# Approximation Algorithms for $k$-Connected Graph Factors

Bodo Manthey[✉] and Marten Waanders

University of Twente, Enschede, The Netherlands
b.manthey@utwente.nl

**Abstract.** Finding low-cost spanning subgraphs with given degree and connectivity requirements is a fundamental problem in the area of network design. We consider the problem of finding $d$-regular spanning subgraphs (or $d$-factors) of minimum weight with connectivity requirements. For the case of $k$-edge-connectedness, we present approximation algorithms that achieve constant approximation ratios for all $d \geq 2 \cdot \lceil k/2 \rceil$. For the case of $k$-vertex-connectedness, we achieve constant approximation ratios for $d \geq 2k - 1$. Our algorithms also work for arbitrary degree sequences if the minimum degree is at least $2 \cdot \lceil k/2 \rceil$ (for $k$-edge-connectivity) or $2k - 1$ (for $k$-vertex-connectivity).

## 1 Introduction

Finding low-cost spanning subgraphs with given degree and connectivity requirements is a fundamental problem in the area of network design. The usual setting is that there are connectivity and degree requirements. Then the goal is to find a cheap subgraph that meets the connectivity requirements and the degree bounds. Beyond simple connectedness, higher connectivity, such as $k$-vertex-connectivity or $k$-edge-connectivity, has been considered in order to increase the reliability of the network. Most variants of such problems are NP-hard. Because of this, finding good approximation algorithms for such network design problems has been the topic of a significant amount of research [1, 4–10, 14, 16–20].

In this paper, we study the problem of finding low-cost spanning subgraphs with given degrees that meet connectivity requirements (they should be $k$-edge-connected or $k$-vertex-connected for a given $k$). Violation of the degree constraint is not allowed. While $d$-regular, spanning subgraphs of minimum weight can be found efficiently using Tutte's reduction to the matching problem [21, 23], even asking for simple connectedness makes the problem NP-hard [2]. For instance, asking for a 2-regular, connected graph of minimum weight is the NP-hard traveling salesman problem (TSP) [11, ND22].

### 1.1 Problem Definitions and Preliminaries

**Graphs and Connectivity.** All graphs in this paper are undirected and simple. Let $G = (V, E)$ a graph. In the following, $n = |V|$ is the number of vertices.

For a subset $X \subseteq V$ of the vertices, let $\text{cut}_G(X)$ be the number of edges in $G$ with one endpoint in $X$ and the other endpoint in $\overline{X} = V \setminus X$. For two disjoint sets $X, Y \subseteq V$ of vertices, let $\text{cut}_G(X, Y)$ be the number of edges in $G$ with one endpoint in $X$ and the other endpoint in $Y$.

Two vertices $u, v \in V$ are *locally $k$-edge-connected in $G$* if there are at least $k$ edge-disjoint paths from $u$ to $v$ in $G$. Equivalently, $u$ and $v$ are locally $k$-edge-connected in $G$ if $\text{cut}_G(X) \geq k$ for all $X \subseteq V$ with $u \in X$ and $v \notin X$. Local $k$-edge-connectedness is an equivalence relation as it is symmetric, reflexive, and transitive. A graph $G$ is *$k$-edge-connected* if all pairs of vertices are locally $k$-edge-connected in $G$.

Let $X \subseteq V$. We call $X$ a *$k$-edge-connected component* of $G$ if the graph induced by $X$ is $k$-edge-connected. We call $X$ a *locally $k$-edge-connected component of $G$* if all $u, v \in X$ are locally $k$-edge-connected in $G$. Note that every $k$-edge-connected component of $G$ is also a locally $k$-edge-connected component, but the reverse is not true.

A graph $G$ is *$k$-vertex-connected*, if the graph induced by the vertices $V \setminus K$ is connected for all sets $K \subseteq V$ with $|K| \leq k - 1$. Equivalently, for any two non-adjacent vertices $u, v \in V$, there exist at least $k$ vertex-disjoint paths connecting $u$ to $v$ in $G$.

For an overview of connectivity and algorithms for computing connectivity and connected components, we refer to two surveys [13, 15].

For a vertex $v \in V$, let $N_G(v) = N(v) = \{u \in V \mid \{u, v\} \in E\}$ be the neighbors of $v$ in $G$. The graph $G$ is *$d$-regular* if $|N(v)| = d$ for all $v \in V$. A $d$-regular spanning subgraph of a graph is called a *$d$-factor*.

By abusing notation, we identify a set $X \subseteq V$ of vertices with the subgraph induced by $X$. Similarly, if the set $V$ of vertices is clear from the context, we identify a set $F$ of edges with the graph $(V, F)$.

**Problem Definitions.** Let $G = (V, E)$ be an undirected, complete graph with non-negative edge weights $w$. The edge weights are assumed to satisfy the triangle inequality, i.e., $w(\{u, v\}) \leq w(\{u, x\}) + w(\{x, v\})$ for all distinct $u, v, x \in V$. For some set $F \subseteq E$ of edges, we denote by $w(F) = \sum_{e \in F} w(e)$ the sum of its edge weights. The weight of a subgraph is the weight of its edge set.

The problems considered in this paper are the following: as input, we are given $G$ and $w$ as above. Then **Min-$d$Reg-$k$Edge** denotes the problem of finding a $k$-edge-connected $d$-factor of $G$ of minimum weight. Similarly, **Min-$d$Reg-$k$Vertex** denotes the problem of finding a $k$-vertex-connected $d$-factor of $G$ of minimum weight.

Some of these problems coincide:

– Min-$d$Reg-1Edge and Min-$d$Reg-1Vertex are identical for all $d$ since 1-edge-connectedness and 1-vertex-connectedness are simple connectedness.
– For $k \in \{1, 2\}$, the problems Min-2Reg-$k$Edge and Min-2Reg-$k$Vertex are identical to the traveling salesman problem (TSP) because of the following: 2-regular graphs consist solely of simple cycles. If they are connected, they are 2-vertex-connected and form Hamiltonian cycles.

- For even $d$ and $k$, the problems Min-$d$Reg-$(k-1)$Edge and Min-$d$Reg-$k$Edge are identical. For even $d$, every $d$-factor can be decomposed into $d/2$ 2-factors. Thus, the size of every cut is even. Therefore, every $d$-regular $(k-1)$-edge-connected graph is automatically $k$-edge-connected for even $k$.
- For $k \in \{1, 2, 3\}$, the two problems Min-3Reg-$k$Edge and Min-3Reg-$k$Vertex are identical since edge- and vertex-connectivity are equal in cubic graphs [24, Theorem 4.1.11].

We also consider the generalizations of the problems Min-$d$Reg-$k$Edge and Min-$d$Reg-$k$Vertex to arbitrary degree sequences: for **Min-$d$Gen-$k$Edge**, we are given as additional input a degree requirement $d_v \in \mathbb{N}$ for every vertex $v$. The parameter $d$ is a lower bound for the degree requirements, i.e., we have $d_v \geq d$ for all vertices $v$. The goal is to compute a $k$-edge-connected spanning subgraph in which every vertex $v$ is incident to exactly $d_v$ vertices. **Min-$d$Gen-$k$Vertex** is analogously defined for $k$-vertex-connectivity. For the sake of readability, we restrict the presentation of our algorithms in Sects. 2 and 3 to Min-$d$Reg-$d$Edge and Min-$d$Reg-$k$Vertex, respectively, and we state the generalized results for Min-$d$Gen-$k$Edge and Min-$d$Gen-$k$Vertex only in Sect. 4.

We use the following notation: $\mathrm{OptE}^k$ denotes a $k$-edge-connected spanning subgraph of minimum weight. $\mathrm{OptV}^k$ denotes a $k$-vertex-connected spanning subgraph of minimum weight. For both, no degree requirements have to be satisfied. $\mathrm{OptF}_d$ denotes a (not necessarily connected) $d$-factor of minimum weight. $\mathrm{OptEF}_d^k$ and $\mathrm{OptVF}_d^k$ denote minimum-weight $k$-edge-connected and $k$-vertex-connected $d$-factors, respectively.

We have $w(\mathrm{OptF}_d) \leq w(\mathrm{OptEF}_d^k) \leq w(\mathrm{OptVF}_d^k)$ since every $k$-vertex-connected graph is also $k$-edge-connected. Both $w(\mathrm{OptEF}_d^k)$ and $w(\mathrm{OptVF}_d^k)$ are monotonically increasing in $k$. Furthermore, $w(\mathrm{OptE}^k) \leq w(\mathrm{OptEF}_d^k)$ for every $d$ and $w(\mathrm{OptV}^k) \leq w(\mathrm{OptVF}_d^k)$ for every $d$.

We denote by MST a minimum-weight spanning tree of $G$.

## 1.2  Previous and Related Results

Without the triangle inequality, the problem of computing minimum-weight $k$-vertex-connected spanning subgraphs can be approximated with a factor of $O(\log k)$ [3], and the problem of computing minimum-weight $k$-edge-connected spanning subgraphs can be approximated with a factor of 2 [16]. However, no approximation at all seems to be possible without the triangle inequality if we ask for specific degrees. This follows from the inapproximability of non-metric TSP [25, Sect. 2.4].

With the triangle inequality, we obtain the same factor of 2 for $k$-edge-connected subgraphs of minimum weight without degree requirements [16]. For $k$-vertex-connected spanning subgraphs of minimum weight without degree constraints, Kortsarz and Nutov [17] gave a $\left(2 + \frac{k-1}{n}\right)$-approximation algorithm.

Cornelissen et al. [5] gave a 2.5 approximation for Min-$d$Reg-2Edge for even $d$ and a 3-approximation for Min-$d$Reg-1Edge and Min-$d$Reg-2Edge for all odd $d$.

**Table 1.** Overview of approximation ratios for Min-$d$Reg-$k$Edge. Cases of odd $k$ and even $d$ are omitted as discussed in Sect. 1.1.

| $k$ | $d$ | ratio | reference |
|---|---|---|---|
| $= 2$ | $= 2$ | 1.5 | same problem as TSP [25, Sect. 2.4] |
| $= 1$ | odd | 3 | Cornelissen et al. [5] |
| $\geq 3$ | $= k$ | $2 + \frac{1}{k}$ | Chan et al. [1] |
| $\geq 2$ | $\geq k$, even | 2.5 | Theorem 2.13 ($k = 2$ by Cornelissen et al. [5]) |
| $\geq 2$ | $\geq k + 1$, odd | $4 - \frac{3}{k}$ | Theorem 2.13 |

**Table 2.** Overview of approximation ratios for Min-$d$Reg-$k$Vertex.

| $k$ | $d$ | ratio | reference |
|---|---|---|---|
| $\in \{1, 2\}$ | $= 2$ | 1.5 | same as problem as TSP [25, Sect. 2.4] |
| $\in \{1, 2, 3\}$ | $= 3$ | | same as Min-$d$Reg-$k$Edge |
| $\geq 2$ | $= k$ | $2 + \frac{k-1}{n} + \frac{1}{k}$ | Chan et al. [1] |
| $\geq 2$ | $= 2k - 1$ | $5 + \frac{2\frac{k}{n}-2}{n} + \frac{2}{k}$ | Theorem 3.2 |
| $\geq 2$ | $\geq 2k$ | $5 + \frac{2k\frac{n}{n}-2}{n}$ | Corollary 3.3 |

Also Min-$k$Reg-$k$Vertex and Min-$k$Reg-$k$Edge admit constant factor approximations for all $k \geq 1$ [1]. We refer to Tables 1 and 2 for an overview.

Fukunaga and Nagamochi [8] considered the problem of finding a minimum-weight $k$-edge-connected spanning subgraph with given degree requirements. Different from the problem that we consider, they allow multiple edges between vertices. This considerably simplifies the problem as one does not have to take care to avoid multiple edges when constructing the approximate solution. For this relaxed variant of the problem, they obtain approximation ratios of 2.5 for even $k$ and $2.5 + \frac{1.5}{k}$ for odd $k$ if the minimum degree requirement is at least 2. We remark that, although an optimal solution with multiple edges cannot be heavier than an optimal solution without multiple edges, an approximation algorithm for the variant with multiple edges does not imply an approximation algorithm for the variant without multiple edges and vice versa.

In many cases of algorithms for network design with degree constraints, only bounds on the degrees are given or some violation of the degree requirements is allowed to simplify the problem. Fekete et al. [7] devised an approximation algorithm for the bounded-degree spanning tree problem. Given lower and upper bounds for the degree of every vertex, spanning trees can be computed that violate every degree constraint by at most 1 and whose weight is no more than the weight of an optimal solution [22]. Often, network design problems are considered as bicriteria problems, where the goal is to simultaneously minimize the total costs and the violation of the degree requirements [9,10,18–20]. In contrast, our goal is to meet the degree requirements exactly.

### 1.3  Our Contribution

We devise polynomial-time approximation algorithms for Min-$d$Reg-$k$Edge (Sect. 2) and for Min-$d$Reg-$k$Vertex (Sect. 3). This answers an open question raised by Cornelissen et al. [5]. Our algorithms can be generalized to arbitrary degree sequences, as long as the minimum degree requirement is at least $2\lceil k/2 \rceil$ for edge connectivity or at least $2k - 1$ for vertex connectivity (Sect. 4).

   We obtain an approximation ratio of $4 - \frac{3}{k}$ for Min-$d$Reg-$k$Edge for odd $d \geq k + 1$, a ratio of 2.5 for Min-$d$Reg-$k$Edge for even $d \geq k$, and an approximation ratio of about 5 for Min-$d$Reg-$k$Vertex for $d \geq 2k - 1$. The precise approximation ratios are summarized in Tables 1 and 2.

   As far as we are aware, there do not exist any approximation results for the problem of finding subgraphs with exact degree requirements besides simple connectivity and 2-edge-connectivity [5]. The only exception that we are aware of is the work by Fukunaga and Nagamochi [8]. However, they allow multiple edges in their solutions, which seems to make the problem simpler to approximate.

   The high-level ideas of our algorithms are as follows. For edge-connectivity, our initial idea was to iteratively increase the connectivity from $k - 1$ to $k$ by considering the $k$-edge-connected components of the current solution and adding edges carefully. However, this does not work as $k$-edge-connected components are not guaranteed to exist in $(k - 1)$-edge-connected graphs. Instead, we introduce $k$-special components (Definition 2.1). By connecting the $k$-special components carefully, we can increase the edge-connectivity of the graph (Lemma 2.8). Every increase of the edge-connectivity costs at most $O(1/k)$ times the weight of the optimal solution (Lemma 2.10), yielding constant factor approximations for all $k$. Our algorithm for Min-$d$Reg-$k$Edge generalizes the algorithm of Cornelissen et al. [5] to arbitrary $k$. A more careful analysis yields that already their algorithm achieves an approximation ratio of 2.5 for Min-$d$Reg-2Edge also for odd $d$.

   For vertex-connectivity, the idea is to compute a $k$-vertex-connected $k$-regular graph and a (possibly not connected) $d$-factor. We iteratively add edges from the $k$-vertex-connected graph to the $d$-factor while maintaining the degrees until we obtain a $k$-vertex-connected $d$-factor. This works for $d \geq 2k - 1$ (Lemma 3.1).

## 2   Edge-Connectivity

In this section, we present an approximation algorithm for Min-$d$Reg-$k$Edge for all combinations of $d$ and $k$, provided that $d \geq 2\lceil k/2 \rceil$. This means that the algorithm works for all $d \geq k$ with the only exception being the case of odd $d = k$.

   The main idea of our algorithm is as follows: We start by computing a $d$-factor. Then we iteratively increase the connectivity as follows: First, we identify edges that we can safely remove without decreasing the connectivity. Second, we find edges that we can add in order to increase the connectivity while repairing the $d$-regularity that we have destroyed in the first step.

   One might be tempted to use the $k$-edge-connected components of the $d$-factor in order to increase the edge-connectivity from $k - 1$ to $k$. The catch is

that there need not be enough $k$-edge-connected components, and it is in fact possible to find $(k-1)$-edge-connected graphs that are $d$-regular with $d \geq k$ without any $k$-edge-connected component. To circumvent this problem, we introduce the notion of $k$-special components, which have the desired properties.

## 2.1   Graph-Theoretic Preparation

Different from the rest of the paper, the graph $G = (V, E)$ is not necessarily complete in this section. The following definition of $k$-special components is crucial for the whole Sect. 2.

**Definition 2.1.** *Let $k \in \mathbb{N}$, and let $G = (V, E)$ be a graph. We call $L \subseteq V$ a k-special component in G if $\mathrm{cut}_G(L) \leq k - 1$ and $L$ is locally $k$-edge connected in $G$.*

For $k = 1$, the $k$-special components are the connected components of $G$. The 2-edge-connected components of a graph yield a tree with a vertex for every 2-edge-connected component and an edge between any 2-edge-connected components that are connected by an edge. The 2-special components of $G$ correspond to the leaves of this tree.

Let us collect some facts about $k$-special components.

**Lemma 2.2.** *Let $G$ have a minimum degree of at least $k$, and let $L$ be a $k$-special component in $G$. Then $|L| \geq k + 1$.*

**Lemma 2.3.** *Let $G$ be a graph. If $L$ is a $k$-special component, then $L$ is a maximal locally $k$-edge-connected component. If $L$ and $L'$ are $k$-special, then either $L = L'$ or $L \cap L' = \emptyset$.*

The following crucial lemma shows the existence of $k$-special components.

**Lemma 2.4.** *Let $k \geq 1$. Let $G = (V, E)$ be a $(k - 1)$-edge-connected graph. Then every non-empty vertex set $X \subsetneq V$ either contains a $k$-special component or satisfies $\mathrm{cut}_G(X) \geq k$.*

The purpose of the next few lemmas is to show that we can always remove an edge from a $k$-special component without decreasing the connectedness of the whole graph. In the following, let $m = \lceil k/2 \rceil + 1$. It turns out that the graph induced by a $k$-special component contains a locally $m$-edge-connected component.

**Lemma 2.5.** *Let $k \geq 1$. Let $G = (V, E)$ be a $(k - 1)$-edge-connected graph of minimum degree at least $2\lceil k/2 \rceil$, and let $L$ be a $k$-special component of $G$. Then there exists an $X \subseteq L$ such that $X$ is a locally $m$-edge-connected component in $L$ and $|X| \geq k + 1$.*

The edges $\{u_i, v_i\}$ mentioned in the next lemma are the edges that we can safely remove. The resulting graph will remain $(k-1)$-edge-connected according to Lemma 2.7. The vertices $u_i$ and $v_i$ in the next lemma will be chosen from $X_i \subseteq L_i$, where $X_i$ is a locally $m$-edge-connected component in $L_i$ as in Lemma 2.5.

**Lemma 2.6.** *Let $k \geq 1$. Let $G = (V, E)$ be a $(k-1)$-edge-connected graph of minimum degree at least $2\lceil k/2 \rceil$. Let $L_1, \ldots, L_s$ be the $k$-special components of $G$. Then there exist vertices $u_i, v_i \in L_i$ for all $i \in \{1, \ldots, s\}$ such that the following properties are met:*

- *$\{u_i, v_i\} \in E$ for all $i$.*
- *$\{u_i, v_j\} \notin E$ for all $i \neq j$.*
- *There exist at least $m$ edge-disjoint paths from $u_i$ to $v_i$ in the graph induced by $L_i$ for every $i$.*

**Lemma 2.7.** *Let $G = (V, E)$ be a $(k-1)$-edge-connected graph of minimum degree at least $2\lceil k/2 \rceil$ with $k$-special components $L_1, \ldots, L_s$, and let $u_1, \ldots, u_s$ and $v_1, \ldots, v_s$ be chosen as in Lemma 2.6. Let $Q = \big\{ \{u_i, v_i\} \mid 1 \leq i \leq s \big\}$. Then $G - Q$ is $(k-1)$-edge-connected.*

By removing the edges $\{u_i, v_i\} \in Q$ and adding the edges $\{u_i, v_{i+1}\} \in S$, we construct a $k$-edge-connected graph from the $(k-1)$-edge-connected graph $G$ according to the following lemma.

**Lemma 2.8.** *Let $G = (V, E)$ be a $(k-1)$-edge-connected graph of minimum degree at least $2\lceil k/2 \rceil$ with $k$-special components $L_1, \ldots, L_s$, and let $u_1, \ldots, u_s$ and $v_1, \ldots, v_s$ be chosen as in Lemma 2.6. Let $Q = \big\{ \{u_i, v_i\} \mid 1 \leq i \leq s \big\}$, and let $S = \big\{ \{u_i, v_{i+1}\} \mid 1 \leq i \leq s \big\}$, where arithmetic is modulo $s$.*
   *Then the graph $\tilde{G} = G - Q + S$ is $k$-edge-connected.*

To conclude this section, we remark that the $k$-special components of a graph can be found in polynomial-time: local $k$-edge-connectedness can be tested in polynomial time. Thus, we can find locally $k$-edge-connected components in polynomial time. Since $k$-special components are maximal locally $k$-edge-connected components, we just have to compute a partition of the graph into locally $k$-edge-connected components and check whether less than $k$ edges leave such a component. Therefore, the sets $L_i$ and $X_i \subseteq L_i$ as well as the vertices $u_i$ and $v_i$ with the properties as in Lemmas 2.5 and 2.6 can be computed in polynomial time.

## 2.2  Algorithm and Analysis

Our approximation algorithm for Min-$d$Reg-$k$Edge (Algorithm 1) starts with an $\ell$-edge-connected $d$-factor $F_\ell$. How we choose $\ell$ and compute $F_\ell$ depends on the parity of $k$, but it is possible that improved approximation algorithms for certain small $k$ lead to other initializations. (If $k$ is even, we use $\ell = 0$ and $F_0 = \text{OptF}_d$. If $k$ is odd, we use $\ell = 2$ and approximate a 2-edge-connected $d$-factor $F_2$ using the algorithm of Cornelissen et al. [5].)
   Then it iteratively uses a subroutine (Algorithm 2) that increases the connectivity. To increase the connectivity, we compute a TSP tour (line 3). We do this using Christofides' algorithm [25, Sect. 2.4], which achieves an approximation ratio of 1.5.

---

**input**  : undirected complete graph $G = (V, E)$, edge weights $w$, integers $k \geq 3$,
        $d \geq 2\lceil k/2 \rceil$
**output**: $k$-edge-connected $d$-factor $R$ of $G$
1  compute a minimum-weight $\ell$-edge-connected $d$-factor $F_\ell$ (or an approximation)
2  **for** $p \leftarrow \ell + 1, \ldots, k$ **do**
3      **if** $F_{p-1}$ *is not p-edge-connected* **then**
4          apply Algorithm 2 to obtain $F_p$
5      **else**
6          $F_p \leftarrow F_{p-1}$
7      **end**
8  **end**
9  $R \leftarrow F_k$

**Algorithm 1.** Approximation algorithm for Min-$d$Reg-$k$Edge.

---

**input**  : undirected complete graph $G = (V, E)$, edge weights $w$, integer $p \geq 1$,
        $(p-1)$-edge-connected subgraph $F_{p-1}$ of $G$ with minimum degree at
        least $p + 1$
**output**: $p$-edge-connected subgraph $F_p$ of $G$ with the same degree at every
        vertex as $F_{p-1}$
1  find the $p$-special components of $F_{p-1}$; let $L_1, \ldots, L_s$ be these $p$-special
   components
2  find vertices $u_i, v_i \in L_i$ for all $i \in \{1, \ldots, s\}$ with the properties stated in
   Lemma 2.6; $Q \leftarrow \big\{\{u_i, v_i\} \mid 1 \leq i \leq s\big\}$
3  compute a TSP tour $T$ on $V$ using Christofides' algorithm
4  take shortcuts to obtain a tour $T'$ on $u_1, \ldots, u_s$ (without loss of generality in
   this order)
5  $S \leftarrow \big\{\{u_i, v_{i+1}\} \mid 1 \leq i \leq s\big\}$ (arithmetic modulo $s$)
6  $F_p \leftarrow F_{p-1} - Q + S$

**Algorithm 2.** Increasing the edge-connectivity of a graph by 1 while maintaining $d$-regularity.

We analyze correctness and approximation ratio using a series of lemmas.

**Lemma 2.9.** *Let $k \geq 1$ be arbitrary, and let $p \in \{\ell, \ell + 1, \ldots, k\}$. Let $F_p$ be computed by Algorithm 1. Then $F_p$ is d-regular and p-edge-connected.*

In order to analyze the approximation ratio and to achieve a constant approximation for all $k$, we exploit a result that Fukunaga and Nagamochi [8] attributed to Goemans and Bertsimas [12] and Wolsey [26].

**Lemma 2.10 (Fukunaga, Nagamochi [8], Theorem 2]).** *Let $T$ be the TSP tour obtained from Christofides' algorithm. Then $w(T) \leq \frac{3}{k} \cdot w(\mathrm{OptE}^k)$.*

A consequence of Lemma 2.10 are the following two statements, which we need to analyze the approximation ratio.

**Lemma 2.11.** *If, in Algorithm 1, we enter line 4 and call Algorithm 2, then*

$$w(F_p) \leq \frac{3}{k} \cdot w(\mathrm{OptEF}_d^k) + w(F_{p-1}).$$

**Lemma 2.12.** *If Algorithm 1 calls Algorithm 2 $q$ times, then*

$$w(F_k) \leq \frac{3q}{k} \cdot w(\mathrm{OptEF}_d^k) + w(F_\ell).$$

**Theorem 2.13.** *For $k \geq 2$ and $d \geq 2\lceil k/2 \rceil$, Algorithm 1 is a polynomial-time approximation algorithm for* Min-$d$Reg-$k$Edge. *It achieves an approximation ratio of 2.5 for even $d$ and an approximation ratio of $4 - \frac{3}{k}$ for odd $d$.*

Algorithm 1 works also for the case of even $d = k$, but there exists already an approximation algorithm with a ratio of $2 + \frac{1}{k}$ for this special case [1]. Note that the proof of Theorem 2.13 does not cover the case of odd $d$ and $k = 1$, but it is already known that this case can be approximated with a factor of 3 [5].

## 3   Vertex Connectivity

In this section, we consider Min-$d$Reg-$k$Vertex for $d \geq 2k - 1$. The basis of the algorithm (Algorithm 3) is the following: Assume that we have a $k$-vertex connected $k$-factor $H$ and a $d$-factor $F$ that lacks $k$-vertex-connectedness. Then we iteratively add edges from $H$ to $F$ to make $F$ $k$-vertex-connected as well. More precisely, we try to add an edge $e \in H \setminus F$ to increase the connectivity of $F$. To maintain that $F$ is $d$-regular, we have to add another edge and remove two edges of $F$. If, in the course of this process, we never have to remove an edge of $H$ from $F$, then the algorithm terminates with a $k$-vertex-connected $d$-regular graph.

In Algorithm 3, the initial $d$-factor $\mathrm{OptF}_d$ can be computed in polynomial time (line 1). Kortsarz and Nutov showed that we can compute a $k$-vertex-connected spanning subgraph $K$ whose total weight is at most a factor of $2 + \frac{k-1}{n}$ larger than the weight of a $k$-vertex-connected graph of minimum weight (line 2). Chan et al. [1] devised an algorithm that turns $k$-vertex-connected graphs $K$ into $k$-regular $k$-vertex-connected graphs $H$ at the expense of an additive $w(\mathrm{OptV}^k)/k$.

With this initialization, we iteratively add edges from $H$ to $F$ while maintaining $d$-regularity of $F$. This works as long as $d$ is sufficiently large according to the following lemma. We parametrize the maximum degree by $\ell$ in order to be able to get a slight improvement for larger $d$ (Corollary 3.3).

**Lemma 3.1.** *Let $k, \ell \geq 2$ and $d \geq k + \ell - 1$. Let $G = (V, E)$ be an undirected complete graph. Let $F$ be a $d$-factor of $G$, and let $H$ be a $k$-vertex-connected graph subgraph of $G$ that has a maximum degree of at most $\ell$. Assume that $F$ is not $k$-vertex-connected.*

*Then there exists an edge $e = \{u_1, u_2\} \in H \setminus F$ such that $u_1$ and $u_2$ are not connected via $k$ vertex-disjoint paths in $F$. Furthermore, given such an edge $e = \{u_1, u_2\}$, there exist vertices $v_1, v_2 \in V$ with $v_1 \neq v_2$ and the following properties:*

---

**input** : undirected complete graph $G = (V, E)$, edge weights $w$, integers $k \geq 2$,
$\quad\quad\quad d \geq 2k - 1$,
**output**: $k$-vertex-connected $d$-factor $R$ of $G$

**1** $F \leftarrow \text{OptF}_d$
**2** approximate a $k$-vertex connected graph $K$ using the algorithm of Kortsarz and
$\quad$ Nutov [17]
**3** compute a $k$-vertex-connected $k$-factor $H$ from $K$ using the algorithm of Chan
$\quad$ et al. [1]
**4 while** $F$ is not $k$-vertex-connected **do**
**5** $\quad$ select an edge $e = \{u_1, u_2\} \in H \setminus F$ such that $u_1$ and $u_2$ are not connected
$\quad\quad$ by $k$ vertex-disjoint paths in $F$
**6** $\quad$ choose vertices $v_1, v_2$ with $\{u_1, v_1\}, \{u_2, v_2\} \in F \setminus H$ and $\{v_1, v_2\} \notin F$
**7** $\quad$ $F \leftarrow \left( F \setminus \{\{u_1, v_1\}, \{u_2, v_2\}\} \right) \cup \{\{u_1, u_2\}, \{v_1, v_2\}\}$
**8 end**
**9** $R \leftarrow F$

---

**Algorithm 3.** Approximation algorithm for Min-$d$Reg-$k$Vertex for $d \geq 2k - 1$.

1. $\{u_1, v_1\}, \{u_2, v_2\} \in F \setminus H$.
2. $\{v_1, v_2\} \notin F$.

With this lemma, we can prove the main result of this section.

**Theorem 3.2.** *For $k, d \in \mathbb{N}$ with $k \geq 2$ and $d \geq 2k - 1$, Algorithm 3 is a polynomial-time approximation algorithm for* Min-$d$Reg-$k$Vertex *with an approximation ratio of $5 + \frac{2k-2}{n} + \frac{2}{k}$.*

Algorithm 3 also works for $k = 1$, but there already exist better approximation algorithms for this case (see Table 2). With the slightly stronger assumption $d \geq 2k$, we can get a slightly better approximation ratio.

**Corollary 3.3.** *For $k, d \in \mathbb{N}$ with $k \geq 2$ and $d \geq 2k$, there exists a polynomial-time approximation algorithm for* Min-$d$Reg-$k$Vertex *with an approximation ratio of $5 + \frac{2k-2}{n}$.*

## 4  Generalization to Arbitrary Degree Sequences

Both algorithms of Sects. 2 and 3 do not exploit $d$-regularity, but only that the degree of each vertex is at least $d$. Thus, we immediately get approximation algorithms for Min-$d$Gen-$k$Edge and Min-$d$Gen-$k$Vertex, where we get a degree requirement of at least $d$ for each vertex.

For $k$-edge-connectedness, we require that the minimum degree requirement is at least $2\lceil k/2 \rceil$.

**Theorem 4.1.** *For $k \geq 2$,* Min-$(2\lceil \frac{k}{2} \rceil)$Gen-$k$Edge *can be approximated in polynomial time with an approximation ratio of $4 - \frac{3}{k}$.*

For $k$-vertex-connectivity, we require that the minimum degree requirement is at least $2k - 1$. (For minimum degree at least $2k$, we get a small improvement similarly to Corollary 3.3.)

**Theorem 4.2.** *For $k \geq 2$, Min-$(2k-1)$Gen-$k$Vertex can be approximated in polynomial time with an approximation ratio of $5 + \frac{2k-2}{n} + \frac{2}{k}$.*

*Min-$(2k)$Gen-$k$Vertex can be approximated in polynomial time with an approximation ratio of $5 + \frac{2k-2}{n}$.*

## 5   Conclusions and Open Problems

We conclude this paper with two questions for further research.

First, for edge-connectivity, we require $d \geq 2\lceil k/2 \rceil$. Since there exists an approximation algorithm for Min-$k$Reg-$k$Edge (for $k \geq 2$) [1], the only case for which it is unknown if a constant factor approximation algorithm exists is the generalized problem Min-$k$Gen-$k$Edge for odd values of $k$. We are particularly curious about approximation algorithms for Min-1Gen-1Edge, where we want to find a cheap connected graph with given vertex degrees. To get such algorithms, vertices with degree requirement 1 seem to be bothersome. (This seems to be a more general phenomenon in network design, as, for instance, the approximation algorithms by Fekete et al. [7] for bounded-degree spanning trees and by Fukunaga and Nagamochi [8] for $k$-edge-connected subgraphs with multiple edges both require that the minimum degree requirement is at least 2.) Still, we conjecture that constant factor approximation algorithms exist for these problems as well.

Second, we would like to see constant factor approximation algorithms for Min-$d$Reg-$k$Vertex for the case $k + 1 \leq d \leq 2k - 2$ and for the general problem Min-$d$Gen-$k$Vertex for $k \leq d \leq 2k - 2$. We conjecture that constant factor approximation algorithms exist for these problems.

## References

1. Chan, Y.H., Fung, W.S., Lau, L.C., Yung, C.K.: Degree bounded network design with metric costs. SIAM J. Comput. **40**(4), 953–980 (2011)
2. Cheah, F., Corneil, D.G.: The complexity of regular subgraph recognition. Discrete Appl. Math. **27**(1–2), 59–68 (1990)
3. Cheriyan, J., Vempala, S., Vetta, A.: An approximation algorithm for the minimum-cost $k$-vertex connected subgraph. SIAM J. Comput. **32**(4), 1050–1055 (2003)
4. Cheriyan, J., Vetta, A.: Approximation algorithms for network design with metric costs. SIAM J. Discrete Math. **21**(3), 612–636 (2007)
5. Cornelissen, K., Hoeksma, R., Manthey, B., Narayanaswamy, N.S., Rahul, C.S.: Approximability of connected factors. In: Kaklamanis, C., Pruhs, K. (eds.) WAOA 2013. LNCS, vol. 8447, pp. 120–131. Springer, Heidelberg (2014)
6. Czumaj, A., Lingas, A.: Minimum $k$-connected geometric networks. In: Kao, M.Y. (ed.) Encyclopedia of Algorithms, pp. 536–539. Springer, Heidelberg (2008)

7. Fekete, S.P., Khuller, S., Klemmstein, M., Raghavachari, B., Young, N.E.: A network-flow technique for finding low-weight bounded-degree spanning trees. J. Algorithms **24**(2), 310–324 (1997)
8. Fukunaga, T., Nagamochi, H.: Network design with edge-connectivity and degree constraints. Theor. Comput. Syst. **45**(3), 512–532 (2009)
9. Fukunaga, T., Nagamochi, H.: Network design with weighted degree constraints. Discrete Optim. **7**(4), 246–255 (2010)
10. Fukunaga, T., Ravi, R.: Iterative rounding approximation algorithms for degree-bounded node-connectivity network design. In: Proceedings of the 53rd Annual Symposium on Foundations of Computer Science (FOCS), pp. 263–272. IEEE Computer Society (2012)
11. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman and Company, New York (1979)
12. Goemans, M.X., Bertsimas, D.: Survivable networks, linear programming relaxations and the parsimonious property. Math. Program. **60**, 145–166 (1993)
13. Kammer, F., Täubig, H.: Connectivity. In: Brandes, U., Erlebach, T. (eds.) Network Analysis. LNCS, vol. 3418, pp. 143–177. Springer, Heidelberg (2005)
14. Khandekar, R., Kortsarz, G., Nutov, Z.: On some network design problems with degree constraints. J. Comput. Syst. Sci. **79**(5), 725–736 (2013)
15. Khuller, S., Raghavachari, B.: Graph connectivity. In: Kao, M.Y. (ed.) Encyclopedia of Algorithms, pp. 371–373. Springer, Heidelberg (2008)
16. Khuller, S., Vishkin, U.: Biconnectivity approximations and graph carvings. J. ACM **41**(2), 214–235 (1994)
17. Kortsarz, G., Nutov, Z.: Approximating node connectivity problems via set covers. Algorithmica **37**(2), 75–92 (2003)
18. Lau, L.C., Naor, J., Salavatipour, M.R., Singh, M.: Survivable network design with degree or order constraints. SIAM J. Comput. **39**(3), 1062–1087 (2009)
19. Lau, L.C., Singh, M.: Additive approximation for bounded degree survivable network design. SIAM J. Comput. **42**(6), 2217–2242 (2013)
20. Lau, L.C., Zhou, H.: A unified algorithm for degree bounded survivable network design. In: Lee, J., Vygen, J. (eds.) IPCO 2014. LNCS, vol. 8494, pp. 369–380. Springer, Heidelberg (2014)
21. Lovász, L., Plummer, M.D.: Matching Theory, North-Holland Mathematics Studies, vol. 121. Elsevier (1986)
22. Singh, M., Lau, L.C.: Approximating minimum bounded degree spanning trees to within one of optimal. J. ACM **62**(1), 1:1–1:19 (2015)
23. Tutte, W.T.: A short proof of the factor theorem for finite graphs. Can. J. Math. **6**, 347–352 (1954)
24. West, D.B.: Introduction to Graph Theory, 2nd edn. Prentice-Hall, Upper Saddle River (2001)
25. Williamson, D.P., Shmoys, D.B.: The Design of Approximation Algorithms. Cambridge University Press, New York (2011)
26. Wolsey, L.A.: Heuristic analysis, linear programming and branch and bound. In: Rayward-Smith, V.J. (ed.) Combinatorial Optimization II, Mathematical Programming Studies, vol. 13, pp. 121–134. Springer, Heidelberg (1980)