Local Certification of Graphs with Bounded Genus

Laurent Feuilloley[a,1,*], Pierre Fraigniaud[b,2], Pedro Montealegre[c,3], Ivan Rapaport[d,4], Éric Rémila[e,5], Ioan Todinca[f]

[a] *Univ. Lyon, Université Lyon 1, LIRIS UMR CNRS 5205, F-69621, Lyon, France*
[b] *IRIF, CNRS and Université de Paris Cité, Paris, France*
[c] *Facultad de Ingeniería y Ciencias, Universidad Adolfo Ibáñez, Santiago, Chile*
[d] *DIM-CMM (UMI 2807 CNRS), Universidad de Chile, Santiago, Chile*
[e] *Univ Lyon, UJM Saint-Etienne, GATE L-SE UMR 5824, Saint-Étienne, France*
[f] *LIFO, Université d'Orléans and INSA Centre-Val de Loire, Orléans, France*

Abstract

Naor, Parter, and Yogev [SODA 2020] recently designed a compiler for automatically translating standard centralized interactive protocols to *distributed* interactive protocols, as introduced by Kol, Oshman, and Saxena [PODC 2018]. In particular, by using this compiler, every linear-time algorithm for deciding the membership to some fixed graph class can be translated into a $\mathsf{dMAM}(O(\log n))$ protocol for this class, that is, a distributed interactive protocol with $O(\log n)$-bit proof size in n-node graphs, and three interactions between the (centralized) computationally-unbounded but non-trustable *prover* Merlin, and the (decentralized) randomized computationally-limited *verifier*

*Corresponding author

Email addresses: laurent.feuilloley@univ-lyon1.fr (Laurent Feuilloley), pierre.fraigniaud@irif.fr (Pierre Fraigniaud), p.montealegre@uai.cl (Pedro Montealegre), rapaport@dim.uchile.cl (Ivan Rapaport), eric.remila@univ-st-etienne.fr (Éric Rémila), ioan.todinca@univ-orleans.fr (Ioan Todinca)

[1] Additional funding from Institute for Research in Market Imperfections and Public Policy (MIPP) and ANR project GrR (ANR-18-CE40-0032).

[2] Additional funding from ANR project DESCARTES, and INRIA project GANG

[3] Additional funding from ANID via PAI + Convocatoria Nacional Subvención a la Incorporación en la Academia Año 2017 + PAI77170068 and FONDECYT 11190482

[4] Additional funding from CONICYT via PIA/Apoyo a Centros Científicos y Tecnológicos de Excelencia AFB 170001 and Fondecyt 1220142

[5] Additional funding from IDEX LYON (project INDEPTH) within ANR-16-IDEX-0005 and MODMAD

Arthur. As a corollary, there is a $\mathsf{dMAM}(O(\log n))$ protocol for recognizing the class of planar graphs, as well as for recognizing the class of graphs with bounded genus.

We show that there exists a distributed interactive protocol for recognizing the class of graphs with bounded genus performing just a *single* interaction, from the prover to the verifier, yet preserving proof size of $O(\log n)$ bits. This result also holds for the class of graphs with bounded *non-orientable genus*, that is, graphs that can be embedded on a *non-orientable* surface of bounded genus. The interactive protocols described in this paper are actually *proof-labeling schemes*, i.e., a subclass of interactive protocols, previously introduced by Korman, Kutten, and Peleg [PODC 2005]. In particular, these schemes do *not* require any randomization from the verifier, and the proofs may often be computed a priori, at low cost, by the nodes themselves. Our results thus extend the recent proof-labeling scheme for planar graphs by Feuilloley et al. [PODC 2020], to graphs of bounded genus, and to graphs of bounded non-orientable genus.

1. Introduction

1.1. Context and Objective

The paper considers the standard setting of distributed network computing, in which processing elements are nodes of a network modeled as a simple connected graph $G = (V, E)$, and the nodes exchange information along the links of that network (see, e.g., [50]). As for centralized computing, distributed algorithms often assume promises on their inputs, and many algorithms are designed for specific families of graphs, including regular graphs, planar graphs, graphs with bounded arboricity, bipartite graphs, graphs with bounded treewidth, etc. Distributed *decision* refers to the problem of checking that the actual input graph (i.e., the network itself) satisfies a given predicate. One major objective of the check up is avoiding erroneous behaviors such as livelocks or deadlocks resulting from running an algorithm dedicated to a specific graph family on a graph that does not belong to this family. The decision rule typically specifies that, if the predicate is satisfied, then all nodes must accept, and otherwise at least one node must reject. A single rejecting node can indeed trigger an alarm (in, e.g., hardwired networks), or launch a recovery procedure (in, e.g., virtual networks such as overlay networks). The main goal of distributed decision is to design efficient

checking protocols, that is, protocols where every node exchange information with nodes in its vicinity only, and where the nodes exchange a small volume of information between neighbors.

Proof-Labeling Schemes.. Some graph predicate are trivial to check locally (e.g., regular graphs), but others do not admit local decision algorithms. For instance, deciding whether the network is bipartite may require long-distance communication for detecting the presence of an odd cycle. *Proof-labeling schemes* [39] provide a remedy to this issue. These mechanisms have a flavor of NP-computation, but in the distributed setting. That is, a non-trustable oracle provides each node with a *certificate*, and the collection of certificates is supposed to be a *distributed proof* that the graph satisfies the given predicate. The nodes check locally the correctness of the proof. The specification of a proof-labeling scheme for a given predicate is that, if the predicate is satisfied, then there must exist a certificate assignment leading all nodes to accept, and, otherwise, for every certificate assignment, at least one node rejects. As an example, for the case of the bipartiteness predicate, if the graph is bipartite, then an oracle can color blue the nodes of one of the partition, and color red the nodes of the other partition. It is then sufficient for each node to locally check that all its neighbors have the same color, different from its own color, and to accept or reject accordingly. Indeed, if the graph is not bipartite, then there is no way that a dishonest oracle can fool the nodes, and make them all accept the graph.

Interestingly, the certificates provided to the nodes by the oracle can often be computed by the nodes themselves, at low cost, during some pre-computation. For instance, a spanning tree construction algorithm is usually simply asked to encode the tree T locally at each node v, say by a pointer $p(v)$ to the parent of v in the tree. However, it is possible to ask the algorithm to also provide a distributed proof that T is a spanning tree. Such a proof may be encoded distributedly by providing each node with a certificate containing, e.g., the ID of the root of T, and the distance $d(v)$ from v to the root (see, e.g., [2, 6, 37]). Indeed, every node v but the root can simply check that $d(p(v)) = d(v) - 1$ (to guarantee the absence of cycles), and that it was given the same root-ID as all its neighbors in the network (for guaranteeing the uniqueness of the tree).

Distributed Interactive Protocols.. The good news is that all (Turing-decidable) predicates on graphs admit a proof-labeling scheme [39]. The

bad news is that there are simple graphs properties (e.g., existence of a non-trivial automorphism [39], non 3-colorability [34], bounded diameter [11], etc.) which require certificates on $\widetilde{\Omega}(n)$ bits in n-node graphs. Such huge certificates do not fit with the requirement that checking algorithms must not only be local, but they must also consume little bandwidth. Randomized proof-labeling schemes [29] enable to limit the bandwidth consumption, but this is often to the cost of increasing the space-complexity of the nodes. However, motivated by cloud computing, which may provide large-scale distributed systems with the ability to interact with an external party, Kol, Oshman, and Saxena [38] introduced the notion of *distributed interactive protocols*. In such protocols, a centralized non-trustable oracle with unlimited computation power (a.k.a. Merlin) exchanges messages with a randomized distributed algorithm (a.k.a. Arthur). Specifically, Arthur and Merlin perform a sequence of exchanges during which every node queries the oracle by sending a random bit-string, and the oracle replies to each node by sending a bit-string called *proof*. Neither the random strings nor the proofs need to be the same for each node. After a certain number of rounds, every node exchange information with its neighbors in the network, and decides (i.e., it outputs accept or reject). It was proved that many predicate requiring large certificate whenever using proof-labeling schemes, including the existence of a non-trivial automorphism, have distributed interactive protocols with proofs on $O(\log n)$ bits [38].

Naor, Parter, and Yogev [45] recently designed a compiler for automatically translating standard centralized interactive protocols to distributed interactive protocols. In particular, by using this compiler, every linear-time algorithm for deciding the membership to some fixed graph class can be translated into a $\mathsf{dMAM}(O(\log n))$ protocol, that is, a distributed interactive protocol with $O(\log n)$-bit proof size in n-node graphs, and three interactions between Merlin and Arthur: Merlin provides every node with a first part of the proof, on $O(\log n)$ bits, then every node challenges Merlin with a random bit-string on $O(\log n)$ bits, and finally Merlin replies to every node with the second part of the proof, again on $O(\log n)$ bits. Every node then performs a single round of communication with its neighbors, exchanging $O(\log n)$-bit messages, and individually outputs accept or reject. As a corollary, there is a $\mathsf{dMAM}(O(\log n))$ protocol for many graph classes, including planar graphs, graphs with bounded genus, graphs with bounded treewidth, etc.

The Limits of Distributed Interactive Protocols.. Although the compiler in [45] is quite generic and powerful, it remains that the resulting interactive protocols are often based on many interactions between Merlin and Arthur. This raises the question of whether there exist protocols based on fewer interactions for the aforementioned classes of graphs, while keeping the proof size small (e.g., on $O(\text{polylog}\, n)$ bits). Note that, with this objective in mind, proof-labeling schemes are particularly desirable as they do not require actual interactions. Indeed, as mentioned before, the certificates may often be constructed a priori by the nodes themselves. Unfortunately, under the current knowledge, establishing lower bounds on the number of interactions between the prover Merlin and the distributed verifier Arthur, as well as lower bounds on the proof size, not to speak about tradeoffs between these two complexity measures, remains challenging. Therefore, it is not known whether $\mathsf{dMAM}(O(\log n))$ protocols are the best that can be achieved for graph classes such as graphs with bounded genus, or graphs with bounded treewidth.

Graphs with Bounded Genus.. In this paper, we focus on the class of graphs with bounded genus, for several reasons. First, this class is among the prominent representative of sparse graphs [47], and the design of fast algorithms for sparse graphs is not only of the utmost interest for centralized, but also for distributed computing (see, e.g., [3, 8, 13, 16, 31, 32, 33, 41, 42, 54]), as many real-world physical or logical networks are sparse. Second, graphs of bounded genus, including planar graphs, have attracted lots of attention recently in the distributed computing framework (see, e.g., [4, 5, 30]), and it was shown that this large class of graphs enjoys distributed exact or approximation algorithms that overcome several known lower bounds for general graphs [40, 51, 53]. Last but not least, it appears that the graph classes for which proof-labeling schemes require certificates of large size are not closed under node-deletion, which yields the question of whether every graph family closed under node-deletion (in particular graph families closed under taking minors) have proof-labeling schemes with certificates of small size. This was recently shown to be true for planar graphs [25], but the question is open beyond this class, putting aside simple classes such as bipartite graphs, forests, etc.

As for the class of planar graphs, and for the class of graphs with bounded genus, every graph class $\mathcal{G}$ that is closed under taking minors has a finite set of forbidden minors. As a consequence, as established in [25], there

is a simple proof-labeling scheme with $O(\log n)$-bit certificates for $\overline{\mathcal{G}}$, i.e., for *not* being in $\mathcal{G}$. The scheme simply encodes a forbidden minor present in G in a distributed manner for certifying that $G \notin \mathcal{G}$. Therefore, for every $k \geq 0$, there exists a simple proof-labeling scheme with $O(\log n)$-bit certificates for genus or non-orientable genus *at least* k. The difficulty is to design a proof-labeling scheme with $O(\log n)$-bit certificates for genus or non-orientable genus *at most* k.

1.2. Our Results

1.2.1. Compact Proof-Labeling Schemes for Graphs of Bounded Genus

Recall that planar graphs are graphs embeddable on the 2-dimensional sphere S^2 (without edge-crossings). Graphs with genus 1 are embeddable on the torus $\mathbb{T}_1$, and, more generally, graphs with genus $k \geq 0$ are embeddable on the closed surface $\mathbb{T}_k$ obtained from S^2 by adding k *handles*. We show that, for every $k \geq 0$, there exists a proof-labeling scheme for the class of graphs with genus at most k, using certificates on $O(\log n)$ bits. This extends a recent proof-labeling scheme for planar graphs [25] to graphs with arbitrary genus $k \geq 0$. Note that the certificate-size of our proof-labeling schemes is optimal, in the sense there are no proof-labeling schemes using certificates on $o(\log n)$ bits, even for planarity [25][6].

For every $k \geq 1$, our proof-labeling schemes also apply to the class of graphs with *non-orientable genus* (a.k.a., Euler genus) at most k, that is, they also hold for graphs embeddable on a *non-orientable* surface with genus k. Graphs with non-orientable k are indeed graphs embeddable on the closed surface $\mathbb{P}_k$ obtained from S^2 by adding k *cross-caps*. Some more precise definitions and descriptions are given later, in Section 2.1.

This paper therefore demonstrates that the ability of designing proof-labeling schemes with small certificates for planar graphs is not a coincidental byproduct of planarity, but this ability extends to much wider classes of sparse graphs closed under taking minors. This provides hints that proof-labeling schemes with small certificates can also be designed for very many (if not all) natural classes of sparse graphs closed under vertex-deletion.

[6] The goal of this paper is not to optimize the size of the certificates as a function of the genus k, but it is not hard to see that our certificates have size $O(2^k \log n)$.

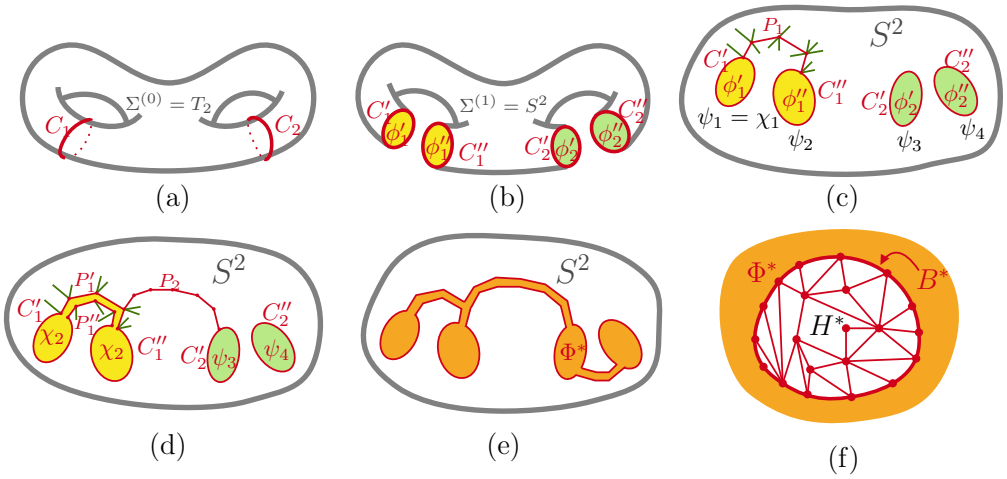


Figure 1: An idealistic scenario where a graph G embedded on $\mathbb{T}_2$ has disjoint non-separating cycles. In this case, we cut along two disjoint non-separating cycles C_1 and C_2, get four supplementary faces, and merge these faces by cutting along disjoint paths, until we get only one face (which is orange in the drawing). Let us give more details, using the notations of the proof. The supplementary faces are $\phi'_1, \phi''_1, \phi'_2$ and ϕ''_2, and they are renamed ψ_1, ψ_2, ψ_3 and ψ_4 respectively, for convenience. These faces are merged step by step: a duplicated path is used to merge such a face ψ_i with the face χ_{i-1} originating from the merge of the faces ψ_1 to ψ_{i-1} (with $\chi_1 = \psi_1$). Picture (d) illustrates the merge of χ_2 with ψ_3. In the end, we have a single merged face that we call Φ^*. The latter face Φ^* can be seen as the infinite face of the planar graph embedding obtained by these transformations.

1.2.2. Our Techniques

Our proof-labeling schemes are obtained thanks to a local encoding of a mechanism enabling to "unfold" a graph G of genus or non-orientable genus k into a planar graph $\widehat{G}$, by a series of vertex-duplications. Specifically, for graphs of genus k, i.e., embeddable on an orientable surface $\mathbb{T}_k$, we construct a sequences $G^{(0)}, \ldots, G^{(k)}$ where $G^{(0)} = G$, $G^{(k)} = \widehat{G}$, and, for every $i = 0, \ldots, k$, $G^{(i)}$ has genus $k - i$. For $i \geq 1$, the graph $G^{(i)}$ is obtained from $G^{(i-1)}$ by identifying a non-separating cycle C_i in $G^{(i-1)}$, and duplicating the vertices and cycles of C_i (see Figure 1(a-b)). (Recall that a non-separating cycle, is a cycle that can be removed without making the graph disconnected.) This enables to "cut" a handle of the surface $\mathbb{T}_{k-i+1}$, resulting in a closed surface $\mathbb{T}_{k-i}$ with genus one less than $\mathbb{T}_{k-i+1}$, while the embedding of $G^{(i-1)}$ on $\mathbb{T}_{k-i+1}$ induces an embedding of $G^{(i)}$ on $\mathbb{T}_{k-i}$. The graph $\widehat{G}$ is planar, and has $2k$ special faces $\phi'_1, \phi''_1, \ldots, \phi'_k, \phi''_k$, where, for $i = 0, \ldots, k$, the faces ϕ'_i

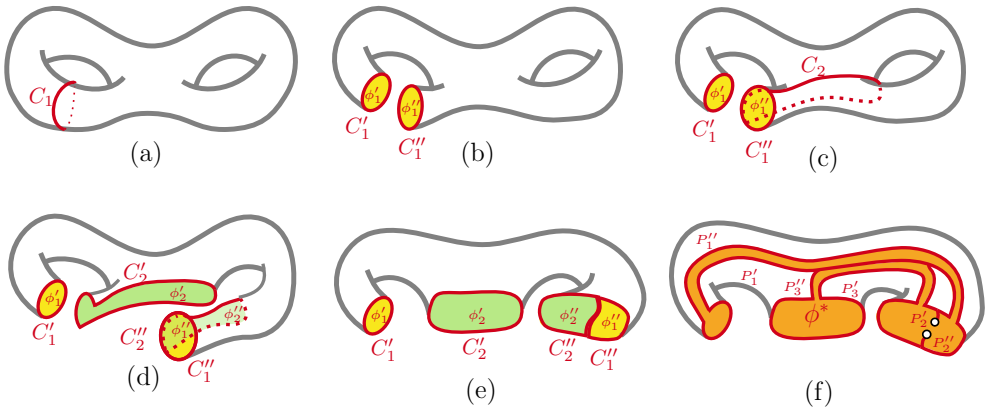


Figure 2: A more complex unfolding a graph G embedded on $\mathbb{T}_2$. Faces created by duplications have not disjoint boundaries, and parts of previous duplicated paths are used to create new paths.

and ϕ_i'' results from the duplication of the face C_i (see Figure 1(c)).

The proof-labeling scheme needs to certify not only the planarity of $\widehat{G}$, but also the existence of the faces $\phi_1', \phi_1'', \ldots, \phi_k', \phi_k''$, and a proof that they are indeed faces, which is non-trivial. Therefore, instead of keeping the $2k$ faces as such, we connect them by a sequence of paths $P_1, \ldots, P_{2k-1}$. By duplicating each path P_i into P_i' and P_i'', the two faces χ and ψ connected by a path P_i is transformed into a single face, while planarity is preserved. Intuitively, the new face is the "union" of χ, ψ, and the "piece in between" P_i' and P_i'' (see Figure 1(d)). The whole process eventually results in a planar graph H with a single special face ϕ (see Figure 1(e-f)). In fact, the paths P_i, $i = 1, \ldots, 2k-1$ do not only serve the objective of merging the $2k$ faces $\phi_1', \phi_1'', \ldots, \phi_k', \phi_k''$ into a single face ϕ, but also serve the objective of keeping track of consistent orientations of the boundaries of these faces. The purpose of these orientations is to provide the nodes with the ability to locally check that the $2k$ faces can indeed be paired for forming k handles.

The planarity of H and the existence of the special face ϕ can be certified by a slight adaptation of the proof-labeling scheme for planarity in [25]. It then remains to encode the sequence of cycle and path-duplications locally, so that every node can roll back the entire process, for identifying the cycles C_i, $i = 1, \ldots, k$, and the paths P_j, $j = 1, \ldots, 2k-1$, and for checking their correctness.

Among many issues, a very delicate problem is that, as opposed to cycles

and paths drawn on a surface, which can be chosen to intersect at few points, these cycles and paths are in graphs embedded on surfaces, and thus they may intersect a lot, by sharing vertices or even edges. Figure 1 displays an idealistic scenario in which the cycles C_i's are disjoint, the paths P_j's are disjoint, and these cycles and paths are also disjoint. However, this does not need to be the case, and the considered cycles and paths may mutually intersect in a very intricate manner. For instance, Figure 2 displays a case in which C_2 intersects with C_1'', P_2' and P_2'' are reduced to single vertices, and P_3'' intersects with P_1'. It follows that the sequence of duplications may actually be quite cumbersome in general, with some nodes duplicated many times. As a consequence, keeping track of the boundaries of the faces is challenging, especially under the constraint that all information must be distributed, and stored at each node using $O(\log n)$ bits only. Also, one needs to preserve specific orientations of the boundaries of the faces, for making sure that not only the two faces ϕ_i' and ϕ_i'' corresponding to a same cycle C_i can be identified, but also that they can be glued together appropriately in a way resulting to a handle, and not be glued like, e.g., a Klein bottle.

The case of graphs embedded on a non-orientable closed surface causes other problems, including the local encoding of the cross-caps, and the fact that decreasing the genus of a non-orientable closed surface by removing a cross-cap may actually result in a closed surface that is orientable. Indeed, eliminating cross-caps is based on *doubling* a non-orientable cycle of the graph, and this operation may result in a graph embedded on a surface that is actually orientable. (This phenomenon did not arise in the case of orientable surfaces, as removing a handle from an orientable closed surface by cycle-duplication results in a graph embedded on an orientable closed surface.) Thus, the proof-labeling scheme for bounded non-orientable genus has to encode not only the identification the cross-caps, but also of faces to be identified for forming handles.

For guaranteeing certificates on $O(\log n)$ bits, our proof-labeling schemes distribute the information evenly to the certificates provided to the nodes, using the fact that graphs of bounded (orientable or non-orientable) genus have bounded *degeneracy*. This property enables to store certificates on $O(\log n)$ bits at each node, even for nodes that have arbitrarily large degrees.

We complete this brief summary of our techniques with two remarks.

On the cuts. Modifying a graph of bounded genus by performing a sequence of cuts for eventually producing a planar graph has already been

used in the literature — see, e.g., [36], where a probabilistic embedding of bounded genus graphs into planar graphs is designed. However, using this techniques in the framework of distributed computing is, to our knowledge, new, and poses additional challenges. In particular, dealing with the intersections of the aforementioned paths and cycles is not much difficult in centralized computing (typically, few virtual nodes may be added, or the graph may even be triangulated, for avoiding intersections), but the techniques used in the centralized setting do not carry over easily to the decentralized setting. More generally, the fact that every step of the transformation of a graph of bounded (non-orientable) genus into a planar graph must be verifiable locally in a distributed manner imposes strong constraints, and restricts the set of techniques that can be used. As a consequence, we could not pick a transformation from the shelf for using it as a black box, but we had to come up with a specific one, bearing close similarities with existing ones, for carefully monitoring every step of it, and checking the ability to implement this step in a distributed manner using small certificates.

On the general approach. An approach conceptually simpler than the one used in this paper would have been to use induction, simply assuming the existence of a proof-labeling scheme with $O(\log n)$-bit certificates for graphs with (non-orientable) genus $k \geq 0$, and then constructing a proof-labeling scheme with $O(\log n)$-bit certificates for graphs with (non-orientable) genus $k + 1$. However, although we are not claiming that such a desirable and conceptually simpler approach is impossible to use, we strongly believe that it may simply be not the right approach, for at least two reasons. First, the proof-labeling scheme in [25] certifies planarity, but it does not provide a way to certify all the faces of a planar embedding. Indeed, it only provides a certification for a single specific face, namely the outer-face. Given an arbitrary cycle in the graph, certifying that this cycle corresponds to the boundary of a face in the planar embedding is not provided in [25], and the design of a compact proof-labeling scheme for this property appears to be non trivial. Note in particular that we do not assume that the graphs we manipulate have unique embeddings, thus a cycle might be a face in one embedding, but not in another embedding. Second, even if the previous problem could be solved, it would remain that the orientations of the pair of faces to be merged at each level of the

induction are crucial. One needs to make sure that the nodes can locally check that the orientations provided by the non-trustable prover are correct, for distinguishing handles from cross-caps. The inductive design of a compact proof-labeling scheme for this property appears to be even more challenging. These two issues are serious obstacles to the development of an inductive construction, and, to overcome them, we adopted the approach consisting to "unwrap" the whole construction. This is less elegant and comprehensible than an inductive construction, but this allowed us to (1) identify a single face in the planar embedding, which can be certified (e.g., using [25]), and (2) provide an orientation to the boundary of this face, for enabling to certify that the orientation given by the non-trustable prover to each and every pair of faces to be merged is indeed correct.

1.3. Related Work

Bounded-degree graphs form one of the most popular class of sparse graphs studied in the context of design and analysis of distributed algorithms, as witnessed by the large literature (see, e.g., [50]) dedicated to construct *locally checkable labelings* (e.g., vertex colorings, maximal independent sets, etc.) initiated a quarter of a century ago by the seminal work in [46]. Since then, other classes of sparse graphs have received a lot of attention, including planar graphs, and graphs of bounded genus. In particular, there is a long history of designing distributed approximation algorithms for these classes, exemplified by the case of the minimum dominating set problem. One of the earliest result for this latter problem is the design of a constant-factor approximation algorithm for planar graphs, performing in a constant number of rounds [41]. This result is in striking contrast with the fact that even a poly-logarithmic approximation requires at least $\Omega(\sqrt{\log n / \log\log n})$ rounds in arbitrary n-node networks [40]. The paper [41] has paved the way for a series of works, either improving on the complexity and the approximation ratio [16, 42, 54], or using weaker models [55], or tackling more general problems [14, 15], or proving lower bounds [35, 16]. The minimum dominating set problem has then been studied in more general classes such as graphs with bounded arboricity [42], minor-closed graphs [13], and graphs with bounded expansion [3]. Specifically, for graphs with bounded genus, it has been shown that a constant approximation can be obtained in time $O(k)$ for graphs of genus k [4], and a $(1+\epsilon)$-approximation algorithm has recently been designed, performing in time $O(\log^* n)$ [5].

Several other problems, such as maximal independent set, maximal matching, etc., have been studied for the aforementioned graph classes, and we refer to [21] for an extended bibliography. In addition to the aforementioned results, mostly dealing with local algorithms, there are recent results in computational models taking into account limited link bandwidth, for graphs that can be embedded on surfaces. For instance, it was shown that a combinatorial planar embedding can be computed efficiently in the CONGEST model [31]. Such an embedding can then be used to derive more efficient algorithms for minimum-weight spanning tree, min-cut, and depth-first search tree constructions [32, 33]. Finally, it is worth mentioning that, in addition to algorithms, distributed data structures have been designed for graphs embedded on surfaces, including a recent optimal adjacency-labeling for planar graphs [8, 17], and routing tables for graphs of bounded genus [30] as well as for graphs excluding a fixed minor [1].

Proof-labeling schemes (PLS) were introduced in [39], and different variants were later introduced. Stronger forms of PLS include *locally checkable proofs* (LCP) [34] in which nodes forge their decisions on the certificates and on the whole states of their neighbors, and t-PLS [24] in which nodes perform communication at distance $t \geq 1$ before deciding. Weaker forms of PLS include *non-deterministic local decision* (NLD) [26] in which the certificates must be independent from the identity-assignment to the nodes. PLS were also extended by allowing the verifier to be randomized (see [29]). Such protocols were originally referred to as *randomized PLS* (RPLS), but are nowadays referred to as distributed Merlin-Arthur (dMA) protocols.

The same way NP is extended to the complexity classes forming the Polynomial Hierarchy, by alternating quantifiers, PLS were extended to a hierarchy of distributed decision classes [7, 23], which can be viewed as resulting from a game between a prover and a *disprover*. Recently, *distributed interactive proofs* were formalized [38], and the classes $\mathsf{dAM}[k](f(n))$ and $\mathsf{dMA}[k](f(n))$ were defined, where $k \geq 1$ denotes the number of alternations between the centralized Merlin and the decentralized Arthur, and $f(n)$ denotes the size of the proof — $\mathsf{dAM}[3](f(n))$ is also referred to as $\mathsf{dMAM}(f(n))$. Distributed interactive protocols for problems like the existence of a non-trivial automorphism (AUT), and non-isomorphism ($\overline{\mathsf{ISO}}$) were designed and analyzed in [38]. The follow up paper [45] improved the complexity of some of the protocols in [38], either in terms of the number of interactions between the prover and the verifier, and/or in terms of the size of the certificates. A sophisticated generic way for constructing distributed IP protocols based on

sequential IP protocols is presented in [45]. One of the main outcome of this latter construction is a dMAM protocol using certificates on $O(\log n)$ bits for all graph classes whose membership can be decided in linear time. For other recent results on distributed interactive proof, see [12, 27].

It is worth noticing that a very recent paper [19] provides an alternative proof of the results of this paper, by certifying (i) the faces of the embedding using the Heffter-Edmonds-Ringel rotation principle, and (ii) the genus of the embedding using Euler's Formula.

1.4. Organization of the Paper

The next section provides the reader with basic notions regarding graphs embedded on closed surfaces, and formally defines our problem. Section 3 describes how to "unfold" a graph G of genus k, for producing a planar graph H with a special face ϕ. The section also describes how, given a planar graph H with a special face ϕ, one can check that (H, ϕ) results from the unfolding of a graph G with genus k. Then, Section 4 presents our first main result, that is, a proof-labeling scheme for the class of graphs with bounded genus. In particular, it describes how to encode the description of the pair (H, ϕ) from Section 3, and, more importantly, how to locally encode the whole unfolding process in a distributed manner, using certificates on $O(\log n)$ bits, which allow the nodes to collectively check that their certificates form a proof that G has genus k. Section 5 presents our second main result, by showing how to extend the proof-labeling scheme of Section 4 to the class of graphs with bounded non-orientable genus. Finally, Section 6 concludes the paper with a discussion about the obstacles to be overcame for the design of a proof-labeling scheme for the class of graphs excluding a fixed minor.

2. Definitions, and Formal Statement of the Problem

This section contains a brief introduction to graphs embedded on surfaces, and provides the formal statement of our problem.

2.1. Closed Surfaces

Most of the notions mentioned in this section are standard, and we refer to, e.g., Massey et al. [43] for more details.

2.1.1. Definition

Recall that a *topological space* is a pair (X, T) where X is a set, and T is a topology on X (e.g., T is a collection of subsets of X, whose elements are called *open* sets, satisfying the following properties : the set X and the empty set are open, any finite intersection of open sets is an open set, and any arbitrary union of open sets is an open set). A topological space may be denoted by X if there is no ambiguity about the topology on X. Also recall that a topological space X is *compact* if, from any set of open sets whose union is X, one can extract a finite set of open sets whose union is finite. A function $f : X \to Y$ between two topological spaces is *continuous* if the inverse image of every open set in Y is open in X. A *homeomorphism* is a bijection that is continuous, and whose inverse is also continuous. A *topological path* in X is a continuous function $P : [0, 1] \to X$. The space X is *path-connected* if for any pairs x, y of points of X, there exists a topological path P such that $P(0) = x$ and $P(1) = y$.

Definition 1. A *closed surface* Σ is a path-connected[7], compact space that is locally homeomorphic to a disk of $\mathbb{R}^2$, (i.e. for each $x \in \Sigma$, there exists an open set S_x containing x such that S_x is homeomorphic to an closed disk of $\mathbb{R}^2$, the topology T_{S_x} used for S_x being the set $T_{S_x} = \{S \cap S_x, S \in T\}$.

2.1.2. Construction

Some closed surfaces can be obtained by the following construction. Let S^2 be the 2-dimensional sphere. For $k \geq 0$, given $2k$ disks $D_1, D_2, ...D_{2k}$ on the surface of S^2, with pairwise disjoint interiors, let us direct clockwise the boundaries of $D_1, \ldots, D_k$, and let us direct counterclockwise the boundaries of $D_{k+1}, \ldots, D_{2k}$. Next, let us remove the interior of each disk, and, for $1 \leq i \leq k$, let us identify (i.e., glue) the boundary of D_i with the boundary D_{i+k} in such a way that directions coincide (see Figure 3). The resulting topological space is denoted by $\mathbb{T}_k$. In particular, $\mathbb{T}_1$ is the torus, and $\mathbb{T}_0 = S^2$. For every i, identifying D_i and D_{i+k} results in a *handle*. It follows that $\mathbb{T}_k$ contains k handles.

Another family of closed surfaces is constructed as follows. Let $D_1, \ldots, D_k$ be $k \geq 1$ disks with pairwise disjoint interiors. Let us again remove the

[7]Path-connected can actually be replaced by connected (i.e., cannot be partitioned in two open sets) here, because, under the hypothesis of local homeomorphy to a disk, the notions of path-connectivity and connectivity are equivalent.

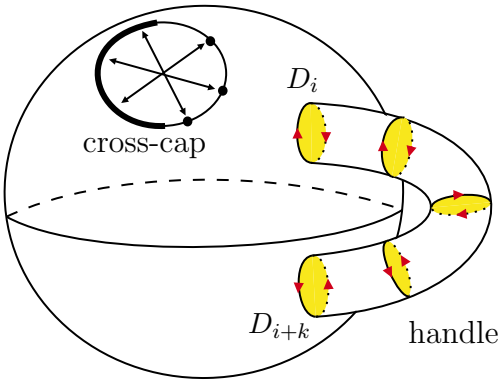


Figure 3: Handles and cross-cap.

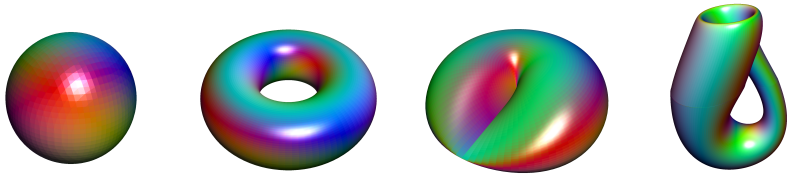


Figure 4: The sphere, torus, projective plane, and Klein Bottle.

interior of each disk. For every $1 \leq i \leq k$, and for every antipodal point v and v' of the boundary of D_i, let us identify (i.e., glue) the points v and v' (see Figure 3). The resulting topological space is denoted by $\mathbb{P}_k$. In particular, $\mathbb{P}_1$ is the projective plane, and $\mathbb{P}_2$ is the Klein bottle ($\mathbb{P}_0$ is not defined). For every i, the operation performed on D_i results in a *cross-cap*. It follows that $\mathbb{P}_k$ contains k cross-caps.

The surfaces resulting from the above constructions can thus be *orientable* (e.g., the sphere $\mathbb{T}_0$ or the torus $\mathbb{T}_1$) or not (e.g., the projective plane $\mathbb{P}_1$ or the Klein Bottle $\mathbb{P}_2$), as displayed on Figure 4.

2.1.3. Orientability

For defining orientability of a closed surface Σ, we use the notion of *curve*, defined as a continuous function $C : S^1 \to \Sigma$, where S^1 denotes the unidimensional sphere (homeomorphic to, e.g., the trigonometric circle). A curve is *simple* if it is injective. A simple curve C is *orientable* if one can define the left side and the right side of the curve at every point of the curve in a consistent manner. Specifically, a curve C is orientable if, for every $x \in C$,

there exists a neighborhood N_x of x such that $N_x \setminus C$ has two connected components, one called the left side $L(N_x)$ of N_x, and the other the right side $R(N_x)$ of N_x, such that, for every $x, x' \in C$ and every $y \in \Sigma$,

$$(y \in N_x \cap N_{x'}) \wedge (y \in L(N_x)) \implies y \in L(N_{x'}).$$

A closed surface Σ is orientable if every simple curve of X is orientable. It is easy to check that orientability is a topological invariant. That is, if Σ and Σ' are two homeomorphic topological spaces, then Σ is orientable if and only if Σ' is orientable.

2.1.4. Genus of a Surface

An orientable closed surface Σ is of *genus* k if it is homeomorphic to a closed surface $\mathbb{T}_k$ constructed as in Section 2.1.2. The Classification Theorem of orientable closed surfaces (see, e.g., [10]) states that every orientable closed surface has a genus. That is, for every orientable surface Σ, there exists a unique $k \geq 0$ such that Σ is of genus k. The fact that every pair of orientable closed surfaces with the same genus k are homeomorphic, justifies that a unique notation can be adopted for these surfaces, and any orientable closed surface of genus k is denoted by $\mathbb{T}_k$. Observe however that two closed surfaces that are homeomorphic are not necessarily homotopic, i.e., they may not be continuously deformable into each other (for instance, the torus is not homotopic to the trefoil knot, although both are homeomorphic).

The genus can also be defined for non-orientable closed surfaces. For $k \geq 1$, a non-orientable closed surface is said to be of *genus* k if it is homeomorphic to a closed surface $\mathbb{P}_k$ constructed as in Section 2.1.2. Again, the Classification Theorem of non-orientable closed surfaces (see, e.g., [10]) states that every non-orientable closed surface has a genus. That is, for every non-orientable closed surface Σ, there exists a unique $k \geq 0$ such that Σ is of genus k. As for orientable surfaces, every pair of non-orientable closed surfaces of genus k are homeomorphic, and a non-orientable closed surface of genus k is denoted by $\mathbb{P}_k$.

2.2. Graphs Embedded on Surfaces

In this section, we recall standard notions related to graph embeddings on surfaces, and we refer to Mohar and Thomassen [44] for more details. Throughout the paper, all considered graphs are supposed to be simple (no multiple edges, and no self-loops), and connected.

2.2.1. Topological Embeddings

Given a graph $G = (V, E)$, and a closed surface Σ, a *topological embedding* of G on Σ is given by (1) an injective mapping $f : V \to \Sigma$, and, (2) a topological path $f_e : [0, 1] \to \Sigma$ defined for every edge e such that:

- if $e = \{v, v'\} \in E$, then $f_e(\{0, 1\}) = \{f(v), f(v')\}$, and
- if $e, e' \in E$ and $e \neq e'$, then $f_e(]0, 1[) \cap f_{e'}(]0, 1[) = \varnothing$.

The second condition is often referred to as the *non-crossing* condition. See Figure 5 for two embeddings of the complete graph K_4 on $\mathbb{T}_1$. Throughout the paper, we may identify a vertex v with its representation $f(v)$, and an edge e with its representation f_e (i.e., the image $f_e([0, 1])$ of $[0, 1]$ by f_e), even referred to as $f(e)$ in the following. The set $\cup_{e \in E} f(e)$ is called the *skeleton* of the embedding, and is denoted by $\mathrm{Sk}(G)$. Each connected component of $\Sigma \setminus \mathrm{Sk}(G)$ is an open set of Σ (as complement of a closed set), called a *face* of the embedding. In fact, in this paper, we will abuse notation, and often refer to G instead of $\mathrm{Sk}(G)$ when referring to the embedding of G on Σ.

2.2.2. 2-Cell Embeddings

We now recall a slightly more sophisticated, but significantly richer form of topological embedding, called *2-cell embedding*. A 2-cell embedding is a topological embedding such that every face is homeomorphic to an open disk of $\mathbb{R}^2$.

In a 2-cell embedding of a graph G, the border of a face can be described by giving a so-called *boundary (closed) walk*, that is, an ordered list $(v_0, \ldots, v_r)$ of non-necessarily distinct vertices of G, where, for $i = 0, \ldots, r-1$, $\{v_i, v_{i+1}\} \in E(G)$, and $\{v_r, v_0\} \in E(G)$. The vertices and edges of a face are the images by the embedding of the vertices and edges of the boundary walk. The boundary walk is however not necessarily a simple cycle, as an edge may appear twice in the walk, once for each direction, and a vertex may even appear many times.

For instance, Figure 5 displays two embeddings of the complete graph K_4 on the torus $\mathbb{T}_1$. The embedding on the left is not a 2-cell embedding. Indeed, this embedding results in three faces, including the two faces with boundary walk (a, b, c) and (a, b, d). The third face is however not homeomorphic to an open disk (there is a hole in it, resulting from the hole in the torus). On the other hand, the embedding on the right in Figure 5 is a 2-cell embedding. Indeed, there are two faces, including the face with boundary walk (a, b, c).

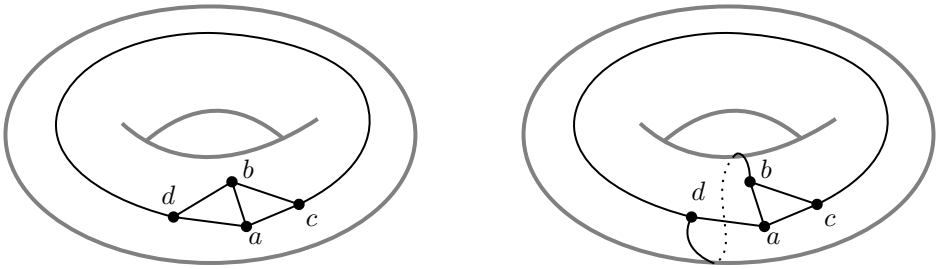


Figure 5: Two embeddings of K_4 on the torus $\mathbb{T}_1$. The one on the left is not a 2-cell embedding since the non-triangular face is not homeomorphic to a closed disk. This situation can occur because K_4 is of genus 0, not 1 (see Lemma 2). The embedding on right is a 2-cell embedding.

The other face is also homeomorphic to an open disk. A boundary walk of this latter face is $(d, a, b, d, c, a, d, b, c)$. This can be seen by starting from d, traversing the edge $\{d, a\}$, and adopting the "left-hand rule" when entering a vertex, leading from a to b, then back to d, next to c, etc. Notice that this boundary walk uses some edges twice. It follows that the closure of a face is not necessarily homeomorphic to a closed disk, even in a 2-cell embedding.

We complete the section with an observation, which allows us to restrict our attention to cycles in graphs instead of arbitrary curves in topological spaces. It also illustrates the interest of 2-cell embeddings (the result does not necessarily hold for arbitrary embeddings, as illustrated by the embedding on the left of Figure 5). In the following, *contractible* means homotopic to a point.

Lemma 1. *For every graph G, and every closed surface Σ, any 2-cell embedding of G on Σ satisfies that every closed curve in Σ is either contractible, or homotopic to a closed cycle of* $\mathrm{Sk}(G)$.

The rough reason why the result holds is that, in a 2-cell embedding, any sub-path of a path traversing a face can be replaced by a sub-path following the border of the face. (This is not necessarily true for a general embedding).

2.2.3. Genus and non-orientable genus of a Graph

For any graph G, there exists $k \geq 0$ such that G can be embedded on $\mathbb{T}_k$, as any embedding of G in the plane with x pairs of crossing edges induces an embedding of G on $\mathbb{T}_x$ without crossings, by replacing each crossing with a handle. Also, if G can be embedded on $\mathbb{T}_k$, then G can be embedded on

$\mathbb{T}_{k'}$ for every $k' \geq k$. The *genus* of a graph G is the smallest k such that there exists an embedding of G on $\mathbb{T}_k$. Similarly, the *non-orientable* genus, or *Euler genus* of G, is defined as the smallest k such that there exists an embedding of G on $\mathbb{P}_k$.

The embeddings of graphs of genus k on $\mathbb{T}_k$ have a remarkable property (see, e.g., [56]).

Lemma 2. *Every embedding of a graph G of genus k on $\mathbb{T}_k$ is a 2-cell embedding.*

The same property does not necessarily hold fo graphs with bounded non-orientable genus. However, some weaker form of Lemma 2 can be established (see, e.g., [49]).

Lemma 3. *For every graph G of non-orientable genus k, there exists a 2-cell embedding of G on $\mathbb{P}_k$.*

The next result is extremely helpful for computing the genus of a graph, and is often referred to as the Euler-Poincaré formula [52].

Lemma 4. *Let $G = (V, E)$, and let Σ be a closed surface of genus k. Let us consider any 2-cell embedding of G on Σ, and let F be the set of faces of this embedding. If Σ is orientable then $|V| - |E| + |F| = 2 - 2k$. If Σ is non orientable then $|V| - |E| + |F| = 2 - k$.*

Recall that, for $d \geq 0$, a graph G is *d-degenerate* if every subgraph of G has a node of degree at most d. Degeneracy will play a crucial role later in the paper, for evenly distributing the information to be stored in the certificates according to our proof-labeling schemes.

Graphs with bounded genus have bounded degeneracy (see, e.g., [44] Theorem 8. 3. 1, this result is due to Heawood), as recalled below for further references.

Lemma 5. *For every $k \geq 0$, every graph of genus at most k is d-degenerate with $d = \max(5, \frac{5+\sqrt{1+48k}}{2})$.*

For every $k \geq 1$, every graph of non-orientable genus at most k is d-degenerate with $d = \max(5, \frac{5+\sqrt{1+24k}}{2})$.

2.3. Formal Statement of the Problem

Proof-Labeling Schemes (PLS) are distributed mechanisms for verifying graph properties. More precisely, let $\mathcal{G}$ be a graph family. A PLS for $\mathcal{G}$ is defined as a prover-verifier pair $(\mathbf{p}, \mathbf{v})$, bounded to satisfy the following. Given any graph $G = (V, E)$ whose n vertices are arbitrarily labeled by n distinct identifiers (ID) picked from a set $\{1, \ldots, n^k\}$, $k \geq 1$, of polynomial range, the prover $\mathbf{p}$ is a non-trustable oracle that provides every vertex $v \in V$ with a *certificate* $c(v)$. The verifier $\mathbf{v}$ is a distributed protocol performing a single round in parallel at all vertices, as follows. Every vertex collects the certificates of all its neighbors, and must output "accept" or "reject", on the basis of its ID, its certificate, and the certificates of its neighbors. The pair $(\mathbf{p}, \mathbf{v})$ is a correct PLS for $\mathcal{G}$ if the following two conditions hold.

Completeness: For every $G \in \mathcal{G}$, and for every ID-assignment to the vertices of G, the (non-trustable) prover $\mathbf{p}$ can assign certificates to the vertices such that the verifier $\mathbf{v}$ *accepts* at all vertices;

Soundness: For every $G \notin \mathcal{G}$, for every ID-assignment to the vertices of G, and for every certificate-assignment to the vertices by the non-trustable prover $\mathbf{p}$, the verifier $\mathbf{v}$ *rejects* in at least one vertex.

The main *complexity measure* for a PLS is the size of the certificates assigned to the vertices by the prover. The objective of the paper is to design schemes with logarithmic-size certificates, for two classes of graphs: the class $\mathcal{G}_k^+$, $k \geq 0$, of graphs embeddable on an orientable closed surface of genus at most k (i.e., the graphs of genus $\leq k$), and the class $\mathcal{G}_k^-$, $k \geq 1$, of graphs embeddable on a non-orientable closed surface of genus at most k (i.e., the graphs of non-orientable genus $\leq k$).

Remark.. Throughout the rest of the paper, for $G \in \mathcal{G}_k^+$ (resp., $G \in \mathcal{G}_k^-$) with genus $k' < k$ (resp., non-orientable genus $k' < k$), our proof-labeling scheme certifies an embedding of G on $\mathbb{T}_{k'}$ (resp., on $\mathbb{P}_{k'}$). Therefore, in the following, k is supposed to denote the exact genus of G.

3. Unfolding a Surface

In this section, we describe how to "flat down" a surface, by reducing it to a disk whose boundary has a specific form. This operation is central for constructing the distributed certificates in our proof-labeling scheme. In

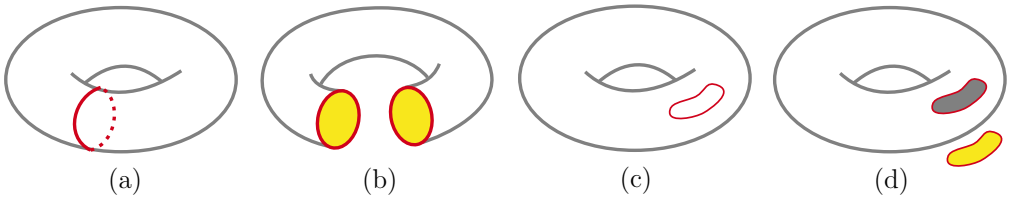


Figure 6: Separating and non-separating cycles.

fact, it provides a centralized certificate for bounded genus. The section is dedicated to orientable surfaces, and the case of non-orientable surfaces will be treated further in the text.

3.1. Separation and Duplication

Given a 2-cell embedding of a graph G on a closed surface Σ, a *non-separating cycle* of the embedding is a simple cycle C in G such that $\Sigma \setminus C$ is connected. Figure 6 illustrates this notion: the cycle displayed on (a) is non-separating, as shown on (b); instead, the cycle displayed on (c) is separating, as shown on (d). The result hereafter is a classic result.

Lemma 6 (Lemma 11 in [48]). *Let G be a graph embeddable on a closed orientable surface Σ with genus $k \geq 1$. For any 2-cell embedding of G on Σ, there exists a non-separating cycle C in G.*

Note that the hypothesis that there is a 2-cell embedding is crucial: a tree can be embedded on any surface, but has no cycle.

3.1.1. Cycle-Duplication

Let G be a graph embeddable on a closed orientable surface Σ. An orientable cycle is a cycle of G whose embedding on Σ yields an orientable curve. Given a 2-cell embedding f of G on Σ, let C be a non-separating orientable cycle of G whose existence is guaranteed by Lemma 6. By definition, the left and right sides of C can be defined on the neighborhood of C. We denote by G_C the graph obtained by the *duplication* of C in G. Specifically, let us assume that $C = (v_0, \ldots, v_r)$. Every vertex $w \notin C$ remains in G_C, as well as every edge non incident to a vertex of C. Every vertex v_i of C is replaced by a *left* vertex v'_i and a *right* vertex v''_i. For every $i = 0, \ldots, r-1$, $\{v'_i, v'_{i+1}\}$ and $\{v''_i, v''_{i+1}\}$ are edges of G_C, as well as $\{v'_r, v'_0\}$ and $\{v''_r, v''_0\}$. Finally, for every $i = 0, \ldots, r$, and every neighbor $w \notin C$ of v_i in G, if $f(\{v_i, w\})$ meets

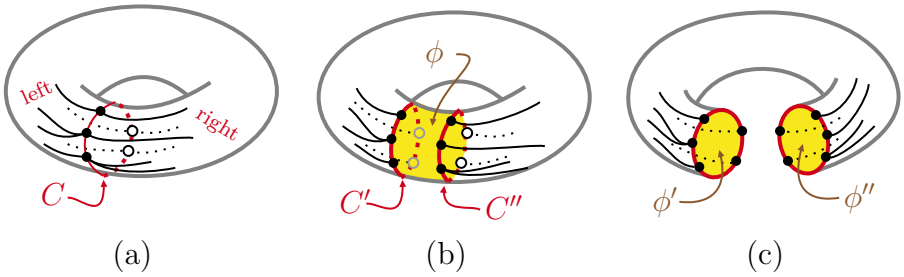


Figure 7: Cycle-duplication and the associated surface.

the left of C, then $\{v'_i, w\}$ is an edge of G_C, otherwise $\{v''_i, w\}$ is an edge of G_C. The embedding f of G on Σ directly induces an embedding of G_C on Σ. Figure 7(a-b) illustrates the operation of duplication, and the resulting embedding on Σ.

The embedding of G_C on Σ is however not a 2-cell embedding, as it contains the face ϕ between C' and C'' on Σ, where $C' = (v'_0, \ldots, v'_r)$ and $C'' = (v''_0, \ldots, v''_r)$ (see Figure 7(b)). Formally, ϕ is the face with boundaries C' and C'', and, as such, it is not homeomorphic to a disc. Let Σ_C be the closed surface[8] obtained from Σ by removing ϕ, and by replacing ϕ with two faces ϕ' and ϕ'' with boundary walks C' and C'', respectively (see Figure 7(c)). The embedding f of G on Σ induces a 2-cell embedding f_C of G_C on Σ_C. Also, since C is a non-separating cycle of G in Σ, the surface Σ_C is path-connected, which ensures that G_C is connected using Lemma 1.

Moreover, as Σ is orientable, Σ_C is also orientable. Indeed, every simple cycle of Σ_C not intersecting ϕ' nor ϕ'' is a cycle of Σ, and is therefore orientable. Furthermore, any simple cycle of Σ_C intersecting ϕ' and/or ϕ'' is homotopic to a cycle separated from both boundaries of ϕ' and ϕ'' by an open set, and thus is homotopic to a cycle of Σ. It follows that Σ_C is a closed orientable surface, and thus, thanks to Lemma 4, the genus of Σ_C is $k - 1$.

3.1.2. Path-Duplication

Again, let us consider a graph G, an orientable closed surface Σ, and a 2-cell embedding f of G on Σ. Let χ, ψ be two distinct faces of the embedding, and let $P = (w_0, \ldots, w_s)$ be a simple path (possibly reduced to a single vertex

[8]Notice that $X \backslash \phi$, $\overline{\phi'} = \phi' \cup C'$ (where $\overline{\phi'}$ denotes the adherence of ϕ'), and $\overline{\phi''} = \phi'' \cup C''$ are compact sets. Thus Σ_C is compact as the union of these three sets.

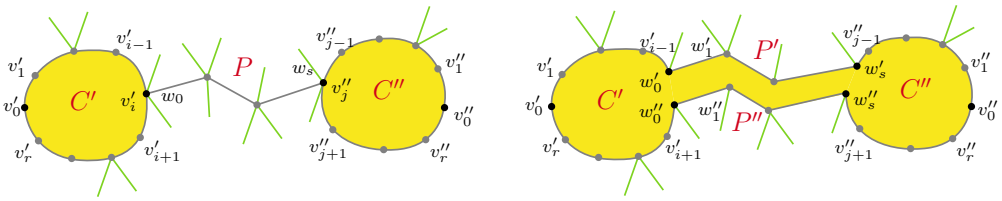


Figure 8: Path-duplication.

belonging to the two cycles) between χ and ψ (see Figure 8). That is, P is such that w_0 is on the boundary of ϕ, w_s is on the boundary of ψ, and no intermediate vertex w_i, $0 < i < s$, is on the boundary of χ or ψ. The path P enables to define a graph G_P obtained by duplicating the path P in a way similar to the way the cycle C was duplicated in the previous section. There is only one subtle difference, as the left and right side of the path cannot be defined at its endpoints. Nevertheless, the left and right sides of P can still be properly defined all along P, including its extremities, by virtually "extending" P so that it ends up in the interiors of χ and ψ. Thanks to this path-duplication, the two faces χ and ψ of G are replaced by a unique face of G_P as illustrated on Figure 8, reducing the number of faces by one.

Remark.. Cycle-duplication and path-duplication are typically used conjointly. A basic example, used for the torus $\mathbb{T}_1$ in the next section, consists of, first, duplicating a cycle C, then connecting the two faces resulting from this duplication by a path P, and, finally, duplicating P for merging these two faces into one single face. Further, for the general case $\mathbb{T}_k$, $k \geq 1$, k cycles $C_1, \ldots, C_k$ are duplicated, and $2k-1$ paths $P_1, \ldots, P_{2k-1}$ are duplicated for connecting the $2k$ faces $\phi'_1, \phi''_1, \ldots, \phi'_k, \phi''_k$ resulting from the k cycle-duplications, ending up in a unique face ϕ^*.

3.2. Unfolding the Torus

As a warm up, we consider the case of a graph embedded on the torus $\mathbb{T}_1$, and show how to "unfold" this embedding.

3.2.1. Making a Graph of Genus 1 Planar

Let G be a graph, and let f be a 2-cell embedding of G on $X = \mathbb{T}_1$ — see Figure 9(a) for an embedding of K_5 on $\mathbb{T}_1$, as an illustrative example. Let $C = (v_0, \ldots, v_r)$ be a non-separating orientable cycle of G, e.g., the cycle

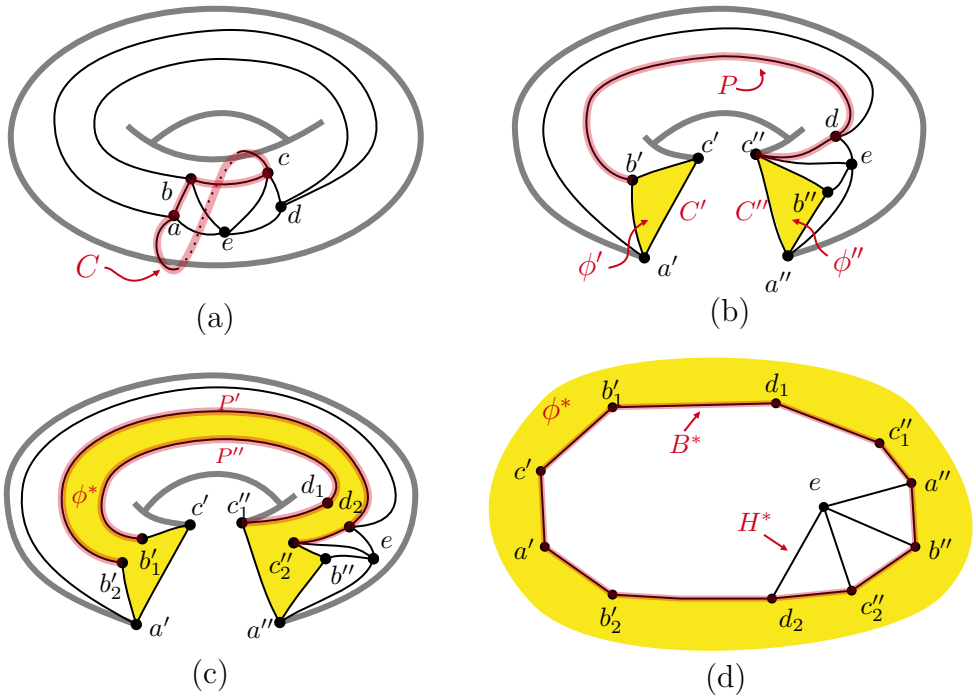


Figure 9: Unfolding K_5 embedded on the torus $\mathbb{T}_1$. After the duplication of the non-separating cycle $C = (a, b, c, a)$ we create the faces ϕ' and ϕ''. Then the duplication of the path $P = (c'', d, b')$ merges ϕ' and ϕ'' into a face ϕ^*. The resulting graph is planar, and ϕ^* can be seen as the infinite face of its embedding.

(a, b, c) on Figure 9(a). Let $C' = (v'_0, \ldots, v'_r)$ and $C'' = (v''_0, \ldots, v''_r)$ be the two cycles resulting from the duplication of C, e.g., the cycles (a', b', c') and (a'', b'', c'') on Figure 9(b). The graph G_C with two new faces ϕ' and ϕ'' is connected. In particular, there exists a simple path $P = (w_0, \ldots, w_s)$ in G_C from a vertex $v'_i \in C'$ to a vertex $v''_j \in C''$, such that every intermediate vertex w_k, $0 < k < s$, is not in $C' \cup C''$, e.g., the path (c'', d, b') on Figure 9(b). Note that it may be the case that $i \neq j$. On Figure 9(b), the path (b'', e, d, b') satisfies $i = j$, but Figure 10 illustrates an embedding of $K_{3,3}$ on $\mathbb{T}_1$ for which $i = j$ cannot occur (simply because every vertex of $K_{3,3}$ has degree 3, and thus it has a single edge not in the cycle). Duplicating P enables to obtain a graph $G_{C,P}$ with a special face ϕ^*, whose boundary contains all duplicated vertices and only them (see Figure 9(c)). The details of the vertex-duplications, and of the edge-connections are detailed hereafter.

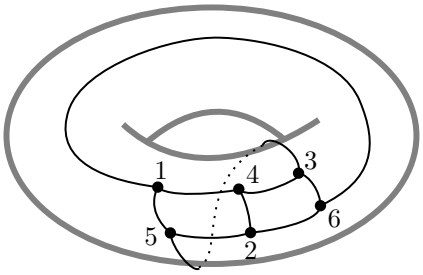


Figure 10: $K_{3,3}$ embedded on the torus $\mathbb{T}_1$.

Connections in path-duplication.. Let $P' = (w'_0, \ldots, w'_s)$ and $P'' = (w''_0, \ldots, w''_s)$ be the two paths obtained by duplicating P. In particular, the vertices $w_0 = v'_i$ and $w_s = v''_j$ are both duplicated in w'_0, w''_0, and w'_s, w''_s, respectively. The edges

$$\{v'_{i-1}, v'_i\},\ \{v'_i, v'_{i+1}\},\ \{v''_{j-1}, v''_j\},\ \text{and}\ \{v''_j, v''_{j+1}\}$$

are replaced by the edges connecting $v'_{i-1}, v'_{i+1}, v''_{j-1}, v''_{j+1}$ to w'_0, w''_0, w'_s, w''_s. For defining these edges, observe that the path P in $\mathbb{T}_1$ induces a path $Q = (v_i, w_1, \ldots, w_{s-1}, v_j)$ in G connecting the vertices v_i and v_j of C, such that, in the embedding on $\mathbb{T}_1$, the edge $\{v_i, w_1\}$ meets C on one side while the edge $\{w_{s-1}, v_j\}$ meets C on the other side (see Figure 11(a-b)).

Figure 11(b) Let us assume, w.l.o.g., that the edges of $C \cup Q$ around v_i are in the order

$$\{v_i, v_{i-1}\},\ \{v_i, v_{i+1}\},\ \{v_i, w_1\}$$

when visited counter-clockwise in $\mathbb{T}_1$. It follows that the edges of $C \cup Q$ around v_j are in the order

$$\{v_j, v_{j-1}\}, \{v_j, v_{j+1}\}, \{v_j, w_{s-1}\}$$

when visited clockwise in $\mathbb{T}_1$ (see Figure 11(b)). These orders are transferred in G_C, that is, the edges of $C' \cup P$ around v'_i are in counter-clockwise order

$$\{v'_i, v'_{i-1}\}, \{v'_i, v'_{i+1}\}, \{v'_i, w_1\},$$

while the edges of $C'' \cup P$ around v''_j are in clockwise order

$$\{v''_j, v''_{j-1}\}, \{v''_j, v''_{j+1}\}, \{v''_j, w_{s-1}\},$$

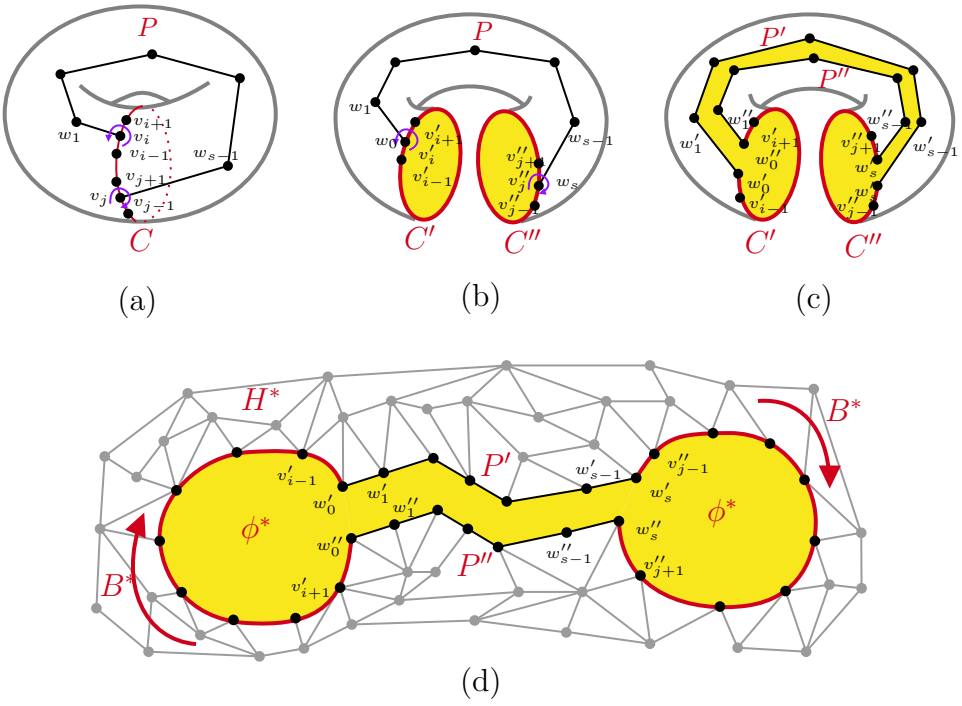


Figure 11: Setting up the connections in path-duplication. A key point, is that v_i is surrounded by the triple (v_{i-1}, v_{i+1}, w_1) counter-clockwise while v_j is surrounded by $(v_{j-1}, v_{j+1}, w_{s-1})$ clockwise. This allows to know where P' and P'' are respectively branched on cycles C' and C''.

as illustrated on Figure 11(b). This guarantees that v'_{i-1} and v''_{j-1} are in the same side of the path P. More generally, the relative positions of v'_{i-1}, v'_{i+1}, v''_{j-1}, and v''_{j+1} w.r.t. P are as follows. Vertices v'_{i-1} and v''_{j-1} are on the same side of P, while vertices v'_{i+1} and v''_{j+1} are on the other side of P (see again Figure 11(b)). As a consequence, it can be assumed that, in the graph $G_{C,P}$ resulting from the duplications of both C and P, the vertices v'_{i-1} and v''_{j-1} are connected to the end points of P', while v'_{i+1} and v''_{j+1} are connected to the end points of P''. It follows that

$$\{w'_0, v'_{i-1}\}, \{w'_s, v''_{j-1}\}, \{w''_0, v'_{i+1}\}, \text{and } \{w''_s, v''_{j+1}\}$$

are edges of $G_{C,P}$ (see Figure 11(c)).

Unfolding.. The embedding f of G on $X = \mathbb{T}_1$ directly induces an embedding of $H^* = G_{C,P}$ on Σ_C, as illustrated on Figure 11(d). As observed before, the

genus of Σ_C is one less than the genus of Σ. Since $X = \mathbb{T}_1$, it follows that the embedding f of G on $\mathbb{T}_1$ actually induces a planar embedding f^* of H^*. The faces of this embedding are merely the faces of G, plus another, special face ϕ^* whose boundary walk is

$$B^* = (w'_0, w'_1, \ldots, w'_s, v''_{j-1}, v''_{j-2}, \ldots, v''_0, v''_r, \ldots, v''_{j+1}, \\ w''_s, w''_{s-1}, \ldots, w''_0, v'_{i+1}, v'_{i+2}, \ldots, v'_r, v'_0, \ldots, v'_{i-1}), \quad (1)$$

as displayed on Figure 11(d)). For instance, on Figure 9(d), $B^* = (b'_1, d_1, c''_1, a'', b'', c''_2, d_2, b'_2, a', c')$. The face ϕ^* can be pointed out as special, as on Figure 11(d), or can be made the external face of the embedding of H^*, as on Figure 9(d). Our interest for H^*, f^*, ϕ^*, and B^* as far as the design of a proof-labeling scheme is concerned, resides in the fact that, as shown hereafter, they form a (centralized) certificate for genus 1.

3.2.2. Certifying Genus 1

Let us first define the notion of *splitting*.

Definition 2. A *splitting* of a graph G into a graph H is a pair $\sigma = (\alpha, \beta)$ of functions, where $\alpha : V(G) \to 2^{V(H)}$, and $\beta : E(G) \to 2^{E(H)}$, such that:

1. the set $\{\alpha(v) : v \in V(G)\}$ forms a partition of $V(H)$;
2. for every $e = \{u, v\} \in E(G)$, $\beta(e)$ is a matching between $\alpha(u)$ and $\alpha(v)$.

Note that $\sigma(G)$ may not be connected, even if G is connected. For every $v \in V(G)$, the vertices $\alpha(v)$ in H are the *avatars* of v in H. The *degree* of a splitting $\sigma = (\alpha, \beta)$ of G into H is $\max_{v \in V(G)} |\alpha(v)|$, and H is said to be a d-splitting of G whenever $d = \max_{v \in V(G)} |\alpha(v)|$. A vertex $v \in V(G)$ is split in H if $|\alpha(v)| \geq 2$, otherwise it is not split in H. If a vertex v is not split, we abuse notation by writing $\alpha(v) = v$, i.e., by referring to v as a vertex of G and as a vertex of H. For any subgraph G' of G, we denote by $\sigma(G')$ the subgraph H' of H with vertex-set $V(H') = \{\alpha(v) : v \in V(G')\}$, and with edge-set $E(H') = \cup_{e \in E(G')} \beta(e)$. With a slight abuse of notation, for a splitting $\sigma = (\alpha, \beta)$ of G into H, we often refer to $\sigma(v)$ instead of $\alpha(v)$ for $v \in V(G)$, and to $\sigma(e)$ instead of $\beta(e)$ for $e \in E(G)$.

Let H be a splitting of a graph G for which there exists a 2-splittting U of G such that H is a 2-splitting of U. Let f be a planar embedding of H, and let ϕ be a face of H embedded on $\mathbb{T}_0$. Let $B = (u_0, \ldots, u_N)$ be a boundary

walk of ϕ. Let $\sigma_{G,U}$ and $\sigma_{U,H}$ be the splitting of G into U, and the splitting of U into H, respectively. Let $\sigma_{G,H} = \sigma_{U,H} \circ \sigma_{G,U}$. We say that (G, H, B, U) is *globally consistent* if there exist vertices $v'_0, \ldots, v'_r$, $v''_0, \ldots, v''_r$, $w'_0, \ldots, w'_s$, $w''_0, \ldots, w''_s$ of H such that

$$B = (w'_0, \ldots, w'_s, v''_{j-1}, \ldots, v''_0, v''_r, \ldots, v''_{j+1}, w''_s, \ldots, w''_0, v'_{i+1}, \ldots, v'_r, v'_0, \ldots, v'_{i-1})$$

where

- for every vertex $u \notin \{v'_k, v''_k : 0 \leq k \leq r\} \cup \{w'_k, w''_k : 0 \leq k \leq s\}$ of H, $\sigma_{G,H}(u) = u$;
- for every $k \in \{1, \ldots, s-1\}$, $\sigma^{-1}_{U,H}(\{w'_k, w''_k\}) = w_k \in V(U)$, and $\sigma_{G,U}(w_k) = w_k$;
- for every $k \in \{0, \ldots, r\} \smallsetminus \{i, j\}$, $\sigma^{-1}_{G,U}(\{v'_k, v''_k\}) = v_k \in V(U)$, and $\sigma_{U,H}(v_k) = v_k$;
- $\sigma^{-1}_{U,H}(\{w'_0, w''_0\}) = v'_i \in V(U)$, $\sigma^{-1}_{U,H}(\{w'_s, w''_s\}) = v''_j \in V(U)$, $\sigma^{-1}_{G,U}(\{v'_i, v''_i\}) = v_i \in V(G)$, and $\sigma^{-1}_{G,U}(\{v'_j, v''_j\}) = v_j \in V(G)$ (note that this applies to both cases $i = j$ and $i \neq j$).

Remark.. The way the vertices of B are listed provides B with a reference direction, say clockwise. This reference direction is crucial for checking that the two faces of U with respective boundary walks $v'_i, v'_{i+1}, \ldots, v'_r, v'_0, \ldots, v'_{i-1}$ and $v''_j, v''_{j-1}, \ldots, v''_0, v''_r, \ldots, v''_{j+1}$ can be merged for forming a handle. Global consistency specifies that, for these two faces to be merged, their directions inherited from the reference direction of B must both be clockwise (cf., Figure 11(d)). Indeed, while one face is traversed clockwise with increasing indices, the other is traversed clockwise with decreasing indices. This matches the specification of handles (cf. Figure 3).

By the construction in Section 3.2.1, for every graph G of genus 1, (G, H^*, B^*, U^*) is globally consistent, where $H^* = G_{C,P}$, $U^* = G_C$, and B^* is the boundary walk of ϕ^* displayed in Eq. (1). The following result is specific to the torus, but it illustrates the basis for the design of our proof-labeling schemes.

Lemma 7. *Let H be a splitting of a graph G, and assume that there exists a planar embedding f of H with a face ϕ and a boundary walk B of ϕ. Let U be a 2-splittting of G such that H is a 2-splitting of U. If (G, H, B, U) is globally consistent, then G can be embedded on the torus $\mathbb{T}_1$.*

PROOF. Using the specifications of the splits, the two sub-paths $(w'_0, \ldots, w'_s)$ and $(w''_0, \ldots, w''_s)$ of B can be identified by merging each pair of vertices w'_k and w''_k, $k \in \{1, \ldots, s-1\}$, into a single vertex $w_k = \sigma^{-1}_{U,H}(\{w'_k, w''_k\})$ of U, by merging the vertices w'_0 and w''_0 into a single vertex v'_i of U, and by merging the vertices w'_s and w''_s into a single vertex v''_j of U. The resulting sequence $v'_i, w_1, \ldots, w_{s-1}, v''_j$ forms a path in U connecting two faces ϕ' and ϕ'', replacing the face ϕ of the planar embedding f of H, with respective boundary walks $(v'_0, v'_1, \ldots, v'_r)$ and $(v''_r, v''_{r-1}, \ldots, v''_0)$, where the vertices are ordered clockwise. These transformations preserve the planarity of the embedding, that is, U is planar. Next, the two cycles $(v'_0, \ldots, v'_r)$ and $(v''_0, \ldots, v''_r)$ can be identified, by merging each pair of nodes v'_k and v''_k into a single node $v_k = \sigma^{-1}_{G,U}(\{v'_k, v''_k\})$ of G. As a result, the two faces ϕ' and ϕ'' are replaced by a handle, providing an embedding of G on $\mathbb{T}_1$.

The outcome of Lemma 7 is that (H^*, f^*, ϕ^*, B^*) is essentially a certificate that G can be embedded on $\mathbb{T}_1$ (up to also providing the "intermediate" splitting U^* resulting from cycle-duplication). In the next section, we show how to generalize this construction for deriving a certificate that a graph G can be embedded on $\mathbb{T}_k$, $k > 1$.

The process described in the previous section for genus 1 can be generalized to larger genus $k \geq 1$, as follows. Again, let G be a graph, and let f be a 2-cell embedding of G on $\mathbb{T}_k$.

3.2.3. The Face-Duplication Phase

Let $\Sigma^{(0)} = \mathbb{T}_k$. As for the torus, let C_1 be a non-separating orientable cycle of $G^{(0)} = G$, and let us consider the embedding of $G^{(1)} = G^{(0)}_{C_1}$ induced by f, on the surface $\Sigma^{(1)} = \Sigma^{(0)}_{C_1}$ of genus $k-1$. This operation can be repeated. Indeed, by Lemma 6, there exists a non separating cycle C_2 of $G^{(1)}$. The graph $G^{(2)} = G^{(1)}_{C_2}$ can be embedded on the surface $\Sigma^{(2)} = \Sigma^{(1)}_{C_2}$ with one face more than the number of faces of the embedding of $G^{(1)}$ on $\Sigma^{(1)}$, and thus two more faces than the number of faces of the embedding of G on $\mathbb{T}_k$. By Lemma 4, $\Sigma^{(2)}$ has thus genus $k-2$. See Figure 1(a-b).

This process can actually be iterated k times, resulting in a sequence of $k+1$ graphs $G^{(0)}, \ldots, G^{(k)}$ where $G^{(0)} = G$, and a sequence of $k+1$ closed surfaces $\Sigma^{(0)}, \ldots, \Sigma^{(k)}$ where $\Sigma^{(0)} = \mathbb{T}_k$. Each graph $G^{(i)}$ is embedded on the closed surface $\Sigma^{(i)}$ of genus $k-i$, as follows. The embedding of $G^{(0)}$ on $\Sigma^{(0)}$ is the embedding of G on Σ, and, for every $i = 0, \ldots, k-1$, the embedding of

$G^{(i+1)}$ on $\Sigma^{(i+1)}$ is induced by the embedding of $G^{(i)}$ on $\Sigma^{(i)}$, after duplication of a non-separating cycle C_{i+1} of $G^{(i)}$ into two cycles C'_{i+1} and C''_{i+1}.

The closed surface $\Sigma^{(k)}$ is of genus 0, i.e. $\Sigma^{(k)}$ is homeomorphic to the sphere $\mathbb{T}_0 = S^2$ (see Figure 1(b)). The graph $G^{(k)}$ is therefore planar, for it contains k more faces than the number of faces in G, as two new faces ϕ'_i and ϕ''_i are created at each iteration i, in replacement to one face ϕ_i, for every $i = 1, \ldots, k$.

3.2.4. The Face-Reduction Phase

The objective is now to replace the $2k$ faces ϕ'_i, ϕ''_i, $i = 0, \ldots, k-1$, by a single face. For this purpose, let us relabel these faces as $\psi_1, \ldots, \psi_{2k}$ (see Figure 1(c)) so that, for $i = 1, \ldots, k$,

$$\phi'_i = \psi_{2i-1}, \text{ and } \phi''_i = \psi_{2i}.$$

Let $\chi_1 = \psi_1$. There exists a simple path P_1 between the two faces χ_1 and ψ_2. Duplicating P_1 preserves the fact that the graph $G^{(k+1)} = G^{(k)}_{P_1}$ can be embedded on the sphere $\mathbb{T}_0$. By this duplication, the two faces χ_1 and ψ_2 are merged into a single face χ_2. Now, there is a simple path P_2 between the two faces χ_2 and ψ_3 (see Figure 1(d)). Again, duplicating P_2 preserves the fact that the graph $G^{(k+2)} = G^{(k+1)}_{P_2}$ can be embedded on the sphere $\mathbb{T}_0$, in which the two faces χ_2 and ψ_3 are now merged into a single face χ_3. By iterating this process, a finite sequence of graphs $G^{(k)}, \ldots, G^{(3k-1)}$ is constructed, where, for $i = 0, \ldots, 2k-1$, the graph $G^{(k+i)}$ is coming with its embedding on $\mathbb{T}_0$, and with a set of special faces $\chi_{i+1}, \psi_{i+2}, \ldots, \psi_{2k}$. A path P_{i+1} between χ_{i+1} and ψ_{i+2} is duplicated for merging these two faces into a single face χ_{i+2}, while preserving the fact that $G^{(k+i+1)} = G^{(k+i)}_{P_{i+1}}$ can be embedded on the sphere $\mathbb{T}_0$.

Eventually, the process results in a single face $\phi^* = \chi_{2k}$ of $H^* = G^{(3k-1)}$ (see Figure 1(e)). This face contains all duplicated vertices. The embedding f of G on $\mathbb{T}_k$ induces a planar embedding of H^* whose external face is ϕ^* (see Figure 1(f)).

3.2.5. Certifying Genus at Most k

Conversely, for a graph G of genus k, an embedding of G on $\mathbb{T}_k$ can be induced from the embedding f^* of H^* on $\mathbb{T}_0$, and from the boundary walk B^* of ϕ^*. The latter is indeed entirely determined by the successive cycle- and path-duplications performed during the whole process. It contains all duplicated vertices, resulting from the cycles $C'_1, \ldots, C'_k$ and $C''_1, \ldots, C''_k$, and

from the paths $P'_1, \ldots, P'_{2k-1}$ and $P''_1, \ldots, P''_{2k-1}$. Note that the duplication process for a vertex may be complex. A vertex may indeed be duplicated once, and then one of its copies may be duplicated again, and so on, depending on which cycle or path is duplicated at every step of the process. This phenomenon actually already occurred in the basic case of the torus $\mathbb{T}_1$ where the duplications of v_i and v_j were more complex that those of the other vertices, and were also differing depending on whether $i = j$ or not (see Section 3.2). Figure 2 illustrates a case in which two cycles C_i and C_j share vertices and edges in $\mathbb{T}_2$, causing a series of duplication more complex than the basic case illustrated on Figure 1. In particular, a same vertex of H^* may appear several times on the boundary walk B^*, and a same edge of H^* may be traversed twice, once in each direction.

Let H be a splitting of a graph G, let f be a planar embedding of H, and let ϕ be a face of H embedded on $\mathbb{T}_0$. Let $B = (u_0, \ldots, u_N)$ be a boundary walk of ϕ, and let $\vec{B}$ be an arbitrary reference direction given to B, say clockwise. Let $\mathcal{U} = (U_0, \ldots, U_{3k-1})$ be a sequence of graphs such that $U_0 = G$, $U_{3k-1} = H$, and, for every $i \in \{0, \ldots, 3k-2\}$, U_{i+1} is a 2-splitting of U_i. The splitting of U_i into U_{i+1} is denoted by $\sigma_i = (\alpha_i, \beta_i)$. The following extends the notion of global consistency defined in the case of the torus $\mathbb{T}_1$. We say that $(G, H, \vec{B}, \mathcal{U})$, is *globally consistent* if the following two conditions hold.

1. **Path-duplication checking.** Let $\chi_{2k} = \phi$, with directed boundary walk $\vec{B}(\chi_{2k}) = \vec{B}$. For every $i = 0, \ldots, 2k-1$, there exist faces $\chi_{i+1}, \psi^{(i)}_{i+2}, \ldots, \psi^{(i)}_{2k}$ of U_{k+i}, with respective directed boundary walks $\vec{B}(\chi_{i+1}), \vec{B}(\psi^{(i)}_{i+2}), \ldots, \vec{B}(\psi^{(i)}_{2k})$, and there exist vertices $u^{(i)}_1, \ldots, u^{(i)}_t$, $v^{(i)}_1, \ldots, v^{(i)}_r$, $w'^{(i)}_0, \ldots, w'^{(i)}_s$, and $w''^{(i)}_0, \ldots, w''^{(i)}_s$ of U_{k+i} such that
   - $\vec{B}(\chi_{i+1}) = (w'^{(i)}_0, \ldots, w'^{(i)}_s, v^{(i)}_1, \ldots, v^{(i)}_r, w''^{(i)}_s, \ldots, w''^{(i)}_0, u^{(i)}_1, \ldots, u^{(i)}_t)$;
   - for every vertex $x \in V(U_{k+i}) \setminus (\{w'^{(i)}_0, \ldots, w'^{(i)}_s\} \cup \{w''^{(i)}_0, \ldots, w''^{(i)}_s\})$, $\sigma_{k+i-1}(x) = x$;
   - for every $j \in \{0, \ldots, s\}$, $|\sigma^{-1}_{k+i-1}(\{w'^{(i)}_j, w''^{(i)}_j\})| = 1$;
   - $\vec{B}(\chi_i) = (x, u^{(i)}_1, \ldots, u^{(i)}_t, x)$ where $x = \sigma^{-1}_{k+i-1}(\{w'^{(i)}_0, w''^{(i)}_0\})$;
   - $\vec{B}(\psi^{(i-1)}_{i+1}) = (y, v^{(i)}_1, \ldots, v^{(i)}_r, y)$ where $y = \sigma^{-1}_{k+i-1}(\{w'^{(i)}_s, w''^{(i)}_s\})$;
   - for $j = i+2, \ldots, 2k$, $\sigma_{k+i-1}(\vec{B}(\psi^{(i-1)}_j)) = \vec{B}(\psi^{(i)}_j)$.

2. **Cycle duplication checking.** Let $\phi_1^{\prime(k)} = \chi_1$, and, for $i = 2, \ldots, k$, let $\phi_i^{\prime(k)} = \psi_{2i-1}^{(0)}$. For $i = 1, \ldots, k$, let $\phi_i^{\prime\prime(k)} = \psi_{2i}^{(0)}$. For every $i = 1, \ldots, k$, there exists faces $\phi_1^{\prime(i)}, \phi_1^{\prime\prime(i)}, \ldots, \phi_i^{\prime(i)}, \phi_i^{\prime\prime(i)}$ of U_i with respective directed boundary walks $\vec{B}(\phi_1^{\prime(i)}), \vec{B}(\phi_1^{\prime\prime(i)}), \ldots, \vec{B}(\phi_i^{\prime(i)}), \vec{B}(\phi_i^{\prime\prime(i)})$ such that
   - $\vec{B}(\phi_i^{\prime(i)}) = (v'_0, v'_1, \ldots, v'_r, v'_0)$ and $\vec{B}(\phi_i^{\prime\prime(i)}) = (v''_0, v''_r, v''_{r-1}, \ldots, v''_1, v''_0)$ for some $r \geq 2$, with $|\sigma_{i-1}^{-1}(\{v'_j, v''_j\})| = 1$ for every $j = 0, \ldots, r$;
   - for $j = 1, \ldots, i-1$, $\sigma_{i-1}(\vec{B}(\phi_j^{\prime(i-1)})) = \vec{B}(\phi_j^{\prime(i)})$, and $\sigma_{i-1}(\vec{B}(\phi_j^{\prime\prime(i-1)})) = \vec{B}(\phi_j^{\prime\prime(i)})$.

By the construction performed in Sections 3.2.3 and 3.2.4, for every graph G of genus k, $(G, H^*, \vec{B}^*, \mathcal{U}^*)$ is globally consistent, where $\mathcal{U}^* = (G^{(0)}, \ldots, G^{(3k-1)})$. The following result generalizes Lemma 7 to graphs of genus larger than 1.

Lemma 8. *Let H be a splitting of a graph G, and assume that there exists a planar embedding f of H with a face ϕ and a boundary walk B of ϕ. Let $\mathcal{U} = (U_0, \ldots, U_{3k-1})$ be a series of graphs such that $U_0 = G$, $U_{3k-1} = H$, and, for every $i \in \{0, \ldots, 3k-2\}$, U_{i+1} is a 2-splitting of U_i. If $(G, H, \vec{B}, \mathcal{U})$ is globally consistent, then G can be embedded on the torus $\mathbb{T}_k$.*

Proof. Condition 1 in the definition of global consistency enables to recover a collection $\psi_1, \ldots, \psi_{2k}$ of faces of U_k. These faces are inductively constructed, starting from the face ϕ of the planar embedding f of $U_{3k-1} = H$. At each iteration i of the induction, U_{k+i-1} has faces $\chi_i, \psi_{i+1}^{(i-1)}, \ldots, \psi_{2k}^{(i-1)}$ obtained from the faces $\chi_{i+1}, \psi_{i+2}^{(i)}, \ldots, \psi_{2k}^{(i)}$ of U_{k+i} by separating the face χ_{i+1} into two faces χ_i and $\psi_{i+1}^{(i-1)}$ connected by a path, while preserving the other faces $\psi_{i+2}^{(i)}, \ldots, \psi_{2k}^{(i)}$. This operation preserves planarity, and thus, in particular, U_k is planar.

The directions of the boundary walks of the faces $\psi_1, \ldots, \psi_{2k}$ are inherited from the original direction given to the boundary walk B. Condition 2 enables to iteratively merge face ψ_{2i} with face ψ_{2i-1}, $i = 1, \ldots, k$, by identifying the vertices of their boundary walks while respecting the direction of these walks, which guarantees that handles are created (and not a Klein-bottle-like construction). The process eventually results in the graph U_0 with k handles, providing an embedding of $U_0 = G$ on $\mathbb{T}_k$.

Thanks to Lemma 8, the overall outcome of this section is that the tuple

$$c = (H^*, f^*, \phi^*, B^*, \mathcal{U}^*)$$

constructed in Sections 3.2.3 and 3.2.4 is a certificate that G can be embedded on $\mathbb{T}_k$. This certificate c and its corresponding verification algorithm are however centralized. In the next section, we show how to distribute both the certificate c, and the verification protocol.

4. Proof-Labeling Scheme for Bounded Genus Graphs

In this section, we establish our first main result.

Theorem 1. *Let $k \geq 0$, and let $\mathcal{G}_k^+$ be the class of graphs embeddable on an orientable closed surface of genus at most k. There is a proof-labeling scheme for $\mathcal{G}_k^+$ using certificates on $O(\log n)$ bits in n-node graphs.*

The proof essentially consists of showing how to distribute the centralized certificate

$$(H, f, \phi, B, \mathcal{U})$$

used in Lemma 8 for a graph G, by storing $O(\log n)$ bits at each vertex of G, while allowing the vertices to locally verify the correctness of the distributed certificates, that is, in particular, verifying that $(G, H, B, \mathcal{U})$ is globally consistent. The rest of the section is entirely dedicated to the proof of Theorem 1. We start by defining the core of the certificates assigned to the nodes, called *histories*. Then, we show how to distribute the histories so that every node stores at most $O(\log n)$ bits, and we describe the additional information to be stored in the certificates for enabling the liveness and completeness properties of the verification scheme to hold. Recall that the nodes of G are given arbitrary distinct IDs picked from a set of polynomial range. The ID of node $v \in V(G)$ is denoted by $\mathrm{id}(v)$. Note that $\mathrm{id}(v)$ can be stored on $O(\log n)$ bits.

4.1. Histories

The description of the certificates is for positive instances, that is, for graphs $G \in \mathcal{G}_k^+$. For such an instance G, the prover performs the construction of Section 3.2.2, resulting in the series of 2-splitting graphs $G^{(0)} = G, G^{(1)}, \ldots, G^{(2k-2)}, G^{(2k-2)} = H^*$, a planar embedding f of H^*,

and the identification of a special face ϕ^* in this embedding, with boundary walk B^*. The successive duplications experienced by a vertex v of the actual graph G during the face-duplication and face-reduction phases resulting in H^* can be encoded as a rooted binary tree unfolding these duplications, called *history*.

For every vertex v of G, the history of v is denoted by $\mathsf{h}(v)$. The history of v is a rooted binary tree of depth $3k-1$ (all leaves are at distance $3k-1$ from the root). For $\ell = 0, \ldots, 3k-1$, the level ℓ of $\mathsf{h}(v)$ consists of the at most 2^ℓ nodes at distance ℓ from the root. The internal nodes of $\mathsf{h}(v)$ with two children are call *binary* nodes, and the internal nodes with one child are called *unary*.

- For $\ell = 0, \ldots, k-1$, the edges connecting nodes of level ℓ to nodes of level $\ell+1$ are corresponding to the duplication of the cycle $C_{\ell+1}$ in $G^{(\ell)}$ (cf. Section 3.2.3), and,
- for $\ell = 0, \ldots, 2k-1$, the edges connecting nodes of level $k+\ell$ to nodes of level $k+\ell+1$ are corresponding to the duplication of the path $P_{\ell+1}$ in $G^{(k+\ell)}$ (cf. Section 3.2.4).

The nodes of $\mathsf{h}(v)$ are provided with additional information, as follows.

4.1.1. Vertices and Adjacencies in the Splitting Graphs

For every $\ell = 1, \ldots, 3k-1$, every node x at level ℓ in $\mathsf{h}(v)$ is provided with the vertex u of $G^{(\ell)}$ it corresponds to, after the duplications of v corresponding to the path from the root to x. In particular, each leaf of $\mathsf{h}(v)$ is provided with the single vertex of $H^* = G^{(3k-1)}$ it corresponds to. Specifically, each internal node x of $\mathsf{h}(v)$ is provided with the set S_x of vertices of H^* marked at the leaves of the subtree of $\mathsf{h}(v)$ rooted at x. For a leaf x, $S_x = \{u\}$, where u is the avatar of v in H^* corresponding to the path from the root to the leaf x. Note that, for two distinct nodes at level ℓ in $\mathsf{h}(v)$, we have $S_x \cap S_y = \varnothing$.

The $3k-1$ splittings successively performed starting from G are 2-splittings, from which it follows that every vertex of G is split a constant number of times for a fixed k. The $\nu \geq 1$ avatars of $v \in V(G)$ in H^* are labeled $(\mathrm{id}(v), 1), \ldots, (\mathrm{id}(v), \nu)$. It follows that the ν leaves of $\mathsf{h}(v)$ are respectively labeled $(\mathrm{id}(v), 1), \ldots, (\mathrm{id}(v), \nu)$. For every node x of $\mathsf{h}(v)$, each set S_x is a subset of $\{(\mathrm{id}(v), 1), \ldots, (\mathrm{id}(v), \nu)\}$, and thus these sets S_x can be stored on $O(\log n)$ bits.

Every node x of $\mathsf{h}(v)$ at level $\ell \in \{0, \ldots, 3k-1\}$, which, as explained above, corresponds to a vertex of $G^{(\ell)}$, is also provided with the set N_x of the neighbors of S_x in $G^{(\ell)}$. The set N_x has the form $N_x = \{X_1, \ldots, X_d\}$ for some $d \geq 1$, where, for $i = 1, \ldots, d$, X_i is a vertex of $G^{(\ell)}$ corresponding to a set of avatars in H^* of some neighbor w of v in G.

Since some vertices $v \in V(G)$ may have arbitrarily large degree (up to $n-1$), the sets N_x may not be storable using $O(\log n)$ bits. As a consequence, some histories may not be on $O(\log n)$ bits, and may actually be much bigger. Nevertheless, a simple trick using the fact that graphs with bounded genus have bounded degeneracy (cf. Lemma 5) allows us to reassign locally the set N_x in the histories so that every node of G stores $O(\log n)$ bits only.

4.1.2. Footprints

Every node x of $\mathsf{h}(v)$ at level $\ell \in \{0, \ldots, 3k-1\}$ is provided with a (possibly empty) set F_x of ordered triples of the form (X, Y, Z) where $X \in N_x$, $Y = S_x$, and $Z \in N_x$, called *footprints*. Intuitively, each footprint encodes edges $\{X, Y\}$ and $\{Y, Z\}$ of $G^{(\ell)}$ occurring in:

- a boundary walk of one of the faces ϕ'_i or ϕ''_i, $i = 1, \ldots, \ell$, if $\ell \leq k$, or
- a boundary walk of one of the faces $\chi_{\ell-k}, \psi_{\ell-k+1}, \ldots, \psi_{2k}$, otherwise.

Note that these two edges are actually directed, from X to Y, and from Y to Z, reflecting that the boundary walk is traveled in a specific direction, inherited from some a priori direction, say clockwise, given to the boundary walk B^* of the face $\phi^* = \chi_{2k}$ (hence the terminology "footprints").

Note that a same vertex of $G^{(\ell)}$ may appear several times in the boundary walk of a face, and a same edge may appear twice, once in every direction. Therefore, a same node x of $\mathsf{h}(v)$ may be provided with several footprints, whose collection form the set F_x, which may be of non-constant size. On the other hand, for a fixed k, a constant number of boundary walks are under concern in total, from which it follows that even if a node x at level ℓ of $\mathsf{h}(v)$ must store a non-constant number of footprints in F_x, each of x's incident edges in $G^{(\ell)}$ appears in at most two footprints of F_x. We use this fact, together with the bounded degeneracy of the graphs of bounded genus, for reassigning locally the sets F_x in the histories so that every node of G stores $O(\log n)$ bits only.

4.1.3. Types

Last, but not least, for every node x of $\mathsf{h}(v)$, each of the two (directed) edges (X, Y) and (Y, Z) in every footprint (X, Y, Z) in F_x also comes with a *type* in

$$\mathsf{T}_k = \{C'_1, \ldots, C'_k, C''_1, \ldots, C''_k, P'_1, \ldots, P'_{2k-1}, P''_1, \ldots, P''_{2k-1}\},$$

which reflects when this edge was created during the cycle- and path-duplications.

Example.. Figure 12 provides examples of histories for some vertices of G in the case displayed on Figure 11. Figure 12(a-b) display the histories of v_i and v_j whenever $i \neq j$, while Figure 12(c) displays the histories of $v_i = v_j$ whenever $i = j$. In this latter case, the leaves w'_0, w''_0, w'_s, w''_s may be labeled as $(\mathrm{id}(v), 1), (\mathrm{id}(v), 2), (\mathrm{id}(v), 3), (\mathrm{id}(v), 4)$, respectively. Then v'_i is labeled $S_{v'_i} = \{(\mathrm{id}(v), 1), (\mathrm{id}(v), 2)\}$, while v''_i is labeled $S_{v''_i} = \{(\mathrm{id}(v), 3), (\mathrm{id}(v), 4)\}$, and root is labeled $S_{v_i} = \{(\mathrm{id}(v), 1), (\mathrm{id}(v), 2), (\mathrm{id}(v), 3), (\mathrm{id}(v), 4)\}$. The neighborhoods N_x of these nodes x of $\mathsf{h}(v_i)$ are depending on the graphs $G^{(0)} = G, G^{(1)}$, and $G^{(2)} = H^*$. Assuming that B^* is directed clockwise, as displayed on Figure 11(d), the leaf w'_0 is provided with footprint (v'_{i-1}, w'_0, w'_1) while the leaf w''_0 is provided with footprint (w''_1, w''_0, v'_{i+1}). Similarly, w'_s and w''_s are respectively provided with footprint $(w'_{s-1}, w'_s, v''_{i-1})$ and $(v''_{i+1}, w''_s, w''_{s-1})$, where the various nodes in these footprints are encoded depending on their labels in H^*, which depend on the IDs given to the neighbors of v_i in G. The footprint at v'_i is $(v'_{i-1}, v'_i, v'_{i+1})$, while the footprint at v''_i is $(v''_{i+1}, v''_i, v'_{i-1})$. In both case, the directions of the edges are inherited from the initial clockwise direction of the boundary walk B^*. The directed edges (v'_{i-1}, v'_i) and (v'_i, v'_{i+1}) receives type C'_1, while the directed edges (v''_{i+1}, v''_i) and (v''_i, v''_{i-1}) receives type C''_2. The four edges (v'_{i-1}, w'_0), (w''_0, v'_{i+1}), (v''_{i+1}, w''_s), and (w'_s, v''_{i-1}) are respectively inheriting the types C'_1, C'_1, C''_1, and C''_1 of the four edges (v'_{i-1}, v'_i), (v'_i, v'_{i+1}), (v''_{i+1}, v''_i), and (v''_i, v''_{i-1}). The directed edges (w'_0, w'_1) and (w'_{s-1}, w'_s) receive type P'_1, while the directed edges (w''_1, w''_0) and (w''_s, w''_{s-1}) receive type P''_1. Observe that the footprints are constructed upward the histories, while the types are assigned downward those trees.

We now detail how the footprints are constructed in general, and how the types are assigned to the edges of the footprints.

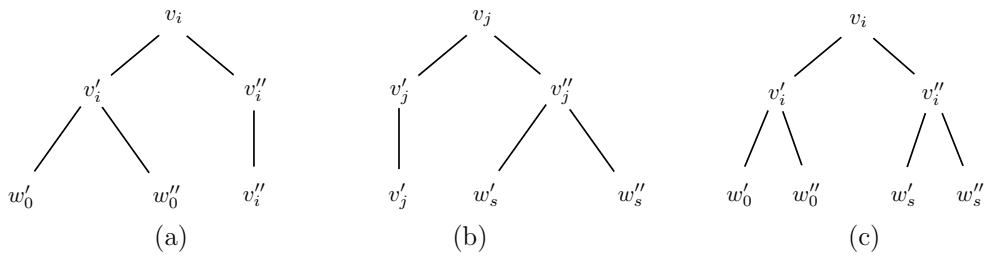


Figure 12: Examples of histories.

4.1.4. Construction of the Footprints

Let us give an arbitrary orientation, say clockwise, to the boundary walk B^* of the special face ϕ^* of H^*. This orientation induces footprints $(\mathsf{pred}(u), u, \mathsf{succ}(u)) \in F_x$ given to every leaf x of every history $\mathsf{h}(v)$, $v \in V(G)$. The vertex $\mathsf{pred}(u) \in V(H^*)$ is the predecessor of the avatar $u \in V(H^*)$ of v in H^*, and $\mathsf{succ}(u) \in V(H^*)$ is its successor. Note that some leaves x have $F_x = \varnothing$, whenever the corresponding node u in H^* does not belong to the boundary walk B^*. On the other hand, as a same node can be visited several times when traveling along the boundary walk B^*, some leaves may be given several footprints $(\mathsf{pred}_1(u), u, \mathsf{succ}_1(u)), \ldots, (\mathsf{pred}_d(u), u, \mathsf{succ}_d(u))$ in F_x, for some $d \geq 1$. The footprints provided to the internal nodes of the histories of the vertices of G are given in a way consistent with the orientation of B^*. More specifically, the footprints are constructed upward the histories, as follows.

Hereafter, the symbol "$\stackrel{\ell}{\rightarrow}$" stands for the operation performed when going from level $\ell-1$ to level ℓ, or vice-versa, from level ℓ to level $\ell-1$. For instance, for three sets S, S', S'' of vertices from H^*, the relation

$$S \stackrel{\ell}{\rightarrow} S', S''$$

states that the vertices S' and S'' of $G^{(\ell)}$ are the results of a cycle- or path-duplication experienced by the vertex S occurring from $G^{(\ell-1)}$ to $G^{(\ell)}$, i.e., the vertex $S = S' \cup S''$ of $G^{(\ell-1)}$ is split into two avatars, S' and S'', in $G^{(\ell)}$. If $\ell \leq k$, the split was caused by a cycle-duplication, otherwise it was caused by a path-duplication. Similarly, for two footprints F' and F'' at two nodes at level ℓ, children of a same binary node, the relation

$$F', F'' \stackrel{\ell}{\rightarrow} F$$

states that, when going upward a history, the two footprints F' and F'' of level ℓ generate the footprint F at level $\ell - 1$.

Three rules, called *Elementary, Extremity*, and *Vacancy*, are applied for the construction of the footprints. Their role is to "role back" the boundary walk B^* of the special face ϕ^* in the planar embedding of H^*. Each edge of the boundary walk B^* is indeed resulting from some duplication, of either a cycle or a path. The footprints encode the histories of all edges of the boundary walk B^* in all graphs $G^{(\ell)}$, $0 \leq \ell \leq 3k-1$, including when the edges were created (referred to as the *types* of the edges), and what were their successive extremities when those extremities are duplicated.

Elementary rule. Assuming $X \xrightarrow{\ell} X', X''$, $Y \xrightarrow{\ell} Y', Y''$, and $Z \xrightarrow{\ell} Z', Z''$, the elementary rule matches two footprints of two children Y' and Y'', and produces none at the parent Y:

$$(X', Y', Z'),\ (Z'', Y'', X'') \xrightarrow{\ell} \perp.$$

The Elementary rule applies to the case of cycle duplication, as well as to the case of path-duplication, but to the internal nodes of the path only (see Figure 13). When two cycles are merged (as the opposite to cycle duplication), their faces are glued together, and disappear. Similarly, when two paths are merged (as the opposite of path-duplication), the resulting path is of no use, and it can be discarded. Note that the two footprints (X', Y', Z') and (Z'', Y'', X'') are ordered in opposite directions. This matches the requirement for correctly glueing the borders of two faces in order to produce a handle (see Figures 3 and 7). This also matches the way the two copies of a path P_i are traversed when traveling along the boundary walk B^* in clockwise direction (cf. Eq. (1) and Figure 8).

Extremity rule. This rule applies only for levels $\ell > k$. It has two variants, defined below.

Single extremity rule. Assuming $X' \xrightarrow{\ell} X'$, $X'' \xrightarrow{\ell} X''$, $Y \xrightarrow{\ell} Y, |Y''$, and $Z \xrightarrow{\ell} Z', Z''$, the single extremity rule matches two footprints of two children Y' and Y'', and produces one footprint at the parent Y:

$$(X', Y', Z'),\ (Z'', Y'', X'') \xrightarrow{\ell} (X', Y, X'').$$

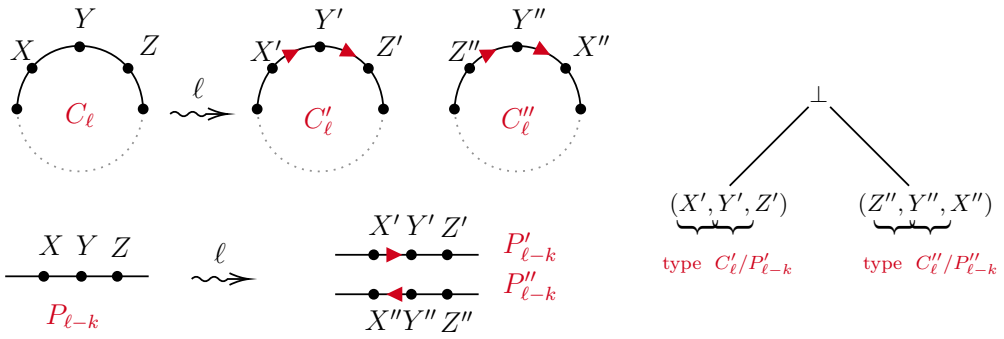


Figure 13: Footprint construction, and type assignment: Elementary rule.

Double extremity rule. Assuming $X' \xrightarrow{\ell} X'$, $X'' \xrightarrow{\ell} X''$, $Y \xrightarrow{\ell} Y', Y''$, $Z' \xrightarrow{\ell} Z'$, and $Z'' \xrightarrow{\ell} Z''$, the double extremity rule matches two footprints of two children Y' and Y'', and produces two footprints at the parent Y:

$$(X', Y', Z'),\ (Z'', Y'', X'') \xrightarrow{\ell} \big\{(X', Y, X''), (Z'', Y, Z')\big\}.$$

The Extremity rule refers to path duplication only (i.e., to levels $\ell > k$), as displayed on Figure 14. It is dedicated to the extremities of the path considered at this phase (see Figure 8). The Single extremity rule (cf. Figure 14(a)) handles the standard case in which the path is not trivial (i.e., reduced to a single vertex), whereas the Double extremity rule (cf. Figure 14(b)) handles the case in which the path connecting two faces is reduced to a single vertex Y (i.e., the two corresponding cycles share at least one vertex Y). Then only the vertex Y is split during the path duplication, while its four neighbors X', X'', Z', and Z' remain intact.

Vacancy rule. The vacancy rule simply forwards a footprint upward:

$$(X', Y,' Z') \xrightarrow{\ell} (X, Y, Z)$$

with $X \xrightarrow{\ell} X', X''$ (resp., $Y \xrightarrow{\ell} Y', Y''$, and $Z \xrightarrow{\ell} Z', Z''$), unless $X \xrightarrow{\ell} X$ (resp., $Y \xrightarrow{\ell} Y$, and $Z \xrightarrow{\ell} Z$), in which case $X = X'$ (resp., $Y = Y'$, and $Z = Z'$).

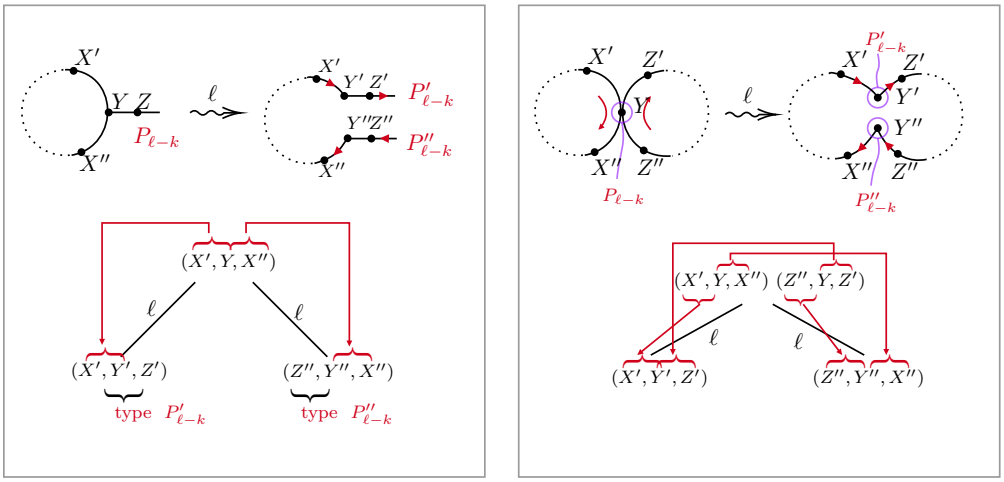


Figure 14: Footprint construction, and type assignment: Extremity rule.

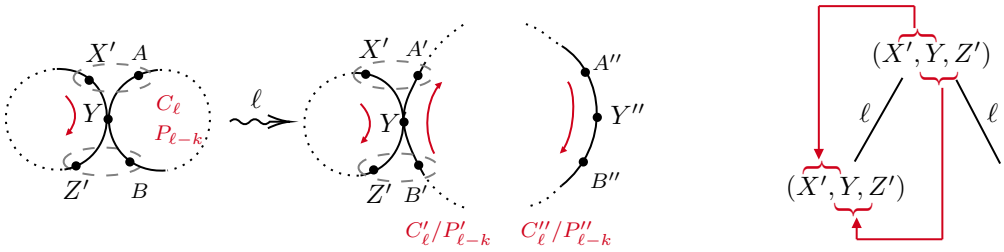


Figure 15: Footprint construction, and type assignment: Vacancy rule.

The Vacancy rule handles the case where one of the twin nodes carries a footprint (X', Y', Z') (resp., (X'', Y'', Z'')), which is copied to the parent node, after updating the vertices in case the latter experienced duplications (see Figure 15).

4.1.5. Assigning Types to Footprints

The types in T_k are assigned to the edges of the footprints, downwards the histories, as follows.

- If the footprints (X', Y', Z') and (Z'', Y'', X'') are matched by application of the Elementary rule at level ℓ, then the two (directed) edges (X', Y') and (Y', Z') (resp., (Z'', Y'') and (Y'', X'')) of $G^{(\ell)}$ are given

type C'_ℓ (resp., C''_ℓ) if $\ell \leq k$, and $P'_{\ell-k}$ (resp. $P''_{\ell-k}$) otherwise. See Figure 13.

- If the footprints (X', Y', Z') and (Z'', Y'', X'') are matched by application of the Single extremity rule at level ℓ, then the two edges (X', Y) and (Y, X'') adopt the types of the edges (X', Y') and (Y'', X''), respectively, while the two edges (Y', Z') and (Z'', Y'') are given type $P'_{k-\ell}$ and $P''_{k-\ell}$, respectively. See Figure 14(a).

- If the footprints (X', Y', Z') and (Z'', Y'', X'') are matched by application of the Double extremity rule at level ℓ, then the four edges (X', Y'), (Y', Z'), (Z'', Y''), and (Y'', X'') adopt the types of the edges (X', Y), (Y, Z'), (Z'', Y), and (Y, X''), respectively. See Figure 14(b)

- If the footprint $(X', Y,' Z')$ is forwarded upward as (X, Y, Z) by application of the Vacancy rule, then (X', Y'), and (Y', Z') adopt the types of the edges (X, Y), and (Y, Z), respectively. See Figure 15.

We have now all the ingredients to state what will be proved as sufficient to certify that a graph G has genus at most k.

4.2. Assignment of the Histories to the Certificates

As it was mentioned in Section 4.1, the history $\mathsf{h}(v)$ of a node v of the actual graph G may not be on $O(\log n)$ bits. The reason for that is that, even if G has a bounded genus k, the node v may have an arbitrarily large degree. As a consequence, the sum of the degrees of its v's avatars in each of the graphs $G^{(0)}, \ldots, G^{(3k-1)}$ may be arbitrarily large. This has direct consequences not only on the memory requirement for storing the neighborhood N_x of each node $x \in \mathsf{h}(v)$, but also on the number of footprints to be stored in F_x. In both cases, this memory requirement may exceed $O(\log n)$ bits. On the other hand, every graph G of bounded genus is sparse, which implies that the average degree of G, and of all its splitting graphs $G^{(0)}, \ldots, G^{(3k-1)}$ is constant. Therefore, the average memory requirement per vertex v for storing all the histories $\mathsf{h}(v)$, $v \in V(G)$, is constant. Yet, it remains that some vertices $v \in V(G)$ may have large histories, exceeding $O(\log n)$ bits.

The simple trick under this circumstances (cf., e.g., [25]) is to consider the space-complexity of the histories not per node of G, but per edge. Indeed, the space-complexity of the information related to each edge e of G, as stored in the histories, is constant, for *every* edge e. For instance, at a node x of

level ℓ in some history $\mathsf{h}(v)$, instead of storing N_x at v, one could virtually store every edge $\{S_x, S_y\}$, $S_y \in N_x$, on the edge $\{v, w\}$ of G, where w is the neighbor of v in G with avatar S_y in $G^{(\ell)}$.

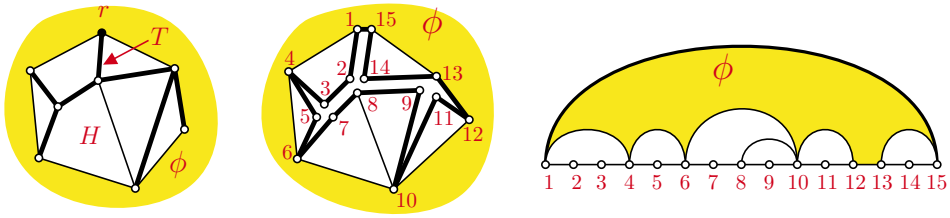
Let us define a *line* proof-labeling scheme as a proof-labeling scheme in which certificates are not only assigned to the vertices of G, but also to the edges of G (i.e., to vertices of the line-graph of G). In a line proof-labeling scheme, the vertices forge their decisions not only on their certificates and on the certificates assigned to their adjacent vertices, but also on the certificates assigned to their incident edges. Our interest for the concept of line proof-labeling scheme is expressed in the following result, after having recalled that, thanks to Lemma 5, every graph of genus at most k is d-degenerate for some constant d depending on k.

Lemma 9. *Let $f : \mathbb{N} \to \mathbb{N}$ such that $f(n) \in \Omega(\log(n))$. Let $d \geq 1$, and let $\mathcal{G}$ be a graph family such that every graph in $\mathcal{G}$ is d-degenerate. If $\mathcal{G}$ has a line proof-labeling scheme with certificate size $O(f(n))$ bits, then $\mathcal{G}$ has a proof-labeling scheme with certificate size $O(f(n))$ bits.*

Proof. Let $(\mathbf{p}, \mathbf{v})$ be line proof-labeling scheme for $\mathcal{G}$. For $G \in \mathcal{G}$, the prover $\mathbf{p}$ assigns certificate $\mathbf{p}(v)$ to every node $v \in V(G)$, and certificate $\mathbf{p}(e)$ to every edge $e \in V(G)$. Since G is d-degenerate, there exists a node v of G with degree $d_v \leq d$. Let $c(v)$ be the certificate of v defined as

$$c(v) = \Big(\mathbf{p}(v), \big\{(\mathrm{id}(u_1), \mathbf{p}(e_1)), \ldots, (\mathrm{id}(u_{d_v}), \mathbf{p}(e_{d_v})\big\}\Big),$$

where $u_1, \ldots, u_{d_v}$ are the d_v neighbors of v in G, and, for every $i = 1, \ldots, d_v$, $e_i = \{v, u_i\}$. Since the IDs can be stored on $O(\log n)$ bits, and since $f(n) \in \Omega(\log n)$, we get that $c(v)$ can be stored on $O(f(n))$ bits. This construction can then be repeated on the graph $G' = G - v$, which still has degeneracy at most d. By iterating this construction, all nodes are exhausted, and assigned certificates on $O(f(n))$ bits, containing all the information originally contained in the node- and edge-certificates assigned by $\mathbf{p}$. We complete the proof by observing that, for every edge $e = \{u, v\}$ of G, the certificate $\mathbf{p}(e)$ assigned by $\mathbf{p}$ to e can be found either in $c(u)$ or in $c(v)$. This suffices for simulating the behavior of $\mathbf{v}$, and thus for the design of a standard proof-labeling scheme for $\mathcal{G}$.



(a) (b) (c)

Figure 16: Illustration of the PLS for planarity in [25].

4.3. Certifying Planarity

In this section, we show how to certify that H is a planar embedding with a special face ϕ with boundary walk B. For this purpose, we just need to slightly adapt a recent proof-labeling scheme for planarity [25].

Lemma 10. *There exists a proof-labeling scheme for certifying that a given graph H has a planar embedding f, including a face ϕ with boundary walk B.*

Proof. Let H be a planar graph with a planar embedding f. The scheme for planarity in [25] constructs the certificates as follows (cf. Figure 16). Let T be an arbitrary spanning tree of H, and let us root T at a vertex $r \in V(H)$ on the outer face ϕ, as displayed on Figure 16(a). The tree T is "flattened" into a cycle C in a splitting H' of H by replacing every vertex $v \in V(H)$ by as many vertices as the number of times v is visited by a DFS traversal of T starting from r (see Figure 16(b)). The scheme in [25] certifies the cycle C, viewed as a path P whose two extremities are avatars of r, with respective DFS numbers 1 and $2n-1$, plus an edge connecting these two avatars (see Figure 16(c)). A property of this construction taken from [25] is that the vertices of H on the outer face ϕ are those which have at least one avatar in H' such that no co-tree edges "jumps over it" when the vertices are displayed as on Figure 16(c). For instance, the avatars $1, 4, 6, 10, 12, 13, 15$ have no co-tree edges jumping over them, and indeed these avatars are the ones of the vertices on the boundary of the outer face ϕ. The scheme of [25] is precisely based on a local encoding of the "lower edge" jumping over every avatars in H'. It follows that this scheme suffices for certifying not only the planarity of H, but also that ϕ is a face of H with boundary B.

4.4. Local Consistency

Let H be a splitting of a graph G, let f be a planar embedding of H, and let ϕ be a face of H with boundary walk B directed, say, clockwise. The directed boundary walk B is denoted by $\vec{B}$. Let $\mathsf{h}(G) = \{\mathsf{h}(v), v \in V(G)\}$ be a collection of histories for the vertices of G, of depth $3k-1$, for some $k \geq 1$. We say that $(G, H, \vec{B}, \mathsf{h}(G))$ is *locally consistent* if the following holds.

1. There exists a sequence of graphs $U_0, \ldots, U_{3k-1}$ with $U_0 = G$, $U_{3k-1} = H$, and, for every $0 \leq \ell < 3k-1$, $U_{\ell+1}$ is a degree-2 splitting of U_ℓ, such that, for every $v \in V(G)$, and for every $\ell = 0, \ldots, 3k-1$, every node x at level ℓ of $\mathsf{h}(v)$ satisfies that S_x is a vertex of U_ℓ, the neighborhood of S_x defined in N_x is consistent with the neighborhood of S_x in U_ℓ, and the footprints in F_x contains edges of U_ℓ. Moreover, if x has two children x' and x'' in $\mathsf{h}(v)$, then there are exactly two footprints, one in $E_{x'}$ and one in $E_{x''}$, for which the Elementary rule or the Extremity rule was applied, all the other footprints in $E_{x'}$ and $E_{x''}$ being subject to the Vacancy rule. Furthermore, if x has a unique child x', then all footprints in $E_{x'}$ are subject to the Vacancy rule. Finally, the typing is consistent with the specified typing rules.
2. The collection of footprints at the leaves of the histories in $\mathsf{h}(G)$ can be ordered as $(x_0, y_0, z_0), \ldots, (x_N, y_N, z_N)$ such that, $y_i = z_{i-1} = x_{i+1}$ for every $i = 0, \ldots, N$, and $\vec{B} = (y_0, \ldots, y_N)$.
3. For every $\ell = 1, \ldots, , 2k-1$, the following must be satisfied:
   (a) the collection of footprints at the nodes at level $k+\ell$ whose both edges have type P'_ℓ (resp., type P''_ℓ) in the histories in $\mathsf{h}(G)$ can be ordered as $(X'_0, Y'_0, Z'_0), \ldots, (X'_{s_\ell}, Y'_{s_\ell}, Z'_{s_\ell})$ (resp., $(Z''_0, Y''_0, X''_0), \ldots, (Z''_{s_\ell}, Y''_{s_\ell}, X''_{s_\ell})$), for some $s_\ell \geq 0$, such that:
      i. for every $i = 0, \ldots, s_\ell$, $Y_i \xrightarrow{k+\ell} \{Y'_i, Y''_i\}$;
      ii. for every $i = 1, \ldots, s_\ell$, $Y'_i = Z'_{i-1}$ and $Y''_i = Z''_{i-1}$;
      iii. for every $i = 0, \ldots, s_\ell - 1$, $Y'_i = X'_{i+1}$ and $Y''_i = X''_{i+1}$;
   (b) the collection of footprints at the nodes at level $k+\ell$ whose both edges have type $C'_{\lceil \frac{\ell+1}{2} \rceil}$ if $\ell+1$ is odd, or type $C''_{\frac{\ell+1}{2}}$ if $\ell+1$ is even, can be ordered as $(X_0, Y_0, Z_0), \ldots, (X_{r_\ell}, Y_{r_\ell}, Z_{r_\ell})$, for some $r_\ell \geq 0$ such that, for every $i = 1, \ldots, r_\ell$, $Y_i = Z_{i-1} = X_{i+1}$;
   (c) the collection of footprints at the nodes at level $k+\ell$ whose both edges have same type $P'_1, P''_1, \ldots, P'_{\ell-1}, P''_{\ell-1}, C'_1, C''_1, \ldots, C'_{\lceil \ell/2 \rceil}, C''_{\lceil \ell/2 \rceil}$, or $C'_{(\ell+1)/2}$ if

$\ell + 1$ is even, can be ordered as $(X_0, Y_0, Z_0), \ldots, (X_{t_\ell}, Y_{t_\ell}, Z_{t_\ell})$, for some $t_\ell \geq 0$, such that for every $i = 1, \ldots, t_\ell$, $Y'_i = Z'_{i-1}$ and $Y''_i = Z''_{i-1}$;

4. For every $\ell = 1, \ldots, k$, the collection of footprints at the nodes at level ℓ whose both edges have type C'_ℓ (resp., type C''_ℓ) in the histories in $\mathsf{h}(G)$ can be ordered as $(X'_0, Y'_0, Z'_0), \ldots, (X'_{r_\ell}, Y'_{r_\ell}, Z'_{r_\ell})$ (resp., $(Z''_0, Y''_0, X''_0), \ldots, (Z''_{r_\ell}, Y''_{r_\ell}, X''_{r_\ell})$), for some $r_\ell \geq 0$, such that:
   (a) for every $i = 0, \ldots, r_\ell$, $Y_i \xrightarrow{\ell} \{Y'_i, Y''_i\}$;
   (b) for every $i = 1, \ldots, r_\ell$, $Y_i = Z_{i-1} = X_{i+1}$;

By construction, $(G, H^*, \vec{B}^*, \mathsf{h}^*(G))$ produced by encoding the unfolding of the embedding of G on $\mathbb{T}_k$, described in Section 3.2.2, is locally consistent. The following result shows that the local notion of historical consistency based on the histories fits with the global notion of historical consistency used in Section 3.2.2.

Lemma 11. *Let H be a splitting of a graph G, let f be a planar embedding of H, let ϕ be a face of H with boundary walk $\vec{B}$ directed clockwise. Let $\mathsf{h}(G)$ be a history of all the vertices in G. If $(G, H, \vec{B}, \mathsf{h}(G))$ is locally consistent, then $(G, H, \vec{B}, \mathcal{U})$ is globally consistent, where $\mathcal{U} = U_0, \ldots, U_{3k-1}$ is a sequence of graphs enabling Condition 1 of the historical consistency of $(G, H, \vec{B}, \mathsf{h}(G))$ to hold.*

Proof. Thanks to Condition 1, for every $0 \leq \ell < 3k-1$, $U_{\ell+1}$ is a degree-2 splitting of U_ℓ. Moreover, by the consistence of the footprints and the typing in the histories, the splitting of from U_ℓ to $U_{\ell+1}$ is locally consistent at each node of U_i with the duplication of a cycle whenever $\ell \leq k$, and with the duplication of a path otherwise.

Condition 2 in the definition of local consistency guarantees that the footprints at the leaves of the histories are correctly set, that is, they collectively encode the boundary walk B.

Condition 3 guarantees that, for $\ell = 1, \ldots, 2k-1$, starting from $\chi_{2k} = B$, one can iteratively decompose the boundary walk of the face $\chi_{\ell+1}$ of $U_{k+\ell+1}$ into a boundary walk of a face $\psi_{\ell+1}$ of $U_{k+\ell}$, a boundary walk of a face χ_ℓ of $U_{k+\ell}$, and the duplication of a path in $U_{k+\ell}$ connecting χ_ℓ to $\psi_{\ell+1}$. It follows that $2k$ faces $\psi_1, \ldots, \psi_{2k}$ of U_ℓ have been identified. Since, the merging of the $2k-1$ paths successively identified in the graphs $U_{k+\ell}$, $\ell = 1, \ldots, 2k-1$ preserves planarity, the graph U_k is planar.

Moreover, each of the boundary walks of the faces $\psi_1, \ldots, \psi_{2k}$ is oriented in a direction inherited from the clockwise orientation of B, as guaranteed by the Elementary, Extremity, and Vacancy rules satisfied by the footprints, whose validity are themselves guaranteed by Condition 1. Condition 4 guarantees that the $2k$ faces $\psi_1, \ldots, \psi_{2k}$ of U_k can be reordered as k pairs (ϕ'_i, ϕ''_i), $i \in \{1, \ldots, k\}$ that can be successively merged for creating handles. More specifically, for $i = k, k-1, \ldots, 1$, Condition 4 guarantees that the boundary walks of ϕ'_i and ϕ''_i are directed such that, by identifying the vertices of U_i that are split of vertices in U_{i-1}, a handle is created, resulting in U_i embedded in $\mathbb{T}_{k-i}$.

4.5. Existence and Unicity of the Paths and Cycles

Our proof-labeling scheme relies on a collection of paths and cycles in the graphs $G^{(0)}, \ldots, G^{(3k-1)}$. The footprints and types encode these paths and cycles locally. One needs to guarantee the existence and unicity of each path and cycle, in each graph $G^{(i)}$, $i = 0, \ldots, 3k-1$. The next lemma, which is standard, achieve this task.

Lemma 12. *Let G be a graph, and let P (resp., C) be a (non-necessary simple) directed path (resp., cycle) in G. Assume each vertex v of P (resp., C) is given a triple $(\mathsf{pred}(v), v, \mathsf{succ}(v))$, where $\mathsf{pred}(v)$ and $\mathsf{succ}(v)$) are the predecessor and successor of v in P (resp., C). If v is an extremity of P, then $\mathsf{pred}(v) = \perp$ or $\mathsf{succ}(v) = \perp$, or both $\mathsf{pred}(v) = \perp$ and $\mathsf{succ}(v) = \perp$ in case P is reduced to v. There exists a proof-labeling scheme with certificates on $O(\log n)$ bits that guarantees the existence and unicity of P.*

Proof. Let P be a directed path in G. The proof-labeling scheme uses a spanning tree T of G rooted at the starting vertex v_0 of P. Every vertex v is given the ID of its parent $p(v)$ in T (v_0 has $p(v_0) = \perp$). The tree T is certified by providing a certificate to every node v containing a pair $(\mathrm{id}(v_0), d(v))$, where $d(v)$ is the distance from v to v_0 in T. Every vertex v checks that it is given the same root-ID as its neighbors in G, and that $d(p(v) = d(v) - 1$. Every node that is given one or many triples $(\mathsf{pred}(v), v, \mathsf{succ}(v))$ checks that, for each of them, $\mathsf{pred}(\mathsf{succ}(v)) = v$ and $\mathsf{succ}(\mathsf{pred}(v)) = v$. (Of course, every such vertex v also checks consistence of the triples given to it, including the fact that $\mathsf{pred}(v) \neq \mathsf{succ}(v)$ unless they are both equal to $\perp$, that it is not given the same successor in two different triples, etc.). If one of the tests is not passed at a vertex, this vertex rejects, otherwise it accepts. The case of

a cycle C is treated the same, where the spanning tree T is rooted at any vertex of C. It is easy to check that this standard proof-labeling scheme satisfies both completeness and soundness.

4.6. Verification Procedure

We now have all ingredients for describing our proof-labeling scheme for $\mathcal{G}_k^+$, $k \geq 0$. First, we describe the certificates assigned to the vertices of a graph G of genus k. The main part of the certificate of v is the history $\mathsf{h}(v)$, as constructed in Section 4.1. As mentioned in Section 4.2, a history may require more than just $O(\log n)$ bits. However, Lemma 9 has shown how to resolve this issue, so that histories can be spread out among the vertices in a way guaranteeing that every vertex stores $O(\log n)$ bits, and, in a single round of communication with its neighbors, every node v can recover its entire history. More importantly even, although a vertex v may not be able to recover the whole history of each of its neighbors in a single round, yet it can recover from each neighbor w the part of $\mathsf{h}(w)$ corresponding to every edge between an avatar of v and an avatar of w, which is sufficient to check the consistency of the neighborhoods, footprints, etc., in all graphs $G^{(0)}, \ldots, G^{(3k-1)}$ used in the construction. In addition, the certificate of every vertex is provided with the information enabling to check planarity of $H = G^{(3k-1)}$ (cf. Lemma 10), and to guarantee the existence and unicity of all the directed cycles C'_i, C''_i, $i = 1, \ldots, k$, and all directed paths P'_j, P''_j, $j = 1, \ldots, 2k-1$ (cf. Lemma 12). The vertices can then check local consistency, as specified in Section 4.4. Since G has genus k, it follows that, whenever the prover assigns the certificates appropriately, all vertices pass all tests, and therefore all vertices accept. Completeness is therefore satisfied by the scheme.

Soundness is guaranteed by Lemmas 8 and 11. Indeed, the latter lemma shows that if the vertices are given certificates that are consistent, and in particular for which the histories are locally consistent, then global consistency is also guaranteed. And the former lemma says that if global consistency is satisfied then the graph can be embedded on $\mathbb{T}_k$. Therefore, if a graph G cannot be embedded on $\mathbb{T}_k$, then global consistency cannot be satisfied, which means that the local consistency of the histories cannot be satisfied either, and therefore, at least one vertex of G fails to pass all tests, and rejects. This completes the proof of Theorem 1.

5. Proof-Labeling Scheme for Bounded Non-orientable genus Graphs

This section is entirely dedicated to the proof of our second main result.

Theorem 2. *Let $k \geq 0$, and let $\mathcal{G}_k^-$ be the class of graphs with non-orientable genus at most k, i.e., embeddable on a non-orientable closed surface of genus at most k. There is a proof-labeling scheme for $\mathcal{G}_k^-$ using certificates on $O(\log n)$ bits in n-node graphs.*

The proof-labeling scheme for $\mathcal{G}_k^-$ is based on the same ingredients as the one for $\mathcal{G}^+$ in Theorem 1 (e.g., Lemma 3 is used in replacement of Lemma 2, etc.). However, new ingredients must be introduced for handling the cross-caps from which non-orientable surfaces result. The proof will thus mainly consist in describing these new ingredients, and in explaining their interactions with the ingredients used for establishing Theorem 1. We start by defining the notion of *doubling* performed on cycles.

5.1. Doubling of a Non-Orientable Cycle

Let us assume that we are given an embedding of a graph G on a non-orientable closed surface Σ of genus k, and let $D = (v_0, v_1, \ldots, v_{p-1}, v_p = v_0)$ be a non-orientable cycle of G. Note that a non-orientable cycle is non-separating. The graph G_D is obtained by *doubling* D, i.e., by multiplying its length by 2. This doubling of D, and the canonical embedding of G_D on a closed surface Σ_D, are obtained as follows (see Figure 17 for an illustration).

- Each vertex v_i, $0 \leq i < p$, is split into two vertices v'_i and v'_{p+i} in such a way that $D' = (v'_0, v'_1, v'_2, \ldots, v'_{2p-1}, v'_{2p} = v'_0)$ is a cycle of G_D, which forms a boundary walk of a face ϕ of X_D.
- The neighbors of each vertex v_i in $G \setminus D$, $0 \leq i < p$, are shared between v'_i and v'_{i+p} in G_D, as follows. The left and right sides of D can be defined locally, i.e., in the neighborhood of each (embedded) edge $\{v_i, v_{i+1}\}$ of D. The edges incident to v'_i and v'_{i+1} in G_D (and, by symmetry, the edges incident to v'_{i+p} and v'_{i+p+1}) correspond to the edges incident to v_i and v_{i+1} on the same side of D in G according to the local definition of left and right sides in the neighborhood of $\{v_i, v_{i+1}\}$.
- The vertices v'_i and v'_{i+p} have no other neighbors.

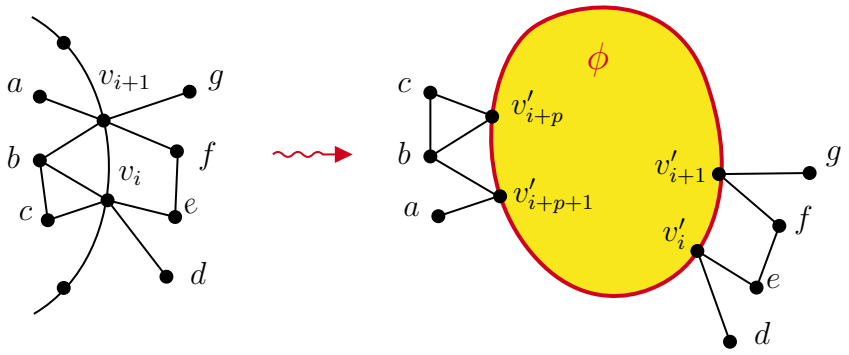


Figure 17: Doubling a non-orientable cycle.

We now show how to unfold $\mathbb{P}_k$, as we did for unfolding $\mathbb{T}_k$ in the oriented case.

5.2. Unfolding $\mathbb{P}_k$ for $k \geq 1$

Let G be a graph with a 2-cell embedding f on $\mathbb{P}_k$. The unfolding of G has three phases, and only the first one, called *doubling phase* is new. The second phase is a face-duplication phase, and the third phase is a face-reduction phase, identical to those described in the case of orientable surfaces. The doubling phase is as follows. Let $\Sigma^{(0)} = \mathbb{P}_k$, and let D_1 be a non-orientable cycle of $G^{(0)} = G$. Let us consider the embedding of $G^{(1)} = G^{(0)}_{D_1}$ induced by f, on the surface $\Sigma^{(1)} = \Sigma^{(0)}_{D_1}$. There are two cases, both using Lemma 4:

- If $\Sigma^{(1)}$ is non-orientable, then $\Sigma^{(1)}$ is homeomorphic to $\mathbb{P}_{k-1}$;
- Otherwise, $\Sigma^{(1)}$ is homeomorphic to $\mathbb{T}_{\frac{k-1}{2}}$.

In the first case, a doubling operation is repeated on $G^{(1)}$, using a non-orientable cycle D_2 of $G^{(1)}$. Doubling operations are performed iteratively until an embedding on an orientable surface is reached. Formally, there exists a a sequence of $m+1$ graphs $G^{(0)}, \ldots, G^{(m)}$, $m \leq k$, respectively embedded on closed surfaces $\Sigma^{(0)}, \ldots, \Sigma^{(m)}$, such that, for $0 \leq i < m$, there exists a non-orientable cycle D_{i+1} of $G^{(i)}$ such that $G^{(i+1)} = (G^{(i)})_{D_{i+1}}$, and $\Sigma^{(i+1)} = (\Sigma^{(i)})_{D_{i+1}}$ (up to homeomorphism). Necessarily, for $0 \leq i < m$, $\Sigma^{(i)} = \mathbb{P}_{k-i}$ (up to homeomorphism), and $\Sigma^{(m)} = \mathbb{T}_{(k-m)/2}$, thanks to Lemma 4. When $\Sigma^{(m)}$ is reached, $G^{(m)}$ contains m special faces, whose boundary walks

are resulting from the successive doubling of $D_1, \ldots, D_m$, respectively. The doubling phase is then completed.

The face duplication phase starts, initialized with the embedding of $G^{(m)}$ on $\Sigma^{(m)}$. Let $k' = \frac{k-m}{2}$. The duplication phase is performed, as in Section 3.2.3. Specifically, there exists a sequence of $k'+1$ graphs $G^{(m)}, \ldots, G^{(m)+k'}$, respectively embedded on closed surfaces $\Sigma^{(m)}, \ldots, X'^{(m+k')}$, such that, for $0 \leq i < k'$, there exists a non-separating cycle C_{i+1} of $G^{(m+i)}$ such that $G^{(m+i+1)} = G^{(m+i)}_{C_{i+1}}$, and $\Sigma^{(m+i+1)} = \Sigma^{(m+i)}_{C_{i+1}}$. Necessarily, for $0 \leq i \leq k'$, $\Sigma^{(m+i)} = \mathbb{T}_{k'-i}$ up to homeomorphism, thanks to Lemma 4. In particular, $\Sigma^{(m+k')} = \mathbb{T}_0$. When $\Sigma^{(m+k')}$ is reached, $G^{(m+k')}$ contains $2k'+m$ special faces, whose boundary walks are resulting from the successive doubling of the cycles $D_1, \ldots, D_m$, and from the duplications of the cycles $C_1, \ldots, C_{k'}$. At this point, the face-duplication phase is completed.

The face-reduction phase starts, as in Section 3.2.4, in order to merge the $2k'+m = k$ special faces of $G^{(m+k')}$ into a single face. Let us denote the $2k'+m = k$ special faces of $G^{(m+k')}$ by $\psi_1, \ldots, \psi_k$. Let $\psi_1 = \chi_1$. There exists a sequence of paths $P_1, \ldots, P_{k-1}$ such that, for $1 \leq i \leq k-1$, the duplication of P_i merges χ_i and ψ_{i+1} in a single face χ_{i+1}. A sequence of planar graphs $G^{(m+k')}, \ldots, G^{(m+k'+k-1)}$ results from these merges, where, for $0 \leq i < k-1$, P_{i+1} is a path of $G^{(m+k'+i)}$, and $G^{(m+k'+i+1)} = G^{(m+k'+i)}_{P_{i+1}}$. For $1 \leq i \leq k-1$, $G^{(m+k'+i)}$ has $k-i$ special faces $\chi_{i+1}, \psi_{i+2}, \ldots, \psi_k$. In particular, $G^{(m+k'+k-1)}$ has a unique special face χ_{k-1}.

To summarize, as in Section 3.2.2, the embedding f of G in $\mathbb{P}_k$ induces a planar embedding of $H^* = G^{(m+k'+k-1)}$ whose external face is $\phi^* = \chi_{k-1}$. The boundary of face ϕ^* contains all the vertices obtained by splittings resulting from doublings or duplications.

5.3. Certifying Non-Orientable Genus at Most k

Conversely, for a graph G of non-orientable genus k, an embedding of G in $\mathbb{P}_k$ can be induced from the embedding f^* of H^* on $\mathbb{T}_0$, and from the boundary walk B^* of ϕ^*. The latter is indeed entirely determined by the successive cycle-duplications, path-duplications, and cycle doublings performed during the whole process. It contains all duplicated vertices resulting from the cycles $D'_1, \ldots, D'_m$, the cycles $C'_1, \ldots, C'_{k'}$ and $C''_1, \ldots, C''_{k'}$, and from the paths $P'_1, \ldots, P'_{k-1}$ and $P''_1, \ldots, P''_{k-1}$.

Now, let H be a splitting of a graph G, let f be a planar embedding of H, and let ϕ be a face of H embedded on $\mathbb{T}_0$. Let $B = (u_0, \ldots, u_N)$ be

a boundary walk of ϕ, and let $\vec{B}$ be an arbitrary direction given to B, say clockwise. Let $\mathcal{U} = (U_0, \ldots, U_{m+k'+k-1})$, with $m + 2k' = k$ and $m \geq 1$, be a sequence of graphs such that $U_0 = G$, $U_{m+k'+k-1} = H$, and, for every $i \in \{0, \ldots, m+k'+k-1\}$, U_{i+1} is a 2-splitting of U_i. The splitting of U_i into U_{i+1} is denoted by $\sigma_i = (\alpha_i, \beta_i)$. The definition of global consistency of $(G, H, \vec{B}, \mathcal{U})$, in the case of orientable surfaces, can trivially be adapted to the case of non-orientable surfaces by revisiting conditions 1 and 2, of Section 3.2.5, in such a way that the indices correspond to the unfolding of $\mathbb{P}_k$. We thus say that $(G, H, \vec{B}, \mathcal{U})$ is *globally consistent* for $\mathbb{P}_k$ if the (revisited) conditions 1 and 2 in Section 3.2.5 hold, plus the following additional condition corresponding to the doubling phase:

- **Cycle doubling checking.** For every $i = 1, \ldots, \ell$, there exist faces $\phi_1'^{(i)}, \phi_2'^{(i)}, \ldots, \phi_i'^{(i)}$ of U_i with respective directed boundary walks $\vec{B}(\phi_1'^{(i)}), \vec{B}(\phi_2'^{(i)}), \ldots, \vec{B}(\phi_i'^{(i)})$ such that
  - $\vec{B}(\phi_i'^{(i)}) = (v'_0, v'_1, \ldots, v'_{2p-1}, v'_{2p} = v'_0)$ with, for $0 \leq j < p$, $\sigma_{i-1}^{-1}(\{v'_j, v'_{j+p}\}) \in V(U_{i-1})$;
  - for $j = 1, \ldots, i-1$, $\sigma_{i-1}(\vec{B}(\phi_j'^{(i-1)})) = \vec{B}(\phi_j'^{(i)})$.

By the construction of Section 5.2, for every graph G of non-orientable genus k, $(G, H^*, \vec{B}^*, \mathcal{U}^*)$ is globally consistent for $\mathbb{P}_k$, where $\mathcal{U}^* = (G^{(0)}, \ldots, G^{(m+k'+k-1)})$. The following lemma is the analog to Lemma 8 for non-orientable surface. Its proof is essentially the same as the proof of Lemma 8, in which an argument should be added, for handling cycle doublings, that is, for identifying opposite vertices of the cycle D'_i in order to create a cross-cap. The details are omitted.

Lemma 13. *Let H be a splitting of a graph G, and assume that there exists a planar embedding f of H with a face ϕ and a boundary walk B of ϕ. Let m, k' be integers such that $1 \leq m \leq k$ and $m + 2k' = k$, and let $\mathcal{U} = (U_0, \ldots, U_{m+k'+k-1})$ be a series of graphs such that $U_0 = G$, $U_{m+k'+k-1} = H$, and, for every $i \in \{0, \ldots, m + k' + k - 2\}$, U_{i+1} is a 2-splitting of U_i. If $(G, H, \vec{B}, \mathcal{U})$ is globally consistent for $\mathbb{P}_k$, then G can be embedded on $\mathbb{P}_k$.*

Thanks to Lemma 13, the overall outcome of this section is that the tuple $c = (H^*, f^*, \phi^*, B^*, \mathcal{U}^*)$ constructed in Section 5.2 is indeed a certificate that G can be embedded on $\mathbb{P}_k$.

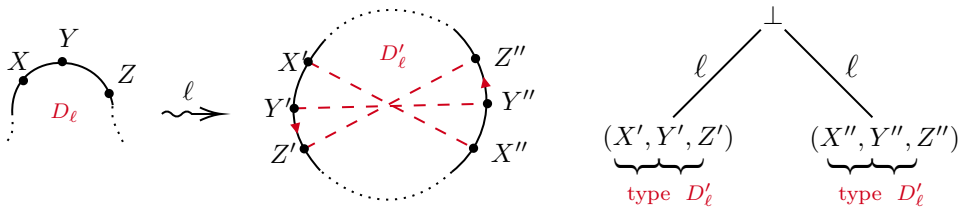


Figure 18: The cross-cap rule.

5.4. From Centralized Certificate to Local Certificate

The method to distribute the centralized certificates uses the same approach and the same tools as those used in Section 4 in the orientable case. Only the differences are pointed out in this section. In the non-orientable case, the set of types is

$$S_k = \{D'_1, \ldots, D'_\ell, C'_1, \ldots, C'_{k'}, C''_1, \ldots, C''_{k'}, P'_1, \ldots, P'_{k-1}, P''_1, \ldots, P''_{k-1}\}.$$

The footprints and their construction are identical to the orientable case, except that a cross-cap rule is introduced (see Figure 18).

Cross-cap rule. Assuming $X \xrightarrow{\ell} X', X''$, $Y \xrightarrow{\ell} Y', Y''$, and $Z \xrightarrow{\ell} Z', Z''$, the cross-cap rule matches two footprints of two children Y' and Y'', and produces none at the parent Y:

$$(X', Y', Z'),\ (X'', Y'', Z'') \xrightarrow{\ell} \perp.$$

The cross-cap rule applies to the case of identifying opposite vertices of the boundary of a face, in the reverse operation of doubling. The corresponding face disappears, and their boundaries can be discarded.

The assignments of types to footprints is performed in the same as in Section 4, and the same distributed algorithm is used for checking the planarity of H. An important difference with the orientable case appears in the definition of the local consistency of distributed certificates (previously defined in Section 4.4). Again, an additional condition is introduced, for reflecting the creation of cross-caps.

- For every $\ell = 1, \ldots, m$, the collection of footprints at the nodes at level ℓ whose both edges have type D'_ℓ in the histories in $\mathsf{h}(G)$ can be ordered as $(X'_0, Y'_0, Z'_0), \ldots, (X'_{2r_\ell-1}, Y'_{2r_\ell-1}, Z'_{2r_\ell-1})$, for some $r_\ell \geq 1$, such that:

1. for every $i = 0, \ldots, r_\ell - 1$, $Y_i \xrightarrow{\ell} \{Y'_i, Y''_{i+r_\ell}\}$;
2. for every $i = 0, \ldots, 2r_\ell - 1$, $Y_i = Z_{i-1} = X_{i+1}$ (where indices are taken modulo $2r_\ell$);

The following lemma is the analog of Lemma 11, but for non-orientable surfaces. Its proof is identical to the proof of Lemma 11, with an additional argument, stating that the conditions added for handling non-orientable surfaces enable opposite vertices of the face surrounded by D'_ℓ in U_ℓ, $1 \leq \ell \leq 2k-1$, to be identified for creating a cross-cap in $U_{\ell-1}$.

Lemma 14. *Let H be a splitting of a graph G, let f be a planar embedding of H, let ϕ be a face of H with boundary walk $\vec{B}$ directed clockwise. Let $\mathsf{h}(G)$ be a history of all the vertices in G. If $(G, H, \vec{B}, \mathsf{h}(G))$ is locally consistent, then $(G, H, \vec{B}, \mathcal{U})$ is globally consistent, where $\mathcal{U} = U_0, \ldots, U_{m+k'+k-1}$ is a sequence of graphs enabling the global consistency of $(G, H, \vec{B}, \mathsf{h}(G))$ to hold.*

5.5. Verification Procedure

The verification procedure is similar to the one described in Section 4.6, and is therefore omitted.

6. Conclusion

In this paper, we have designed proof-labeling schemes for the class of graphs of bounded genus, as well as for the class of graphs with bounded non-orientable genus. All our schemes use certificates on $O(\log n)$ bits, which is optimal, as it is known that even certifying the class of planar graphs requires proof-labeling schemes with certificates on $\Omega(\log n)$ bits [25]. The existence of "compact" proof-labeling schemes (i.e., schemes using certificates of polylogarithmic size) for other classes of sparse graphs is still not known. In particular, proving or disproving the existence of such a scheme for H-minor-free graphs appears to be a challenging problem. Indeed, Robertson and Seymour's decomposition theorem states that every H-minor-free graph can be expressed as a tree structure of "pieces", where each piece is a graph that can be embedded in a surface on which H cannot be embedded, plus a bounded number of so-called *apex* vertices, and a bounded number of so-called *vortex* subgraphs. The decomposition theorem provides a powerful tool for the design of (centralized or distributed) algorithms. However, this theorem is not a characterization, that is, there are graphs that are not

H-minor-free, and yet can be expressed as a tree structure satisfying the required properties (surfaces of bounded genus, bounded number of apices, bounded number of vortices, etc.). It follows that, although Robertson and Seymour's decomposition theorem should most probably play a crucial role for designing a compact proof-labeling scheme for H-minor-free graphs (if such a scheme exists), this development may require identifying additional properties satisfied by these graphs.

Recently several papers have tackled the question of certifying minor-closed classes, with various restrictions. In [22, 28] the authors show meta-theorems for certification with the corollaries that if the forbidden minors are paths and planar graphs, respectively, one can certify the class with logarithmic and polylogarithmic certificates, respectively. Small minors have also been studied in [9]. Finally, [20] studies the question in the approximate certification model (in the spirit of property testing) (as introduced in [11] and studied in [18] for planar graphs).

Acknowledgements.. The first and fifth authors are thankful to Gelasio Salazar for his very detailed answers to their questions about closed surfaces. The authors thank the reviewers for the detailed comments.

References

[1] Ittai Abraham, Cyril Gavoille, and Dahlia Malkhi. Compact routing for graphs excluding a fixed minor. In *19th International Conference on Distributed Computing (DISC)*, LNCS 3724, pages 442–456. Springer, 2005.

[2] Yehuda Afek, Shay Kutten, and Moti Yung. The local detection paradigm and its application to self-stabilization. *Theor. Comput. Sci.*, 186(1-2):199–229, 1997.

[3] Saeed Akhoondian Amiri, Patrice Ossona de Mendez, Roman Rabinovich, and Sebastian Siebertz. Distributed domination on graph classes of bounded expansion. In *30th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 143–151, 2018.

[4] Saeed Akhoondian Amiri, Stefan Schmid, and Sebastian Siebertz. A local constant factor MDS approximation for bounded genus graphs. In *ACM Symposium on Principles of Distributed Computing (PODC)*, pages 227–233, 2016.

[5] Saeed Akhoondian Amiri, Stefan Schmid, and Sebastian Siebertz. Distributed dominating set approximations beyond planar graphs. *ACM Trans. Algorithms*, 15(3):39:1–39:18, 2019.

[6] Baruch Awerbuch, Boaz Patt-Shamir, and George Varghese. Self-stabilization by local checking and correction (extended abstract). In *32nd Symposium on Foundations of Computer Science (FOCS)*, pages 268–277, 1991.

[7] Alkida Balliu, Gianlorenzo D'Angelo, Pierre Fraigniaud, and Dennis Olivetti. What can be verified locally? *J. Comput. Syst. Sci.*, 97:106–120, 2018.

[8] Marthe Bonamy, Cyril Gavoille, and Michal Pilipczuk. Shorter labeling schemes for planar graphs. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 446–462. SIAM, 2020.

[9] Nicolas Bousquet, Laurent Feuilloley, and Théo Pierron. Local certification of graph decompositions and applications to minor-free classes. In *25th International Conference on Principles of Distributed Systems, OPODIS 2021*, volume 217 of *LIPIcs*, pages 22:1–22:17, 2021.

[10] H. R Brahana. Systems of circuits on two-dimensional manifolds. *Annals of Mathematics*, 23:144–168, 1922.

[11] Keren Censor-Hillel, Ami Paz, and Mor Perry. Approximate proof-labeling schemes. *Theor. Comput. Sci.*, 811:112–124, 2020.

[12] Pierluigi Crescenzi, Pierre Fraigniaud, and Ami Paz. Trade-offs in distributed interactive proofs. In *33rd International Symposium on Distributed Computing (DISC)*, LIPIcs 146, pages 13:1–13:17. Dagstuhl, 2019.

[13] Andrzej Czygrinow and Michał Hańćkowiak. Distributed almost exact approximations for minor-closed families. In *14th Annual European Symposium on Algorithms (ESA)*, pages 244–255, 2006.

[14] Andrzej Czygrinow, Michał Hańćkowiak, Edyta Szymanska, Wojciech Wawrzyniak, and Marcin Witkowski. Distributed local approximation of the minimum k-tuple dominating set in planar graphs. In *18th Int.*

Conference on Principles of Distributed Systems (OPODIS), pages 49–59, 2014.

[15] Andrzej Czygrinow, Michał Hańćkowiak, Edyta Szymanska, Wojciech Wawrzyniak, and Marcin Witkowski. Improved distributed local approximation algorithm for minimum 2-dominating set in planar graphs. *Theor. Comput. Sci.*, 662:1–8, 2017.

[16] Andrzej Czygrinow, Michał Hańćkowiak, and Wojciech Wawrzyniak. Fast distributed approximations in planar graphs. In *22nd Int. Symp. on Distributed Computing (DISC)*, pages 78–92, 2008.

[17] Vida Dujmovic, Louis Esperet, Cyril Gavoille, Gwenaël Joret, Piotr Micek, and Pat Morin. Adjacency labelling for planar graphs (and beyond). *J. ACM*, 68(6):42:1–42:33, 2021.

[18] Gábor Elek. Planarity can be verified by an approximate proof labeling scheme in constant-time. *J. Comb. Theory, Ser. A*, 191:105643, 2022.

[19] Louis Esperet and Benjamin Lévêque. Local certification of graphs on surfaces. *Theor. Comput. Sci.*, 909:68–75, 2022.

[20] Louis Esperet and Sergey Norin. Testability and local certification of monotone properties in minor-closed classes. In *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022*, volume 229 of *LIPIcs*, pages 58:1–58:15, 2022.

[21] Laurent Feuilloley. Bibliography of distributed approximation beyond bounded degree. *CoRR*, abs/2001.08510, 2020.

[22] Laurent Feuilloley, Nicolas Bousquet, and Théo Pierron. What can be certified compactly? compact local certification of MSO properties in tree-like graphs. In Alessia Milani and Philipp Woelfel, editors, *PODC '22: ACM Symposium on Principles of Distributed Computing*, pages 131–140. ACM, 2022.

[23] Laurent Feuilloley, Pierre Fraigniaud, and Juho Hirvonen. A hierarchy of local decision. *Theor. Comput. Sci.*, 856:51–67, 2021.

[24] Laurent Feuilloley, Pierre Fraigniaud, Juho Hirvonen, Ami Paz, and Mor Perry. Redundancy in distributed proofs. *Distributed Comput.*, 34(2):113–132, 2021.

[25] Laurent Feuilloley, Pierre Fraigniaud, Pedro Montealegre, Ivan Rapaport, Éric Rémila, and Ioan Todinca. Compact distributed certification of planar graphs. *Algorithmica*, 83(7):2215–2244, 2021.

[26] Pierre Fraigniaud, Amos Korman, and David Peleg. Towards a complexity theory for local distributed computing. *J. ACM*, 60(5):35:1–35:26, 2013.

[27] Pierre Fraigniaud, Pedro Montealegre, Rotem Oshman, Ivan Rapaport, and Ioan Todinca. On distributed Merlin-Arthur decision protocols. In *26th Int. Colloquium Structural Information and Communication Complexity (SIROCCO)*, LNCS 11639, pages 230–245. Springer, 2019.

[28] Pierre Fraigniaud, Pedro Montealegre, Ivan Rapaport, and Ioan Todinca. A meta-theorem for distributed certification. In Merav Parter, editor, *Structural Information and Communication Complexity - 29th International Colloquium, SIROCCO 2022,*, volume 13298, pages 116–134, 2022.

[29] Pierre Fraigniaud, Boaz Patt-Shamir, and Mor Perry. Randomized proof-labeling schemes. *Distributed Computing*, 32(3):217–234, 2019.

[30] Cyril Gavoille and Nicolas Hanusse. Compact routing tables for graphs of bounded genus. In *26th Int. Coll. on Automata, Languages and Programming (ICALP)*, LNCS 1644, pages 351–360. Springer, 1999.

[31] Mohsen Ghaffari and Bernhard Haeupler. Distributed algorithms for planar networks I: planar embedding. In *ACM Symposium on Principles of Distributed Computing (PODC)*, pages 29–38, 2016.

[32] Mohsen Ghaffari and Bernhard Haeupler. Distributed algorithms for planar networks II: low-congestion shortcuts, MST, and min-cut. In *27th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 202–219, 2016.

[33] Mohsen Ghaffari and Merav Parter. Near-optimal distributed DFS in planar graphs. In *31st Int. Symp. on Distributed Computing (DISC)*, LIPIcs, pages 21:1–21:16. Dagstuhl, 2017.

[34] Mika Göös and Jukka Suomela. Locally checkable proofs in distributed computing. *Theory of Computing*, 12(1):1–33, 2016.

[35] Miikka Hilke, Christoph Lenzen, and Jukka Suomela. Brief announcement: local approximability of minimum dominating set on planar graphs. In *ACM Symposium on Principles of Distributed Computing (PODC)*, pages 344–346, 2014.

[36] Piotr Indyk and Anastasios Sidiropoulos. Probabilistic embeddings of bounded genus graphs into planar graphs. In Jeff Erickson, editor, *Proceedings of the 23rd ACM Symposium on Computational Geometry, Gyeongju, South Korea, June 6-8, 2007*, pages 204–209. ACM, 2007.

[37] Gene Itkis and Leonid A. Levin. Fast and lean self-stabilizing asynchronous protocols. In *35th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 226–239, 1994.

[38] Gillat Kol, Rotem Oshman, and Raghuvansh R. Saxena. Interactive distributed proofs. In *ACM Symposium on Principles of Distributed Computing (PODC)*, pages 255–264, 2018.

[39] Amos Korman, Shay Kutten, and David Peleg. Proof labeling schemes. *Distributed Computing*, 22(4):215–233, 2010.

[40] Fabian Kuhn, Thomas Moscibroda, and Roger Wattenhofer. What cannot be computed locally! In *23rd ACM Symposium on Principles of Distributed Computing (PODC)*, pages 300–309, 2004.

[41] Christoph Lenzen, Yvonne Anne Oswald, and Roger Wattenhofer. What can be approximated locally?: case study: dominating sets in planar graphs. In *20th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 46–54, 2008.

[42] Christoph Lenzen, Yvonne Anne Pignolet, and Roger Wattenhofer. Distributed minimum dominating set approximations in restricted families of graphs. *Distributed Computing*, 26(2):119–137, 2013.

[43] W.S. Massey, J.H. Ewing, F.W. Gerhing, and P.R. Halmos. *A Basic Course in Algebraic Topology*. Graduate Texts in Mathematics. Springer New York, 1991.

[44] B. Mohar and C. Thomassen. *Graphs on Surfaces*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 2001.

[45] Moni Naor, Merav Parter, and Eylon Yogev. The power of distributed verifiers in interactive proofs. In *31st ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1096–115, 2020.

[46] Moni Naor and Larry J. Stockmeyer. What can be computed locally? *SIAM J. Comput.*, 24(6):1259–1277, 1995.

[47] Jaroslav Nesetril and Patrice Ossona de Mendez. *Sparsity - Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and combinatorics*. Springer, 2012.

[48] Ronald Ortner. Embeddability of arrangements of pseudocircles into the sphere. *European Journal of Combinatorics*, 29(2):457–469, 2008.

[49] Torrence D. Parsons, Giustina Pica, Tomaz Pisanski, and Aldo G. S. Ventre. Orientably simple graphs. *Mathematica Slovaca*, 37(4):391–394, 1987.

[50] David Peleg. *Distributed Computing: A Locality-Sensitive Approach*. SIAM, 2000.

[51] David Peleg and Vitaly Rubinovich. A near-tight lower bound on the time complexity of distributed minimum-weight spanning tree construction. *SIAM J. Comput.*, 30(5):1427–1442, 2000.

[52] Henri Poincaré. Sur la généralisation d'un théorème d'Euler relatif aux polyèdres. *C.R. Hebdo. Séances Académie des Sciences*, 117:144–145, 1893.

[53] Atish Das Sarma, Stephan Holzer, Liah Kor, Amos Korman, Danupon Nanongkai, Gopal Pandurangan, David Peleg, and Roger Wattenhofer. Distributed verification and hardness of distributed approximation. *SIAM J. Comput.*, 41(5):1235–1265, 2012.

[54] Wojciech Wawrzyniak. A strengthened analysis of a local algorithm for the minimum dominating set problem in planar graphs. *Inf. Process. Lett.*, 114(3):94–98, 2014.

[55] Wojciech Wawrzyniak. A local approximation algorithm for minimum dominating set problem in anonymous planar networks. *Distributed Computing*, 28(5):321–331, 2015.

[56] J. W. T. Youngs. Minimal imbeddings and the genus of a graph. *Journal of Mathematical Mechanics*, 12:303–315, 1963.