Discrete Applied Mathematics 123 (2002) 447–472

DISCRETE
APPLIED
MATHEMATICS

# Graph connectivity and its augmentation: applications of MA orderings

Hiroshi Nagamochi[a,*], Toshihide Ibaraki[b]

[a]*Department of Information and Computer Sciences, Toyohashi University of Technology, Hibarigaoka, Tenpaku, Toyohashi 441-8580, Japan*
[b]*Department of Applied Mathematics and Physics, Kyoto University, Sakyo, Kyoto 606-8501, Japan*

Received 4 October 1999; received in revised form 26 June 2000; accepted 30 October 2000

## Abstract

This paper surveys how the maximum adjacency (MA) ordering of the vertices in a graph can be used to solve various graph problems. We first explain that the minimum cut problem can be solved efficiently by utilizing the MA ordering. The idea is then extended to a fundamental operation of a graph, edge splitting. Based on this, the edge-connectivity augmentation problem for a given $k$ (and also for the entire range of $k$) can be solved efficiently by making use of the MA ordering, where it is asked to add the smallest number of new edges to a given graph so that its edge-connectivity is increased to $k$. Other related topics are also surveyed. © 2002 Elsevier Science B.V. All rights reserved.

*Keywords:* Graphs; MA ordering; Minimum cuts; Edge-connectivity; Graph augmentation; Polynomial time algorithms

## 1. Introduction

This paper surveys how the maximum adjacency (MA) ordering of vertices of a graph can be used to solve various graph problems. Let an undirected multigraph (i.e., an undirected graph with integer edge weights) be given, which has $n$ vertices. An ordering $v_1, v_2, \ldots, v_n$ of vertices is called an *MA ordering* if an arbitrary vertex is chosen as $v_1$, and after choosing the first $i$ vertices $v_1, \ldots, v_i$, the $(i+1)$th vertex $v_{i+1}$ is chosen from the vertices $u$ that have the largest number of edges between $\{v_1, \ldots, v_i\}$ and $u$. The ordering was proposed in [20,69] to compute a sparse $k$-edge (resp., $k$-vertex)

---

[*] Corresponding author.
*E-mail addresses:* naga@ics.tut.ac.jp (H. Nagamochi), ibaraki@amp.i.kyoto-u.ac.jp (T. Ibaraki).

PII: S0166-218X(01)00349-3

connected spanning subgraph in a given $k$-edge (resp., $k$-vertex) connected graph. It was called a legal ordering in [20], and the name "MA ordering" was coined by Matula [67]. The MA ordering is identical with the maximum cardinality ordering which was discovered by Tarjan and Yannakakis [87] to test the chordality of graphs. But in an MA ordering, we also label the edges in order to decompose the graph into spanning forests (see Section 2.2).

An important property of an MA ordering is that it identifies a minimum cut between some two vertices, which are specified by the ordering. Based on this, we can solve the minimum cut problem of a graph in $\mathrm{O}(mn + n^2 \log n)$ time [70,76], where $n$ and $m$ are the numbers of vertices and edges, respectively. This is an improvement over the conventional minimum cut algorithms which execute the computation of maximum flows $n$ times [1,37]. The idea is then extended to a fundamental operation of a graph, edge splitting [71,78]. Based on the new edge splitting algorithm, the edge-connectivity augmentation problem can also be solved efficiently. The edge-connectivity augmentation problem asks to add to a given graph the smallest number of new edges so that the edge-connectivity of the resulting graph is increased to a target $k$. This problem has important applications such as the network construction problem [84], the rigidity problem in grid frameworks [3,27], the data security problem [29,56] and the rectangular dual graph problem in floor-planning [88].

The edge-connectivity and vertex-connectivity augmentation problems were first studied in 1976 by Eswaran and Tarjan [15] and Plesnil [82], and both problems were shown to be polynomially solvable for $k = 2$. For general $k$, Watanabe and Nakamura [89] established in 1987 a min–max theorem for the edge-connectivity augmentation problem, based on which they gave an $\mathrm{O}(k^2(kn + m)n^4)$ time algorithm. Afterwards, Frank [17] gave a unified approach to various edge-connectivity augmentation problems by making use of the edge-splitting theorems of Lovász [62,63] and Mader [64,65]. Then Nagamochi and Ibaraki [75] proposed an $\mathrm{O}((nm + n^2 \log n) \log n)$ time algorithm for the edge-connectivity augmentation problem for a given target $k$, by combining their minimum cut algorithm and the approach of Frank. If the graph under consideration is weighted by real numbers, this algorithm can be further simplified and can be extended to solve the edge-connectivity augmentation problem for the entire range of targets $k$ in $\mathrm{O}(nm + n^2 \log n)$ time [75].

This paper first defines the MA ordering and reviews its properties in Section 2. Then Section 3 describes the new minimum cut algorithm and its application to edge splitting. In Sections 4 and 5, we show how to extend the minimum cut algorithm to solve the edge-connectivity augmentation problem for integer weighted and real-weighted graphs, respectively. Recent results on some other augmentation problems are briefly surveyed in Section 6.

## 2. Preliminaries

### 2.1. Definitions

Let $G = (V, E)$ be an undirected graph with a vertex set $V$ and an edge set $E$. The vertex set and the edge set of a graph $G$ may be denoted by $V[G]$ and $E[G]$,

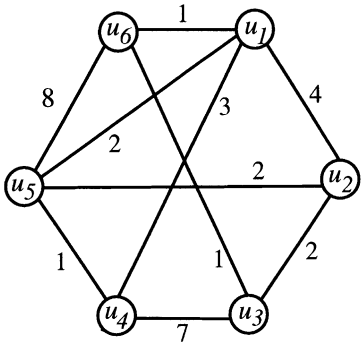


Fig. 1. An integer-weighted graph $G=(V,E)$.

respectively. A singleton set $\{x\}$ is sometimes written as $x$. The notation $\subseteq$ (resp., $\subset$) denotes the set inclusion (resp., proper set inclusion). For two nonempty and disjoint subsets $X, Y \subseteq V$, let $E_G(X,Y)$ denote the set of edges between $X$ and $Y$, and $d_G(X,Y)$ denote its cardinality $|E_G(X,Y)|$. In particular, they are also written as $E_G(X)$ and $d_G(X)$, respectively, if $Y = V - X$. If $a=(u,v)$ is an edge of $G$, then $u$ (resp., $v$) is a neighbor of $v$ (resp., $u$) and $u, v$ are said to be the end vertices of $a$. Edges with the same pair of end vertices are called *multiple edges*. A graph is called a *multigraph* if it is allowed to have multiple edges; *simple* otherwise. We denote

$$n=|V|, \quad e=|E|, \quad m=|\{(x,y)\mid d_G(x,y)\geqslant 1\}|.$$

The input size of a multigraph $G=(V,E)$ can be measured by $n$ and $e$. However, it can also be represented by an edge-weighted graph having integer multiplicity $d_G(u,v)$ as the weight of edge $(u,v)$. In this case, the input size if $O(n+m)$ (under the assumption that the logarithm of the maximum weight is constant), and this measure will be used in the rest of this paper. For example, Fig. 1 shows an integer weighted graph, which can be viewed as a multigraph having the multiplicity equal to its edge weight. Throughout this paper, we understand that a graph is a multigraph, unless otherwise specified, except that, in Section 5, we deal with a graph whose edges are weighted by real numbers. Such a graph will be called a *real weighted graph* to distinguish it from an *integer weighted graph.*

We call $E_G(X)$ (or $X$), satisfying $\emptyset \neq X \subset V$, *a cut*, and define its value by $d_G(X)$. A cut $X$ separates vertices $u$ and $v$ if $u \in X$ and $v \notin X$ (or $u \notin X$ and $v \in X$). It is known that the cut function $d_G$ always satisfies the following *submodular* inequality:

$$d_G(X)+d_G(Y)\geqslant d_G(X\cap Y)+d_G(X\cup Y), \tag{1}$$

where $d_G(\emptyset)=0$ is assumed for convenience. A graph is called *k-edge-connected* if the value of any cut is at least $k$. The minimum value among all the cuts that separate two vertices $u$ and $v$ is called *the local edge-connectivity* between $u$ and $v$, and is denoted by $\lambda_G(u,v)$. By the well-known theorem of Menger, $\lambda_G(u,v)$ is equal to the maximum number of edge disjoint paths between $u$ and $v$ in $G$ (e.g., [1,37]). The minimum value among all the cuts in $G$ is called its *edge-connectivity*. Finally, for a subset $X \subseteq V$, define the *inner edge-connectivity* of $X$ by

$$\lambda_G(X) = \min\{d_G(X') \mid \emptyset \neq X' \subset X\}.$$

In particular, $\lambda_G(V)$ is equivalent to the edge-connectivity of $G$.

For a connected graph $G=(V,E)$, a subset $S \subset V$ is called a *vertex-cut* if $G-S$ has at least two connected components, and $G$ is called *k-vertex-connected* if $|V| \geqslant k+1$ and there is no vertex-cut $S$ with size $k-1$. The maximum number of vertex-disjoint paths from $u$ to $v$ is called *the local vertex-connectivity* between $u$ and $v$, and is denoted by $\kappa_G(u,v)$ (if $u$ and $v$ are not adjacent, then $\kappa_G(u,v)$ is equal to the minimum size of vertex-cuts separating $u$ and $v$).

### 2.2. MA ordering

For any pair of vertices $u,v \in V$ in a graph $G$, we can compute the local edge-connectivity $\lambda_G(u,v)$ by using the conventional maximum flow algorithm (e.g., [1,37,84]). However, the local edge-connectivity $\lambda_G(u,v)$ for some pair $u,v \in V$ (which are specified by the algorithm) can be computed by a significantly simpler method. An ordering $v_1, v_2, \ldots, v_n$ of vertices in $G$ is called an *MA ordering* if it satisfies

$$d_G(\{v_1, v_2, \ldots, v_i\}, v_{i+1}) \geqslant d_G(\{v_1, v_2, \ldots, v_i\}, v_j), \quad 1 \leqslant i < j \leqslant n.$$

Such an ordering can be found by choosing an arbitrary vertex $v_1$, and choosing a vertex $u \in V - \{v_1, \ldots, v_i\}$ that has the largest number of edges between $\{v_1, \ldots, v_i\}$ and $u$ as the $(i+1)$th vertex $v_{i+1}$ after choosing the first $i$ vertices $v_1, \ldots, v_i$. For example, an MA ordering of the graph $G$ in Fig. 1 is obtained as $v_1 = u_1, v_2 = u_2, v_3 = u_5, v_4 = u_6, v_5 = u_4$ and $v_6 = u_3$ (see Fig. 2).

By using the data structure of Fibonacci heap [23], an MA ordering starting from an arbitrarily chosen vertex $v_1$ can be obtained in $O(n+e)$ or in $O(m + n\log n)$ time [70]. The following property of an MA ordering is the starting point of the rest of development.

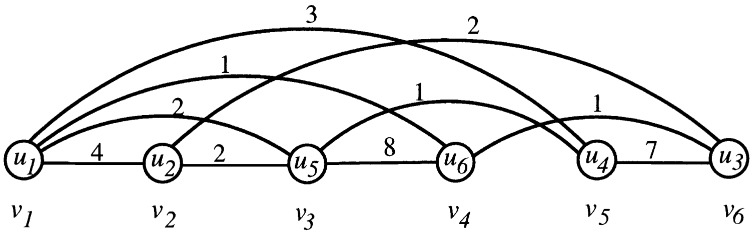


Fig. 2. An MA ordering for the graph $G$ in Fig. 1.

**Theorem 2.1** (Nagamochi and Ibaraki [70]). *For a graph* $G \doteq (V,E)$, *let* $x = v_{n-1}$ *and* $v_n$ *be the last two vertices in an MA ordering. Then* (i) $\lambda_G(x, v_n) = d_G(v_n)$.

(ii) $\kappa_G(x, v_n) = d_G(v_n)$ *if* $G$ *is simple and unweighted.*

(The above results are originally proved for the case where $x$ is the vertex $v_p$ with the largest index $p$ which is adjacent to $v_n$. However, this case implies the theorem because adding an edge between $v_{n-1}$ and $v_n$ preserves the MA ordering.)

Theorem 2.1(i) says that $X = \{v_n\}$ is a minimum cut that separates $v_{n-1}$ and $v_n$. We first outline below a proof of this theorem when $G$ is a multigraph before giving a short proof for $G$ with edges weighted by real numbers.

It is trivial from the definition to see $\lambda_G(v_{n-1}, v_n) \leqslant d_G(v_n)$.

To show the converse, $\lambda_G(v_{n-1}, v_n) \geqslant d_G(v_n)$, we consider the following decomposition of the edge set $E$ of $G$. First choose an arbitrary maximal forest $F_1 \subseteq E$ in $G$, then choose a maximal spanning forest $F_2 \subseteq E - F_1$ in $G - F_1$, where $G - F_1$ is a brief notation of the graph $(V, E - F_1)$. Similarly, let $F_i$ be a maximal spanning forest in $G - (F_1 \cup \cdots \cup F_{i-1})$ for $i = 3, 4, \ldots$ . An MA ordering actually provides such a decomposition. For each $i = 2, \ldots, n$, consider the set $E_G(\{v_1, \ldots, v_{i-1}\}, v_i)$ of edges between $\{v_1, \ldots, v_{i-1}\}$ and $v_i$, and let $e_{i,k} \in E_G(\{v_1, \ldots, v_{i-1}\}, v_i)$ be the edge that appears as the $k$th edge when the edges in $E_G(\{v_1, \ldots, v_{i-1}\}, v_i)$ are arranged in the order $e_{i,1} = (v_{j_1}, v_i), e_{i,2} = (v_{j2}, v_i), \ldots, e_{i,p} = (v_{j_p}, v_i)$, where $1 \leqslant j_1 \leqslant j_2 \leqslant \cdots \leqslant j_p$ holds. By letting

$$F_k = \{e_{2,k}, e_{3,k}, \ldots, e_{n,k}\}, \quad k = 1, 2, \ldots, |E| \tag{2}$$

(some of $e_{i,k}$ may be void), we have a partition $(F_1, \ldots, F_{|E|})$ of $E$. Then it is not difficult to see from the definition of an MA ordering that $(V, F_i)$ is a maximal spanning forest in $G - (F_1 \cup F_2 \cup \cdots \cup F_{i-1})$. Notice that exactly one edge $e_{n,k}$ from each $F_k$, $k = 1, \ldots, d_G(v_n)$ is incident to the last vertex $v_n$. Now suppose that there is an edge between $v_{n-1}, v_n$ (otherwise we can add such an edge without destroying the MA ordering). Then $v_{n-1}$ and $v_n$ are connected in $F_{d_G(v_n)}$ by edge $e_{n, d_G(v_n)} = (v_{n-1}, v_n)$. Thus, by the maximality of the forests (as edge sets), $v_{n-1}$ and $v_n$ are connected (by a path) in each $F_k$ with $k < d_G(v_n)$. Hence, there are at least $d_G(v_n)$ edge-disjoint paths between $v_{n-1}$ and $v_n$, implying $\lambda_G(v_{n-1}, v_n) \geqslant d_G(v_n)$.

For example, we have such spanning forests $F_1, \ldots, F_{10}$ in the MA ordering in Fig. 2, where

$$\begin{aligned}
&F_1 = \{(u_2,u_1),(u_5,u_1),(u_6,u_1),(u_4,u_1),(u_3,u_2)\},\\
&F_2 = \{(u_2,u_1),(u_5,u_1),(u_6,u_5),(u_4,u_1),(u_3,u_2)\},\\
&F_3 = \{(u_2,u_1),(u_5,u_2),(u_6,u_5),(u_4,u_1),(u_3,u_6)\},\\
&F_4 = \{(u_2,u_1),(u_5,u_2),(u_6,u_5),(u_4,u_5),(u_3,u_4)\},\\
&F_5 = \{(u_6,u_5),(u_3,u_4)\},\\
&F_6 = \{(u_6,u_5),(u_3,u_4)\},\\
&F_7 = \{(u_6,u_5),(u_3,u_4)\},\\
&F_8 = \{(u_6,u_5),(u_3,u_4)\},\\
&F_9 = \{(u_6,u_5),(u_3,u_4)\},\\
&F_{10} = \{(u_3,u_4)\}.
\end{aligned}$$

An MA ordering has the following hierarchical structure. For the above forests $F_1, F_2, \ldots,$ consider the spanning subgraph $G_k = (V, F_1 \cup F_2 \cup \cdots \cup F_k)$, $k = 1, 2, \ldots$ . We easily see that the MA ordering $v_1, \ldots, v_n$ for $G$ remains to be an MA ordering for all $G_k$. Therefore, it holds $\lambda_{G_k}(v_{n-1}, v_n) = d_{G_k}(v_n) = k$ for $1 \leqslant k \leqslant d_G(v_N)$.

Although the above proof works when $G$ is a multigraph or edge weighted by integral (or rational) numbers, it is shown [70] that Theorem 2.1(i) remains valid even for real edge weights using some technical argument by approximating real numbers with rational numbers. The correctness of this theorem has been proved by several researchers [19,24,70,76,77,86]. To complete the proof of Theorem 2.1(i), we describe a simple proof for the case of real weights, which is due to Frank [19] (the proof is also found in [12]). We proceed by an induction on the numbers of vertices. The theorem is true for $|V| = 2$. Let $n = |V|$ for a given graph $G = (V, E)$, and assume that the theorem holds for all graphs which have $n'(< n)$ vertices. Consider an MA ordering $v_1, v_2, \ldots, v_{n-2}, v_{n-1}, v_n$ in $G$. As observed in the above, we can assume that the last two vertices are not adjacent, and $d_G(\{v_1, \ldots, v_{n-2}\}, v_{n-1}) = d_G(v_{n-1})$ and $d_G(\{v_1, \ldots, v_{n-2}\}, v_n) = d_G(v_n)$ holds. Notice that the ordering $v_1, v_2, \ldots, v_{n-2}, v_{n-1}$ is an MA ordering in the graph $G - v_n$ obtained from $G$ by deleting $v_n$, and hence by inductive hypothesis $\lambda_{G-v_n}(v_{n-2}, v_{n-1}) = d_{G-v_n}(v_{n-1})(= d_G(v_{n-1}))$. Similarly, we have $\lambda_{G-v_{n-1}}(v_{n-2}, v_n) = d_{G-v_{n-1}}(v_n)(= d_G(v_n))$ since $v_1, v_2, \ldots, v_{n-2}, v_n$ is an MA ordering in $G - v_{n-1}$. Therefore, we obtain $\lambda_G(v_{n-1}, v_n) \geqslant \min\{\lambda_{G-v_n}(v_{n-2}, v_{n-1}), \lambda_{G-v_{n-1}}(v_{n-2}, v_n)\}$ $= \min\{d_G(v_{n-1}), d_G(v_n)\} = d_G(v_n)$ (by the choice of $v_{n-1}$ in the MA ordering). This proves Theorem 2.1(i).

In fact, we can actually compute the maximum flow with flow value $d_G(v_n)$ between $v_{n-1}$ and $v_n$ in time complexity $O(m \log n)$ by using the hierarchical structure of MA orderings [77]. Based on the fact that all minimum cuts separating $v_{n-1}$ and $v_n$ can be obtained in linear time from the maximum flow between them [81], all minimum cuts with value $\lambda(G)$ and the corresponding *cactus structure* (introduced by Dinits et al. [13]), which is a compact representation of all minimum cuts in $G$, can be computed in $O(nm \log n)$ time without relying on the conventional maximum flow algorithm [79].

An MA ordering is also used to find a sparse spanning subgraph of a given graph while preserving the vertex and edge-connectivities of the original graph $G$ [20,69].

**Theorem 2.2.** *For an unweighted multigraph* $G = (V, E)$, *let a set of forests* $F_1, F_2, \ldots, F_{|E|}$ *be the partition of E obtained from an MA ordering by* (2), *where* $F_i = F_{i+1} = \cdots = F_{|E|} = \emptyset$ *possibly holds for some i. Let* $G_k = (V, F_1 \cup F_2 \cup \cdots \cup F_k)$, *for* $k = 1, 2, \ldots, |E|$. *Then each* $G_k$ *has at most* $k(|V| - 1)$ *edges and satisfies*

(i) $\lambda_{G_k}(u, v) \geqslant \min\{\lambda_G(u, v), k\}$ *for all* $u, v \in V$,

(ii) $\kappa_{G_k}(u, v) \geqslant \min\{\kappa_G(u, v), k\}$ *for all* $u, v \in V$ *if G is simple.*

Since the above decomposition of $G$ into forests $F_1, \ldots, F_{|E|}$ can be found in $O(m + n \log n)$ time, such $G_k$ is widely used as a fast preprocessing for sparsifying a given graph $G$, in order to reduce the time complexity of many graph connectivity algorithms (see [25,28,31,58,66] for its applications). A graph search for finding such partition of $E$ in Theorem 2.2 is studied by Cheriyan et al. [10].

## 3. Computing a minimum cut and an edge splitting

### 3.1. A minimum cut algorithm

Based on Theorem 2.1(i), we can compute a minimum cut of a given graph $G=(V,E)$ as follows [70]:

```
Algorithm MIN-CUT
Input: A graph $G = (V,E)$.
Output: A minimum cut $X$ in $G$.
Step 1: Let $G_1 := G$ and $i := 1$.
Step 2: while $i < n$ do
            Compute the local edge-connectivity $\lambda_{G_i}(u_i, v_i) = d_G(v_i)$ for the last two
            vertices $u_i, v_i \in G_i$, in an MA ordering of $G_i$ (where $v_i$ is assumed to be
            the last vertex), and contract vertices $u_i$, $v_i$ into a single vertex, denoting
            the resulting graph by $G_{i+1}$. Let $i := i + 1$.
        end /* while */
Step 3: Find $i = i^*$ that minimizes $\lambda_{G_i}(u_i, v_i)$ among all $i = 1, 2, \ldots, n-1$. Then
        output the set of vertices $X$ contracted to $v_{i^*}$ before obtaining $G_{i^*}$.
```

It is not difficult to see that

$$\lambda(G) = \min\{\lambda_{G_i}(u_i, v_i) \mid i = 1, 2, \ldots, n-1\} \tag{3}$$

holds, because, for each $i$, either a minimum cut separating $u_i$ and $v_i$ in $G_i$ is also a minimum cut of $G_i$, or a minimum cut of $G_i$ is given as a minimum cut of $G_{i+1}$. If the $i = i^*$ that attains the minimum of (3) is identified, then a minimum cut $X \subset V$ of $G$ is obtained as the set of all the vertices contracted into the vertex $v_{i^*}$ (i.e., $d_{G_{i^*}}(v_{i^*}) = d_G(X)$ holds). The running time of this minimum cut algorithm is $O(n(m + n\log n))$.

**Theorem 3.1.** *For a given graph $G=(V,E)$, algorithm MIN-CUT outputs a minimum cut of $G$ in $O(nm + n^2 \log n)$ time.*

In this decade, there has been a significant progress in the study of how to compute a minimum cut of a graph, from both practical and theoretical view points. Beside the above MIN-CUT, the following new algorithms may be worth mentioning.

(a) For a digraph with root $s$, we can consider a minimum cut that has a minimum number of arcs from $X$ to $V - X$ over all $X$ with $s \in X \subset V$. Such a minimum cut $X$ can be found in $O(\lambda m \log(n^2/m))$ time due to Gabow's matroidal approach [25], where $\lambda$ is the value of a minimum cut. This algorithm can also be used to compute a minimum cut in an undirected graph.

(b) For a real weighted digraph with root $s$, a minimum cut can be computed in $O(mn \log(n^2/m))$ time, due to Hao and Orlin's maximum flow algorithm [30]. Similarly to (a), this algorithm can also be used as an $O(mn \log(n^2/m))$ time minimum cut algorithm for an undirected graph.

(c) For a weighted undirected graph, all minimum cuts can be computed in $\tilde{O}(n^2)$ time by a randomized algorithm by Karger and Stein [60,61]. The running time was reduced to almost linear by Karger [57]. Also an NC algorithm was found for this problem [59].

Practical performance of these algorithms (including the above MIN-CUT) has been extensively and systematically studied in [8]. According to this report, a heuristic proposed by Padberg and Rinaldi [80] for reducing a given graph is effective in practice. The algorithm by Hao and Orlin, followed by MIN-CUT, is practically most efficient for many benchmark graphs.

Recently, extensions of algorithm MIN-CUT to the minimization of the class of symmetric submodular functions (and a slightly wider class of functions, called posi-modular and sub-modular functions) were discussed in [72,73,83,85].

Now suppose that, given a graph $G'=(V',E')$, a designated vertex $s\in V'$ and $k\geqslant 0$, we want to test whether the inner edge-connectivity satisfies $\lambda_{G'}(V'-s)\geqslant k$ or not. This can be done by slightly modifying MIN-CUT. The key point here is to compute an MA ordering starting from $v_1=s$.

**Algorithm CONTRACT**

```
Input: A graph $G'=(V',E')$, a designated vertex $s\in V'$ and a real $k\geqslant 0$.
Output: Yes if $\lambda_{G'}(V'-s)\geqslant k$ holds; otherwise No.
Step 1: If there is a vertex $v\in V'-s$ with $d_{G'}(v)<k$, then halt after outputting No.
Step 2: $H:=G'$;
  while $|V[H]|\geqslant 4$ do
    Find a pair of vertices $v,w\in V[H]-s$ such that $\lambda_H(v,w)\geqslant k$ (by
    applying an MA ordering). Then contract them into a single vertex $x^*$,
    and denote the resulting graph also by $H$. Let $X^*\subset V'$ denote the set of
    all vertices contracted into $x^*$ so far. If $d_H(x^*)<k$, then halt after
    outputting No (since $d_H(x^*)=d_{G'}(X^*)<k$ is detected).
  end /* while */
Step 3: /* $H$ has at most three vertices */ Halt after outputting Yes.
```

The correctness of Step 1 is immediate. In Step 2, before starting the while loop in each iteration, we see by induction that graph $H$ satisfies $d_H(u)\geqslant k$ for all $u\in V[H]-s$. Hence a pair of $v,w\in V[H]-s$ with $\lambda_H(v,w)\geqslant k$ can be found as the last two vertices in an MA ordering with $v_1=s$ (we start from $v_1=s$ to avoid the case in which vertex $s$ is used as one of the contracted vertices). By theorem 2.1(i), it holds $\lambda_H(v,w)=d_H(w)\geqslant k$. Since this tells that no cut with value less than $k$ separates $v$ and $w$, all the cuts with values less than $k$ (if such cuts exist) remain in $H$ after contracting $v$ and $w$. By induction, this shows that CONTRACT runs correctly. Its running time is $O(nm+n^2\log n)$.

### 3.2. Edge-splitting theorem

The operation of an *edge splitting* in a graph $G'=(V',E')$ at $s\in V'$ is to replace two edges $(u,s)$ and $(s,w)$ incident to a vertex $s$ with a single edge $(u,w)$. For a real weighted graph $G'=(V',E')$, the definition is extended as follows. For two vertices $u,v\in V'-s$ (possibly $u=v$) and a nonnegative real $\delta\leqslant\min\{d_{G'}(s,u),d_{G'}(s,v)\}$, we decrease weights $d_{G'}(s,u)$ and $d_{G'}(s,v)$, respectively, by $\delta$, and increase weight

$d_{G'}(u,v)$ by $\delta$ (after introducing a new edge $(u,v)$ of zero weight if there was no edge $(u,v)$). After splitting by $\delta$, the value of a cut in $G'$ either remains unchanged or decreases by $2\delta$. Clearly the original edge-splitting for an integer weighted graph can be regarded as an extended edge-splitting with $\delta = 1$.

Let the given graph $G' = (V', E')$ satisfy $\lambda_{G'}(V' - s) \geqslant k$. Then splitting edges $(s,u)$ and $(s,v)$ by $\delta$ is called *$(k,s)$-feasible* if the resulting graph $G''$ also satisfies $\lambda_{G''}(V' - s) \geqslant k$. The next theorem is due to Lovász (see also [17] for a different proof).

**Theorem 3.2** (Lovász [62,63]). (a) *Let $G' = (V', E')$ be a graph, $s \in V'$ be a designated vertex with even $d_{G'}(s)$, and $k$ be an integer with $2 \leqslant k \leqslant \lambda_{G'}(V' - s)$. Then for any neighbor $u$ of $s$, there are a neighbor $v$ of $s$ and an integer $\delta \geqslant 1$ such that splitting $(s,u)$ and $(s,v)$ by $\delta$ is $(k,s)$-feasible.*

(b) *Let $G' = (V', E')$ be a real weighted graph, $s \in V'$ be a designated vertex, and $k$ be a real number with $0 \leqslant k \leqslant \lambda_{G'}(V' - s)$. Then for any neighbor $u$ of $s$, there are a neighbor $v$ of $s$ and a $\delta > 0$ such that splitting $(s,u)$ and $(s,v)$ by $\delta$ is $(k,s)$-feasible, and, after this splitting, the pair $(s,u)$ and $(s,v)$ is no longer splittable.*

A sequence of splitting of edges incident to a designated vertex $s$ is called a *complete* splitting if no edge is incident to $s$ after the splittings. By repeatedly applying Theorem 3.2(a), we can obtain a complete $(k,s)$-feasible splitting at a vertex $s$ with even $d_{G'}(s)$. It is shown in [71,78] that such a complete $(k,s)$-feasible splitting can be obtained in $O((nm + n^2 \log n) \log n)$ time by applying a modification of algorithm CONTRACT (which will be called AUGMENT, and will be described in Section 4.2) $O(\log n)$ times. The description of these edge-splitting algorithms will be given in Section 4.2.

There is the corresponding edge-splitting theorem in a digraph [65], where splitting two directed edges $(u,s)$ and $(s,v)$, one has head $s$ and the other has tail $s$, means replacing them with a single directed edge $(u,v)$ which tail $u$ and head $v$. Other extensions of these edge splitting theorems have been studied in [2–4,14,27,40,46,52,64,68].

## 4. Edge-connectivity augmentation problem for a target *k*

In this section, we first describe the approach by Frank for solving the edge-connectivity augmentation problem, and then explain in the second subsection how to implement his algorithm to make it run efficiently. This is based on the efficient edge splitting algorithm to be described in the last subsection.

### 4.1. Edge-connectivity augmentation

Given a graph $G$ and an integer $k \geqslant 2$, called a *target*, we consider the problem of finding the smallest number of edges to be added to $G$ to obtain a $k$-edge-connected graph, where we allow multiple edges in the resulting graph. A family $\mathscr{X}$ of mutually disjoint subsets of $V$ is called a *subpartition* of $V$. The next min–max result, which was a cornerstone in the study of connectivity augmentation problems, is due to Watanabe and Nakamura [89].

**Theorem 4.1** (Watanabe and Nakamura [89]). *For a graph* $G=(V,E)$ *and an integer* $k \geqslant \max\{\lambda_G(V)+1, 2\}$, *let*

$$\alpha_k(G) = \max \sum_{X \in \mathscr{X}} (k - d_G(X)), \tag{4}$$

*where the maximum is taken over all nonempty subpartitions* $\mathscr{X}$ *of* $V$. *Then the minimum number of edges to be added to make* $G$ $k$*-edge-connected is equal to* $\lceil \alpha_k(G)/2 \rceil$.

Now we prove this theorem by following the argument of Frank [17], which also provides an efficient algorithm for the edge-connectivity augmentation problem. First, let a subpartition $\mathscr{X}$ realize the maximum of (4) (here $k - d_G(X) > 0$ can be assumed for all $X \in \mathscr{X}$, since any cut $X$ with $k - d_G(X) \leqslant 0$ can be discarded from $\mathscr{X}$). Then $\lceil \alpha_k(G)/2 \rceil$ edges are necessary to made $G$ $k$-edge-connected, because each $k - d_G(X)$ in the summation (4) shows the deficiency for the cut $X$ with respect to $k$, and adding one edge can reduce the deficiency of at most two distinct cuts in $\mathscr{X}$ by 1, respectively.

Thus, the crucial part is to prove that $\lceil \alpha_k(G)/2 \rceil$ edges are sufficient to make $G$ $k$-edge-connected. For this, we follow the algorithmic proof by Frank [17], which is based on Lovász's edge-splitting theorem (Theorem 3.1).

```
Algorithm INCREASE
Input: A graph $G=(V,E)$ and an integer $k \geqslant \max\{\lambda_G(V)+1, 2\}$.
Output: A $k$-edge connected graph $G_k^*$ optimally augmented from $G$.
  Step 1: Add to $G=(V,E)$ a new vertex $s$ together with edges of integer
      weights between $s$ and some vertices in $V$ so that the resulting graph
      $G_k' = (V' = V \cup \{s\}, E')$ satisfies the next conditions (i) and (ii):
        Optimality conditions:
        (i) $\lambda_{G_k'}(V'-s) \geqslant k$ (i.e., $\lambda_{G_k'}(x,y) \geqslant k$ for all $x, y \in V$).
        (ii) The weight $d_{G_k'}(s,v)$ of each edge $(s,v)$ incident to $s$ is minimal in
            the sense that decreasing weight $d_{G_k'}(s,v)$ by any amount $\varepsilon > 0$
            violates (i).
      If the degree $d_{G_k'}(s)$ is odd, then choose an arbitrary vertex $v_1 \in V$ and
      increase the weight of edge $(s, v_1)$ by 1.
  Step 2: Let $G_k'$ be the graph obtained in Step 1. Now $d_{G_k'}(s)$ is even, and it
      holds $2 \leqslant k \leqslant \lambda_{G_k'}(V'-s)$. By Lovász's theorem, there is a complete $(k,s)$-
      feasible edge-splitting at $s$ in $G_k'$. Then output the graph $G_k^*$ obtained by the
      complete splitting, ignoring the isolated vertex $s$.
```

We first note that it is easy to find a set of weighted edges incident to $s$ that satisfies the above optimality conditions. For example, after adding an edge of weight $k$ between $s$ and each vertex in $V$, decrease each of their as long as (i) holds. However, we shall present in the next subsection a more systematic efficient method.

To execute Step 2, recall that Lovász's theorem says that the output graph $G_k^*$ is $k$-edge connected, and we can view the split edges as the edges added to the original graph $G$. We now show that the output graph $G_k^*$ is in fact optimally augment from $G$.

The sum of the weights of added edges in $G_k^*$ is the half of the original degree of vertex $s$ in $G'$ (or the degree plus one if it is odd) after Step 1; i.e., $\lceil \frac{1}{2} d_{G_k'}(s) \rceil$. Thus, to prove the optimality of $G_k^*$, it suffices to show that $d_{G_k'}(s) \leqslant \alpha_k(G)$. For this, we introduce some terminology. In a graph $G' = (V \cup \{s\}, E')$, a cut $X \subset V$ is called $(k,s)$-*semi-critical* if

$$d_{G'}(s,X) > 0, \quad k \leqslant d_{G'}(X) \leqslant k+1 \quad \text{and} \quad \lambda_{G'}(X) \geqslant k.$$

A $(k,s)$-semi-critical cut $X \subset V$ is called $(k,s)$-*critical* if $d_{G'}(X) = k$. A family $\mathscr{X} = \{X_1, X_2, \ldots, X_p\}$ of mutually disjoint subsets $X_i \subset V$ is called a *subpartition* (possibly $\mathscr{X} = \emptyset$). If every neighbor of $s$ belongs to a subset $X_i \in \mathscr{X}$ (i.e., $\sum_{i=1}^{p} d_G(s, X_i) = d_G(s)$), then $\mathscr{X}$ is called a *covering* subpartition. A subpartition $\mathscr{X}$ is called $(k,s)$-*critical* (resp., $(k,s)$-*semi-critical*) if every $X_i \in \mathscr{X}$ is $(k,s)$-critical (resp., $(k,s)$-semi-critical) or $\mathscr{X} = \emptyset$.

Consider the graph $G_k'$ after Step 1 of INCREASE, in which any neighbor $v$ of $s$ is contained in some cut $X \subset V$ with $d_{G_k'}(X) = k$ by optimality condition (ii). Such a cut $X$ is $(k,s)$-critical, since $G_k'$ satisfies $\lambda_{G_k'}(X) \geqslant k$ by optimality condition (i). Thus, any neighbor $v$ of $s$ is contained in a $(k,s)$-critical cut $X_v \subset V$. We then choose a minimal family $\mathscr{X}' \subseteq \{X_v \mid v \text{ is a neighbor of } s \text{ in } G_k'\}$ under the constraint that $\mathscr{X}' = \{X_1, \ldots, X_p\}$ is covering. If $X_i \cap X_j \neq \emptyset$ holds for some $X_i, X_j \in \mathscr{X}'$, then we modify $\mathscr{X}'$ by replacing $X_i$ and $X_j$ with $X_i - X_j$ and $X_j - X_i$. We see that, after this modification, $\mathscr{X}'$ remains covering since properties $d_{G_k'}(X_i - X_j) = d_{G_k'}(X_j - X_i) = k$ and $d_{G_k'}(s, X_i \cap X_j) = 0$ follow from the optimality condition (i) and the submodularity (1) of cut function $d_{G_k'}$ (note that $d_{G_k'}(X_i) = d_{G_k'}(X_j) = k$ by assumption, $d_{G_k'}(X_i - X_j), d_{G_k'}(X_j - X_i) \geqslant k$ by optimality condition (i), and $d_{G_k'}(X_i) + d_{G_k'}(X_j) \geqslant d_{G_k'}(X_i - X_j) + d_{G_k'}(X_j - X_i) + 2d_{G_k'}(X_i \cap X_j, V' - (X_i \cup X_j))$ by (1).) By repeatedly modifying $\mathscr{X}'$ until it contains only mutually disjoint cuts, we obtain a $(k,s)$-critical covering subpartition $\mathscr{X}'$ until it contains only mutually disjoint cuts, we obtain a $(k,s)$-critical covering subpartition $\mathscr{X}'$ in $G_k'$. Denote the resulting $\mathscr{X}'$ by $\mathscr{X}$. Now, optimality condition (ii) is rewritten as the following condition:

**Optimality condition**

(ii′) $G_k'$ has a $(k,s)$-critical covering subpartition $\mathscr{X}$.

For each cut $X \in \mathscr{X}$ in (ii′), $d_{G_k'}(s,X)$ represents the deficiency for cut $X$ with respect to $k$ in $G$. Thus we have

$$d_{G_k'}(s) = \sum_{X \in \mathscr{X}} d_{G_k'}(s,X) = \sum_{X \in \mathscr{X}} (k - d_G(X)) \leqslant \alpha_k(G).$$

This proves the optimality of $G_k^*$ and hence the correctness of Theorem 4.1.

### 4.2. *Efficient implementation of INCREASE*

Now we consider how to execute INCREASE efficiently. For this, we first show that Step 1 of INCREASE can be implemented by using a modification of algorithm

CONTRACT. Let $G' = (V \cup \{s\}, E)$ be the graph obtained from $G$ by adding a vertex $s$, and apply CONTRACT to $G'$ with parameter $k$. CONTRACT halts whenever it finds a cut $X$ with value less than $k$. In this case, we modify CONTRACT so that it continues the execution after increasing the value of the detected cut $X$ up to $k$ by adding weighted edges between $s$ and $X$. More precisely, in Step 1 of CONTRACT, we add to $G'$ an edge $(s, v)$ with weight $d_{G'}(s, v) = k - d_{G'}(v)$ if $d_{G'}(v) < k$ holds. In Step 2, if $d_H(x^*) < k$ holds, then we choose an arbitrary vertex $x' \in X^*$ and increase the weight of edge $(s, x')$ (resp., $(s, x^*)$) by $k - d_H(x^*)$ in $G'$ (resp., in $H$).

Let us call the modified algorithm AUGMENT, which is described as follows:

**Algorithm AUGMENT**

**Input**: A graph $G' = (V, E)$ and a real $k \geqslant 0$.

**Output**: A graph $G'_k$ which satisfies the optimality conditions (i) and (ii′), and a $(k, s)$-critical covering subpartition $\mathscr{X}$ of $G'_k$.

```
Step 1: Let $U = \{u_1, u_2, \ldots, u_p\}$ be the set of vertices $u_i \in V$ such that $d_G(u_i) < k$;
    $V' = V \cup \{s\}$; $E' := E \cup \{(s, u_1), \ldots, (s, u_p)\}$;
    for each $u_i \in U$ do
        $d_{G'}(s, u_i) := k - d_G(u_i)$
    end; /* for */
    Let $G' = (V', E')$ be the resulting edge-weighted graph;
    $\mathscr{X} := \{\{u_1\}, \{u_2\}, \ldots, \{u_p\}\}$; /* possibly $\mathscr{X} = \emptyset$ */
Step 2: $H := G'$;
    while $|V[H]| \geqslant 4$ do /* $d_H(u) \geqslant k$ holds for all vertices $u \in V[H] - s$ */
        Find two vertices $v, w \in V[H] - s$ such that $\lambda_H(v, w) \geqslant k$;
            /* Such $v, w$ can be obtained by applying an MA ordering */
        Contract $v$ and $w$ in $H$ into a single vertex $x^*$, and let $H$ be the resulting
        graph; if $d_H(x^*) < k$ then
            Let $X^* (\subseteq V' - s)$ denote the set of vertices that have been contracted
                into $x^*$ so far;
            Choose an arbitrary vertex $u \in X^*$, and let $d_{G'}(s, u) := d_{G'}(s, u) + k -$
                $d_H(x^*)$ (after letting $E' := E' \cup \{(s, u)\}$ and $d_{G'}(s, u) := 0$, if
            $(s, u) \notin E[G']$); Let $G'$ be the resulting graph;
            Let $H$ denote the graph obtained from $H$ by setting $d_H(s, x^*) :=$
                $d_H(s, x^*) + k - d_H(x^*)$ (after creating edge $(s, x^*)$ in $H$ and letting
                $c_H(s, x^*) := 0$, if $(s, x^*) \notin E[H]$);
            $\mathscr{X} := \mathscr{X} \cup \{X^*\}$, after discarding all $X'$ with $X' \subset X^*$ from $\mathscr{X}$
        end; /* if */
    end /* while */
Step 3: Output $G'_k := G'$ and $\mathscr{X}$.
```

Let $G'_k$ be the graph output by AUGMENT. Then $G'_k$ satisfies $\lambda_{G'}(V' - s) = \lambda_{G'_k}(V) \geqslant k$ (i.e., optimality condition (i)). It is not difficult to see that if a cut $X^*$ with value less than $k$ is found and its value is increased up to $k$ during Step 2 of AUGMENT, then the cut $X^*$ becomes $(k, s)$-critical in $G'$. Thus, each neighbor of $s$ is always contained in some $(k, s)$-critical cut in $G'$ during AUGMENT. This implies that optimality condition (ii′) holds in the output graph $G'_k$. AUGMENT runs in the same time

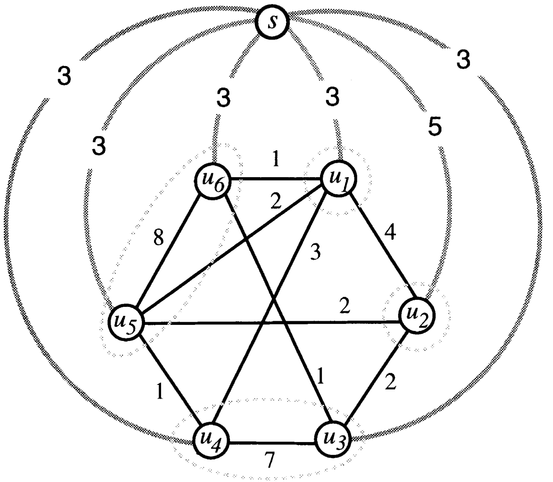


Fig. 3. The graph $G'_k$ obtained from $G$ of Fig. 1 by applying AUGMENT for $k = 13$, where subsets of vertices circled by broken lines form a $(k,s)$-critical covering subpartition $\mathscr{X}$.

complexity $O(nm + n^2 \log n)$ as that of CONTRACT. As an example, Fig. 3 shows the graph $G'_k$ obtained from the graph $G$ in Fig. 1 by applying AUGMENT for target $k = 13$. A $(k,s)$-critical covering subpartition in $G'_k$ is given by $\mathscr{X} = \{\{u_1\}, \{u_2\}, \{u_3, u_4\}, \{u_5, u_6\}\}$.

Now we turn to Step 2 of INCREASE. This step can be executed in $O((nm + n^2 \log n) \log n)$ time by using an algorithm [71,78] for finding a complete $(k,s)$-feasible splitting at $s$. A description of this edge splitting algorithm will be given in the next subsection (as it uses algorithm AUGMENT of this subsection).

It is also interesting to note that, after Step 1 of INCREASE, we already know the optimal value (i.e., the minimum sum of weights of edges to be added to $G'$) to obtain a $k$-edge-connected graph.

### 4.3. Edge-splitting algorithm

In this subsection, we sketch the edge-splitting algorithms proposed in [71,78]. Let $G' = (V' = V \cup \{s\}, E')$ and $k$ satisfy Theorem 3.2 (i.e., $\lambda_{G'}(V' - s) \geqslant k$). Then we split arbitrarily all edges incident to $s$ to obtain a complete splitting (which is not necessarily $(k,s)$-feasible). Let $G^I$ denote the resulting graph and let $B$ be the set of edges created by the complete splitting.

Suppose that Step 2 of algorithm AUGMENT is applied to the resulting graph $G' := G^I$ and the $k$ (where $\mathscr{X}$ is set to be empty) to test whether the above complete splitting at $s$ is $(k,s)$-feasible or not (i.e., $\lambda_{G^I}(V' - s) \geqslant k$ holds or not). If $\lambda_{G^I}(V' - s) \geqslant k$, then Step 2 of AUGMENT outputs $\mathscr{X} = \emptyset$ without finding any cut $X \subset V' - s$ with its value less than $k$. In this case, we conclude that the complete splitting is $(k,s)$-feasible.

On the other hand, if $\lambda_{G^I}(V'-s)<k$, then a cut $X^* \subset V'-s$ with $d_{G^I}(X^*)<k$ is detected in the while-loop of Step 2. For this cut $X^*$, the subgraph $G^I[X^*]$ induced from $G^I$ by $X^*$ must contain at least $\lceil \frac{1}{2}(k-d_{G^I}(X^*))\rceil$ edges in $B$. This is because the original assumption $\lambda_{G'}(V'-s)\geqslant k$ implies that splitting $\delta$ pairs of edges $(s,u)$ and $(s,v)$ in $G'$ decreases cut value $d_{G'}(X^*)\geqslant k$ by $2\delta$ only when $\{u,v\}\subseteq X^*$ holds. We now say that an edge $(u,v)$ in a graph is *hooked up* at a vertex $s$ if we replace the edge $(u,v)$ with two new edges $(s,u)$ and $(s,v)$. We then increase the cut value of $X^*$ at least to $k$ by hooking up $\lceil \frac{1}{2}(k-d_{G^I}(X^*))\rceil$ edges in $B\cap E(G^I[X^*])$ (instead of increasing $d_{G'}(s,u)$ up to $k$ for some $u\in X^*$, as in the original AUGMENT). This modified iteration of the while-loop is repeated while $|V[H]|\geqslant 4$ holds.

Let $G^{II}$ be the graph obtained from $G^I$ by hooking up all the chosen edges for all the cuts $X^*$ with $d_{G^I}(X^*)<k$ found in the above process. As a byproduct of this computation, a $(k,s)$-semi-critical covering subpartition $\mathcal{Y}$ in $G^{II}$ can be found by retaining all the detected cuts $X^*$ and choosing all the maximal ones among them. The detail is described as follows:

```
**Algorithm HOOK-UP**
Input: A graph $G^I=(V'=V\cup\{s\},E^I)$, a designated vertex $s\in V$ with $d_{G^I}(s)=0$,
  and a set $B\subseteq E^I$.
Output: A graph $G^{II}$ obtained from $G^I$ by hooking up some edges in $B$, where $G^{II}$
  satisfies $\lambda_{G^{II}}(V'-s)\geqslant k$, and a $(k,s)$-semi-critical covering subpartition $\mathcal{Y}$ in $G^{II}$.
Step 1: $H:=G^I; \mathcal{Y}:=\emptyset$; $B':=\emptyset$; /* $B'$ denotes the set of edges to be hooked up. */
Step 2: while $|V[H]|\geqslant 4$ do
     Find vertices $v,w\in V[H]-s$ with $\lambda_H(v,w)=d_H(w)\geqslant k$;
       /* Such $v,w$ can be found by an MA ordering. */
     Contract $v$ and $w$ into a single vertex $x^*$ and let $H$ be the resulting graph;
     if $d_H(x^*)<k$ then
       Let $X^*\subseteq V'-s$ be the set of all vertices contracted so far into $x^*$;
       Choose a set $\Delta B\subseteq B$ of arbitrary $\lceil\frac{1}{2}(k-d_{G^I}(X^*))\rceil$ edges in $B\cap E[G^I[X^*]]$;
       $B:=B-\Delta B$; $B':=B'\cup\Delta B$;
       Let $G^I$ denote the graph obtained by hooking up these edges in $\Delta B$ at $s$
         in $G^I$;
       Let $H$ denote the graph obtained by adding new $2\lceil\frac{1}{2}(k-d_{G^I}(X^*)\rceil$ edges
         between $s$ and $x^*$ in $H$;
       $\mathcal{Y}:=\mathcal{Y}\cup\{X^*\}$, after discarding from $\mathcal{Y}$ all $X'\in\mathcal{Y}$ such that $X'\subset X^*$;
     end; /* if */
     end /* while */
Step 3: Output $G^{II}:=G^I$ and $\mathcal{Y}$.
```

For simplicity, we assume for a moment that $k$ is an even integer. We consider how to find a complete $(k,s)$-feasible splitting in the output graph $G^{II}$ based on the information of the output $(k,s)$-semi-critical covering subpartition $\mathcal{Y}$. Clearly, splitting two edges $(s,u)$ and $(s,v)$ is not $(k,s)$-feasible if $u$ and $v$ belong to the same subset $X\in\mathcal{Y}$ since $d_{G^{II}}(X)\leqslant k+1$. We call splitting edges $(s,u)$ and $(s,v)$ *$\mathcal{Y}$-astride* if $u\in X$ and $v\in X'$ hold for distinct $X,X'\in\mathcal{Y}$. It is not difficult to see that $G^{II}$ always admits a complete $\mathcal{Y}$-astride edge-splitting and such splitting can be easily found.

Let $\mathcal{Y}_1 := \mathcal{Y}$ and $G_1^{II} := G^{II}$ initially. Then we split all edges incident to $s$ in $G_1^{II}$ by a complete $\mathcal{Y}_1$-astride edge-splitting, and apply algorithm HOOK-UP to test whether this complete edge-splitting is $(k,s)$-feasible or not in the resulting graph $G_2^{II} := G^{II}$. HOOK-UP obtains a new $(k,s)$-semi-critical covering subpartition $\mathcal{Y}_2$ if the edge-splitting is not $(k,s)$-feasible. We repeat this process until, for some $i > 1$, the complete $\mathcal{Y}_i$-astride edge-splitting becomes $(k,s)$-feasible in $G_i^{II}$. In this iteration, we can prove that $|\mathcal{Y}_{i+1} \leqslant \frac{1}{2}|\mathcal{Y}_i|$ and $|\mathcal{Y}_i| \neq 1$ hold for all $i \geqslant 1$ [78]. Therefore, $\mathcal{Y}_i$ becomes empty after $O(\log n)$ iterations, and a complete $(k,s)$-feasible edge-splitting of the original graph $G' = (V' = V \cup \{s\}, E')$ can be obtained. The entire running time is $O((nm + n^2 \log n) \log n)$.

For an odd integer $k$, we have to be more careful to find a complete $\mathcal{Y}_i$-astride edge-splitting in $G_i^{II}$, which satisfies a certain condition to guarantee the property $|\mathcal{Y}_{i+1}| \leqslant \alpha |\mathcal{Y}_i|$ for some constant $\alpha < 1$ [71]. Another article [78] handles the case of odd $k$ in a slightly different way. In the given graph $G' = (V' = V \cup \{s\}, E')$, it first finds a maximal set $E_k \subset E'$ of edges incident to $s$ such that $G' - E_k$ satisfies $\lambda_{G'-E_k}(V' - s) \geqslant k - 1$ and $d_{G'-E_k}(s)$ is even. (Such $E_k$ can be computed by AUGMENT for integer $k - 1$ after removing all edges incident to $s$). Then by applying the above algorithm for the even integer $k - 1$, we can find a complete $(k-1,s)$-feasible edge-splitting. Now the problem is to increase the edge-connectivity $k-1$ of the resulting graph $G^*$ to $k$ by adding some edges which can be created by splitting appropriate edges in $E_k$. This problem can be solved in linear time after constructing the cactus structure of $G^*$, which provides us the system of all cuts with value $k - 1$.

**Theorem 4.2.** *For a given graph* $G' = (V \cup s, E')$ *and an integer* $k$ *satisfying* $2 \leqslant k \leqslant \lambda_{G'}(V' - s)$, *a complete* $(k,s)$*-feasible edge splitting can be obtained in* $O((nm + n^2 \log n) \log n)$ *time.*

Therefore, by the argument in Section 4.2, we have the next result.

**Theorem 4.3.** *For a given graph* $G = (V,E)$ *and an integer* $k \geqslant \max\{\lambda_G(V) + 1, 2\}$, *algorithm INCREASE optimally augments* $G$ *into* $G_k^*$ *so that the output* $G_k^*$ *is* $k$*-edge-connected. INCREASE runs in* $O((nm + n^2 \log n) \log n)$ *time.*

## 5. Edge-connectivity augmentation for the entire range of targets

In this section, we consider a real weighted graph $G = (V,E)$, where edges are weighted by *nonnegative reals*. As we allow edges of zero weights, $G$ can be assumed to be a complete graph. All the definitions used so far are also generalized to consider this model; e.g., the cut value $d_G(X)$ is defined to be the weight sum of the edges between $X$ and $V - X$, and the edge-connectivity is the minimum value of the cuts in $G$. For such a real weighted graph $G$, we are permitted to increase an edge weight by an arbitrary nonnegative real, and the goal is to minimize the sum of weights to be increased in order to make $G$ $k$-edge-connected, where the target $k$ is a given nonnegative real number. We call this problem the *fractional version* of the edge-connectivity augmentation problem. It is not difficult to see that the problem can be formulated as a linear program, and hence can be solved in polynomial time. However, we present in the sequel of a graph theoretic algorithm, which is more efficient.

For a given target $k \geqslant 0$, let us denote by $\Lambda_G(k)$ the optimal value of the problem (i.e., the sum of added weights), and by $G^*(k)$ an optimally augmented graph. $\Lambda_G(k)$ is called the *edge-connectivity augmentation function*. We shall show that not only $\Lambda_G(k)$ for all $k \geqslant 0$ but also $G^*(k)$ for all $k \geqslant 0$ can be computed in $O(nm + n^2 \log n)$ time by using a compact representation of optimally augmented graphs.

First note that, based on the fractional version of Lovász's theorem (i.e Theorem 3.2(b)), it can be shown that, for a real weighted graph $G' = (V' = V \cup s, E')$ and a real $k \leqslant \lambda_{G'}(V' - s)$, there exists a complete $(k,s)$-feasible edge-splitting at $s$, where size $\delta > 0$ in each edge-splitting is not necessarily integer. Based on this, the fractional version can also be solved by algorithm INCREASE (where we need not round up an odd degree of $s$ to an even degree at the end of Step 1). As remarked at the end of Section 4.2, the optimal value for a given target $k$ is obtained after Step 1 of INCREASE. In other words, it holds $\Lambda_G(k) = d_{G'_k}(s)/2$ for the graph $G'_k$ obtained after Step 1.

### 5.1. Edge-connectivity augmentation function

From the fact that the fractional version of the edge-connectivity augmentation problem is formulated as a linear program, it is clear that edge-connectivity augmentation function $\Lambda_G(k)$ is piecewise linear, convex and increasing over $k \in [0, +\infty)$.

For example, the graph in Fig. 1 has the edge-connectivity augmentation function of Fig. 4. In what follows, we show that the function $\Lambda_G$ can be identified in $O(mn + n^2 \log n)$ time. We already know that, for a fixed target $k$, $\Lambda_G(k)$ can be computed in

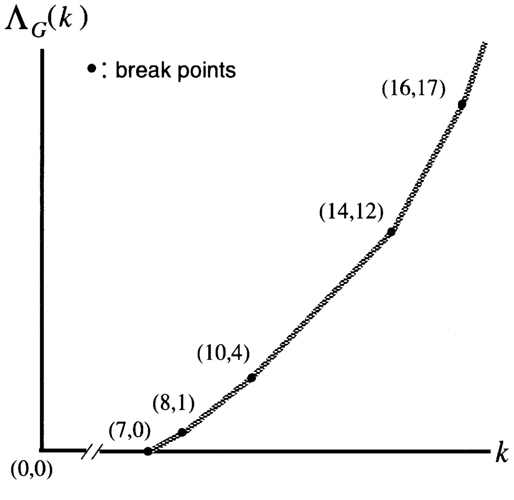


Fig. 4. The edge-connectivity augmentation function of graph $G$ in Fig. 1.

$O(mn+n^2\log n)$ time by AUGMENT (i.e., Step 1 of INCREASE). To compute $\Lambda_G(k)$ over the entire range $k \geqslant 0$, we try to execute AUGMENT simultaneously for all $k$. To realize this computation in finite time and space, we introduce the *ranged graph*.

For two reals $a$ and $b$ with $a < b$, the interval $[a,b]$ is called a *range*, and its *size* $\pi([a,b])$ is defined as $b-a$. Let $R = \{[a_1,b_1],[a_2,b_2],\dots,[a_t,b_t]\}$ be a set of ranges. The size of $R$, denoted by $\pi(R)$, is defined as the sum of all range sizes in $R$:

$$\pi(R) = (b_1 - a_1) + (b_2 - a_2) + \cdots + (b_t - a_t),$$

where $\pi(\emptyset)$ is defined to be 0. For a given real $h$, the *upper h-truncation* of a range $[a,b]$ is defined by

$$[a,b]|^h = \begin{cases} [a,\min\{b,h\}] & \text{if } a < h, \\ \emptyset & \text{otherwise.} \end{cases}$$

Based on this, the upper $h$-truncation of a set $R$ of ranges is defined by

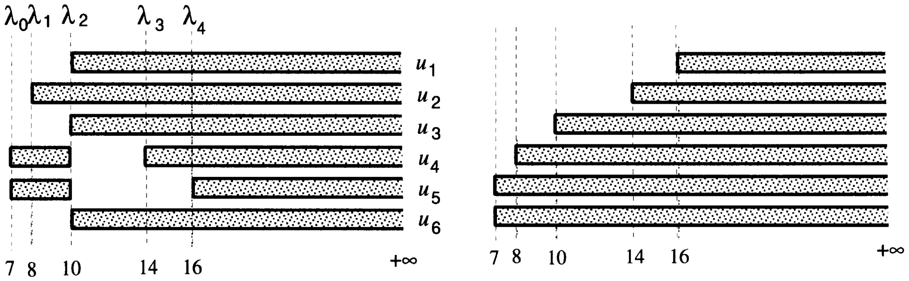
$$R|^h = \{[a_i,b_i]^h \neq \emptyset \mid [a_i,b_i] \in R\}.$$

For example, we have $\{[1,3],[2,5],[4,7]\}|^3 = \{[1,3],[2,3]\}$.

Now we modify algorithm AUGMENT for a given target $k$ (described in Section 4.2) in order to deal with all targets $k$. Let us define the ranged graph $\mathscr{R}(G) = (V \cup \{s\}, E \cup E_s, R)$ for a given weighted graph $G=(V,E)$ as follows. Let $E_s = \{(s,v) \mid v \in V\}$ be the set of edges between $s$ and all vertices in $V$. In $\mathscr{R}(G)$, let each edge in $E$ have the same weight as in $G$, but let each edge $(s,v) \in E_s$ have a set $R_v$ of ranges (i.e., $R = \{R_v \mid v \in V\}$). This $R_v$ is initially set to be $R_v = \{[d_G(v), +\infty]\}$ and will be changed during execution of AUGMENT. Given a real $h \geqslant 0$, we define the weight of each edge $(s,v) \in E_s$ by $\pi(R_v|^h)$ (i.e., the sum of sizes of the upper $h$-truncated ranges in $R_v$). The resulting real weighted graph is denoted by $\mathscr{R}(G)|^h$. In other words, the ranged graph $\mathscr{R}(G)$ represents the real weighted graphs $\mathscr{R}(G)|^h$ for all reals $h \geqslant 0$.

To execute Step 1 of AUGMENT for all targets $k$, we construct the ranged graph $\mathscr{R}(G)$ having a set $R_v = \{[d_G(v), +\infty]\}$ of ranges for each $v \in V$. By definition, graph $\mathscr{R}(G)|^k$ is the same as the weighted graph $G'$ obtained after Step 1 of the original AUGMENT for a target $k$. In Step 2 of AUGMENT for all targets, it is important to choose a pair of vertices $v$ and $w$ for contraction so that this pair $v$ and $w$ is commonly used for all targets $k$; otherwise, we cannot maintain the process of contractions for all targets in a single ranged graph. Fortunately, the existence of such a pair is guaranteed by the hierarchical structure of MA orderings described in Section 2.2. After the contraction of $v$ and $w$, there is a way of updating the range sets $R_v$, $v \in V$ so that $\mathscr{R}(G)|^k$ for any $k \geqslant 0$ is always equivalent to the graph $G'_k$ computed by AUGMENT for a single target $k$. However we omit the details (see [75]). The final ranged graph $\mathscr{R}(G)$ is totally optimal in the following sense:

> **Total optimality condition**: Let $\mathscr{R}(G)$ be the ranged graph for a weighted graph $G$. If the weighted graph $G'_k = \mathscr{R}(G)|^k$ satisfies the optimality condition (i) and (ii') for all nonnegative reals $k$, then $\mathscr{R}(G)$ is called a *totally optimal ranged graph*.



**(a) A set of ranges satisfying the total optimality condition.** **(b) A rearranged set of ranges.**

Fig. 5. A totally optimal ranged graph $\mathscr{R}(G)$ for the graph $G$ in Fig. 1: (a) A set of ranges satisfying the total optimality condition. (b) A rearranged set of ranges.

We can also prove that, for the final ranged graph $\mathscr{R}(G)$ computed by AUGMENT for all targets $k$, the total number of ranges $\sum_{v\in V}|R_v|$ is bounded by $10n\log_2 n$, where $|R_v|$ denotes the number of ranges in $R_v$. Given a totally optimal ranged graph $\mathscr{R}(G)$, we can obtain the optimal value $\Lambda_G(k)$ for a target $k$ by

$$\Lambda_G(k) = d_{G'_k}(s)/2 = \sum_{v\in V} \pi(R_v|^k)/2. \tag{5}$$

For example, Fig. 5(a) shows the range sets $R_{u_i}$ of vertices $u_i$ in a totally optimal ranged graph $\mathscr{R}(G)$ obtained for the graph $G$ in Fig. 1. Fig. 5(b) then shows the range sets obtained from those in Fig. 5(a) by moving some part of ranges to other ranges while keeping the property (5). The edge-connectivity function in Fig. 4 is then immediately obtained from Fig. 5(b). In general, the range sets of a totally optimal ranged graph can be transformed in this way into $n$ ranges [75]. From this, we see that the number of break points in the edge-connectivity augmentation function is at most $n$.

**Theorem 5.1.** *For a real weighted graph $G=(V,E)$, its edge-connectivity augmentation function $\Lambda_G(k)$ for all targets $k \geqslant 0$ can be computed in $O(nm+n^2\log n)$ time.*

### 5.2. Optimal solutions for all targets

We turn to the problem of computing optimally augmented graphs $G^*(k)$ for all $k \geqslant 0$. Again, we do not have to execute Step 2 of INCREASE for all targets $k$ separately (actually it already seems difficult to execute Step 2 directly on the ranged graphs). We show that $G^*(k)$ can be obtained from the totally optimal ranged graph $\mathscr{R}(G)$ computed by AUGMENT of Section 5.1. First let $R=\{R_v \mid v\in V\}$ be the range sets of $\mathscr{R}(G)$, and let $\lambda_i$, $i=0,1,\ldots,q$ be the set of all the distinct end points appearing in ranges $R_v$, $v\in V$, where $\lambda_0<\lambda_1<\cdots<\lambda_q\ (=+\infty)$ is assumed. For example, Fig. 5(a) gives

$$\lambda_0=7,\quad \lambda_1=8,\quad \lambda_2=10,\quad \lambda_3=14,\quad \lambda_4=16\quad \text{and}\quad \lambda_5=+\infty.$$

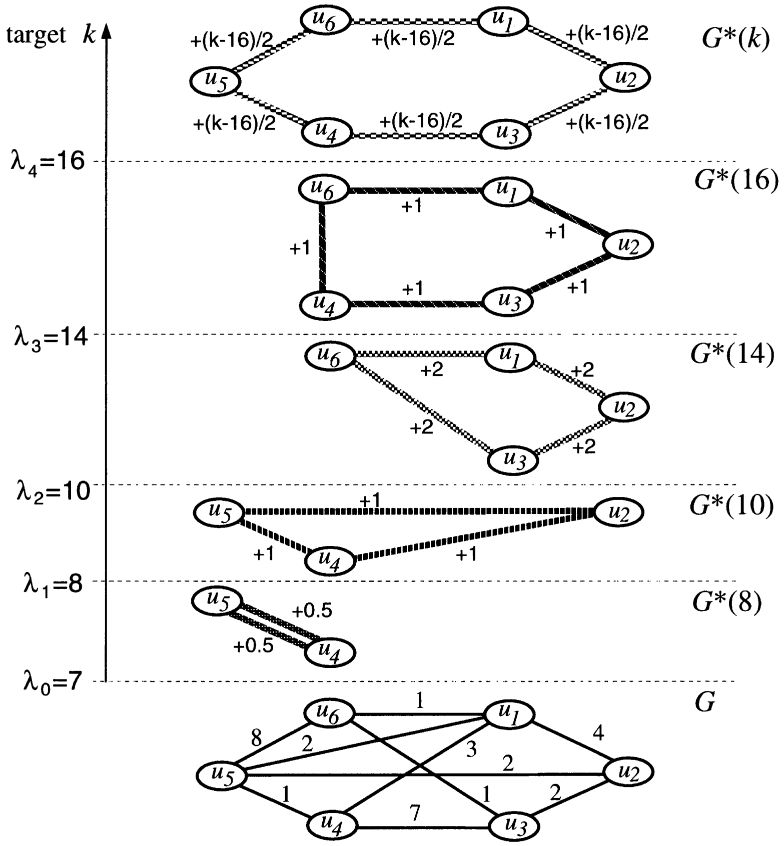


Fig. 6. A representation of optimal solutions $G^*(k)$ for all targets $k \in [0, +\infty)$.

Next consider $q$ ranges $[\lambda_0, \lambda_1], [\lambda_1, \lambda_2], \ldots, [\lambda_{q-1}, \lambda_q]$, and let $X_i$ be the set of vertices $v$ such that one of the ranges in $R_v$ contains $[\lambda_{i-1}, \lambda_i]$. Construct an arbitrary cycle $C_i$ which visits all vertices in $X_i$. For Fig. 5(a), such cycles are chosen as $C_1 = (u_4, u_5)$, $C_2 = (u_2, u_4, u_5)$, $C_3 = (u_1, u_2, u_3, u_6)$, $C_4 = (u_1, u_2, u_3, u_4, u_6)$, $C_5 = (u_1, u_2, u_3, u_4, u_5, u_6)$. See Fig. 6.

We are now ready to construct an optimally augmented graph $G^*(k)$ for any target $k \geqslant 0$. Let $i_k$ be the maximum index such that $\lambda_{i_k} < k$. We increase the edge weights by $(\lambda_i - \lambda_{i-1})/2$ along each cycle $C_i$ for $i = 1, 2, \ldots, i_k$, and by $(k - \lambda_{i_k})/2$ along $C_{i_k+1}$. Obviously, the sum of the increased weights is equal to $\Lambda_G(k)$. It is not trivial to see that the resulting graph is in fact $k$-edge-connected; but can be shown from the total optimality condition [75]. From an analysis of the algorithm, we can also observe that the least number $q$ of such cycles $C_1, \ldots, C_q$ is at most $6n + 4n \log_2 n$ [75].

**Theorem 5.2** (Nagamochi and Ibaraki [75]). *For a real weighted graph* $G = (V, E)$, *the above* $\lambda_0 < \lambda_1 < \cdots < \lambda_q$ *and* $C_1, C_2, \ldots, C_q$ (*which can provide optimally augmented* *graphs* $G^*(k)$ *for all targets* $k \geqslant 0$) *can be obtained in* $\mathrm{O}(nm + n^2 \log n)$ *time*, *where* $q \leqslant 6n + 4n \log_2 n$ *holds*.

Notice that optimally augmented graphs $G^*(k)$ constructed in this way have the monotone structure such that, for $k < k'$, $G^*(k')$ is obtained in increasing the weights of some edges in $G^*(k)$. Such monotone structures of optimal solutions are also observed in some other edge-connectivity augmentation problems [9].

## 6. Other augmentation problems

In this section, we summarize the results on augmentation problems of other types, including problems of increasing vertex-connectivity.

The problems of increasing the edge- or vertex-connectivity by adding the minimum number of edges are extensively surveyed by Frank [18], and here we give only a brief summary, where we first describe some results covered by the survey with the following four categories and then show some recent results:

*Edge-connectivity augmentation of undirected graphs*: After the edge-connectivity augmentation algorithm for an undirected graph proposed by Watanabe and Nakamura [89], the edge-splitting theorem was first used to solve the same problem by Cai and Sun [7]. Frank [17] then refined it by using the theorem of Lovász [63]. Moreover he showed that a more general augmentation problem can be solved polynomially in an undirected graph [17]. The local edge-connectivity augmentation problem asks to find a minimum set $F$ of new edges to be added to a given undirected graph $G$ such that, for each pair of vertices $u$ and $v$, the resulting local edge-connectivity $\lambda_{G+F}(u,v)$ becomes larger than or equal to the target value $r(u,v)$ prescribed for each pair of $u, v \in V$. He proved that the problem can be solved in $\mathrm{O}(n^3 m \log(n^2/m))$ time by applying Mader's edge-splitting theorem [64], a generalization of Lovász's theorem, which preserves the local edge-connectivity in an undirected graph.

*Edge-connectivity augmentation of digraphs*: A digraph is called *k-edge-connected* if it remains strongly connected by removal of any $(k-1)$ edges. The edge-connectivity augmentation problem in digraphs asks to find a minimum set of new directed edges to be added to a given digraph such that the augmented digraph becomes $k$-edge-connected for a prescribed target $k \geqslant 1$. Frank [17] pointed out that the edge-connectivity augmentation problem in a digraph can be solved in $\mathrm{O}(n^3 m \log(n^2/m))$ time by Mader's edge-splitting theorem [65] in digraphs, which preserves the edge-connectivity in a digraph. In [17], the edge-connectivity augmentation problem in digraphs (undirected graphs) is still polynomially solvable even if lower and upper bounds are imposed on degree of each vertex.

One may introduce the local edge-connectivity augmentation problem in digraphs, whose undirected graph version is successfully solved by an edge-splitting theorem. However, there does not exist the corresponding edge-splitting theorem that preserves the local edge-connectivity in a digraph. In fact, the problem of increasing the local edge-connectivity (or the local vertex-connectivity) in a directed graph is NP-hard even if the target values satisfy $r(u,v) \in \{0,1\}$ for all pairs $u, v \in V$ [17].

Given a digraph $G=(V,E)$ with two specified subsets $S,T \subset V$ (which are not necessarily disjoint), Frank and Jordàn [21] first studied the problem of finding a minimum number of new edges directed from $S$ to $T$ to make $G$ *k-edge-connected from S to T* (a digraph is called $k$-edge-connected from $S$ to $T$ if it has $k$ edge disjoint directed paths from every vertex $s \in S$ to every vertex $t \in T$). They gave a min–max formula for the problem, based on which a polynomial time algorithm is obtained (where the algorithm is not combinatorial but rely on the ellipsoid method).

*Vertex-connectivity augmentation of undirected graphs*: The vertex-connectivity augmentation problem in undirected graphs is to increase the vertex-connectivity of a given undirected graph $G$ to a target $k$ by adding a minimum number of new edges. The problem is polynomially solvable for $k=2,3$ and 4, due to [15,35,36,82,90], and [32,33], respectively. For a general $k \geqslant 5$, it is not known whether the vertex-connectivity augmentation problem is NP-hard or not, even if a given graph is $(k-1)$-vertex-connected. As an approximation solution, Jordàn [49] proved that a solution with its absolute error from the optimal value being at most $k-3$ can be found in $O(n^5)$ time. The problem of increasing the local-vertex-connectivity to prescribed target values $r(u,v)$ by a minimum number of edges is shown to be NP-hard in general. Jordàn [48] proved the NP-hardness of the problem in the case where a given graph $G=(V,E)$ is $(n/2)$-vertex-connected and there is a subset $S \subset V$ such that $r(u,v)=(n+2)+1$ for all $u,v \in S$ and $r(u,v)=0$ otherwise. However, the complexity of the problem is not known if target values $r(u,v)$, $u,v \in V$ are independent of $n$ (e.g., the case of $r(u,v) \in \{0,2\}$).

*Vertex-connectivity augmentation of digraphs*: A digraph is called *k-vertex-connected* if it has at least $(k+1)$ vertices and remains strongly connected by removal of any $(k-1)$ vertices. The vertex-connectivity augmentation problem in digraphs is shown to be polynomially solvable by Frank and Jordàn [21]. This result is based on a min–max formula for the problem. They found that the minimum number of new directed edges to make a given digraph $G=(V,E)$ $k$-vertex-connected is given by the maximum of $\sum_i (k-|V-(A_i \cup B_i)|)$ over all families $\{(A_1,B_1),\ldots,(A_p,B_p)\}$ of pairs of disjoint subsets $A_i,B_i \subseteq V$ such that $G$ has no directed edge from $A_i$ to $B_i$ for each $i$ and $A_i \cap A_j = \emptyset$ or $B_i \cap B_j = \emptyset$ for each $i<j$. The algorithm obtained from this theorem again relies on the ellipsoid method. Afterwards, Frank and Jordàn [22] observed that the vertex-connectivity augmentation problem can be directly reduced to the problem of increasing edge-connectivity from $S$ to $T$ in digraphs. However, it remains open to design combinatorial polynomial time algorithms for solving these two problems.

Let us summarize some recent results obtained after the survey by Frank [18] was done.

As already observed in Theorem 4.3, the time complexity for solving the edge-connectivity augmentation problem in undirected graphs is reduced to $O((mn+n\log n)\log n)$ [71,78]. For this problem, efficient randomized algorithms are also proposed [5,6], among which the algorithm by Benczúr and Karger [6] runs in $O(n^2)$ time. By characterizing all graphs $G'_k$ satisfying the optimality conditions (i)–(ii) in Section 4.1, Nagamochi and Ibaraki [74] showed that an optimal solution $F$ that minimizes the number of vertices incident to $F$ over all optimal solutions can be found in $O((mn+n\log n)\log)$ time.

Gabow [26] improved both the running times of the local edge-connectivity augmentation algorithm of undirected graphs and the edge-connectivity augmentation algorithm of digraphs in [17] to $O(n^2 m \log(n^2/m))$.

As to the vertex-connectivity augmentation problem, Jordàn reduced the absolute error of his algorithm to $\lceil (k-1)/2 \rceil$ [50]. By investigating structure of shredders (which are defined as vertex-cuts removal of which creates more than two components), Cheriyan and Thurimella [11] improves the time complexity of Jordàn's algorithm [49] to $O(\min\{k, \sqrt{n}\}k^2n^2 + kn^2 \log n)$ time. Very recently, Ishii and Nagamochi [39] obtained an approximation algorithm for the problem without assuming that a given graph is $(k-1)$-vertex-connected for a target $k$. Given an $\ell$-vertex-connected graph, they proved that a solution with absolute error $(k-\ell)(k-1) + \max\{0, (k-\ell-1)(\ell-3)-1\}(=O((k-\ell)k))$ can be found in $O((k-\ell)(k^2n^2 + k^3n^{3/2}))$ time (see also [47] for a slightly better error bound).

The problem of increasing both edge- and vertex-connectivities in a given graph has been studied in [34,40–44]. Hsu and Kao [34] first treated the problem of augmenting the edge- and vertex-connectivities simultaneously, and presented a linear time algorithm for the problem of augmenting an undirected graph $G=(V,E)$ with two specified vertex sets $X, Y \subseteq V$ by adding a minimum number of edges such that the local vertex-connectivity (resp., local edge-connectivity) between every two vertices in $X$ (resp., in $Y$) becomes at least 2. Afterwards, Ishii et al. considered the problem of augmenting a multigraph $G=(V,E)$ with two integers $\ell$ and $k$ by adding a minimum number of edges such that $G$ becomes $\ell$-edge-connected and $k$-vertex-connected. They showed polynomial time algorithms for $k=2$ [40,45] and for a fixed $\ell$ and $k=3$ [42] (when a given graph is 2-vertex-connected) and [44] (for an arbitrary graph). For general $\ell$ and $k$, they also gave a polynomial time approximation algorithm which produces a solution whose size is at most $\max\{\ell+1, 2k-4\}$ over the optimum if a given graph is $(k-1)$-vertex-connected [43] (see [38] for the series of results by Ishii et al.).

For the edge-connectivity augmentation problem in an undirected graph, several types of restrictions on how to add edges have been studied (other than lower and upper bounds on degrees of vertices [17]). Jordán [51] proved that, if the given graph is simple and edges must be added without creating multiple edges, then the problem can be solved in polynomial time for a fixed target $k$, although the problem is shown to be NP-hard for general $k$. Bang-Jensen et al. [4] showed that if a partition $\{V_1, \ldots, V_p\}$ of $V$ is given as a constraint such that only edges connecting distinct subsets $V_i$ and $V_j$ can be added to $G$, then the problem can be solved in polynomial time. The problem of augmenting a connected planar graph to a 2-vertex-connected planar graph is shown to be NP-hard [53,55], and a 5/3-approximation algorithm is proposed by Fialko and Mutzel [16]. If a given graph $G$ is restricted to be outerplanar, then the following problems are polynomially solvable: Augment $G$ to a 2-edge-connected (resp., 2-vertex-connected) planar graph (Kant [53,54]), and for an even $k$ or $k=3$, augment $G$ to a $k$-edge-connected planar graph (Nagamochi and Eades [68]).

## 7. Concluding remarks

In this article, we started with the definition of an MA ordering of vertices in a graph, and then surveyed some of the new algorithms for solving the minimum cut problem and the edge-connectivity augmentation problem, as applications of MA orderings. Triggered by the work of Frank [17] that unified various approaches from the view point of edge-splitting, connectivity augmentation problems have been intensively studied in this decade. Additional constraints such as simplicitly or planarity of a graph have also been taken into account. In these developments, min–max type theorems played an important role in solving the corresponding connectivity augmentation problems exactly. Even if the min–max type theorem holds only approximately in the sense that there remains a small gap between the minimum and maximum objective functions, the problems are sometimes solvable by characterizing the graph structure that yields such a gap (see [4,3,27]). These results may lead to new frameworks for solving other types of combinatorial optimization problems related to connectivity and edge-splitting.

## Acknowledgements

Finally, this research was partially supported by the Scientific Grant-in-Aid by the Ministry of Education, Science, Sports and Culture of Japan. The authors would like to thank an anonymous referee for his/her helpful comments.

## References

[1] R.K. Ahuja, T.L. Magnanti, J.B. Orlin, Network Flows: Theory, Algorithms, and Applications, Prentice-Hall, Englewood Cliffs, NJ, 1993.

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

[3] J. Bang-Jensen, H.N. Gabow, T. Jordán, Z. Szigeti, Edge-connectivity augmentation with partition constraints, SIAM J. Discrete Math. 12 (1999) 160–207.

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

[5] A.A. Benczúr, Augmenting undirected connectivity in $\tilde{O}(n^3)$ time, Proceedings of 26th ACM Symposium on Theory of Computing, 1994, pp. 658–667.

[6] A.A. Benczúr, D.R. Karger, Augmenting undirected edge connectivity in $\tilde{O}(n^2)$ time, Proceedings of 9th Annual ACM-SIAM Symposium on Discrete Algorithms, 1998, pp. 500–519.

[7] G.-R. Cai, Y.-G. Sun, The minimum augmentation of any graph to $k$-edge-connected graph, Networks 19 (1989) 151–172.

[8] C.S. Chekuri, A.V. Goldberg, D.R. Karger, M.S. Levine, C. Stein, Experimental study of minimum cut algorithms, Proceedings of 8th Annual ACM-SIAM Symposium on Discrete Algorithm, 1997, pp. 324 –333.

[9] E. Cheng, T. Jordán, Successive edge-connectivity augmentation problems, Math. Program. Ser. B 84 (1999) 577–594.

[10] J. Cheriyan, M.-Y. Kao, R. Thurimella, Scan-first search and sparse certificates: an improved parallel algorithm for $k$-vertex connectivity, SIAM J. Comput. 22 (1993) 157–174.

[11] J. Cheriyan, R. Thurimella, Fast algorithms for $k$-shredders and $k$-node connectivity augmentation, J. Algorithms 33 (1999) 15–50.

[12] W.J. Cook, W.H. Cunningham, W.R. Pulleyblank, A. Schrijver, Combin. Optimization, Wiley-Interscience, Wiley, Inc., New York, 1998.

[13] E.A. Dinits, A.V. Karzanov, M.V. Lomonosov, On the structure of a family of minimal weighted cuts in a graph, in: A.A. Fridman (Ed.), Studies in Discrete Optimization, Nauka, Moscow, 1976, pp. 290–306 (in Russian).

[14] S. Enni, A note on mixed graphs and directed splitting off, J. Graph Theory 27 (1998) 213–221.

[15] K.P. Eswaran, R.E. Tarjan, Augmentation problems, SIAM J. Comput. 5 (1976) 653–665.

[16] S. Fialko, P. Mutzel, A new approximation algorithm for the planar augmentation problem, Proceedings of 9th Annual ACM-SIAM Symposium on Discrete Algorithms, 1998, pp. 260–269.

[17] A. Frank, Augmenting graphs to meet edge-connectivity requirements, SIAM J. Discrete Math. 5 (1992) 25–53.

[18] A. Frank, Connectivity augmentation problems in network design, in: J.R. Birge, K.G. Murty (Ed.), Mathematical Programming: State of the Art 1994, The University of Michigan, Ann Arbor, MI 1994, pp. 34–63.

[19] A. Frank, On the edge-connectivity algorithm of Nagamochi and Ibaraki, Laboratoire Artemis, IMAG, Université J. Fourier, Grenoble, March 1994.

[20] A. Frank, T. Ibaraki, H. Nagamochi, On sparse subgraphs preserving connectivity properties, J. Graph Theory 17 (1993) 275–281.

[21] A. Frank, T. Jordán, Minimal edge-coverings of pairs of sets, J. Combin. Theory Ser. B 65 (1995) 73–110.

[22] A. Frank, T. Jordán, Directed vertex-connectivity augmentation, Math. Program. Ser. B 84 (1999) 537–554.

[23] M.L. Fredman, R.E. Tarjan, Fibonacci heaps and their uses in improved network optimization algorithms, J. ACM 34 (1987) 596–615.

[24] S. Fujishige, Another simple proof of the validity of Nagamochi and Ibaraki's min-cut algorithm and Queyranne's extension to symmetric submodular function minimization, J. Oper. Res. Soc. Japan 41 (1998) 626–628.

[25] H.N. Gabow, A matroid approach to finding edge connectivity and packing arborescences, Proceedings of 23rd ACM Symposium on Theory of Computing, New Orleans, Louisiana, 1991, pp. 112–122.

[26] H.N. Gabow, Efficient splitting off algorithms for graphs, Proceedings of 26th ACM Symposium on Theory of Computing, 1994, 696–705.

[27] H.N. Gabow, T. Jordán, How to make a square grid framework with cables rigid, SIAM J. Comput. 30 (2000) 649–680.

[28] A.V. Goldberg, S. Rao, Flows in undirected unit capacity network, Proceedings of 38th Annual IEEE Symposium on Foundations of Computer Science, 1997, pp. 32–35.

[29] D. Gusfield, Optimal mixed graph augmentation, SIAM J. Comput. 16 (1987) 599–612.

[30] J. Hao, J.B. Orlin, A faster algorithm for finding the minimum cut in a directed graph, J. Algorithms 17 (1994) 424–446.

[31] M.R. Henzinger, S. Rao, H.N. Gabow, Computing vertex connectivity: new bounds from old techniques, J. Algorithms 34 (2000) 222–250.

[32] T. Hsu, On four-connecting a triconnected graph, J. Algorithms 35 (2000) 202–234.

[33] T. Hsu, Undirected vertex-connectivity structure and smallest four-vertex connectivity augmentation, Lecture Notes in Computer Science 1004, Sixth International Symposium on Algorithms and Computation, Springer, Berlin, 1995, pp. 274–283.

[34] T. Hsu, M. Kao, A unifying augmentation algorithm for two-edge-connectivity and biconnectivity, J. Combin. Optim. 2 (1998) 237–256.

[35] T. Hsu, V. Ramachandran, A linear time algorithm for triconnectivity augmentation, Proceedings of 32nd IEEE Symposium on Foundations of Computer Science, 1991, pp. 548–559.

[36] T. Hsu, V. Ramachandran, Finding a smallest augmentation to biconnect a graph, SIAM J. Comput. 22 (1993) 889–912.

[37] T.C. Hu, Integer Programming and Network Flows, Addison-Wesley, Reading, MA, 1969.

[38] T. Ishii, Studies on multigraph connectivity augmentation problems, Ph.D. Thesis, Dept. of Applied Mathematics and Physics, Kyoto University, Kyoto, Japan, 2000.

[39] T. Ishii, H. Nagamochi, On the minimum augmentation of an $\ell$-connected graph to a $k$-connected graph, Lecture Notes in Computer Science, 1851, Springer, Berlin, Seventh Biennial Scandinavian Workshop on Algorithm Theory, Bergen, Norway, July 5–7, 2000, pp. 286–299.

[40] T. Ishii, H. Nagamochi, T. Ibaraki, Augmenting edge and vertex connectivities simultaneously, Lecture Notes in Computer Science, 1350, Springer, Berlin, Eighth International Symposium on Algorithms and Computation, 1997, pp. 102–111.

[41] T. Ishii, H. Nagamochi, T. Ibaraki, Optimal augmentation of a biconnected graph to a $k$-edge-connected and triconnected graph, Proceedings of 9th Annual ACM-SIAM Symposium on Discrete Algorithms, 1998, pp. 280–289.

[42] T. Ishii, H. Nagamochi, T. Ibaraki, $k$-edge and 3-vertex connectivity augmentation in an arbitrary multigraph, Lecture Notes in Computer Science, vol. 1533, Springer, Berlin, in: K.-Y. Chwa, O.H. Ibarra (Eds.), Algorithms and Computation, Ninth International Symposium on Algorithms and Computation, 1998, pp. 159–168.

[43] T. Ishii, H. Nagamochi, T. Ibaraki, Augmenting a $(k-1)$-vertex-connected multigraph to an $\ell$-edge-connected and $k$-vertex-connected multigraph, Lecture Notes in Computer Science, 1643, Springer, Berlin, Seventh Annual European Symposium on Algorithms, 1999, pp. 414–425.

[44] T. Ishii, H. Nagamochi, T. Ibaraki, Optimal augmentation of a 2-vertex-connected multigraph to a $k$-edge-connected and 3-vertex-connected multigraph, J. Combin. Optim. 4 (2000) 35–78.

[45] T. Ishii, H. Nagamochi, T. Ibaraki, Multigraph augmentation under biconnectivity and general edge-connectivity requirements, Networks 37 (2001) 144–155.

[46] B. Jackson, Some remarks on arc-connectivity, vertex splitting, and orientation in graphs and digraphs, J. Graph Theory 12 (1988) 429–436.

[47] B. Jackson, T. Jordán, A near optimal algorithm for vertex connectivity augmentation, in: Proc. 11th Annual Int. Symp. on Algorithms and Computation ISAAC'00, Lecture Notes in Computer Science, Vol. 1969, Springer, Berlin, 2000, pp. 326–337.

[48] T. Jordán, Connectivity augmentation problem in graphs, Ph.D. Thesis, Department of Computer Science, Eötvös University, Budapest, Hungary, 1994.

[49] T. Jordán, On the optimal vertex-connectivity augmentation, J. Combin. Theory Ser. B 63 (1995) 8–20.

[50] T. Jordán, A note on the vertex-connectivity augmentation problem, J. Combin. Theory Ser. B 71 (1997) 294–301.

[51] T. Jordán, Two NP-complete augmentation problems, Odense University Preprints #8, 1997.

[52] T. Jordán, Edge-splitting problems with demands, Lecture Notes in Computer Science, 1610, Springer, Berlin, Seventh Conference on Integer Programming and Combinatorial Optimization, 1999, pp. 273–288.

[53] G. Kant, Algorithms for Drawing Planar Graphs, Ph.D. Thesis, Dept. Of Computer Science, Utrecht University, 1993.

[54] G. Kant, Augmenting outerplanar graphs, J. Algorithms 21 (1996) 1–25.

[55] G. Kant, H.L. Bodlaender, Planar graph augmentation problems, Lecture Notes in Computer Science, vol. 621, Springer, Berlin, 1992, pp. 258–271.

[56] M. Kao, Data security equals graph connectivity, SIAM J. Discrete Math. 9 (1996) 87–100.

[57] D.R. Karger, Minimum cuts in near-linear time, Proceedings of 28th ACM Symposium on Theory of Computing, 1996, pp. 56–63.

[58] D.R. Karger, M.S. Levine, Finding maximum flows in undirected graphs seems easier than bipartite matching, Proceedings of 30th ACM Symposium on Theory of Computing, 1998, pp. 69–78.

[59] D.R. Karger, R. Motwani, An NC algorithm for minimum cuts, SIAM J. Comput. 26 (1997) 255–272.

[60] D.R. Karger, C. Stein, An $\tilde{O}(n^2)$ algorithm for minimum cuts, Proceedings of 25th ACM Symposium on Theory of Computing, 1993, pp. 757–765.

[61] D.R. Karger, C. Stein, A new approach to the minimum cut problems, J. ACM 43 (1996) 601–640.

[62] L. Lovász, Conference on Graph Theory, Prague, 1974.

[63] L. Lovász, Combinatorial Problems and Exercises, North-Holland, Amsterdam, 1979.

[64] W. Mader, A reduction method for edge-connectivity in graphs, Ann. Discrete Math. 3 (1978) 145–164.

[65] W. Mader, Konstruktion aller $n$-fach kantenzusammenhängenden Digraphen, European J. Combin. 3 (1982) 63–67.

[66] D.W. Matula, A linear time $2+\varepsilon$ approximation algorithm for edge connectivity, Proceedings of 4th Annual ACM-SIAM Symposium on Discrete Algorithms, 1993, pp. 500–504.

[67] D.W. Matula, Two results on search and edge connectivity, Graph Algorithms and Applications, Dagtuhl-Seminar-Report, vol. 145, 1996, p. 16.

[68] H. Nagamochi, P. Eades, Edge-splitting and edge-connectivity augmentation in planar graphs, Lecture Notes in Computer Science, 1412, Springer, Berlin, in: R.E. Bixby, E.A. Boyd, R.Z. Ríos-Mercado (Eds.), Sixth Conference on Integer Programming and Combinatorial Optimization, 1998, pp. 96–111.

[69] H. Nagamochi, T. Ibaraki, A linear-time algorithm for finding a sparse $k$-connected spanning subgraph of a $k$-connected graph, Algorithmica 7 (1992) 583–596.

[70] H. Nagamochi, T. Ibaraki, Computing edge-connectivity in multigraphs and capacitated graphs, SIAM J. Discrete Mathematics 5 (1992) 54–66.

[71] H. Nagamochi, T. Ibaraki, Deterministic $\tilde{O}(nm)$ time edge-splitting in undirected graphs, J. Combin. Optim. 1 (1997) 5–46.

[72] H. Nagamochi, T. Ibaraki, A note on minimizing submodular functions, Inform. Process. Lett. 67 (1998) 239–244.

[73] H. Nagamochi, T. Ibaraki, Polyhedral structure of submodular and posi-modular systems, Lecture Notes in Computer Science, 1533, in: K.-Y. Chwa, O.H. Ibarra (Eds.), Springer, Berlin, Algorithms and Computation, Ninth International Symposium on Algorithms and Computations, 1998, pp. 169–178.

[74] H. Nagamochi, T. Ibaraki, Augmenting edge-connectivity over the entire range in $\tilde{O}(nm)$ time, J. Algorithms 30 (1999) 253–301.

[75] H. Nagamochi, T. Ibaraki, Polyhedral structure of submodular and posi-modular systems, Discrete Appl. Math. 107 (2000) 165–189.

[76] H. Nagamochi, T. Ono, T. Ibaraki, Implementing an efficient minimum capacity cut algorithm, Math. Program. 67 (1994) 325–341.

[77] H. Nagamochi, T. Ishii, T. Ibaraki, A simple and constructive proof of a minimum cut algorithm, Inst. Electron. Inform. Comm. Eng. Trans. Fundamentals E82-A (1999) 2231–2236.

[78] H. Nagamochi, S. Nakamura, T. Ibaraki, A simplified $\tilde{O}(nm)$ time edge-splitting algorithm in undirected graphs, Algorithmica 26 (2000) 56–67.

[79] H. Nagamochi, Y. Nakao, T. Ibaraki, A fast algorithm for cactus representations of minimum cuts, J. Japan Soc. Ind. Appl. Math. 17 (2000) 245–264.

[80] M. Padberg, G. Rinaldi, An efficient algorithm for the minimum capacity cut problem, Math. Program. 47 (1990) 19–36.

[81] J.C. Picard, M. Queyranne, On the structure of all minimum cuts in a network and applications, Math. Program. Study 13 (1980) 8–16.

[82] J. Plesnik, Minimum block containing a given graph, Archiv der Mathmatik, XXVII 1976, Fasc. 6, pp. 668–672.

[83] M. Queyranne, Minimizing symmetric submodular functions, Math. Program. 82 (1998) 3–12.

[84] S. Raghavan, T.L. Magnanti, Network Connectivity, in: M. Dell'Amico, F. Maffioli, S. Martello (Eds.), Annotated Bibliographies in Combinatorial Optimization, Wiley, New York, 1997.

[85] R. Rizzi, On minimizing symmetric set functions, Combinatorica 20 (2000) 445–450.

[86] M. Stoer, F. Wagner, A simple min-cut algorithm, J. ACM 44 (1997) 585–591.

[87] R.E. Tarjan, M. Yannakakis, Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs, SIAM J. Comput. 13 (1984) 566–579.

[88] S. Tsukiyama, K. Koike, I. Shirakawa, An algorithm to eliminate all complex triangles in a maximal planar graph for use in VLSI floor-plan, Proceedings of ISCAS'86, 1986, pp. 321–324.

[89] T. Watanabe, A. Nakamura, Edge-connectivity augmentation problems, J. Comput. System Sci. 35 (1987) 96–144.

[90] T. Watanabe, A. Nakamura, A minimum 3-connectivity augmentation of a graph, J. Comput. System Sci. 46 (1993) 91–128.