SIAM J. COMPUT.
Vol. 42, No. 3, pp. 1185–1200




# EFFICIENT EDGE SPLITTING-OFF ALGORITHMS MAINTAINING ALL-PAIRS EDGE-CONNECTIVITIES*

LAP CHI LAU† AND CHUN KONG YUNG‡

**Abstract.** We present new edge splitting-off results maintaining all-pairs edge-connectivities of an undirected graph. We first give an alternate proof of Mader's theorem, and use it to obtain a deterministic $\tilde{O}(m + {r_{\max}}^2 \cdot n^2)$-time complete edge splitting-off algorithm for unweighted graphs, where $r_{\max}$ denotes the maximum edge-connectivity requirement. This improves upon the best known algorithm by Gabow by a factor of $\tilde{\Omega}(n)$. We then prove a new structural property, and use it to further speed up the algorithm to obtain a randomized $\tilde{O}(m + {r_{\max}}^3 \cdot n)$-time algorithm. These edge splitting-off algorithms can be used directly to speed up various graph algorithms.

**Key words.** graph algorithms, edge connectivity, splitting off

**AMS subject classifications.** 05C85, 68R10

**DOI.** 10.1137/100790239

## 1. Introduction.

The edge splitting-off operation plays an important role in many basic graph problems, in both proving theorems and obtaining efficient algorithms. Splitting-off a pair of edges $(xu, xv)$ means deleting these two edges and adding a new edge $uv$ if $u \neq v$. This operation was introduced by Lovász, who showed that splitting-off can be performed to maintain the *global edge-connectivity* of a graph.

THEOREM 1.1 (Lovász [18, 19]). *Let $G = (V, E)$ be an undirected graph that has at least $k \geq 2$ edge-disjoint paths between $s$ and $t$ for all $s, t \in V - \{x\}$. If the degree of $x$ is even, then some edge pair $(xu, xv)$ can be split off so that in the resulting graph there are still at least $k$ edge-disjoint paths between $s$ and $t$ for all $s, t \in V - \{x\}$.*

Mader extended Lovász's result significantly to prove that splitting-off can be performed to maintain the *local edge-connectivity* for all pairs.

THEOREM 1.2 (Mader [20]). *Let $G = (V, E)$ be an undirected graph that has at least $r(s, t)$ edge-disjoint paths between $s$ and $t$ for all $s, t \in V - \{x\}$, and has at least two edge-disjoint paths between $x$ and each of its neighbors. If the degree of $x$ is not equal to 3, then some edge pair $(xu, xv)$ can be split off so that in the resulting graph there are still at least $r(s, t)$ edge-disjoint paths between $s$ and $t$ for all $s, t \in V - \{x\}$.*

These splitting-off theorems have applications in various graph problems. Lovász [18] and Mader [20] used their splitting-off theorems to derive Nash-Williams' graph orientation theorems [24]. Subsequently, these theorems and their extensions have found applications in a number of problems, including edge-connectivity augmentation problems [9, 7, 3], network design problems [13, 16, 6], tree packing problems [1, 17, 5], and graph orientation problems [11].

Efficient splitting-off algorithms have been developed to give fast algorithms for the above problems [12, 23, 3, 21, 5]. However, most of the efficient algorithms have

*Received by the editors March 29, 2010; accepted for publication (in revised form) April 10, 2013; published electronically June 20, 2013. A preliminary version appeared in Proceedings of IPCO 2010. The research is supported by HK RGC grant 413411.
http://www.siam.org/journals/sicomp/42-3/79023.html

†Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, China (chi@cse.cuhk.edu.hk).

‡Department of Computer Science, University of Toronto, Toronto, Canada (ckyung@cs.toronto.edu).

been developed only in the global edge-connectivity setting, even though there are important applications in the more general local edge-connectivity setting.

In this paper, we present new edge splitting-off results maintaining all-pairs edge-connectivities. First we give an alternate proof of Mader's theorem (Theorem 1.2). Based on this, we develop a faster deterministic algorithm for edge splitting-off maintaining all-pairs edge-connectivities (Theorem 1.3). Then, we prove a new structural property (Theorem 1.4), and use it to design a randomized procedure to further speed up the splitting-off algorithm (Theorem 1.3). These algorithms improve the best known algorithm [12] by a factor of $\tilde{\Omega}(n)$, and can be applied directly to speed up various graph algorithms that use edge splitting-off.

In the following, we consider an undirected graph $G = (V, E)$. We often write a single vertex set $\{x\}$ as $x$. The degree of a vertex set $S \subset V$ is denoted by $d(S)$. The neighbor set of a vertex $v$ is denoted by $N(v)$, and a vertex in $N(v)$ is called a $v$-neighbor. Denote by $r(s,t)$ the edge-connectivity requirement between $s, t \in V$, and by $r_{\max} = \max_{s,t \in V-x} r(s,t)$ the maximum edge-connectivity requirement. The connectivity requirement is *global* if $r(s,t) = k$ for all $s, t \in V$; otherwise it is *local*. An edge $uv \in E$ is a cut edge if the edge-connectivity between $u$ and $v$ is 1.

**1.1. Efficient complete edge splitting-off algorithm.** Mader's theorem can be applied repeatedly until $d(x) = 0$, when $d(x)$ is even and there is no cut edge incident on $x$. This is called a *complete edge splitting-off* at $x$ (or a vertex splitting-off at $x$), which is a key subroutine in algorithms for connectivity augmentation, graph orientation, and tree packing.

A straightforward algorithm to compute a complete splitting-off sequence is to attempt splitting-off $(xu, xv)$ for every pair $u, v \in N(x)$, and then checking whether the connectivity requirements are violated by computing all-pairs edge-connectivities in the resulting graph, and repeating this procedure until $d(x) = 0$.

Several efficient algorithms have been proposed for the complete splitting-off problem, but only Gabow's algorithm [12] can be used in the local edge-connectivity setting with running time $O({r_{\max}}^2 \cdot n^3)$. Our algorithms improve the running time of Gabow's algorithm by a factor of $\tilde{\Omega}(n)$. In applications where $r_{\max}$ is small, the improvement of the randomized algorithm could be a factor of $\tilde{\Omega}(n^2)$.

THEOREM 1.3. *In the local edge-connectivity setting, there is a deterministic $\tilde{O}(m + {r_{\max}}^2 \cdot n^2)$-time algorithm and a randomized $\tilde{O}(m + {r_{\max}}^3 \cdot n)$-time algorithm for the complete edge splitting-off problem in unweighted graphs.*

**1.2. Techniques.** Mader's theorem shows the existence of one *admissible* edge pair, whose splitting-off maintains the local edge-connectivity requirements of the graph. Given an edge $xv$, we say an edge $xw$ is a *non-admissible partner* of $xv$ if $(xv, xw)$ is not admissible. Our proof of Mader's theorem is based on the notion of a *non-admissible set*: a subset $U \subseteq N(x)$ is called a non-admissible set if $(xu, xv)$ is not admissible for every $u, v \in U$. Using the 3-dangerous set structure in [6], we prove in Lemma 3.1 that under the conditions of Mader's theorem, every non-admissible set is contained in some proper vertex subset $S \subset V - x$ with $d(S) \leq r_{\max} + 1$. A simple argument then shows that a small degree vertex subset cannot contain all $x$-neighbors. Hence, there must exist an admissible edge pair, proving Mader's theorem.

Our edge splitting-off algorithms are conceptually very simple, which can be seen as refinements of the straightforward algorithm. There are two main steps in the improvement. The first is to reduce the number of splitting-off attempts, and the second is to check the edge-connectivities more efficiently after each splitting-off attempt. In section 3.2, we show how to find a complete edge splitting-off sequence by

using at most $O(|N(x)|)$ splitting-off attempts, instead of $O(|N(x)|^2)$ attempts by the straightforward algorithm. This is based on the notion of a non-admissible set and the alternative proof of Mader's theorem in section 3.1.

In section 3.4, we show how to reduce the problem of checking local edge-connectivities for all pairs to the problem of checking edge-connectivities from a particular *indicator vertex*. This allows us to check at most $O(n)$ pairs to determine whether the edge-connectivities are preserved, instead of checking $O(n^2)$ pairs which requires $\Omega(n^2)$ time, which would be too slow. To guarantee the existence of an indicator vertex, we preprocess the graph by contracting appropriate vertex sets, and show that it will be an indicator vertex throughout the algorithm. Using the sparsifying algorithm by Nagamochi and Ibaraki [22, Theorem 2.6] and the fast Gomory–Hu tree algorithm by Bhalgat et al. [4] (Theorem 2.6), we can obtain a compact representation of the local edge-connectivity information in $\tilde{O}(r_{\max}^2 \cdot n)$ time. Then, it only takes $O(n)$ time to check the local edge-connectivities from an indicator vertex. Hence we can efficiently determine whether the local edge-connectivity requirements are satisfied in $\tilde{O}(r_{\max}^2 \cdot n)$ time. This gives us the deterministic splitting-off algorithm in Theorem 1.3.

Mader's theorem shows the existence of one admissible pair on a vertex $x$. We strengthen his result by proving a tight upper bound on the number of non-admissible partners of a given edge $xv$ which may be of independent interest.

THEOREM 1.4. *Suppose there is no cut edge incident on $x$ and $r_{\max} \geq 2$. Then the number of non-admissible partners for any given edge $xv$ is at most $2r_{\max} - 2$.*

This improves the result of Bang-Jensen and Jordán [2] by a factor of $r_{\max}$, and the bound is the best possible as there are examples for achieving it (see Figure 4.2). The proof of Theorem 1.4 is based on a new inductive argument and will be presented in section 4. Theorem 1.4 implies that when $d(x)$ is considerably larger than $r_{\max}$, most of the edge pairs incident on $x$ are admissible. Therefore, we can split off many edge pairs randomly in parallel before checking the edge-connectivities once. This gives us the randomized splitting-off algorithm in Theorem 1.3.

**1.3. Applications.** Our edge splitting-off algorithms can be used directly to improve the running time of various graph algorithms [24, 9, 13, 12, 17, 6]. We will take the edge-connectivity augmentation problem as an example to illustrate this. Consider an unweighted graph $G = (V, E)$ and edge-connectivity requirements $r(s, t)$ for all $s, t \in V$. The edge-connectivity augmentation problem asks for a smallest set $F$ of new edges such that for all $s, t \in V$, there are $r(s, t)$ edge disjoint paths between $s$ and $t$ in $G' = (V, E + F)$. Frank [9] gave a simple algorithm by using edge splitting-off. There are two main steps in the algorithm: (i) first add an external vertex $x$ and a minimum set of edges incident to $x$ so that edge-connectivity requirements are satisfied for all $s, t \in V$ in $G'$; (ii) then perform a complete edge splitting-off at $x$ to obtain $G' = (V, E + F)$. The best known implementation of Frank's algorithm is by Gabow [12], which requires $O({r_{\max}}^2 n^3)$ time. Using our edge splitting-off algorithm, the second step takes only $\tilde{O}(m + {r_{\max}}^2 n^2)$ time. The first step is not more difficult than an edge splitting-off problem. A minimal set of edges can be found by first adding excessive edges and then removing the redundant ones. It is similar to a preprocessing step of our algorithm, which removes as many pairs of $(xu, xu)$ as possible for every $u \in V - x$. Therefore, the total complexity can be improved from $O({r_{\max}}^2 n^3)$ to $\tilde{O}(m + {r_{\max}}^2 n^2)$.

**2. Preliminaries.** Let $G = (V, E)$ be a graph. For $X, Y \subseteq V$, denote by $\delta(X, Y)$ the set of edges with one endpoint in $X - Y$ and the other endpoint in $Y - X$ and $d(X, Y) := |\delta(X, Y)|$, and also define $\bar{d}(X, Y) := d(X \cap Y, V - (X \cup Y))$. For $X \subseteq V$,

define $\delta(X) := \delta(X, V-X)$ and the *degree* of $X$ as $d(X) := |\delta(X)|$. Denote the degree of a vertex as $d(v) := d(\{v\})$. Also denote the set of neighbors of $v$ by $N(v)$, and call a vertex in $N(v)$ a *v-neighbor*.

Let $\lambda(s,t)$ be the maximum number of edge-disjoint paths between $s$ and $t$ in $V$, and let $r(s,t)$ be an *edge-connectivity requirement* for $s,t \in V$. The connectivity requirement is *global* if $r(s,t) = k$ for all $s,t \in V$; otherwise it is *local*. We say a graph $G$ satisfies the connectivity requirements if $\lambda(s,t) \geq r(s,t)$ for any $s,t \in V$. The requirement $r(X)$ of a set $X \subseteq V$ is the maximum edge-connectivity requirement between $u$ and $v$ with $u \in X$ and $v \in V - X$. By Menger's theorem, to satisfy the requirements, it suffices to guarantee that $d(X) \geq r(X)$ for all $X \subset V$. The *surplus* $s(X)$ of a set $X \subseteq V$ is defined as $d(X) - r(X)$. A graph satisfies the edge-connectivity requirements if $s(X) \geq 0$ for all $\emptyset \neq X \subset V - x$, since $r(x,v) = 0$ for all $v$ in the complete edge splitting-off problem where $x$ is the vertex to be split off. For $X \subset V - x$, $X$ is called *dangerous* if $s(X) \leq 1$ and *tight* if $s(X) = 0$. A dangerous set is *maximal* if it is not a proper subset of any other dangerous set. The following proposition will be used throughout our proofs.

PROPOSITION 2.1 (see [10, Proposition 2.3]). *For $X, Y \subseteq V$ at least one of the following inequalities holds:*

$$s(X) + s(Y) \geq s(X \cap Y) + s(X \cup Y) + 2d(X,Y), \tag{2.1a}$$

$$s(X) + s(Y) \geq s(X - Y) + s(Y - X) + 2\bar{d}(X,Y). \tag{2.1b}$$

In this paper, we consider multigraphs, i.e., there can be multiple copies of an edge $uv$. We define the *capacity* of an edge pair to be the number of copies of the edge pair that can be split off while satisfying edge-connectivity requirements. In our algorithms, we always split off an edge pair to its capacity. Following the definition of Gabow [12], we say that a splitting-off operation at $x$ *voids* a vertex $u$ if $d(x,u) = 0$ after the splitting-off.

**2.1. Admissible pair and dangerous set.** In edge splitting-off problems, the objective is to split off a pair of edges incident on a designated vertex $x$ to maintain the edge-connectivity requirements for all other pairs in $V - x$. For this purpose, we may assume that the edge-connectivity requirements between $x$ and other vertices are zero, i.e., $r(V - x) = 0$, and thus the set $V - x$ is not a dangerous set. The following simple proposition characterizes the relation between admissible pair and dangerous set.

PROPOSITION 2.2 (see [10, Claim 3.1]). *A pair $xu$, $xv$ is not admissible if and only if $u, v$ are contained in a dangerous set.*

The following lemma proved in [6] shows that if the conditions in Mader's theorem are satisfied, then there is no "3-dangerous-set structure" as shown in Figure 2.1. This lemma is important in the efficient edge splitting-off algorithm.

LEMMA 2.3 (see [6, Lemma 2.7]). *If $d(x) \neq 3$ and there is no cut edge incident on $x$, then there are no maximal dangerous sets $X, Y, Z$ and $u, v, w \in N(x)$ with $u \in X \cap Y$, $v \in X \cap Z$, $w \in Y \cap Z$, and $u, v, w \notin X \cap Y \cap Z$.*

The next lemma is about a known reduction step of contracting tight sets, which will be useful in preprocessing the graph to obtain an indicator vertex. Suppose there is a *nontrivial* tight set $T$, i.e., $T$ is a tight set and $|T| \geq 2$. Clearly there are no admissible pairs $xu, xv$ with $u, v \in T$. Let $G/T$ be the graph obtained by contracting $T$ into a single vertex $t$. It is the graph arising from $G$ by removing vertex set $T$, adding a new vertex $t$, and adding $d(v,T)$ parallel edges between $t$ and $v$ for every $v \in V - T$.

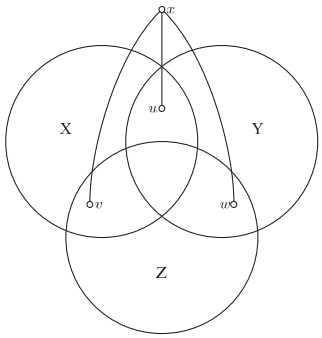


FIG. 2.1. *The 3-dangerous-set structure.*

Define the connectivity requirement $r(t, v)$ as $\max_{u \in T} r(u, v)$ while other connectivity requirements remain the same. The following lemma says that one can consider the admissible pairs in $G/T$ without losing any information about the admissible pairs in $G$. This lemma is useful in proofs to assume that every tight set is a singleton and is useful in algorithms to allow us to make progress by contracting nontrivial tight sets. The proof is provided in the appendix.

LEMMA 2.4 (see [20], [10, Claim 3.2]). *Let $T$ be a nontrivial tight set. For any $x$-neighbor $w$ in $G/T$, let $w'$ be the corresponding vertex in $G$ if $w \neq t$, and let $w'$ be any $x$-neighbor in $T$ in $G$ if $w = t$. Suppose $(xu, xv)$ is an admissible pair in $G/T$; then $(xu', xv')$ is an admissible pair in $G$.*

**2.2. Gomory–Hu tree.** As a key tool in checking local edge-connectivity, we need to construct a Gomory–Hu tree [14], which is a compact representation of all pairwise edge-connectivities. Let $G = (V, E)$ be an undirected graph. A Gomory–Hu tree of $G$ is a weighted tree $T = (V, F)$ such that for every pair $s, t \in V$, the weight of a min-cut separating them is the same in $G$ and in $T$. In other words, the local edge-connectivity between $s, t \in V$ in $G$ is equal to the weight of the minimum weighted edge on the $s$-$t$ path in $T$. In a partial Gomory–Hu tree $T_k$ of $G$ [15], vertices $s, t \in V$ are represented by the same node in $T_k$ if the local edge-connectivity between them is at least $k$ in $G$; otherwise, they are represented by different nodes and the local edge-connectivity between them equals the weight of the minimum weighted edge on the $s$-$t$ path in $T_k$. With a (partial) Gomory–Hu tree, it takes only linear time to check the local edge-connectivity between a pair of vertices. Bhalgat et al. [4] gave a fast randomized algorithm to construct a partial Gomory–Hu tree. We will use the following theorem by setting $k = r_{\max}$. The following result can be obtained by using the algorithm in [15], with the fast tree packing algorithm in [4].

THEOREM 2.5 (see [15, 4]). *A partial Gomory-Hu tree $T_k$ can be constructed in $\tilde{O}(km)$ expected time.*

Nagamochi and Ibaraki [22] gave a fast algorithm to find a sparse subgraph that satisfies edge-connectivity requirements, which will be used in section 3.3 as a preprocessing step. This allows us to reduce the number of edges to $O(r_{\max} \cdot n)$. Hence we can construct a partial Gomory–Hu tree $T_{r_{\max}}$ in $\tilde{O}({r_{\max}}^2 n)$ time.

THEOREM 2.6 (see [22, Lemma 2.1]). *There is an $O(m)$-time algorithm to construct a subgraph with $O(r_{\max} \cdot n)$ edges that satisfies all the connectivity requirements.*

**2.3. Hypothesis.** Throughout this paper, we assume that there is no cut edge incident on $x$. This holds at the input graph by our assumption (a condition of Mader's

theorem), and so the local edge-connectivity between each pair of $x$-neighbors is at least two. Therefore, we can reset the connectivity requirement between $u$ and $v$ as $\max\{r(u,v),2\}$ for all $u,v \in N(x)$, and hence splitting-off any admissible pair would maintain the property that there is no cut edge incident on $x$.

*Hypothesis* 2.7. There is no cut edge incident on $x$.

As mentioned earlier, we assume that $r(x) = 0$ for simplicity. We can handle the case with nonzero $r(x)$ by a few extra steps. First we add a new vertex $x'$ and $r(x)$ copies of $xx'$ edges (or $r(x)+1$ copies to keep $d(x)$ even). Then we can just apply the edge splitting-off results in this paper. After a complete edge splitting-off at $x$, we simply remove $x$ and rename $x'$ to $x$. Similarly, when $d(x)$ is odd (and $r(x) < 3$), we can add a new vertex $x'$ and three copies of $xx'$ edges. Therefore, we can always assume $d(x)$ to be even in a complete edge splitting-off algorithm. Note that these two assumptions are not essential in some structural results, such as Theorem 1.4.

**3. Efficient complete edge splitting-off algorithm.** Here we present the deterministic splitting-off algorithm as stated in Theorem 1.3. First we present an alternative proof of Mader's theorem in section 3.1. Extending the ideas in the alternative proof we show how to find a complete edge splitting-off sequence by only $O(|N(x)|)$ edge splitting-off attempts in section 3.2. Then, in section 3.3, we show how to efficiently perform one edge splitting-off attempt, by doing some preprocessing and applying some fast algorithms to check edge-connectivities. Combining these two steps yields an $\tilde{O}(m + {r_{\max}}^2 \cdot n^2)$ randomized algorithm for the complete splitting-off problem. Finally, in section 3.6, we describe how to modify some steps in section 3.3 to obtain an $\tilde{O}(m + {r_{\max}}^2 \cdot n^2)$ deterministic algorithm for the problem.

**3.1. Mader's theorem.** We present an alternative proof of Mader's theorem, which can be extended to obtain an efficient algorithm. Recall that non-admissible sets are subsets of $x$-neighbors with no admissible pair inside. The following lemma about non-admissible sets can be used directly to derive Mader's theorem.

LEMMA 3.1. *Suppose $d(x) \neq 3$. Then, for any non-admissible set $U \subseteq N(x)$ with $|U| \geq 2$, there is a dangerous set containing $U$.*

*Proof.* We prove the lemma by a simple induction. The statement holds trivially for $|U| = 2$ by Proposition 2.2. Consider $U = \{u_1, u_2, \ldots, u_{k+1}\} \subseteq N(x)$, where every pair $(u_i, u_j)$ is non-admissible. By induction, since every pair $(u_i, u_j)$ is non-admissible, there are maximal dangerous sets $X, Y$ such that $\{u_1, \ldots, u_{k-1}, u_k\} \subseteq X$ and $\{u_1, \ldots, u_{k-1}, u_{k+1}\} \subseteq Y$. Since $(u_k, u_{k+1})$ is non-admissible, by Proposition 2.2, there is a dangerous set $Z$ containing $u_k$ and $u_{k+1}$. If $u_{k+1} \notin X$ and $u_k \notin Y$ and there is some $u_i \notin Z$ by the maximality of $X$ and $Y$, then $X, Y$, and $Z$ form a 3-dangerous-set structure with $u = u_i, v = u_k, w = u_{k+1}$. By Lemma 2.3, this 3-dangerous-set structure does not exist. Hence either $X, Y$, or $Z$ contains $U$. ∎

To prove Mader's theorem, consider a vertex $x \in V$ with $d(x) \neq 3$ and there is no cut edge incident on it. Suppose that there is no admissible pair incident on $x$. Then, by Lemma 3.1, there is a dangerous set $D$ containing all the vertices in $N(x)$. But this is impossible since

$$r(V - D - x) = r(D) \geq d(D) - 1 = d(V - D - x) + d(x) - 1 \geq d(V - D - x) + 1,$$

contradicting that the connectivity requirements are satisfied in $G$. This completes the proof.

**3.2. An upper bound on splitting-off attempts.** Extending the ideas in the proof of Lemma 3.1, we present an algorithm to find a complete splitting-off sequence

by making at most $O(|N(x)|)$ splitting-off attempts (to split off to capacity). In the algorithm, we maintain a non-admissible set $C$: initially $C = \emptyset$, and at each iteration, we try to add one $x$-neighbor into $C$. The algorithm will apply one of the following three operations guaranteed by the following lemma. Here we assume that $\{u\}$ is a non-admissible set for every $u \in N(x)$. This can be achieved by a preprocessing step that splits off every $(u, u)$ to capacity (i.e., removes as many admissible pairs of $(xu, xu)$ as possible).

LEMMA 3.2. *Suppose that $C$ is a non-admissible set and there is a vertex $u \in N(x)-C$. Then, using at most three splitting-off attempts, at least one of the following operations can be applied:*

(1) *splitting-off an edge pair to capacity that voids an $x$-neighbor,*

(2) *deducing that every pair in $C \cup \{u\}$ is non-admissible and adding $u$ to $C$,*

(3) *contracting a tight set $T$ containing at least two $x$-neighbors.*

*Proof.* We consider three cases based on the size of $C$. When $|C| = 0$, we simply assign $C = \{u\}$. When $|C| = 1$, pick the vertex $v \in C$, and split off $(u, v)$ to capacity. Either case (1) applies when either $u$ or $v$ becomes void, or case (2) applies in the resulting graph after $(u, v)$ is split off to capacity. Hence, when $|C| \leq 1$, either case (1) or case (2) applies after only one splitting-off attempt.

The interesting case is when $|C| \geq 2$ and let $v_1, v_2 \in C$. Since $C$ is a non-admissible set, by Lemma 3.1, there is a maximal dangerous set $D$ containing $C$. First, we split off $(u, v_1)$ and $(u, v_2)$ to capacity. If case (1) applies, then we are done, so we assume that none of the three $x$-neighbors voids, implying that $(u, v_1)$ and $(u, v_2)$ are non-admissible in the resulting graph $G'$ after splitting-off these edge pairs to capacity. Note that the edge pair $(v_1, v_2)$ is also non-admissible since non-admissible edge pair in $G$ remains non-admissible in $G'$. By Lemma 3.1, there exists a maximal dangerous set $D'$ covering the non-admissible set $\{u, v_1, v_2\}$. Then inequality (2.1b) cannot hold for $D$ and $D'$, since that would imply

$$1+1 \geq s(D)+s(D') \geq s(D-D')+s(D'-D)+2\bar{d}(D, D') \geq 0+0+2d(x, \{v_1, v_2\}) \geq 2 \cdot 2.$$

Therefore inequality (2.1a) must hold for $D$ and $D'$, hence $1 + 1 = s(D) + s(D') \geq s(D \cap D')+s(D \cup D')$. This implies that either $D \cup D'$ is a dangerous set for which case (2) applies, since $C \cup \{u\}$ is contained in a dangerous set and hence every pair is a non-admissible pair by Proposition 2.2, or $D \cap D'$ is a tight set for which case (3) applies since $v_1$ and $v_2$ are $x$-neighbors. To distinguish between case (2) and case (3), we check the existence of any tight set containing $v_1$ and $v_2$ by one splitting-off attempt of $(xv_1, xv_2)$, as $v_1, v_2$ are contained in a tight set if and only if after splitting-off one copy of $(xv_1, xv_2)$ the connectivity requirement of some pair is violated by two.[1] Therefore, by making at most three splitting-off attempts $((xu, xv_1), (xu, xv_2), (xv_1, xv_2))$, one of the three operations can be applied. ∎

The following result can be obtained by applying Lemma 3.2 repeatedly.

LEMMA 3.3. *The algorithm computes a complete edge splitting-off sequence using at most $O(|N(x)|)$ numbers of splitting-off attempts.*

*Proof.* The algorithm maintains the property that $C$ is a non-admissible set, which holds at the beginning when $C = \emptyset$. It is clear that in case (2) the set $C$

[1] In section 3.5 we show how to find such a tight set efficiently by looking at a Gomory–Hu tree. It is not essential to contract tight sets in splitting-off attempts, although it is easier to explain in this way. In case (3) of Lemma 3.2, we just need the conclusion that the two $x$-neighbors are contained in some tight set. This allows us to group those two $x$-neighbors and handle them together in the rest of the algorithm using Lemma 2.4.

remains non-admissible. In case (1), by splitting-off an admissible pair, every pair of vertices in $C$ remains non-admissible. Also, in case (3), by contracting a tight set, every pair of vertices in $C$ remains non-admissible by Lemma 2.4.

The algorithm terminates when there is no vertex in $N(x) - C$. At that time, if $C = \emptyset$, then we have found a complete splitting-off sequence; if $C \neq \emptyset$, then by Mader's theorem (or by the proof in section 3.1), this happens only if $d(x) = 3$ and $d(x)$ is odd at the beginning. In any case, the longest splitting-off sequence is found and the given complete edge splitting-off problem is solved.

It remains to prove that the total number of splitting-off attempts in the whole algorithm is at most $O(|N(x)|)$. To see this, we claim that each of the operations in Lemma 3.2 will be performed at most $|N(x)|$ times. Indeed, cases (1) and (3) will be applied at most $|N(x)|$ times since each application reduces the number of $x$-neighbors by at least one, and case (2) will be applied at most $|N(x)|$ times since each application reduces the number of $x$-neighbors in $N(x) - C$ by one. ∎

**3.3. Algorithm outline.** The following is an outline of the whole algorithm for the complete splitting-off problem. First we use the $O(m)$ time algorithm in Theorem 2.6 by Nagamochi and Ibaraki [22] to construct a subgraph of $G$ with $O(r_{\max} \cdot n)$ edges satisfying the connectivity requirements. To find a complete splitting-off sequence, we can thus restrict our attention to maintaining the local edge-connectivity in this subgraph. In the next preprocessing step, we reduce the problem further to an instance where there is a particular *indicator vertex*. A vertex $t \neq x$ is an *indicator vertex* if for any pair of vertices $u, v \in V - x$ with $\lambda(u, v) \leq r_{\max}$, then it holds that $\lambda(u, v) = \min\{\lambda(u, t), \lambda(v, t)\}$. With an indicator vertex $t$, to check the change in local edge-connectivity for all pairs with $\lambda(u, v) \leq r_{\max}$, we only need to check the edge-connectivity from $t$ to every vertex $v$ with $\lambda(v, t) \leq r_{\max}$.

An indicator vertex is the key in checking the violation of connectivity requirement with only $O(n)$ queries (instead of $O(n^2)$ queries). To obtain an instance with an indicator vertex, we compute a partial Gomory–Hu tree $T_{r_{\max}}$ and contract appropriate tight sets. Recall that each edge of a Gomory–Hu tree represents a min-cut in the original graph. With a potential indicator vertex $t$, we can easily identify any $u, v \in V - x$ with $\lambda(u, v) \leq r_{\max}$ and $\lambda(u, v) > \min\{\lambda(u, t), \lambda(v, t)\}$. For such a pair $(u, v)$, either $v$ is on the $u$-$t$ path or $u$ is on the $v$-$t$ path in $T_{r_{\max}}$. In any case, we can contract the corresponding tight set to eliminate a vertex pair that violates the indicator property of $t$. We will present the details in section 3.4. The total preprocessing time is at most $\tilde{O}(m + {r_{\max}}^2 \cdot n)$, by using the fast Gomory–Hu tree algorithm in Theorem 2.5.

After these two preprocessing steps, we can perform a splitting-off attempt (split off a pair to capacity) efficiently. For a vertex pair $(u, v)$, we replace $\min\{d(x, u), d(x, v)\}$ copies of $xu$ and $xv$ by copies of $uv$, and then determine the maximum violation of connectivity requirements by constructing a partial Gomory–Hu tree and checking the edge-connectivities from the indicator vertex $t$ to every other vertex. If $q$ is the maximum violation of the connectivity requirements, then exactly $\min\{d(x, u), d(x, v)\} - \lceil q/2 \rceil$ copies of $(xu, xv)$ are admissible. The details will be discussed in section 3.5. Therefore, using Theorem 2.5, one splitting-off attempt can be performed in $\tilde{O}(r_{\max} \cdot m + n) = \tilde{O}({r_{\max}}^2 \cdot n)$ expected time. By Lemma 3.3, the complete splitting-off problem can be solved by at most $O(|N(x)|) = O(n)$ splitting-off attempts. Hence we obtain the following result.

THEOREM 3.4. *The complete edge splitting-off problem can be solved in* $\tilde{O}(m + {r_{\max}}^2 \cdot |N(x)| \cdot n) = \tilde{O}(m + {r_{\max}}^2 \cdot n^2)$ *expected time.*

This algorithm can be derandomized to a deterministic one with the same running time. The details will be discussed in section 3.6.

**3.4. Indicator vertex.** We will show how to reduce the problem into an instance with a particular indicator vertex $t \neq x$, with the property that if $\lambda(u,v) \leq r_{\max}$ for $u, v \neq x$, then $\lambda(u,v) = \min\{\lambda(u,t), \lambda(v,t)\}$. Hence, if we could maintain the local edge-connectivity from $t$ to $v$ for every $v \in V - x$ with $\lambda(v,t) \leq r_{\max}$, then the connectivity requirements for every pair in $V - x$ will be satisfied. Furthermore, by maintaining the local edge-connectivity, the indicator vertex $t$ will remain an indicator vertex, and therefore this procedure needs to be executed only once. Without loss of generality, we assume that the connectivity requirement for each pair of vertices $u, v \in V - x$ is equal to $\min\{\lambda(u,v), r_{\max}\}$, and $r(x,v) = 0$ for every $v \in V - x$.

First, we compute a partial Gomory–Hu tree $T_{r_{\max}}$ in $\tilde{O}(r_{\max} \cdot m)$ time by Theorem 2.5, which is $\tilde{O}({r_{\max}}^2 \cdot n)$ after applying the sparsifying algorithm in Theorem 2.6. Recall that each node in $T_{r_{\max}}$ represents a subset of vertices in $G$, and the edge-connectivities between these vertices are at least $r_{\max}$. In the following, we will use a capital letter (say, $U$) to denote both a node in $T_{r_{\max}}$ and the corresponding subset of vertices in $G$. If $T_{r_{\max}}$ has only one node, then this means that the local edge-connectivity between every pair of vertices in $G$ is at least $r_{\max}$. In this case, any vertex $t \neq x$ is an indicator vertex. So assume that $T_{r_{\max}}$ has at least two nodes. Let $X$ be the node in $T_{r_{\max}}$ that contains $x$ in $G$, let $U_1, \ldots, U_p$ be the nodes adjacent to $X$ in $T_{r_{\max}}$, and let $XU_1$ be the edge in $T_{r_{\max}}$ with largest weight among $XU_i$ for $1 \leq i \leq p$.

Suppose $X$ contains a vertex $t \neq x$ in $G$. The idea is to contract tight sets so that $t$ will become an indicator vertex in the resulting graph. For any edge $XU_i$ in $T_{r_{\max}}$, let $T_i'$ be the component of $T_{r_{\max}}$ that contains $U_i$ when $XU_i$ is removed from $T_{r_{\max}}$. We claim that each $U_i^* := \cup_{U \in T_i'} U$ is a tight set in $G$; see Figure 3.1. By the definition of a Gomory–Hu tree, the local edge-connectivity between any vertex $u_i \in U_i$ and $t$ is equal to the edge weight of $XU_i$ in $T_{r_{\max}}$. Also, by the definition of a Gomory–Hu tree, $d_G(U_i^*)$ is equal to the weight of edge $XU_i$ in $T_{r_{\max}}$. Therefore, $U_i^*$ is a tight set in $G$, because $r(u_i, t) = \lambda(u_i, t) = d(U_i^*)$ for some pair $u_i, t \in V - x$. By Proposition 2.2, we can contract each $U_i^*$ into a single vertex $u_i$ for $1 \leq i \leq p$ without losing any information about admissible pairs in $G$. Since each $U_i^*$ becomes a single vertex, the vertex $t$ becomes an indicator vertex in the resulting graph.

Suppose $X$ contains only $x$ in $G$, i.e., $X = x$. Then $U_1^*$ may not be a tight set, since there may not exist a pair $u, v \in V - x$ with $r(u,v) = \lambda(u,v) = d(U_1^*)$. (Note that there is a vertex $v$ with $\lambda(x,v) = d(U_1^*)$, but $r(x,v) = 0$ for every vertex $v$.) In this case, we contract some tight sets so that any vertex in $U_1$ will become an indicator vertex. Let $W_1 \neq x, \ldots, W_q \neq x$ be the nodes (if any) adjacent to $U_1$ in

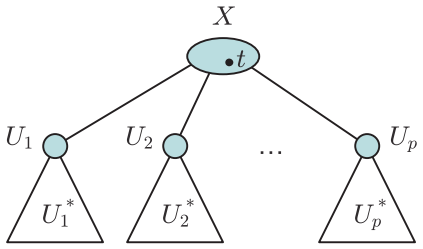


FIG. 3.1. *The case when $X$ contains a vertex $t \neq x$. In this case each $U_i^*$ is a tight set. After contracting each $U_i^*$ into a single vertex, the vertex $t$ becomes an indicator vertex.*

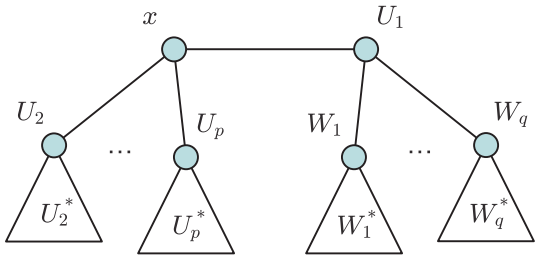


FIG. 3.2. *The case when $X$ contains only $x$. In this case each $U_i^*$ is a tight set for $2 \le i \le p$, and each $W_j^*$ is a tight set for $1 \le j \le q$. After contracting each $U_i^*$ for $2 \le i \le p$ and each $W_j^*$ for $1 \le j \le q$ into a single vertex, any vertex $t \in U_1$ becomes an indicator vertex.*

$T_{r_{\max}}$; see Figure 3.2. By using similar arguments as before, it can be shown that each $U_i^*$ is a tight set for $2 \le i \le p$ (through $u_i \in U_i$ and $u_1 \in U_1$). Therefore we can contract each $U_i^*$ into a single vertex $u_i$ for $2 \le i \le p$. Similarly, we can argue that each $W_j^*$ (defined analogously as $U_i^*$) is a tight set, and hence we can contract each $W_j^*$ into a single vertex $w_j$ for each $1 \le j \le q$. We can see that any vertex $t \in U_1$ is an indicator vertex in the resulting graph, because $\lambda(t,v) \ge \min\{\lambda(w,v), r_{\max}\}$ for any pair of vertices $v, w$ since $XU_1$ is the edge with the largest weight in $T$.

Henceforth, we can consider this resulting graph instead of $G$ for the purpose of computing a complete splitting-off sequence, and using $t$ as the indicator vertex to check connectivities. The running time of this procedure is dominated by the partial Gomory–Hu tree computation, which is at most $\tilde{O}({r_{\max}}^2 \cdot n)$.

**3.5. Splitting-off to capacity.** We show how to split off a pair $u, v$ of $x$-neighbors to capacity efficiently. Let $G'$ be the graph obtained from $G$ by splitting-off $\min\{d(x,u), d(x,v)\}$ copies of the edge pair $(xu, xv)$. Let $t$ be the indicator vertex as defined in section 3.4. By the definition of $t$, if the edge-connectivity requirement from $t$ to every vertex $v \in V - x$ (which is equal to $\min\{\lambda(v,t), r_{\max}\}$) is satisfied, then the local edge-connectivity for every pair $u, v \in V - x$ is satisfied. Let $q = \max_{w \in V-x} \{r(w,t) - \lambda_{G'}(w,t)\}$ be the maximum violation of the edge-connectivity requirement from $t$ in $G'$. Let $G''$ be the graph obtained from $G$ by splitting-off $\min\{d(x,u), d(x,v)\} - \lceil q/2 \rceil$ copies of the edge pair $(xu, xv)$. We show in the following claim that the edge-connectivity requirement from $t$ to every vertex $v \in V - x$ is satisfied in $G''$. This implies that exactly $\min\{d(x,u), d(x,v)\} - \lceil q/2 \rceil$ copies of the edge pair $(xu, xv)$ are admissible in $G$.

*Claim* 3.5. For any vertex $s \in V - x$, it holds that $\lambda_{G''}(s,t) \ge r(s,t)$ in $G''$.

*Proof.* By Menger's theorem, we only need to check that $d_{G''}(X) \ge r(s,t)$ for any $X$ separating $s$ and $t$. The edge-connectivity requirements are satisfied in $G$, and so we have $d_G(X) \ge r(s,t)$. The splitting-off operation decreases the degree of $X$ if and only if $u, v \in X$. By definition, $q \ge r(s,t) - \lambda_{G'}(s,t) \ge r(s,t) - d_{G'}(X)$. On the other hand, we have $d_{G''}(X) \ge d_{G'}(X) + q$, since $\lceil q/2 \rceil$ copies of the edge pair are "unsplit." Therefore $d_{G''}(X) \ge r(s,t)$. ∎

To check the maximum violation, we compute a partial Gomory-Hu tree $T_{r_{\max}}$ in $\tilde{O}({r_{\max}}^2 \cdot n)$ time. Then we can easily compute $q = \max_{v \in V-x} \{r(v,t) - \lambda_{G'}(v,t)\}$ in $O(n)$ time. Therefore, splitting-off a pair to capacity can be implemented in $\tilde{O}({r_{\max}}^2 \cdot n)$ time. Furthermore, $u$ and $v$ are contained in a tight set in $G''$ if and only if $q = \max_{v \in V-x} \{r(v,t) - \lambda_{G'}(v,t)\}$ is even. Hence, we can use one splitting-off attempt to check the existence of any tight set containing $u$ and $v$, as needed in Lemma 3.2.

Finally, we remark that one can identify the violating cut (i.e., a dangerous set or a tight set) easily from the Gomory–Hu tree.

**3.6. Deterministic algorithm.** We describe how to modify the randomized algorithm in Theorem 3.4 to obtain a deterministic algorithm with the same running time. Every step in the algorithm is deterministic except the Gomory–Hu tree construction in Theorem 2.5. The randomized Gomory–Hu tree construction is used in two places. First it is used in finding an indicator vertex in section 3.4, and for this purpose it is executed only once. Here we can replace it by a slower deterministic partial Gomory–Hu tree construction algorithm. It is well known that a Gomory–Hu tree can be computed using at most $n-1$ max-flow computations [14]. By using the Ford–Fulkerson flow algorithm, one can obtain an $O({r_{\max}}^2 \cdot n^2)$-time deterministic algorithm to construct a partial Gomory–Hu tree $T_{r_{\max}}$.

The randomized partial Gomory-Hu construction is also used in every splitting-off attempt to check whether the connectivity requirements are satisfied. With the indicator vertex $t$, this task reduces to checking the edge-connectivities from $t$ to other vertices, and there is a fast deterministic algorithm for this simpler task by Bhalgat et al. [4].

THEOREM 3.6 (see [4]). *Given an undirected graph $G$ and a vertex $t$, there is an $\tilde{O}(r_{\max} \cdot m)$-time deterministic algorithm to compute $\min\{\lambda_G(t,v), r_{\max}\}$ for all vertices $v \in G$.*

Thus, we can replace the randomized partial Gomory–Hu tree algorithm by this algorithm, and so Theorem 3.4 still holds deterministically. Hence there is a deterministic $\tilde{O}(m + {r_{\max}}^2 \cdot n^2)$ time algorithm for the complete splitting-off problem.

**4. Structural property and randomized algorithm.** Before we give the proof of Theorem 1.4, we first show how to use it in a randomized edge splitting-off procedure to speed up the algorithm. By Theorem 1.4, when the degree of $x$ is much larger than $2r_{\max}$, even a random edge pair will be admissible with probability at least $1 - 2r_{\max}/(d(x)-1)$. Using this observation, we show how to reduce $d(x)$ to $O(r_{\max})$ in $\tilde{O}(m + {r_{\max}}^3 \cdot n)$ time. Then, by Theorem 3.4, the remaining edges can be split off in $\tilde{O}(m + {r_{\max}}^2 \cdot d(x) \cdot n) = \tilde{O}(m + {r_{\max}}^3 \cdot n)$ time. So, the total running time of the complete splitting-off algorithm is improved to $\tilde{O}(m + {r_{\max}}^3 \cdot n)$, proving Theorem 1.3.

The idea is to split off many random edge pairs in parallel, before checking if some connectivity requirement is violated. Suppose that $2^{l+q-1} < d(x) \le 2^{l+q}$ and $2^{l-1} < r_{\max} \le 2^l$ for some positive integers $l$ and $q$. To reduce $d(x)$ to $2^{l+q-1}$, we need to split off at most $2^{l+q-1}$ $x$-edges. Since each $x$-edge has fewer than $2r_{\max}$ non-admissible partners by Theorem 1.4, the probability that a random edge pair is admissible is at least

$$\frac{(d(x)-1) - 2r_{\max}}{d(x)-1} \ge \frac{2^{l+q-1} - 2^l}{2^{l+q-1}} = \frac{2^{q-1}-1}{2^{q-1}}.$$

Now, consider a random splitting-off operation that split off at most $2^{q-2}$ edge pairs at random in parallel. The operation is successful if all the edge pairs are admissible. After each operation, we run the checking algorithm as in section 3.5 to determine whether this operation is successful. Consider an iteration that repeats the operation until an operation succeeds. The probability for the operation to succeed is at least $(\frac{2^{q-2}-1}{2^{q-2}})^{2^{q-2}} = \Omega(1)$. The expected number of operations in an iteration is $O(1)$. Hence, the expected runtime for an iteration is $\tilde{O}({r_{\max}}^2 \cdot n)$.

Since each iteration reduces the degree of $x$ by $2^{q-2}$, with at most $2^{l+1} = O(r_{\max})$ iterations, we can then reduce $d(x)$ to $2^{l+q-1}$, i.e., reduce $d(x)$ by half. This procedure is applicable as long as $q \geq 3$. Therefore, we can reduce $d(x)$ to $2^{l+2}$ by using this procedure for $O(\log n)$ times. The total expected runtime is thus $\tilde{O}(m + 2^{l+1} \cdot \log n \cdot {r_{\max}}^2 \cdot n) = \tilde{O}(m + {r_{\max}}^3 \cdot n)$.

**4.1. Proof of Theorem 1.4.** In this section, we prove that each edge has at most $2r_{\max} - 2$ non-admissible partners. Given an edge pair $(xv, xw)$, if it is a non-admissible pair, then there is a dangerous set $D$ with $\{xv, xw\} \subseteq \delta(D)$ by Proposition 2.2, and we say such a dangerous set $D$ *covers* $xv$ and $xw$. Let $P$ be the set of non-admissible partners of $xv$ in the initial graph. Our goal is to show that $|P| \leq 2r_{\max} - 2$. This bound is the best possible as there are examples achieving it (see Figure 4.2).

Let $\mathcal{D}_P$ be a minimal set of maximal dangerous sets such that (i) each set $D \in \mathcal{D}_P$ covers the edge $xv$, and (ii) each edge in $P$ is covered by some set $D \in \mathcal{D}_P$. $D_P$ is minimal so that no smaller collection of maximal dangerous sets satisfies these two properties. The structure of $D_P$ is illustrated in Figure 4.1. We prove the theorem by considering different values of $|\mathcal{D}_P|$.

The theorem follows immediately if $|\mathcal{D}_P| \leq 1$. When $|\mathcal{D}_P| = 2$, let $\mathcal{D}_P = \{X, Y\}$. By Proposition 4.1, $d(X \cap Y, X - Y), d(X \cap Y, Y - X) \geq 1$ as $X, Y$ are dangerous sets. Since $d(D) \leq r_{\max} + 1$ and $D$ covers $xv$ for each $D \in \mathcal{D}_P$, we have $d(x, X - Y), d(x, Y - X) \leq r_{\max} - 1$, proving $|P| \leq 2r_{\max} - 2$.

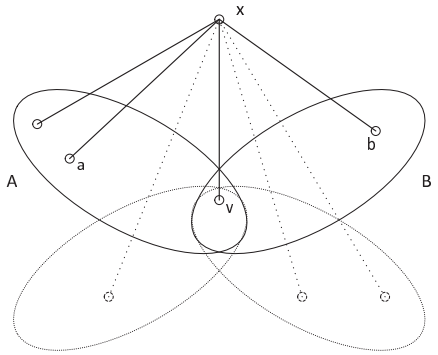


FIG. 4.1. *A minimal collection of maximal dangerous set that covers all non-admissible partners of* $xv$.

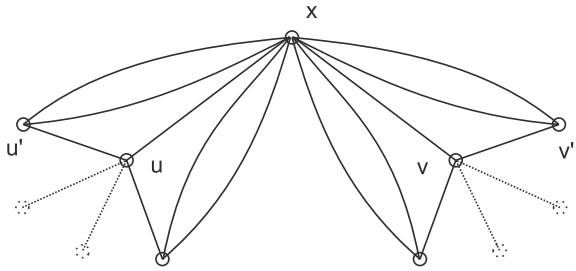


FIG. 4.2. *The example when* $xu$ *has* $2r_{\max} - 2$ *non-admissible partners. Apart from* $x$, $u$ *has* $r_{\max} - 1$ *neighbors, and each of them has one edge connecting to* $u$ *and two edges connecting to* $x$. *The configuration of* $v$ *is the same as that of* $u$. *We can see that* $\lambda(u, v) = r_{\max}$. *If we split off* $(xu, xu')$ *for any* $u' \in N(u)$, *then* $d(\{u, u'\}) = r_{\max} - 1$. *Hence,* $xu'$ *is a non-admissible partner of* $xu$ *for every* $u' \in N(u)$.

The interesting case is $|\mathcal{D}_P| \geq 3$. We first prove in Lemma 4.5 that $|\mathcal{D}_P| \leq r_{\max} - 1$. Then, we use an inductive argument to show that $|P| \leq r_{\max} - 1 + |\mathcal{D}_P|$. These give $|P| \leq 2r_{\max} - 2$ and prove the theorem. The following propositions are used whose proofs are provided in the appendix for completeness.

PROPOSITION 4.1 (see [2, Lemma 5.4]). *For any disjoint vertex sets $S_1, S_2$ with $d(S_1, S_2) = 0$ and $d(x, S_1) \geq 1$ and $d(x, S_2) \geq 1$, $S_1 \cup S_2$ is not a dangerous set.*

PROPOSITION 4.2 (see [2, Lemma 5.4]). *If $d(x, v) \geq 2$, then inequality* (2.1a) *holds for two dangerous sets $X, Y$ containing $v$.*

LEMMA 4.3 (see [2, Lemma 5.4]). *If $|\mathcal{D}_P| \geq 3$, then inequality* (2.1a) *holds for every $X, Y \in \mathcal{D}_P$.*

LEMMA 4.4 (see [25, Lemma 2.6]). *If $|\mathcal{D}_P| \geq 3$, then $X \cap Y = \{v\}$ and is a tight set for any $X, Y \in \mathcal{D}_P$.*

We assume there is no cut edge incident on $x$ (Hypothesis 2.7) and $r_{\max} \geq 2$ as required in the proof by Theorem 1.4. By contracting nontrivial tight sets, each edge in $P$ is still a non-admissible partner of $xv$ by Lemma 2.4. Henceforth, we may assume that all tight sets are singletons.

**4.1.1. An upper bound on $|\mathcal{D}_P|$.** The following proof uses very similar ideas as in [2, 25].

LEMMA 4.5. *$|\mathcal{D}_P| \leq r_{\max} - 1$ when $|\mathcal{D}_P| \geq 3$.*

*Proof.* Suppose $|\mathcal{D}_P| \geq 3$. By Lemma 4.4, we have $X \cap Y = \{v\}$ for any $X, Y \in \mathcal{D}_P$. For each set $X \in \mathcal{D}_P$, we have $d(x, v) \geq 1$ and $d(x, X - v) \geq 1$ by the minimality of $\mathcal{D}_P$. Therefore, we must have $d(v, X - v) \geq 1$ by Proposition 4.1. By Lemma 4.4, $X - v$ and $Y - v$ are disjoint for each pair $X, Y \in \mathcal{D}_P$. Since $d(v, X - v) \geq 1$ for each $X \in \mathcal{D}_P$ and $d(x, v) \geq 1$, it follows that $|\mathcal{D}_P| \leq d(v) - 1$. By Lemma 4.4, $\{v\}$ is a tight set, and thus $|\mathcal{D}_P| \leq d(v) - 1 \leq r_{\max} - 1$. ∎

**4.1.2. An inductive argument.** When $|\mathcal{D}_P| \geq 3$, we prove $|P| \leq r_{\max} - 1 + |\mathcal{D}_P|$ by induction on $|\mathcal{D}_P|$. We first outline the inductive argument. By the minimality of $\mathcal{D}_P$, we can argue that there exists an admissible pair in $P$. After splitting-off such pair, $|P|$ is reduced by two. We prove in Lemma 4.6 that $|\mathcal{D}_P|$ is also reduced by at least two. By repeating this reduction, the remaining edges in $P$ are covered by one or two dangerous sets at the end. Then, we can conclude that $|P| \leq r_{\max} - 1 + |\mathcal{D}_P|$. In the following, we will provide the details of the inductive argument.

The theorem follows immediately if $d(x, X - v) = 1$ for every dangerous set $X \in \mathcal{D}_P$. Hence, we assume that there is a dangerous set $A \in \mathcal{D}_P$ with $d(x, A - v) \geq 2$; this property will be used only at the very end of the proof. For any $B \in \mathcal{D}_P - A$, there exist $x$-neighbors $a \in A \cap P, b \in B \cap P$ such that the edge pair $(xa, xb)$ is admissible. Otherwise, by Lemma 3.1, there exists a dangerous set covering $(A \cup B) \cap P$, which contradicts the minimality of $\mathcal{D}_P$.

After splitting-off $(xa, xb)$, let the resulting graph be $G'$ and $P' = P - \{xa, xb\}$. Let $\mathcal{D}_{P'}$ be a set of maximal dangerous sets such that (i) each set $D \in \mathcal{D}_{P'}$ covers the edge $xv$, (ii) each edge in $P'$ is covered by some set $D \in \mathcal{D}_{P'}$ The following lemma shows that there exists some $\mathcal{D}_{P'}$ such that $|\mathcal{D}_{P'}| \leq |\mathcal{D}_P| - 2$.

LEMMA 4.6. *When $|\mathcal{D}_P| \geq 3$, the edges in $P'$ can be covered by a set $\mathcal{D}_{P'}$ of maximal dangerous sets in $G'$ such that* (i) *each set in $\mathcal{D}_{P'}$ covers $xv$,* (ii) *each edge in $P'$ is covered by some set in $\mathcal{D}_{P'}$, and* (iii) *$|\mathcal{D}_{P'}| \leq |\mathcal{D}_P| - 2$.*

*Proof.* We use the dangerous sets in $\mathcal{D}_P$ to construct $\mathcal{D}_{P'}$. By Lemma 4.3, each pair of sets in $\mathcal{D}_P$ satisfies inequality (2.1a). By the minimality of $\mathcal{D}_P$, we have $s(A \cup B) = 2$ before splitting-off $(xa, xb)$. Hence $A \cup B$ is a tight set after

splitting-off. Contract $A \cup B$ into $v$. By Lemma 2.4, each $e \in \mathcal{D}_P$ is still a non-admissible partner of $xv$ in $G'$. Therefore, for each $C \in D_P$, $P \cap C + v$ is still a non-admissible set. By Lemma 3.1, there exists a dangerous set covering $P \cap C + v$ in $G'$. Hence, there exists $\mathcal{D}_{P'}$, a collection of at most $|\mathcal{D}_P - A - B|$ maximal dangerous sets covering $P'$. □

Remove and replace some of the maximal dangerous sets in $D_{P'}$ so that it becomes the smallest set to satisfy properties (i) and (ii). Recall that we chose $A$ with $d(x, A - v) \geq 2$, and hence $d(x, v) \geq 2$ after the splitting-off and contraction of tight sets. Therefore, by Proposition 4.2, inequality (2.1a) holds for every two maximal dangerous sets in $\mathcal{D}_{P'}$.

By induction, when $|\mathcal{D}_P| \geq 3$, we have $|P| = |P'| + 2 \leq r_{\max} - 1 + |\mathcal{D}_{P'}| + 2 \leq r_{\max} - 1 + |\mathcal{D}_P|$. In the base case with $|\mathcal{D}_P| = 3$, only one dangerous set leaves after splitting-off. Hence $|P| = |P'| + 2 \leq (r_{\max} + 1) - 1 + 2 \leq r_{\max} - 1 + |\mathcal{D}_P|$. In another base case with $|\mathcal{D}_P| = 2$, inequality (2.1a) holds for $C, D \in \mathcal{D}_P$. The argument in Lemma 4.6 can be used to show that the edges in $P'$ are covered by one *tight* set after splitting-off $(xa, xb)$. Thus $|P| = |P'| + 2 \leq r_{\max} - 1 + 2 \leq r_{\max} - 1 + |\mathcal{D}_P|$. This completes the proof that $|P| \leq r_{\max} - 1 + |\mathcal{D}_P|$.

**Concluding remarks.** Theorem 1.4 can be applied to constrained edge splitting-off problems, and give additive approximation algorithms for constrained augmentation problems. The efficient algorithms can also be adapted to these problems. We refer the reader to [26] for these results.

In a recent work [8], it was shown that one can reduce the number of splitting-off attempts to $O(|N(x)|)$ (Lemma 3.3) by a simple reduction using an expander graph.

**Appendix. Proofs of some known results.** We include the proofs of some known results for completeness and for the convenience of the readers.

*Proof of Lemma* 2.4. Suppose, by way of contradiction, that $(xu', xv')$ is not an admissible pair in $G$. Then there exists a maximal dangerous set $D$ containing $u', v'$ in $G$, i.e., $s_G(D) \leq 1$. If either $D \cap T = \emptyset$ or $T \subseteq D$, then

$$s_{G/T}(D - T + t) = d_{G/T}(D - T + t) - r_{G/T}(D - T + t) = d_G(D) - r_G(D) = s_G(D) \leq 1.$$

Hence $D - T + t$ is also a dangerous set in $G/T$, contradicting the assumption that $(xu, xv)$ is an admissible pair in $G/T$. Therefore $D - T \neq \emptyset$ and $T - D \neq \emptyset$. Inequality (2.1a) cannot hold for $D$ and $T$. For otherwise, $D \cup T$ is also a dangerous set in $G$, since

$$1 + 0 = s_G(D) + s_G(T) \geq s_G(D \cup T) + s_G(D \cap T),$$

contracting the maximality of $D$ since $T - D \neq \emptyset$. Therefore inequality (2.1b) must hold for $D$ and $T$ in $G$, and hence

$$1 + 0 = s_G(D) + s_G(T) \geq s_G(D - T) + s_G(T - D) + 2\bar{d}_G(D, T).$$

It follows that $(D - T)$ is a dangerous set and $xu, xv \in \delta_G(D - T)$ (since $\bar{d}_G(D, T) = 0$), which contradicts that $(xu, xv)$ is an admissible pair in $G/T$. Therefore $D$ does not exist, proving the lemma. □

*Proof of Proposition* 4.1. For any disjoint vertex sets $S_1$ and $S_2$ with $d(S_1, S_2) = 0$, we have

$$\begin{aligned} s(S_1 \cup S_2) &= d(S_1 \cup S_2) - r(S_1 \cup S_2) \geq d(S_1) + d(S_2) - \max\{r(S_1), r(S_2)\} \\ &\geq \min\{d(S_1), d(S_2)\}. \end{aligned}$$

Since $d(x, S_1) \geq 1$ and $d(x, S_2) \geq 1$, it follows that either $d(S_1) = 1$ or $d(S_2) = 1$, which implies a cut edge incident on $x$, violating the assumption. □

*Proof Proposition* 4.2. Suppose to the contrary that inequality (2.1a) does not hold. Then inequality (2.1b) must hold for $X$ and $Y$, which is impossible since

$$\begin{aligned} 1+1 &\geq s(X)+s(Y) \geq s(X-Y)+s(Y-X)+2\bar{d}(X,Y) \\ &\geq s(X-Y)+s(Y-X)+2d(x,v) \\ &\geq s(X-Y)+s(Y-X)+2\cdot 2. \end{aligned}$$ □

*Proof of Lemma* 4.3. Suppose, by way of contradiction, that inequality (2.1b) holds for $X$ and $Y$. Then

$$\begin{aligned} 1+1 &\geq s(X)+s(Y) \geq s(X-Y)+s(Y-X)+2\bar{d}(X,Y) \\ &\geq s(X-Y)+s(Y-X)+2(d(x,v)+d(V-x-X\cup Y, X\cap Y)) \\ &\geq s(X-Y)+s(Y-X)+2+2d(V-x-X\cup Y, X\cap Y). \end{aligned}$$

It implies that both $X-Y$ and $Y-X$ are tight and $d(V-x-(X\cup Y), X\cap Y)=0$. Since tight sets are singletons, we let $X-Y=\{a\}$ and $Y-X=\{b\}$. By the minimality of $\mathcal{D}_P$, $a$ and $b$ are both $x$-neighbors and no $Z \in \mathcal{D}_P - X - Y$ contains either $a$ or $b$. Since $d(V-x-(X\cup Y), X\cap Y)=0$ and $a, b \notin Z$, it follows that $d(Z-(X\cap Y\cap Z), X\cap Y\cap Z)=0$. Let $S_1 = X\cap Y\cap Z$ and $S_2 = Z - X\cap Y\cap Z$, and thus $d(S_1, S_2)=0$. Since $v \in X\cap Y\cap Z$, we have $d(x, S_1)\geq 1$. By the minimality of $\mathcal{D}_P$, there is an $x$-neighbor in $S_2 = Z-(X\cap Y\cap Z)$ and thus $d(x, S_2)\geq 1$. By Proposition 4.1, $Z = S_1 \cup S_2$ is not a dangerous set, contradicting the definition of $\mathcal{D}_P$. □

*Proof of Lemma* 4.4. Since $X, Y \in \mathcal{D}_P$ are maximal dangerous sets, $X\cup Y$ is not a dangerous set. By Lemma 4.3, inequality (2.1a) holds for $X$ and $Y$, and it follows that $X\cap Y$ is a tight set. Since each tight set is a singleton and $v \in X\cap Y$, the proposition follows. □

**Acknowledgments.** We thank two anonymous reviewers for many useful comments and suggestions that significantly improved the presentation of the paper.

## REFERENCES

[1] J. Bang-Jensen, A. Frank, and B. Jackson, *Preserving and increasing local edge-connectivity in mixed graphs*, SIAM J. Discrete Math., 8 (1995), pp. 155–178.

[2] J. Bang-Jensen and T. Jordán, *Edge-connectivity augmentation preserving simplicity*, SIAM J. Discrete Math., 11 (1998), pp. 603–623.

[3] A. A. Benczúr and D. R. Karger, *Augmenting undirected edge connectivity in $O(n^2)$ time*, J. Algorithms, 37 (2000), pp. 2–36.

[4] A. Bhalgat, R. Hariharan, T. Kavitha and D. Panigrahi, *An $\tilde{O}(mn)$ Gomory-Hu tree construction algorithm for unweighted graphs*, Proceedings of the 39th Annual ACM Symposium on Theory of Computing, (2007), pp. 605–614,.

[5] A. Bhalgat, R. Hariharan, T. Kavitha, and D. Panigrahi, *Fast edge splitting and Edmonds' arborescence construction for unweighted graphs*, in Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms, 2008, pp. 455–464.

[6] Y. H. Chan, W. S. Fung, L. C. Lau, and C. K. Yung, *Degree bounded network design with metric costs*, in Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science, 2008, pp. 125–134.

[7] E. Cheng and T. Jordán, *Successive edge-connectivity augmentation problems*, Math. Program., 84 (1999), pp. 577–593.

[8] H. Y. Cheung, L. C. Lau, and K. M. Leung, *Graph connectivities, network coding, and expander graphs*, in Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science, 2011, pp. 190–199.

[9] A. Frank, *Augmenting graphs to meet edge-connectivity requirements*, SIAM J. Discrete Math., 5 (1992), pp. 25–53.
[10] A. Frank, *On a theorem of Mader*, Ann. Discrete Math., 101 (1992), pp. 49–57.
[11] A. Frank and Z. Király, *Graph orientations with edge-connection and parity constraints*, Combinatorica, 22 (2002), pp. 47–70.
[12] H. N. Gabow, *Efficient splitting off algorithms for graphs*, in Proceedings of the 26th Annual ACM Symposium on the Theory of Computing, 1994, pp. 696–705.
[13] M. X. Goemans and D. J. Bertsimas, *Survivable networks, linear programming relaxations and the parsimonious property*, Math. Program., 60 (1993), pp. 145–166.
[14] R. E. Gomory and T. C. Hu, *Multi-terminal network flows*, J. SIAM, 9 (1961), pp. 551–570.
[15] R. Hariharan, T. Kavitha, and D. Panigrahi, *Efficient algorithms for computing all low s-t edge connectivities and related problems*, in Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms, (2007), pp. 127–136.
[16] T. Jordán, *On minimally k-edge-connected graphs and shortest k-edge-connected Steiner networks*, Discrete Appl. Math., 131 (2003), pp. 421–432.
[17] L. C. Lau, *An approximate max-Steiner-tree-packing min-Steiner-cut theorem*, Combinatorica, 27 (2007), pp. 71–90.
[18] L. Lovász, *Lecture*, Presented at Conference of Graph Theory, Prague, 1974.
[19] L. Lovász, *Combinatorial Problems and Exercises*, North-Holland, Amsterdam, 1979.
[20] W. Mader, *A reduction method for edge-connectivity in graphs*, Ann. Discrete Math., 3 (1978), pp. 145–164.
[21] H. Nagamochi, *A fast edge-splitting algorithm in edge-weighted graphs*, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 2006, pp. 1263–1268.
[22] H. Nagamochi and T. Ibaraki, *Linear time algorithm for finding a sparse k-connected spanning subgraph of a k-connected graph*, Algorithmica, 7 (1992), pp. 583–596.
[23] H. Nagamochi and T. Ibaraki, *Deterministic O(nm) time edge-splitting in undirected graphs*, J. Combin. Optim., 1 (1997), pp. 5–46.
[24] C. S. J. A. Nash-Williams, *On orientations, connectivity and odd vertex pairings in finite graphs*, Canad. J. Math., 12 (1960), pp. 555–567.
[25] Z. Szigeti, *Edge-splittings preserving local edge-connectivity of graphs*, Discrete Appl. Math., 156 (2008), pp. 1011–1018.
[26] C. K. Yung, *Edge Splitting-Off and Network Design Problems*, MS thesis, The Chinese University of Hong Kong, 2009.