Fully Dynamic Planarity Testing with Applications *

Zvi Galil †
Department of Computer Science
Columbia University, and
Tel-Aviv University

Giuseppe F. Italiano ‡
IBM T. J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Neil Sarnak
IBM T. J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Abstract

The fully dynamic planarity testing problem consists of performing an arbitrary sequence of the following three kinds of operations on a planar graph G: (i) insert an edge if the resultant graph remains planar; (ii) delete an edge; and (iii) test whether an edge could be added to the graph without violating planarity. We show how to support each of the above operations in $O(n^{2/3})$ time, where n is the number of vertices in the graph. The bound for tests and deletions is worst-case, while the bound for insertions is amortized. This is the first algorithm for this problem with sub-linear running time, and it affirmatively answers a question posed in [11].

The same data structure has further applications in maintaining the biconnected and triconnected components of a dynamic planar graph. The time bounds are the same: $O(n^{2/3})$ worst-case time per edge deletion, $O(n^{2/3})$ amortized time per edge insertion, and $O(n^{2/3})$ worst-case time to check whether two vertices are either biconnected or triconnected.

*A preliminary version of this paper appears in [22].

†Work partially supported by NSF Grant CCR-9014605.

‡On leave from Università di Roma, Italy.

1 Introduction

In the last decade there has been a growing interest in dynamic problems on graphs. In particular, much attention has been devoted to the dynamic maintenance of connected components [14, 16, 43] and higher connectivity [8, 10, 18, 19, 20, 21, 32, 35, 53], transitive closure [29, 30, 31, 37, 47, 55], planarity [7, 8, 46], shortest paths [2, 5, 13, 39, 44], and minimum spanning trees [10, 11, 16]. In these problems one would like to answer queries on graphs that are undergoing a sequence of updates, such as insertions and deletions of edges and vertices. The goal of a dynamic graph algorithm is to update efficiently the solution of a problem after dynamic changes, so queries can be answered faster than solving again the problem from scratch. For this reason, devising dynamic algorithms is usually more difficult than dealing with static algorithms and often requires the design of new data structures and new algorithmic techniques. We say that a problem is *fully dynamic* if the update operations include both insertions and deletions of edges. On the other hand, a problem is called *partially dynamic* if only one type of update, i.e., either insertions or deletions, is allowed. Partially dynamic problems that deal with insertions only are called *incremental*.

Planarity testing is a basic problem which has inspired an extensive amount of research in graph theory [9, 34, 54], data structures [4, 7, 15, 46], and sequential [28, 38] as well as parallel [33, 42] algorithms. Informally, a graph is planar if it can be embedded onto the plane without edge crossings. The planarity testing problem consists of answering the question whether a given graph is planar and if so of constructing such an embedding onto the plane. Hopcroft and Tarjan [28] showed that this can be done in $O(n)$ time, where n is the number of vertices in the graph. The *incremental planarity testing problem* consists of performing an intermixed sequence of the following operations on a planar graph G with n vertices. Operation $\mathit{Insert}(x, y)$ adds an edge (x, y) to G provided that the resulting graph remains planar. A $\mathit{Test}(x, y)$ returns *true* if (x, y) can be added to G while maintaining planarity, and it returns *false* otherwise. Di Battista and Tamassia [7, 8] showed how to do incremental planarity testing in $O(\log n)$ [1] amortized time per operation. Later, Westbrook achieved an $O(\alpha(q, n))$ expected bound for biconnected graphs [52], where α is a very slowly growing function (a functional inverse of Ackermann's function), and q is the total number of operations. Recently La Poutré [36] obtained an $O(\alpha(q, n))$ bound for general planar graphs. In the *fully dynamic planarity testing problem* in addition to *Test* and *Insert*, we allow *Delete* operations, where a $\mathit{Delete}(x, y)$ removes the edge (x, y) from G. Notice that an $\mathit{Insert}(x, y)$ as defined above consists of a $\mathit{Test}(x, y)$ which checks whether (x, y) could be inserted without violating planarity, and of a primitive $\mathit{add}(x, y)$ which actually inserts the edge (x, y) if it maintains planarity. In the following, we use the term *update* to refer to either the actual insertion (i.e., an *add* primitive) or the actual deletion (i.e., a *Delete* operation) of an edge in the graph. Planarity testing appears naturally in many applications, such as VLSI layout, graphics, and computer aided design. In all these applications, there seems to be a need for dealing with dynamic updates.

As mentioned in [11], there has been no solution for this problem better than either a repeated application of a the static planarity testing algorithm [28] or of an incremental planarity testing algorithm [36]. (A previous fully dynamic algorithm of Tamassia [46] is committed to an embedding and cannot check whether a dynamic graph remains planar if this embedding changes.) The algorithms in [28, 36] when applied to the fully dynamic setting achieve the following performance. Using the static algorithm [28], one can handle efficiently updates (both edge insertions and edge

[1] In this paper all the logarithms are assumed to be base two.

deletions) in $O(1)$ time, but then it takes $O(n)$ time to perform *Test* operations. On the other hand, the best incremental algorithm [36] supports *Test* operations in $O(\alpha(q,n))$ time but each update can require as much as $O(n)$ time in the worst case.

The main result of this paper is a new data structure which supports each *Test*, *Insert* and *Delete* operation in $O(n^{2/3})$ time. The times for edge deletions and queries are worst-case, while the time for edge insertion is amortized. This can be summarized as shown in Table 1.

	Static [28]	**Incremental** [36]	**Fully Dynamic** *(new)*
Test	$O(n)$	$O(\alpha(q,n))$	$O(n^{2/3})$
Update	$O(1)$	$O(n)$	$O(n^{2/3})$

Table 1

Insertions of vertices and deletions of disconnected vertices can be done in $O(1)$ amortized time, while maintaining the same time performance for the other operations. Using a similar approach, we are able to achieve a fully dynamic algorithm for checking whether two vertices in a planar graph are either biconnected or triconnected, in $O(n^{2/3})$ time per operation. The times for edge deletions and queries are worst-case, while the time for edge insertion is amortized. The same bound for biconnectivity was already known (and with worst-case time also for insertions) [21]; however, the new algorithm is simpler. On the other hand, this is the first fully dynamic algorithm known for maintaining information about triconnectivity in a planar graph.

A fully dynamic setting is different and often much more complex than a partially dynamic setting. In the incremental setting, for example, maintaining a minimum spanning tree can be done in $O(\log m)$ time per insertion using dynamic trees [45], where m is the current number of edges in the graph. Incremental maintenance of the connected components of an undirected graph can be done in $O(\alpha(q,n))$ amortized time per insertion using set-union data structures [49], where q is the total number of operations, and n is the total number of vertices in the graph, Similarly, the incremental maintenance of the 2-edge-connected components of a graph require $O(\alpha(q,n))$ amortized time per operation [53]. However, no better bound than $O(\sqrt{n}\log(m/n))$ is known for all the corresponding fully dynamic problems [10]. Fully dynamic problems on planar graphs become substantially more difficult if changes of the embedding are allowed. We distinguish the two cases by referring to planar graphs with fixed embedding as *embedded planar* or *plane* graphs. For instance, the fully dynamic maintenance of minimum spanning trees and connected components on plane graphs require $O(\log n)$ time per operation [11], while the fully dynamic maintenance of 2-edge connectivity on plane graphs requires $O(\log^2 n)$ time in the worst-case [26]. However, if the graph is planar (i.e., update operations cause changes in the embedding) the best bound for these problems is $O(\sqrt{n})$. This last bound is obtained by specializing to planar graphs the best algorithm known for general graphs. Thus, while it is known how to take advantage of planarity on fully dynamic *plane* graphs, existing fully dynamic algorithms for *planar* graphs seem to be able to merely exploit sparsity.

The techniques used by our new algorithm are the following. First, we maintain a suitable partition of a planar graph into edge clusters, using the separator theorem of Lipton and Tarjan [40]. We keep the time required for updates small by ensuring that each update operation changes at most a constant number of clusters. We remark that *clusterization techniques* are widely used in fully dynamic algorithms [16, 18, 19]. Our main contribution is that we define and maintain a compressed representation of each cluster. Despite its succinctness, this compressed representation certifies the planarity of a cluster by capturing all the possible ways neighbor clusters interact with each other during changes of the embedding. This allows us to carry out *Test* operations without having to look at the entire graph each time.

The rest of the paper consists of seven sections. In Section 2 we give some preliminary definitions and lemmas. Section 3 defines our partition of a planar graph into edge clusters. In Section 4, we show how to compute a compressed representation of each cluster in this partition. Section 5 presents a fully dynamic algorithm for planarity testing. In Section 6 we present our fully dynamic biconnectivity algorithm, and Section 7 deals with fully dynamic triconnectivity. In Section 8 we give some concluding remarks.

2 Preliminaries

We assume that the reader is familiar with graph terminology as in [1]. Given two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, we use $G_1 \cup G_2$ for the graph $G = (V_1 \cup V_2, E_1 \cup E_2)$, but use $G_1 \cap G_2$ for $V_1 \cap V_2$. We now first introduce some definitions and properties of vertex connectivity. Then, we introduce some terminology on planar graphs.

2.1 Vertex connectivity

Let $G = (V, E)$ be an undirected multigraph (i.e. a graph with possibly multiple edges), and let x and y be any two vertices in G. We say that a vertex set $V' \subseteq V - \{x, y\}$ is a *vertex-cut* for vertices x and y if the removal of all the vertices in V' disconnects x and y. We denote by $|V'|$ the cardinality of a vertex-cut V'. A vertex-cut V' for x and y is said to be a *min-vertex-cut* if there is no other vertex-cut V'' for x and y such that $|V''| < |V'|$. Then x and y are k-vertex-connected if and only if a min-vertex-cut for x and y contains at least k vertices. A graph G is said to be *k-vertex-connected* if all its pairs of vertices are k-vertex-connected. A min-vertex-cut of cardinality 1 is called *articulation point*; a graph with no articulation points is said to be *biconnected.* A min-vertex-cut of cardinality 2 is called *separation pair*; a biconnected graph with no separation pairs is said to be *triconnected.* Note that as special cases, a singleton vertex, a graph consisting of single edge, and a triangle consisting of vertices v_1, v_2, v_3 and edges (v_1, v_2), (v_2, v_3), and (v_3, v_1), are all triconnected simple graphs. We refer to such graphs as *trivial triconnected graphs.* It is known (see reference [50]) that a simple triconnected graph with more than three edges has at least six edges and four vertices: we refer to a triconnected simple graph with more than three edges as *non-trivial triconnected graph.* The following lemma is a consequence of the max-flow min-cut theorem (see for instance reference [6]).

Lemma 2.1 *Let $G = (V, E)$ be a non-trivial triconnected graph. Let $V_1, V_2 \subseteq V$ be two vertex sets such that $V_1 \cap V_2 = \emptyset$, and $|V_1| = |V_2| = h \leq 3$. Then there are h vertex-disjoint paths between the h vertices in V_1 and the h vertices in V_2. Similarly, let $G = (V, E)$ is a biconnected graph, and let $V_1, V_2 \subseteq V$ be two vertex sets such that $V_1 \cap V_2 = \emptyset$, $|V_1| = |V_2| = h \leq 2$. Then there are h vertex-disjoint paths between the h vertices in V_1 and the h vertices in V_2.*

In what follows, we define and prove some properties of the biconnected components and of the triconnected components of a multigraph.

Let $G = (V, E)$ be a multigraph with n vertices and m edges. Let $E_1, E_2, \ldots, E_h$ be the partition of E into equivalence classes such that two edges e' and e'' are in the same equivalence class if and only if either (i) $e' = e''$ or (ii) there is a simple cycle (i.e., a cycle with no repetition of edges and vertices) of G containing both e' and e''. For $1 \le i \le h$ let V_i be the set of vertices that are endpoints of the edges in E_i. The subgraphs $G_i = (V_i, E_i)$ are called the *biconnected components* (or *blocks*) of G. An articulation point appears in more than one biconnected component. The set of edges of every graph can be partitioned into biconnected components in a unique way and every edge belongs exactly to one biconnected component. The *block tree* of G is composed of square nodes (corresponding to vertices in G) and round nodes (corresponding to biconnected components in G): whenever a vertex belongs to a given biconnected component, there is a tree edge between the corresponding square and round nodes in the block tree. As a result, each path in a block tree alternates between square and round nodes. We refer to the block tree of G as $T_2(G)$. It is easy to see that $T_2(G)$ has $O(n)$ round and square nodes. Furthermore, the computation of the block tree is immediate once the biconnected components of G have been found. Consequently, the time required to compute $T_2(G)$ is $O(m + n)$ [48]. We associate with each round node ρ of $T_2(G)$ the corresponding biconnected component, referred to as the *pertinent graph of* ρ. The space required by $T_2(G)$ will now be $O(m+n)$, but the time required for its computation remains $O(m+n)$. Note that given $T_2(G)$ it is possible to reassemble the biconnected components into the original graph using the extra information stored in the pertinent graphs.

Let $G = (V, E)$ be a biconnected multigraph. The *triconnected components of* G [27, 50] are defined according to a recursive decomposition of G with respect to its separation pairs. We now sketch this decomposition.

Let $V' \subseteq V$ be a set of vertices of G. Then E can be partitioned into equivalence classes as follows: two edges are in the same class if and only if they are in a same path that does not contain any vertex of V' (except as endpoints). These equivalence classes are called the *separation classes of* G *with respect to* V', or in short *w.r.t.* V'. A class consisting only of one edge is called a *trivial separation class*. Let $E_1, E_2, \ldots, E_p$ be the separation classes of G w.r.t. V'. Each such class E_j defines a *nontrivial bridge* $B_j = (V_j, E_j)$ of V', where B_j is the subgraph of G induced by E_j (that is, $v \in V_j$ if and only if there is an edge of E_j incident on v). An edge (u, v), with u and v both in V', is called a *trivial bridge*. The vertices of a bridge which are elements of V' are called its *attachments*. Note that any two bridges B_i and B_j, $i \neq j$, share at most their attachments.

Let $\{u, v\}$ be a separation pair for G. Note that $\{u, v\}$ is a separation pair if and only if there are either two nontrivial bridges of $\{u, v\}$ or there are at least three bridges, one of which is nontrivial. Let $E' = E_j$ be a separation class of G w.r.t. $\{u, v\}$, and let $E'' = \cup_{i \neq j} E_i$, such that $|E'| \ge 2$, $|E''| \ge 2$, and either the subgraph induced by E' (denoted by G') or the subgraph induced by E'' (denoted by G'') is biconnected. We apply a *split* to G by forming two new graphs G_1 and G_2 from G: $G_1 = G' \cup \{(u, v)\}$ and $G_2 = G'' \cup \{(u, v)\}$. G_1 and G_2 are called the *split graphs of* G *w.r.t.* $\{u, v\}$, and the edge (u, v) associated with the split is called a *virtual edge* in G_1 and G_2. Edges that are not virtual, and therefore correspond to edges of G, are called *actual edges*.

Lemma 2.2 [51] *Let* $G = (V, E)$ *be a biconnected graph, and let* $\{u, v\}$ *be a separation pair of* G*, with split graphs* G_1 *and* G_2*. Then* G_1 *and* G_2 *are biconnected.*

Because of Lemma 2.2, we can recursively apply splits to the split graphs of G. Indeed, the triconnected components of G are obtained by recursively applying splits to the split graphs until no further split is possible. It can be seen that the triconnected components of a multigraph are unique and can be only of three types: a triconnected simple graph (i.e., a triconnected graph with no multiple edges), or a simple cycle (called a *polygon*), or a pair of vertices with at least three edges between them (called a *bond*).

A splitting is denoted by $split(u, v, \sigma)$ where $\{u, v\}$ is the separation pair involved, and σ is a label used to distinguish this split from other splits. The inverse of a split is denoted by $merge(u, v, \sigma)$, and it reassembles two split graphs into one by deleting the virtual edges. By applying all the possible merges to the triconnected components, we obtain the original multigraph G.

The recursive decomposition of a multigraph G into its triconnected components is naturally described by a labeled tree, called the *tree of triconnected components* of G, and referred to as $T_3(G)$. $T_3(G)$ has three types of nodes corresponding to the three types of triconnected components of G: P-nodes (corresponding to polygons), B-nodes (corresponding to bonds), and T-nodes (corresponding to the triconnected subgraphs). We now define $T_3(G)$ according to the recursive decomposition of G into its split graphs. At the beginning, we have a singleton node γ corresponding to G. Let $\{u, v\}$ be a separation pair of G, and let G_1 and G_2 be the two split graphs of G after a $split(u, v, \sigma)$. Then γ is split into two nodes γ_1 and γ_2 (corresponding to G_1 and G_2 respectively) joined by an edge labeled (u, v, σ). We apply recursively these splits to γ_1 and γ_2. If a tree node γ_i correspond to a graph Γ_i that cannot be split, then Γ_i is a triconnected component of G. We label γ_i P, B, or T, depending on whether Γ_i is a polygon, a bond, or a triconnected graph. We associate Γ_i with γ_i. Γ_i is referred to as the *pertinent graph* of γ_i. As implicit in [27], the tree $T_3(G)$ and the set of triconnected components of G are unique and do not depend on the order we split G and its subgraphs.

By definition, an edge (γ_i, γ_j) of $T_3(G)$ labeled (u, v, σ) corresponds to a $split(u, v, \sigma)$. The inverse $merge(u, v, \sigma)$ would contract this edge and merge the two pertinent graphs of γ_i and γ_j. Consequently, applying *merge* operations to all the edges of $T_3(G)$ reassembles the original multigraph G. Furthermore, if m' is the total size of all the pertinent graphs of nodes in $T_3(G)$, then the reassembled graph will have $O(m')$ vertices and edges. Let T be a tree of triconnected components. We denote the graph obtained from T after applying all the merge operations in T as $G_3(T)$. In other words, $G_3(T)$ is a graph whose tree of triconnected component is T. Obviously, $G = G_3(T_3(G))$. We recall that the tree of triconnected components gives a succinct encoding of all the separation pairs of a graph G. The following fact is a consequence of the definition of the tree of triconnected components.

Fact 2.1 *Let $G = (V, E)$ be a biconnected graph and let $T_3(G)$ be the tree of triconnected components of G. No two P-nodes and no two B-nodes are adjacent in $T_3(G)$.*

So far, we have made use of a non-rooted version of the tree of triconnected components. However, it is possible to use a rooted representation of $T_3(G)$ as follows. Root $T_3(G)$ arbitrarily at one of its nodes. Let γ_i be a non-root node of $T_3(G)$ and let γ_j be its parent. In the remainder of the paper, we will refer to the pertinent graph of a node γ_h as Γ_h. Let the edge (γ_i, γ_j) be labeled (s_i, t_i, σ_i). Note that by definition the virtual edge (s_i, t_i) is in Γ_i. We define s_i and t_i as the *poles* of Γ_i and $e_i = (s_i, t_i)$ as the *polar edge* of Γ_i. If γ_i is the root of $T_3(G)$ we choose arbitrarily any edge of Γ_i as the *polar edge*. Let v be a vertex of G and let γ_i be the highest node in $T_3(G)$ such that v

belongs to the pertinent graph of γ_i. Node γ_i is called the *allocation node of* v, and is referred to as $\gamma(v)$. We now prove some properties of allocation nodes.

Lemma 2.3 *Let G be a biconnected graph, and let $T_3(G)$ be its rooted tree of triconnected components. Let v be a vertex of G, and let γ_j be a node in $T_3(G)$ such that its pertinent graph Γ_j contains v. Let $\gamma(v)$ be the allocation node of v. Then v is a pole in the pertinent graphs of all nodes in the path of $T_3(G)$ between γ_j and $\gamma(v)$ ($\gamma(v)$ excluded).*

Proof: Define $\gamma_i = \gamma(v)$. Since Γ_j contains v, by definition of allocation node, γ_i must be an ancestor of γ_j. If $\gamma_i = \gamma_j$, then the path of $T_3(G)$ between these two nodes contains no node (recall that γ_i is excluded), and the lemma is trivially true. So, assume that γ_i is a proper ancestor of γ_j. Let γ_h and γ_k be two consecutive nodes on the path between γ_j and γ_i, γ_k the parent of γ_h. Recall that the only common vertices in the two pertinent graphs Γ_h and Γ_k are the two vertices in the split pair (separating Γ_h and Γ_k), and that these two vertices are by definition the poles of Γ_h. Since $v \in \Gamma_j$ and $v \in \Gamma_i$, this implies that v must be a pole in Γ_h. This proves that every node $\gamma_h \neq \gamma_i$ in the path between γ_j and γ_i must have v as a pole. $\Box$

Recall that by definition the edge leaving node γ_i in $T_3(G)$ is labeled with the poles of Γ_i. Therefore, a corollary of Lemma 2.3 is that all the edges in the path between γ_j and $\gamma(v)$ are labeled with v.

Lemma 2.3 gives a linear-time algorithm for computing the allocation nodes of all the vertices of a given graph G. We simply visit $T_3(G)$ in a top-down fashion, starting from its root γ_r. γ_r is the allocation node of all the vertices of Γ_r. Each time we visit a non-root node γ_i entering through edge (a_i, b_i, σ_i), γ_i is the allocation node of all the vertices of Γ_i but a_i and b_i. If G has m vertices and n edges, the total time needed to compute all the allocation nodes of vertices of G is $O(n)$.

We conclude this section by mentioning that the total number of nodes in $T_3(G)$ is $O(n)$ and the total space required by $T_3(G)$ (including the pertinent graphs) is $O(m+n)$. As in the case of $T_2(G)$, also $T_3(G)$ can be easily computed while computing the triconnected components of G. Therefore the total time required to compute $T_3(G)$ is $O(m+n)$. Figure 1 shows a tree of triconnected components of a biconnected graph.

[Figure 1]

2.2 Planar graphs

A graph is *planar* if it can be drawn in the plane such that no two edge cross (except at their endpoints). Such a drawing is called an *embedding* of the planar graph. We represent embeddings by assigning to each vertex v a cyclic ordering of the edges leaving v, as defined by Guibas and Stolfi in [24]. For each edge $e = (u, v)$ of the planar graph G, define two directed versions of e: one going from u to v, and another (symmetric) going from v to u. Denote a directed edge by putting an arrow on it: $\vec{e}$. For each directed edge $\vec{e}$ denote its left and right faces $Left(\vec{e})$ and $Right(\vec{e})$ respectively. Given a vertex v, the *edge ring of* v is defined as the circular list (ordered counterclockwise) of directed edges originating from v. We will refer to this list as $EdgeRing(v)$. For each vertex v, define the *face ring of* v as the circular list (ordered counterclockwise) of faces that are on the right of edges in $EdgeRing(v)$. We refer to this list as $FaceRing(v)$. Given a face f, we define the *contour* of f as the cycle of directed edges $\vec{e}$ such that $Left(\vec{e}) = f$.

We now characterize topological properties of min-vertex-cuts in the embedding of a planar graph.

Lemma 2.4 [51] *The contour of any face of an embedded planar biconnected graph is a simple cycle.*

Lemma 2.5 [51] *The contour of any face of an embedded planar 2-edge-connected graph is a cycle (not necessarily simple).*

Notice that Lemma 2.4 can be rephrased by saying that vertex v is an articulation point of an embedded planar graph G if and only if there is a face f that appears at least twice in $FaceRing(v)$. Equivalently, v is an articulation point if and and only if it appears twice in the contour of a face. Similarly, Lemma 2.5 is equivalent to saying that an edge e is a bridge if and only if it appears twice in the contour of a face, or if and only if $Left(\vec{e}) = Right(\vec{e})$.

Lemma 2.6 [51] *Let G be an embedded planar graph, and let $\{v_1, v_2\}$ be a pair of vertices of G. Then $\{v_1, v_2\}$ is a separation pair of a biconnected component of G if and only if there exist two faces f_1 and f_2 such that the following is true:*

(i) v_1 and v_2 are on both f_1 and f_2; and

(ii) v_1 and v_2 are not adjacent in at least one of f_1 and f_2.

The definition of tree of triconnected components is closely related to the decomposition trees of planar graphs given in [3, 7]. Indeed, there is a close relationship between triconnectivity and planarity: it is well known that a triconnected planar graph G has a unique embedding in the sphere [25]. This implies that a triconnected planar graph has essentially only one embedding in the plane too (recall that an embedding in the plane can be obtained from an embedding in the sphere by simply choosing one face as the external face). Consequently, it is meaningful to talk about faces of a triconnected planar graph G, since they do not change with the embedding.

It is known that the tree of triconnected components of a biconnected planar graph G encodes all the possible embeddings of G. Given an embedding of a biconnected graph G, another embedding of G can be found by arbitrarily applying any sequence of the following operations: (i) arbitrarily permuting the separation classes of a bond, and (ii) reversing any separation class (i.e., flipping it over) around the corresponding separation pair. This was the base of the incremental planarity testing algorithm in [7, 8]. In case of a biconnected planar graph G, however, we can have another representation of $T_3(G)$, where each pertinent graph has an orientation of its edges. Before giving this representation, we need more definitions.

A *planar st–graph* $G = (V, E)$ is a planar acyclic directed graph with one source s and one sink t, such that there is an embedding of G having s and t in the contour of the same face [38]. A *planar st–orientation* of an undirected graph G is an orientation of the edges of G yielding a planar st–graph. It is known that every biconnected planar graph has an st–orientation, and such an orientation can be found in $O(n)$ time [15]. Note that each pertinent graph in $T_3(G)$ is biconnected because of Lemma 2.2. Consequently, each pertinent graph Γ_i in $T_3(G)$ has an st–orientation.

Fact 2.2 [7] *Let $G = (V, E)$ be a biconnected graph, with a given st–orientation with source s and sink t. Let $T_3(G)$ be the tree of triconnected components of G. Let γ_i be any node in $T_3(G)$, and let Γ_i be the pertinent graph of γ_i. The st–orientation of G induces an st–orientation of Γ_i, as follows. If γ_i is a non-root node, the source and sink of Γ_i are respectively its poles s_i and t_i. If γ_i is the root, the source and sink of Γ_i are respectively s and t.*

Fact 2.3 *Let $G = (V, E)$ be a directed st–planar graph, and let $e = (a, b)$ be an edge in G, directed from a to b. Then there is in G a path $p_{s,a}$ from s to a, and a path $p_{b,t}$ from b to t, such that $p_{s,a}$ and $p_{b,t}$ are vertex-disjoint. If a, b, s and t are all in the contour of a same face of G, then also $p_{s,a}$ and $p_{b,t}$ can be chosen in the contour of a same face.*

We now introduce some more terminology on planar graphs. Given a biconnected planar graph $G = (V, E)$, and an edge $e_1 \in E$, we say that an edge $e_2 \neq e_1$ [respectively a vertex v] of G is *co-facial with* e_1 if and only if there is an embedding of G where e_1 and e_2 [respectively v] appear in the contour of the same face. Let $T_3(G)$ be the tree of triconnected components of a biconnected graph, rooted arbitrarily at one of its nodes. Let γ_i be a node of $T_3(G)$, Γ_i the pertinent graph of γ_i, and $e_i = (s_i, t_i)$ the polar edge of Γ_i. We say that an edge e [respectively a vertex v] of Γ_i is *peripheral* if there is an embedding of Γ_i having e [respectively v] and e_i in the contour of the same face. We denote by $\Pi(\Gamma_i)$ (referred to as the *periphery of* Γ_i) the vertices and edges (excluding virtual edges) that are peripheral in Γ_i. Let γ_i be a non-root node of $T_3(G)$ and let γ_j be its parent. Let the edge between γ_i and γ_j be labeled (s_i, t_i, σ_i) so that $e_i = (s_i, t_i)$ is the polar edge of Γ_i. We say that γ_i is a *peripheral* node if edge (s_i, t_i) is peripheral in Γ_j, the pertinent graph of γ_j. Note that all the children of either a P- or a B-node are peripheral. Let γ_i be any node of $T_3(G)$. The *periphery of* γ_i, denoted by $\Pi(\gamma_i)$, is recursively defined as follows. If γ_i is a leaf, $\Pi(\gamma_i) = \Pi(\Gamma_i)$. If γ_i is a non-leaf, let $\gamma_{i,1}$, $\gamma_{i,2}$, …, $\gamma_{i,p}$, $p \geq 1$, be the peripheral children of γ_i. Then the periphery of γ_i is defined as $\Pi(\gamma_i) = \Pi(\Gamma_i) \cup [\cup_{j=1}^{p} \Pi(\gamma_{i,j})]$.

Let γ_i be a node of $T_3(G)$, and let e_i be its polar edge. Let T_{γ_i} be the subtree of $T_3(G)$ rooted at γ_i. Let $G_3(T_{\gamma_i})$ be the graph obtained after applying all the merges to T_{γ_i}. By definition of tree of triconnected components, $G_3(T_{\gamma_i})$ contains edge e_i.

Fact 2.4 *$\Pi(\gamma_i)$ consists exactly of all and only the vertices and edges of $G_3(T_{\gamma_i})$ that are co-facial with the polar edge e_i.*

Let λ be a simple cycle of a biconnected graph G, and let β_1, β_2, …, β_h be the bridges of G w.r.t. λ. We say that two bridges β_i and β_j *interlace* if either (i) both β_i and β_j have two attachments ($\{u_i, v_i\}$ and $\{u_j, v_j\}$ respectively), which are all distinct and appear on λ in the order u_i, u_j, v_i, v_j; or (ii) β_i and β_j share three attachments. We say that three bridges interlace if each two of them interlace. The bridges of separation pairs and of simple cycles are crucial to planarity, as the following lemmas show.

Lemma 2.7 [12] *Let $G = (V, E)$ be a biconnected graph. Let λ be a simple cycle of G, and let B_1, B_2, …, B_p be the bridges of G w.r.t. λ. $\lambda \cup B_1 \cup B_2 \cup \ldots \cup B_p$ has a planar embedding in which all these bridges are internal to λ if and only if the following two conditions are satisfied:*

(i) $\lambda \cup B_i$ is planar, $1 \leq i \leq p$; and

(ii) no two bridges interlace,

The following lemma is a corollary of Lemma 2.7.

Lemma 2.8 [12] *Let G be a biconnected graph, and λ be a simple cycle in G. G is planar if and only if the bridges B_1, B_2, ..., B_p of G w.r.t. λ satisfy the following two conditions:*

(i) $\lambda \cup B_i$ is planar, $1 \leq i \leq p$.

(ii) The set of bridges can be partitioned into two subsets, such that no two bridges in the same subset interlace.

Lemma 2.9 [12] *A graph is planar if and only if its biconnected components are planar.*

Lemma 2.10 [12] *Let $G = (V, E)$ be a biconnected graph with a separation pair $\{a, b\}$, and let B_1, B_2, ..., B_p be the bridges of G w.r.t. $\{a, b\}$. Then G is planar if and only if for $1 \leq i \leq p$ $B_i \cup \{(a, b)\}$ is planar.*

Lemma 2.11 [51] *Let $G = (V, E)$ be a biconnected graph with a separation pair $\{a, b\}$. Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two split graphs of G with respect with $\{a, b\}$. If there is a planar embedding of G_1 with virtual edge (a, b) between faces f_1' and f_1'', and a planar embedding of G_2 with virtual edge (a, b) between faces f_2' and f_2'', then there is a planar embedding of G having two faces f_1 and f_2 defined as follows. The contour of f_1 is given by deleting virtual edges (a, b) and merging the remaining contours of f_1' and f_1'' ($f_1 = (f_1' - \{(a, b)\}) \cup (f_1'' - \{(a, b)\})$). Similarly, the contour of f_2 is given by deleting virtual edges (a, b) and merging the remaining contours of f_2' and f_2'' ($f_2 = (f_2' - \{(a, b)\}) \cup (f_2'' - \{(a, b)\})$).*

Conversely, assume that there is a planar embedding of G with the two vertices a and b of a separation pair in the contour of two faces f_1 and f_2. Then there is a planar embedding of G_1 with virtual edge (a, b) between faces f_1' and f_2' and a planar embedding of G_2 with virtual edge (a, b) between faces f_1'' and f_2'', such that $f_1 = (f_1' - \{(a, b)\}) \cup (f_1'' - \{(a, b)\})$ and $f_2 = (f_2' - \{(a, b)\}) \cup (f_2'' - \{(a, b)\})$.

Notice that in both Lemmas 2.8 and Lemma 2.10, the necessary and sufficient conditions for planarity hold only for biconnected graphs. However, because of Lemma 2.9 they are meaningful also for non-biconnected graphs. Indeed, since each biconnected component of a planar graph can be embedded independently of the others, the conditions of Lemmas 2.8 and Lemma 2.10 must hold for every biconnected component of a non-biconnected graph. For instance, combining Lemmas 2.9 and 2.10 gives the following corollary.

Corollary 2.1 *Let $G = (V, E)$ be a graph, let G' be a biconnected component of G, let $\{a, b\}$ be a separation pair of G', and let B_1, B_2, ..., B_p be the bridges of G' w.r.t. $\{a, b\}$. Then G is planar if and only if:*

(i) For $1 \leq i \leq p$ $B_i \cup \{(a, b)\}$ is planar.

(ii) All the other biconnected components of G (different from G') are planar.

We now prove some lemmas that will be used throughout the paper.

Lemma 2.12 *Let G be a non-trivial triconnected simple planar graph. Then the following is true:*

(i) Any two faces of G share at most two vertices.

(ii) If faces f_1 and f_2 share two vertices v_1 and v_2, then v_1 and v_2 are adjacent in both faces f_1 and f_2.

Proof: Since G is triconnected, there is essentially only one embedding of G, and the faces of G are well defined. Condition (ii) is a simple consequence of Lemma 2.6. Therefore, to prove the lemma we only need to prove condition (i).

Assume by contradiction that two faces of G, say f_1 and f_2, share three vertices, say v_1, v_2 and v_3. We have only two possible cases: either (a) there are two such vertices which are not adjacent in either f_1 or f_2, or (b) v_1, v_2, and v_3 are all adjacent in both f_1 and f_2. Note that case (a) is not possible, since by Lemma 2.6 G would not be triconnected. Consequently, v_1, v_2, and v_3 must be all adjacent in both f_1 and f_2. We further distinguish two subcases: either (b1) v_1, v_2 and v_3 appear only once in the contour of faces f_1 and f_2, or (b2) there is an i, $1 \leq i \leq 3$ and a j, $1 \leq j \leq 2$, such that vertex v_i appears twice in the contour of face f_j. Recall that G is triconnected, and therefore Lemma 2.4 rules out case (b2).

In summary, vertices v_1, v_2, and v_3 must be all adjacent in both faces f_1 and f_2, and furthermore they must appear only once in the contour of the two faces. Since v_1, v_2, and v_3 are adjacent, there are edges $e_1 = (v_1, v_2)$, $e_2 = (v_2, v_3)$, and $e_3 = (v_3, v_1)$ in G. Because v_1, v_2 and v_3 appear only once in the contour of f_1 and f_2, both these contours consist of edges e_1, e_2, and e_3. This implies that G is a triangle, contradicting the fact that G is a non-trivial triconnected graph. □

The following is a corollary of Lemma 2.12.

Corollary 2.2 *Let $G = (V, E)$ be any planar non-trivial triconnected simple graph. Then no two edges of G can be both in the contour of two faces in the embedding of G.*

Define an operation that replaces an edge in a graph with a path whose intermediate vertices are all new. Denote this operation as a *chain replacement*. We say that two graphs G_1 and G_2 are *homeomorphic* if both can be obtained from the same graph by means of chain replacements. An immediate consequence of this definition is that two homeomorphic graphs have essentially the same embeddings (replacing chains with edges).

Lemma 2.13 *Let $G = (V, E)$ be a graph. Let $G' = (V', E')$ be a planar subgraph of G that is homeomorphic to a triconnected simple graph. Let B_1, B_2, ..., B_p be the bridges of G w.r.t. V', such that $G' \cup B_i$ is planar, $1 \leq i \leq p$. Then for B_i, $1 \leq i \leq p$, all the attachments of B_i are in a same face of G'. Furthermore, all the planar embeddings of $G' \cup B_i$ have B_i inside a face containing all the attachments of B_i.*

Proof: Consider a planar embedding of $G' \cup B_i$. Since G' is homeomorphic to a triconnected graph, there is only one planar embedding for G'. By definition of bridge, two vertices of B_i cannot be in different faces of G' because in this case any path between them must cross edges of G'. Thus, B_i must be embedded inside a face of G'. Of course, the contour of this face must contain all the attachments of G'. □

Lemma 2.14 *Let $G = (V, E)$ be a graph. Let $G' = (V', E')$ be a subgraph of G that is homeomorphic to a non-trivial triconnected graph. Let B_1, B_2, ..., B_p be the bridges of G w.r.t. V'. Let $f_1, f_2, \ldots, f_q$ be the faces of G'. Then G is planar if and only if the following two conditions hold.*

(i) for $1 \leq i \leq p$, $G' \cup B_i$ is planar; and

(ii) The set of bridges can be partitioned into q subsets $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_q$, such that all the bridges in $\mathcal{F}_i$, $1 \leq i \leq q$, have attachments only on the contour of face f_i, and do not interlace in the contour of f_i.

Proof: We first assume that conditions (i) and (ii) hold, and show that G is planar. By condition (ii), for each $1 \leq i \leq p$, the attachments of bridge B_i are all in the contour of the same face. Since G' is homeomorphic to a triconnected graph, it has no articulation points. By Lemma 2.4 the contour of each face of G' is a simple cycle. By Lemma 2.7 and condition (ii), all the bridges with attachments on the contour of face f can be embedded inside f. This gives a planar embedding of G.

Conversely, assume that G is planar, and consider a planar embedding of G. Clearly condition (i) must hold, and by Lemma 2.13 all the attachments of each B_i are in the contour of the same face. We now prove condition (ii). For $1 \leq i \leq q$, define $\mathcal{F}_i$ to be the set of all bridges of G w.r.t. V' that are embedded inside face f_i of G'. Again, since G' is homeomorphic to a triconnected graph, the contour of each face of G' is a simple cycle. Then by Lemma 2.7, the partition of the bridges into $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_q$ satisfies condition (ii). $\Box$

We end this section by describing some transformations that, given a planar embedding of a graph, yield a new embedding which is itself planar. These transformations will be used frequently throughout the paper.

(1) *Edge Contraction*: Contract edge (u, v) identifying $u \equiv v$.

(2) *Biconnected Expansion*: Let u be a vertex of G. Add a new planar biconnected component having u as the only articulation point.

(3) *Nontrivial Bridge Contraction.* Let B_i be a nontrivial bridge of separation pair $\{u, v\}$: replace B_i with edge (u, v).

(4) *Trivial Bridge Expansion.* Let (u, v) be an edge of G: replace $\{u, v\}$ with a nontrivial bridge B_i having u and v as attachments and such that $B_i \cup \{(u, v)\}$ is planar.

(5) *Parallel Edge Contraction*: Let $e_1, e_2, \ldots, e_p$ be parallel edges between vertices u and v: replace $e_1, e_2, \ldots, e_p$ with only one edge e_0 between u and v.

(6) *Parallel Edge Expansion*: Let $e_0 = (u, v)$ be an edge of G: replace e_0 with $p \geq 1$ parallel edges $e_1, e_2, \ldots, e_p$.

Note that transformation (4) is the inverse of (3), and (6) is the inverse of (5).

3 The Partition into Clusters

Let G be any planar graph with n vertices having non-negative vertex costs summing to no more than one. Then the separator algorithm of Lipton and Tarjan [40] partitions the vertices of G into three sets A, B, and C such that no edge joins a vertex in A with a vertex in B, neither A nor B has total cost exceeding $\frac{2}{3}$, and C contains no more than $2\sqrt{2n}$ vertices.

We now describe our partition into edge clusters which is reminiscent of the one used by Frederickson in [17]. The *size* of a cluster is given by the number of edges it contains. Let $\mathcal{C}$ be a cluster. If only edges of $\mathcal{C}$ are incident to a vertex v, we say that v is *internal* to $\mathcal{C}$. If both edges in $\mathcal{C}$ and edges not in $\mathcal{C}$ are incident to v, we say that v is a *boundary vertex of* $\mathcal{C}$. For any cluster $\mathcal{C}$, we denote by $B(\mathcal{C})$ the total number of boundary vertices in $\mathcal{C}$. Let z be a positive integer to be fixed later. Given a planar graph G, we maintain a partition of G into $O(\frac{n}{z})$ clusters of size $O(z)$ each, and such that $\sum_{\mathcal{C}} B(\mathcal{C}) = O(\frac{n}{\sqrt{z}})$. Note that $\sum_{\mathcal{C}} B(\mathcal{C})$ gives the total number of boundary vertices in the partition (each counted with its multiplicity). This partition can be computed using the recursive approach described in [41]. At the beginning, we start with only one cluster corresponding to the whole graph and having only internal vertices. Each time we have a cluster whose size is greater than z, we apply the planar separator theorem to this cluster (with uniform vertex costs) and replace it by two new clusters: one having edges in $A \cup C$ and the other having edges in $B \cup C$. Edges whose endpoints are both in C can be given to either $A \cup C$ or $B \cup C$. Note that clusters need not to be connected. As shown in [17], this yields the desired partition into $O(\frac{n}{z})$ clusters of size $O(z)$ each, having a total of $O(\frac{n}{\sqrt{z}})$ boundary vertices. Since a planar separator can be found in linear time [40], the total time required by this computation is easily seen to be $O(n \log n)$. Goodrich showed how to speed up this computation to $O(n)$ time [23].

4 The Compressed Representation of a Cluster

Let $\mathcal{C}$ be a cluster, and let $B(\mathcal{C})$ be the total number of boundary vertices in $\mathcal{C}$. In this section we describe how to compute a compressed representation of $\mathcal{C}$, denoted by $\widetilde{\mathcal{C}}$, which is the basic building block of our fully dynamic algorithm. Informally, $\widetilde{\mathcal{C}}$ will be a contraction of $\mathcal{C}$ that contains all its boundary vertices, plus only $O(B(\mathcal{C}))$ extra vertices and edges. Despite its succinctness, $\widetilde{\mathcal{C}}$ will still capture all the ways in which neighbor clusters interact with $\mathcal{C}$ during possible changes of the embedding of the graph.

Formally, let $G = (V, E)$ be a planar graph, and let $G_0 = (V_0, E_0)$ be a connected subgraph of G ($V_0 \subseteq V$, $E_0 \subseteq E$). Define $\overline{E_0} = E - E_0$, and let $\overline{G_0} = (\overline{V_0}, \overline{E_0})$ be the graph induced by $\overline{E_0}$ ($v \in \overline{V_0}$ if and only if there is an edge $e \in \overline{E_0}$ incident to v). We refer to $\overline{G_0}$ as the *complement of* G_0 *in* G. Denote by $V_{G_0,G}$ the set of the vertices of V to which both edges from E_0 and $\overline{E_0}$ are incident. We call $V_{G_0,G}$ the *boundary of* G_0. Note that while $E_0 \cap \overline{E_0} = \emptyset$, we have that $V_0 \cap \overline{V_0} = V_{G_0,G}$. If $v \in \overline{V_0}$, we say that v is *external* to G_0. Notice that also the boundary of G_0 is considered to be external to G_0. Let $G_1 = (V_1, E_1)$ be a graph that contains all the vertices in the boundary of G_0 (i.e., $V_{G_0,G} \subseteq V_1$) but no other vertex of $\overline{G_0}$ (i.e., $V_1 \cap (V - V_0) = \emptyset$). Then substituting G_1 in place of G_0 in G gives rise to a new graph $G' = (V', E')$, with vertex set $V' = (V - V_0) \cup V_1$ and edge set $E' = \overline{E_0} \cup E_1$. Let G' be $\overline{G_0} \cup G_1$. Fix two vertices x and y external to G_0. If it happens that $G \cup \{(x, y)\}$ is planar if and only if $G' \cup \{(x, y)\}$ is planar, we say that the substitution of G_1 in place of G_0 in G is *planarity-preserving for* x *and* y. If this substitution is planarity-preserving for any pair of vertices external to G_0, we simply say that it is *planarity-preserving*.

Let $\mathcal{C}$ be a cluster and let $B(\mathcal{C})$ be its number of boundary vertices. We say that a graph $\widetilde{\mathcal{C}}$ *is a valid compression of* $\mathcal{C}$ if and only if it satisfies the following three properties.

(i) $\widetilde{\mathcal{C}}$ contains all the boundary vertices of $\mathcal{C}$, and no other vertex of G external to $\mathcal{C}$; and

(ii) $\widetilde{\mathcal{C}}$ has at most $O(B(\mathcal{C}))$ vertices and edges; and

(iii) the substitution of $\widetilde{\mathcal{C}}$ in place of $\mathcal{C}$ is planarity-preserving.

We now give an algorithm to compute a valid compression $\widetilde{\mathcal{C}}$ of a cluster $\mathcal{C}$. The invariant we will maintain is that $\widetilde{\mathcal{C}}$ can be computed by means of a sequence of substitutions, each of which is planarity-preserving. The computation of $\widetilde{\mathcal{C}}$ is done in two steps. First, in Section 4.1 we compute the forest of block trees of $\mathcal{C}$ (recall that $\mathcal{C}$ might not be connected) and prune it by leaving only $O(B(\mathcal{C}))$ biconnected components. Second, in Section 4.2 we show how to compress each remaining biconnected component. These two steps will be combined in Section 4.3 to give a valid compression of a cluster.

4.1 Step 1: Compressing a Block Tree

We now describe an algorithm to perform the first step. Without loss of generality we assume that the cluster $\mathcal{C}$ is connected. Otherwise we can apply the same algorithm to all the connected components of $\mathcal{C}$. Given $\mathcal{C}$, we compute $T_2(\mathcal{C})$, the block tree of $\mathcal{C}$. Since $\mathcal{C}$ has $O(z)$ vertices and edges, $T_2(\mathcal{C})$ has size $O(z)$.

Color *red* the square nodes of $T_2(\mathcal{C})$ that correspond to the boundary vertices of $\mathcal{C}$, and color *black* all the other nodes. Note that round nodes are always black. Define the *degree* of a node (both round and square) of $T_2(\mathcal{C})$ to be the number of nodes adjacent to it. Define the *restricted degree* of a round node ρ to be its degree minus the number of black leaves adjacent to ρ.

We compress $T_2(\mathcal{C})$, obtaining a new block tree whose number of round nodes will be linear in the number of red (square) nodes. There are two reasons why the number of round nodes of $T_2(\mathcal{C})$ may not be linear in the number of red nodes. First, there can be round nodes of restricted degree one. Second, even if there are no round nodes of restricted degree one, there may be chains alternating between round nodes of restricted degree two and black square nodes of degree two; we refer to maximal chains of this kind as *black chains*. We first delete round nodes of restricted degree one by applying repeatedly the following rule.

(B1) If there is a round node ρ of degree r which is adjacent to $r-1$ (square) leaves, all of which are black, then delete ρ and its $r-1$ adjacent square leaves (see Figure 2).

Lemma 4.1 *Rule (B1) gives rise to planarity-preserving substitutions in* $\mathcal{C}$.

Proof: The interpretation of the above rule in the original cluster $\mathcal{C}$ is the following. Rule (B1) takes a biconnected subgraph G_1 of $\mathcal{C}$ that has exactly one articulation point v, and such that the only possible boundary vertex in G_1 is v. Then v is substituted in place of G_1. Let $\widehat{G}$ be the graph obtained from G after this substitution.

Let x and y be any two vertices external to cluster $\mathcal{C}$, and assume that $G \cup \{(x, y)\}$ is planar, i.e., there is a planar embedding of G that has x and y in the contour of the same face. Since $\widehat{G}$

can be obtained from G by applying edge contractions and parallel edge contractions, and such contractions preserve planarity, the planar embedding of $\widehat{G}$ induced by these contractions has x and y in the contour of the same face. Therefore $\widehat{G} \cup \{(x, y)\}$ is planar too. Conversely, assume $\widehat{G} \cup \{(x, y)\}$ planar. Since $G \cup \{(x, y)\}$ can be obtained from $\widehat{G} \cup \{(x, y)\}$ by means of a biconnected expansion, $G \cup \{(x, y)\}$ must be planar too. □

We now show how to compress black chains. Let $\pi = \{v_0, \rho_1, v_1, \rho_2, \ldots v_{p-1}, \rho_p, v_p\}$ be such a chain; namely, ρ_i, $1 \leq i \leq p$, is a round node of restricted degree two, v_i, $1 \leq i \leq p-1$, is a degree-two black square node, and v_0 and v_p are either red nodes or black nodes of degree greater than two. For $1 \leq i \leq p$, let G_i be the pertinent graph of ρ_i. We refer to $G(\pi) = G_1 \cup G_2 \cup \ldots \cup G_p$ as the *graph corresponding to the black chain* π. We define v_0 and v_p to be the *terminals* of the black chain π. We say that a round node ρ_i is *free* if the graph $G_i \cup \{(v_{i-1}, v_i)\}$ is planar. We define ρ_i to be *blocking* otherwise. A black chain π is said to be *free* if it contains only free round nodes. Otherwise, π contains at least one blocking round node and it is referred to as a *blocking chain.* Free and blocking chains are substantially different with respect to planarity, as the following lemma shows.

Lemma 4.2 *Let $\pi = \{v_0, \rho_1, v_1, \rho_2, \ldots v_{p-1}, \rho_p, v_p\}$ be a black chain, and let G_i be the pertinent graph of ρ_i, $1 \leq i \leq p$. Let $G(\pi) = G_1 \cup G_2 \cup \ldots \cup G_p$ be the graph corresponding to π. $G(\pi) \cup \{(v_0, v_p)\}$ is planar if and only if for all $1 \leq i \leq p$ $G_i \cup \{(v_{i-1}, v_i)\}$ is planar.*

Proof: Let $\widetilde{G} = G(\pi) \cup \{(v_0, v_p)\}$. Assume $\widetilde{G}$ is planar. Note that for any i, $1 \leq i \leq p$, $G_i \cup \{(v_{i-1}, v_i)\}$ can be obtained from $\widetilde{G}$ by means of edge contractions and parallel edge contractions. Since such contractions preserve planarity, $G_i \cup \{(v_{i-1}, v_i)\}$, $1 \leq i \leq p$, must be planar too.

To prove the converse assume that for $1 \leq i \leq p$, $G_i \cup \{(v_{i-1}, v_i)\}$ is planar. Consider the simple cycle λ consisting of vertices $v_0, v_1, \ldots, v_p$ and edges $(v_i, v_{(i+1) \bmod (p+1)})$, $0 \leq i \leq p$. Clearly, λ is planar. Note that $\widetilde{G}$ can be obtained from λ by replacing edge (v_{i-1}, v_i) with G_i, $1 \leq i \leq p$. Due to the planarity of $G_i \cup \{(v_{i-1}, v_i)\}$, each such transformation is a trivial bridge expansion. Consequently, $\widetilde{G}$ must be planar too. □

Lemma 4.2 states that a black chain π is free if and only if there is an embedding of its corresponding graph $G(\pi)$ having both terminals v_0 and v_p in the contour of the same face.

Let π be a chain with terminals v_0 and v_p. Given a round node ρ_i in π with pertinent graph G_i, $1 \leq i \leq p$, we denote by $G_i(v_0, v_p)$ the graph obtained from G_i by replacing vertex v_{i-1} with vertex v_0 and vertex v_i with vertex v_p. If π is blocking, we arbitrarily choose a blocking node ρ_j in π, and we refer to such node as the *witness* of π. Our technique to compress black chains is the following.

(B2) If the black chain π is free, then choose arbitrarily a node ρ_i, $1 \leq i \leq q$ in the chain. $G(\pi) = G_1 \cup G_2 \cup \ldots \cup G_p$ is replaced by $G_i(v_0, v_p)$. If the black chain is a blocking chain with witness ρ_j, replace $G(\pi)$ by $G_j(v_0, v_p)$ (see Figure 2).

[Figure 2]

Lemma 4.3 *The compression of a black chain achieved by rule (B2) gives rise to planarity-preserving substitutions in $\mathcal{C}$.*

Proof: Let π be a black chain between square nodes v_0 and v_p in the block tree. Let G be the graph before compressing the black chain π, and let $\widehat{G}$ be the graph obtained after compressing π by means of rule (B2). Let x and y be any two vertices external to cluster $\mathcal{C}$. We want to show that $G \cup \{(x,y)\}$ is planar if and only if $\widehat{G} \cup \{(x,y)\}$ is planar.

Let v_i, $0 \leq i \leq p$, be the square nodes in π, and let ρ_i, $1 \leq i \leq p$, be the round nodes (all of restricted degree two) in π, such that ρ_i is adjacent to v_{i-1} and v_i. For $1 \leq i \leq p$, let $G_i = (V_i, E_i)$ be the pertinent graph associated with round node ρ_i. Note that G can be decomposed as $G = G_0 \cup G(\pi)$, where $G(\pi) = G_1 \cup G_2 \cup \ldots \cup G_p$ consists of the biconnected subgraphs (corresponding to the round nodes) of the black chain π, and G_0 corresponds to the rest of the graph. Since there are no red nodes in π, the only possible boundary vertices of $V_1 \cup V_2 \cup \ldots \cup V_p$ are v_0 and v_p. As a result, any bridge of G w.r.t. $V_1 \cup V_2 \cup \ldots \cup V_p$ (the vertex set of $G(\pi)$) has attachments only in $\{v_0, v_p\}$ (i.e., $G_0 \cap G(\pi) = \{v_0, v_p\}$). Since x and y are external to $\mathcal{C}$, $x, y \in G_0$.

After compressing the black chain by means of rule (B2), we have that $\widehat{G} = G_0 \cup G_j(v_0, v_p)$ for some j, $1 \leq j \leq p$, and consequently $\widehat{G}$ can be obtained from G by simply replacing $G(\pi)$ with $G_j(v_0, v_p)$. Let $G_0^{(v_0)}$ be the set of bridges of G w.r.t. $V_1 \cup V_2 \cup \ldots \cup V_p$ that include as attachment v_0. Similarly, let $G_0^{(v_p)}$ be the set of bridges of G w.r.t. $V_1 \cup V_2 \cup \ldots \cup V_p$ that include as attachment v_p (note that $G_0 = G_0^{(v_0)} \cup G_0^{(v_p)}$).

Assume that $G \cup \{(x,y)\}$ is planar. Since $\widehat{G}$ can be obtained from G by means of edge contractions and parallel edge contractions, and such contractions preserve planarity, $\widehat{G} \cup \{(x,y)\}$ must be planar too.

Conversely, assume that $\widehat{G} \cup \{(x,y)\}$ is planar. Note that G is obtained from $\widehat{G}$ by replacing $G_j(v_0, v_p)$ with $G(\pi)$. We now distinguish two cases.

Assume first that $G_0^{(v_0)} \cap G_0^{(v_p)} = \emptyset$, and that x and y are either both in $G_0^{(v_0)}$ or both in $G_0^{(v_p)}$. Without loss of generality, assume they are both in $G_0^{(v_0)}$. Then v_0 and v_p are articulation points both in $G \cup \{(x,y)\}$ and in $\widehat{G} \cup \{(x,y)\}$. By Lemma 2.9 the planarity of $\widehat{G} \cup \{(x,y)\}$ implies the planarity of $G_0^{(v_0)} \cup \{(x,y)\}$. Applying again Lemma 2.9 this time to G, yields that G_i, $1 \leq i \leq p$, and $G_0^{(v_p)}$ are planar. Since v_i, $1 \leq i \leq p$, are all articulation points in $G \cup \{(x,y)\}$, by Lemma 2.9 $G \cup \{(x,y)\}$ is planar.

Assume now that either $G_0^{(v_0)} \cap G_0^{(v_p)} \neq \emptyset$ or x and y are one in $G_0^{(v_0)}$ and the other in $G_0^{(v_p)}$. This implies that the graph $\widetilde{G} = G_0^{(v_0)} \cup G_0^{(v_p)} \cup \{(x,y)\}$ is connected. Let $\widetilde{G} = G_0 \cup \{(x,y)\}$. Note that $\widehat{G} \cup \{(x,y)\} = \widetilde{G} \cup G_j(v_0, v_p)$, and that $G \cup \{(x,y)\} = \widetilde{G} \cup G(\pi)$, with $\widetilde{G} \cap G_j(v_0, v_p) = \{v_0, v_p\}$ and $\widetilde{G} \cap G(\pi) = \{v_0, v_p\}$. We claim that the planarity of $\widehat{G} \cup \{(x,y)\}$ implies the planarity of both $\widetilde{G} \cup \{(v_0, v_p)\}$ and $G(\pi) \cup \{(v_0, v_p)\}$.

Indeed if $G_j(v_0, v_p)$ consists of the single edge (v_0, v_p), then $G_j = \{(v_{j-1}, v_j)\}$ and the corresponding round node ρ_j is trivially a free node. Rule (B2) implies that π must be a free chain and consequently $G(\pi) \cup \{(v_0, v_p)\}$ is planar. Furthermore, $\widehat{G} \cup \{(x,y)\} = \widetilde{G} \cup G_j(v_0, v_p) = \widetilde{G} \cup \{(v_0, v_p)\}$ is by hypothesis planar. If $\widetilde{G}$ consists of the single edge (v_0, v_p), then $(v_0, v_p) = (x,y)$ and $\widehat{G} = G_j(v_0, v_p)$. $\widetilde{G} \cup \{(v_0, v_p)\}$ is trivially planar in this case, and the planarity of $\widehat{G} \cup \{(x,y)\}$ implies that π is a free chain and therefore $G(\pi) \cup \{(v_0, v_p)\}$ is planar.

If neither $\widetilde{G}$ nor $G_j(v_0, v_p)$ is a singleton edge, then $\{v_0, v_p\}$ is a separation pair in both $G \cup \{(x,y)\}$ and $\widehat{G} \cup \{(x,y)\}$. Because of Corollary 2.1, the planarity of $\widehat{G} \cup \{(x,y)\}$ implies the planarity of $\widetilde{G} \cup \{(v_0, v_p)\}$ and the planarity of $G_j(v_0, v_p) \cup \{(v_0, v_p)\}$. Due to the definition of rule (B2), if $G_j(v_0, v_p) \cup \{(v_0, v_p)\}$ is planar, then π must be a free chain, and therefore $G(\pi) \cup \{(v_0, v_p)\}$ is planar.

In all the above three cases, both $\widetilde{G} \cup \{(v_0, v_p)\}$ and $G(\pi) \cup \{(v_0, v_p)\}$ are planar. If $\widetilde{G}$ consists of the single edge (v_0, v_p), then $G \cup \{(x, y)\} = G(\pi) \cup \{(v_0, v_p)\}$ is planar. Similarly, if $G(\pi)$ consists of the single edge (v_0, v_p), then $G \cup \{(x, y)\} = \widetilde{G} \cup \{(v_0, v_p)\}$ is planar. Otherwise, $\{v_0, v_p\}$ is a separation pair of $G \cup \{(x, y)\} = \widetilde{G} \cup G(\pi)$. Because of Corollary 2.1, the planarity of $\widetilde{G} \cup \{(v_0, v_p)\}$ and $G(\pi) \cup \{(v_0, v_p)\}$ implies the planarity of $G \cup \{(x, y)\} = \widetilde{G} \cup G(\pi)$. □

Let $\mathcal{T}(\mathcal{C})$ be the tree obtained from $T_2(\mathcal{C})$ after applying any sequence of the rules (B1) and (B2) until no rule can be further applied. Notice that $\mathcal{T}(\mathcal{C})$ has only square (black or red) leaves. Furthermore, all the round nodes are black and cannot be leaves. We recall here that cluster $\mathcal{C}$ has $O(z)$ vertices and edges, and $B(\mathcal{C})$ boundary vertices. The following lemma characterizes some properties of $\mathcal{T}(\mathcal{C})$.

Lemma 4.4 *$\mathcal{T}(\mathcal{C})$ can be computed in $O(|\mathcal{C}|)$ time, has $O(B(\mathcal{C}))$ round nodes and $O(B(\mathcal{C}))$ internal square nodes.*

Proof: As said before, the block tree $T_2(\mathcal{C})$ can be computed in $O(|\mathcal{C}|)$ time and has size $O(|\mathcal{C}|)$. Rule (B1) can be repeatedly applied in a bottom up fashion, starting from the leaves of $T_2(\mathcal{C})$ in a total of $O(|T_2(\mathcal{C})|) = O(|\mathcal{C}|)$ time. All the black chains in the resulting block tree can be located in a total of $O(|\mathcal{C}|)$ time. Note each round node ρ_i is considered at most once while applying rule (B2). Let G_i be the pertinent graph of ρ_i. Checking whether ρ_i is free or blocking can be done in $O(|G_i|)$ time by running an off-line planarity testing on $G_i \cup \{(v_{i-1}, v_i)\}$ [28]. Since also the compression given by rule (B2) can be computed in time proportional to the size of $G(\pi)$, it follows that the total time spent in computing $\mathcal{T}(\mathcal{C})$ will be $O(|\mathcal{C}|)$.

To bound the total number of round nodes and internal square nodes of $\mathcal{T}(\mathcal{C})$ we use the following argument. By definition, the number of red nodes in $\mathcal{C}$ is $B(\mathcal{C})$. Because of rule (B1), the children of any round node in $\mathcal{T}(\mathcal{C})$ cannot all be black leaves. Call *terminal* the round nodes of $\mathcal{T}(\mathcal{C})$ that have at least one child that is a square red leaf. Note that the number of terminal round nodes in $\mathcal{T}(\mathcal{C})$ is at most $O(B(\mathcal{C}))$. Because of rule (B2), each black chain in the original block tree is compressed into a chain of at most a constant numbed of round and internal square nodes. Consequently, the number of square internal and round nodes in $\mathcal{T}(\mathcal{C})$ is linear in the number of terminal round nodes in $\mathcal{T}(\mathcal{C})$, and therefore it is $O(B(\mathcal{C}))$. □

4.2 Step 2: Compressing a Biconnected Component

In this section we consider the following problem. Let $G = (V, E)$ be a planar graph, and let $G' = (V', E')$ be a biconnected subgraph of G contained in a cluster $\mathcal{C}$. Let $\overline{G'} = (\overline{V'}, \overline{E'})$ be the complement of G' in G. As usual define the *boundary of* G' as the set of all vertices $v \in V$ such that there is at least one edge of E' and at least one edge of $\overline{E'} = E - E'$ incident to v. We denote the boundary of G' as $V_{G',G}$. We are interested in finding a planar graph $G'' = (V'', E'')$ that satisfies properties similar to the valid compression of a cluster. That is

(i) G'' contains the boundary of G' ($V_{G',G} \subseteq V''$) and no other external vertex of G'; and

(ii) G'' has at most $O(|V_{G',G}|)$ vertices and edges; and

(iii) the substitution of G'' in place of G' is planarity-preserving.

In the rest of this section we show how to compute such a G'' starting from G'. Denote by n' the number of vertices of G'. Since G' is planar, the number of edges of G' is $O(n')$. We start by computing $T_3(G')$, the tree of triconnected components of G'. As said in Section 2 this can be done in $O(n')$ time. We now distinguish two cases, depending on whether G' is triconnected or not.

4.2.1 Compressing a Triconnected Graph

We now describe our planarity-preserving compression of a non-trivial triconnected graph (in short *triconnected compression*). Given a non-trivial triconnected simple planar graph G' with $O(r)$ selected vertices, denoted by red vertices, we wish to reduce G' to a graph G'', of size $O(r)$, that is non-trivial triconnected and such that this reduction is planarity-preserving. An edge whose endpoints are both red is also colored red. G'' will contain all the red vertices and red edges and will satisfy two important additional properties:

(1) red vertices that are in the same face in G' will be in the same face in G''; and

(2) red vertices that are not in the same face in G' will not be in the same face in G''.

We first define a graph $\widetilde{G}$ as follows. The vertices of $\widetilde{G}$ are the red vertices of G'. Consider a face f of G' and its vertices. If the contour of f contains at least two red vertices, there is an edge in $\widetilde{G}$ between two red vertices u and v if in the contour of f in G' there is a path between u and v that contains only non-red vertices (and possibly consisting of only one edge). With this transformation, we get a new (possibly leaner) face in $\widetilde{G}$ starting from face f of G'. Apply this transformation to all the faces of G'. Note that there can be multiple edges in $\widetilde{G}$ derived from different paths of G' between the same two red vertices.

Since G' is triconnected, it has essentially only one embedding. Therefore it is meaningful to talk about the faces of G'. Furthermore, we represent $\widetilde{G}$ embedded, considering the embedding of $\widetilde{G}$ induced by the embedding of G'. Consequently, also in this case we talk about faces of $\widetilde{G}$. Call *good faces* the faces of $\widetilde{G}$ that correspond to the original faces of G'. Call *bad faces* the remaining faces of $\widetilde{G}$.

We call this step the *face step*. At the end of the face step, we have that any two red vertices that are adjacent in G' are adjacent in $\widetilde{G}$ too. Furthermore, two or more red vertices are in the contour of the same face f of G' if and only if they are all in the contour of the same good face in $\widetilde{G}$. This implies that $\widetilde{G}$ satisfies constraint (1) above because of the good faces. However, $\widetilde{G}$ might still not satisfy constraint (2) because of the bad faces.

Note that $\widetilde{G}$ may be split into disconnected subgraphs, with total size $O(r)$. In what follows, we deal with each connected component of $\widetilde{G}$. We will then show how to reconnect the components into one triconnected graph.

Lemma 4.5 *Let $\widetilde{G}$ be the embedded planar graph defined above starting from a non-trivial simple planar triconnected graph G'. Then $\widetilde{G}$ has no bad faces if and only if all the vertices of G' are red.*

Proof: If all the vertices of G' are red, then by construction $\widetilde{G} = G'$, and all the faces of $\widetilde{G}$ are good. Conversely, assume by contradiction that all the faces of $\widetilde{G}$ are good, but not all the vertices of G' are red. Thus, there must be at least one edge $e = (u, v)$ in $\widetilde{G}$ corresponding to a path $p_{u,v}$ of

G' containing at least one non-red vertex w. Let f'_1 and f'_2 be the faces on the two sides of (u, v) in $\widetilde{G}$. Since all the faces of $\widetilde{G}$ are good, f'_1 and f'_2 correspond to two faces f_1 and f_2 of G' on the two sides of the path $p_{u,v}$. Note that u, v and w are in the contour of faces f_1 and f_2. There are three facts that we exploit in the remainder of the proof, all of them are consequences of the hypothesis that G' is triconnected.

First, u and v must be adjacent on both f_1 and f_2: indeed if this was not the case then by Lemma 2.6 $\{u, v\}$ would be a separation pair in G', a contradiction. Since by hypothesis u and v are not adjacent in the path $p_{u,v}$, they must be adjacent on the other side (namely, the contour of face f_1 contains the path $p_{u,v}$ and the edge (u, v)). The second fact that we exploit is that $p_{u,v}$ must contain only two edges: (u, w) and (w, v). Indeed if $p_{u,v}$ contains at least three edges, denote by u_1 and v_1 the two vertices of $p_{u,v}$ closer to u and v respectively ($u_1 \neq v_1$). Note that u_1 and v (or v_1 and u) are not adjacent in either f_1 or f_2, and therefore by Lemma 2.6 $\{u_1, v\}$ or $\{u, v_1\}$ are both separation pairs of G', a contradiction. Third, by Lemma 2.4 neither u nor v nor w can appear more than once in the contour of either face f_1 or face f_2.

The above three facts imply that vertices u, v, and w must be all adjacent in both faces f_1 and f_2, and furthermore they must appear only once in the contour of the two faces. Because of the first property, there are edges $e_1 = (u, v)$, $e_2 = (v, w)$, and $e_3 = (w, u)$ in G'. Because of the second property, the contour of both f_1 and f_2 consists exactly of edges e_1, e_2, and e_3. But this implies that G' is a triangle, contradicting the hypothesis that G' is a non-trivial triconnected graph. □

Corollary 4.1 *Let $\widetilde{G}$ be the embedded planar graph defined above starting from a non-trivial simple planar triconnected graph G'. Consider the embedding of each connected component of $\widetilde{G}$ separately. If there is at least one non-red vertex in G', then the embedding of each connected component of $\widetilde{G}$ has at least one bad face.*

Proof: If $\widetilde{G}$ is connected, then the corollary follows from Lemma 4.5. If $\widetilde{G}$ is not connected, then consider the embedding of all the connected components of $\widetilde{G}$ induced by the embedding of G'. Note that by construction the contour of a good face of $\widetilde{G}$ must be a simple cycle: since $\widetilde{G}$ is not connected, the contour of the external face of $\widetilde{G}$ is not even connected and thus is not a simple cycle. Consequently, if we take the embedding of all the connected components together, the external face of such embedding is bad. Since the embedding of each connected component is considered separately (from the other components), the embedding of each connected component has its own external face. Each such face does not correspond to any original face in G', and therefore it is a bad face. □

Lemma 4.6 *Let $\widetilde{G}$ be the embedded planar graph defined above starting from a non-trivial simple planar triconnected graph G'. Then the following is true:*

(i) Any two good faces of $\widetilde{G}$ share at most two vertices.

(ii) If good faces f_1 and f_2 share two vertices v_1 and v_2, then v_1 and v_2 are adjacent in both faces f_1 and f_2.

Proof: Condition (i) can be proved by contradiction. Indeed, if there were two good faces f_1 and f_2 of $\widetilde{G}$ sharing three vertices, then by construction there would be two faces of G' sharing the same three vertices, which is impossible by Lemma 2.12.

Similarly to prove (ii), assume by contradiction that there are two good faces f_1 and f_2 of $\widetilde{G}$ sharing two vertices v_1 and v_2, and such that v_1 and v_2 are not adjacent in at least one of f_1 and f_2. Then, by construction of $\widetilde{G}$, there exist two faces in G' sharing v_1 and v_2 and such that v_1 and v_2 are not adjacent in at least one of the two faces. By Lemma 2.12, this contradicts the triconnectivity of G'. $\square$

Lemma 4.7 *Any connected component of $\widetilde{G}$ is either a singleton vertex or a 2-edge-connected graph.*

Proof: Let $\widetilde{G}_i$ be a connected component of $\widetilde{G}$ which has at least two vertices. Let $e = (u, v)$ be an edge of $\widetilde{G}_i$. By construction, e corresponds to a path in G' between red vertices u and v, and such that this path is contained in the contour of a face f of G'. Since G' is triconnected, it has no articulation points. By Lemma 2.4 the contour of f gives a simple cycle in G'. After the face step, e will be in a cycle in $\widetilde{G}_i$. This shows that each edge of $\widetilde{G}_i$ is in a cycle, and therefore $\widetilde{G}_i$ is 2-edge-connected. $\square$

If all the faces of $\widetilde{G}$ are good, we define $G'' = \widetilde{G}$ and stop. Note that by Lemma 4.5 all the vertices of G' are red, and therefore $G'' = G'$.

Assume now that $\widetilde{G}$ has at least one bad face. Let $\widetilde{G}_i$ be a connected component of $\widetilde{G}$. By Lemma 4.7, either $\widetilde{G}_i$ consists of a single vertex or it is 2-edge-connected. If $\widetilde{G}_i$ is 2-edge-connected, by Lemma 2.5 the contour of each face f of $\widetilde{G}_i$ gives a cycle (not necessarily simple) in $\widetilde{G}_i$. For each bad face f of $\widetilde{G}_i$ do the following. Let $v_0, v_1, \ldots, v_{\ell-1}$, $\ell \geq 2$, be the vertices in the contour of f. For $0 \leq i \leq \ell - 1$, do the following. Insert a new vertex n_i as middle-point in edge $(v_i, v_{(i+1) \bmod \ell})$. Insert new vertices s_i and t_i in the interior of face f, joining them to the contour of face f with edges $(n_{(i-1) \bmod \ell}, t_i)$, (t_i, v_i), (v_i, s_i), and (s_i, n_i). Finally, insert edges (t_i, s_i) and $(s_i, t_{(i+1) \bmod \ell})$. This construction is called *face separator* (see Figure 3). The edges (t_i, s_i) and $(s_i, t_{(i+1) \bmod \ell})$, $0 \leq i \leq \ell - 1$, are the contour of a new face called the *crown of the face separator*. The other new faces created by edges $\{(n_{(i-1) \bmod \ell}, t_i), (t_i, v_i), (v_i, n_{(i-1) \bmod \ell})\}$ and $\{(v_i, s_i), (s_i, n_i), (n_i, v_i)\}$ are called the *triangles of the face separator*. If $\widetilde{G}_i$ consists of a single vertex v, the face separator is a wheel with hub v and four new vertices u_1, u_2, u_3, and u_4. The triangles of the face separator are the four faces containing v in their contour, and the face having u_1, u_2, u_3, and u_4 in its contour is the crown of the face separator.

We now summarize some properties that follow immediately from the definition of face separators. First, the crown of a face separator contains at least four edges. Second, any two vertices in the same face separator (both a triangle or a crown of the face separator) are triconnected. Third, crowns of a face separator can share vertices only with triangles. Finally, whenever a triangle shares two vertices with any other face, these two vertices must be adjacent in both faces.

[Figure 3]

Lemma 4.8 *Let $\widetilde{H}$ be any connected component obtained after all the face separators have been inserted. Then*

(i) If any two red vertices are adjacent in G', they are adjacent in $\widetilde{H}$; and

(ii) $\widetilde{H}$ is triconnected; and

(iii) Any two or more red vertices are in the contour of the same face of $\widetilde{H}$ if and only if they are in the contour of the same face of G' (in the same order).

Proof: Let u and v be two red vertices of $\widetilde{H}$, and assume that they are adjacent in G'. Let $\widetilde{G}$ be the graph obtained from G' after the face step. Let f_1 and f_2 be the two faces of G' that are on the two sides of edge (u, v). By construction, u and v will be in two good faces f_1'' and f_2'' of $\widetilde{G}$ corresponding to f_1 and f_2 respectively, and adjacent in these faces. Since f_1'' and f_2'' are good faces, no face separators are inserted in f_1'' and f_2''. Consequently, u and v will still be adjacent in $\widetilde{H}$. This proves (i).

We now prove that $\widetilde{H}$ must be triconnected. Recall that G' is triconnected. We proceed by contradiction and assume that $\widetilde{H}$ is not triconnected. Then there is either an articulation point v or a separation pair $\{v_1, v_2\}$ in $\widetilde{H}$. We prove that consequently there must be respectively an articulation point or a separation pair in G' too, getting the desired contradiction. Since G' is triconnected, it has a unique planar embedding. Throughout the proof, we consider the planar embedding of $\widetilde{H}$ induced by the embedding of G'.

First, assume there is an articulation point v in $\widetilde{H}$. By Lemma 2.4, there must be a face f that is encountered at least twice while going counterclockwise around v in the embedding of $\widetilde{H}$. Note that the faces of $\widetilde{H}$ are only of three types: triangles of a face separator, crowns of a face separator, and good faces. By construction, triangles and crowns of face separators do not have articulation points. Therefore f must be a good face. Again as a consequence of the construction of $\widetilde{H}$, if f is a good face, and v is an articulation point in it, v must be a red vertex. This implies that v is in a face f of G', and f is encountered at least twice while going around v in G'. By Lemma 2.4, v is an articulation point in G' too, a contradiction.

The previous argument shows that $\widetilde{H}$ is at least biconnected. We now show by contradiction that it must be triconnected. Assume that there is a separation pair, say $\{v_1, v_2\}$, in $\widetilde{H}$. By Lemma 2.6 there must exist two faces of $\widetilde{H}$, say f_1 and f_2, such that v_1 and v_2 are in f_1 and f_2, and such that v_1 and v_2 are not adjacent in at least one of f_1 and f_2. Again, we observe that each face of $\widetilde{H}$ can be only of three different types: either a triangle of a face separator, or a crown of a face separator, or a good face. We have only two possibilities: either one of the faces is a triangle, or both faces are good faces. If either one of the faces is a triangle, v_1 and v_2 would be adjacent in both f_1 and f_2. Similarly, if f_1 and f_2 are both good faces of $\widetilde{H}$, by Lemma 4.6 v_1 and v_2 would also be adjacent in both f_1 and f_2. This yields the desired contradiction, proving (ii).

We now prove (iii). Note that since both G' and $\widetilde{H}$ are triconnected, they have only one embedding. The fact that red vertices are in the contour of the same face of G' if and only if they are in the contour of the same face of $\widetilde{H}$ (with the same order) is now a consequence of the construction of $\widetilde{H}$. When building $\widetilde{G}$, consider the embedding of $\widetilde{G}$ induced by the embedding of G'. If some red vertices are in the contour of the same face f of G', at the beginning of the construction of $\widetilde{H}$, they will be in the same good face of $\widetilde{G}$. On the other hand, if two red vertices v_1 and v_2 are not in the contour of the same face of G', then they will not be in the contour of the same good face in $\widetilde{G}$. If they are not even in the contour of a bad face of $\widetilde{G}$, they will not be in the contour of the same face of $\widetilde{H}$ as well. If x and y are in the contour of the same bad face of $\widetilde{G}$ a face separator will separate them in $\widetilde{H}$. Consequently, x and y will not be in the contour of the same face of $\widetilde{H}$ as well. $\Box$

The only problem left is that this construction might give rise to disconnected graphs. So assume this is the case. Denote by H the graph consisting of all the connected graphs $\widetilde{H}$ defined before.

Note that by Corollary 4.1, each $\widetilde{H}$ has at least one face separator. For each $\widetilde{H}$ arbitrarily pick one face separator, and choose the planar embedding of $\widetilde{H}$ having as external face the crown of such face separator. To merge two disconnected graphs $\widetilde{H}_1$ and $\widetilde{H}_2$ of H we simply take two pairs of three adjacent vertices t_i, s_i and t_{i+1} in the chosen face separators (external faces) of $\widetilde{H}_1$ and $\widetilde{H}_2$. Now merge $\widetilde{H}_1$ and $\widetilde{H}_2$ by identifying the two pairs of the three adjacent vertices and by removing multiple edges. Since the two pairs of adjacent vertices are in the external faces of $\widetilde{H}_1$ and $\widetilde{H}_2$, this merging produces a planar graph. As noted before, the contour of the crown of a face separator (i.e., the contour of the external faces of $\widetilde{H}_1$ and $\widetilde{H}_2$) contains at least four edges. Since the new external face produced by the merging of the previous two faces has at least four edges, this merge can be always repeated as needed. Furthermore, since the s_i's and t_i's from the face separators are not any of the original vertices from G', the adjacency properties between red vertices are not changed. We now show that if $\widetilde{H}_1$ and $\widetilde{H}_2$ are triconnected, then this merging always produces a triconnected graph.

Lemma 4.9 *Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two non-trivial triconnected graphs with $V_1 \cap V_2 = \emptyset$. Let $e_1 = (a_1, b_1)$ and $e_2 = (b_1, c_1)$ be two edges in G_1 incident upon the same vertex b_1, and let $e_3 = (a_2, b_2)$ and $e_4 = (b_2, c_2)$ be two edges in G_2 incident upon the same vertex b_2. Let G be the graph obtained from G_1 and G_2 after identifying $a_1 = a_2$, $b_1 = b_2$, $c_1 = c_2$, and after removing e_3 and e_4. Then G is a non-trivial triconnected graph.*

Proof: Note that $G_1 \subseteq G$, and $G_2 \subseteq G$. Let $a = a_1 = a_2$, $b = b_1 = b_2$, and $c = c_1 = c_2$ be the three vertices of G involved in the merging. Assume by contradiction that G is not triconnected and let $\{u, v\}$ be a separation pair of G. Clearly, the separation pair $\{u, v\}$ cannot be entirely either in G_1 or in G_2 since by hypothesis these two graphs are triconnected. This implies that $u \neq a, b, c$ and $v \neq a, b, c$. Consequently, one vertex of $\{u, v\}$, say u, must be in $G_1 - \{a_1, b_1, c_1\}$ and the other, v, in $G_2 - \{a_2, b_2, c_2\}$. But this is also impossible, since in this case u and v would be articulation points in the triconnected graphs G_1 and G_2 respectively. This proofs that G is triconnected. The fact that G is non-trivial follows immediately from $|E| = |E_1| + |E_2| - 2 \geq 6$. $\Box$

The subgraphs of H are merged pair-wise until a connected graph is obtained. This graph is G'', the triconnected compression of G'. As a consequence of this merging, we obtain:

Lemma 4.10 *Let G'' be the graph defined above. If G' is a non-trivial triconnected planar graph then G'' is non-trivial triconnected planar. Furthermore, the following is true:*

(i) If any two red vertices are adjacent in G', they are adjacent in G''; and

(ii) Any two or more red vertices are in the contour of the same face of G'' if and only if they are in contour of the same face of G' (in the same order).

Lemma 4.11 *Let $G' = (V', E')$ be a triconnected graph with n' vertices and r red vertices and red edges. Then the graph $G'' = (V'', E'')$ defined above contains the red vertices and red edges of G', has at most $O(r)$ vertices and edges, and the substitution of G'' in place of G' is planarity-preserving. Furthermore G'' can be computed in $O(n')$ time.*

Proof: The lemma is obvious if there are no bad faces after the face step, since by Lemma 4.5 all the vertices and edges of G' are red, and by construction $G'' = G'$.

Assume now that there is at least one bad face after the face step. G'' is initialized with the red vertices of G'. The red edges are inserted during the face step. Note that red vertices are never deleted from G''. Furthermore, if e is a red edge, it is between two faces f_i and f_j that will not undergo the construction of a face separator. Therefore, also red edges are never destroyed, and G'' contains the red vertices and red edges of G'.

To bound the size of G'', we observe that we build G'' with a constant number of steps, and at each step the extra vertices and edges introduced is always linear in the size of the previous graph. The time bound is a consequence of the fact that each step can be implemented in linear time.

We wish to prove that the substitution of G'' in place of G' is planarity-preserving. Denote by $\widehat{G}$ the graph obtained by replacing G'' in place of G' in G. Let x and y be two vertices of G external to G', let $V_{G',G}$ be the boundary of G' in G, and let $V_{G'',\widehat{G}}$ be the boundary of G'' in $\widehat{G}$. Since by construction G'' has the same boundary as G', we have that the red vertices of G' (and G'') are the vertices in $V_{G',G} = V_{G'',\widehat{G}}$.

Assume $G \cup \{(x, y)\}$ is planar. We now prove that also $\widehat{G} \cup \{(x, y)\}$ must be planar. Consider the separation classes $E_1, E_2, \ldots, E_q$, $q \geq 1$, of $G \cup \{(x, y)\}$ w.r.t. V' (the vertex set of G') as defined in Section 2.1. Let $B_1 = (V_1, E_1), B_2 = (V_2, E_2), \ldots, B_q = (V_q, E_q)$ be the bridges defined by these separation classes. For $1 \leq i \leq q$, let $u_{i,1}, u_{i,2} \ldots u_{i,p_i}$ be the attachments of bridge B_i. Note that by definition $u_{i,j} \in V_{G',G} = V_{G'',\widehat{G}}$, $1 \leq i \leq q$, $1 \leq j \leq p_i$. In other words, $G = G' \cup B_1 \cup B_2 \cup \ldots \cup B_q$ and $\widehat{G} = G'' \cup B_1 \cup B_2 \cup \ldots \cup B_q$.

By Lemma 2.13 and Lemma 2.14, since G' is a non-trivial triconnected subgraph of the planar graph $G \cup \{(x, y)\}$, we have that

(i) for $1 \leq i \leq q$, the attachments $u_{i,1}, u_{i,2} \ldots u_{i,p_i}$ of bridge B_i are on the contour of a face of G'; and

(ii) any two bridges B_i and B_j whose attachments are on the contour of a face of G' are not interlacing.

Note that by Lemma 4.10, any two or more boundary vertices of G' are in the contour of the same face of G'' if and only if they are in the contour of the same face of G'. Furthermore, because of Lemma 4.10 (part (ii)), G'' preserves the same order of red vertices around a face. Consequently, properties (i) and (ii) above hold also for G'' and therefore $\widehat{G} \cup \{(x, y)\}$ is planar too. The converse can be proved with the same argument. □

4.2.2 Compressing a Non-triconnected Graph

We now consider the case where G' is a biconnected subgraph of G, but G' is not triconnected. We show how to compute a graph G'' that satisfies the three properties given in Section 4.2. Let $V_{G',G}$ be the boundary of G' in G. Recall that given a vertex v in G', its allocation node $\gamma(v)$ is the highest node in $T_3(G')$ that has v in its pertinent graph. Color *red* each node of $T_3(G')$ that is an allocation node of a boundary vertex of G', and color black all the other nodes of $T_3(G')$. As a result, we have at most $|V_{G',G}|$ red nodes in $T_3(G')$. To compute G'' we use the same approach used in Section 4.1 with $T_2(G)$: namely we compress $T_3(G')$ yielding a compression of G'. However, this time the details will be more involved.

As in Section 4.1, the new G'' will be obtained from G' by applying a sequence of elementary transformations. Each such transformation will not destroy boundary vertices of G' and will be planarity-preserving. Let $K^{(0)} = G'$, and define $K^{(\tau+1)}$ as the graph obtained after applying the $(\tau+1)$-th transformation on $K^{(\tau)}$, $\tau \geq 0$. Similarly, we denote the trees of triconnected components as $T_3(K^{(\tau)})$, $\tau \geq 0$. Let $h \geq 0$ be such that $K^{(h)} = G''$. The sequence of h transformations applied to G' changes the whole graph G as well, producing the graphs $G = G^{(0)}, G^{(1)}, \ldots, G^{(h)}$, where $G^{(\tau)}$ is obtained from $G^{(\tau-1)}$ by substituting $K^{(\tau)}$ in place of $K^{(\tau-1)}$, $1 \leq \tau \leq h$. We now define the rules used to change G' into G''. As in Section 4.1 there will be one rule that deletes black leaves, and one that compresses black chains. The analog of rule (B1) is the following:

(T1) Delete a black node of degree 1 in the tree of triconnected components.

Lemma 4.12 *Rule (T1) does not delete boundary vertices and gives rise to planarity-preserving substitutions in G'.*

Proof: For some $\tau \geq 0$, let $T_3(K^{(\tau)})$ be the tree before applying rule (T1). Recall that $K^{(\tau)}$ can be reassembled by performing all the possible merges in $T_3(K^{(\tau)})$. Let γ_i be a black node of degree 1 in $T_3(K^{(\tau)})$. This means that γ_i is not allocation node of any boundary vertex, and has only one tree edge incident to it. Let (a, b, σ) be the label of this tree edge. Then $\{a, b\}$ is a separation pair in the graph $K^{(\tau)}$. Denote by H' and H'' the two split graphs of $K^{(\tau)}$ w.r.t. $\{a, b\}$, with H' being the pertinent graph of γ_i.

Let H_1 and H_2 be the two graphs obtained after deleting the virtual edge (a, b) from H' and H'' respectively. Then $K^{(\tau)} = H_1 \cup H_2$. Notice that since by hypothesis γ_i is black, the only possible boundary vertices of H_1 are a and b. Consequently the whole graph $G^{(\tau)}$ can be decomposed into G_0 and H_1, $G^{(\tau)} = G_0 \cup H_1$, such that $H_1 \cap G_0 = \{a, b\}$. Rule (T1) deletes γ_i from $T_3(K^{(\tau)})$, producing a new tree of triconnected components $T_3(K^{(\tau+1)}) = T_3(K^{(\tau)}) - \{\gamma_i\}$. Then by definition of tree of triconnected components, rule (T1) replaces the triconnected component H' with edge (a, b). That is, the graph obtained from $G^{(\tau)}$ after applying rule (T1) is $G^{(\tau+1)} = G_0 \cup \{(a, b)\}$. Clearly, this does not destroy any boundary vertex. Let x and y be any two vertices external to $K^{(\tau)}$. Then x and y must be in G_0. Assume $G^{(\tau)} \cup \{(x, y)\}$ is planar; since $G^{(\tau+1)}$ can be obtained from $G^{(\tau)}$ by applying contractions, $G^{(\tau+1)} \cup \{(x, y)\}$ is planar too. To prove the converse, assume $G^{(\tau+1)} \cup \{(x, y)\}$ is planar. But since $G^{(\tau)} \cup \{(x, y)\}$ can be obtained from $G^{(\tau+1)} \cup \{(x, y)\}$ by replacing edge (a, b) with H_1 (nontrivial bridge expansion), then $G^{(\tau)} \cup \{(x, y)\}$ must be planar too. $\square$

Denote by $T_3(G'_{(T1)})$ the tree obtained from $T_3(G')$, when rule (T1) cannot be applied any more: $T_3(G'_{(T1)}) = T_3(K^{(h_1)})$, for some h_1, $0 \leq h_1 \leq h$. As usual let $G'_{(T1)} = K^{(h_1)}$ denote the graph obtained by reassembling $T_3(G'_{(T1)})$. Note that because of rule (T1), there are no more than $|V_{G',G}|$ nodes of degree 1 in $T_3(G'_{(T_1)})$, because each one of them must be red. Since all the red nodes of $T_3(G')$ are still in $T_3(G'_{(T1)})$, $G'_{(T1)}$ contains the boundary of G'. By Lemma 4.12, the substitution of $G'_{(T1)}$ in place of G' is planarity-preserving. But even though there are at most $O(|V_{G',G}|)$ nodes of degree 1 in $T_3(G'_{(T1)})$, $G'_{(T1)}$ might still have more than $O(|V_{G',G}|)$ vertices and edges. This might happen for two different reasons. First, $T_3(G'_{(T1)})$ itself can be larger than $O(|V_{G',G}|)$ because of possible degree-two black nodes in $T_3(G'_{(T_1)})$. Second, even if there are no degree-two black nodes in $T_3(G'_{(T_1)})$ (and consequently the number of nodes in $T_3(G'_{(T_1)})$ is $O(|V_{G',G}|)$), the overall size of the pertinent graphs of nodes left in $T_3(G'_{(T_1)})$ (and therefore the size of the reassembled graph) can be more than $O(|V_{G',G}|)$. We tackle the two problems separately. First, we consider the problem of compressing the pertinent graph of red nodes and of black nodes of degree at least three in $T_3(G'_{(T_1)})$, giving a compression rule denoted by (T2). Then we will give another compression rule, denoted by (T3), which will allow us to compress the black chains (i.e., paths of degree-two black nodes).

Let γ_i be a node of $T_3(K^{(\tau)})$ which is either a red node or a black node of degree at least three, and let $\Gamma_i = (V_i, E_i)$ be its pertinent graph. Let $v_1, v_2, \ldots, v_p$, $p \geq 1$, be the boundary vertices of $K^{(\tau)}$ whose allocation node is γ_i. We color these vertices *red* in Γ_i. Let $e_1, e_2, \ldots, e_q$, $q \geq 3$, be the edges of $T_3(K^{(\tau)})$ incident to γ_i, and let (s_j, t_j, σ_j) be the label of edge e_j, $1 \leq j \leq q$. We also denote by e_j the virtual edge (s_j, t_j). For $1 \leq j \leq q$, we color in Γ_i the virtual edge e_j and its endpoints s_j and t_j *red*. All the other vertices and edges of Γ_i are colored black. We refer to $V'_i = \{v_1, \ldots, v_p, s_1, t_1, \ldots, s_q, t_q\}$ as the *red set* of Γ_i. We compress Γ_i into a new graph G''_i, which has $O(|V'_i|)$ vertices and edges and such that the substitution corresponding to the replacement of Γ_i with G''_i is planarity-preserving. We perform three different types of compression depending on whether γ_i is a P-, a B-, or a T-node.

(T2.a) If γ_i is a B-node, then Γ_i is a bond consisting of (at least three) parallel edges between two vertices a and b. Denote these edges by $e_1, \ldots, e_q, e_{q+1}, \ldots, e_p$, $p \geq 3$, $p \geq q$, with e_j, $1 \leq j \leq q$, being red edges. Rule (T2) in this case deletes the black edges $e_s, e_{s+1}, \ldots, e_p$, where $s = \max\{q+2, 4\}$. In other words, if $q > 1$, (T2) deletes all the black edges $e_{q+2} \ldots, e_p$. If $q = 1$, (T2) deletes the black edges $e_{q+3} \ldots, e_p$. In any case, the compression of Γ_i contains at least the three multiple edges e_1, e_2, and e_3.

(T2.b) If γ_i is a P-node, then Γ_i is a polygon consisting of a simple cycle on vertices $V_i = \{v_0, v_1, \ldots, v_{p-1}\}$, $p \geq 3$, and with edges $E_i = \cup_{j=0}^{p-1}\{(v_j, v_{(j+1) \bmod p})\}$. If the edge (v_j, v_{j+1}) is red, then we color red also the two vertices v_j and v_{j+1}. Let $V'_i \subseteq V_i$ and $E'_i \subseteq E_i$ be respectively the set of red vertices and red edges in Γ_i. Define *black paths* of P those paths whose endpoints are red but do not contain any red vertex or edge inside. (T2) contracts a black path with more than three edges into a path with exactly three edges, and it is repeatedly applied until there are no more black paths with more than three edges in Γ_i. The compression of Γ_i is always a cycle, whose number of edges is at least three, and is linear in the number of red vertices and edges of Γ_i.

(T2.c) If γ_i is a T-node, then Γ_i is a triconnected graph: we apply to Γ_i the compression algorithm given in Section 4.2.1, with red vertices defined above and red edges defined above plus the edges whose endpoints are red (as in Section 4.2.1).

Let γ_i a node of $T_3(G')$ that we compress according to rule (T2), and let $\Gamma_i = (V_i, E_i)$ be its corresponding triconnected component. Denote by n_{γ_i} the number of boundary vertices of $K^{(\tau)}$ that have γ_i as allocation node (these are red vertices in Γ_i), and by m_{γ_i} the number of edges of $T_3(G')$ incident to γ_i (these are red edges in Γ_i). Recall that other red vertices and red edges may be added: indeed in case (T2.b) the endpoints of a red edge are colored red, while in case (T2.c) edges between two red vertices are colored red. However, in both cases the number of added red vertices and red edges is linear in n_{γ_i} and m_{γ_i}, and the size of the red set of γ_i is always $O(n_{\gamma_i} + m_{\gamma_i})$.

Lemma 4.13 *Rule (T2) given above does not delete boundary vertices and gives rise to planarity-preserving substitutions in G'. Furthermore, after compressing a node γ_i according to (T2), the size of the triconnected component associated with γ_i reduces to $O(n_{\gamma_i} + m_{\gamma_i})$.*

Proof: It is easy to see that transformation (T2) does not delete any boundary vertex. We now show that it is planarity-preserving. Let $T_3(K^{(\tau)})$ be the tree before applying this compression, and let $K^{(\tau)}$ be the graph reassembled after performing all the possible merges in $T_3(K^{(\tau)})$.

If γ_i is a B-node as in case (T2.a), $e_1, \ldots, e_q, e_{q+1}, \ldots, e_p$ correspond to the separation classes of the separation pair $\{a, b\}$. Furthermore, since the edges $e_{q+1}, \ldots, e_p$ do not correspond to tree edges in $T_3(K^{(\tau)})$, they correspond to trivial separation classes of $\{a, b\}$ (that is, classes consisting of singleton edges). Then $K^{(\tau)}$ can be decomposed as follows: $K^{(\tau)} = H_1 \cup \{e_{q+1}\} \cup \ldots \cup \{e_p\}$. Consequently, the whole graph $G^{(\tau)}$ can be decomposed as $G^{(\tau)} = G_0 \cup \{e_{q+1}\} \cup \ldots \cup \{e_p\}$. After the transformation we have that either $K^{(\tau+1)} = H_1 \cup \{e_{q+1}\}$ or $K^{(\tau+1)} = H_1 \cup \{e_{q+1}\} \cup \{e_{q+2}\}$ and therefore either $G^{(\tau+1)} = G_0 \cup \{e_{q+1}\}$ or $G^{(\tau+1)} = G_0 \cup \{e_{q+1}\} \cup \{e_{q+2}\}$. The change induced in $G^{(\tau)}$ consists only of deleting some multiple edges in $e_{q+2}, \ldots, e_p$. Therefore, since we perform only parallel edge contractions, given any two vertices x and y, $G^{(\tau+1)} \cup \{(x, y)\}$ is planar if and only if $G^{(\tau)} \cup \{(x, y)\}$ is planar. Since the graph resulting from transformation (T2.a) contains at most two black edges (e_{q+1} and e_{q+2}), the new pertinent graph of γ_i has $O(n_{\gamma_i} + m_{\gamma_i})$ vertices and edges.

Now consider the case in which γ_i is a P-node. We have that $G^{(\tau+1)}$ is obtained from $G^{(\tau)}$ by just contracting each black path of Γ_i with more than three edges into a path with exactly three edges. Note that, because of rule (T1), black edges of γ_i do not correspond to non-trivial separation classes of $G^{(\tau)}$ but correspond to edges of $G^{(\tau)}$. Consequently, a path of black edges of Γ_i corresponds to a path of degree-two vertices (except the endpoints) in $G^{(\tau)}$. Recall that we can always replace a path of degree-two vertices with an edge without changing the planar embedding (nontrivial bridge contraction); and vice versa we can always replace an edge with a path of degree-two vertices without changing the embedding (trivial bridge expansion). Note that we can go from $G^{(\tau)} \cup \{(x, y)\}$ to $G^{(\tau+1)} \cup \{(x, y)\}$ (and vice versa) by means of a sequence of nontrivial bridge contractions followed by a sequence of trivial bridge expansions. This shows that $G^{(\tau+1)} \cup \{(x, y)\}$ is planar if and only if $G^{(\tau)} \cup \{(x, y)\}$ is planar. After all the black paths have been contracted, the bounds on the size of the new pertinent graph of γ_i are straightforward.

Finally, if γ_i is a T-node, the lemma follows from Lemma 4.11. □

4.2.3 Compressing a black chain

Denote by $T_3(G'_{(T2)})$ the tree obtained from $T_3(G'_{(T1)})$, when rule (T2) cannot be applied any more: $T_3(G'_{(T2)}) = T_3(K^{(h_2)})$, for some h_2, $0 \le h_1 \le h_2 \le h$. As usual let $G'_{(T2)} = K^{(h_2)}$ denote the graph obtained by reassembling $T_3(G'_{(T2)})$. A *black chain* in $T_3(G'_{(T2)})$ is a directed path consisting of black nodes of degree two. We say that the black chain is *trivial* if it contains only one node, and *non-trivial* otherwise.

In this section we consider the problem of reducing the size of $T_3(G'_{(T_2)})$ down to $O(|V_{G',G}|)$. We recall that the only obstacles left are the non-trivial black chains of $T_3(G'_{(T_2)})$. Since we will not consider trivial black chains any further, in the remainder of this section we use the term *black chains* to denote *non-trivial black chains*.

Let γ_1 and γ_ℓ, be the two endpoints of a maximal black chain $\pi = \{\gamma_2, \gamma_3, \ldots, \gamma_{\ell-1}\}$, $\ell \geq 4$, in $T_3(K^{(\tau)})$, $h_2 \leq \tau \leq h$, such that γ_ℓ is a proper ancestor of γ_1. That is, all the nodes γ_j, $2 \leq j \leq \ell-1$, are degree-two black nodes, while γ_1 is not a degree-two black node, and either γ_ℓ is the root of $T_3(K^{(\tau)})$ or γ_ℓ is not a degree-two black node. Note that the removal of the two edges (γ_1, γ_2) and $(\gamma_{\ell-1}, \gamma_\ell)$ partitions the tree $T_3(K^{(\tau)})$ into three subtrees. Let these subtrees be T', T'', and T''', such that T' contains γ_1, $T'' = \pi = \{\gamma_2, \ldots, \gamma_{\ell-1}\}$ is the black chain, and T''' contains γ_ℓ. From now on, we refer to the black chain as π. Let $G_3(T')$, $G_3(\pi)$ and $G_3(T''')$ be the graphs obtained by performing all merges to T', π and T''' respectively. We refer to $G_3(\pi)$ as the *graph corresponding to the black chain* π.

For $1 \leq j \leq \ell - 1$, let e_j be the tree edge between γ_j and γ_{j+1}, and let e_j be labeled (s_j, t_j, σ_j). For $2 \leq j \leq \ell-1$, let Γ_j be the pertinent graph of γ_j (corresponding to its triconnected component). We also denote by e_j the virtual edge (s_j, t_j) in Γ_j and Γ_{j+1}. This creates no danger of confusion with the tree edge $e_j = (s_j, t_j, \sigma_j)$: the tree edge and the two virtual edges are associated with the same split σ_j. The two edges $e_1 = (s_1, t_1)$ and $e_{\ell-1} = (s_{\ell-1}, t_{\ell-1})$ are referred to as the *terminals* of the black chain π. Note that e_1 and $e_{\ell-1}$ are the only virtual edges of $G_3(\pi)$: all the other edges are actual edges. Moreover, $e_{j-1} = (s_{j-1}, t_{j-1})$ and $e_j = (s_j, t_j)$ are the only virtual edges in Γ_j, $2 \leq j \leq \ell - 1$.

Our goal is to compress the black chain π into a graph of constant size containing vertices s_1, t_1, $s_{\ell-1}$, and $t_{\ell-1}$, such that this compression is planarity-preserving. In other words, the compressed graph must have the same planar behavior with respect to vertices s_1, t_1, $s_{\ell-1}$ and $t_{\ell-1}$ as the black chain π. We will give a compression technique that follows the same ideas given for the black chains of the block trees in Section 4.1. Namely, we will partition black chains into blocking chains and free chains. As in Section 4.1, a chain π will be blocking if there is no planar embedding of $G_3(\pi)$, having the two terminals in the contour of the same face. However, since now the terminals of a chain are edges and not simply vertices as in Section 4.1, there is a much richer combinatorial structure in the chain π, and the details will be more involved. Before giving our compression rule, we need some technical lemmas and more terminology.

Lemma 4.14 *Let $T_3(K)$ be the tree of triconnected components of a biconnected graph K. Let $\pi = \{\gamma_2, \ldots, \gamma_{\ell-1}\}$ be a chain of degree-two nodes in $T_3(K)$, and let T' and T''' be the two subtrees of $T_3(K)$ at the endpoints of the black chain. Then there is a path p_1 between s_1 and t_1 in $G_3(T')$ and a path $p_{\ell-1}$ between $s_{\ell-1}$ and $t_{\ell-1}$ in $G_3(T''')$, such that p_1 and $p_{\ell-1}$ consist both of actual edges and are disjoint.*

Proof: Since $G_3(T')$ is a split graph of K w.r.t. $\{s_1, t_1\}$, (s_1, t_1) is the only virtual edge of $G_3(T')$, and by Lemma 2.2 $G_3(T')$ is biconnected. This implies that there is a simple cycle containing edge (s_1, t_1), or equivalently, there is a path p_1 in $G_3(T')$ between s_1 and t_1 that contains actual edges only. Similarly, there is a path $p_{\ell-1}$ in $G_3(T''')$ between vertices $s_{\ell-1}$ and $t_{\ell-1}$ that contains actual edges only. Since $G_3(T')$ is a split graph of K with respect to separation pair $\{s_1, t_1\}$, and $G_3(T''')$ is a split graph of K with respect to separation pair $\{s_{\ell-1}, t_{\ell-1}\}$, the paths p_1 and $p_{\ell-1}$ are disjoint. □

Lemma 4.15 *Let $\pi = \{\gamma_2, \ldots, \gamma_{\ell-1}\}$ be a chain of degree-two nodes. If there is a node γ_i, for some i, $2 \leq i \leq \ell - 1$, such that in its pertinent graph Γ_i we have that $s_{i-1} = s_i$ and $t_{i-1} = t_i$, then γ_i is a B-node.*

Proof: Note that if $s_{i-1} = s_i$ and $t_{i-1} = t_i$, the virtual edges e_{i-1} and e_i are multiple edges in Γ_i. Consequently, γ_i can be neither a P-node nor a T-node, because the pertinent graphs of such nodes do not have multiple edges. Then γ_i must be a B-node. □

Lemma 4.16 *Let $\pi = \{\gamma_2, \ldots, \gamma_{\ell-1}\}$ be a chain of degree-two nodes. If γ_r, for some $2 \leq r \leq \ell - 1$, is either a B-node or a T-node, then in Γ_r there is a path between vertices s_r and t_r, and a path between vertices s_{r-1} and t_{r-1} that contain no virtual edges.*

Proof: If γ_r is a B-node, then $s_{r-1} = s_r$ and $t_{r-1} = t_r$. Consequently, γ_r contains at least one actual edge (s_r, t_r) and the lemma is trivially true. If γ_r is a T-node, then its pertinent graph Γ_r is a simple triconnected graph having as virtual edges (s_{r-1}, t_{r-1}) and (s_r, t_r). Consequently, there are at least three vertex disjoint paths between s_r and t_r, and three vertex disjoint paths between vertices s_{r-1} and t_{r-1}. Due to the fact that there are only two virtual edges in Γ_r, in Γ_r there is a path between s_r and t_r and a path between s_{r-1} and t_{r-1} that contain no virtual edges. □

Define $H_1 = (V_1, E_1)$ by deleting the virtual edge $e_1 = (s_1, t_1)$ from $G_3(T')$, $H_2 = (V_2, E_2)$ by deleting virtual edges $e_1 = (s_1, t_1)$ and $e_{\ell-1} = (s_{\ell-1}, t_{\ell-1})$ from $G_3(\pi)$, and $H_3 = (V_3, E_3)$ by deleting virtual edge $e_{\ell-1} = (s_{\ell-1}, t_{\ell-1})$ from $G_3(T''')$. Note that $K^{(\tau)} = H_1 \cup H_2 \cup H_3$, where $H_1 \cap H_2 = \{s_1, t_1\}$, $H_2 \cap H_3 = \{s_{\ell-1}, t_{\ell-1}\}$. Since there are no red nodes in π, the only possible boundary vertices of H_2 are s_1, t_1, $s_{\ell-1}$, and $t_{\ell-1}$. Let G be a graph obtained by adding edges to $G^{(\tau)}$: $G = G^{(\tau)} \cup N$, where the graph N has the same vertex set as $G^{(\tau)}$. If the endpoints of every edge of N are both external to cluster $\mathcal{C}$, we say that G is an *expansion of $G^{(\tau)}$ w.r.t. $\mathcal{C}$*. Our main use of expansions will be by only one edge: (x, y). Note that $H_2 \subseteq G^{(\tau)} \subseteq G$, and that again the only possible boundary vertices of H_2 in G are s_1, t_1, $s_{\ell-1}$, and $t_{\ell-1}$. As a result, any bridge of G w.r.t. V_2 (the vertex set of H_2) has attachments only in $\{s_1, t_1, s_{\ell-1}, t_{\ell-1}\}$. Let G_1 be the set of bridges of G w.r.t. V_2 whose attachments include either s_1 or t_1, and let $G_{\ell-1}$ be the set of bridges of G w.r.t. V_2 whose attachments include either $s_{\ell-1}$ or $t_{\ell-1}$. Then the whole graph G can be decomposed into three parts: $G = G_1 \cup H_2 \cup G_{\ell-1}$, where $H_1 \subseteq G_1$, $H_3 \subseteq G_{\ell-1}$, and G_1 and $G_{\ell-1}$ may or may not have empty intersection. We further partition G_1 and $G_{\ell-1}$ as follows. Let $G_{\ell-1}^{(s_{\ell-1})}$ contain the bridges of $G_{\ell-1}$ that include $s_{\ell-1}$ but not $t_{\ell-1}$ as attachment; let $G_{\ell-1}^{(t_{\ell-1})}$ contain the bridges of $G_{\ell-1}$ that include $t_{\ell-1}$ but not $s_{\ell-1}$ as attachment; finally let $G_{\ell-1}^{(e_{\ell-1})}$ contain the bridges of $G_{\ell-1}$ that include both $s_{\ell-1}$ and $t_{\ell-1}$ as attachments. Note that $G_{\ell-1} = G_{\ell-1}^{(e_{\ell-1})} \cup G_{\ell-1}^{(s_{\ell-1})} \cup G_{\ell-1}^{(t_{\ell-1})}$. We define a similar partition of G_1 into three sets $G_1^{(e_1)}$, $G_1^{(s_1)}$, and $G_1^{(t_1)}$. Using this notation, the whole graph G is decomposed into seven parts: $G = G_1^{(e_1)} \cup G_1^{(s_1)} \cup G_1^{(t_1)} \cup H_2 \cup G_{\ell-1}^{(e_{\ell-1})} \cup G_{\ell-1}^{(s_{\ell-1})} \cup G_{\ell-1}^{(t_{\ell-1})}$. However, some of these parts can have non-empty intersections: for instance a bridge of G w.r.t. V_2 that has attachments $s_1, s_{\ell-1}, t_{\ell-1}$ appears both in $G_1^{(s_1)}$ and in $G_{\ell-1}^{(e_{\ell-1})}$. We claim that not all the six graphs $G_1^{(e_1)}$, $G_1^{(s_1)}$, $G_1^{(t_1)}$, $G_{\ell-1}^{(e_{\ell-1})}$, $G_{\ell-1}^{(s_{\ell-1})}$ and $G_{\ell-1}^{(t_{\ell-1})}$ are necessary for our purposes. Indeed the following two lemmas prove that only $G_1^{(e_1)}$, $G_{\ell-1}^{(e_{\ell-1})}$, $G_{\ell-1}^{(s_{\ell-1})}$, and $G_{\ell-1}^{(t_{\ell-1})}$ are sufficient to describe completely G.

Lemma 4.17 *Let $K^{(\tau)} \subseteq \mathcal{C}$ be the biconnected subgraph of $G^{(\tau)}$ that contains a black chain $\pi = \{\gamma_2, \gamma_3, \ldots, \gamma_{\ell-1}\}$. The only possible boundary vertices of $K^{(\tau)}$ in H_2 are $s_{\ell-1}$ and $t_{\ell-1}$. Furthermore, for any expansion G of $G^{(\tau)}$ w.r.t. $\mathcal{C}$, we have that: (i) $G_1^{(s_1)} = \emptyset$ unless $s_1 = s_{\ell-1}$; (ii) $G_1^{(t_1)} = \emptyset$ unless $t_1 = t_{\ell-1}$; (iii) $G_1^{(e_1)} \neq \emptyset$; and (iv) $G_{\ell-1}^{(e_{\ell-1})} \neq \emptyset$.*

Proof: Let $T_3(K^{(\tau)})$ be the tree of triconnected components of $K^{(\tau)}$. Recall that a node of $T_3(K^{(\tau)})$ is red if and only if it is the allocation node of a boundary vertex of $K^{(\tau)}$, and it is black otherwise. By definition of black chain, all the nodes γ_i, $2 \leq i \leq \ell-1$, are black. Consequently, in H_2 only vertices s_1, t_1, $s_{\ell-1}$ and $t_{\ell-1}$ can possibly be boundary vertices of $K^{(\tau)}$. We now show that if s_1 [respectively t_1] is a boundary vertex of $K^{(\tau)}$, then $s_1 = s_{\ell-1}$ [respectively $t_1 = t_{\ell-1}$], which proves that the only possible boundary vertices of $K^{(\tau)}$ in H_2 are $s_{\ell-1}$ and $t_{\ell-1}$. Indeed, assume that s_1 is a boundary vertex of $K^{(\tau)}$. Then, the allocation node $\gamma(s_1)$ of s_1 must be a red vertex in $T_3(K^{(\tau)})$. Note that s_1 appears in Γ_2, the pertinent graph of γ_2. By definition of allocation node, we have that $\gamma(s_1)$ must be an ancestor of γ_2 in $T_3(K^{(\tau)})$. However, γ_i, $2 \leq i \leq \ell-1$, is not a red node in $T_3(K^{(\tau)})$. Consequently, $\gamma(s_1)$ must be a proper ancestor of $\gamma_{\ell-1}$. By Lemma 2.3, $s_1 = s_{\ell-1}$. A similar argument shows that if t_1 is a boundary vertex of $K^{(\tau)}$ then $t_1 = t_{\ell-1}$.

We now prove that $G_1^{(s_1)} = \emptyset$ unless $s_1 = s_{\ell-1}$. Assume that $G_1^{(s_1)} \neq \emptyset$. Then there is a bridge of G w.r.t. V_2 (the vertex set of H_2), whose attachments include s_1 but not t_1. Since $\{s_1, t_1\}$ is a separation pair of $K^{(\tau)}$, all the paths between a vertex in G_1 and a vertex in $G_{\ell-1}$ that are entirely contained in $K^{(\tau)}$ must go through H_2. This implies that any path between a vertex in $G_1^{(s_1)}$ and a vertex in $G_{\ell-1}$ that is entirely contained in $K^{(\tau)}$ must go through s_1. Since $K^{(\tau)}$ is a biconnected subgraph, this implies that if $G_1^{(s_1)} \neq \emptyset$ then $G_1^{(s_1)} \cap K^{(\tau)} = \{s_1\}$ (otherwise s_1 would be an articulation point inside $K^{(\tau)}$, a contradiction). Consequently, s_1 must be a boundary vertex of $K^{(\tau)}$. The first part of the lemma implies $s_1 = s_{\ell-1}$. The same argument shows that $G_1^{(t_1)} = \emptyset$ unless $t_1 = t_{\ell-1}$.

To complete the proof of the lemma, we need to show that $G_1^{(e_1)} \neq \emptyset$ and that $G_{\ell-1}^{(e_{\ell-1})} \neq \emptyset$. Consider T' and T''', the two subtrees of $T_3(K^{(\tau)})$ that are at the endpoints of the black chain. As usual, let $G_3(T')$ and $G_3(T''')$ be the graphs obtaining by applying all the merges to T' and T'''. Recall that the only virtual edge contained in $G_3(T')$ [respectively $G_3(T''')$] is (s_1, t_1) [respectively $(s_{\ell-1}, t_{\ell-1})$], and that $H_1 = G_3(T') - \{(s_1, t_1)\}$, and $H_3 = G_3(T''') - \{(s_{\ell-1}, t_{\ell-1})\}$. Note that T' contains at least γ_1, and T''' contains at least γ_ℓ. This implies that $G_3(T')$ and $G_3(T''')$ contain actual edges, and therefore $H_1 \neq \emptyset$, and $H_3 \neq \emptyset$. Since $\{s_1, t_1\}$ is a separation pair of $K^{(\tau)}$ with separation classes H_1 and $H_2 \cup H_3$, we have that $H_1 \subseteq G_1^{(e_1)}$. Similarly, $\{s_{\ell-1}, t_{\ell-1}\}$ is a separation pair of $K^{(\tau)}$ with separation class H_3 and $H_1 \cup H_2$, which implies $H_3 \subseteq G_{\ell-1}^{(e_{\ell-1})}$. This yields $G_1^{(e_1)} \neq \emptyset$ and $G_{\ell-1}^{(e_{\ell-1})} \neq \emptyset$. □

Lemma 4.18 *Let G be an expansion of $G^{(\tau)}$ w.r.t. cluster $\mathcal{C}$.*

(i) If $s_1 \neq s_{\ell-1}$ and $t_1 \neq t_{\ell-1}$, then $G_1^{(s_1)} = G_1^{(t_1)} = \emptyset$.

(ii) If $s_1 = s_{\ell-1}$ and $t_1 \neq t_{\ell-1}$, then $G_1^{(s_1)} = G_{\ell-1}^{(s_{\ell-1})} \cup G_{\ell-1}^{(e_{\ell-1})}$ and $G_1^{(t_1)} = \emptyset$.

(iii) If $s_1 \neq s_{\ell-1}$ and $t_1 = t_{\ell-1}$, then $G_1^{(s_1)} = \emptyset$ and $G_1^{(t_1)} = G_{\ell-1}^{(t_{\ell-1})} \cup G_{\ell-1}^{(e_{\ell-1})}$.

(iv) If $s_1 = s_{\ell-1}$ and $t_1 = t_{\ell-1}$, then $G_1^{(s_1)} = G_{\ell-1}^{(s_{\ell-1})}$ and $G_1^{(t_1)} = G_{\ell-1}^{(t_{\ell-1})}$.

Proof: Case (i) follows immediately from Lemma 4.17. Consider now case (ii). By Lemma 4.17 we have that $G_1^{(t_1)} = \emptyset$. Recall that $G^{(s_1)}$ is the set of bridges of $G^{(\tau)}$ w.r.t. V_2 whose attachments include s_1 but not t_1, and that the only attachments of bridges of $G^{(\tau)}$ w.r.t. to V_2 can be $s_1 = s_{\ell-1}$, t_1 and $t_{\ell-1}$ (the boundary vertices of H_2). Consequently, $G_1^{(s_1)}$ is the set of bridges whose attachments are either $\{s_{\ell-1}\}$ or $\{s_{\ell-1}, t_{\ell-1}\}$, which implies $G_1^{(s_1)} = G_{\ell-1}^{(s_{\ell-1})} \cup G_{\ell-1}^{(e_{\ell-1})}$. A similar argument (interchanging s's and t's) can be repeated for case (iii).

Assume now that $s_1 = s_{\ell-1}$ and $t_1 = t_{\ell-1}$. This implies $G_1 = G_{\ell-1}$, and therefore $G_1^{(s_1)} = G_{\ell-1}^{(s_{\ell-1})}$ and $G_1^{(t_1)} = G_{\ell-1}^{(t_{\ell-1})}$. □

Let $\pi = \{\gamma_2, \gamma_3, \ldots, \gamma_{\ell-1}\}$ be a black chain, and let $G_3(\pi)$ be the corresponding graph. As usual, let Γ_i be the pertinent graph of γ_i, $2 \leq i \leq \ell - 1$. We say that π is a *blocking chain*, if and only if there is no embedding of $G_3(\pi)$ having $e_1 = (s_1, t_1)$ and $e_{\ell-1} = (s_{\ell-1}, t_{\ell-1})$ in the contour of the same face. We say that π is a *free chain* otherwise. We will partition blocking chains into four different types of chains, but first we need more terminology.

Let π be a blocking chain. Recall that a triconnected graph (such as the pertinent graph of a T-node) has only one embedding. A *blocking T-node of type 1* is a T-node γ_i, $2 \leq i \leq \ell - 1$, in π such that e_{i-1} and e_i are not in the contour of the same face of Γ_i.

A *blocking T-node of type 2* is a T-node γ_i, $2 \leq i \leq \ell - 1$, in π such that e_{i-1} and e_i are not in the contour of the same face of Γ_i, and such that either

(i) $s_i \neq s_{\ell-1}$; or

(ii) $s_i = s_{\ell-1}$, but e_{i-1} is not in the same face contour as $s_{\ell-1}$.

A *blocking T-node of type 3* is a T-node γ_i, $2 \leq i \leq \ell - 1$, in π such that e_{i-1} and e_i are not in the contour of the same face of Γ_i, and such that either

(i) $t_i \neq t_{\ell-1}$; or

(ii) $t_i = t_{\ell-1}$, but e_{i-1} is not in the same face contour as $t_{\ell-1}$.

A T-node γ_i that is both of type 2 and 3 is called a *blocking T-node of type 4*. The following corollary is an immediate consequence of these definitions.

Corollary 4.2 *Let $\widehat{\pi}$ be a blocking chain. T-nodes of type 2 and 3 are also of type 1. T-nodes of type 4 are also of type 1, 2 and 3.*

Blocking T-nodes of type 1, 2, 3, and 4 are simply referred to as *blocking T-nodes*. By Corollary 4.2, each blocking T-node is at least of type 1. Consequently, a node that is not a blocking T-node must have e_{i-1} and e_i in the same face of its embedding. A (blocking) T-node of type i, $1 \leq i \leq 4$, which is not of type j, $j > i$, is called a *pure (blocking) T-node of type i*.

Lemma 4.19 *Let $\pi = \{\gamma_2, \ldots, \gamma_{\ell-1}\}$ be a be a chain of degree-two nodes in $T_3(K^{(\tau)})$. Then there can be at most one pure T-node of type 1 in π. If there is one, say γ_i, $2 \leq i \leq \ell - 2$, then either (i) $i = \ell - 1$ or (ii) $i = \ell - 2$ and $\gamma_{\ell-1}$ is a B-node.*

Proof: Assume there is at least one pure T-node of type 1 in π. Let γ_i be the pure T-node of type 1 with smallest i in π, and let Γ_i be the pertinent graph of γ_i. Since γ_i is of a *pure* T-node of type 1, γ_i is of type 1 but not of type h, $h \geq 2$, and $s_i = s_{\ell-1}$, $t_i = t_{\ell-1}$. Let $\pi_{2,i-1} = \{\gamma_2, \ldots, \gamma_{i-1}\}$ and $\pi_{i+1,\ell-1} = \{\gamma_{i+1}, \ldots, \gamma_{\ell-1}\}$ be the two subchains into which γ_i partitions the chain π. By our assumption, $\pi_{2,i-1}$ has no pure T-node of type 1.

Since $s_i = s_{\ell-1}$ and $t_i = t_{\ell-1}$, by Lemma 2.3 we have that $s_j = s_{\ell-1}$ and $t_j = t_{\ell-1}$, $i \leq j \leq \ell-1$. This implies that $s_{j-1} = s_j$ $(= s_{\ell-1})$ and that $t_{j-1} = t_j$ $(= t_{\ell-1})$, $i+1 \leq j \leq \ell-1$. By Lemma 4.15, γ_j is a B-node, $i+1 \leq j \leq \ell-1$. This shows that there is no T-node in $\pi_{i+1,\ell-1}$. Consequently, γ_i is the only pure T-node of type 1 in π. Since by Fact 2.1 there are no two adjacent B-nodes in a tree of triconnected components, either (i) $i = \ell - 1$ (i.e., $\pi_{i+1,\ell-1} = \emptyset$) or (ii) $i = \ell - 2$ (i.e., $\pi_{i+1,\ell-1} = \{\gamma_{\ell-1}\}$) and $\gamma_{\ell-1}$ is a B-node. $\Box$

Lemma 4.20 *Let $\widehat{\pi} = \{\widehat{\gamma}_2, \ldots, \widehat{\gamma}_{\widehat{\ell}-1}\}$ be a chain of degree-two nodes in $T_3(K^{(\tau)})$. Let the tree edge between $\widehat{\gamma}_i$ and $\widehat{\gamma}_{i+1}$ be labeled $(\widehat{s}_i, \widehat{t}_i, \widehat{\sigma}_i)$, $1 \leq i \leq \widehat{\ell} - 1$. There is a simple cycle λ of $G_3(\widehat{\pi})$ containing the two edges $\widehat{e}_1 = (\widehat{s}_1, \widehat{t}_1)$ and $\widehat{e}_{\widehat{\ell}-1} = (\widehat{s}_{\widehat{\ell}-1}, \widehat{t}_{\widehat{\ell}-1})$, and vertices $\widehat{s}_i$ and $\widehat{t}_i$, $1 \leq i \leq \widehat{\ell} - 1$. Furthermore, $\widehat{s}_1, \widehat{s}_2, \ldots, \widehat{s}_{\widehat{\ell}-1}, \widehat{t}_{\widehat{\ell}-1}, \widehat{t}_{\widehat{\ell}-2}, \ldots, \widehat{t}_1$ appear in this order around λ. If $\widehat{\pi}$ is free, then there is an embedding of $G_3(\widehat{\pi})$ having the simple cycle λ as the contour of a face.*

Proof: For $2 \leq i \leq j \leq \widehat{\ell} - 1$, consider the subchain of $\widehat{\pi}$ containing the nodes between $\widehat{\gamma}_i$ and $\widehat{\gamma}_j$: $\widehat{\pi}_{i,j} = \{\widehat{\gamma}_i, \ldots, \widehat{\gamma}_j\}$. Note that $\widehat{\pi} = \widehat{\pi}_{2,\widehat{\ell}-1}$. For $2 \leq i \leq j \leq \widehat{\ell} - 1$, let $G_3(\widehat{\pi}_{i,j})$ be the graph obtained after applying all the merges to the subchain $\widehat{\pi}_{i,j}$. By definition of tree of triconnected components, the only virtual edges of $G_3(\widehat{\pi}_{i,j})$ are $\widehat{e}_{i-1} = (\widehat{s}_{i-1}, \widehat{t}_{i-1})$ and $\widehat{e}_j = (\widehat{s}_j, \widehat{t}_j)$. In order to prove the lemma, we prove that for $2 \leq i \leq j \leq \widehat{\ell} - 1$, there is a simple cycle $\lambda_{i,j}$ in $G_3(\widehat{\pi}_{i,j})$ containing edges $\widehat{e}_{i-1}$ and $\widehat{e}_j$, and vertices $\widehat{s}_k$ and $\widehat{t}_k$, $i-1 \leq k \leq j$, such that $\widehat{s}_{i-1}, \widehat{s}_i, \ldots, \widehat{s}_j, \widehat{t}_j, \widehat{t}_{j-1}, \ldots, \widehat{t}_{i-1}$ appear in this order around $\lambda_{i,j}$. Moreover, if $\widehat{\pi}_{i,j}$ is free there is an embedding of $G_3(\widehat{\pi}_{i,j})$ having a face with contour $\lambda_{i,j}$. The lemma will follow for $i = 2$ and $j = \widehat{\ell} - 1$.

We proceed by induction on $(j - i)$. The base of the induction is for $j - i = 0$. In this case, $\widehat{\pi}_{i,i}$ consists of the single node $\widehat{\gamma}_i$, and $G_3(\widehat{\pi}_{i,i}) = \widehat{\Gamma}_i$. Recall that by Fact 2.2 the directed version of $\widehat{\Gamma}_i$ is st–planar with source $\widehat{s}_i$ and sink $\widehat{t}_i$. By Fact 2.3, given an edge (a, b) of the st–planar graph $\widehat{\Gamma}_i$, there is a path $p_{\widehat{s}_i,a}$ from $\widehat{s}_i$ to a and a path $p_{b,\widehat{t}_i}$ from b to $\widehat{t}_i$ such that the two paths are vertex-disjoint. Choosing $(a, b) = (\widehat{s}_{i-1}, \widehat{t}_{i-1})$ gives two paths $p_{\widehat{s}_i,\widehat{s}_{i-1}}$ and $p_{\widehat{t}_{i-1},\widehat{t}_i}$. Furthermore, by Fact 2.3 if $\widehat{e}_{i-1}$ and $\widehat{e}_i$ are in the contour of a same face, the two paths $p_{\widehat{s}_i,\widehat{s}_{i-1}}$ and $p_{\widehat{t}_{i-1},\widehat{t}_i}$ can be chosen in the contour of the same face. The undirected versions of $p_{\widehat{s}_i,\widehat{s}_{i-1}}$, $\widehat{e}_{i-1} = (\widehat{s}_{i-1}, \widehat{t}_{i-1})$, $p_{\widehat{t}_{i-1},\widehat{t}_i}$ and $\widehat{e}_i = (\widehat{s}_i, \widehat{t}_i)$ gives the desired cycle $\lambda_{i,i}$.

We now prove the induction step. Assume that the induction hypothesis holds for $j - i \leq r$, $r \geq 0$. We will show that it holds for $j - i = r + 1$. Let $\widehat{\pi}_{i,j}$ be any a subchain such that $j - i = r + 1$. Choose a k, $i \leq k \leq j - 1$, and consider the two subchains $\widehat{\pi}_{i,k}$ and $\widehat{\pi}_{k+1,j}$. By the induction hypothesis, there is a simple cycle $\lambda_{i,k}$ in $G_3(\widehat{\pi}_{i,k})$ containing vertices $\widehat{s}_{i-1}, \widehat{s}_i, \ldots, \widehat{s}_k$, $\widehat{t}_k, \widehat{t}_{k-1}, \ldots, \widehat{t}_{i-1}$ (in this order), and edges $\widehat{e}_{i-1} = (\widehat{s}_{i-1}, \widehat{t}_{i-1})$ and $\widehat{e}_k = (\widehat{s}_k, \widehat{t}_k)$. If $\widehat{\pi}_{i,k}$ is free, $\lambda_{i,k}$ is the contour of a face. Similarly, there is a simple cycle $\lambda_{k+1,j}$ in $G_3(\widehat{\pi}_{k+1,j})$ containing vertices $\widehat{s}_k, \widehat{s}_{k+1}, \ldots, \widehat{s}_j$, $\widehat{t}_j, \widehat{t}_{j-1}, \ldots, \widehat{t}_k$ (in this order), and edges $\widehat{e}_k = (\widehat{s}_k, \widehat{t}_k)$ and $\widehat{e}_j = (\widehat{s}_j, \widehat{t}_j)$. If $\widehat{\pi}_{k+1,j}$ is free, $\lambda_{k+1,j}$ is the contour of a face. Since $\{\widehat{s}_k, \widehat{t}_k\}$ is a separation pair in $G_3(\widehat{\pi})$, $\lambda_{i,k}$ and $\lambda_{k+1,j}$ share only vertices $\widehat{s}_k$, $\widehat{t}_k$ and edge $(\widehat{s}_k, \widehat{t}_k)$. Note that $G_3(\widehat{\pi}_{i,j})$ is obtained from a merge operation on the two graphs $G_3(\widehat{\pi}_{i,k})$ and $G_3(\widehat{\pi}_{k+1,j})$. Then the cycle $\lambda_{i,j}$ obtained by merging the two cycles $\lambda_{i,k}$ and $\lambda_{k+1,j}$ and deleting the two virtual edges $\widehat{e}_k$ proves the induction step. □

Note that by Lemma 4.20, a chain π with terminals $e_1 = (s_1, t_1)$ and $e_{\ell-1} = (s_{\ell-1}, t_{\ell-1})$ is such that $s_i \neq t_j$ for any i and j, $1 \leq i, j \leq \ell - 1$.

Lemma 4.21 *Let π be a chain with terminals e_1 and $e_{\ell-1}$, and let $G_3(\pi)$ be the corresponding graph of π. Then the following is true:*

(i) If there is a node of type 1 in π, then there is no embedding of $G_3(\pi)$ having e_1 and $e_{\ell-1}$ in the contour of the same face.

(ii) If there is a node of type 2 in π, then there is no embedding of $G_3(\pi)$ having e_1 and $s_{\ell-1}$ in the contour of the same face.

(iii) If there is a node of type 3 in π, then there is no embedding of $G_3(\pi)$ having e_1 and $t_{\ell-1}$ in the contour of the same face.

Proof: We first prove (i). Assume there is a node of type 1 in π. Namely, there is a T-node γ_i such that $e_{i-1} = (s_{i-1}, t_{i-1})$ and $e_i = (s_i, t_i)$ are not in the contour of the same face of Γ_i. Let $\pi_{2,i-1} = \{\gamma_2, \ldots, \gamma_{i-1}\}$ and $\pi_{i+1,\ell-1} = \{\gamma_{i+1}, \ldots, \gamma_{\ell-1}\}$ be the two subchains into which γ_i partitions π.

We now find a path $p_{2,i-1}$ between vertices s_{i-1} and t_{i-1} in $G_3(\pi_{2,i-1})$, such that the only virtual edge contained in $p_{2,i-1}$ is e_1. Similarly, we find a path $p_{i+1,\ell-1}$ between vertices s_i and t_i in $G_3(\pi_{i+1,\ell-1})$, such that the only virtual edge contained in $p_{i+1,\ell-1}$ is $e_{\ell-1}$. As boundary cases, if $\pi_{2,i-1} = \emptyset$ (i.e., $i = 2$) then $p_{2,i-1} = e_1$, and if $\pi_{i+1,\ell-1} = \emptyset$ (i.e., $i = \ell - 1$), $p_{2,i-1} = e_{\ell-1}$. We now assume that $\pi_{2,i-1} \neq \emptyset$ and $\pi_{i+1,\ell-1} \neq \emptyset$ and show how to find $p_{2,i-1}$ and $p_{i+1,\ell-1}$. By Lemma 4.20 applied to the black chain $\widehat{\pi} = \pi_{2,i-1}$, there is a simple cycle $\lambda_{2,i-1}$ in $G_3(\pi_{2,i-1})$ containing the virtual edges $e_1 = (s_1, t_1)$ and $e_{i-1} = (s_{i-1}, t_{i-1})$, and there is a simple cycle $\lambda_{i+1,\ell-1}$ in $G_3(\pi_{i+1,\ell-1})$ containing the virtual edges $e_i = (s_i, t_i)$ and $e_{\ell-1} = (s_{\ell-1}, t_{\ell-1})$. Let $p_{2,i-1} = \lambda_{2,i-1} - \{(s_{i-1}, t_{i-1})\}$ be the path in $G_3(\pi_{2,i-1})$ between s_{i-1} and t_{i-1}, and let $p_{i+1,\ell-1} = \lambda_{i+1,\ell-1} - \{(s_i, t_i)\}$ be the path in $G_3(\pi_{i+1,\ell-1})$ between s_i and t_i. Note that e_1 is in $p_{2,i-1}$ and $e_{\ell-1}$ is in $p_{i+1,\ell-1}$.

Let G_i'' be the graph obtained from Γ_i by replacing the virtual edge e_{i-1} with the path $p_{2,i-1}$, and the virtual edge e_i with the path $p_{i+1,\ell-1}$. Note that $G_i'' \subset G_3(\pi)$, and that G_i'' is homeomorphic to the non-trivial triconnected graph Γ_i, and therefore it has a unique embedding. By hypothesis e_{i-1} and e_i are not in the contour of the same face of Γ_i, which implies that e_1 and $e_{\ell-1}$ are not in the contour of the same face of G_i''. Since $G_i'' \subseteq G_3(\pi)$, this implies that there is no embedding of $G_3(\pi)$ having e_1 and $e_{\ell-1}$ in the contour of the same face.

We now turn to (ii). Assume there is a node of type 2 in π, i.e., there is a T-node γ_i such that e_{i-1} and e_i are not in the contour of the same face of Γ_i, and such that one of the following two conditions hold: either (a) $s_i \neq s_{\ell-1}$ or (b) $s_i = s_{\ell-1}$ but $e_{i-1} = (s_{i-1}, t_{i-1})$ and $s_{\ell-1}$ are not in the contour of the same face of Γ_i. Again let $\pi_{2,i-1} = \{\gamma_2, \ldots, \gamma_{i-1}\}$ and $\pi_{i+1,\ell-1} = \{\gamma_{i+1}, \ldots, \gamma_{\ell-1}\}$ be the two subchains as defined above. Let $p_{2,i-1}$ and $p_{i+1,\ell-1}$ be the two paths of $G_3(\pi_{2,i-1})$ and $G_3(\pi_{i+1,\ell-1})$ containing respectively edges e_1 and $e_{\ell-1}$, and let G_i'' be the graph homeomorphic to Γ_i as defined before. We now distinguish the two cases (a) and (b) above. Assume first that $s_i \neq s_{\ell-1}$ (i.e., we are in case (a)). Recall that $e_{\ell-1}$ (and therefore $s_{\ell-1}$) is in $p_{i+1,\ell-1}$ and e_1 is in

$p_{2,i-1}$. Since G_i'' has a unique embedding, and by hypothesis e_{i-1} and e_i are not in the contour of the same face of Γ_i, this implies that e_1 and $s_{\ell-1}$ are not in the contour of the same face of G_i''. If we are in case (b), then $s_i = s_{\ell-1}$ but e_{i-1} and $s_{\ell-1}$ are not in the contour of the same face of Γ_i. Consequently, e_1 and $s_{\ell-1}$ will not be in the contour of the same face of G_i''. In both cases (a) and (b), e_1 and $s_{\ell-1}$ are not in the contour of the same face in the unique embedding of G_i''. Since $G_i'' \subseteq G_3(\pi)$, there is no embedding of $G_3(\pi)$ having e_1 and $s_{\ell-1}$ in the contour of the same face.

The proof of (iii) is analogous to the proof of (ii). □

The conditions in Lemma 4.21 are actually necessary and sufficient, as the following lemma shows.

Lemma 4.22 *Let π be a chain with terminals e_1 and $e_{\ell-1}$, and let $G_3(\pi)$ be the corresponding graph of π. Then the following is true:*

(i) There is no embedding of $G_3(\pi)$ having e_1 and $e_{\ell-1}$ in the contour of the same face if and only if there is a node of type 1 in π.

(ii) There is no embedding of $G_3(\pi)$ having e_1 and $s_{\ell-1}$ in the contour of the same face if and only if there is a node of type 2 in π.

(iii) There is no embedding of $G_3(\pi)$ having e_1 and $t_{\ell-1}$ in the contour of the same face if and only if there is a node of type 3 in π.

(iv) There is no embedding of $G_3(\pi)$ having e_1 and $t_{\ell-1}$, and e_1 and $s_{\ell-1}$ in the contour of the same face if and only if there is a node of type 4 in π.

Proof: We first prove (i). If there is a node of type 1 in π, then by Lemma 4.21 there is no embedding of $G_3(\pi)$ having e_1 and $e_{\ell-1}$ in the same face. Conversely, assume that there is no embedding of $G_3(\pi)$ having e_1 and $e_{\ell-1}$ in the contour of the same face. By Fact 2.4, we have that $e_1 \notin \Pi(\gamma_{\ell-1})$. By the definition of periphery given in Section 2.2, this implies that there must be a T-node γ_i in π, $2 \leq i \leq \ell-1$, such that its two virtual edges e_{i-1} and e_i cannot be put in the same face of Γ_i. Since in the pertinent graph of either a P-node or a B-node any two edges can be put in the same face, γ_i must be a T-node. In summary, there must be a T-node γ_i in π such that e_{i-1} and e_i are not in the same face of the unique embedding of Γ_i: this is exactly a T-node of type 1.

We now turn to (ii). If there is a node of type 2 in π, then by Lemma 4.21 there is no embedding of $G_3(\pi)$ having e_1 and $s_{\ell-1}$ in the same face. Conversely, assume that there is no T-node of type 2 in π. This implies that for any T-node γ_i in π, $2 \leq i \leq \ell-1$, $s_i = s_{\ell-1}$ and e_{i-1} is in the same face contour as $s_{\ell-1}$ in Γ_i. For any B-node or P-node γ_i, choose an embedding in which e_{i-1} is in the same face contour as e_i (as noticed before, this is always possible since each edge of the pertinent graph of either a B-node or a P-node is peripheral). Applying all the merges to π with the chosen embedding of the pertinent graphs by Lemma 2.11 gives an embedding of $G_3(\pi)$ in which e_1 and $s_{\ell-1}$ are in the contour of the same face.

The proof of (iii) is completely analogous to the proof of (ii), while (iv) follows from the fact that a node of type 4 is by definition both of type 2 and of type 3. □

Because of Lemma 4.22, we say that a node of type 1 blocks e_1 from $e_{\ell-1}$, a node of type 2 blocks e_1 from $s_{\ell-1}$, and a node of type 3 blocks e_1 from $t_{\ell-1}$.

Lemma 4.23 *Let π be a black chain. If there is a type 2 node in π which is not of type 3, then there is no type 3 node in π which is not of type 2. Similarly, if there is a type 3 node in π which is not of type 2, then there is no type 2 node in π which is not of type 3.*

Proof: Assume by contradiction that there are two nodes γ_i and γ_j, $2 \leq i, j \leq \ell - 1$, in π such that (i) γ_i is of type 2 but not of type 3 (i.e., γ_i blocks e_1 from $s_{\ell-1}$ but not from $t_{\ell-1}$); and (ii) γ_j is of type 3 but not of type 2 (i.e., γ_j blocks e_1 from $t_{\ell-1}$ but not from $s_{\ell-1}$). The contradiction we show is that in π neither γ_i can be an ancestor of γ_j nor γ_j can be an ancestor of γ_i.

Since γ_i is of type 2, then in Γ_i e_{i-1} is not in the same face contour as e_i. But since γ_i is not of type 3, it must be that $t_i = t_{\ell-1}$ and e_{i-1} is in the same face contour as $t_i = t_{\ell-1}$. By Lemma 2.3, $t_k = t_{\ell-1}$, $i \leq k \leq \ell - 1$. For $i+1 \leq k \leq \ell - 1$, $e_{k-1} = (s_{k-1}, t_{k-1}) = (s_{k-1}, t_{\ell-1})$ is incident to $t_{\ell-1}$. This implies that for $i \leq k \leq \ell - 1$, e_{k-1} is in the same face contour as $t_k = t_{\ell-1}$ in Γ_k (recall that e_{i-1} is in the same face contour as $t_i = t_{\ell-1}$). Thus no ancestor of γ_i can be of type 3, and consequently γ_j cannot be an ancestor of γ_i. A similar proof shows that γ_i cannot be an ancestor of γ_j, and gives the contradiction. $\Box$

We define a blocking chain π to be a *chain of type* i, $1 \leq i \leq 4$, if it contains at least one T-node of type i, but no T-node of type j, $j > i$. Note that because of Lemma 4.23, if π has T-nodes of type 2 and of type 3, it must also have a T-node of type 4.

The following corollary is a consequence of Lemma 4.22.

Corollary 4.3 *Let π be a black chain.*

(i) *π is a free chain if and only if there is an embedding of $G_3(\pi)$ having e_1 and $e_{\ell-1}$ in the contour of the same face.*

(ii) *π is of type 1 if and only if there is no embedding of $G_3(\pi)$ having e_1 and $e_{\ell-1}$ in the contour of the same face, but there is an embedding having e_1 and $s_{\ell-1}$ in the contour of the same face, and an embedding having e_1 and $t_{\ell-1}$ in the contour of the same face (these two embeddings may or may not be the same).*

(iii) *π is of type 2 if and only if there is no embedding of $G_3(\pi)$ having e_1 and $s_{\ell-1}$ in the contour of the same face, but there is an embedding of $G_3(\pi)$ having e_1 and $t_{\ell-1}$ in the contour of the same face.*

(iv) *π is of type 3 if and only if there is no embedding of $G_3(\pi)$ having e_1 and $t_{\ell-1}$ in the contour of the same face, but there is an embedding of $G_3(\pi)$ having e_1 and $s_{\ell-1}$ in the contour of the same face.*

(v) *Finally, π is of type 4 if and only if there is no embedding of $G_3(\pi)$ having e_1 in the same face contour as either $s_{\ell-1}$ or $t_{\ell-1}$.*

We now describe our transformation (T3), that compresses a maximal black chain π, that is a black chain not included in any other black chain. Let $G^{(\tau)}$ be a graph, let $\mathcal{C}$ be a cluster in $G^{(\tau)}$, and let $K^{(\tau)}$ be the biconnected component of $\mathcal{C}$, whose tree $T_3(K^{(\tau)})$ contains the black chain π. We have two different transformations: one for free chains, and another for blocking chains. Rather than describing (T3) on the whole graph $G^{(\tau)}$, we will describe this transformation on $G_3(\pi)$, that is the graph corresponding to π.

Consider first the case of a free chain π: there is an embedding of $G^{(\tau)}$ having $s_1, t_1, s_{\ell-1}$ and $t_{\ell-1}$ in the contour of the same face. Consider $\gamma(s_1)$ and $\gamma(t_1)$, respectively the allocation nodes of s_1 and t_1 in $T_3(K^{(\tau)})$. By definition, $\gamma(s_1)$ and $\gamma(t_1)$ must be proper ancestors of γ_1, since both s_1 and t_1 appear in Γ_2, the pertinent graph of γ_2. Define two nodes γ_p and γ_q, $2 \leq p, q \leq \ell$, as follows. If the allocation node $\gamma(s_1)$ is in the chain π, then there is a node γ_j, $2 \leq j \leq \ell - 1$, such that $\gamma(s_1) = \gamma_j$: define $\gamma_p = \gamma_j$. If $\gamma(s_1)$ is not in the chain π, then define $\gamma_p = \gamma_\ell$. Similarly, define γ_q to be the allocation node of t_1 if this allocation node is in the chain π, and to be γ_ℓ otherwise. Note that by Lemma 2.3, $s_i = s_1$ for $2 \leq i \leq p-1$, and $t_i = t_1$ for $2 \leq i \leq q-1$, and by definition of allocation node $s_j \neq s_1$, $p \leq j \leq \ell - 1$, and $t_j \neq t_1$, $q \leq j \leq \ell - 1$.

Without loss of generality assume $p \leq q$ (otherwise interchange s_j and t_j in what follows). To compress $G_3(\pi)$, we partition the black chain π into at most five subpaths: $\pi_{2,p-1} = \{\gamma_2, \ldots, \gamma_{p-1}\}$, $\pi_{p,p} = \{\gamma_p\}$, $\pi_{p+1,q-1} = \{\gamma_{p+1}, \ldots, \gamma_{q-1}\}$, $\pi_{q,q} = \{\gamma_q\}$, and $\pi_{q+1,\ell-1} = \{\gamma_{q+1}, \ldots, \gamma_{\ell-1}\}$ (we consider a subpath $\pi_{i,j} = \{\gamma_i, \ldots, \gamma_j\}$ empty whenever $j < i$). Let $\pi_{i,j}$ be any such subchain. We denote by $G_3(\pi_{i,j})$ the graph obtained by applying merge operations to the tree nodes in $\pi_{i,j}$. Note that the only virtual edges in $G_3(\pi_{i,j})$ are e_{i-1} and e_j. We define $H_{i,j}$ to be the graph obtained by deleting the virtual edges e_{i-1} and e_j from $G_3(\pi_{i,j})$. Since $G_3(\pi)$ is obtained by merging the graphs $G_3(\pi_{2,p-1})$, $G_3(\pi_{p,p})$, $G_3(\pi_{p+1,q-1})$, $G_3(\pi_{q,q})$, and $G_3(\pi_{q+1,\ell-1})$, we have that $H_2 = H_{2,p-1} \cup H_{p,p} \cup H_{p+1,q-1} \cup H_{q,q} \cup H_{q+1,\ell-1}$. Similarly $G^{(\tau)} = G_1 \cup H_{2,p-1} \cup H_{p,p} \cup H_{p+1,q-1} \cup H_{q,q} \cup H_{q+1,\ell-1} \cup G_{\ell-1}$ (see Figure 4).

[Figure 4]

We now give a compressed representation of each non-empty subchain ($\pi_{2,p-1}$, $\pi_{p,p}$, $\pi_{p+1,q-1}$, $\pi_{q,q}$, or $\pi_{q+1,\ell-1}$). Let $\pi_{i,j} = \{\gamma_i, \ldots, \gamma_j\}$ be such a subchain.

If $i = j$, then the subchain $\pi_{i,j}$ consists of a single node $\gamma_i = \gamma_j$. We call this a *singleton* subchain (note that $\pi_{p,p}$ and $\pi_{q,q}$ are singleton subchains). In this case we compress the pertinent graph Γ_i of γ_i using rule (T2) with red nodes $\{s_{i-1}, t_{i-1}, s_i, t_i\}$. The graph obtained with this compression will be referred to (interchangeably) as either Γ'_i or $G'_{i,i}$. We call this a *node compression.*

If $i < j$, then $\pi_{i,j}$ consists of at least two nodes. Let $(s_{i-1}, t_{i-1}, \sigma_{i-1})$ be the label of the tree edge entering node γ_i and let (s_j, t_j, σ_j) be label of the tree edge leaving node γ_j. This implies that the only virtual edges in $G_3(\pi_{i,j})$ are $e_{i-1} = (s_{i-1}, t_{i-1})$ and $e_j = (s_j, t_j)$. Note that we cannot have both $s_{i-1} = s_j$ and $t_{i-1} = t_j$. Indeed, if this was true by Lemma 2.3 we would have $s_h = s_j$ and $t_h = t_j$, for $i - 1 \leq h \leq j$. Because of Lemma 4.15, all the nodes γ_h, $i - 1 \leq h \leq j$, are B-nodes. But since by Fact 2.1 in a tree of triconnected components no two B-nodes can be adjacent, this implies $i = j$ and that $\pi_{i,j}$ is a singleton chain. Consequently, if $i < j$ either $s_{i-1} \neq s_j$ or $t_{i-1} \neq t_j$ (namely $|\{s_{i-1}, t_{i-1}, s_j, t_j\}| \geq 3$). We represent $\pi_{i,j}$ with a triconnected graph $G'_{i,j} = (V'_{i,j}, E'_{i,j})$ that is a wheel on vertices s_{i-1}, t_{i-1}, s_j, and t_j and having as hub an extra (new) vertex w. Formally, $V'_{i,j} = \{s_{i-1}, t_{i-1}, s_j, t_j, w\}$ and $E'_{i,j}$ contains edges (s_{i-1}, s_j), (s_j, t_j), (t_j, t_{i-1}), (t_{i-1}, s_{i-1}), (s_{i-1}, w), (s_j, w), (t_j, w), and (t_{i-1}, w). We delete self-loops and multiple edges resulting from a possible repetition in the definition of $V'_{i,j}$. All the previous edges are considered actual edges. Furthermore, we add new virtual edges $e_{i-1} = (s_{i-1}, t_{i-1})$ and $e_j = (s_j, t_j)$. Note that there are two edges between s_{i-1} and t_{i-1}, and two edges between s_i and t_i: in both cases one is actual and the other is virtual. We call this a *wheel replacement.*

By definition, $\pi_{2,p-1}$ is such that $s_i = s_1$ and $t_i = t_1$, $2 \leq i \leq p - 1$. By Lemma 4.15, $\pi_{2,p-1}$ is either empty or contains only one B-node. If they are not empty, $\pi_{p,p}$ and $\pi_{q,q}$ gives always rise to a node compression, and $\pi_{p+1,q-1}$ and $\pi_{q+1,\ell-1}$ can yield either a node compression or a wheel replacement. Note that each node compression and each wheel replacement has always a constant number of vertices and edges. In summary, in this case transformation (T3) replaces the graph $G_3(\pi)$ with a graph G'_π obtained from the merges of $G'_{2,p-1}$, Γ'_p, $G'_{p+1,q-1}$, Γ'_q, $G'_{q+1,\ell-1}$, and such that the size of G'_π is constant.

Consider now the case where π is a blocking chain. We first reduce to two the number of multiple pure T-nodes of the same type by deleting all the multiple pure T-nodes of the same type but the first and the last. This leaves a chain with a constant number of blocking T-nodes. Note that the subchains between these blocking T-nodes are free: we compress each such subchain to $O(1)$ size using the rule described for free chains. Finally, we compress each blocking T-node left using a node compression.

For both free and blocking chains, transformation (T3) replaces the corresponding graph $G_3(\pi)$ of the black chain π with a new graph G'_π. Recall that H_2 is the graph obtained from $G_3(\pi)$ after deleting the virtual edges e_1 and $e_{\ell-1}$. Similarly, define H'_2 to be the graph to be obtained from G'_π after deleting the virtual edges e_1 and $e_{\ell-1}$. Then the effect of transformation (T3) in the original graph $G^{(\tau)}$ is to replace H_2 with a graph H'_2. In other words, (T3) gives rise to the graph $G^{(\tau+1)} = G_1 \cup H'_2 \cup G_{\ell-1}$ obtained from $G^{(\tau)} = G_1 \cup H_2 \cup G_{\ell-1}$, where H'_2 has a constant number of vertices and edges.

It is easy to see that rule (T3) does not destroy boundary vertices. We now show that it is planarity-preserving. We prove two different lemmas: one for free chains, and the other for blocking chains.

Lemma 4.24 *Let π be a free chain in a planar graph $G^{(\tau)}$. Then rule (T3) gives rise to planarity-preserving substitutions.*

Proof: If π is a free chain, then there is an embedding of $G_3(\pi)$ having the two virtual edges $e_1 = (s_1, t_1)$ and $e_{\ell-1} = (s_{\ell-1}, t_{\ell-1})$ in the contour of the same face. Or equivalently, the vertices s_1, t_1, $s_{\ell-1}$ and $t_{\ell-1}$ can all be put in a same face of H_2. Before applying rule (T3), $G^{(\tau)} = G_0 \cup H_2$, where $G_0 \cap H_2 = \{s_1, t_1, s_{\ell-1}, t_{\ell-1}\}$. Afterwards, the new graph is $G^{(\tau+1)} = G_0 \cup H'_2$, where again $G_0 \cap H'_2 = \{s_1, t_1, s_{\ell-1}, t_{\ell-1}\}$. We check that this transformation is planarity-preserving for any two vertices in G_0.

Compressing a free chain is implemented by using two basic primitives: node compressions and wheel replacements. Since a node compression is planarity-preserving because of Lemma 4.13, we need to analyze wheel replacements only. Recall that a wheel replacement can only happen for the subpaths $\pi_{p+1,q-1}$ and $\pi_{q+1,\ell-1}$. We only consider the subchain $\pi_{q+1,\ell-1} = \{\gamma_{q+1}, \ldots, \gamma_{\ell-1}\}$, and assume that it gives rise to a wheel replacement, and that this replacement is done before the replacement of $\pi_{p+1,q-1} = \{\gamma_{p+1}, \ldots, \gamma_{q-1}\}$. An analogous argument can be repeated for the subchain $\pi_{p+1,q-1} = \{\gamma_{p+1}, \ldots, \gamma_{q-1}\}$, including the case that we have two wheel replacements. Transformation (T3) substitutes the wheel $G'_{q+1,\ell-1}$ in place of $G_3(\pi_{q+1,\ell-1})$, the graph corresponding to $\pi_{q+1,\ell-1}$. This yields a new graph $\widehat{G}$ from $G^{(\tau)}$. In order for $G'_{q+1,\ell-1}$ to be a wheel replacement, it must be that either $s_q \neq s_{\ell-1}$ or $t_q \neq t_{\ell-1}$ (otherwise by Lemma 4.15 $G'_{q+1,\ell-1}$ is a B-node compression). In what follows, we assume that $s_q \neq s_{\ell-1}$ and $t_q \neq t_{\ell-1}$; if either $s_q = s_{\ell-1}$ or $t_q = t_{\ell-1}$ we can apply the same argument given below.

Recall that $\{s_q, t_q\}$ is a separation pair in $G_3(\pi)$ separating $H_{q+1,\ell-1}$ from $H_{2,p-1} \cup H_{p,p} \cup H_{p+1,q-1} \cup H_{q,q}$. As a consequence of $G_0 \cap H_2 = \{s_1, t_1, s_{\ell-1}, t_{\ell-1}\}$, in $H_{q+1,\ell-1}$ the only attachments of bridges in G_0 can possibly be $s_{\ell-1}$, $t_{\ell-1}$, s_q (if $s_q = s_1$) and t_q (if $t_q = t_1$). In summary, the boundary of $H_{q+1,\ell-1}$ in $G^{(\tau)}$ is given by the vertices $s_{\ell-1}$, $t_{\ell-1}$, s_q and t_q.

Let x and y be any two vertices in G_0, and assume that $\widehat{G} \cup \{(x,y)\}$ is planar. Recall that $H'_{q+1,\ell-1} = (V'_{q+1,\ell-1}, E'_{q+1,\ell-1})$ has vertex set $V'_{q+1,\ell-1} = \{s_1, t_q, s_{\ell-1}, t_{\ell-1}, w\}$ and edge set $E'_{q+1,\ell-1} = \{(t_q,w), (s_q,w), (s_{\ell-1},w), (t_{\ell-1},w), (s_q,s_{\ell-1}), (t_q,t_{\ell-1}), (s_q,t_q), (s_{\ell-1},t_{\ell-1})\}$.

Let $\overline{H_{q+1,\ell-1}}$ be the complement of $H_{q+1,\ell-1}$ in $G^{(\tau)} \cup \{(x,y)\}$, and let $\overline{H'_{q+1,\ell-1}}$ be the complement of $H'_{q+1,\ell-1}$ in $\widehat{G} \cup \{(x,y)\}$. Since $G^{(\tau)} \cup \{(x,y)\}$ and $\widehat{G} \cup \{(x,y)\}$ differs only in $H_{q+1,\ell-1}$ and $H'_{q+1,\ell-1}$, $\overline{H_{q+1,\ell-1}} = \overline{H'_{q+1,\ell-1}}$.

Recall that $H'_{q+1,\ell-1}$ is a triconnected wheel and has s_q, $s_{\ell-1}$, $t_{\ell-1}$ and t_q in the contour of the same face in its unique embedding embedding (and exactly in this order). The planarity of $\widehat{G} \cup \{(x,y)\}$ implies that there must be an embedding of $\overline{H'_{q+1,\ell-1}} = \overline{H_{q+1,\ell-1}}$ having s_q, $s_{\ell-1}$, $t_{\ell-1}$ and t_q in the contour of the same face and exactly in this order.

Consider now $H_{q+1,\ell-1} = (V_{q+1,\ell-1}, E_{q+1,\ell-1})$. Recall that it is obtained by applying all the merges to the subchain $\pi_{q+1,\ell-1} = \{\gamma_{q+1} \ldots, \gamma_{\ell-1}\}$, and by discarding the virtual edges $e_q = (s_q, t_q)$ and $e_{\ell-1} = (s_{\ell-1}, t_{\ell-1})$. By hypothesis, π is a free chain, and therefore there are no T-nodes of type h, $h \geq 1$, in π. This implies that also in the subchain $\pi_{q+1,\ell-1}$ of π there are no T-nodes of type h, $h \geq 1$. By hypothesis $G^{(\tau)}$ is planar, and therefore we have that $G_3(\pi_{q+1,\ell-1}) = \{e_q\} \cup H_{q+1,\ell-1} \cup \{e_{\ell-1}\} \subseteq G^{(\tau)}$ is planar. Because of Corollary 4.3 applied to the chain $\pi_{q+1,\ell-1}$, there is a planar embedding of $\{e_q\} \cup H_{q+1,\ell-1} \cup \{e_{\ell-1}\}$ having e_q and $e_{\ell-1}$ in the contour of the same face. This implies that there is an embedding of $H_{q+1,\ell-1}$ having s_q, $s_{\ell-1}$, $t_{\ell-1}$ and t_q all in the contour of the same face (and exactly in this order by Lemma 4.20). Note that also the unique embedding of the wheel $H'_{q+1,\ell-1}$ has s_q, $s_{\ell-1}$, $t_{\ell-1}$ and t_q all in the contour of the same face (and exactly in this order). Because of this property, we can substitute $H_{q+1,\ell-1}$ in place of $H'_{q+1,\ell-1}$ while maintaining the planarity of the embedding: indeed, as shown before, there must be an embedding of $\overline{H'_{q+1,\ell-1}} = \overline{H_{q+1,\ell-1}}$ having s_q, $s_{\ell-1}$, $t_{\ell-1}$ and t_q in the contour of the same face (and exactly in this order). Consequently, if $\widehat{G} \cup \{(x,y)\}$ is planar, then $G^{(\tau)} \cup \{(x,y)\}$ must be planar too.

We now prove the converse: namely, we show that if $G^{(\tau)} \cup \{(x,y)\}$ is planar, then $\widehat{G} \cup \{(x,y)\}$ must be planar. This part of the proof is more complicated because $H'_{q+1,\ell-1}$ has only one possible embedding, while $H_{q+1,\ell-1}$ can have many embeddings. Our proof is by contradiction: namely, we assume that $G^{(\tau)} \cup \{(x,y)\}$ is planar, but $\widehat{G} \cup \{(x,y)\}$ is not, and derive a contradiction. Let π be the black chain. By Lemma 4.20, there is a simple cycle λ in $G_3(\pi)$ containing vertices $s_1, \ldots, s_p, \ldots, s_{q-1}, s_q, \ldots s_{\ell-1}, t_{\ell-1}, \ldots, t_q, t_{q-1}, \ldots, t_p, \ldots, t_1$ (encountered in this order while going around λ), and virtual edges $e_1 = (s_1, t_1)$ and $e_{\ell-1} = (s_{\ell-1}, t_{\ell-1})$.

Let T' and T''' be the subtrees of $T_3(K^{(\tau)})$ at the two endpoints of the black chain. By Lemma 4.14, there is a path p_1 in $G_3(T')$ between s_1 and t_1 that contains actual edges only, and there is a path $p_{\ell-1}$ in $G_3(T''')$ between vertices $s_{\ell-1}$ and $t_{\ell-1}$ containing actual edges only, such that p_1 and $p_{\ell-1}$ are disjoint. Since $\{s_1, t_1\}$ and $\{s_{\ell-1}, t_{\ell-1}\}$ are separation pairs of $K^{(\tau)}$, replacing virtual edge (s_1, t_1) with path p_1, and virtual edge $(s_{\ell-1}, t_{\ell-1})$ with path $p_{\ell-1}$, yields a simple cycle $\lambda^{(\tau)}$ of $K^{(\tau)}$ containing again vertices $s_1, \ldots, s_p, \ldots, s_{q-1}, s_q, \ldots s_{\ell-1}, t_{\ell-1}, \ldots, t_q, t_{q-1}, \ldots, t_p, \ldots, t_1$ (encountered in this order while going around $\lambda^{(\tau)}$). Since $\lambda^{(\tau)}$ is in $K^{(\tau)}$, it is contained in $G^{(\tau)}$. Let B_j be any bridge of $G^{(\tau)} \cup \{(x,y)\}$ w.r.t. $\lambda^{(\tau)}$. Since by hypothesis $G^{(\tau)} \cup \{(x,y)\}$ is planar, by Lemma 2.10, $\lambda^{(\tau)} \cup B_j$ is planar. Since $\{s_{p-1}, t_{p-1}\}$, $\{s_p, t_p\}$, $\{s_{q-1}, t_{q-1}\}$, and $\{s_q, t_q\}$ are all separation pairs in $G_3(\pi)$, the subgraphs $H_{2,p-1}$, $H_{p,p}$, $H_{p+1,q-1}$, $H_{q,q}$, $H_{q+1,\ell-1}$ are all edge-disjoint. Consequently, each bridge B_j of $G^{(\tau)} \cup \{(x,y)\}$ w.r.t. $\lambda^{(\tau)}$ is contained in at most one such subgraph.

Recall that $\widehat{G}$ is obtained from $G^{(\tau)}$ by replacing $H_{q+1,\ell-1}$ with the wheel $G'_{q+1,\ell-1}$ on vertices s_q, t_q, $s_{\ell-1}$, and $t_{\ell-1}$. Let $\widehat{\lambda}$ be the simple cycle of $\widehat{G}$ obtained from $\lambda^{(\tau)}$ by replacing the path between s_q and $s_{\ell-1}$ with the edge $(s_q, s_{\ell-1})$ of $H'_{q+1,\ell-1}$, and path between t_q and $t_{\ell-1}$ with the edge $(t_q, t_{\ell-1})$ of $H'_{q+1,\ell-1}$.

Note that in $\widehat{G} \cup \{(x,y)\}$ the bridges of $H_{q+1,\ell-1}$ are replaced with the bridges of $H'_{q+1,\ell-1}$. Since the other bridges not in $H_{q+1,\ell-1}$ are left unchanged, we have that for each such unchanged bridge B_j, also $\widehat{\lambda} \cup B_j$ is planar. We now show that this is true also for the bridges of $H'_{q+1,\ell-1}$. Note that $H'_{q+1,\ell-1}$ gives rise to three bridges in $\widehat{\lambda}$: the edge (s_q, t_q), the edge $(s_{\ell-1}, t_{\ell-1})$, and β_3 that contains the hub of the wheel and has attachments $\{s_q, t_q, s_{\ell-1}, t_{\ell-1}\}$. It is easy to see that $\widehat{\lambda} \cup \{(s_q, t_q)\}$, $\widehat{\lambda} \cup \{(s_{\ell-1}, t_{\ell-1})\}$, and $\widehat{\lambda} \cup \beta_3$ are all planar, and that (s_q, t_q), $(s_{\ell-1}, t_{\ell-1})$ and β_3 do not interlace in $\widehat{\lambda}$.

In summary, for every bridge B_j of $\widehat{\lambda}$, $\widehat{\lambda} \cup B_j$ is planar. By Lemma 2.8 the non-planarity of $\widehat{G} \cup \{(x,y)\}$ implies that there must be three interlacing bridges (namely each two of them interlace) in $\widehat{\lambda}$. Since $\widehat{G} \cup \{(x,y)\}$ differs from the planar graph $G^{(\tau)} \cup \{(x,y)\}$ only because it has $H'_{q+1,\ell-1}$ in place of $H_{q+1,\ell-1}$, at least one of such bridges must be in $H'_{q+1,\ell-1}$. Note that the attachments of the bridges (s_q, t_q) and $(s_{\ell-1}, t_{\ell-1})$ are properly contained in the attachments of β_3 and β_3 does not interlace with any one of the two. Consequently, β_3 interlaces with any bridge that interlaces with either (s_q, t_q) or $(s_{\ell-1}, t_{\ell-1})$. This implies that if there are three interlacing bridges β', β'' and β''' in $\widehat{\lambda}$, with β''' being in $H'_{q+1,\ell-1}$ and $\beta''' \neq \beta_3$, then also β', β'' and β_3 interlace in $\widehat{\lambda}$. This shows that without loss of generality, we can assume that one of the three bridges interlacing in $\widehat{\lambda}$ is β_3.

We now characterize the other two bridges of $\widehat{G} \cup \{(x,y)\}$ interlacing with β_3, say β_1 and β_2, showing that β_1 and β_2 must be in $G_0 \cup \{(x,y)\}$. Recall that $\widehat{G} \cup \{(x,y)\} = H_{2,p-1} \cup H_{p,p} \cup H_{p+1,q-1} \cup H_{q,q} \cup H'_{q+1,\ell-1} \cup G_0 \cup \{(x,y)\}$. Note that β_1 and β_2 cannot be in $H'_{q+1,\ell-1}$, since $H'_{q+1,\ell-1}$ consists only of β_3 plus the two edges (s_q, t_q) and $(s_{\ell-1}, t_{\ell-1})$ (which do not interlace with β_3). Thus, to prove that β_1 and β_2 are in $G_0 \cup \{(x,y)\}$, it is enough to show that they cannot be in $H_{2,p-1} \cup H_{p,p} \cup H_{p+1,q-1} \cup H_{q,q}$. Consider the cycle $\lambda^{(\tau)}$ in $G^{(\tau)}$. By Lemma 4.20, $\lambda^{(\tau)}$ contains vertices $s_1, \ldots, s_q, \ldots, s_{\ell-1}, t_{\ell-1}, \ldots, t_q, \ldots, t_1$, and they are encountered in this order while going around $\lambda^{(\tau)}$. Furthermore, the boundary vertices of $H_{q+1,\ell-1}$ are s_q, t_q, $s_{\ell-1}$, and $t_{\ell-1}$. Partition $\lambda^{(\tau)}$ into four paths:

(i) $\lambda(t_q, s_q)$ containing vertices t_j and s_j, $1 \leq j \leq q$.

(ii) $\lambda(s_q, s_{\ell-1})$ containing vertices s_j, $q \leq j \leq \ell - 1$.

(iii) $\lambda(s_{\ell-1}, t_{\ell-1})$ containing vertices $s_{\ell-1}$ and $t_{\ell-1}$.

(iv) $\lambda(t_{\ell-1}, t_q)$ containing vertices t_j, $q \leq j \leq \ell - 1$.

Since $\{(s_1, t_1)\} \cup H_{2,p-1} \cup H_{p,p} \cup H_{p+1,q-1} \cup H_{q,q} \cup \{(s_q, t_q)\}$ is a split graph of $G_3(\pi)$ with respect to separation pair $\{s_q, t_q\}$, the attachments of bridges of $\lambda^{(\tau)} \cup H_{2,p-1} \cup H_{p,p} \cup H_{p+1,q-1} \cup H_{q,q}$ w.r.t. $\lambda^{(\tau)}$ are all contained in $\lambda(t_q, s_q)$. Recall that $\widehat{\lambda}$ can be obtained from $\lambda^{(\tau)}$ by replacing $\lambda(s_q, s_{\ell-1})$ with the edge $(s_q, s_{\ell-1})$, and $\lambda(t_{\ell-1}, t_q)$ with the edge $(t_{\ell-1}, t_q)$. Consequently, also the attachments of bridges of $\widehat{\lambda} \cup H_{2,p-1} \cup H_{p,p} \cup H_{p+1,q-1} \cup H_{q,q}$ w.r.t. $\widehat{\lambda}$ are all contained in $\lambda(t_q, s_q)$. Since β_3

has attachments in $\{s_q, s_{\ell-1}, t_{\ell-1}, t_q\}$, this implies that no bridge of $\widehat{G} \cup \{(x,y)\}$ contributed by $H_{2,p-1} \cup H_{p,p} \cup H_{p+1,q-1} \cup H_{q,q}$ interlaces in $\widehat{\lambda}$ with β_3. Since β_1 and β_2 can be neither in $H'_{q+1,\ell-1}$ nor in $H_{2,p-1} \cup H_{p,p} \cup H_{p+1,q-1} \cup H_{q,q}$, they must be entirely contained in $G_0 \cup \{(x,y)\}$.

Due to the fact that $H_2 \cap G_0 = \{s_{\ell-1}, s_1, t_1, t_{\ell-1}\}$, the only possible attachments of bridges of $G_0 \cup \{(x,y)\}$ in $\widehat{\lambda}$ are the four vertices $s_{\ell-1}$, s_1, t_1, and $t_{\ell-1}$. Since β_1 and β_2 interlace with β_3, they must have at least one attachment in $\{s_1, t_1\}$ and at least one attachment in $\{s_{\ell-1}, t_{\ell-1}\}$. By Lemma 4.18 (applied to $G = G^{(\tau)} \cup \{(x,y)\}$, an expansion of $G^{(\tau)}$ w.r.t. $\mathcal{C}$), if they have one attachment in $\{s_1, t_1\}$ (say s_1) they must also have the other attachment (t_1). Indeed if such a bridge had attachment s_1 but not t_1, $G_1^{(s_1)}$ would not be empty and therefore by Lemma 4.18 $s_1 = s_{\ell-1}$. But then this bridge would have at most the two attachments $s_1 = s_{\ell-1}$ and $t_{\ell-1}$ and therefore would not interlace with β_3.

This implies that the only possibilities for β_1 and β_2 to interlace with β_3 are either (i) both β_1 and β_2 include as attachments $\{s_1, t_1, s_{\ell-1}\}$; or (ii) both β_1 and β_2 include as attachments $\{s_1, t_1, t_{\ell-1}\}$; or (iii) one of them (say β_1) includes as attachments $\{s_1, t_1, s_{\ell-1}\}$, and the other (say β_2) includes as attachments $\{s_1, t_1, t_{\ell-1}\}$ (see Figure 5).

[Figure 5]

Since $G^{(\tau)}$ differs from $\widehat{G}$ only because it has $H_{q+1,\ell-1}$ rather than $H'_{q+1,\ell-1}$, β_1 and β_2 are bridges in $\lambda^{(\tau)}$ too. We now show that there must be a bridge in $H_{q+1,\ell-1}$ that is interlacing with β_1 and β_2. Indeed, consider $\pi_{q+1,\ell-1} = \{\gamma_{q+1}, \ldots, \gamma_{\ell-1}\}$. Since by hypothesis it gives rise to a wheel replacement, $\pi_{q+1,\ell-1}$ consists of at least two nodes or equivalently, $q+1 < \ell-1$. (recall that if $\pi_{q+1,\ell-1}$ had only node, it would have given rise to a node compression). Moreover, by Lemma 4.15 it cannot be $s_q = s_{\ell-1}$ and $t_q = t_{\ell-1}$, and since we assumed that $H'_{q+1,\ell-1}$ is a wheel on four vertices, $s_q \neq s_{\ell-1}$ and $t_q \neq t_{\ell-1}$. Since by Fact 2.1 no two P-nodes can be adjacent, we cannot have that both γ_q or γ_{q+1} are P-nodes. This implies that there must be at least one B-node or one T-node γ_r, for $r = q$ or $r = q+1$. By Lemma 4.16, there is a path in Γ_r between vertices s_q and t_q (this is true if $r = q$ and if $r = q+1$). This path is a part of a bridge B_j of H_2 whose attachments include s_q and t_q.

We observe that γ_q, the allocation node of t_1, is in the chain π. Indeed it is in π because otherwise we would have $\gamma_q = \gamma_\ell$ and the wheel $H'_{q+1,\ell-1}$ would not exist. Consequently, $t_q \neq t_1$. Since $t_q \neq t_1$, and as said before, $s_q \neq s_{\ell-1}$ and $t_q \neq t_{\ell-1}$, the bridge $B_j \subseteq H_2$ interlaces with β_1 and β_2. In summary, also in $H_2 \subseteq G^{(\tau)}$ there is a bridge of $G^{(\tau)}$ w.r.t. $\lambda^{(\tau)}$ that is interlacing with β_1 and β_2 as well. By Lemma 2.8, $G^{(\tau)} \cup \{(x,y)\}$ is not planar, which contradicts the hypothesis.

The argument for a wheel replacement in $H_{p+1,q-1}$ is similar to the argument used for a wheel replacement in $H_{q+1,\ell-1}$. The only difference is that now the simple cycle $\widehat{\lambda}$ is obtained from $\lambda^{(\tau)}$ by substituting the edges (s_p, s_{q-1}) and (t_{q-1}, t_p) in place of the corresponding paths in $\lambda^{(\tau)}$. If we have two wheel replacements (both $H_{q+1,\ell-1}$ and $H_{p+1,q-1}$), then the same argument used for the first wheel replacement applies also to the second wheel replacement. $\square$

Before proving that also the substitution for blocking chains is planarity-preserving, we need few technical lemmas.

Lemma 4.25 *Let $\widehat{\pi}$ be a blocking chain with terminals $\widehat{e}_1 = (\widehat{s}_1, \widehat{t}_1)$ and $\widehat{e}_{\ell-1} = (\widehat{s}_{\ell-1}, \widehat{t}_{\ell-1})$. Let $\widehat{\pi}$ be contained in a graph $\widehat{G}$. Let $G_3(\widehat{\pi})$ be the graph corresponding to the chain $\widehat{\pi}$, and let $\widehat{H}_2 =$*

$(\widehat{V}_2, \widehat{E}_2)$ *be the graph* $G_3(\widehat{\pi}) - \{\widehat{e}_1, \widehat{e}_{\ell-1}\}$*. Let* x *and* y *be any two vertices of* $\widehat{G}$ *external to* $\widehat{H}_2$*. Let* $\widehat{\mathcal{B}}$ *be the set of bridges of* $\widehat{G} \cup \{(x,y)\}$ *w.r.t.* $\widehat{V}_2$ *(the vertex set of* $\widehat{H}_2$*). If* $\widehat{G} \cup \{(x,y)\}$ *is planar then:*

- *(i) If* $\widehat{\pi}$ *is of type 2, the attachments of bridges in* $\widehat{\mathcal{B}}$ *are contained either in* $\{\widehat{s}_1, \widehat{t}_1, \widehat{t}_{\ell-1}\}$ *or in* $\{\widehat{s}_{\ell-1}, \widehat{t}_{\ell-1}\}$*.*
- *(ii) If* $\widehat{\pi}$ *is of type 3, the attachments of bridges in* $\widehat{\mathcal{B}}$ *are contained either in* $\{\widehat{s}_1, \widehat{t}_1, \widehat{s}_{\ell-1}\}$ *or in* $\{\widehat{s}_{\ell-1}, \widehat{t}_{\ell-1}\}$*.*
- *(iii) If* $\widehat{\pi}$ *is of type 4, then the attachments of bridges in* $\widehat{\mathcal{B}}$ *are contained either in* $\{\widehat{s}_1, \widehat{t}_1\}$ *or in* $\{\widehat{s}_{\ell-1}, \widehat{t}_{\ell-1}\}$*.*

Proof: First, recall that Lemma 4.18 (applied to $G = \widehat{G} \cup \{(x,y)\}$) rules out the possibility of having certain bridges in $\widehat{\mathcal{B}}$. Indeed if $\widehat{s}_1 \neq \widehat{s}_{\ell-1}$ then any bridge in $\widehat{\mathcal{B}}$ containing $\widehat{s}_1$ must contain $\widehat{t}_1$ too, and similarly if $\widehat{t}_1 \neq \widehat{t}_{\ell-1}$ any bridge in $\widehat{\mathcal{B}}$ containing $\widehat{t}_1$ must contain $\widehat{s}_1$ too. Therefore the only attachments feasible for a bridge in $\widehat{\mathcal{B}}$ are: (a) $\widehat{s}_1$, $\widehat{t}_1$, $\widehat{s}_{\ell-1}$, $\widehat{t}_{\ell-1}$; (b) $\widehat{s}_1$, $\widehat{t}_1$, $\widehat{s}_{\ell-1}$; (c) $\widehat{s}_1$, $\widehat{t}_1$, $\widehat{t}_{\ell-1}$; (d) $\widehat{s}_1$, $\widehat{t}_1$; (e) $\widehat{s}_{\ell-1}$, $\widehat{t}_{\ell-1}$; (f) $\widehat{s}_{\ell-1}$; (g) $\widehat{t}_{\ell-1}$. Note that (b) - (g) above include also the cases $\widehat{t}_1 = \widehat{t}_{\ell-1}$ and $\widehat{s}_1 = \widehat{s}_{\ell-1}$.

Assume that $\widehat{\pi}$ is of type 2. By Corollary 4.3 there is no embedding of $G_3(\widehat{\pi})$ having $\widehat{e}_1$ and $\widehat{s}_{\ell-1}$ in the same face. Let $\widehat{p}_1$ and $\widehat{p}_{\ell-1}$ be the two paths outside $\widehat{H}_2$ given by Lemma 4.14. Since $\widehat{H}_2 \cup \widehat{p}_1 \cup \widehat{p}_{\ell-1}$ is homeomorphic to $G_3(\widehat{\pi})$, there is no embedding of $\widehat{H}_2 \cup \widehat{p}_1 \cup \widehat{p}_{\ell-1}$ with $\widehat{s}_1$, $\widehat{t}_1$, and $\widehat{s}_{\ell-1}$ in the same face. If there were a bridge B_j in $\widehat{\mathcal{B}}$ with attachments $\widehat{s}_1$, $\widehat{t}_1$, and $\widehat{s}_{\ell-1}$, then $\widehat{H}_2 \cup \widehat{p}_1 \cup \widehat{p}_{\ell-1} \cup B_j \subseteq \widehat{G} \cup \{(x,y)\}$ would not be planar. But since $\widehat{G} \cup \{(x,y)\}$ is supposed to be planar, such a bridge cannot exists. This rules out bridges with attachments $\widehat{s}_1$, $\widehat{t}_1$, and $\widehat{s}_{\ell-1}$, and leaves as only possibilities bridges with attachments either in $\{\widehat{s}_1, \widehat{t}_1, \widehat{t}_{\ell-1}\}$ or in $\{\widehat{s}_{\ell-1}, \widehat{t}_{\ell-1}\}$. This proves (i). The proof of (ii) is similar.

To prove (iii), we observe that by Corollary 4.3 there is no embedding of $G_3(\widehat{\pi})$ having $\widehat{e}_1$ in the same face as $\widehat{s}_{\ell-1}$, and $\widehat{e}_1$ in the same face as $\widehat{t}_{\ell-1}$. This implies that there is no embedding of $\widehat{H}_2 \cup \widehat{p}_1 \cup \widehat{p}_{\ell-1}$ with $\widehat{s}_1$, $\widehat{t}_1$, and $\widehat{s}_{\ell-1}$ in the same face, and no embedding with $\widehat{s}_1$, $\widehat{t}_1$ and $\widehat{t}_{\ell-1}$ in the same face. As before, this rules out bridges with attachments $\widehat{s}_1$, $\widehat{t}_1$, $\widehat{s}_{\ell-1}$, and bridges with attachments $\widehat{s}_1$, $\widehat{t}_1$, $\widehat{t}_{\ell-1}$ and gives the lemma. $\square$

Lemma 4.26 *Let* $\pi = \{\gamma_2, \gamma_3, \ldots, \gamma_{\ell-1}\}$ *be a blocking chain in a planar graph* G*, and let* γ_i*,* γ_j *and* γ_k*,* $2 \leq i < j < k \leq \ell - 1$*, be three pure blocking* T*-nodes of the same type. Then deleting* γ_j *from* π *gives rise to a planarity-preserving transformation.*

Proof: Let h be the type of γ_i, γ_j and γ_k. Note that h must be greater than 1, since by Lemma 4.19 there can be at most one pure node of type 1 in a blocking chain. Consequently, we have to prove the lemma only for chains of type z, $2 \leq z \leq 4$.

Define $\pi' = \pi - \{\gamma_j\}$ as the chain obtained after the deletion of γ_j. Let Γ_j be the pertinent graph of γ_j with virtual edges $e_{j-1} = (s_{j-1}, t_{j-1})$ and $e_j = (s_j, t_j)$. Note that deleting γ_j results in deleting the edges in $\Gamma_j - \{e_{j-1}, e_j\}$ and identifying s_{j-1} with s_j and t_{j-1} with t_j. The chain π' is obtained from π by deleting γ_j and by identifying these two pairs of vertices. Let z be the type of the chain π. Because of γ_i, γ_j and γ_k, $z \geq h \geq 2$, and furthermore π' is also of type z.

Let G be the graph containing π. Let $G_3(\pi)$ be the graph corresponding to the chain π, and let $H_2 = (V_2, E_2)$ be the graph defined as $H_2 = G_3(\pi) - \{e_1, e_{\ell-1}\}$. Let G' be the graph obtained from G after deleting γ_j from π. Namely, G' is obtained from G after substituting π with π', or, what is the same, H_2 with $H_2'' = (V_2'', E_2'')$ defined as $H_2'' = G_3(\pi') - \{e_1, e_{\ell-1}\}$. Node γ_j splits π into three parts: $\pi_{2,j-1} = \{\gamma_2, \ldots, \gamma_{j-1}\}$, $\pi_{j,j} = \{\gamma_j\}$, and $\pi_{j+1,\ell-1} = \{\gamma_{j+1}, \ldots, \gamma_{\ell-1}\}$. Since $\gamma_i \in \pi_{2,j-1}$, $\pi_{2,j-1} \neq \emptyset$, which implies that $G_3(\pi_{2,j-1})$ is non-empty and has at least one actual edge. Similarly, since $\gamma_k \in \pi_{j+1,\ell-1}$, $\pi_{j+1,\ell-1} \neq \emptyset$, which implies that also $G_3(\pi_{j+1,\ell-1})$ is non-empty and has at least one actual edge. Pairs $\{s_{j-1}, t_{j-1}\}$ and $\{s_j, t_j\}$ are separation pairs in $G_3(\pi)$. Indeed they split $G_3(\pi)$ into three (non-empty) split graphs: $G_3(\pi_{2,j-1})$ containing s_1 and t_1, $G_3(\pi_{j,j}) = \Gamma_j$, and $G_3(\pi_{j+1,\ell-1})$ containing $s_{\ell-1}$ and $t_{\ell-1}$. Using this terminology, $G_3(\pi')$ is the merge of $G_3(\pi_{2,j-1})$ and $G_3(\pi_{j+1,\ell-1})$, and H_2'' is obtained from H_2 by deleting all the edges of $\Gamma_j - \{e_{j-1}, e_j\}$ and by identifying $s_{j-1} = s_j$ and $t_{j-1} = t_j$. Since $\pi' = \pi_{2,j-1}\, \pi_{j+1,\ell-1}$, $\{s_{j-1}, t_{j-1}\} = \{s_j, t_j\}$ is a separation pair in $G_3(\pi')$, and splits $G_3(\pi')$ into two (non-empty) split graphs: $G_3(\pi_{2,j-1})$ containing s_1 and t_1, and $G_3(\pi_{j+1,\ell-1})$ containing $s_{\ell-1}$ and $t_{\ell-1}$.

Let x and y be any two vertices of G external to H_2. Since x and y are external to H_2, x and y are vertices of G' too. To prove the lemma, we prove that $G \cup \{(x,y)\}$ is planar if and only if $G' \cup \{(x,y)\}$ is planar.

Assume first that $G \cup \{(x,y)\}$ is planar. Since Γ_j is by hypothesis a triconnected graph, there is a sequence of edge contractions such that, when applied to $\Gamma_j - \{e_{j-1}, e_j\}$, identifies $s_{j-1} = s_j$ and $t_{j-1} = t_j$, reducing $\Gamma_j - \{e_{j-1}, e_j\}$ to a single edge e between $s_{j-1} = s_j$ and $t_{j-1} = t_j$. Note that the same sequence of edge contractions applied to $G \cup \{(x,y)\}$ would produce $G' \cup \{(x,y), e\}$. Since edge contractions preserve planarity, the planarity of $G \cup \{(x,y)\}$ implies the planarity of $G' \cup \{(x,y), e\}$, and consequently the planarity of $G' \cup \{(x,y)\}$.

To prove the converse, assume that $G' \cup \{(x,y)\}$ is planar. Let $\mathcal{B}'$ be the graph composed of all the bridges of $G' \cup \{(x,y)\}$ with respect to V_2'', and let $\mathcal{B}$ be the graph composed of all the bridges of $G \cup \{(x,y)\}$ with respect to V_2. Since $G \cup \{(x,y)\}$ and $G' \cup \{(x,y)\}$ differ only in H_2, $\mathcal{B} = \mathcal{B}'$. By definition of $\mathcal{B}'$, $G' \cup \{(x,y)\} = H_2'' \cup \mathcal{B}'$ and $G \cup \{(x,y)\} = H_2 \cup \mathcal{B}'$, or what is the same $G \cup \{(x,y)\} = [G_3(\pi) - \{e_1, e_{\ell-1}\}] \cup \mathcal{B}'$ and $G' \cup \{(x,y)\} = [G_3(\pi') - \{e_1, e_{\ell-1}\}] \cup \mathcal{B}'$. We distinguish three cases according to the different types of π (recall that π and π' are of the same type).

If π (and consequently π') is of type 2 then by Lemma 4.19 γ_i, γ_j and γ_k must be all pure nodes of type 2. By definition of type 2 nodes, $t_i = t_j = t_{\ell-1}$, and because of Lemma 2.3 $t_r = t_{\ell-1}$, $i \leq r \leq \ell - 1$. This implies that $G_3(\pi_{2,j-1})$ contains s_1, t_1, and $t_{\ell-1}$ (this includes also the case $t_1 = t_{\ell-1}$), and that $G_3(\pi_{j+1,\ell-1})$ contains $s_{\ell-1}$ and $t_{\ell-1}$. By Lemma 4.25 applied to the chain π', the attachments of bridges in $\mathcal{B}'$ are contained either in $\{s_1, t_1, t_{\ell-1}\}$ or in $\{s_{\ell-1}, t_{\ell-1}\}$. Partition the graph $\mathcal{B}'$ into two graphs: $\mathcal{B}_1'$ containing only bridges with attachments in $\{s_1, t_1, t_{\ell-1}\}$, and $\mathcal{B}_2'$ containing only bridges with attachments in $\{s_{\ell-1}, t_{\ell-1}\}$. This partition is not uniquely defined, since there are bridges that can be put indifferently in $\mathcal{B}_1'$ or in $\mathcal{B}_2'$ (such as a bridge having as only attachment $t_{\ell-1}$). We assign each such bridge arbitrarily to $\mathcal{B}_1'$ (see Figure 6).

[Figure 6]

Define $G_{2,j-1} = [G_3(\pi_{2,j-1}) - \{e_1\}] \cup \mathcal{B}_1'$ and $G_{j+1,\ell-1} = [G_3(\pi_{j+1,\ell-1}) - \{e_{\ell-1}\}] \cup \mathcal{B}_2'$. Since both $G_3(\pi_{2,j-1})$ and $G_3(\pi_{j+1,\ell-1})$ contain at least one actual edge, $G_{2,j-1} \neq \emptyset$ and $G_{j+1,\ell-1} \neq \emptyset$ (recall that e_1 and $e_{\ell-1}$ are both virtual edges). All the possible attachments of bridges in $\mathcal{B}_1'$ (namely s_1, t_1

and $t_{\ell-1}$) are contained in $G_3(\pi_{2,j-1})$, and therefore $G_{2,j-1}$ is connected. Similarly, all the possible attachments of bridges in $\mathcal{B}'_2$ (namely $s_{\ell-1}$ and $t_{\ell-1}$) are contained in $G_3(\pi_{j+1,\ell-1})$, and therefore also $G_{j+1,\ell-1}$ is connected. Recall that $\mathcal{B}'_1$ and $\mathcal{B}'_2$ contain bridges of $G' \cup \{(x,y)\}$ w.r.t. V''_2, and that by definition any two bridges share at most their attachments. Since all the attachments of $\mathcal{B}'_1$ are in $G_2(\pi_{2,j-1})$ and all the attachments of $\mathcal{B}'_2$ are in $G_3(\pi_{j+1,\ell-1})$, and since $G_3(\pi_{2,j-1})$, Γ_j and $G_3(\pi_{j+1,\ell-1})$ are split graphs of $G_3(\pi)$ separated by split pairs $\{s_{j-1}, t_{\ell-1}\}$ and $\{s_j, t_{\ell-1}\}$, this implies that $G_{2,j-1} \cap \Gamma_j = \{s_{j-1}, t_{\ell-1}\}$, $\Gamma_j \cap G_{j+1,\ell-1} = \{s_j, t_{\ell-1}\}$, and $G_{2,j-1} \cap G_{j+1,\ell-1} = \{t_{\ell-1}\}$. So, $G_{2,j-1}$, Γ_j, and $G_{j+1,\ell-1}$ are split graphs of $G \cup \{(x,y)\}$ separated by $\{s_{j-1}, t_{\ell-1}\}$ and $\{s_j, t_{\ell-1}\}$. Since $\mathcal{B} = \mathcal{B}'$, also $G' \cup \{(x,y)\}$ is split into two split graphs: $G_{2,j-1}$ and $G_{j+1,\ell-1}$ separated by the split pair $\{s_{j-1}, t_{\ell-1}\} = \{s_j, t_{\ell-1}\}$.

By Corollary 2.1, $G \cup \{(x,y)\}$ is planar if $G_{2,j-1} \cup \{e_{j-1}\}$, Γ_j and $G_{j+1,\ell-1} \cup \{e_j\}$ are planar. But Γ_j is planar by Corollary 2.1 because $G_3(\pi)$ is planar (since G is planar). Again by Corollary 2.1, the planarity of $G' \cup \{(x,y)\}$ implies the planarity of $G_{2,j-1} \cup \{e_{j-1}\}$ and $G_{j+1,\ell-1} \cup \{e_j\}$. Thus, $G \cup \{(x,y)\}$ is planar.

A similar argument can be applied if π (and consequently π') is of type 3. Indeed by Lemmas 4.19 and 4.23 γ_i, γ_j and γ_k must be all pure nodes of type 3. By definition of type 3 nodes, $s_i = s_j = s_k = s_{\ell-1}$, and by Lemma 2.3 $s_r = s_{\ell-1}$, $i \leq r \leq \ell - 1$. This time $s_1, t_1,$and $s_{\ell-1}$ are in $G_3(\pi_{2,j-1})$, and $s_{\ell-1}$ and $t_{\ell-1}$ are in $G_3(\pi_{j+1,\ell-1})$. By Lemma 4.25, the attachments of bridges in $\mathcal{B}'$ are contained either in $\{s_1, t_1, s_{\ell-1}\}$ or in $\{s_{\ell-1}, t_{\ell-1}\}$. The same argument applied for type 2 chains shows that $\{s_{\ell-1}, t_{j-1}\}$ and $\{s_{\ell-1}, t_j\}$ are split pairs that separate $G \cup \{(x,y)\}$ into three split graphs $G_{2,j-1}$, Γ_j, and $G_{j+1,\ell-1}$. Furthermore, also $G' \cup \{(x,y)\}$ is split into two split graphs: $G_{2,j-1}$ and $G_{j+1,\ell-1}$ separated by the split pair $\{s_{\ell-1}, t_{j-1}\} = \{s_{\ell-1}, t_j\}$. Again by Corollary 2.1 the planarity of $G' \cup \{(x,y)\}$ implies the planarity of $G \cup \{(x,y)\}$.

If π is of type 4, then by Lemma 4.25, the attachments of bridges in $\mathcal{B}'$ are contained in $\{s_1, t_1\}$ and $\{s_{\ell-1}, t_{\ell-1}\}$. Again, these bridges imply that the pairs $\{s_{j-1}, t_{j-1}\}$ and $\{s_j, t_j\}$ split $G \cup \{(x,y)\}$ into three split graphs: (i) $G_{2,j-1}$ containing s_1 and t_1, (ii) Γ_j, and (iii) $G_{j+1,\ell-1}$ containing $s_{\ell-1}$ and $t_{\ell-1}$. The same argument applied to chains of type 2 shows that also in this case deleting γ_j from π is planarity-preserving. $\square$

Lemma 4.27 *Let π be a blocking chain. Then rule (T3) gives rise to planarity-preserving substitutions in G'.*

Proof: If π is a blocking chain, then rule (T3) consists first of deleting multiple blocking T-nodes of the same type. This is planarity-preserving because of Lemma 4.26. Then the subchains between the remaining blocking T-nodes are compressed: this is planarity-preserving because of Lemma 4.24 since these subchains are free. Finally, the remaining blocking T-nodes are compressed according to rule (T2), which is planarity-preserving because of Lemma 4.11. $\square$

We refer to the graph $G'' = (V'', E'')$ obtained by compressing G' by means of rules (T1), (T2), and (T3) as the *compressed version of* G' with boundary $V_{G',G}$. The following lemma is a consequence of Lemmas 4.12, 4.13, 4.24 and 4.27.

Lemma 4.28 *Let $G = (V, E)$ be a planar graph, and let $G' = (V', E')$ be a biconnected subgraph of G. Let $V_{G',G}$ be the boundary of G', and let $G'' = (V'', E'')$ be the compressed version of G' with boundary $V_{G',G}$. Then substituting G'' in place of G' in G is planarity-preserving.*

We now characterize the time and space required by the computation of G''.

Lemma 4.29 *Let $G' = (V', E')$ be an n'-vertex biconnected planar graph with boundary $S \subseteq V'$. Then G'', as defined above has $O(|S|)$ edges and vertices, and can be computed in $O(n')$ time.*

Proof: Let $T_3(G')$ be the tree of triconnected components of G'. By definition, $T_3(G')$ has $O(|S|)$ red nodes. Apply the sequence of rules (T1), (T2) and (T3) as described before. The effect of these transformation is as follows. (T1) deletes the black leaves in this tree. Let γ_i be either a red node or a black node of degree at least three, and let Γ_i be its pertinent graph. Denote by r_i the total number of red vertices in Γ_i (namely the red vertices that have γ_i as allocation node, plus the red vertices created by the tree edges incident to γ_i). Then (T2) compresses Γ_i to size $O(r_i)$. Finally (T3) compresses black chains in the tree to $O(1)$ size. Consequently, after all the rules (T1), (T2) and (T3) have been applied, the graph G'' obtained by applying all the possible merges to the compressed tree has size $O(|S|)$.

To bound the time need to compute G'', we observe that $T_3(G')$ can be computed in $O(n')$ time and is of size at most $O(n')$ [27]. We then have to apply a sequence of the three rules (T1), (T2), and (T3) to the tree of triconnected components, as described above. Rule (T1) consists of deleting black nodes of degree 1. Consequently, we can support the sequence of all the possible rules (T1) that are applicable in a total of $O(n')$ time by visiting $T_3(G')$ in a bottom-up fashion starting from the leaves.

We now consider rule (T2), which consists of applying a compression to the pertinent graph Γ_i of a node γ_i in the tree. If γ_i is either a B- or a P-node, rule (T2) can be implemented in time $O(|\Gamma_i|)$. The same time is required if γ_i is a T-node because of Lemma 4.11. Consequently, the total time required to apply all the rules (T2) to the tree is $O(\sum_i |\Gamma_i|)$, which is $O(n')$.

We now consider rule (T3). The black chains of the tree can be found in a total of $O(n')$. If a chain is free, we first have to find the allocation nodes $\gamma(s_1)$ and $\gamma(t_1)$. As shown after Lemma 2.3, the allocation nodes of all the vertices of G' can be computed once and for all before applying any (T3) rule in a total of $O(n')$ time. Then we apply a constant number of node compressions and wheel replacements. Let π be the free chain to be compressed, and let z be the number of vertices in $G_3(\pi)$. Node compressions in π require at most a total of $O(z)$ time as previously shown. All the wheel replacements in π delete at most $O(z)$ vertices and edges and add at most a constant number of vertices and edges, and therefore can be accomplished in a total of $O(z)$ time. Consequently, the compression of all the free chains requires a total of $O(n')$ time.

If a chain is of type h, $h \geq 1$, we have to find first the type of all the T-nodes in the chain. Checking the type of a T-node γ_i can be done by simply looking at the embedding of its pertinent graph Γ_i, and therefore can be supported in $O(|\Gamma_i|)$ time. This gives a total of $O(n')$ for all the possible black chains in the tree. At this point, we reduce to 2 the number of multiple T-nodes of the same type, as suggested by Lemma 4.26. This yields a constant number of blocking T-nodes and of free subchains among those blocking T-nodes. Compressing the blocking nodes and the free subchains requires again a total of $O(n')$ time for all the possible blocking chains.

Since we first apply only rule (T1), in total time $O(n')$, then only rule (T2), in total time $O(n')$, and then only rule (T3), in total time $O(n')$, the total time needed to compute G'' is $O(n')$. □

4.3 The Valid Compression of a Cluster

We now put the previous two steps together and describe the computation of $\widetilde{\mathcal{C}}$ starting from cluster $\mathcal{C}$. We first compute the connected components of $\mathcal{C}$, which can be done in $O(|\mathcal{C}|) = O(z)$ time. Denote by $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_p$, $p \geq 1$, the connected components of $\mathcal{C}$. Let $B(\mathcal{C}_i)$ be the number of boundary vertices of $\mathcal{C}$ that are in connected component $\mathcal{C}_i$, $1 \leq i \leq p$. Note that

$$\sum_{i=1}^{p} B(\mathcal{C}_i) = B(\mathcal{C}),$$

and

$$\sum_{i=1}^{p} |\mathcal{C}_i| = |\mathcal{C}| = O(z).$$

For each connected component $\mathcal{C}_i$, $1 \leq i \leq p$, do the following. Compute the tree $\mathcal{T}(\mathcal{C}_i)$ as described in Section 4.1. By Lemma 4.4, the time required for the computation of $\mathcal{T}(\mathcal{C}_i)$ is $O(|\mathcal{C}_i|)$. Furthermore, $\mathcal{T}(\mathcal{C}_i)$ has $O(B(\mathcal{C}_i))$ round and internal square nodes. Now, using the information about the pertinent graphs of the round nodes of $\mathcal{T}(\mathcal{C}_i)$, compute the graph associated with $\mathcal{T}(\mathcal{C}_i)$. Denote by $\mathcal{C}'_i$ this graph. In other words $\mathcal{C}'_i$ is the graph whose block tree is $\mathcal{T}(\mathcal{C}_i)$, and it can be reassembled starting from $\mathcal{T}(\mathcal{C}_i)$ in $O(|\mathcal{C}'_i|)$ time. Note that by Lemmas 4.1 and 4.3, the substitution of $\mathcal{C}'_i$ in place of $\mathcal{C}_i$ is planarity-preserving.

Denote by $B'_{i,1}, B'_{i,2}, \ldots, B'_{i,q_i}$, $q_i \geq 1$, the biconnected components of $\mathcal{C}'_i$. Note that

$$\sum_{j=1}^{q_i} |B'_{i,j}| = O(|\mathcal{C}'_i|).$$

Since by Lemma 4.4 there are $O(B(\mathcal{C}_i))$ round nodes in $\mathcal{T}(\mathcal{C}_i)$, we have that $q_i = O(B(\mathcal{C}_i))$. For each j, $1 \leq j \leq q_i$, do the following. Consider $B'_{i,j}$, and let $S_{i,j}$ be the set of all the vertices in $B'_{i,j}$ corresponding to square nodes of $\mathcal{T}(\mathcal{C}_i)$ that are either internal or red nodes. Note that by Lemma 4.4

$$\sum_{j=1}^{q_i} |S_{i,j}| = O(B(\mathcal{C}_i)).$$

Consequently,

$$\sum_{i=1}^{p} \sum_{j=1}^{q_i} |S_{i,j}| = \sum_{i=1}^{p} O(B(\mathcal{C}_i)) = O(B(\mathcal{C})).$$

For each j, $1 \leq j \leq q_i$, apply to $B'_{i,j}$ the compression of a biconnected subgraph described in Section 4.2 with boundary $S_{i,j}$. This yields a new graph $B''_{i,j}$, which by Lemma 4.29 has $O(|S_{i,j}|)$ vertices and edges, and can be computed in $O(|B'_{i,j}|)$ time. Moreover, by Lemma 4.28, the substitution of $B''_{i,j}$ in place of $B'_{i,j}$ is planarity-preserving.

Substituting $B''_{i,j}$ in place of $B'_{i,j}$, $1 \leq i \leq p$, $1 \leq j \leq q_i$ yields a new graph $\widetilde{\mathcal{C}}$ starting from cluster $\mathcal{C}$. We now show that $\widetilde{\mathcal{C}}$ is indeed a valid compression of cluster $\mathcal{C}$.

Lemma 4.30 *Given a cluster $\mathcal{C}$, $\widetilde{\mathcal{C}}$ as computed above is a valid compression of $\mathcal{C}$. The time required to compute $\widetilde{\mathcal{C}}$ is $O(z)$.*

Proof: To show that $\widetilde{\mathcal{C}}$ is a valid compression of $\mathcal{C}$, we need to show that the following three properties are true.

(i) $\widetilde{\mathcal{C}}$ contains all the boundary vertices of $\mathcal{C}$, and no other vertex of G external to $\mathcal{C}$;

(ii) $\widetilde{\mathcal{C}}$ has at most $O(B(\mathcal{C}))$ vertices and edges; and

(iii) the substitution of $\widetilde{\mathcal{C}}$ in place of $\mathcal{C}$ is planarity-preserving.

First notice that the manipulations used on $\mathcal{C}$ to produce $\widetilde{\mathcal{C}}$ maintain the boundary vertices of $\mathcal{C}$, and do not make use of any other vertices if G external to $\mathcal{C}$. Consequently, property (i) is true.

To prove (ii), we observe that the total size of $\widetilde{\mathcal{C}}$ is

$$\sum_{i=1}^{p}\sum_{j=1}^{q_i}|B''_{i,j}| = \sum_{i=1}^{p}\sum_{j=1}^{q_i}O(|S_{i,j}|) = \sum_{i=1}^{p}O(B(\mathcal{C}_i)) = O(B(\mathcal{C})).$$

We now turn to (iii). Note that $\widetilde{\mathcal{C}}$ is obtained by first replacing each connected component $\mathcal{C}_i$ of $\mathcal{C}$ with $\mathcal{C}'_i$ obtained as shown in Section 4.1. By Lemmas 4.1 and 4.3, the substitution of $\mathcal{C}'_i$ in place of $\mathcal{C}_i$ is planarity-preserving for vertices external to $\mathcal{C}$. Next, each biconnected component $B'_{i,j}$ of $\mathcal{C}'_i$ is substituted with $B''_{i,j}$ using the compression described in Section 4.2. By Lemma 4.28, also this substitution is planarity-preserving for vertices external to $\mathcal{C}$. As a result, also property (iii) is true. This proves that $\widetilde{\mathcal{C}}$ is a valid compression of $\mathcal{C}$.

We now analyze the time required to compute $\widetilde{\mathcal{C}}$ starting from $\mathcal{C}$. The connected components $\mathcal{C}_i$, $1 \leq i \leq p$, of $\mathcal{C}$ can be computed in a total of $O(z)$ time. Then the computation of $\mathcal{C}'_i$ from $\mathcal{C}_i$ requires $O(|\mathcal{C}_i|)$ time because of Lemma 4.4. This gives again a total time bound of

$$O(\sum_{i=1}^{p}|\mathcal{C}_i|) = O(|\mathcal{C}|) = O(z).$$

Finally, each biconnected component $B'_{i,j}$ of $\mathcal{C}'_i$ is compressed. By Lemma 4.29, this requires a total of

$$O(\sum_{i=1}^{p}\sum_{j=1}^{q_i}|B'_{i,j}|) = O(\sum_{i=1}^{p}|\mathcal{C}'_i|) = O(|\mathcal{C}|) = O(z)$$

time. □

5 The Fully Dynamic Algorithm

We now describe our algorithm. Let e be any edge of G. We define the *proper cluster* of e, denoted by $\mathcal{C}_e$, as the cluster containing edge e (recall that an edge is contained in exactly one cluster of the partition). Let x be any vertex of G. We define the *proper cluster* of x, denoted by $\mathcal{C}_x$, as follows. If x is internal to a cluster $\mathcal{C}$, then $\mathcal{C}_x = \mathcal{C}$. Otherwise x is a boundary vertex: $\mathcal{C}_x$ is arbitrarily chosen as one of the clusters that have x in its boundary. Finally, for each boundary vertex v, we denote by $\mathcal{B}(v)$ the set of clusters that have v in their boundary. If v is an internal vertex, $\mathcal{B}(v) = \emptyset$.

5.1 Test(x, y)

To support a *Test*(x, y) operation we proceed as follows. Let $\mathcal{C}_x$ and $\mathcal{C}_y$ be the proper clusters of x and y respectively. Note that $\mathcal{C}_x$ and $\mathcal{C}_y$ may not be necessarily be distinct. Then we replace all the clusters but $\mathcal{C}_x$ and $\mathcal{C}_y$ with their valid compression, and leave $\mathcal{C}_x$ and $\mathcal{C}_y$ unaltered. This gives rise to a new graph $\mathcal{G}_{x,y}(G)$, called the *supergraph induced by x and y*.

Lemma 5.1 *$\mathcal{G}_{x,y}(G)$ has $O(z+\frac{n}{\sqrt{z}})$ vertices and edges. Furthermore, it can be computed in $O(z+\frac{n}{\sqrt{z}})$ time.*

Proof: By Lemma 4.30, the valid compression of a cluster $\mathcal{C}$ has $O(B(\mathcal{C}))$ vertices and edges, where $B(\mathcal{C})$ is the number of boundary vertices in $\mathcal{C}$. As a result, $\mathcal{G}_{x,y}(G)$ has

$$O(z+\sum_{\mathcal{C}\neq\mathcal{C}_x,\mathcal{C}_y} B(\mathcal{C}))$$

vertices and edges and can be computed in the same time bound. The lemma now follows from the bound on the total number of boundary vertices, namely

$$\sum_{\mathcal{C}} B(\mathcal{C}) = O(\frac{n}{\sqrt{z}}).$$

□

We now run the off-line planarity testing algorithm of Hopcroft and Tarjan [28] on $\mathcal{G}_{x,y}(G) \cup \{(x,y)\}$. Because of Lemma 5.1 this requires $O(z+\frac{n}{\sqrt{z}})$ time. If $\mathcal{G}_{x,y}(G) \cup \{(x,y)\}$ is planar we return *true*, and *false* otherwise. The correctness of this approach derives from the following lemma.

Lemma 5.2 *$\mathcal{G}_{x,y}(G) \cup \{(x,y)\}$ is planar if and only if $G \cup \{(x,y)\}$ is planar.*

Proof: $\mathcal{G}_{x,y}(G)$ is obtained from G by applying a sequence of rules (B1), (B2), (T1), (T2) and (T3) inside all the clusters but $\mathcal{C}_x$ and $\mathcal{C}_y$. The lemma now follows since by Lemmas 4.1, 4.3, 4.12, 4.13, 4.24 and 4.27 all these transformations are planarity-preserving. □

As a straightforward consequence of Lemmas 5.1 and 5.2 we have the following lemma.

Lemma 5.3 *Each* Test *operation can be supported in $O(z+\frac{n}{\sqrt{z}})$ time.*

5.2 Add(x,y) and Delete(x,y)

We first describe an implementation of the primitive *add*(x,y) which takes care of the actual insertion of the edge (x,y). We assume that the edge (x,y) keeps the graph planar. Indeed, as mentioned in Section 1, an *Insert*(x,y) operation is carried out by first executing a *Test*(x,y). If it returns *true*, then the primitive *add*(x,y) is carried out. Otherwise, the edge is not inserted.

Let $\mathcal{C}_x$ and $\mathcal{C}_y$ be the proper clusters of x and y respectively. We assign the new edge (x,y) to cluster $\mathcal{C}_x$. This implies that y becomes a boundary vertex of both $\mathcal{C}_x$ and $\mathcal{C}_y$, if it was not already so. Indeed if $\mathcal{C}_x \notin \mathcal{B}(y)$ [respectively $\mathcal{C}_y \notin \mathcal{B}(y)$], then we add $\mathcal{C}_x$ [respectively $\mathcal{C}_y$] to $\mathcal{B}(y)$. Consequently, the insertion of edge (x,y) causes changes to both cluster $\mathcal{C}_x$ and $\mathcal{C}_y$, and to their valid compressions. The change in the clusters can be easily taken into account. By Lemma 4.30, the valid compressions of $\mathcal{C}_x$ and $\mathcal{C}_y$ can be recomputed from scratch in $O(z)$ time.

When an edge (x,y) is deleted, let $\mathcal{C}_{(x,y)}$ be the cluster containing edge (x,y). Then it might happen that, because of the deletion of edge (x,y), either one of its endpoints ceases to be a boundary vertex. We describe how to detect this for x, and the same argument applies to y. We first check whether there are other edges of $\mathcal{C}_{(x,y)}$ (beside (x,y)) that are incident to x. If there are other edges, then x does not change its status and remains either boundary or internal as before. If there are no other edges of $\mathcal{C}_{(x,y)}$ incident to x, we check $\mathcal{B}(x)$. If $|\mathcal{B}(x)| = 0$ then x was not a boundary vertex, and again nothing changes. Otherwise, since there are no more edges of $\mathcal{C}_{(x,y)}$ (other than (x, y) itself) incident to x, then x is no longer a boundary vertex in $\mathcal{C}_{(x,y)}$: we delete $\mathcal{C}_{(x,y)}$ from $\mathcal{B}(x)$. If now $\mathcal{B}(x)$ contains only one cluster, say $\mathcal{C}'$, then x ceases to be a boundary vertex also in $\mathcal{C}'$; therefore $\mathcal{C}'$ is removed from $\mathcal{B}(x)$. In summary, $\mathcal{C}_{(x,y)}$ and at most two other clusters and their valid compressions have to be updated. As before, this can be done in no more than $O(z)$ time.

Let (x, y) be the edge inserted [respectively deleted]. In addition to the changes in a constant number of clusters, the insertion [respectively deletion] of (x, y) affects the partition in two ways. First, the total number of boundary vertices is increased at most by two [respectively decreased at most by four]. Second, there is at most one cluster ($\mathcal{C}_x$ in the case of insertions and $\mathcal{C}_{(x,y)}$ in the case of deletions) whose size increases [respectively decreases] by one. To maintain the desired invariants of the partition (namely $O(\frac{n}{z})$ clusters of size $O(z)$ each, and with at most $O(\frac{n}{\sqrt{z}})$ total boundary vertices), we recompute the partition from scratch exactly every $T(z, n) = \min\{z, \frac{n}{\sqrt{z}}\}$ *Insert* operations. After $T(z, n)$ *Insert* operations the total increase in the size of all clusters can be at most z and the total increase in the number of boundary vertices can be at most $\frac{2n}{\sqrt{z}}$. Note that *Test* and *Delete* do not increase the size of any cluster or the total number of boundary vertices. Therefore if the invariants of the partition were holding before $T(z, n)$ *Insert* operations, they will still hold afterwards. Since recomputing the partition from scratch requires $O(n)$ time, this charges $O(\frac{n}{T(z,n)})$ amortized time to each *Insert* operation.

Lemma 5.4 *Each* Insert *operation can be supported in* $O(z + \frac{n}{\sqrt{z}} + \frac{n}{T(z,n)})$ *amortized time, while each* Delete *operation requires* $O(z)$ *worst-case time.*

Proof: Inserting edge (x, y) requires first to carry out *Test*(x, y) plus the actual insertion of edge (x, y). By Lemma 5.3, a *Test* operation can be supported in $O(z + \frac{n}{\sqrt{z}})$ time. As said before, the actual insertion of a new edge can be done in $O(z)$ time plus the $O(\frac{n}{T(z,n)})$ amortized time charged for recomputing the partition every $T(z, n)$ *Insert* operations. □

Theorem 5.1 *Each* Test*, and* Delete *can be supported in* $O(n^{2/3})$ *worst-case time, while an* Insert *requires* $O(n^{2/3})$ *amortized time.*

Proof: Fix $z = \lceil n^{2/3} \rceil$. Then $T(z, n) = \min\{z, \frac{n}{\sqrt{z}}\} = \Theta(n^{2/3})$. The claimed bounds now follow from Lemmas 5.3 and 5.4. □

6 Fully Dynamic Biconnectivity on Planar Graphs

In this section, we consider the problem of performing any sequence of the following three types of operations on a planar graph. As before, *Insert*(x, y) adds an edge (x, y) to G provided that the resulting graph remains planar, while *Delete*(x, y) removes the edge (x, y). *SameBiconnected*(x, y) returns *true* if x and y are biconnected, and returns *false* otherwise. Note that we allow changes in the embedding. Galil and Italiano [21] showed how to perform *Insert*, *Delete* and *SameBiconnected* in $O(n^{2/3})$ time in the worst case. We use a slight modification of the data structure presented in

Section 4 combined with techniques borrowed from [21]. This allows us to achieve the same bounds as in [21], but with the bound for insertions being amortized rather than worst-case. However, the new algorithm is simpler, and it is the base of a new fully dynamic triconnectivity algorithm that will be presented in Section 7.

Again, we partition the planar graph into $O(\frac{n}{z})$ edge clusters of size $O(z)$ each, in such a way the total number of boundary vertices in the partition will be $O(\frac{n}{\sqrt{z}})$. We then compress each cluster so as to preserve some important properties with respect to biconnectivity.

We now describe the techniques used to compress a cluster $\mathcal{C}$. The rules used are similar to rules (B1) and (B2) given in Section 4. Again, we start from the block tree $T_2(\mathcal{C})$ of $\mathcal{C}$. We color red all the square nodes of $T_2(\mathcal{C})$ corresponding to boundary vertices of $\mathcal{C}$, and we color black all the other nodes. The equivalent of rule (B1) is the following.

(R1) Delete any black (either round or square) leaf.

Applying repeatedly rule (R1) on $T_2(\mathcal{C})$ yields a block tree, whose leaves are all red. Again, there can be chains of degree-two black nodes, and we refer to them as *black chains*. The rule used to compress black chains is slightly different from (B2).

(R2) If there is any degree-two round node ρ whose two adjacent square nodes, say u and v, are both black, then delete round node ρ and merge u and v into a new square node (i.e., all the edges previously incident to u and v are now incident to the new node).

The tree obtained from $T_2(\mathcal{C})$ after applying repeatedly rules (R1) and (R2) is denoted by $\mathcal{T}(\mathcal{C})$ and called the *compression* of $\mathcal{C}$. We omit the proof of the following lemma which is similar to the proof of Lemma 4.4.

Lemma 6.1 *$\mathcal{T}(\mathcal{C})$ has $O(B(\mathcal{C}))$ round and square nodes and can be computed in $O(|\mathcal{C}|)$ time.*

We now describe how to perform our operations. Operations *Insert* and *Delete* are almost the same as in the case of our fully dynamic planarity algorithm. The only difference lies in the data structures that we maintain for each cluster $\mathcal{C}$. This time, we maintain the block tree $T_2(\mathcal{C})$ and its compression $\mathcal{T}(\mathcal{C})$. Again, all these data structures can be initialized in $O(|\mathcal{C}|)$ time.

To perform a *SameBiconnected*(x, y) operation, we do the following. Let $\mathcal{C}_x$ and $\mathcal{C}_y$ be the proper clusters of x and y as defined in Section 5. Once again, we replace all the clusters but $\mathcal{C}_x$ and $\mathcal{C}_y$ with their compression. Then we replace $\mathcal{C}_x$ and $\mathcal{C}_y$ with their block trees $T_2(\mathcal{C}_x)$ and $T_2(\mathcal{C}_y)$. We obtain a new graph which is called the *super-graph induced by x and y*. Similarly to the terminology used in Section 5, we denote this graph by $\mathcal{G}_{x,y}(G)$. By Lemma 6.1, $\mathcal{G}_{x,y}(G)$ has size $O(z + \frac{n}{\sqrt{z}})$.

The following lemma shows that $\mathcal{G}_{x,y}(G)$ satisfies nice properties with respect to biconnectivity.

Lemma 6.2 *There is an articulation point separating x and y in G if and only if there is at least one square node that is an articulation point separating x and y in $\mathcal{G}_{x,y}(G)$.*

Proof: Define $\widehat{G}$ to be the graph obtained from G by replacing each cluster with its block tree. Note that there are two square nodes x and y in both $\widehat{G}$ and $\mathcal{G}_{x,y}(G)$. We first show that

(i) there is an articulation point separating x and y in G if and only if there is a square node that is an articulation point separating x and y in $\widehat{G}$.

We then complete the proof by showing that

(ii) there is square node that is an articulation point separating x and y in $\widehat{G}$ if and only if there is at least one square node that is an articulation point separating x and y in $\mathcal{G}_{x,y}(G)$.

$\widehat{G}$ can be obtained from G by first making all the vertices of G square nodes and then by repeatedly applying the following transformation.

(S1) Biconnected subgraph replacement: Given a biconnected subgraph $\widetilde{G}$ of G, remove all the edges in $\widetilde{G}$ and introduce a round node ρ. Connect to ρ all the vertices in $\widetilde{G}$.

We can define a transformation which is the inverse of (S1): it takes a round node ρ and replace it with the corresponding biconnected subgraph of G.

If there is an articulation point separating x and y in G, then (S1) preserves this articulation point. It is possible to re-obtain G from $\widehat{G}$ by means of the inverse transformation of (S1), which again preserve articulation points. This proves (i).

To prove (ii) we notice that $\mathcal{G}_{x,y}(G)$ can be obtained from $\widehat{G}$ by means of a sequence of the following two transformations.

(S2) Leaf elimination: Given a (round or square) node v ($v \neq x$ and $v \neq y$) of degree one, delete v together with its incident edge.

(S3) Round node contraction: Given a round node ρ of degree two that is not adjacent to x and to y, denote by u and v the two square nodes adjacent to ρ. Delete round node ρ and merge u and v into a new square node (i.e., all the edges previously incident to u and v are now incident to the new node).

Indeed transformations (S2) and (S3) correspond to the two rules given earlier: (S2) corresponds to rule (R1), and (S3) to rule (R2). Again, we can define inverse transformations of (S2) and (S3). The inverse of (S2) adds a new leaf to the tree, and the inverse of (S3) replaces a square node with square nodes u and v joined by a degree-two round node ρ. If there is a square node that is an articulation point separating x and y in $\widehat{G}$, then transformations (S2) and (S3) preserve at least one square node that is an articulation point separating x and y. Indeed during (S2) v cannot be an articulation point separating x and y, while in (S3) if u is an articulation point separating x and y, also v must be an articulation point separating x and y and therefore the new square node will be an articulation point separating x and y too. Similarly, the inverse transformations of (S2) and (S3) preserve articulation points separating x and y from $\mathcal{G}_{x,y}(G)$. $\Box$

As a consequence of Lemma 6.2, we can check whether two vertices x and y are 2-vertex-connected by first constructing $\mathcal{G}_{x,y}(G)$, the super-graph induced by x and y, and then by checking whether there is a square node of $\mathcal{G}_{x,y}(G)$ that separates x and y. As in Section 5, this can be done in a total of $O(z + \frac{n}{\sqrt{z}})$ time, and gives the following theorem.

Theorem 6.1 *Each* SameBiconnected *and* Delete *operation can be supported in* $O(n^{2/3})$ *worst-case time, while an* Insert *operation requires* $O(n^{2/3})$ *amortized time.*

7 Fully Dynamic Triconnectivity on Planar Graphs

In this section, we present a fully dynamic algorithm for maintaining information about triconnectivity on planar graphs. In additions to the operations *Insert* and *Delete* as defined before, we would like to perform a *SameTriconnected*(x, y), which returns *true* if x and y are 3-vertex-connected, and returns *false* otherwise.

We maintain new data structures to check triconnectivity together with the data structure described in Section 6 for fully dynamic biconnectivity. To check whether x and y are 3-vertex-connected, we first check whether they can be separated by an articulation point. If they can, then x and y are not even 2-vertex-connected: we stop and return *false*. If there is no articulation point whose removal disconnects x and y, then x and y are in a same biconnected component: we check whether there is a separating pair for x and y in this biconnected component. If there is at least one, we return *false*. Otherwise, we return *true*.

Checking for articulation points while performing edge insertions and edge deletions can be done as shown in Section 6. We now concentrate on how to check for separation pairs. We show that we need very few new data structures. Indeed we use the substitutions (B1), (B2), (T1) and (T2) presented in Section 4, which turn out to be not only planarity-preserving, but also *triconnectivity-preserving*. The only new transformation we need here is a rule for compressing black chains: the new rule will be much simpler than rule (T3) given in Section 4. The definition of triconnectivity-preserving is similar to the definition of planarity-preserving and can be given as follows.

Let $G = (V, E)$ be a planar graph, and let $G_0 = (V_0, E_0)$ be a connected subgraph of G. Let $\overline{G_0} = (\overline{V_0}, \overline{E_0})$ be the complement of G_0 in G, and let $V_{G_0,G}$ be the boundary of G_0, as defined in Section 4. Let $G_1 = (V_1, E_1)$ be a graph that contains all the vertices in the boundary of G_0 but no other vertex of $\overline{G_0}$. Denote by $G' = (V', E')$ the graph obtained after substituting G_1 in place of G_0 in G. Fix any two vertices x and y external to G_0, such that x and y are 2-vertex-connected. If it happens that x and y are 3-vertex-connected in G if and only if they are 3-vertex-connected in G', we say that the substitution of G_1 in place of G_0 in G is *triconnectivity-preserving for x and y*. If this substitution is triconnectivity-preserving for any pair of 2-vertex-connected vertices external to G_0, we simply say that it is *triconnectivity-preserving*.

We now prove that the four substitutions (B1), (B2), (T1) and (T2) given in Section 4 are triconnectivity-preserving. We refer the reader to Section 4 for the definition of such substitutions and for the terminology used.

Lemma 7.1 *Rule (B1) gives rise to triconnectivity-preserving substitutions in $\mathcal{C}$.*

Proof: Recall that rule (B1) takes a biconnected subgraph G_1 of $\mathcal{C}$ that has exactly one articulation point v, and such that the only possible boundary vertex in G_1 is v. Then v is substituted in place of G_1. Namely, $G = G_1 \cup G_0$, such that $G_1 \cap G_0 = \{v\}$. Let $\widehat{G}$ be the graph obtained from G after this substitution: $\widehat{G} = G_0$. Let x and y be any two 2-vertex-connected vertices external to cluster $\mathcal{C}$. Since $G_1 \subseteq \mathcal{C}$, we have that $x, y \in G_0$. But then this implies that x and y are 3-vertex-connected in G if and only if they are 3-vertex-connected in $G_0 = \widehat{G}$. □

Lemma 7.2 *Rule (B2) gives rise to triconnectivity-preserving substitutions in $\mathcal{C}$.*

Proof: Recall that rule (B2) compresses a black chain π. Using the terminology of Section 4, G can be decomposed as $G = G_0 \cup G(\pi)$, where $G(\pi) = G_1 \cup G_2 \cup \ldots \cup G_p$. Since there are no red nodes in the chain π, the only possible boundary vertices of $V_1 \cup V_2 \cup \ldots \cup V_p$ are v_0 and v_p. As a result, any bridge of G w.r.t. $V_1 \cup V_2 \cup \ldots \cup V_p$ (the vertex set of $G(\pi)$) has attachments only in $\{v_0, v_p\}$. Namely, $G_0 \cap G(\pi) = \{v_0, v_p\}$. Then rule (B2) produces a graph $\widehat{G} = G_0 \cup G_j(v_0, v_p)$ for some j, $1 \leq j \leq p$, and consequently $\widehat{G}$ can be obtained from G by simply replacing $G(\pi)$ with $G_j(v_0, v_p)$. Again $G_0 \cap G_j(v_0, v_p) = \{v_0, v_p\}$.

Let x and y be any two 2-vertex-connected vertices external to cluster $\mathcal{C}$. Since the only possible boundary vertices of $G(\pi)$ are v_0 and v_p, $x, y \in G_0$. We now prove that rule (B2) is triconnectivity-preserving. Assume x and y are 3-vertex-connected in G. Then there are at least three vertex-disjoint paths between x and y in G. Note that at most one of them can have a subpath between v_0 and v_p in $G(\pi)$, while the other two must be entirely in G_0. Since $G(\pi)$ is replaced by $G_j(v_0, v_p)$, and there is still a path in $G_j(v_0, v_p)$ between v_0 and v_p, x and y must be 3-vertex-connected also in $\widehat{G}$. The converse can be proved with a similar argument. □

Lemma 7.3 *Rule (T1) gives rise to triconnectivity-preserving substitutions.*

Proof: As shown in Lemma 4.12, rule (T1) takes as input a graph $G = G_0 \cup H_1$, such that $H_1 \cap G_0 = \{a, b\}$ and $H_1 \cup \{(a, b)\}$ is a triconnected component in G, and it replaces H_1 with edge (a, b). That is, the graph obtained from G after applying rule (T1) is $\widehat{G} = G_0 \cup \{(a, b)\}$. Let x and y be any two 2-vertex-connected vertices external to $\mathcal{C}$. Then x and y must be in G_0. Similarly to the proof of Lemma 7.2, x and y are 3-vertex-connected in G if and and only if they are 3-vertex-connected in $\widehat{G}$. □

Lemma 7.4 *Rule (T2) gives rise to triconnectivity-preserving substitutions.*

Proof: We denote as usual by G the graph before applying rule (T2), and by $\widehat{G}$ the graph obtained after applying rule (T2). We distinguish the three cases as in Section 4, according to whether we have a type (T2.a), (T2.b), or (T2.c) rule.

First, we consider rule (T2.a) that consists of compressing a bond. As shown in Lemma 4.13, the change induced in G by this rule consists of deleting either multiple edges $e_{q+2}, e_{q+3}, \ldots, e_p$ or multiple edges $e_{q+3}, e_{q+4}, \ldots, e_p$. In other words, G can be decomposed as $G = G_0 \cup \{e_{q+1}\} \cup \ldots \cup \{e_p\}$, and after the transformation we have that either $\widehat{G} = G_0 \cup \{e_{q+1}\}$ or $\widehat{G} = G_0 \cup \{e_{q+1}\} \cup \{e_{q+2}\}$. Therefore, given any two vertices x and y in G_0, x and y are 3-vertex-connected in G if and only if they are 3-vertex-connected in $\widehat{G}$.

Next, we turn to rule (T2.b) that compresses a polygon. Let x and y two 2-vertex-connected vertices external to $\mathcal{C}$. As shown in Section 4, this rule contracts chains of vertices (different from x and y) of degree two into a constant number of edges. Obviously, x and y are 3-vertex connected in G if and only if they are 3-vertex-connected in $\widehat{G}$.

Finally, we analyze rule (T2.c) that consists of compressing a triconnected subgraph. This was defined in Section 4.2.1 as triconnected compression. We now prove that this triconnected compression is triconnectivity-preserving. Let G' be a the triconnected subgraph to be compressed, and let G'' be the triconnected compression of G' as defined in Section 4.2.1. As usual, let $G = G_0 \cup G'$ and $\widehat{G} = G_0 \cup G''$ the graph before and after applying this substitution, and let x, y be any two 3-vertex-connected vertices in G_0. Then there are three vertex-disjoint paths in G between x

and y. Denote by h, $0 \leq h \leq 3$, the number of such paths that contain at least one edge in G'. Notice that each such path must be between two boundary vertices of G'. Since by Lemma 4.11 G'' has the same boundary vertices as G', because of Lemma 2.1 there will be also h vertex-disjoint paths in G'' between the same boundary vertices. Consequently, x and y are 3-vertex-connected in $\widehat{G}$. The same argument proves that if x and y are 3-vertex-connected in $\widehat{G}$, they must be 3-vertex-connected in G. $\square$

We now define a new rule (T3') which compresses black chains in a triconnectivity-preserving fashion. Let $\pi = \{\gamma_1, \gamma_2, \ldots, \gamma_{\ell-1}\}$ be the black chain to be compressed with terminals $e_1 = (s_1, t_1)$ and $e_{\ell-1} = (s_{\ell-1}, t_{\ell-1})$. Let $G_3(\pi)$ be graph corresponding to π, and let $H_2 = G_3(\pi) - \{e_1, e_{\ell-1}\}$. Note that even though $G_3(\pi)$ is biconnected because of Lemma 2.2, H_2 is not necessarily biconnected. Our compression of H_2 is as follows. We apply rules (B1) and (B2) to H_2 with boundary vertices s_1, t_1, $s_{\ell-1}$ and $t_{\ell-1}$. Because of Lemma 4.4, this yields a graph H_2' with $O(1)$ biconnected components and consequently $O(1)$ articulation points. Color red s_1, t_1, $s_{\ell-1}$, $t_{\ell-1}$, and the articulation points of H_2': this gives a total of $O(1)$ red vertices. Replace each biconnected component of H_2' with a simple cycle through its red vertices. The new graph obtained is called H_2'', has $O(1)$ vertices and edges. Rule (T3') consists of substituting H_2'' in place of H_2.

Lemma 7.5 *Rule (T3') gives rise to triconnectivity-preserving substitutions.*

Proof: We denote as usual by G the graph before applying rule (T3'), and by $\widehat{G}$ the graph obtained after applying rule (T3'). Recall that transformation (T3') substitutes H_2'' in place of H_2 in the cluster $\mathcal{C}$ containing the black chain.

Rule (T3') consists of two steps. During the first step, we apply rules (B1) and (B2) with boundary $\{s_1, t_1, s_{\ell-1}, t_{\ell-1}\}$, thus compressing $H_2 \subset G$ to a new graph H_2'. Substituting H_2' in place of H_2 yields a new graph $\widetilde{G}$. This substitution is triconnectivity-preserving because of Lemmas 7.1 and 7.2.

As said before, H_2' has $O(1)$ biconnected components and $O(1)$ articulation points. Then we color red the four vertices s_1, t_1, $s_{\ell-1}$, $t_{\ell-1}$ and the $O(1)$ articulation points of H_2'. Fix a biconnected component B of H_2': notice that its boundary in the graph $\widetilde{G}$ is given by the red vertices in B. The second step consists of replacing each biconnected component of H_2' with a simple cycle through its red vertices.

To complete the proof of the lemma, we show that, given a graph G', and a biconnected subgraph $B \subseteq \mathcal{C}$, replacing B with a simple cycle C through its boundary gives rise to a triconnectivity-preserving transformation. Let G'' be the graph obtained after this replacement, and let x and y be any two vertices external to $\mathcal{C}$. Assume x and y are 3-vertex-connected in G'. Then there are three vertex-disjoint paths between x and y in G'. Since the boundary of H_2' in G' is given by $\{s_1, t_1, s_{\ell-1}, t_{\ell-1}\}$, at most two such paths can go through H_2': this implies that at most two such paths can go through $B \subseteq H_2'$. Let p_1' and p_2' be such paths, and let a_1, b_1 [respectively a_2, b_2] be the two vertices where path p_1' [respectively p_2'] gets into B and out of B. Note that a_1, a_2, b_1, b_2 are in the boundary of B and therefore are red vertices. Recall that B is replaced by a simple cycle C through its red vertices: there are now by Lemma 2.1 two vertex-disjoint paths p_1'' and p_2'' between $\{a_1, a_2\}$ and $\{b_1, b_2\}$.

If p_1'' and p_2'' have the same endpoints as p_1' and p_2' (namely p_1'' is between a_1 and b_1, and p_2'' is between a_2 and b_2), replacing p_1' and p_2' with p_1'' and p_2'' respectively gives two vertex-disjoint paths between x and y going through H_2'' in G''. Therefore, x and y are 3-vertex-connected in G'' too.

If p_1'' and p_2'' do not have the same endpoints as p_1' and p_2' (namely p_1'' is between a_1 and b_2, and p_2'' is between a_2 and b_1), there are still two vertex-disjoint paths between x and y going through H_2'' in G'': one going through a_1 and b_2, and the other going through a_2 and b_1. Again, x and y are 3-vertex-connected in G'' too.

Conversely, if x and y are 3-vertex-connected in G'', then at most two vertex-disjoint paths can go through the cycle C, and again by Lemma 2.1 there are two vertex-disjoint paths in the biconnected subgraph B which can be used to give two vertex-disjoint paths between x and y going through H_2'. This proves that x and y must be 3-vertex-connected in G' too. □

Lemmas 7.1, 7.2, 7.3, 7.4, and 7.5 imply the following theorem.

Theorem 7.1 *Each* SameTriconnected *and* Delete *operation can be supported in $O(n^{2/3})$ worst-case time, while an* Insert *operation requires $O(n^{2/3})$ amortized time.*

8 Conclusion

We have presented a sub-linear-time algorithm for the fully dynamic planarity testing problem. We are able to perform any sequence of *Test*, *Insert* and *Delete* operations in $O(n^{2/3})$ time each. The bounds for *Test* and *Delete* are worst-case, while the bound for *Insert* is amortized. Our algorithm yields also fully dynamic algorithms for maintaining information about biconnectivity and triconnectivity in planar graphs with the same time bounds. The heart of all these algorithms is the notion of compressed representation for the properties we wish to maintain. It would be interesting to study also fully dynamic planarity testing on embedded planar graphs. The algorithm given by Tamassia [46] runs in $O(\log n)$ time per operation; however, it works only for biconnected graphs, and it has significant restrictions on the edge deletions that can be supported.

Acknowledgments

We are grateful to Moti Yung for many fruitful discussions.

References

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *The Design and Analysis of Computer Algorithms.* Addison-Wesley, Reading, MA, 1974.

[2] G. Ausiello, G. F. Italiano, A. Marchetti-Spaccamela, and U. Nanni. Incremental algorithms for minimal length paths. *J. Algorithms*, 12:615–638, 1991.

[3] D. Bienstock and C. Monma. On the complexity of covering vertices by faces in a planar graph. *SIAM J. Comput.*, 17:53–76, 1988.

[4] K. Booth and G. Lueker. Testing for the consecutive ones property, interval graphs and graph planarity using pq-tree algorithms. *J. Comp. Syst. Sci.*, 13:335–379, 1976.

[5] R. F. Cohen and R. Tamassia. Dynamic expression trees and their applications. In *Proc. 2nd Annual ACM-SIAM Symp. on Discrete Algorithms*, pages 52–61, 1991.

[6] G. B. Dantzig and D. R. Fulkerson. On the max-flow min-cut theorem of networks. *Annals of Math. Study*, 38:215–221, 1956.

[7] G. Di Battista and R. Tamassia. Incremental planarity testing. In *Proc. 30th Annual Symp. on Foundations of Computer Science*, pages 436–441, 1989.

[8] G. Di Battista and R. Tamassia. On-line graph algorithms with SPQR-trees. In *Proc. 17th Int. Colloquium on Automata, Languages and Programming*, pages 598–611. Lecture Notes in Computer Science 443, Springer-Verlag, Berlin, 1990.

[9] J. Edmonds. A combinatorial representation for polyhedral surfaces. *Not. Am. Math. Soc.*, 7:646, 1960.

[10] D. Eppstein, Z. Galil, G. F. Italiano, and A. Nissenzweig. Sparsification – A technique for speeding up dynamic graph algorithms. In *Proc. 33rd Annual Symp. on Foundations of Computer Science*, 1992.

[11] D. Eppstein, G. F. Italiano, R. Tamassia, R. E. Tarjan, J. Westbrook, and M. Yung. Maintenance of a minimum spanning forest in a dynamic plane graph. *J. Algorithms*, 13:33–54, 1992.

[12] S. Even. *Graph Algorithms.* Computer Science Press, Potomac, MD, 1979.

[13] S. Even and H. Gazit. Updating distances in dynamic graphs. *Methods of Operations Research*, 49:371–387, 1985.

[14] S. Even and Y. Shiloach. An on-line edge deletion problem. *J. Assoc. Comput. Mach.*, 28:1–4, 1981.

[15] S. Even and R. E. Tarjan. Computing an st-numbering. *Theoret. Comput. Sci.*, 2:339–344, 1976.

[16] G. N. Frederickson. Data structures for on-line updating of minimum spanning trees. *SIAM J. Comput.*, 14:781–798, 1985.

[17] G. N. Frederickson. Fast algorithms for shortest paths in planar graphs, with applications. *SIAM J. Comput.*, 16:1004–1022, 1987.

[18] G. N. Frederickson. Ambivalent data structures for dynamic 2-edge-connectivity and k smallest spanning trees. In *Proc. 32nd Annual Symp. on Foundations of Computer Science*, pages 632–641, 1991.

[19] Z. Galil and G. F. Italiano. Fully dynamic algorithms for 2-edge-connectivity. *SIAM J. Comput.* to appear.

[20] Z. Galil and G. F. Italiano. Maintaining the 3-edge-connected components of a graph on-line. *SIAM J. Comput.* to appear.

[21] Z. Galil and G. F. Italiano. Maintaining biconnected components of dynamic planar graphs. In *Proc. 18th Int. Colloquium on Automata, Languages and Programming*, pages 339–350. Lecture Notes in Computer Science 510, Springer-Verlag, Berlin, 1991.

[22] Z. Galil, G. F. Italiano and N. Sarnak. Fully dynamic planarity testing. In *Proc. 24th ACM Symp. on Theory of Computing*, pages 495–506, 1992.

[23] M. T. Goodrich. Planar separators and parallel polygon triangulation. In *Proc. 24th ACM Symp. on Theory of Computing*, pages 507–516, 1992.

[24] L. Guibas and J. Stolfi. Primitives for the manipulation of general subdivision and the computation of Voronoi diagrams. *ACM Trans. on Graphics*, 4:74–123, 1985.

[25] F. Harary. *Graph Theory.* Addison-Wesley, Reading, MA, 1969.

[26] J. Hershberger, M. Rauch, and S. Suri. Fully dynamic 2-connectivity on planar graphs. In *Proc. 3rd Scandinavian Workshop on Algorithm Theory*, pages 233-244. Lecture Notes in Computer Science 621, Springer-Verlag, Berlin, 1992.

[27] J. Hopcroft and R. E. Tarjan. Dividing a graph into triconnected components. *SIAM J. Comput.*, 2:135–158, 1973.

[28] J. Hopcroft and R. E. Tarjan. Efficient planarity testing. *J. Assoc. Comput. Mach.*, 21:549–568, 1974.

[29] T. Ibaraki and N. Katoh. On-line computation of transitive closure for graphs. *Inform. Process. Lett.*, 16:95–97, 1983.

[30] G. F. Italiano. Amortized efficiency of a path retrieval data structure. *Theoret. Comput. Sci.*, 48:273–281, 1986.

[31] G. F. Italiano. Finding paths and deleting edges in directed acyclic graphs. *Inform. Process. Lett.*, 28:5–11, 1988.

[32] A. Kanevsky, R. Tamassia, G. Di Battista, and J. Chen. On-line maintenance of the four-connected components of a graph. In *Proc. 32nd Annual Symp. on Foundations of Computer Science*, pages 793–801, 1991.

[33] P. Klein and J. H. Reif. An efficient algorithm for planarity. *J. Comp. Syst. Sci.*, 37, 1988.

[34] Kuratowski. Sur le problem des courbes gauches en topologie. *Fund. Math.*, 15:271–283, 1930.

[35] J. A. La Poutré. *Dynamic graph algorithms and data structures.* PhD thesis, Department of Computer Science, Utrecht University, September 1991.

[36] J. A. La Poutré. Personal communication, 1992.

[37] J. A. La Poutré and J. van Leeuwen. Maintenance of transitive closure and transitive reduction of graphs. In *Proc. Workshop on Graph-Theoretic Concepts in Computer Science*, pages 106–120. Lecture Notes in Computer Science 314, Springer-Verlag, Berlin, 1988.

[38] A. Lempel, S. Even, and I. Cederbaum. An algorithm for planarity testing of graphs. In *Theory of Graphs: International Symposium*, pages 215–232, 1967.

[39] C. C. Lin and R. C. Chang. On the dynamic shortest path problem. In *Proc. Int. Workshop on Discrete Algorithms and Complexity*, pages 203–212, 1989.

[40] R. J. Lipton and R. E. Tarjan. A separator theorem for planar graphs. *SIAM J. Appl. Math.*, 36:177–189, 1979.

[41] R. J. Lipton and R. E. Tarjan. Applications of a planar separator theorem. *SIAM J. Comput.*, 9:615–627, 1980.

[42] V. Ramachandran and J. H. Reif. An optimal parallel algorithm for graph planarity. In *Proc. 30th Annual Symp. on Foundations of Computer Science*, pages 282–287, 1989.

[43] J. H. Reif. A topological approach to dynamic graph connectivity. *Inform. Process. Lett.*, 25:65–70, 1987.

[44] H. Rohnert. A dynamization of the all pairs least cost path problem. In *Proc. 2nd Annual Symp. on Theoretical Aspects of Computer Science*, pages 279–286. Lecture Notes in Computer Science 182, Springer-Verlag, Berlin, 1985.

[45] D. D. Sleator and R. E. Tarjan. A data structure for dynamic trees. *J. Comp. Syst. Sci.*, 24:362–381, 1983.

[46] R. Tamassia. A dynamic data structure for planar graph embedding. In *Proc. 15th Int. Colloquium on Automata, Languages and Programming*, pages 576–590. Lecture Notes in Computer Science 317, Springer-Verlag, Berlin, 1988.

[47] R. Tamassia and F. P. Preparata. Dynamic maintenance of planar digraphs, with applications. *Algorithmica*, 5:509–527, 1990.

[48] R. E. Tarjan. Depth-first search and linear graph algorithms. *SIAM J. Comput.*, 1:146–160, 1972.

[49] R. E. Tarjan and J. van Leeuwen. Worst-case analysis of set union algorithms. *J. Assoc. Comput. Mach.*, 31:245–281, 1984.

[50] W. T. Tutte. *Connectivity in graphs.* University of Toronto Press, 1966.

[51] W. T. Tutte. *Graph theory.* Cambridge University Press, 1984.

[52] J. Westbrook. Fast incremental planarity testing. In *Proc. 19th Int. Colloquium on Automata, Languages and Programming*, pages 342–353. Lecture Notes in Computer Science 623, Springer-Verlag, Berlin, 1992.

[53] J. Westbrook and R. E. Tarjan. Maintaining bridge-connected and biconnected components on-line. *J. Algorithms*, 7:583–596, 1992.

[54] H. Whitney. Non-separable and planar graphs. *Trans. Amer. Math. Soc.*, 34:339–362, 1930.

[55] D. M. Yellin. A dynamic transitive closure algorithm. Technical Report 13535, IBM Research Division, T. J. Watson Research Center, 1988.