Maintaining the 4-Edge-Connected Components of a Graph On-Line *

E. Dinitz †
Department of Computer Science
Technion
Haifa, ISRAEL 32000

Abstract

Two vertices v and u of an undirected graph are called k-edge-connected if there exist k edge-disjoint paths between v and u. The equivalence classes of this relation are called the k-edge-connected components. We suggest graph structures and an incremental algorithm to maintain k-edge-connected components for the case k = 4. Any sequence of q queries Same-k-Component? and updates Insert-Edge on an n-vertex graph can be performed in O(qo(q, n) + n log n) time, with O(m + n log n) preprocessing (m is the number of edges in the initial graph). Besides, an algorithm for maintaining k-edge-connected components (k arbitrary) in a (k-1)-edge-connected graph is presented. The complexity is O((q + n)a(q, n)), with O(m + k\(^2\)n log(n/k)) preprocessing.

Key words: on-line algorithms, dynamic data structures, graph algorithms, edge-connectivity.

1 Introduction

Connectivity is a fundamental property of graphs, which is used in network reliability analysis and other applications. Much work has been devoted to the development of adequate structures and efficient algorithms for connectivity problems. Recently, dynamic maintenance of connectivity structures has been realized as an important direction of research. This has given a new impetus to the development of corresponding structures and algorithms. In this paper we are concerned with the problem of maintaining the k-edge-connected components of an undirected graph G = (V, E) under inserting edges (a partially dynamic, or incremental setting), and call it the k-problem.

Efficient schemes for 1-, 2- and 3-problem are known [WT92], [GI93]. The first paper relies on the bridge-tree; in the second, the tree-of-cyces structure for modelling the system of 3-edge-connected components of a 2-edge-connected graph has been suggested, and used as the basis. As a matter of fact, the "cactus" structure for k-edge-connected components of a (k-1)-edge-connected graph (k arbitrary), generalizing both of the above structures, had been suggested earlier in [DKL76] 1. Therefore, the k-problem for a (k-1)-edge-connected graph becomes transparent. In Sect. 2, an algorithm is suggested for it with the complexity O((q + n)a(q, n)) for an n-vertex graph undergoing q queries and updates; the O(m + k\(^2\)n log(n/k)) preprocessing is required (a is a functional inverse of Ackermann's function and m is the number of edges in the initial graph; the algorithms in this paper are assumed to be base two). Difficulties arise when the graph connectivity is less than k - 1. In [D92], the relation between the cactus structure for the 4-edge-connected components contained in a 3-edge-connected component S and the cactus structure for the 3-edge-connected components of the 2-edge-connected component containing S has been cleared up. Here we use this relation to solve the 4-problem. Any sequence of q queries Same-k-Component? and updates Insert-Edge can be performed in O(qo(q, n) + n log n) time, with O(m + n log n) preprocessing 2.

Comment. Vertex-connectivity results, though more complicated, have been leading in the literature. The maintenance of triconnected components by means of sophisticated SPQR-trees [BT90] preceded the clear graph model of [GI93]. The paper [KTBC91] gives an insight for understanding the structure of the system of 4-vertex-connected components of a graph, and, using SPQR-trees, suggests an efficient algorithm

---

*This paper has been discovered in the West only in [NGM90].
†Up to 1990: E.A.Dinic, Moscow.

---
for its maintenance. The reduction [GI91] allows, in principle, to obtain similar results in our area (see Theorem 10 in [EGIN92]). But the text of [KTBC91] deals completely with 3-vertex-connected graphs; for the general case, the result is only stated (Theorem 7). So, up to the moment, no algorithm for the general case is known. Besides, a simpler problem is to be solved in a specific, simpler way; in particular, for gaining better insight for \( k \)-problems, \( k > 4 \).

Consider an undirected connected graph \( G = (V, E) \) with \( n \) vertices and \( m \) edges. Multiple edges are allowed. A (simple) edge-cut \( C \) of \( G \) is a set of its edges, whose removal disconnects \( G \), and no proper subset of \( C \) disconnects \( G \). We call \( C \) a cut, and a \( k \)-cut if \( |C| = k \). A 1-cut is also called a bridge. It is well known that the removal of an edge-cut \( C \) results in dividing \( G \) into exactly two connected subgraphs \( G(X_C) \) and \( G(\overline{X_C}) \). Conversely, the partition \( (X_C, \overline{X_C}) \) of \( V \) defines the cut \( C \) uniquely; so we denote a cut also by means of its induced vertex partition-in-two. A graph is called \( l \)-edge-connected if it has no \( l' \)-cuts, \( l' < l \). The edge-connectivity \( \lambda(G) \) is defined as the maximum \( l \) such that \( G \) is \( l \)-edge-connected (equivalently: \( \lambda(G) \) is the minimum number of edges in a cut of \( G \)).

In this paper, we omit usually the word "edge" when we refer to edge-cuts and the edge-connectivity. The equivalence classes of the relation "cannot be separated by a \( k' \)-cut, \( k' < k \)" defined on \( V \) are called \( k \)-edge-connected components (in this paper, \( k \)-components) of \( G \). Clearly, for any \( G \) and \( k \), the set of all \( (k + 1) \)-components is a refinement of the partition of \( V \) into \( k \)-components.

Given a partition of \( V \), the corresponding factor graph is defined to be the result of shrinking each part \( S \), together with all edges joining its vertices, (the subgraph induced by \( S \)) to a single super-vertex (or node). Shrinking all the \( k \)-components provides the natural bijection of the sets of the \( k' \)-cuts and of the \( k' \)-components, \( 1 \leq k' < k \), in the factor graph and the sets of those in \( G \); both edges of cuts and vertex partitions by cuts are preserved. (See [GI93] for a formal proof). So we use interchangeably the notions of a \( k \)-component, the subgraph induced by it in \( G \), and the corresponding node of the factor graph. For example, we speak of a cycle or a path of components.

We are interested in the incremental maintenance of \( k \)-edge-connected components of the graph \( G \), that is, in designing a scheme that allows an arbitrary sequence of operations \( \text{Insert-Edge}(v, u) \) and queries \( \text{Same-k-Component}(v, u) \). Below we rely on several following simple statements. As a result of \( \text{Insert-Edge} \) operation, the connectivity of the whole \( G \), and of each of its subgraphs, cannot decrease. If a new edge connects any two vertices of the same \( k \)-component, its insertion implies no changes in the structure for \( k \)-components. The only possible modification of this structure implied by inserting a new edge is that some subsets of existing \( k \)-components merge into one \( k \)-component, while other \( k \)-components remain the same (because the underlying equivalence relation can only grow or remain the same).

For results known we follow [DKL76], [WT92], [D92], [GI93]. The explanation of the results is done by means of figures and comments on them, relying for the correctness on the techniques of [WT92], [GI93].

2 Case of a \((k - 1)\)-Connected Graph

\( k = 2 \). Let us define the 1|2-structure of a connected graph as the result of shrinking of its 2-components (bridge-components). It is well known to be a tree; let us denote it \( T^1 \) (for illustration, see Fig. 1). Its nodes are the 2-components and its edges are the bridges. Let us assign to a pair of distinct nodes \( N_i \) and \( N_j \) of \( T^1 \) the unique path \( N_i - N_j \) connecting them in \( T^1 \). For any vertex \( w \in V \) we denote by \( N^1_w \) both the \( k \)-component containing \( w \) and the corresponding node of the structure. Upon insertion of a new edge \((v, u)\), all edges of the path \( N^1_w - N^1_v \) stop being bridges. So they, and \((v, u)\), and all the 2-components lying on the path \( N^1_w - N^1_v \) shrink to a single 2-component \( X^2 \) of the 1|2-structure. Prior to the insertion of \((v, u)\), any other bridge divided \( T^1 \) into two parts such that the vertices \( N^1_u \) and \( N^1_v \) lie in the same part. Such a bridge remains to be a bridge, so there are no other changes in \( T^1 \).

As for techniques, the union-find algorithm [TL84] supports finds and unions, and the only task remaining is to find the path connecting two given nodes \( N^1_u \) and \( N^1_v \) of the 1|2-structure of \( G \). Let \( T^1 \) be arbitrarily rooted, and let its edges be directed towards the root. We trace the paths from \( N^1_u \) to the root and from \( N^1_v \) to the root, alternating among them: one edge from
the first, one edge from the second and so on. We stop
when we reach, on either path, a node already marked
as visited at the other path. The number of steps is
not greater than twice the length of the path we look
for. The same techniques apply also for the problems
below.

$k=3$. The $2\beta$-structure $CT^3$ of a 2-connected graph
is the result of shrinking of its 3-components. It is a
tree-of-cycles (for illustration, follow Fig. 2). The latter
is characterized by the property: each of its edges
belongs to some cycle, and no two cycles have an edge
in common. Any 2-cut of the $2\beta$-structure (and also
of $G$) is a pair of edges in the same cycle of $CT^3$. One
can consider an auxiliary tree [GI93]. Its nodes cor-
respond to the nodes of $CT^3$ (square nodes) and to the
cycles of it (round nodes); its edges correspond to the
incidence relation between them. The edges incident
to a round node are ordered cyclically. In what follows
we omit references to this auxiliary tree, and simply
speak of, e.g., "the unique path $N_1-N_2$, of alternating
nodes and cycles, between nodes $N_1$ and $N_2$ in $CT^3$.

Consider the insertion of a new edge $(v, u)$ to $G$.
All the 2-cuts that divide the path $N^3_v-N^3_u$ (assumed
as node/vertex set partitions) get this additional edge
(e.g. $C_1$ in Fig. 2). So they stop being 2-cuts, and all
the 3-components that belong to $N^3_v-N^3_u$ shrink to a
single 3-component (node) $X^3$. Each cut that divides
a cycle not belonging to the path $N^3_v-N^3_u$ remains
unchanged (e.g. $C_2$ in Fig. 2), and the cycle itself as well.
Each 2-cut that divides some cycle in $N^3_v-N^3_u$ but does
not separate the two nodes of $N^3_v-N^3_u$ lying on this
cycle also remains to be a 2-cut (e.g. $C_3$ in Fig. 2). Each
cycle in $N^3_v-N^3_u$ is "squeezed": it breaks in two halves,
and each half containing more than one edge forms a
new cycle. A single-edge half-cycle does not form a
cycle, but is absorbed by $X^3$. The squeezed node $X^3$
is the articulation point of all the new cycles. We call
such an operation the squeezing along the path, and
$X^3$ the squeezed component.

$k=4$. Starting from the case $k=4$, one fails to
describe relevant structures by factor graphs. So we
define another type of structures. We call a pair of
partitions-in-two of a set $V$ parallel if they divide $V$
together into three non-empty parts. An arbitrary
family $\mathcal{P}$ of mutually parallel partitions-in-two of $V$
is represented by its structure tree $T(\mathcal{P})$ (for illustration,
see Fig. 3). Each maximal subset of elements of $V$
lying on the same side with respect to every cut of
$\mathcal{P}$ corresponds to a certain node of $T(\mathcal{P})$. Some
of its non-terminal nodes correspond to the empty set
(empty nodes). $|\mathcal{P}|$ edges of $T(\mathcal{P})$, regarded as 1-cuts
of it, generate all the partitions in $\mathcal{P}$. If a partition
is eliminated from $\mathcal{P}$, the corresponding edge in $T(\mathcal{P})$
shrinks and its ends merge.

According to Crossing Lemma [B75], [DKL76], if any
two minimum $l$-cuts of $G (l = \lambda(G))$ divide $V$ into
four (non-empty) parts, then their shrinking forms a 4-
node super-cycle with $l/2$ parallel edges of $G$ between
every pair of neighbouring nodes. For $l = 3$ it is im-
possible, so any two 3-cuts of a 3-connected graph $G$
are parallel. The structure tree of the family of 3-cuts
of $G$ is called the $3\beta$-structure $T^3$ of $G$ (for illustration,
follow Fig. 4). Its non-empty nodes correspond exactly
to all the 4-components of the graph $G$. Recall
that edges of $T^3$ (structure edges) are not edges of $G$.

One may imagine an edge $(v, u)$ of $G$ going along the
path $N^4_v-N^4_u$ in $T^3$ as a thin or dashed line. There
are exactly three such edge-paths going along each edge
$e$ of $T^3$; the corresponding edges of $G$ form the 3-cut
$C, e \in C$, represented by this structure edge. When
we insert a new edge $(v, u)$ into the graph $G$, all the
edges of the path $N^4_v-N^4_u$ in $T^3$, and only they, shrink.
This is because all of the corresponding partitions of $V$,
and only they, get the additional edge $(v, u)$. So
the problem of maintaining $T^3$ is algorithmically the
same as for $T^1$.

$k=4$. The paper [DKL76] suggests the $(k-1)|k-
structure implied by the system of $(k-1)$-cuts of a
graph $\lambda(G) = k-1$, which generalizes $T^1, CT^3$ and
$T^3$ to the case of arbitrary $k$. Its non-empty nodes corre-


3 General Case
One can see that any $l$-cut, $l \geq 2$, divides only one
2-component and contains only its inner edges. So
Figure 2: Modification of the 2|3-structure; the auxiliary tree

Figure 3: The structure tree for a system of mutually parallel partitions and its modification under elimination of some partitions

Figure 4: Modification of the 3|4-structure
for \( k \geq 3 \), the structures for \( k \)-components may be assumed separately for each 2-component of \( G \). Every 2-component is 2-connected, so one can rely on its 2\( \beta \)-structure.

Recall that when a new edge generates a cycle in a bridge-tree, all the 2-components lying on it merge. The main structural statement of [GI93] is that (i) each of these 2-components is squeezed (see Sect. 2), exactly as if the new edge connecting the two attachment points of this component to the new cycle has been inserted (for illustration, follow Fig. 6), (ii) the new cycle connects the modified 2-components in the joint 2-component, and its nodes are their squeezed nodes, and (iii) these are all the changes in the 2\( \beta \)-structures of 2-components of \( G \) implied by the edge insertion.

Now we pass to the structure for the 4-components of \( G \). It is interesting that all the changes in it, under inserting a new edge, are the same as if (i)-operations were done separately for each 2-component involved, as before. Moreover, we present a method of decomposing this structure (and its transformations) into parts corresponding to 3-components of \( G \).

Closed 3-components. Decomposition into 2-components. The main concept of [D92] is the closure of a 3-component \( S \) of \( G \). To define it, we start with the induced subgraph \( G(S) \). Assume a cycle of the 2\( \beta \)-structure incident to \( S \) with distinct attachment vertices \( v_1 \) and \( v_2 \) in \( S \) (for illustration, follow Fig. 7). It implies a new special edge \((v_1, v_2)\); a pointer to the original cycle is attached to this edge. All such edges are added to \( G(S) \), and the result is called the closed 3-component \( \overline{S} \). Its properties are the following: (i) it is 3-connected; (ii) the sets of all distinct partitions of \( S \) by 3-cuts of \( \overline{S} \) and by 3-cuts of \( G \) coincide; (iii) every 3-cut of \( G \) divides exactly one 3-component, and the related partition corresponds to a 3-cut of its closure; (iv) to each 3-cut \( C \) of \( \overline{S} \) there corresponds a "bunch" of all 3-cuts of \( G \) dividing \( S \) in the same way; a cut \( C \) of the bunch is obtained by changing each special edge in \( C \) by an edge of the cycle pointed from it. According to property (ii), the 4-components of \( \overline{S} \) are exactly the 4-components of \( G \) included in \( S \). Hence the 3\( |4 \)-structures of all 3-components of \( G \) give all the 4-components of \( G \). In what follows, we eventually speak of the 3\( |4 \)-structure of \( S \) instead of the 3\( |4 \)-structure of \( \overline{S} \).

When we analyze the squeezing operation and track the closed 3-components, we see that the new cycle of the 2\( \beta \)-structure (see Fig. 6) implies special edges, one for each squeezed 3-component. The resulting transformation of the 2-components would be the same if these special edges were added a priori to each involved 2-component separately (since all the squeezings are the same). So we obtain the reduction of the 4-problem to the case of a 2-connected graph, namely, to the 2-components of the graph \( G \). Henceforth, the graph \( G \) is assumed to be 2-connected.

Decomposition of a squeezed 2-component. The insertion of a new edge \((v, u)\) to a 2-connected graph \( G \) does not change its 3-components, except for the components that merge (henceforth, involved). Besides, there are no changes in the set of special edges for the components that remain the same. In fact, all the cycles incident to such a component \( S \) in the 2\( \beta \)-structure either remain the same, or are reduced to smaller ones with the same attachment vertices (for example, see node \( S \) in Fig. 2). So we have to analyze only the transformation of the 3\( |4 \)-structures of the involved 3-components to the 3\( |4 \)-structure of the united squeezed 3-component \( \overline{S} \).

The new edge \((v, u)\) transforms only \((v, u)\)-separating cuts: (i) former 2-cuts turn into new 3-cuts, and appear in \( \overline{T}^3(\overline{X}^3) \); (ii) former 3-cuts turn into 4-cuts, and disappear from \( \overline{T}^3(\overline{X}^3) \); other 3-cuts remain the same (for illustration, follow Fig. 8). Let \( Y \) be an involved 3-component, and \( v \not\in Y \). There is a cycle that is incident to \( Y \) and lies on the path from \( Y \) to \( X^3 \). Let us break the special edge \( e_v \) of \( \overline{Y} \), correspond-
ing to this cycle, by a new link node $N^Y_v$. If $v \in Y$, let $N^Y_v = N^Y_u$. The node $N^Y_v$ is defined similarly. One can see that for each 3-cut $C$ of $G$ that divides $Y$ and separates $v$ from $u$, the corresponding cut of the modified $\overline{Y}$ separates now $N^Y_v$ from $N^Y_u$, and vice versa. We add the edge $(N^Y_v, N^Y_u)$ to $\overline{Y}$ and call the resulting object the extended closed 3-component $\overline{Y}'$. The claim is that the 3|4-structure of $\overline{Y}'$ contains exactly the same 3-cuts that remain in $X^3$, and also the link cuts that isolate $N^Y_v$, $N^Y_u$, if they are link nodes.

**Transformations: global view.** When we merge two 3-components $Y$ and $Y_1$ belonging to some cycle, their closures are "glued together". The special edges of $\overline{Y}$ and $\overline{Y_1}$, pointing to this cycle, disappear. Let $(v_1, v_2)$ and $(v_3, v_4)$ be these special edges (indices are ordered along the cycle) (see Fig. 8). Then the new edges $(v_2, v_3)$ and $(v_4, v_1)$, ordinary or special (corresponding to the halves of the cycle), appear in the squeezed 3-component $X^3$. This pair of edges, together with $(v, u)$, forms a 3-cut in the closed squeezed 3-component $\overline{X^3}$. (This cut correspond to the bunch of the former 2-cuts of $G$, separating $Y$ and $Y_1$). The chain of such new 3-cuts corresponds to the chain of cycles that constituted previously the path between $N^3_u$ and $N^3_v$. These cuts divide the vertex set of $X^3$ into vertex sets of involved 3-components. Observe that these are the only $(v, u)$-separating 3-cuts in $\overline{X^3}$. So we already know some path $\pi(v, u)$ in the structure tree $T^3(\overline{X^3})$.

Let us analyze which decomposition of a family $P$ of mutually parallel partitions-in-two of $V$ is defined by its element. The decomposition of $T(P)$ generated by a partition $(V_1, V_2, V_3)$ is obtained by shrinking $V_1$ and, independently, $V_2$ into new link nodes $\bar{V}_1$ and $\bar{V}_2$, respectively (see Fig. 9). In the two resulted structures $T(P_1)$ and $T(P_2)$, each partition belonging to $P$ is present once (in a modified form), except for $(\bar{V}_1, V_2)$, which is presented twice, as $\bar{V}_1$ and $\bar{V}_2$-isolating partitions. These partitions, and the corresponding edges of the structure trees, are called link partitions and edges. The composition operation glues together $T(P_1)$ and $T(P_2)$ by eliminating the link nodes and identifying the link edges. The decomposition generated by a subfamily is defined as the result of the decompositions generated by all its elements (irrelevant of their order). One can see that the decomposition of the structure tree $T^3(\overline{X^3})$ generated by the edges of $\pi(v, u)$ gives us exactly the extended closed involved 3-components. Thus, to construct $T^3(\overline{X^3})$ we can transform separately the 3|4-structures of the involved 3-components to that of the extended components, and compose the results (see Fig. 8).

**Local view.** Now we pass to the transformation of the structure tree $T^3(\overline{X^3})$ into $T^3(\overline{Y'})$, where $Y$ is an involved 3-component. Henceforth we omit superscripts "4" and "Y" in our notation. There are three kinds of transformations (see Fig. 10). If both $v$ and $u$ belong to $Y$ (i.e. $X^3 = Y \cup (v, u)$), then the edge $(N_v, N_u)$ is added to the 3-connected graph $\overline{Y}$ (the $I$-operation). We have discussed the corresponding transformation in Sect. 2. If $v \notin Y$, the link node $N_v$ breaks a special edge $e_v = (L'_v, L''_v)$ of $\overline{Y}$ into two edges $e'_v = (L'_v, N_v)$ and $e''_v = (N_v, L''_v)$, special or ordinary. The case $u \notin Y$ is treated similarly. The transformation implied by the insertion of an edge between a link and an ordinary node is called the $T$-operation, and of an edge between two link nodes is called the $H$-operation.

**T-operation.** For a T-operation, $v \in Y$ (see Fig. 11), the relevant objects in the tree $T^3(\overline{Y})$ are its node $N_v$, the edge-path $P(e_v)$ corresponding to $e_v = (L'_v, L''_v)$, and the path $P$ in $T^3(\overline{Y})$ connecting $N_v$ with $P(e_v)$. We claim that the modification of

![Figure 6: Joint modification of the 1|2- and 2|3-structures](image)

![Figure 10: Modification of a closed 3-component: I-, T- and H-operations](image)
Figure 8: Decomposition of the 3|-4-structure transformation

Figure 9: Decomposition of a parallel partition family

\( T^3(Y) \) consists in shrinking the path \( P \) to the node \( X^4 \) and adding the link node \( N_u \) with the link edge \((N_u, X^4)\). Indeed, an edge \( e_1 \in P \) separates \( N_v \) from both \( L'_u \) and \( L''_u \), so it separates \( N_v \) from \( N_u \) and has to disappear. An edge \( e_2 \in \{P(e_u)\} \) separates \( L'_u \) from \( L''_u \). Assume that \( N_u \) is separated from \( L'_u \) (for \( L''_u \), everything is similar). The change of \( e_u \) by \( e'_u \) in the corresponding 3-cut results in a 3-cut not separating \( N_v \) from \( N_u \), so the edge \( e_3 \) remains in \( T^3(Y) \). Any other edge \( e_3 \) of \( T^3(Y) \) does not divide the set of the three nodes \( N_v, L'_u, \) and \( L''_u \). So the corresponding cut does not separate \( N_v \) and \( N_u \), and \( e_3 \) remains in the \( T^3(Y) \). Every remaining cut does not separate the nodes \( X^4 \) and \( N_u \). So the 3-cut \((e_u, u', e'_u, e''_u)\) that isolates the node \( N_u \) must correspond to the edge \((N_u, X^4)\) in the \( 3\-4\)-structure \( T^3(Y) \). The case \( u \in Y \) is treated similarly.

**H-operation.** For an H-operation (see Fig. 12), the relevant objects in the tree \( T^3(Y) \) are the nodes \( N_v \) and \( N_u \), the edge-paths \( P(e_u) \) and \( P(e_v) \), and the path \( P \) in \( T^3(Y) \) connecting \( P(e_u) \) and \( P(e_v) \). If these edge-paths have no common edge, then the modification of \( T^3(Y) \) consists in shrinking the path \( P \) to the node \( X^4 \) and adding the link nodes \( N_u \) and \( N_u \) with the link edges \((N_u, X^4)\) and \((N_u, X^4)\). The proof is similar to that for the T-operation.

Let us consider the finding of the path \( P \) for an H-operation (for a T-operation the process is similar). We trace "in parallel" the four paths to the root: from the nodes \( L'_u, L''_u, L'_v, \) and \( L''_v \) and mark each traversed node by "v" or "u", respectively. We stop tracing when some node receives the both marks. At this moment we can build a path \( \hat{P} \) that includes \( P \) (using the pointers to the predecessors related to marks).

Every edge belonging to \( \hat{P} \) belongs to \( P(e_u) \), or to \( P(e_v) \). These edges are easy to distinguish, if we maintain for every edge of a 3|-4-structure the list-of-the-cut that contains the three original edges of the related 3-cut. For an edge in \( \hat{P} \), \( e_u \), or \( e_v \) appears in its list-of-the-cut.

The list-of-the-cut for the edge that is common to \( P(e_u) \) and \( P(e_v) \) contains both \( e_u \) and \( e_v \). One can prove that there cannot be more than one common edge for any two edge-paths \( P(e_1) \) and \( P(e_2) \) in a 3|-4-structure \( T^3 \). Suppose to the contrary that edges \((N_1, N_2)\) and \((N_2, N_3)\) belong to both \( P(e_1) \) and \( P(e_2) \) (see Fig. 13). Let \((U_1, U_2, U_3)\), \( N_i \in U_i \), be the partition-in-three of the tree \( T^3 \) generated by these two edges. Consider the cut \( C \) corresponding

---

\(^3\)The fact that \( e_u \) is changed only by \( e'_u \), and not by \( e''_u \), means that some of the cuts of \( G \) in the \( e_2 \)-bunch remain to be 3-cut, while other do not (the \( e_2 \)-bunch decreases).
to the partition \((U_2, U_1 \cup U_3)\). The edges of \(C\) can be only the edges of 3-cuts corresponding to \((N_1, N_2)\) and \((N_2, N_3)\), except for \(e_1\) and \(e_2\). So there are no more than two edges in this cut, a contradiction: the graph is not 3-connected.

Assume an H-operation when \(P(e_u) \cap P(e_v) = \emptyset\) (see Fig. 14). We claim that the modification of \(T^3(Y)\) consists in inserting a new empty node \(X^4\) that breaks the edge \(e\) into two edges \(e'\) and \(e''\), and adding the link nodes \(N_e\) and \(N_u\) with the link edges \((N_e, X^4)\) and \((N_u, X^4)\). The proof goes along the same lines, except for the analysis of the situation with the edge \(e\). Assume that \(e\) separates \(L_u'\) and \(L_v'\) from \(L_u''\) and \(L_v''\), and \(e'\) lies on the path from \(L_v''\) to \(X^4\) (this does not reduce the generality). The 3-cut corresponding to \(e'\) is obtained from the 3-cut corresponding to \(e\) by changing the edges \(e_u\) and \(e_v\) by the edges \(e_u'\) and \(e_v'\); similarly for \(e''\). These two 3-cuts separate the nodes \(X^4, N_v\) and \(N_u\) from all the other nodes in \(T^3(Y)\). So the link edges incident to \(N_v\) and \(N_u\) must be incident just to \(X^4\).

4 Complexity for the General Case

In [GI93], an algorithm for the 3-problem with complexity \(O((q + n)\alpha(q, n))\) is suggested, where \(q\) is the number of queries \(\text{Same}-k-Component}(v, u)\) and updates \(\text{Insert-Edge}(v, u)\). The space required is \(O(n)\). To solve the 4-problem, we use here the same tools, and hence only three operations are beyond the techniques of [GI93]: the preliminary construction of 3|4-structures of 3-components, the re-rooting of 3|4-structures of 3-components after merging, and the finding the path \(P\) for a T- or H-operation. An operation of the first type can be performed with complexity \(O(m + n \log n)\) [GI91].

An operation of the second type, if one takes for the new root the root of the larger tree, implies the re-orientation of edges in the path from the root of the smaller tree to the link edge that connects the trees. We define the weight of the structure tree of a 3-component to be the cardinality of this component. Hence the weight of a structure tree never changes, except for tree merging, and does not exceed \(n\). When an edge is re-oriented, the weight of its tree is at least doubled. So no edge is re-oriented more than \(\log n\) times. New edges arise only in squeezing operations (see Sect. 2). Each squeezing increases the number of structure edges by the same value as it decreases the number of 3-components. The initial number of structure edges for \(G\) is \(O(n)\) [D92], while that of
3-components does not exceed \( n \). So the number of structure edges in the whole process is \( O(n) \), and the complexity of all the operations of the second type is \( O(n \log n) \).

Let us give a bound on the total extra complexity of the search for paths of type \( P \) in \( T \) - and \( H \) -operations for a sequence of queries. Consider the search for the path \( P \) for an \( H \) -operation (for a \( T \) -operation the process is similar). Let \( P = \{ M_v, M_u \} \), where \( M_v \in P(\epsilon_0) \), \( M_u \in P(\epsilon_2) \) (for illustration see Fig. 12).

The node \( M_v(\epsilon_1) \) divides the path \( P(\epsilon_1(\epsilon_1)) \) into two parts. Let us denote the length of the shorter (or equal) part by \( l(\epsilon_0, \epsilon_1) = \min \{ |L_{\epsilon_0} - M_v(\epsilon_1)|, |L_{\epsilon_1} - M_v(\epsilon_1)| \} \). To be definite, assume \( l(\epsilon_0, \epsilon_1) = |L_{\epsilon_0} - M_v(\epsilon_1)| \). Consider a while a weaker trace process that begins only from \( L'' \) and \( T'' \) and executes only two steps "in parallel". It is known that the number of its double steps cannot be greater than the length of the path between \( L'' \) and \( T'' \), i.e. \( l(\epsilon_0,\epsilon_1) + |P| + l(\epsilon_2) \). So the same bound stands for our stronger process. We have to count only the number of trace steps when no edge, of the four related, belongs to \( P \) (all steps related to the tracing of \( P \) can be counted by the technique used in [GI93]). Thus it is sufficient to estimate the sum of the values \( l(\epsilon_0,\epsilon_1) = \max \{ |L_{\epsilon_0} - \epsilon_2| - 2.0 \} \) and \( l(\epsilon_2) = \max \{ |L_{\epsilon_2} - \epsilon_2| - 2.0 \} \) for all \( H \) - and \( T \) -operations throughout the algorithm (henceforth, \( SUM \)).

Let \( \{ \epsilon_i, i \in I \} \) be the set of all edges of closed 3-components that enter more than two 3-cuts in their components (i.e. \( |P(\epsilon_i) - 3| \), \( n_t = |P(\epsilon_i)| - 2 \). Let \( n_t \) denote the (current) number of 3-components. The potential function \( \Phi \) is defined as

\[
\Phi = \sum_{i \in I} n_t \log n_t + n_0 \log n_0.
\]

Assume that a special edge \( \epsilon_i \) is broken in an \( H \) - or \( T \) -operation, and \( l(\epsilon_i) > 0 \). The latter condition implies that \( \epsilon \) is broken to edges \( \epsilon_i^! \) and \( \epsilon_i^" \), where \( \epsilon_i^! P^! \) are added to \( P \) in the place of \( \epsilon_i \).

\[
|P(\epsilon_i^!)| + |P(\epsilon_i^")| = |P(\epsilon_i)| + 2 \Rightarrow
\Rightarrow |P(\epsilon_i^!)| + (|P(\epsilon_i^")| - 2) = |P(\epsilon_i)| - 2 \Rightarrow
\Rightarrow n_t = n_t^! + n_t^" \Rightarrow l(\epsilon_i) = n_i^" / 2.
\]

The decrease of the value of the potential function caused by this break is

\[
\Phi - \Phi^* = n_t \log n_t - (n_t^! \log n_t^! + n_t^" \log n_t^") =
= n_t \log (n_t^" / n_t^!) + n_t^" \log (n_t^" / n_t^!) \geq n_t \log 2 =
= n_t \log (n_t^" / n_t^!).
\]

Thus \( SUM \) is bounded by the sum of all decreases of \( \Phi \) throughout the algorithm.

Other operations of the algorithm do not increase the set \( I \) and the values \( n_i \) and \( n_0 \), as a rule; so \( \Phi \) is not increased too. Consider the only exception: a squeezing operation that generates new special edges (see Fig. 8). When some two involved components glue together, two special edges \( \epsilon_i \) and \( \epsilon_j \) disappear, with \( n_i = n_j \) (if defined) equal to the length of the cycle incident to both components minus two. At most, there appear two new edges \( \epsilon_i \) and \( \epsilon_j \), with \( n_i \) and \( n_j \) equal to the lengths of some smaller cycles minus two. This implies a decrease of \( \Phi \). Further, assume that there are \( r \) involved 3-components. Then the new edge \( (v, u) \) (that appears in the closed squeezed component) takes part in a new 3-cut. The resulting increase of \( \Phi \) is equal to \( (r - 3) \log (r - 3) \), at most (if \( r > 4 \)).

At the same time, \( n_t \) decreases by \( r - 1 \). The resulting decrease of \( \Phi \) is not smaller than \( (r - 1) \log n_t \), which is greater than the maximum increase of it due to the addition of \( (v, u) \). Thus a squeezing operation decreases \( \Phi \) too.

So the the sum of all decreases of \( \Phi \) is bounded by the difference of its values at the beginning and at the end of the sequence of queries. The latter is non-negative. The former is not greater than the value of \( \Phi \) for a spanning tree. In fact, every graph may be built incrementally beginning from a spanning tree, and \( \Phi \) can only decrease. Therefore the whole number of extra trace steps is \( O(n \log n) \). The declared complexity
$O((q + n)\alpha(q, n) + n \log n) = O(q\alpha(q, n) + n \log n)$ follows.

5 Concluding Remarks

1. The generalization to the case of arbitrary (possibly non-connected) graphs is straightforward. The only problem arises from the need of re-rooting of bridge-trees under their merging; this can worsen the complexity for $k$-problems, $k \leq 3$. But for $4$-problem such operations are used in the algorithm, so the generalization is smooth.

2. A.Vainshtein and the author have developed recently a general approach to the $k$-problem, $k$ arbitrary. D.Naor in [N91] has suggested the following hierarchical cactus representation (HCR) of the $k$-edge-connected components of a graph for all $k$ \(^4\). For any $U \subseteq V$ we call a cut of the whole $G$ $U$-minimal if it divides $U$ and has the minimum value among all such cuts. The observation is that for any $U$ the system of all $U$-minimum cuts has a cactus representation. The cuts of the $U$-cactus correspond to the vertex partitions of $U$ by $U$-minimal cuts. So one can build the cactus of $G$, whose nodes correspond to the $(\lambda(G) + 1)$-components, the cactus representation for each of these components, etc. However, it turned out that a single cut of the $U$-cactus corresponds, in general, to many cuts of $G$, which have the same value and separate $U$ in the same way. This property makes the problem of dynamic maintenance of HCR($G$) rather complicated. The relation between the $2|\beta|$- and $3|\beta|$-structures of a graph (see Sect. 3) may serve as a weak illustration for difficulties arising for general $k > \lambda(G)$. Up to the moment, we have developed some detailed static description of HCR($G$) and discovered some of its incremental properties. Seemingly, this can serve as a basic tool for incremental algorithms for the $k$-problem.

Acknowledgments

The author is very grateful to A.Vainshtein for his great help in improving the presentation of the paper.

References

[B75] Bixby, R.E. The minimum number of edges and vertices in a graph with edge connectivity $n$ and $m$ $n$-bonds. Networks 5 (1975), 253–298


