A 3-Approximation Algorithm for Finding Optimum 4,5-Vertex-Connected Spanning Subgraphs

Yefim Dinitz[*][†] Zeev Nutov[‡][§]

Abstract

The problem of finding a minimum weight k-vertex connected spanning subgraph in a graph $G = (V, E)$ is considered. For $k \geq 2$, this problem is known to be NP-hard. Based on the paper of Auletta, Dinitz, Nutov and Parente in this issue, we derive a 3-approximation algorithm for $k \in \{4, 5\}$. This improves the best previously known approximation ratios $4\frac{1}{6}$ and $4\frac{17}{30}$, respectively. The complexity of the suggested algorithm is $O(|V|^5)$ for the deterministic and $O(|V|^4 \log |V|)$ for the randomized version.

The way of solution is as follows. Analyzing a subgraph constructed by the algorithm of the aforementioned paper, we prove that all its "small" cuts have exactly two sides and separate a certain fixed pair of vertices. Such a subgraph is augmented to a k-connected one (optimally) by at most four executions of a min-cost k-flow algorithm.

[*]Up to 1990, E. A. Dinic, Moscow.

[†]Dept. of Mathematics and Computer Science, Ben-Gurion University of the Negev, Beer-Sheva 84105, Israel. *E-mail*: `dinitz@cs.bgu.ac.il`.

[‡]This work was done as a part of the author's D.Sc. thesis at the Dept. of Mathematics, Technion, Haifa, Israel.

[§]Max-Planck-Institut für Informatik, Im Stadtwald, 66123 Saarbrücken, Germany. *E-mail*: `nutov@mpi-sb.mpg.de`.

1 Introduction

In this paper we consider the following problem (for motivation see [1]):

> *Minimum weight k-connected subgraph problem*: given an integer k and a k-connected graph with a nonnegative weight function on its edges, find its minimum weight k-vertex-connected spanning subgraph.

The case $k = 1$ is reduced to the problem of finding a minimum weight spanning tree. Beginning from $k = 2$, the minimum weight k-connected subgraph problem is known to be NP-hard. (To see this, notice that a 2-connected spanning subgraph of a graph $G = (V, E)$ has $|V|$ edges if and only if G has a Hamiltonian cycle. A generalization to the case of any $k > 2$ is rather easy: let us add to such a G $k-2$ new vertices connected each to all vertices in the graph by edges of weight zero, arriving at an equivalent instance of a k-connected spanning subgraph problem.)

An approximation algorithm is called *α-approximation*, or is said to achieve an *approximation ratio α*, if it is a polynomial time algorithm that produces a solution of weight no more than α times the value of an optimal solution. For an arbitrary k, the best known approximation algorithm is due to Ravi and Williamson [12]; it achieves the approximation ratio $2H(k)$, where $H(k) = 1 + \frac{1}{2} + \ldots + \frac{1}{k}$ is the kth Harmonic number. For a general instance of the minimum weight k-connected subgraph problem, better approximation ratios were obtained only for $k = 2, 3$: see 2-approximation algorithms for $k = 2$ in [11, 1] and for $k = 3$ in [1] (for a review see [1].)

The main result of this paper is a 3-approximation algorithm for the minimum weight 4- and 5-connected subgraph problems. This improves the best previously known performance guarantees $4\frac{1}{6}$, $4\frac{17}{30}$ [12], respectively.

The main subroutine of our algorithm is "Out-Connected Subgraph Algorithm", abbreviated OCSA, of [1]. For $k \in \{4, 5\}$, OCSA finds a subgraph which is $(k-1)$-connected and whose weight at most twice the value of an optimal solution to the problem. Our algorithm finds an additional set of edges to be added to this subgraph to make the resulting subgraph k-connected. We show that in the subgraph computed by OCSA, there exist two vertices such that all "small" cuts separate one of them from the other and have exactly two sides (such a property was also used in [9] for $k = 3$ and in [2] for $k = 4$). This property enables us to find an optimal set of edges to be added by at most four executions of a min-cost k-flow algorithm.

In addition, a faster randomized version ROCSA of OCSA is suggested. Its time complexity is $O(k^2n^2m\log n) = O(k^2n^4\log n)$, where n is the number of vertices and m is the number of edges in the input graph; this is by factor $\frac{n}{\log n}$ better than the complexity of OCSA. We note that ROCSA is a randomized approximation algorithm for our problem for the cases $k = 2, 3$. It has the same approximation ratio 2 as the best previously known algorithms for $k = 2, 3$ ([11], [1], respectively), but its time complexity $O(n^2m\log n) = O(n^4\log n)$ is better than $O(n^3m) = O(n^5)$ achieved by those algorithms.

The complexity of the algorithm suggested for cases $k = 4, 5$ is the same as the complexity of OCSA and ROCSA: $O(n^5)$ for the deterministic and $O(n^4\log n)$ for the randomized versions.

Auletta and Parente [2] obtained independently a 3-approximation algorithm for the case $k = 4$. However, we use additional ideas which lead to a much simpler analysis and algorithm for $k = 4$ and enable us to derive a 3-approximation algorithm for $k = 5$.

Recently, in [8], it was shown that the algorithms of [1] and of this paper can be combined with the algorithm of [12] to achieve a slightly better approximation guarantee than $2H(k)$ for all k.

This paper is organized as follows. In Section 2 we give notations and describe known results used in the paper. Section 3 presents the randomized algorithm ROCSA. Section 4 presents an analysis of 4- and 5-out-connected graphs and introduces our 3-approximation algorithm for the minimum weight 4- and 5-connected subgraph problems.

The preliminary version of this paper is a part of [3].

2 Preliminaries and Notations

Let $G = (V, E)$ be an undirected simple graph (i.e., without loops and multiple edges) with vertex set V and edge set E. For $S, T \subset V$ we denote by $E(S, T)$ the set of edges with one end in S and the other end in T. For a vertex v of a graph G we denote by $N_G(v)$ the set of neighbors of v in G, and by $d_G(v) = |N_G(v)|$ the *degree* of v in G. In the case G is understood, we omit the subscript "G" in these notations.

A graph G with a nonnegative weight (cost) function w on its edges is referred to as a *weighted graph* and is denoted by (G, w), or simply by G if w is understood. For a weight function w and $E' \subseteq E$, we use the notation $w(E') = \sum\{w(e) : e \in E'\}$. For a subgraph $G' = (V', E')$ of a weighted graph (G, w), $w(G')$ is defined to be $w(E')$. A subgraph $G' = (V', E')$ is called *spanning* if $V' = V$; in this paper, we use only spanning subgraphs and, thus, sometimes omit the word "spanning". Similar notations are used for digraphs.

A subset $C \subseteq V$ is a *(vertex) cut* of a connected graph G if $G \setminus C$ is disconnected; if $|C| = k$ then such C is called a *k-cut*. A *side* of a cut C is the vertex set of a connected component of $G \setminus C$. Given a cut C and two nonempty disjoint subsets $S, T \subset V \setminus C$, we say that C *separates S from T* if $G \setminus C$ contains no path with one end in S and the other end in T. Clearly, if C is a cut and $\{C, S, T\}$ is a partition of V, then C separates S from T if and only if $E(S, T) = \emptyset$. A cut that separates a vertex s from a vertex t is called an *$\{s, t\}$-cut*. An $\{s, t\}$-cut C is called *$\{s, t\}$-minimum cut* if C has the minimum cardinality among all $\{s, t\}$-cuts. A graph G is *k-connected* if it is a complete graph on $k + 1$ vertices or if it has at least $k + 2$ vertices and contains no l-cut with $l < k$. The *connectivity* of G, denoted by $\kappa(G)$, is defined to be the maximum k for which G is k-connected. In what follows we assume that $|V| \geq k + 2$; thus $\kappa(G)$ is the cardinality of a minimum cut of G.

A set of paths is said to be *internally disjoint* if no two of them have an internal vertex in common. Following [4], a graph (resp., digraph) such that there exist k internally disjoint paths from a certain vertex r to any its other vertex is said to be *k-out-connected from r*. Using Menger's Theorem, it is not hard to see that in a graph which is k-out-connected from r, any l-cut with $l < k$, if such exists, must contain r.

A graph G is called *minimally k-connected* if $\kappa(G) = k$, but for any $e \in E$, $\kappa(G \setminus e) < k$. Observe that among the subgraphs which are optimal solutions for the minimum weight k-connected subgraph problem, there always exists a minimally k-connected one.

The *underlying graph* of a digraph D is the simple graph $U(D)$ obtained from D by replacing, for every $u, v \in V$, the set of arcs with endnodes u, v, if nonempty, by an edge (u, v). The *directed version* of a weighted graph (G, w) is the weighted digraph $D(G)$ obtained from G by replacing every undirected edge of G by the two antiparallel directed edges with the same ends and of the same weight. For simplicity of notations, we denote the weight function of $D(G)$ also by w.

Throughout the paper, let $\mathcal{G} = (\mathcal{G}, w)$ denote the input graph, n and m denote the numbers of its vertices and edges, respectively, and w^* denote the value of an optimal solution to our problem.

3 Randomized Out-Connected Subgraph Algorithm

The algorithm OCSA [1], finds a spanning subgraph of weight at most $2w^*$ which is k-out-connected from a vertex of degree k.[1] By [1, Corollary 3.2], such a subgraph is $(\lceil \frac{k}{2} \rceil + 1)$-connected; in particular, if $k \in \{2, 3\}$ then it is k-connected and if $k \in \{4, 5\}$ then it is $(k-1)$-connected. The time complexity of OCSA is $O(k^2 n^3 m)$. Here we suggest a randomized version ROCSA of OCSA (of Monte-Carlo type) with the complexity $O(k^2 n^2 m \log n)$.

Let us describe OCSA briefly. The following auxiliary problem is considered. Let $(\mathcal{D}, w)$ be a weighted digraph and r be its vertex; among all k-out-connected from r subdigraphs of $\mathcal{D}$ such that r has outdegree k in them, if such exist, find one of the minimal weight. In [1], this problem is reduced to the problem of finding an optimal k-out-connected from r subdigraph in $\mathcal{D}$, where large penalties are added to the weights of edges incident to r. The latter problem can be solved by a single run of the algorithm [5] in $O(k^2 n^2 m)$ time.

For any input $(\mathcal{G}, w)$ and k, OCSA:

1. Constructs the directed version $\mathcal{D} = D(\mathcal{G})$ of $\mathcal{G}$.

2. For every vertex r of $D(\mathcal{G})$, finds a solution D_r to the above auxiliary problem.

3. Maintains the cheapest graph $\tilde{G}$ (with the corresponding vertex $\tilde{r}$) among the underlying graphs of the subdigraphs D_r; at the end of the algorithm, $\tilde{G}, \tilde{r}$ are output.

[1]Here and further we mean the degree w.r.t. the subgraph.

We define a randomized algorithm ROCSA as a modification of OCSA by setting that (i) the sequence of examined vertices r is chosen at random and (ii) the algorithm ends and $\tilde{G}, \tilde{r}$ are output after the examination of $\lceil \log_{\frac{3}{2}} n \rceil$ first vertices.

Let us analyze the output of ROCSA. Let G^* be any optimal minimally k-connected spanning subgraph. According to the analysis of OCSA in [1], $\tilde{G}$ becomes a $(\lceil \frac{k}{2} \rceil + 1)$-connected subgraph of $\mathcal{G}$ of weight at most $2w^*$ at least as soon as a vertex with degree k in G^* is examined. Mader [7] shows that, for $k \geq 2$, any minimally k-connected graph with n vertices has at least $\frac{(k-1)n+2}{2k-1} > \frac{1}{3}n$ vertices of degree k. This implies that with probability $1 - \frac{1}{n}$ at least one of the $\lceil \log_{\frac{3}{2}} n \rceil$ vertices examined in ROCSA has degree k in G^*. We arrive at the following statement.

Theorem 3.1 *For any integer $k \geq 2$ and any weighted k-connected graph $\mathcal{G}$, with probability $1 - \frac{1}{n}$, the output of ROCSA is a spanning subgraph of $\mathcal{G}$ which is k-out-connected from a vertex of degree k, and hence is $(\lceil \frac{k}{2} \rceil + 1)$-connected, and whose weight is at most $2w^*$. The time complexity of ROCSA is $O(k^2 n^2 m \log n)$.*

As a consequence, we obtain:

Corollary 3.2 *For $k \in \{2, 3\}$, ROCSA is a randomized 2-approximation algorithm for the minimum weight k-connected subgraph problem, with the time complexity $O(n^2 m \log n) = O(n^4 \log n)$.*

4 Minimum Weight 4- and 5-Connected Subgraph Problems

In this section we present a 3-approximation algorithm for the minimum weight 4- and 5-connected subgraph problems. The idea is to execute OCSA as the first phase, which produces a $(k-1)$-connected subgraph, and in the second phase, to augment this subgraph by an appropriate set of edges. Such a set of edges is found as follows. We show that, for $k \in \{4, 5\}$, the subgraph $\tilde{G}$ produced by OCSA contains certain two vertices such that each its $(k-1)$-cut separates one of them from the other. We "destroy" all $(k-1)$-cuts of

$\tilde{G}$ by adding an edge set F of edges, such that $\tilde{G} \cup F$ contains k internally disjoint paths between these two vertices; such an optimal edge set F is found by using a minimum cost k-flow algorithm.

We need several preliminary statements.

Lemma 4.1 ([1]) *Let G be a k-out-connected graph from a vertex r, and let C be an l-cut of G with $l < k$. Then $r \in C$, and for every side S of C holds: $l \geq k - |S \cap N(r)| + 1$. Thus $\kappa(G) \geq k - \lfloor \frac{d(r)}{2} \rfloor + 1$. In particular, if $d(r) = k$ then G is $(\lceil \frac{k}{2} \rceil + 1)$-connected.*

Let S be a side of an l-cut of G, $l < k$. Then, by Lemma 4.1, $|S \cap N_G(r)| \geq k - l + 1 \geq 2$. Thus Lemma 4.1 implies the following statement.

Corollary 4.2 *Let G be a k-out-connected graph from a vertex r of degree k, $k \in \{4, 5\}$. Then G is $(k-1)$-connected, and for every $(k-1)$-cut C of G holds:*

(i) *C has exactly two sides;*

(ii) *If $k = 4$ then $|C \cap N(r)| = 0$ and each side of C contains exactly two vertices from $N(r)$;*

(iii) *If $k = 5$ then $|C \cap N(r)| \leq 1$ and each side of C contains two or three vertices from $N(r)$; thus at least one side contains exactly two vertices from $N(r)$.*

The following fact is well known (its proof is presented here for completeness of exposition).

Lemma 4.3 *Let C, C' be two $\{s, t\}$-minimum cuts of a graph G such that each of C and C' has exactly two sides, say S, T and S', T', respectively, where $s \in S, S'$ and $t \in T, T'$. Then $R_s = (C \cup C') \setminus (T \cup T')$ separates $S \cap S'$ from $T \cup T'$, $R_t = (C \cup C') \setminus (S \cup S')$ separates $T \cap T'$ from $S \cup S'$, and both R_s, R_t are $\{s, t\}$-minimum cuts as well.*

Proof: Notice that each of $\{R_s, S \cap S', T \cup T'\}$ and $\{R_t, S \cup S', T \cap T'\}$ is a partition of V. Also, $E(S \cap S', T \cup T') = E(S \cup S', T \cap T') = \emptyset$, since $E(S, T) = E(S', T') = \emptyset$. This implies that R_s separates $S \cap S'$ from $T \cup T'$,

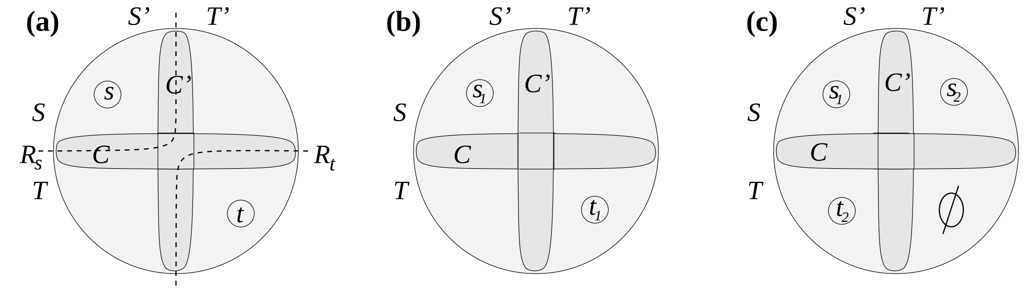

Figure 1: Illustrations for the proofs of Lemmas 4.3 and 4.4.

and R_t separates $T \cap T'$ from $S \cup S'$ (for illustration see Fig. 1a). In particular, we obtain that R_s and R_t are both $\{s,t\}$-cuts. To see that R_s and R_t are both $\{s,t\}$-minimum cuts, observe that $|R_s| + |R_t| = |C| + |C'|$ and that $|R_s|, |R_t| \geq |C| = |C'|$. $\square$

We begin our analysis from the following statement.

Lemma 4.4 *Let G be a k-out-connected graph from a vertex r of degree k, $k \in \{4,5\}$. Let C be a $(k-1)$-cut of G and S a side of C that contains exactly two vertices from $N(r)$, say, s_1 and s_2. Then, for any $(k-1)$-cut of G, both s_1 and s_2 are contained in the same side of C.*

Proof: Recall that, by Corollary 4.2, G is $(k-1)$-connected, and every $(k-1)$-cut of G contains at most one vertex from $N(r)$ and has exactly two sides, containing 2 or 3 vertices from $N(r)$ each. Let T be the side of C that is distinct from S. Assume in negation that there exists a $(k-1)$-cut C' with sides S' and T' such that s_1 and s_2 are not contained in the same side of C'. Since C' contains at most one of s_1, s_2, we assume w.l.o.g. that $s_1 \in S'$, $s_2 \notin S'$. Let us consider two cases:

$T \cap T' \neq \emptyset$: Let $t_1 \in T \cap T'$ (see Fig. 1b). Then, clearly, both C and C' are $\{s_1, t_1\}$-minimum cuts. Thus, by Lemma 4.3, the cut $R_{s_1} = (C \cup C') \setminus (T \cup T')$ is a $(k-1)$-cut with sides $S \cap S', T \cup T'$. But $(S \cap S') \cap N_r = \{s_1\}$, a contradiction to Corollary 4.2.

$T \cap T' = \emptyset$: In this case $s_2 \in S \cap T'$, since $|T' \cap N(r)| \geq 2$, $|C \cap N(r)| \leq 1$, and $(S \setminus S') \cap N(r) = s_2$. Moreover, $T \cap S' \neq \emptyset$, since $|T \cap N(r)| \geq 2$ and

$|C' \cap N(r)| \leq 1$. Let $t_2 \in T \cap S'$ (see Fig. 1c). Then the contradiction is obtained by using similar arguments for s_2, t_2 instead of s_1, t_1.

□

Remark. Observe that Lemma 4.4, together with Corollary 4.2, implies that a k-out-connected graph G from a vertex r of degree k, $k \in \{4,5\}$, has a vertex s neighboring to r which does not belong to any cut smaller than k (in fact, there are four such vertices). Such a vertex s can be easily recognized as a neighbor of r for which $\kappa(G \setminus s) \geq k-1$. This observation, together with Lemma 4.1, already implies a 4-approximation algorithm for the minimum 4- and 5-connected subgraph problems, as follows. Let $\tilde{G}, \tilde{r}$ be the output of the OCSA for the input $(\mathcal{G}, w)$ and k, and let, for s as above, $\tilde{G}_s$ be the underlying graph of the minimum weight k-out-connected from s subdigraph of $D(\mathcal{G})$. Then, for $k \in \{4,5\}$, the graph $\tilde{G} \cup \tilde{G}_s$ has no $(k-1)$-cuts, i.e., is k-connected. Similarly to the proof of [1, Lemma 4.3], $w(\tilde{G}_s) \leq 2w^*$. Thus, $w(\tilde{G} \cup \tilde{G}_s) \leq w(\tilde{G}) + w(\tilde{G}_s) \leq 2w^* + 2w^* = 4w^*$, as required.

The following Lemma is crucial to derive our 3-approximation algorithm for $k \in \{4,5\}$.

Lemma 4.5 *Let G be a k-out-connected graph from a vertex r of degree k, $k \in \{4,5\}$. Then there exists a pair of vertices $\{s,t\} \subset N(r)$ such that every $(k-1)$-cut of G has exactly two sides and separates s from t. Moreover, for any distinct four vertices $s_1, s_2, t_1, t_2 \in N(r)$, there exists such a pair s,t with $s \in \{s_1, s_2\}$, $t \in \{t_1, t_2\}$.*

Proof: If G is k-connected, then the proof is immediate. Otherwise, by Corollary 4.2, $\kappa(G) = k-1$ and any $(k-1)$-cut has exactly two sides.

Let us suppose first that there exists a $(k-1)$-cut C which has exactly two vertices from $N(r)$ in each its side, say s_1, s_2 and t_1, t_2. Then, by Lemma 4.4, any $(k-1)$-cut separates s from t for any $s \in \{s_1, s_2\}$ and $t \in \{t_1, t_2\}$.

Otherwise, by Corollary 4.2, $k = 5$ and every $(k-1)$-cut has exactly two vertices from $N(r)$ in its one side and three vertices from $N(r)$ in its other side. Let C be a $(k-1)$-cut, and let $N(r)$ be partitioned into $\{s_1, s_2\}$ and $\{t_1, t_2, t_3\}$ by the two sides of C. Suppose now that there exists a $(k-1)$-cut C' which does not separate $\{s_1, s_2\}$ from $\{t_1, t_2, t_3\}$. Then Lemma 4.4 implies that C' has s_1, s_2 and one of t_1, t_2, t_3, say, t_3, in its one side, and t_1, t_2 in its other side. But then, by Lemma 4.4, every $(k-1)$-cut must have s_1, s_2 in its one side and t_1, t_2 in its other side. Also in this case, any pair $\{s_i, t_j\}$, $i, j \in \{1, 2\}$, is appropriate.

The last part of the statement is checked by a simple case analysis. $\square$

The following statement stems straightforwardly from Menger's Theorem.

Fact 4.6 *Let G be a connected graph that has a pair $\{s, t\}$ of vertices such that any l-cut of G with $l < k$ has exactly two sides and separates s from t. Let F be a set of edges. Then $G \cup F$ is k-connected if and only if in $G \cup F$ there are k internally disjoint paths between s and t.*

Let now $\mathcal{G}$ be a weighted graph, and let G be a spanning subgraph of $\mathcal{G}$ with a pair $\{s, t\}$ as in the above statement. By Fact 4.6, finding an optimal edge set F such that $G \cup F$ is k-connected is equivalent to finding an optimal edge set F such that in $G \cup F$ there are k internally disjoint paths between s and t. The latter problem can be solved in polynomial time via a reduction to the min-cost k-flow problem with unit capacities of vertices (see, e.g., [4]); we use such a reduction below at Phase 2 of our algorithm.

We now present our 3-approximation algorithm for the minimum weight 4- and 5-connected subgraph problems.

4 and 5-Connected Subgraph Algorithm

Input: A k-connected weighted graph $(\mathcal{G}, w)$ and an integer k, $k \in \{4, 5\}$.

Output: A k-connected spanning subgraph of $\mathcal{G}$.

```
Phase 1: Execute OCSA or ROCSA on the input $\mathcal{G}, w, k$, and let $(\tilde{G}, \tilde{r})$ be
    the output;

Phase 2: Change the weight of any edge in $\tilde{G}$ to zero;
    Choose any four vertices $s_1, s_2, t_1, t_2 \in N_{\tilde{G}}(\tilde{r})$;
    For every pair of vertices $s \in \{s_1, s_2\}$, $t \in \{t_1, t_2\}$ do:
    1) Find in $\mathcal{G}$, with capacity 1 of every vertex in $V \setminus \{s, t\}$, a minimum
       cost $k$-flow from $s$ to $t$, using the algorithm [10];
    2) Find $k$ internally disjoint paths between $s$ and $t$, by path decompo-
       sition of this flow; let $F(s, t)$ be the union of their edge sets;
```

```
3) Check whether $\tilde{G} \cup F(s,t)$ is $k$-connected, using algorithm [6];
   If it is, then terminate with $\tilde{G} \cup F(s,t)$;
end for;
```

Theorem 4.7 *The* 4- *and* 5-*Connected Subgraph Algorithm is a* 3-*approximation algorithm for the minimum weight k-connected subgraph problems, $k \in \{4, 5\}$. The time complexity of the algorithm is $O(n^3 m) = O(n^5)$ deterministic* (*using OCSA*) *or $O(n^2 m \log n) = O(n^4 \log n)$ randomized* (*using ROCSA*).

Proof: At Phase 2 the algorithm finds edge sets $F(s,t)$ such that in $\tilde{G} \cup F(s,t)$ there are k internally disjoint paths between s and t. By Lemma 4.5 and Fact 4.6, for at least one of the pairs $\{s,t\}$ examined by the algorithm, $\tilde{G} \cup F(s,t)$ is k-connected. This shows the correctness of the algorithm.

We turn now to show the approximation ratio. Let $\tilde{E}$ be the edge set of $\tilde{G}$, and let $F = F(s,t)$ be the additional edge set chosen by the algorithm. The set $F \setminus \tilde{E}$ is an optimal augmenting edge set for $\tilde{G}$ (see the discussion before the description of the algorithm); clearly, its weight is at most w^*. By [1], $w(\tilde{G}) \leq 2w^*$. Summarizing: $w(\tilde{G} \cup F) = w(\tilde{G}) + w(F \setminus \tilde{E}) \leq 2w^* + w^* = 3w^*$.

Let us discuss the time complexity. At Phase 2 the algorithm finds a minimum cost k-flow and checks k-connectivity at most four times. Computing a minimum cost k-flow takes $O(n \log n(m + n \log n))$ time [10], and k-connectivity is checked in $O(k^2 n^2 + k^3 n^{1.5})$ time [6]. Hence, the dominating time is spent for executing Phase 1, i.e., the complexity of the algorithm using OCSA or ROCSA is the same as the complexity of OCSA or ROCSA, respectively. □

Remark. It can be easily shown that the same approximation ratio is valid even if at Phase 2 of the algorithm we do not reduce the weights of edges in $\tilde{G}$ to zero. However, this reduction guarrantees optimality of the edge set addition at Phase 2.

References

[1] V. Auletta, Ye. Dinitz, Z. Nutov, and D. Parente: A 2-approximation algorithm for finding an optimum 3-vertex-connected spanning subgraph.

J. of Algorithms, this issue, ??–??.

[2] V. Auletta and D. Parente: Better algorithms for minimum-weight connectivity problems. *Proc. 14th Annual Symposium on Theoretical Aspects of Computer Science (STACS '97)*, R. Reischuk and M. Morvan Eds., Lecture Notes in Comp. Sci., Vol. 1200, Springer-Verlag, Berlin (1997), 547–558. (See also Tech. Rep. Universita' di Salerno 6/96 (July 1996).)

[3] Ye. Dinitz and Z. Nutov. Finding minimum weight k-vertex connected spanning subgraphs: approximation algorithms with factor 2 for $k = 3$ and factor 3 for $k = 4, 5$. *Proc. 3rd Italian Conf. on Algorithms and Complexity, CIAC'97* (Rome, March 1997), Lecture Notes in Comp. Sci., v. 1203, Springer-Verlag, 1997, 13–24.

[4] A. Frank: Connectivity augmentation problems in network design. *Mathematical Programming, the State of the Art.* J. R. Birge and K. G. Murty eds. (1994), 34–63.

[5] H. N. Gabow: A representation for crossing set families with application to submodular flow problems. *Proc. 4th Annual ACM-SIAM Symp. on Discrete Algorithms* (1993), 202–211.

[6] Z. Galil: Finding the vertex connectivity of graphs. *SIAM J. of Computing* **9** (1980), 197–199.

[7] W. Mader: Ecken vom Grad n in minimalen n-fach zusammenhängenden Graphen. *Archiv. Math.* (*Basel*) **23** (1972), 219–224.

[8] Z. Nutov: Improved approximation algorithms for finding optimum k-vertex-connected spanning subgraphs. TR CORR 98-03, University of Waterloo, Waterloo, Canada, Feb. 1998.

[9] Z. Nutov and M. Penn: Faster approximation algorithms for weighted triconnectivity augmentation problems. *Operations Research Letters* **21**, (1997), 219–223.

[10] J. B. Orlin: A faster strongly polynomial minimum cost flow algorithm. *Operations Research* **41** (1993), 338–350.

[11] M. Penn and H. Shasha-Krupnik: Improved approximation algorithms for weighted 2 & 3 vertex connectivity augmentation problems. *J. of Algorithms* **22** (1997), 187–196.

[12] R. Ravi and D. P. Williamson: R. Ravi and D. P. Williamson: An approximation algorithm for minimum-cost vertex-connectivity problems. *Algorithmica* **18**, (1997), 21–43.