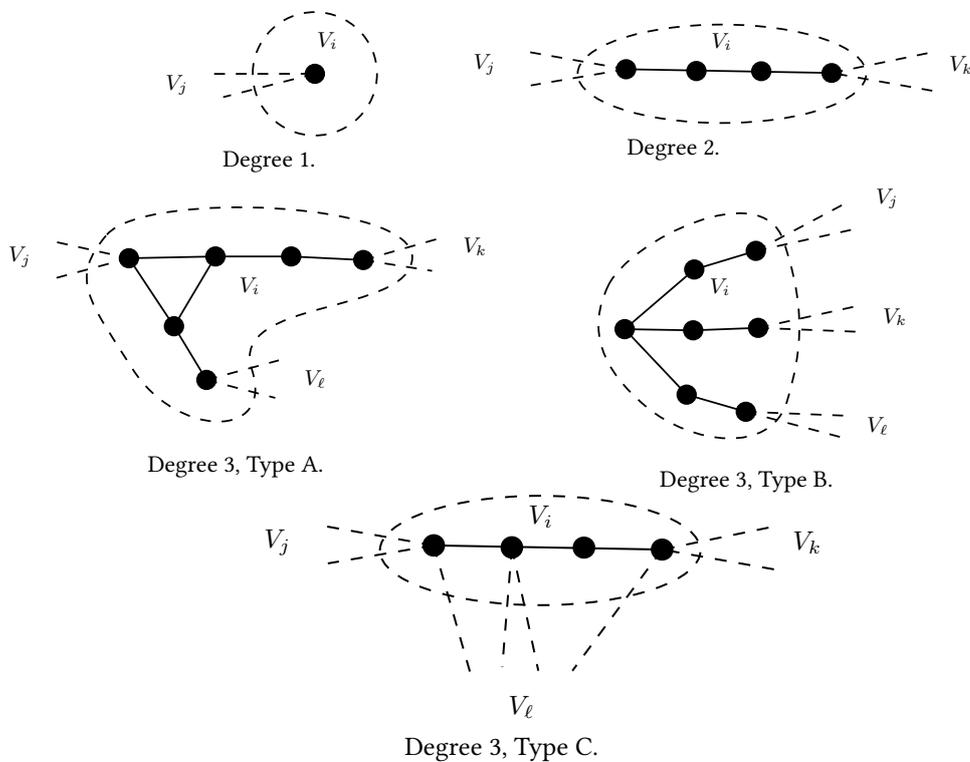
**Proof.** The proof is straightforward. If some  $V_i$  is not an almost tree, then we can select a subset  $V_i'$  of  $V_i$  which is an almost tree. When we consider the subsets where all the  $V_j$ 's are the same but  $V_i$  which is replaced  $V_i'$ , we still have a model of  $H$ , and it does not contain all the vertices, a contradiction with the fact that  $G$  is  $H$ -minimal. ◀

It follows that we can characterize the form of the  $V_i$ 's such that  $h_i$  has small degree in  $H$ .

► **Corollary 40.** *Let  $H$  be a graph, and  $G$  be a  $H$ -minimal graph. There exists a  $H$ -model of  $G$  such that:*

1. *If the degree of  $h_i$  in  $H$  is one, then  $V_i$  is reduced to a single vertex.*

- 874 2. If the degree of  $h_i$  in  $H$  is two, then  $V_i$  is reduced to a single path  $P$  and, if  $h_j, h_k$  are the two  
 875 neighbors of  $h_i$  then exactly one endpoint of  $P$  is connected to  $V_j$ , the other is connected to  $V_k$ , and  
 876 all the other vertices of  $P$  are neither connected to  $V_j$  nor  $V_k$ .
- 877 3. If the degree of  $h_i$  in  $H$  is three, then the subgraph induced by  $V_i$  is of one of the three following  
 878 types:
- 879 ■ Type A: the subgraph is a triangle with a path attached to each of the three corners (which might  
 880 be reduced to a single vertex) where the other endpoint of the path is attached to a  $V_j$ , and no  
 881 other vertex is attached to  $V_j$ .
  - 882 ■ Type B: the subgraph is an induced subdivided star where only the last vertex of each branch is  
 883 connected to a set  $V_j$ , and in that case it is connected to exactly one  $V_j$ .
  - 884 ■ Type C: the subgraph is a path, and there exists  $j, k$  such that the only connections with  $V_j$  and  
 885  $V_k$  are on the endpoints of the path. Any connection is possible for the vertices of the path with  
 886 the last set  $V_\ell$ .



■ **Figure 8** The types of the  $V_i$ 's in Corollary 40

- 887 **Proof.** 1. If the degree of  $h_i$  is one, then a Steiner tree only needs the node that is connected to  
 888 the rest of the model, so  $V_i$  has only one node.
- 889 2. If the degree is two, then any Steiner tree contains a path between a node in  $V_j$  and a node in  
 890  $V_k$ , and the shortest such paths is an induced path, thus the subgraph induced by  $V_i$  must be an  
 891 induced path, with only the endpoints connected to the rest of the graph.
- 892 3. For degree 3, there are several cases.
- 893 ■ If  $V_i$  contains a cycle, then by minimality the removal of any vertex on this cycle disconnects  
 894  $V_i$  from another branch. It follows that  $V_i$  has type A.

- 895     – If  $V_i$  does not have a cycle, it has at most three leaves. If it has exactly three leaves then it  
 896     has type B.  
 897     – Otherwise,  $V_i$  is a path, and by the degree-2 case, only the endpoints can connect to some  
 898     sets  $V_j$  and  $V_k$ , but the connections to the third set  $V_\ell$  are not controlled. This is type C.  
 899     ◀

## 900 9.2 $H$ with an isolated vertex and extension

901 ▶ **Theorem 41.** *Let  $H$  be a graph on 5 vertices containing an isolated vertex. We can certify  $H$ -free  
 902 graphs with certificates of size  $O(\log n)$ .*

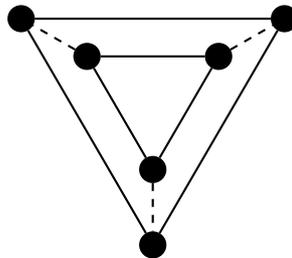
903     The rest of this section is devoted to the proof of Theorem 41.

904     Let  $H'$  be the graph  $H$  where an isolated vertex has been removed. A  $H$ -free graph is either a  
 905  $H'$ -free graph, or it is a graph  $G$  such that all the models of  $H'$  contain all the vertices of  $G$ . Since  
 906  $H'$ -minor free graphs can be certified within  $O(\log n)$  bits by Theorem 2, we can assume that  $G$  is  
 907  $H'$ -minimal.

908     The core of the proof consists in proving the following lemma.

909 ▶ **Lemma 42.** *For every graph  $H'$  on four vertices, any  $H'$ -minimal graph is in one of the following  
 910 categories:*

- 911     1. subdivided copies of  $H'$ ,  
 912     2. graphs of size 4,  
 913     3. induced cycles,  
 914     4. induced cycle plus a node,  
 915     5. the graphs of the type of Figure 9,  
 916     6. graphs with at most five vertices of degree larger than 2,  
 917     7. the complete bipartite graph  $K_{3,3}$ .



918 ■ **Figure 9** This drawing represents a class of graphs built by taking two vertex-disjoint triangles, and linking  
 919 pairs of corners of the triangles by vertex-disjoint paths.

918 **Proof.** Let  $v_1, \dots, v_4$  be the vertices of  $H'$  and  $V_1, \dots, V_4$  be a model of  $H'$ . Let us now distinguish  
 919 the cases depending on the maximum degree in  $H'$ . For each case, we characterize the form of  $G$ .

920 **Case 1:  $H'$  is acyclic.** We claim that if  $H'$  is acyclic, then  $G$  has to be a copy of  $H'$  (which is  
 921 Item 1 in the lemma). If  $H'$  has a node of degree 3, then  $G$  should have a node of degree 3, and by  
 922 minimality  $G$  is exactly one node with degree 3 and its three neighbors, that is, exactly the same as  
 923  $H'$ . If  $H'$  has no node of degree 3, then in the model of  $H'$  every  $V_i$  contains exactly one node by  
 924 minimality (using Corollary 40), and again  $G$  has to be a copy of  $H'$ .

925 **Case 2:  $H'$  has at most one degree-3 vertex.** By Case 1,  $H'$  contains a  $C_3$  or a  $C_4$ .

926 If  $H' = C_4$ , it means that  $G$  contains a cycle of size at least 4 and that every such cycle contains  
 927 all the nodes of the  $G$ . In other words, either  $G$  has size exactly four, or  $G$  is an induced cycle  
 928 (Items 2 and 3 in the lemma).

929 If  $H'$  is a triangle plus a node, then we claim that  $G$  is an induced cycle plus a unique vertex  
 930 (Item 4 in the lemma). Indeed, since  $H'$  has at most one degree 3 node, the vertex not in the triangle  
 931 has degree at most one. Thus, for any cycle  $C$  of  $G$ , the cycle plus any node incident to  $C$  is a  
 932  $H'$ -minor. Since  $G$  is  $H'$ -minimal, the graph  $G$  is an induced cycle plus a node.

933 **Case 3:  $H'$  has two degree 3 vertices.** In this case,  $H'$  is a triangle plus a vertex of degree two  
 934 (that is, a diamond) or three (that is, a  $K_4$ ). Let  $V_1, V_2, V_3, V_4$  be a model of  $H'$  where (at least)  $V_3$   
 935 and  $V_4$  are associated to degree 3 vertices of  $H'$ .

936 Assume that both  $V_3$  and  $V_4$  have type A or B. In this case, there is a unique vertex  $x \in V_3$   
 937 incident to a vertex  $y$  in  $V_4$ . If we add  $y$  to  $V_3$ , and remove it from  $V_4$ , then the size of  $V_4$  is decreasing,  
 938 and the  $V_i$ 's still form a model of  $H'$ . We can repeat this operation until  $V_4$  does not have type A  
 939 or B. Therefore, we can assume that  $V_3$  or  $V_4$  has type C.

940 *Case 3.a.* Assume first that  $H'$  is a diamond. Then  $v_1$  and  $v_2$  have degree 2, and by Corollary 40,  
 941 the subsets  $V_1$  and  $V_2$  are paths. Moreover, if there is an edge between them, and one  $V_i$  has two  
 942 vertices, we could remove one of these vertices, and still have a model of  $H'$ . Therefore, each  $V_i$  is  
 943 reduced to a single vertex (otherwise  $G$  is not  $H'$ -minimal) hence  $G$  is  $K_4$  (Item 2 in the lemma).  
 944 Otherwise, if one of  $V_3, V_4$  has type A, then  $G \setminus V_1$  contains a diamond as a minor, a contradiction  
 945 since  $G$  is  $H'$ -minimal. Moreover, as we already observed, at least one of  $V_3, V_4$ , say  $V_3$ , has type C.

946 Assume that  $V_4$  has type B, and let  $u \in V_4$  be the vertex of  $V_4$  adjacent to  $V_3$ . If  $u$  sees two  
 947 vertices of  $V_3$ , then replacing  $(V_3, V_4)$  by  $(V_3 \cup \{u\}, V_4 \setminus u)$  gives a model where  $V_3 \cup \{u\}$  has  
 948 type A, a contradiction. Therefore,  $u$  sees a unique vertex of  $V_3$  and  $G$  is a subdivided diamond  
 949 (Item 1 in the lemma).

950 Otherwise,  $V_4$  has type C, hence  $V_3$  and  $V_4$  induce two paths (with maybe edges between them).

951 There cannot be two edges between  $V_3$  and  $V_4$ , otherwise,  $G \setminus V_1$  contains a diamond-minor, a  
 952 contradiction. Therefore, there is only one edge between  $V_3$  and  $V_4$  and  $G$  is again a subdivision of  
 953 a diamond (Item 1 in the lemma). And this finishes the analysis for the case where  $H'$  is a diamond.

954 *Case 3.b.* Assume that  $H' = K_4$ . Let us first prove the following claim:

955  $\triangleright$  **Claim 43.** If  $G$  is  $K_4$ -minimal, and  $G$  contains two vertex-disjoint cycles  $C_1, C_2$  with three  
 956 pairwise non-incident edges between  $C_1$  and  $G \setminus C_1$ , then  $G$  is the graph depicted on Figure 9.

957 **Proof.** Let  $a_1b_1, a_2b_2, a_3b_3$  be the edges from the statement, with  $a_i \in C_1$ .

958 Assume first that  $C_2$  is not a triangle. Then we can remove a vertex of  $G \setminus C_1$  in such a way it  
 959 remains connected and still contains the  $b_i$ 's. This gives a  $K_4$ -model, a contradiction. Hence, we  
 960 assume that  $C_2$  is a triangle.

961 We say that  $u \in C_2$  has a private path to one of the  $a_i$ 's if it has such a path that avoids  $C_2 \setminus \{u\}$ .  
 962 If some vertex  $u \in C_2$  has no such path, then  $(G \setminus C_1) \setminus \{u\}$  is connected, hence  $G \setminus u$  contains a  
 963  $K_4$  minor, a contradiction. Moreover, if two vertices  $u, v \in C_2$  have a private path to the same  $a_i$ ,  
 964 then we get a  $K_4$  minor avoiding some other  $a_j$ . Therefore, each vertex of  $C_2$  is associated with a  
 965 unique  $a_i$  by considering private paths. Observe that there is exactly one path for each of the three  
 966 choice of endpoints (since if there were two paths, one could remove a vertex which lies in one path  
 967 but not in the other and get a  $K_4$  minor).

968 It remains to show that  $C_1$  is a triangle. To this end, observe that the structure we found on  
 969  $G \setminus C_2$  ensures that the hypotheses of the statement are still met when exchanging  $C_1$  with  $C_2$ ,  
 970 and the first argument of the proof shows that  $C_1$  is a triangle.  $\blacktriangleleft$

971 The remarks at the beginning of Case 3 ensure that all but at most one set  $V_i$  (say  $V_4$ ) are of  
972 type C. We now do a case analysis of the type of  $V_4$ .

973 Assume that  $V_4$  has type A. Then  $V_4$  contains a triangle  $C$ . Moreover,  $G \setminus C$  contains a cycle  
974 since it contains  $V_1, V_2, V_3$  which are pairwise connected. So by Claim 43, the graph is of the form  
975 of Figure 9 (Item 5 in the lemma).

976 Assume now that  $V_4$  has type B. That is,  $V_4$  is a subdivided star with three branches. Let  $x$  be  
977 the vertex of  $V_4$  of degree three and  $a_1, a_2, a_3$  be the endpoints of the subdivided star rooted in  $x$   
978 (note that the  $a_i$ 's are indeed distinct from  $x$ ). Without loss of generality, each  $a_i$  is connected to  $V_i$   
979 (and not to some other  $V_j$  since otherwise  $G \setminus a_j$  contains a  $K_4$ -minor). If some  $a_i$  is connected  
980 to at least two vertices of  $V_i$ , then  $G \setminus (\{a_i\} \cup V_i)$  contains a cycle as well as  $V_i \cup \{a_i\}$  with the  
981 conditions of Claim 43. So the graph is the graph of Figure 9 (Item 5 in the lemma). Therefore, each  
982  $a_i$  is connected to exactly one vertex of  $V_i$  and  $G$  is a subdivided  $K_4$  (Item 1 in the lemma).

983 Assume now that  $V_4$  has type C. That is, the four sets are of type C. We focus on  $V_1$ . We extend  
984  $V_1$  greedily, that is, if we can add a vertex  $v$  of some  $V_j$  to  $V_1$ , in such a way can still find a model  
985 of  $K_4$  where one of the set is  $V_1 \cup v$ , we do it. So from now on, we can assume that any addition of  
986 a neighbor of a vertex of  $V_j$  to  $V_1$  does not keep a model. We can assume that  $V_1$  has still type C,  
987 otherwise we can apply the previous cases. Now we claim that  $V_2 \cup V_3 \cup V_4$  is a cycle  $C$ . Indeed, it  
988 must contain a cycle, since  $V_2, V_3, V_4$  is a model of the triangle. If  $w$  is not in this cycle, either it  
989 is adjacent to  $V_1$  and then can be added to  $V_1$  (a contradiction) or it is not, and then  $G \setminus w$  has a  
990 model of  $K_4$  and then  $H = K_4 + K_1$  appears as a minor in  $G$ . Let  $C$  be the cycle containing all the  
991 vertices of  $V_2, V_3, V_4$  and  $X_1, X_2, X_3$  the neighbors of  $V_1$  in respectively  $V_2, V_3, V_4$ .

992 Assume that the cycle  $C$  is not induced. Any chord of the cycle separate the cycle into two  
993 sides. If there is a chord of  $C$  that leaves one side of the cycle with at least one element of each of  
994  $X_1, X_2, X_3$ , then we can remove a vertex on the opposite side of the cycle and still have a  $K_4$ -minor,  
995 a contradiction with the  $H$ -minimality of  $G$ . So, without loss of generality, the chord separates  $X_1$   
996 on one side and at least one element from  $X_2, X_3$  on the other side. Let  $e$  be the chord and  $P, P'$  be  
997 the two parts of  $C$  separated by  $e$  where  $P'$  only contains  $X_1$ . In this case, we can apply Claim 43  
998 with the cycle  $e + P'$ , and a cycle using  $V_1$ , a part of  $P$  and edges between  $V_1$  and  $X_2, X_3$ . Hence,  
999 this case again boils down to the graphs of Figure 9 (Item 5 in Lemma 42).

1000 So we can assume that  $C$  is an induced cycle. If  $V_1$  is reduced to a single vertex, then the graph  
1001  $G$  is a wheel (an induced cycle plus a vertex incident to at least 3 vertices of the cycle) and it is  
1002  $K_4$ -minimal (Item 4 in Lemma 42). So we can assume that  $V_1$  has at least two vertices. And it is a  
1003 path since it has type C and both endpoints of the path have neighbors in  $C$  (otherwise the model is  
1004 not minimal). Since we have a  $K_4$ -model, we need the whole set  $V_1$  to have at least three different  
1005 neighbors on  $C$ .

1006 First, note that every vertex of  $V_1$  has at most 2 neighbors on  $C$  (otherwise, we have a  $K_4 + K_1$   
1007 model since  $V_1$  contains at least two vertices). More generally, if a subpath of  $V_1$  has at least three  
1008 neighbors on  $C$ , we have a contradiction. So we can assume that, both extremities of  $V_1$  have a  
1009 private neighbor in  $C$  and, in total  $V_1$  is adjacent to at most 4 vertices in  $C$ .

1010 Now if  $N(V_1) \cap C$  has size 4, we claim that only extremities of  $V_1$  have neighbors in  $C$  and each  
1011 extremity has exactly two neighbors. Let us denote by  $v_1, \dots, v_\ell$  the vertices of  $V_1$  with neighbors  
1012 in  $C$  (in that order in the path  $V_1$ ). Since  $|N(\cup_{i \leq \ell-1} v_i) \cap C| \leq 2$  and  $|N(\cup_{i=2}^\ell v_i) \cap C| \leq 2$ , we have  
1013  $\ell = 2$ . So  $v_1$  and  $v_2$  are the two extremities of  $V_1$  and  $N(v_1) \cap C = \{a, b\}$  and  $N(v_2) \cap C = \{c, d\}$   
1014 where  $a, b, c, d$  are pairwise distinct. If  $a, b, c, d$  appear in that order then we are in the case of Item 6  
1015 in Lemma 42. So we can assume up to symmetry that the vertices  $a, c, b, d$  appear in that order in  
1016  $C$ . If the cycle is has length 4 and  $v_1 v_2$  is an edge, then the graph is  $K_{3,3}$  (Item 7). Now if at least  
1017 one of  $ac, cb, bd, da, v_1 v_2$  is not an edge, we have a  $K_4 + K_1$ . Assume that one of  $ac, cb, bd, da$  is  
1018 subdivided, w.l.o.g.  $ac$ . Let  $x$  be a vertex between  $a$  and  $c$  in  $C$ , then  $C' = v_1 b C_{bd} C_{da} a v_1$  is an

1019 induced cycle and there are three paths in  $G \setminus x$  from  $v_2$  to  $C'$  (to  $v_1, b, d$ ), a contradiction. If  $v_1 v_2$   
 1020 is subdivided then we get the conclusion with the same cycle but three paths are leaving from  $c$  (to  
 1021  $a, b$  and  $d$ )

1022 So, we can now assume that  $V_1$  has at most 3 neighbors in  $C$ . But now, let  $v_1, \dots, v_\ell$  be the  
 1023 vertices of  $V_1$  with neighbors in  $C$ . Since  $v_1$  and  $v_\ell$  have private neighbors on  $C$ , all the other  
 1024 vertices are adjacent to the same vertex  $w$  of  $C$ . Let us denote by  $a$  and  $b$  respectively the private  
 1025 neighbors of  $v_1$  and  $v_\ell$ . Note that  $v_\ell v_1 v_1 a P_{ab} b v_\ell$  is an induced cycle  $C'$  (where  $P_{ab}$  is the subpath of  
 1026  $C$  from  $a$  to  $b$  avoiding  $w$ ). If the only vertex outside of  $C'$  is  $w$ , then we are in Item 4 in Lemma 42.  
 1027 If  $w$  sees only one  $v_i$ , then the graph is a subdivided  $K_4$ , which is Item 1. Otherwise, there are at  
 1028 least 4 paths starting from  $w$  to  $C'$  and one of them is subdivided. The removal of a vertex in a  
 1029 subdivided path still leaves a graph with a  $K_4$  minor, a contradiction. It completes the proof. ◀

1030 We can now derive the theorem from the lemma. All the classes of Lemma 42 can be certified  
 1031 with certificates of size  $O(\log n)$ , and in these classes it is easy to certify  $H'$ -minimality. This is  
 1032 because in all these classes there is a constant number of special vertices: the nodes before the  
 1033 subdivision for Item 1, the additional node for Item 4, the corners of the triangles for Item 5, the  
 1034 nodes of degree larger than 2 in Item 6, and none for Items 2 and 3. The vertices that are not special  
 1035 have degree 2. Certifying these classes boils down to having spanning trees pointing to the special  
 1036 vertices, and having a certification of every path of non-special nodes, to transfer the knowledge  
 1037 of the endpoints from one side of the paths to the other. Then basic consistency checks verify the  
 1038 certification. Because the structure is so constrained, it is easy also to check whether the graph is  
 1039  $H'$ -minimal.

1040 Let us finish this section with an observation. Since the graph  $G$  is connected, the same proof  
 1041 holds for a graph  $H'$  plus a single vertex of degree one as long as all the vertices of  $H'$  are equivalent.  
 1042 A graph is *vertex transitive* if for every pair of vertices  $(u, v)$ , there exists an automorphism of  $H$   
 1043 mapping  $u$  to  $v$ .

1044 ▶ **Lemma 44.** *Let  $H$  be a graph on 5 vertices obtained by adding a pending edge to a vertex transitive*  
 1045 *graph. Then  $H$ -minor free graph can be certified with  $O(\log n)$  bits.*

## 1046 10 Lower bounds

1047 In this section, we show logarithmic lower bounds for  $H$ -minor-freeness for every 2-connected  
 1048 graph  $H$ . These results generalize the lower bounds of [22] for  $K_k$  and  $K_{p,q}$ . Our technique is a  
 1049 simple reduction from the certification of paths, via a local simulation. In contrast, the proofs of [21]  
 1050 were ad-hoc adaptations of the constructions of [29] and [20], with explicit counting arguments.  
 1051 Moreover, our lower bounds apply in the stronger model of locally checkable proofs, where the  
 1052 verifier can look at a constant distance.

1053 ▶ **Theorem 45.** *For every 2-connected graph  $H$ , certifying  $H$ -minor-freeness requires  $\Omega(\log n)$  bits.*

1054 Let us start by proving a couple of lemmas. Let  $H$  be a 2-connected graph, and let  $e = uv$  be  
 1055 an arbitrary edge of  $H$ . Let  $H^-$  be the graph  $H \setminus e$ . Note that  $H^-$  is connected. We are going to  
 1056 consider copies of  $H^-$ , that we index as  $H_i^-$ 's, and where the copies of the nodes  $u$  and  $v$  will be  
 1057 called  $u_i$  and  $v_i$ . Let  $\mathcal{P}$  be the class of all the graphs that can be made by taking some  $k$  copies of  
 1058  $H^-$ , and by identifying for every  $i \in [1, k - 1]$ ,  $v_i$  with  $u_{i+1}$ . In other words,  $\mathcal{P}$  is the set of paths,  
 1059 where every edge is a copy of  $H^-$ . The class  $\mathcal{C}$  is the same as  $\mathcal{P}$  except that we close the paths into  
 1060 cycles, that is, we identify  $v_k$  with  $u_1$ .

1061 ▶ **Lemma 46.** *The graphs of  $\mathcal{P}$  are all  $H$ -minor-free, and the graphs of  $\mathcal{C}$  all contain  $H$  as a minor.*

1062 **Proof.** Let  $G$  be a graph of  $\mathcal{P}$ . Note that every vertex  $v_i$  (identified with  $u_{i+1}$ ) for  $i \in \{1, \dots, k-1\}$ ,  
 1063 is a cut vertex of  $G$ . Therefore, since  $H$  is 2-connected, a model of  $H$  can only appear between two  
 1064 such nodes. By construction this cannot happen, as the graphs between the cut vertices are all  $H^-$ .  
 1065 Thus  $G$  is  $H$ -minor-free.

1066 Now let  $G$  be a graph of  $\mathcal{C}$ . We claim that  $G$  contains  $H$  as a minor. Consider the following  
 1067 model of  $H$ . Any  $H_i^-$  is a model of  $H$  except for the edge  $uv$ . Since we have made a cycle of  $H_i^-$ 's,  
 1068 there is a path between  $v_i$  and  $u_i$  outside  $H_i^-$ , and this path finishes the model of  $H$ . ◀

1069 ► **Lemma 47.** *Let  $H$  be a 2-connected graph. If there is a certification with  $O(f(n))$  bits for  $H$ -  
 1070 minor-free graphs, then there is a  $O(f(n))$  certification for paths.*

1071 **Proof.** Suppose there exists a certification with  $O(f(n))$  bits for  $H$ -minor-free graphs. The certifi-  
 1072 cation of paths boils down to differentiate between paths and cycles, since the nodes can locally  
 1073 check that they have degree 2. Consider the following certification of paths. The idea is that the  
 1074 nodes of the path (or cycle) will simulate the computation they would do if instead of being linked  
 1075 by edges, they were linked by copies of  $H^-$ . The prover will give to every node the certificates  
 1076 of  $H$ -minor-freeness for these simulated graphs, that is, for every node the certificates of the two  
 1077 copies of  $H^-$  adjacent to it in the simulated graph. Every node will check with its neighbor in the  
 1078 real graph that they have been given the same certificates for these virtual  $H^-$ . Then every node  
 1079 will run the verification algorithm for  $H$ -freeness in the simulated graph.

1080 By construction, the simulated graph is either in  $\mathcal{P}$  or in  $\mathcal{C}$ . Thus, if the verification algorithm  
 1081 accepts, that is, if the simulated graph is  $H$ -minor-free, then the graph is in  $\mathcal{P}$ , and then the real  
 1082 graph is a path. If the verification algorithm rejects, that is if the simulated graph is not  $H$ -minor-free  
 1083 then the graph is in  $\mathcal{C}$ , and then the real graph is a cycle. In other words we have designed a local  
 1084 certification for paths, with certificates of size  $O(f(n))$ . ◀

1085 **Proof of Theorem 45.** Now Theorem 45 follows from the fact that paths cannot be certified with  
 1086  $o(\log n)$  bits [29, 33]. Note that the proof applies in the locally checkable proof setting, as soon as  
 1087 the number of copies of  $H^-$  is large enough, since the lower bound for paths also applies to locally  
 1088 checkable proofs. ◀

## 1089 11 Discussion

### 1090 Milestones to go further

1091 In this paper, we develop several tools and use them to show that some minor closed graph classes  
 1092 can be certified with  $O(\log n)$  bits. One can probably use the tools we developed to certify new  
 1093 classes, we simply wanted to illustrate the interest of these tools. Let us now discuss the tools that  
 1094 are missing in order to tackle the general question on  $H$ -minor-freeness and which steps can be  
 1095 interesting to tackle it.

1096 First, as we explained in Section 9, certification of  $H$ -minor free classes seems easier when  $H$  is  
 1097 sparse. One first question that might be interested to look at is the following:

1098 ► **Question 48.** *Let  $T$  be a tree. Can  $T$ -minor free graphs be certified with  $O(\log n)$  bits?*

1099 The answer to this question for small graphs  $H$  (up to 5 vertices) is not very interesting, since  
 1100 the number of vertices of degree at least 3 is bounded (and then the whole structure of the graph is  
 1101 "simple"). Even if it remains simple for any  $H$ , there is no trivial argument allowing us to certify  
 1102 these nodes with  $O(\log n)$  bits. In the light of the recent results that establish that  $O(\log n)$  bits is  
 1103 doable for paths minors [23], and  $O(\log^2 n)$  is doable for planar minors [26], Question 48 seems to  
 1104 be the simplest open question.

1105 A natural approach to tackle Conjecture 1 would consist in an induction on the size of  $H$ . Indeed,  
 1106 knowing how to certify  $H \setminus x$  for any possible  $x$  may help to certify  $H$ . The basic idea would  
 1107 consist in separating two cases. 1) When  $H$  is not heavily connected where we can heavily use the  
 1108 fact that we can  $H \setminus x$  can be certified. And 2) when  $H$  is heavily connected, try to use a more  
 1109 general argument. A first step toward step 1) would consist in proving that if  $H$ -minor-freeness can  
 1110 be certified then so is  $H + K_1$ -minor-freeness<sup>2</sup>. We proved it for five vertices in Theorem 41, but  
 1111 the proof heavily uses the structure of the graphs on four vertices. One can then naturally ask the  
 1112 following general question:

1113 ► **Question 49.** *Let  $H$  be a graph. Can  $(H + K_1)$ -minor free graphs be certified with  $O(\log n)$  bits*  
 1114 *when  $H$  can be certified with  $O(\log n)$  bits?*

1115 As in the proof of Theorem 41, we know that we can assume that  $G$  is  $H$ -minimal. Even if most  
 1116 of the techniques for Lemma 41 are specific, Corollary 40 gives some (basics) general properties of  
 1117  $H$ -minimal graphs which might be useful to tackle this question.

1118 In structural graph theory, a particular class of  $H$ -minimal graphs received a considerable  
 1119 attention which are minimally non-planar graphs, in other words, graphs  $G$  that are minimal and  
 1120 that contains either a  $K_5$  or a  $K_{3,3}$  as a minor. It might be interesting to determine if minimally  
 1121 non-planar graphs can be certified with  $O(\log n)$  bits.

1122 Note that if we can answer positively Question 49 positively, the second step would consist in  
 1123 proving the conjecture when we add to  $H$  a vertex attached to a single vertex of  $H$ . Proving this  
 1124 case would, in particular, imply a positive answer to Question 48.

1125 If we want to consider dense graphs, the questions seem to become even harder. In particular,  
 1126 one of the first main complicated  $H$ -minor class to deal with is probably the class of  $K_5$ -minor-free  
 1127 graphs. There are several reasons for that. First, it is the smallest 4-connected graph and the  
 1128 hardness to certify seem to be highly related to the connectivity of the graph that is forbidden as a  
 1129 minor. The second reason is that it is the smallest graph for which  $H$ -minor free graphs is a super  
 1130 class of planar graphs. In other words, we cannot take advantage of the “planarity” of the graph  
 1131 (formally or informally) to certify the graph class. We then ask the following question:

1132 ► **Question 50.** *Can  $K_5$ -minor free graphs be certified with  $O(\log n)$  bits?*

1133 Wagner proved in [42] that a graph is  $K_5$ -minor-free if and only if it can be built from planar  
 1134 graphs and from a special graph  $V_8$  by repeated clique sums. A *clique sum* consists in taking two  
 1135 graphs of the class and gluing them on a clique and then (potentially) remove edges of that clique.  
 1136 While it should have been easy to certify this sum if we keep the edges of the clique, the fact that  
 1137 they might disappear makes the work much more complicated for certification.

1138 More generally, many decompositions are using the fact that we replace a subgraph by a smaller  
 1139 structure (a single vertex or an edge for instance) only connected to the initial neighbors of that  
 1140 structure in the graph. Certifying such structures is a challenging question whose positive answer  
 1141 can probably permit to break several of the current hardest cases.

## 1142 Obstacles towards lower bounds

1143 There are also several obstacles preventing us to prove extra-logarithmic lower bounds for the  
 1144 certificate size of  $H$ -minor-free graphs. Basically, the only techniques we know consist in (explicit  
 1145 or implicit) reductions to communication complexity. In particular [29] and [7] designed lower

---

<sup>2</sup>  $H + K_1$  is the graph  $H$  plus an isolated vertex.

1146 bounds for respectively non-3-colorable graphs and bounded diameter graphs as reductions from  
 1147 the disjointness problem in non-deterministic communication complexity.

1148 Let us remind what these reductions look like. In such a reduction, one considers a family of  
 1149 graphs with two vertex sets  $A$  and  $B$ , with few edges in between. These graphs are defined in  
 1150 such a way that the input of Alice for the disjointness problem can be encoded in the edges of  
 1151  $A$  and the input of Bob in the edges of  $B$ . Then, given a certification scheme, Alice and Bob can  
 1152 basically simulate the verification algorithm, and deduce an answer for the disjointness problem. If  
 1153 a certification with small labels existed for the property at hand, then the communication protocol  
 1154 would contradict known lower bounds which proves a lower bound for certification.

1155 The difficulty of using this proof for  $H$ -minor free graphs comes from the fact that it is difficult  
 1156 to control where a minor can appear, that is, to control the models of  $H$ . For example, it is difficult  
 1157 to control that if  $H$  appears in the graph, then the nodes  $V_i$  associated with some node  $i$  of  $H$  are  
 1158 on Alice's side. As a comparison, for proving properties on the diameter, [7] used a construction  
 1159 where all the longest paths in the graph had to start from Alice side and finish in Bob side, but such  
 1160 a property seems difficult to obtain for minors.

### 1161 Connectivity questions

1162 A large part of the paper is devoted to certify connectivity and related notions that are of independent  
 1163 importance, for instance to certify the robustness of a network. For these, we do not have lower  
 1164 bounds, and leave the following question open.

1165 ► **Question 51.** *Does the certification of  $k$ -connectivity require  $\Omega(\log n)$  bits?*

1166 For this question it is tempting to try a construction close to the one we have used for  $H$ -minor-  
 1167 free graphs. For example, one could think that the nodes of the path/cycle could simulate the  $k$ -th  
 1168 power of the graph which is  $k$ -connected if and only if the graph is a cycle. But this does not  
 1169 work: we want the *yes*-instances for the property (e.g. the  $k$ -connected graphs) to be mapped to  
 1170 *yes*-instances for acyclicity (e.g. paths), and not with the *no*-instances, which are the cycles.

1171 An interesting open problem about  $k$ -connectivity also is on the positive side:

1172 ► **Question 52.** *Can  $k$ -connectivity be certified with  $O(\log n)$  bits for any  $k \geq 4$ ?*

1173 Beyond the question of certifying the connectivity itself, we would like to be able to decom-  
 1174 pose graphs based on  $k$ -connected components, like what we did with the block-cut tree for  
 1175 2-connectivity. Such decomposition are more complicated and less studied than block-cut trees, but  
 1176 for 3-connectivity such a tool is SPQR trees [3]. Unfortunately, similarly to the clique sum operation  
 1177 we mentioned earlier, some steps of the SPQR tree construction are based on edges that can be  
 1178 removed in later steps, making it hard to certify this structure.

### 1179 Acknowledgments

1180 We thank Jens M. Schmidt for pointing out a mistake in the characterization of 3-connectivity we used  
 1181 in an earlier version of this paper. We also thank the reviewers of the conference versions for their  
 1182 comments.

### 1183 References

- 1184 1 Yehuda Afek, Shay Kutten, and Moti Yung. Memory-efficient self stabilizing protocols for  
 1185 general networks. In *Distributed Algorithms, 4th International Workshop, WDAG '90*, volume  
 1186 486, pages 15–28, 1990. doi:10.1007/3-540-54099-7\_2.

- 1187 2 Kenneth Appel, Wolfgang Haken, et al. Every planar map is four colorable. *Bulletin of the*  
1188 *American mathematical Society*, 82(5):711–712, 1976.
- 1189 3 Giuseppe Di Battista and Roberto Tamassia. Incremental planarity testing (extended ab-  
1190 stract). In *30th Annual Symposium on Foundations of Computer Science*, pages 436–441, 1989.  
1191 doi:10.1109/SFCS.1989.63515.
- 1192 4 Zvika Brakerski and Boaz Patt-Shamir. Distributed discovery of large near-cliques. *Distributed*  
1193 *Comput.*, 24(2):79–89, 2011. doi:10.1007/s00446-011-0132-x.
- 1194 5 Keren Censor-Hillel, Eldar Fischer, Gregory Schwartzman, and Yadu Vasudev. Fast distributed al-  
1195 gorithms for testing graph properties. *Distributed Comput.*, 32(1):41–57, 2019. doi:10.1007/s00446-  
1196 018-0324-8.
- 1197 6 Keren Censor-Hillel, Orr Fischer, Tzlil Gonen, François Le Gall, Dean Leitersdorf, and Rotem  
1198 Oshman. Fast distributed algorithms for girth, cycles and small subgraphs. In *34th International*  
1199 *Symposium on Distributed Computing, DISC 2020*, volume 179 of *LIPIcs*, pages 33:1–33:17, 2020.  
1200 doi:10.4230/LIPIcs.DISC.2020.33.
- 1201 7 Keren Censor-Hillel, Ami Paz, and Mor Perry. Approximate proof-labeling schemes. *Theor.*  
1202 *Comput. Sci.*, 811:112–124, 2020. doi:10.1016/j.tcs.2018.08.020.
- 1203 8 Joseph Cheriyan and S. N. Maheshwari. Finding nonseparating induced cycles and independent  
1204 spanning trees in 3-connected graphs. *J. Algorithms*, 9(4):507–537, 1988. doi:10.1016/0196-  
1205 6774(88)90015-6.
- 1206 9 Markus Chimani, Martina Juhnke-Kubitzke, Alexander Nover, and Tim Römer. Cut polytopes  
1207 of minor-free graphs. *arXiv preprint arXiv:1903.01817*, 2019.
- 1208 10 Maria Chudnovsky, Neil Robertson, Paul Seymour, and Robin Thomas. The strong perfect graph  
1209 theorem. *Annals of mathematics*, pages 51–229, 2006.
- 1210 11 Shlomi Dolev. *Self-Stabilization*. MIT Press, 2000. ISBN 0-262-04178-2. URL [http://www.cs.  
1211 bgu.ac.il/%7Edolev/book/book.html](http://www.cs.bgu.ac.il/%7Edolev/book/book.html).
- 1212 12 Gábor Elek. Planarity can be verified by an approximate proof labeling scheme in constant-time.  
1213 *J. Comb. Theory, Ser. A*, 191:105643, 2022. doi:10.1016/j.jcta.2022.105643.
- 1214 13 Mark N. Ellingham, Emily A. Marshall, Kenta Ozeki, and Shoichi Tsuchiya. A characterization  
1215 of  $K_2$ , 4-minor-free graphs. *SIAM J. Discret. Math.*, 30(2):955–975, 2016. doi:10.1137/140986517.
- 1216 14 David Eppstein. Parallel recognition of series-parallel graphs. *Inf. Comput.*, 98(1):41–55, 1992.  
1217 doi:10.1016/0890-5401(92)90041-D.
- 1218 15 Louis Esperet and Benjamin Lévêque. Local certification of graphs on surfaces. *Theor. Comput.*  
1219 *Sci.*, 909:68–75, 2022. doi:10.1016/j.tcs.2022.01.023.
- 1220 16 Louis Esperet and Sergey Norin. Testability and local certification of monotone properties in  
1221 minor-closed classes. In *49th International Colloquium on Automata, Languages, and Program-*  
1222 *ming, ICALP 2022*, volume 229 of *LIPIcs*, pages 58:1–58:15, 2022. doi:10.4230/LIPIcs.ICALP.2022.58.
- 1223 17 Laurent Feuilloley. Bibliography of distributed approximation on structurally sparse graph  
1224 classes. *CoRR*, abs/2001.08510, 2020.
- 1225 18 Laurent Feuilloley. Introduction to local certification. *Discrete Mathematics & Theoretical*  
1226 *Computer Science*, vol. 23, no. 3, 2021. doi:10.46298/dmtcs.6280.
- 1227 19 Laurent Feuilloley and Pierre Fraigniaud. Survey of distributed decision. *Bulletin of the EATCS*,  
1228 119, 2016. url: [bulletin.eatcs.org](http://bulletin.eatcs.org) link, arXiv: 1606.04434.
- 1229 20 Laurent Feuilloley and Juho Hirvonen. Local verification of global proofs. In *32nd International*  
1230 *Symposium on Distributed Computing, DISC 2018*, volume 121 of *LIPIcs*, pages 25:1–25:17, 2018.  
1231 doi:10.4230/LIPIcs.DISC.2018.25.
- 1232 21 Laurent Feuilloley, Pierre Fraigniaud, Pedro Montealegre, Ivan Rapaport, Eric Rémila, and Ioan  
1233 Todinca. Local certification of graphs with bounded genus. *CoRR*, abs/2007.08084, 2020.

- 1234 22 Laurent Feuilloley, Pierre Fraigniaud, Pedro Montealegre, Ivan Rapaport, Éric Rémila, and Ioan  
1235 Todinca. Compact distributed certification of planar graphs. *Algorithmica*, 83(7):2215–2244,  
1236 2021. doi:10.1007/s00453-021-00823-w.
- 1237 23 Laurent Feuilloley, Nicolas Bousquet, and Théo Pierron. What can be certified compactly?  
1238 compact local certification of MSO properties in tree-like graphs. In *PODC '22: ACM Symposium*  
1239 *on Principles of Distributed Computing*, pages 131–140. ACM, 2022. doi:10.1145/3519270.3538416.
- 1240 24 Paola Flocchini and Flaminia L. Luccio. Routing in series parallel networks. *Theory Comput.*  
1241 *Syst.*, 36(2):137–157, 2003. doi:10.1007/s00224-002-1033-y.
- 1242 25 Pierre Fraigniaud and Dennis Olivetti. Distributed detection of cycles. *ACM Trans. Parallel*  
1243 *Comput.*, 6(3):12:1–12:20, 2019. doi:10.1145/3322811.
- 1244 26 Pierre Fraigniaud, Pedro Montealegre, Ivan Rapaport, and Ioan Todinca. A meta-theorem  
1245 for distributed certification. In *Structural Information and Communication Complexity - 29th*  
1246 *International Colloquium, SIROCCO 2022*, page To appear., 2022.
- 1247 27 Mohsen Ghaffari and Bernhard Haeupler. Distributed algorithms for planar networks II:  
1248 low-congestion shortcuts, MST, and min-cut. In *Proceedings of the Twenty-Seventh An-*  
1249 *annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016*, pages 202–219. SIAM, 2016.  
1250 doi:10.1137/1.9781611974331.ch16.
- 1251 28 Mohsen Ghaffari and Bernhard Haeupler. Low-congestion shortcuts for graphs excluding dense  
1252 minors. In *PODC '21: ACM Symposium on Principles of Distributed Computing*, pages 213–221.  
1253 ACM, 2021. doi:10.1145/3465084.3467935.
- 1254 29 Mika Göös and Jukka Suomela. Locally checkable proofs in distributed computing. *Theory of*  
1255 *Computing*, 12(19):1–33, 2016. doi:10.4086/toc.2016.v012a019.
- 1256 30 Bernhard Haeupler, Taisuke Izumi, and Goran Zuzic. Near-optimal low-congestion shortcuts  
1257 on bounded parameter graphs. In *Distributed Computing - 30th International Symposium, DISC*  
1258 *2016*, volume 9888, pages 158–172. Springer, 2016. doi:10.1007/978-3-662-53426-7\_12.
- 1259 31 Bernhard Haeupler, Jason Li, and Goran Zuzic. Minor excluded network families admit fast  
1260 distributed algorithms. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed*  
1261 *Computing, PODC 2018*, pages 465–474, 2018.
- 1262 32 Benjamin Jauregui, Pedro Montealegre, and Ivan Rapaport. Distributed interactive proofs for  
1263 the recognition of some geometric intersection graph classes. In *Structural Information and*  
1264 *Communication Complexity - 29th International Colloquium, SIROCCO 2022*, page To appear.,  
1265 2022.
- 1266 33 Amos Korman, Shay Kutten, and David Peleg. Proof labeling schemes. *Distributed Computing*,  
1267 22(4):215–233, 2010. doi:10.1007/s00446-010-0095-3.
- 1268 34 Alexandr V Kostochka. The minimum hadwiger number for graphs with a given mean degree  
1269 of vertices. *Metody Diskret. Analiz.*, (38):37–58, 1982.
- 1270 35 Lee F. Mondschein. *Combinatorial Ordering and the Geometric Embedding of Graphs*. PhD thesis,  
1271 M.I.T. Lincoln Laboratory / Harvard University, 1971.
- 1272 36 Pedro Montealegre, Diego Ramírez-Romero, and Ivan Rapaport. Compact distributed interactive  
1273 proofs for the recognition of cographs and distance-hereditary graphs. In *Stabilization, Safety,*  
1274 *and Security of Distributed Systems - 23rd International Symposium, SSS 2021*, volume 13046,  
1275 pages 395–409, 2021. doi:10.1007/978-3-030-91081-5\_26.
- 1276 37 Moni Naor, Merav Parter, and Eylon Yogev. The power of distributed verifiers in interactive  
1277 proofs. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020*,  
1278 pages 1096–115. SIAM, 2020. doi:10.1137/1.9781611975994.67.
- 1279 38 H. E. Robbins. A theorem on graphs, with an application to a problem of traffic control. *The*  
1280 *American Mathematical Monthly*, 46(5):281–283, 1939. ISSN 00029890, 19300972.

- 1281 39 Neil Robertson and Paul D Seymour. Graph minors—a survey. *Surveys in combinatorics*, 103:  
1282 153–171, 1985.
- 1283 40 Jens M. Schmidt. Mondschein sequences (a.k.a.  $(2, 1)$ -orders). *SIAM J. Comput.*, 45(6):1985–2003,  
1284 2016. doi:10.1137/15M1030030.
- 1285 41 Andrew Thomason. An extremal function for contractions of graphs. In *Mathematical Proceedings*  
1286 *of the Cambridge Philosophical Society*, volume 95, pages 261–265. Cambridge University Press,  
1287 1984.
- 1288 42 K. Wagner. Über eine eigenschaft der ebenen komplex. In *Math. Ann.*, volume 114, pages  
1289 570–590, 1937.
- 1290 43 Hassler Whitney. Non-separable and planar graphs. *Transactions of the American Mathematical*  
1291 *Society*, 34:339–362, 1932. doi:10.1090/S0002-9947-1932-1501641-2.