Theory Comput Syst
DOI 10.1007/s00224-016-9723-z

# Approximation Algorithms for Connected Graph Factors of Minimum Weight

**Kamiel Cornelissen[1] · Ruben Hoeksma[2] · Bodo Manthey[1] · N. S. Narayanaswamy[3] · C. S. Rahul[3] · Marten Waanders[1]**

**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). To complement our approximation algorithms, we prove

Results of this paper have been presented at the 11th Workshop on Approximation and Online Algorithms (WAOA 2013) [5] and at the 13th Workshop on Approximation and Online Algorithms (WAOA 2015) [22].

Bodo Manthey
b.manthey@utwente.nl

Kamiel Cornelissen
k.cornelissen@utwente.nl

Ruben Hoeksma
rubenh@dii.uchile.cl

N. S. Narayanaswamy
swamy@cse.iitm.ac.in

C. S. Rahul
rahulcs@cse.iitm.ac.in

[1] University of Twente, Enschede, The Netherlands

[2] University of Chile, Santiago, Chile

[3] Indian Institute of Technology Madras, Chennai, India

that the problem with simple connectivity cannot be approximated better than the traveling salesman problem. In particular, the problem is APX-hard.

**Keywords** Graph factors · Edge-connectivity · Vertex-connectivity · Approximation algorithms

## 1 Introduction

The traveling salesman problem (Min-TSP) is a basic combinatorial optimization problem: given a complete graph $G = (V, E)$ with edge weights that satisfy the triangle inequality, the goal is to find a Hamiltonian cycle of minimum total weight. Phrased differently, we are looking for a subgraph of $G$ of minimum weight that is 2-regular, connected, and spanning. While Min-TSP is NP-hard [11, ND22], omitting the requirement that the subgraph must be connected makes the problem polynomial-time solvable [21, 27]. In general, $d$-regular, spanning subgraphs (also called $d$-factors) of minimum weight can be found in polynomial time using Tutte's reduction [21, 27] to the matching problem. Cheah and Corneil [2] have shown that deciding whether a given graph $G = (V, E)$ has a $d$-regular connected spanning subgraph is NP-complete for every $d \geq 2$, where $d = 2$ is just the Hamiltonian cycle problem [11, GT37]. Thus, finding a connected $d$-factor of minimum weight is also NP-hard for all $d$.

The problem of finding connected $d$-factors of minimum weight is a fundamental problem in network design, where 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, 6–10, 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.

### 1.1 Problem Definitions and Preliminaries

#### *1.1.1 Graphs and Connectivity*

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

For a subset $X \subseteq V$ of 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 $\mathrm{cut}_G(X) \geq k$ 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 subgraph 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 of $G$, 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 $u$ and $v$ in $G$.

We note that testing if two vertices are locally $k$-edge-connected, if a graph is $k$-edge-connected, or if a graph is $k$-vertex-connected can be done in polynomial time. 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*. Instead of using the same value $d$ for all vertices, we also consider spanning subgraphs where the degree of each vertex $v$ is required to be $d_v$ (Section 4).

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

### 1.1.2 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:

- The two problems Min-$d$Reg-1Edge and Min-$d$Reg-1Vertex are identical for all $d$ since 1-edge-connectedness and 1-vertex-connectedness are simply connectedness.
- For $k \in \{1, 2\}$, the problems Min-2Reg-$k$Edge and Min-2Reg-$k$Vertex are identical to the traveling salesman problem (Min-TSP).

- 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 [28, Theorem 4.1.11].

We also consider the generalizations of 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 adjacent 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 Sections 2 and 3 to Min-$d$Reg-$k$Vertex and Min-$d$Reg-$k$Edge, and we state the generalized results for Min-$d$Gen-$k$Vertex and Min-$d$Gen-$k$Edge only in Section 4.

We use the following notation: $\mathrm{OptE}^k$ denotes a $k$-edge-connected subgraph of minimum weight. $\mathrm{OptV}^k$ denotes a $k$-vertex-connected 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 within a factor of $O(\log k)$ [3], and the problem of computing minimum weight $k$-edge-connected spanning subgraphs can be approximated with in 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 [29, Section 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, Theorem 4.2] gave a $\left(2 + \frac{k-1}{n}\right)$-approximation algorithm.

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 of results on $k$-vertex-connected and $k$-edge-connected $d$-factors.

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.

**Table 1** Overview of approximation ratios for Min-$d$Reg-$k$Vertex

| $k$ | $d$ | ratio | reference |
|---|---|---|---|
| $\in \{1,2\}$ | $= 2$ | 1.5 | same as problem as Min-TSP [29, Section 2.4] |
| $\in \{1,2,3\}$ | $= 3$ | | same as Min-$d$Reg-$k$Edge |
| $= 1$ | arbitrary | | same as Min-$d$Reg-1Edge |
| $\geq 2$ | $= k$ | $2 + \frac{k-1}{n} + \frac{1}{k}$ | Chan et al. [1] |
| $\geq 2$ | $= 2k - 1$ | $5 + \frac{2k-2}{n} + \frac{2}{k}$ | Theorem 2.2 |
| $\geq 2$ | $\geq 2k$ | $5 + \frac{2k-2}{n}$ | Corollary 2.3 |

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 [26]. 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.

Recently [23], Min-$d$Reg-1Edge has been considered for the case that $d$ grows with the number $n$ of vertices. It turns out that the problem becomes simpler for large enough $d$, admitting a PTAS for $d \geq n/c$ for any constant $c$.

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

| $k$ | $d$ | ratio | reference |
|---|---|---|---|
| $= 2$ | $= 2$ | 1.5 | same problem as Min-TSP [29, Section 2.4] |
| $\geq 3$ | $= k$ | $2 + \frac{1}{k}$ | Chan et al. [1] |
| $= 1$ | odd | 3 | Theorem 3.23 |
| $\geq 2$ | $\geq k$, even | 2.5 | Theorem 3.23 |
| $\geq 2$ | $\geq k + 1$, odd | $4 - \frac{3}{k}$ | Theorem 3.23 |

Cases of odd $k$ and even $d$ are omitted as discussed in Section 1.1.2

### 1.3 Our Contribution

We devise polynomial-time approximation algorithms for Min-$d$Reg-$k$Vertex (Section 2) and for Min-$d$Reg-$k$Edge (Section 3). Our algorithms can be generalized to arbitrary degree sequences, as long as the minimum degree requirement is at least $2k-1$ for vertex connectivity or at least $2\lceil k/2 \rceil$ for edge connectivity (Section 4).

Roughly, we obtain an approximation ratio of about 5 for Min-$d$Reg-$k$Vertex for $d \geq 2k-1$, an approximation ratio of roughly 4 for Min-$d$Reg-$k$Edge for odd $d \geq k+1$ and a ratio of 2.5 for Min-$d$Reg-$k$Edge for even $d \geq k$. 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. 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 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 2.1).

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 3.2). By connecting the $k$-special components carefully, we can increase the edge-connectivity of the graph (Lemma 3.11). Every increase of the edge-connectivity costs at most a fraction $O(1/k)$ of the weight of the optimal solution (Lemma 3.18), yielding constant factor approximations for all $k$.

Finally, we prove that Min-$d$Reg-1Edge is APX-hard. We extend this result and the NP-hardness of finding connected $d$-factors to the case where $d$ grows with the number $n$ of vertices.

## 2 Vertex Connectivity

In this section, we consider Min-$d$Reg-$k$Vertex for $d \geq 2k-1$. The basis of the algorithm (Algorithm 1) 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 1, the initial $d$-factor $\mathrm{OptF}_d$ can be computed in polynomial time (line 1) by Tutte's reduction to the matching problem [21, 27]. 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 $w(\mathrm{OptV}^k)/k$.

**Algorithm 1** $\left(5 + \frac{2k-2}{n} + \frac{2}{k}\right)$-approximation for Min-$d$Reg-$k$Vertex for $d \geq 2k-1$

```
input : undirected complete graph $G = (V, E)$, edge weights $w$, integers $k \geq 2$,
        $d \geq 2k - 1$
output: $k$-vertex-connected $d$-factor $R$ of $G$
1 $F \leftarrow \mathrm{OptF}_d$
2 approximate a $k$-vertex connected graph $K$ using the algorithm of Kortsarz and
  Nutov [17, Theorem 4.2]
3 compute a $k$-vertex-connected $k$-factor $H$ from $K$ using the algorithm of Chan et al. [1]
4 while $F$ is not $k$-vertex-connected do
5     select an edge $e = \{u_1, u_2\} \in H \setminus F$ such that $u_1$ and $u_2$ are not connected by $k$
      vertex-disjoint paths in $F$
6     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     $F \leftarrow \big(F \setminus \{\{u_1, v_1\}, \{u_2, v_2\}\}\big) \cup \{\{u_1, u_2\}, \{v_1, v_2\}\}$
8 end
9 $R \leftarrow F$
```

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 of $H$ by $\ell$ in order to be able to get a slight improvement for larger $d$ (Corollary 2.3).

**Lemma 2.1** *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 $G$ that is not $k$-vertex connected, and let $H$ be a $k$-vertex connected subgraph of $G$ that has a maximum degree of at most $\ell$.*

*Then there exists an edge $e = \{u_1, u_2\} \in H \setminus F$ and vertices $v_1, v_2 \in V$ with $v_1 \neq v_2$ with the following properties:*

1. *The vertices $u_1$ and $u_2$ are not connected via $k$ vertex-disjoint paths in $F$.*
2. $\{u_1, v_1\}, \{u_2, v_2\} \in F \setminus H$.
3. $\{v_1, v_2\} \notin F$.

*Proof* Since $F$ is not $k$-vertex-connected, there exists a subset $X \subseteq V$ of vertices with $|X| \leq k-1$ such that the subgraph $\tilde{G}_F$ of $F$ induced by $V \setminus X$ is not connected. Since $H$ is $k$-vertex-connected, the subgraph $\tilde{G}_H$ of $H$ induced by $V \setminus X$ is connected. This means that there exists an edge $e = \{u_1, u_2\}$ in $\tilde{G}_H$ that connects two different components of $\tilde{G}_F$. In particular, $e \in H \setminus F$ and $u_1$ and $u_2$ are not connected via $k$ vertex-disjoint paths in $F$.

Let $e_1^1, \ldots, e_d^1$ be the edges of $F$ incident to $u_1$. Since $H$ has a maximum degree of at most $\ell$ and the edge $e$ is incident to $u_1$, at most $\ell - 1$ of these $d$ edges are contained in $H$. Because of this and since $d \geq k + \ell - 1$, at least $k$ of these edges are not contained in $H$. Let $e_1^1, \ldots, e_k^1$ be $k$ such edges.

In the same way, let $e_1^2, \ldots, e_k^2$ be $k$ edges of $F \setminus H$ incident to $u_2$.

Let $e_i^j = \{u_j, z_{j,i}\}$. We call $z_{1,i}$ and $z_{2,i'}$ connected if $z_{1,i} = z_{2,i'}$ or if they are connected with an edge in $F$. If all endpoints $z_{1,1}, \ldots, z_{1,k}$ were connected to all endpoints $z_{2,1}, \ldots, z_{2,k}$, then this would give us $k$ vertex-disjoint paths from $u_1$ to $u_2$, contradicting the assumption of the lemma. Thus, there exist $e_i^1 = \{u_1, z_{1,i}\}$ and $e_{i'}^2 = \{u_2, z_{2,i'}\}$ such that $z_{1,i}$ and $z_{2,i'}$ are not connected. We set $v_1 = z_{1,i}$ and $v_2 = z_{2,i'}$. These vertices have the desired properties. $\square$

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

**Theorem 2.2** *For $k, d \in \mathbb{N}$ with $k \geq 2$ and $d \geq 2k-1$, Algorithm 1 is a polynomial-time approximation algorithm for* Min-$d$Reg-$k$Vertex *with an approximation ratio of* $5 + \frac{2k-2}{n} + \frac{2}{k}$.

*Proof* Because of Lemma 2.1 with $\ell = k$, we can always find vertices $v_1$ and $v_2$ as required in line 6, and such vertices can be found in polynomial time. Since no edge of $H$ is ever removed from $F$ and every iteration adds one or two edges of $H$ to $F$, the while loop runs through at most $|H|$ iterations. Thus, the overall running-time is bounded by a polynomial.

Let us analyze the approximation ratio. If we add an edge $e \in H$ to $F$, then we add in fact $e = \{u_1, u_2\}$ and $e' = \{v_1, v_2\}$. On the other hand, we remove $\{u_1, v_1\}$ and $\{u_2, v_2\}$ from $F$. By the triangle inequality, $w(e') \leq w(e) + w(\{u_1, v_1\}) + w(\{u_2, v_2\})$. Thus, if we add $e$ and $e'$, then the weight of $F$ increases by at most $2w(e)$. Therefore, the total weight of $R$ is bounded by $w(R) \leq w(\mathrm{OptF}_d) + 2w(H)$.

We have $w(\mathrm{OptF}_d) \leq w(\mathrm{OptVF}_d^k)$. Furthermore, we have $w(H) \leq w(K) + w(\mathrm{OptV}^k)/k$ [1]. Moreover, $w(K) \leq (2 + \frac{k-1}{n}) \cdot w(\mathrm{OptV}^k)$ [17] and $w(\mathrm{OptV}^k) \leq w(\mathrm{OptVF}_d^k)$. Thus,

$$\begin{aligned} w(R) &\leq w(\mathrm{OptF}_d) + 2w(H) \leq w(\mathrm{OptF}_d) + 2w(K) + \frac{2}{k} \cdot w(\mathrm{OptV}^k) \\ &\leq w(\mathrm{OptVF}_d^k) + \left(4 + \frac{2k-2}{n}\right) \cdot w(\mathrm{OptV}^k) + \frac{2}{k} \cdot w(\mathrm{OptV}^k) \\ &\leq \left(5 + \frac{2k-2}{n} + \frac{2}{k}\right) \cdot \mathrm{OptVF}_d^k. \end{aligned}$$

$\square$

Algorithm 1 also works for $k = 1$, but for this case, there already exist better approximation algorithms (see Table 1).

With the slightly stronger assumption $d \geq 2k$, we can get a slightly better approximation ratio.

**Corollary 2.3** *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* $5 + \frac{2k-2}{n}$.

*Proof* In line 3 of Algorithm 1, we compute a $k$-vertex-connected graph $H$ of maximum degree $k+1$ instead of a $k$-regular $k$-vertex-connected graph. According to Chan et al. [1], this can be done in polynomial time with $w(H) \leq w(K)$. Now we use Lemma 2.1 with $\ell = k+1$. □

## 3 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 $d = k$. It includes the case of simple connectivity, i.e., the case of $k = 1$.

The main idea of our algorithm is as follows: We start by computing a $d$-factor (without requiring any connectedness). 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 $k$. This works for $k = 1$ and $k = 2$. However, for larger $k$, 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 $d \geq k$ that do not contain any non-trivial $k$-edge-connected component. To circumvent this problem, we introduce the notion of $k$-special components, which have the desired properties.

### 3.1 Graph-Theoretic Preparation

Different from the rest of the paper, the graph $G = (V, E)$ is not necessarily complete in this section.

**Lemma 3.1** *Let $k \in \mathbb{N}$, and let $G = (V, E)$ be a graph of minimum degree at least $k$. Let $X \subseteq V$ be a non-empty subset of vertices. Then at least one of the following two properties hold:*

- $|X| \geq k+1$.
- $\mathit{cut}_G(X) \geq k$.

*Proof* Let $\ell = |X|$. Every vertex in $X$ can be adjacent to at most $|X|-1 = \ell-1$ other vertices of $X$. Since $G$ has a minimum degree of at least $k$, every vertex of $X$ must have at least $k-\ell+1$ neighbors outside of $X$. This shows $\mathrm{cut}_G(X) \geq (k-\ell+1)\cdot\ell$. For $\ell = 1$ and $\ell = k$, this last expression evaluates to $k$. Since it is a concave function of $\ell$, we have $\mathrm{cut}_G(X) \geq k$ for $1 \leq \ell \leq k$. □

The following definition of $k$-special components is crucial for the whole Section 3. As far as we are aware, this definition has not appeared yet in the literature.

**Definition 3.2** 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 $G$. For $k = 2$, the $k$-special components are the leaves of the following graph: we have a node for every 2-edge-connected component. Two nodes of this graph are connected if the corresponding 2-edge-connected components are connected via a single edge. This graph is a tree. The 2-special components of $G$ correspond to the leaves of this tree.

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

**Lemma 3.3** *Let $G$ have a minimum degree of at least $k$, and let $L$ be a $k$-special component in $G$. Then $|L| \geq k + 1$.*

*Proof* This follows immediately from Lemma 3.1 and Definition 3.2. □

**Lemma 3.4** *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 $L \cap L' = \emptyset$.*

*Proof* If $L$ were not maximal, then we would have $\mathrm{cut}_G(L) \geq k$. If $L$ and $L'$ intersect but are not identical, then $\mathrm{cut}_G(L) \geq k$ or $\mathrm{cut}_G(L') \geq k$. Hence, if $L$ and $L'$ intersect, then they must be identical. □

The following lemma is crucial as it proves the existence of $k$-special components.

**Lemma 3.5** *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 $\mathrm{cut}_G(X) \geq k$.*

*Proof* Assume to the contrary that there exists a set $X$ with $\mathrm{cut}_G(X) \leq k - 1$ that does not contain a $k$-special component. We choose $X$ minimal in the sense such that no non-empty proper subset $Y \subsetneq X$ with $\mathrm{cut}_G(Y) \leq k - 1$ does not contain a $k$-special component.

Now consider any $Y \subsetneq X$. If $Y$ contains a $k$-special component $L \subseteq Y$, then also $X$ contains a $k$-special component. We can conclude that $\mathrm{cut}_G(Y) \geq k$ for all non-empty $Y \subsetneq X$.

If $X$ itself is locally $k$-edge-connected, then $X$ is a $k$-special component because $\mathrm{cut}_G(X) \leq k - 1$ by assumption. Thus, we can conclude that $X$ is not locally $k$-edge-connected. Hence, there exist vertices $u, v \in X$ and a set $U \subseteq V$ with $u \in U$ and $v \in V \setminus U = \overline{U}$ such that $\mathrm{cut}_G(U) \leq k - 1$.

We know that $X \cap U, X \cap \overline{U} \neq \emptyset$. Furthermore, not both $\overline{X} \cap U$ and $\overline{X} \cap \overline{U}$ can be empty, as $X$ is a proper subset of $V$. We denote the number of edges between the four parts of the graph by $k_1, \ldots, k_6$ according to Fig. 1.

We consider three cases. The first case is that $\overline{X} \cap U = \emptyset$. We only have to deal with $k_2$, $k_3$, and $k_5$. Among others, we have the constraint $k_3 + k_5 \geq k$ since $X \cap U$

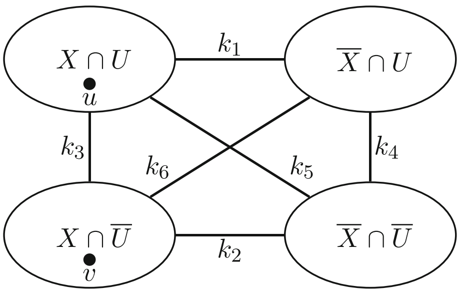

**Fig. 1** The situation in the proof of Lemma 3.5. The set $X \subsetneq V$ satisfies $\text{cut}_G(X) \leq k-1$. Since $X$ is not locally $k$-edge-connected, we have $\text{cut}_G(U) \leq k-1$. The numbers $k_1, \ldots, k_6$ denote the number of edges between the four parts

is a proper subset of $X$, which implies $\text{cut}_G(X \cap U) \geq k$. And we have the constraint $k_3 + k_5 \leq k-1$ since $\text{cut}_G(U) \leq k-1$ and $\overline{X} \cap U = \emptyset$. These two cannot be satisfied simultaneously.

The second case is that $\overline{X} \cap \overline{U} = \emptyset$. In the same way as in Case 2, we have $k_3 + k_6 \geq k$ and $k_3 + k_6 \leq k-1$, which cannot be satisfied simultaneously.

The last and general case is that both $\overline{X} \cap U \neq \emptyset$ and $\overline{X} \cap \overline{U} \neq \emptyset$. Since $X \cap U$ and $X \cap \overline{U}$ are proper subsets of $X$, we have $\text{cut}_G(X \cap U), \text{cut}_G(X \cap \overline{U}) \geq k$. This translates to $k_1 + k_3 + k_5 \geq k$ and $k_2 + k_3 + k_6 \geq k$.

We have $\text{cut}_G(X) \leq k-1$, which implies $k_1 + k_2 + k_5 + k_6 \leq k-1$. And we have $\text{cut}_G(U) \leq k-1$, we implies $k_3 + k_4 + k_5 + k_6 \leq k-1$.

Finally, the graph $G$ is $(k-1)$-edge-connected. Thus, there are at least $k-1$ edge-disjoint paths from $\overline{X} \cap U$ to $\overline{X} \cap \overline{U}$. This translates to $k_4 + k_5 + k_6 + \min(k_1, k_2, k_3) \geq k-1$. The latter corresponds to three inequalities: $k_i + k_4 + k_5 + k_6 \geq k-1$ for $i \in \{1, 2, 3\}$.

We have to show that there are no choices for $k_1, \ldots, k_6$ that satisfy all inequalities simultaneously. We have

$$k_3 + k_4 + k_5 + k_6 \geq k-1 \quad \text{and} \quad k_3 + k_4 + k_5 + k_6 \leq k-1.$$

Thus, $k_4 = k - 1 - k_3 - k_5 - k_6$. We plug this into $k_i + k_4 + k_5 + k_6 \geq k-1$ for $i = 1, 2$ and obtain $k_3 \leq k_1$ and $k_3 \leq k_2$.

Adding $k_1 + k_3 + k_5 \geq k$ and $k_2 + k_3 + k_6 \geq k$ yields

$$k_1 + k_2 + 2k_3 + k_5 + k_6 \geq 2k.$$

We have $k_1 + k_2 + k_5 + k_6 \leq k-1$. Multiplying this with 2 and combining it with $k_3 \leq k_1$ and $k_3 \leq k_2$ yields

$$k_1 + k_2 + 2k_3 + 2k_5 + 2k_6 \leq 2k-2.$$

By the non-negativity of $k_5$ and $k_6$, we thus must have

$$k_1 + k_2 + 2k_3 + k_5 + k_6 \leq 2k-2,$$

which cannot be satisfied simultaneously with the inequality $k_1 + k_2 + 2k_3 + k_5 + k_6 \geq 2k$ derived above. □

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 3.7). The next lemma is useful for this.

**Lemma 3.6** *Let $k \geq 1$, let $G = (V, E)$ be a $(k-1)$-edge-connected graph, and let $L \subseteq V$ be a $k$-special component of $G$. Then $L$ is $(\lfloor k/2 \rfloor + 1)$-edge-connected.*

*Proof* Consider $u, v \in L$. As $u$ is locally $k$-edge-connected to $v$, there are at least $k$ edge-disjoint paths from $u$ to $v$. Since $L$ is $k$-special, we have $\mathrm{cut}_G(L) \leq k-1$. Thus, at most $\lfloor \frac{k-1}{2} \rfloor$ of these edge-disjoint paths can leave $L$. Hence, there are at least $k - \lfloor \frac{k-1}{2} \rfloor = \lfloor k/2 \rfloor + 1$ paths running solely through vertices in $L$. □

**Lemma 3.7** *Let $k \geq 1$, and let $m = \lceil k/2 \rceil + 1$. 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 $|X| \geq k+1$.*

*Proof* For even $k$, Lemma 3.6 implies that we can choose $X = L$. We have $|X| \geq k+1$ by Lemma 3.3.

For $k = 1$, we have $m = 2$, and $G$ has a minimum degree of 2. The $k$-special components are just the connected components of $G$. Every connected component in a graph of minimum degree 2 contains a 2-edge-connected component, which satisfies $|X| \geq 2$.

Now let $k \geq 3$ be odd. The minimum degree in this case is $2\lceil k/2 \rceil = k+1$. By Lemma 3.6, we know that $L$ is $(m-1)$-edge-connected. If $L$ is also $m$-edge-connected, then we can choose $X = L$. We have $|X| \geq k+1$ by Lemma 3.3.

Otherwise, there exists a set $Y \subseteq L$ with $\mathrm{cut}_L(Y) \leq m-1$. Lemma 3.5 implies the existence of an $m$-special component $X \subseteq Y$ of $L$. This implies that $X$ is locally $m$-edge-connected in $L$. To finish the proof, we have to show that $\ell = |X| \geq k+1$. We have $\mathrm{cut}_L(X) \leq m-1$ since $X$ is $m$-special within $L$.

There are at most $\binom{\ell}{2}$ edges within $X$. Since $\mathrm{cut}_G(L) \leq k-1$, the number of edges that connect $X$ to $L \setminus X$ is at least

$$g(\ell) = (k+1) \cdot \ell - 2 \cdot \binom{\ell}{2} - (k-1) = -\ell^2 + (k+2) \cdot \ell - k + 1.$$

This number can be at most $m - 1 \leq k$ since $X$ is $m$-special within $L$. For $\ell = 2$, we obtain $-4+2k+4-k+1 = k+1$. For $\ell = k$, we obtain $-k^2+k^2+2k-k+1 = k+1$. Since $g$ is a concave function of $\ell$, we have $g(\ell) \geq k+1$ for $2 \leq \ell \leq k$. We have to rule out $\ell = 1$ to show that $\ell = |X| \geq k+1$, which finishes the proof.

Assume that $|X| = 1$, and let $X = \{x\}$. The situation is depicted in Fig. 2, where $a$ denotes the number of edges between $X$ and $V \setminus L$, $b$ denotes the number of edges between $L \setminus X$ and $V \setminus L$ , and $c$ denotes the number of edges between $X$ and $L \setminus X$.

We have $\mathrm{cut}_G(L) = a + b \leq k-1$ since $L$ is a $k$-special component in $G$. We have $\mathrm{cut}_L(X) = c \leq m-1$ since $X$ is an $m$-special component in $L$. We have $\mathrm{cut}_G(X) = a + c \geq k+1$ because $x$ has a degree of at least $k+1$. The first two inequalities imply $a + b + c \leq k + m - 2$. From this and the third one, we obtain

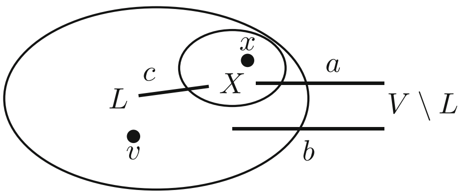

**Fig. 2** The situation in the proof of Lemma 3.7. We have $X = \{x\}$. The numbers (**a**), (**b**), and (**c**) denote the number of edges between $X$ and $V \setminus L$, $L \setminus X$ and $V \setminus L$, and $L \setminus X$ and $X$

$b \le m - 3$. Choose any $v \in L \setminus X$. Since $L$ is a $k$-special component in $G$, we have at least $k$ edge-disjoint paths from $x$ to $v$ in $G$. At most $c \le m - 1$ of these paths can run solely through $L$. Thus, $b \ge k - c \ge k - m + 1$. Together with the previously derived $b \le m - 3$, we obtain $k - m + 1 \le m - 3$. This is equivalent to $k + 4 \le 2m$, which does not hold since $m = \lceil k/2 \rceil + 1 = \frac{k+3}{2}$. We can conclude that $|X| = 1$ is impossible. □

Given Lemma 3.7, the next lemma follows. The edges $\{u_i, v_i\}$ mentioned in this lemma are the edges that we can safely remove. The resulting graph will remain $(k-1)$-edge-connected (Lemma 3.10). 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 3.7.

**Lemma 3.8** *Let $k \ge 1$, and let $m = \lceil k/2 \rceil + 1$. Let $G = (V, E)$ be a $(k-1)$-edge-connected graph of minimum degree at least $2\lceil k/2 \rceil$. Let $L_1, \dots, L_s$ be the $k$-special components of $G$. Then there exist vertices $u_i, v_i \in L_i$ for all $i \in \{1, \dots, s\}$ such that the following properties are met:*

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

*Proof* Consider any $i \in \{1, \dots, s\}$. According to Lemma 3.7, there exists a locally $m$-edge-connected set $X_i \subseteq L_i$ with $|X_i| \ge k+1$. Since $|X_i| \ge k+1$ and $\mathrm{cut}_G(L_i) \le k - 1$ (because $L_i$ is $k$-special), there must be a vertex $u_i \in X_i$ with $N(u_i) \subseteq L_i$.

We choose $v_i$ to be another vertex in the locally $m$-edge-connected component $X_i$ contained in $L_i$ such that $\{u_i, v_i\} \in E$. If such a $v_i$ exists, then there are $m$ edge-disjoint paths from $u_i$ to $v_i$ in $L_i$ since $u_i, v_i \in X_i$ and $X_i$ is locally $m$-edge-connected in $L_i$.

Such vertices $v_i$ exist because of the construction of the sets $X_i$ in Lemma 3.7: Either $X_i = L_i$. Then this follows since $N(u_i) \subseteq L_i = X_i$. Or $X_i \subsetneq L_i$. In this case, we have $\mathrm{cut}_{L_i}(X_i) \le m - 1 = \lceil k/2 \rceil$ since $X_i$ is an $m$-special component in $L_i$. Since $u_i$ has at least $2\lceil k/2 \rceil$ neighbors, all of which are in $L_i$, at least one $v_i \in X_i$ that is adjacent to $u_i$ must exist. □

**Lemma 3.9** *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 $L_1, \dots, L_s$. Let $u_1, \dots, u_s$ and $v_1, \dots, v_s$ be chosen as in Lemma 3.8. Let $Q = \big\{\{u_i, v_i\} \mid 1 \le i \le s\big\}$.*

*For all $i$ and $j$ with $i \neq j$, there does not exist a set $C \subseteq E$ with $|C| \leq k-1$ such that $u_i$ is disconnected from $v_i$ and $u_j$ is disconnected from $v_j$ in $G - Q - C$.*

*Proof* Within $L_i$, there are $m$ edge-disjoint paths from $u_i$ to $v_i$ by Lemma 3.8. After removing $Q$ from $G$, there remain at least $m - 1 = \lceil k/2 \rceil$ edge-disjoint paths within $L_i$.

The same holds for $L_j$ with $u_j$ and $v_j$. Thus, any set $C$ of edges that simultaneously disconnects $u_i$ from $v_i$ and $u_j$ from $v_j$ must satisfy $|C| \geq 2 \cdot \lceil k/2 \rceil \geq k$. □

**Lemma 3.10** *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 3.8. Let $Q = \big\{\{u_i, v_i\} \mid 1 \leq i \leq s\big\}$. Then $G - Q$ is $(k-1)$-edge-connected.*

*Proof* We have to show that $\mathrm{cut}_{G-Q}(X) \geq k-1$ for all non-empty $X \subsetneq V$. To do this, we distinguish two cases: In the first case, for every $i$, either $u_i, v_i \in X$ or $u_i, v_i \notin X$. In this case, $\mathrm{cut}_{G-Q}(X) = \mathrm{cut}_G(X) \geq k-1$ since no edges between $X$ and $V \setminus X$ are removed.

In the second case, there exists an $i$ such that $|\{u_i, v_i\} \cap X| = 1$. If there are two or more $i$ with this property, then $\mathrm{cut}_{G-Q}(X) \geq k-1$ follows from Lemma 3.9.

Now assume that there is only a single $i$ with $|\{u_i, v_i\} \cap X| = 1$. Without loss of generality, we assume that $u_i \in X$ and $v_i \notin X$. Since $u_i$ and $v_i$ are locally $k$-edge-connected in $G$, we have $\mathrm{cut}_G(X) \geq k$. Only the edge $\{u_i, v_i\}$ is removed from this cut, thus $\mathrm{cut}_{G-Q}(X) \geq k-1$. □

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 3.11** *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 3.8. 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.*

*Proof* We prove the lemma by a series of claims. In the following, arithmetic is modulo $s$.

**Claim 3.12** *For every $i$, the vertices $u_i$ and $v_{i+1}$ are locally $k$-edge-connected in $\tilde{G}$.*

*Proof* The graph $G - Q$ is $(k-1)$-edge-connected and $S$ contains the edge $\{u_i, v_{i+1}\}$. □

**Claim 3.13** *For all $i$ and $j$, the vertices $u_i$ and $u_j$ are locally $k$-edge-connected in $\tilde{G}$.*

*Proof* Without loss of generality, we assume $i < j$.

Assume to the contrary that there is a set $C$ of at most $k-1$ edges such that $C$ disconnects $u_i$ from $u_j$. Since $G-Q$ is $(k-1)$-edge-connected by Lemma 3.10, we must have $C \cap S = \emptyset$.

Lemma 3.9 says that if $C$ disconnects $u_a$ from $v_a$ for some $a$, then it cannot disconnect $u_b$ from $v_b$ for $a \neq b$. Thus, $C$ can disconnect at most one pair $u_a$, $v_a$. Consider the paths from $u_i$ to $u_j$ via $v_{i+1}, u_{i+1}, v_{i+2}, u_{i+2}, \ldots, v_{j-1}, u_{j-1}, v_j$ or via $v_i, u_{i-1}, v_{i-1}, u_{i-2}, \ldots, v_{j+2}, u_{j+1}, v_{j+1}$ after adding $S$. The set $S$ contains all the connections between $u$ and $v$ vertices in these paths. Since $C$ can disconnect at most one pair $u_a, v_b$ by Lemma 3.9, one of these paths must still exist. Thus, $u_i$ and $v_i$ are still connected in $\tilde{G}$ after removing at most $k-1$ edges. □

**Claim 3.14** *For all $i$ and $j$, vertices $u_i$, $v_i$, $u_j$, and $v_j$ are pairwise locally $k$-edge-connected in $\tilde{G}$.*

*Proof* This follows by transitivity of local $k$-edge-connectedness and Claims 3.12 and 3.13 above. □

**Claim 3.15** *For all $i$, the vertices in $L_i$ are locally $k$-edge-connected in $\tilde{G}$.*

*Proof* We show that every $x \in L_i$ is locally $k$-edge-connected to $u_i$ in $\tilde{G}$. Then the claim follows by transitivity of local $k$-edge-connectedness.

Let $X \subsetneq V$ with $x \in X$ and $u_i \notin X$. If $v_j \in X$ (including the case $j = i$) or $u_j \in X$ for some $j \neq i$, then $\mathrm{cut}_{\tilde{G}}(X) \geq k$ by Claim 3.14. Thus, the case $u_1, \ldots, u_s, v_1, \ldots, v_s \notin X$ remains to be considered. Since $L_i$ is locally $k$-edge-connected in $G$, we have $\mathrm{cut}_G(X) \geq k$. This cut does not change by removing $Q$. Hence, $\mathrm{cut}_{\tilde{G}}(X) \geq k$. □

Transitivity of local $k$-edge-connectedness, Claim 3.14, and Claim 3.15 imply that $L = \bigcup_{i=1}^{s} L_i$ is locally $k$-edge-connected in $\tilde{G}$. The vertices that are not part of any $k$-special component in $G$ remain to be considered.

Consider an arbitrary non-empty $X \subsetneq V$. If we can show that $\mathrm{cut}_X(\tilde{G}) \geq k$, then we have completed the proof of this lemma. If $X \cap L \neq \emptyset$ and $(V \setminus X) \cap L \neq \emptyset$, then $\mathrm{cut}_{\tilde{G}}(X) \geq k$ because $L$ is locally $k$-edge-connected in $\tilde{G}$. Thus, $L \subseteq X$ or $L \subseteq V \setminus X$. By symmetry, we restrict ourselves to the second case. Then we have $\mathrm{cut}_G(X) = \mathrm{cut}_{\tilde{G}}(X)$ because no edge connecting $X$ to $V \setminus X$ is in $Q$ or $S$.

If $\mathrm{cut}_G(X) \leq k-1$, then $X$ contains a $k$-special component by Lemma 3.5, contradicting our assumption. Thus, we have $\mathrm{cut}_G(X) \geq k$. □

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 3.7 and 3.8 can be computed in polynomial time.

Finally, we need the following lemma for improving the approximation ratios in the next section. It basically implies that we can jump from disconnected graphs directly to 2-edge-connected graphs in one iteration.

**Lemma 3.16** *Let $G = (V, E)$ be any undirected graph (not necessarily connected) of minimum degree 2. Let $L_1, \dots, L_s$ be the 2-special components of $G$. Then we have the following properties:*

1. *There exists a set of edges $Q = \big\{\{u_i, v_i\} \mid 1 \le i \le s\big\}$ such that $\{u_i, v_i\} \in E$, $u_i, v_i \in L_i$, and neither $u_i$ nor $v_i$ are connected to any vertex outside $L_i$.*
2. *The graph $G' = G - Q$ contains exactly the same connected components as $G$.*
3. *Let $S = \big\{\{u_i, v_{i+1}\} \mid 1 \le i \le s\big\}$, where arithmetic is modulo $s$. Then the graph $\tilde{G} = G - Q + S$ is 2-edge-connected.*

*Proof* Every 2-special component is 2-edge-connected. Consider any 2-special component $L_i$ of $G$. We have $\text{cut}_G(L_i) \le 1$ by definition. If $\text{cut}_G(L_i) = 0$, then $L_i$ contains at least three vertices since $G$ has minimum degree 2. Hence, the existence of an edge $\{u_i, v_i\} \in E$ with $u_i, v_i \in L_i$ follows. If $\text{cut}_G(L_i) = 1$, then there is one vertex $x_i$ that connects $L_i$ to the rest of the graph. This vertex must be incident to at least two other vertices of $L_i$ as $L_i$ is 2-edge-connected. Now a similar argument as for the case $\text{cut}_G(L_i) = 0$ applies. This proves Item (1).

Since every 2-special component is 2-edge-connected, removal of $Q$ does not cause any 2-special component to split. Thus, we have Item (2).

Let us now prove Item (3). The graph $\tilde{G}$ is connected: any two connected components of $G$, which are still connected in $G'$, contain 2-special components $L_i$ and $L_j$, respectively. These are connected via a path that passes through $u_i, v_{i+1}, u_{i+1}, \dots, u_{j-1}, v_j$. We have to show that $\tilde{G}$ remains connected after the removal of any edge.

First, if we remove an edge $\{u_i, v_{i+1}\} \in S$, then we still have a path from $u_i$ to $v_{i+1}$ via $v_i, u_{i-1}, v_{i-1}, \dots, v_{i+2}, u_{i+1}$. Second, if we remove a bridge edge $e = \{x, y\}$ of $G$ (a bridge edge of a graph is an edge whose removal increases the number of connected components), then there still must exist a path from $x$ to some $u_i$ that does not visit $y$. Similarly, there must exist a path from $y$ to some $u_j$ that does not visit $x$. Now $u_i$ and $u_j$ are connected as in the first case. Third, if we remove any other edge $e = \{x, y\}$ of $\tilde{G}$, then $x$ and $y$ belong to the same 2-edge-connected component $C$ of $G$. If $C = L_i$ is 2-special, then there is a path from $x$ to $y$ in $G - e$. If this path uses $\{u_i, v_i\}$, we can reroute it via $v_{i+1}, u_{i+1}, \dots, v_{i-1}, u_{i-1}$ to obtain a path in $\tilde{G}$. If this path does not use $\{u_i, v_i\}$, then it still exists in $\tilde{G}$. If $C$ is not equal to a 2-special component, then $C$ is a 2-edge-connected component of $\tilde{G}$ and, thus, remains connected after removal of $e$. □

## 3.2 Algorithm and Analysis

Our approximation algorithm for Min-$d$Reg-$k$Edge (Algorithm 2) starts with any $d$-factor $F_0$ without requiring connectivity. Then it iteratively uses a subroutine (Algorithm 3) that increases the connectivity.

**Algorithm 2** Approximating $k$-edge-connected $d$-factors of minimum weight

```
input  : undirected complete graph $G = (V, E)$, edge weights $w$, integers $k \geq 1$,
         $d \geq 2\lceil k/2 \rceil$
output: $k$-edge-connected $d$-factor $R$ of $G$
1  compute a minimum-weight $d$-factor $F_0$
2  if $F_0$ is not 2-edge-connected then
3      find the 2-special components of $F_0$; let $L_1, \ldots, L_s$ be these 2-special components
4      find nodes $u_i, v_i \in L_i$ for $i \in \{1, \ldots, s\}$ as in Lemma 3.16; $Q \leftarrow \big\{\{u_i, v_i\} \mid 1 \leq i \leq s\big\}$
5      compute a TSP tour $T$ on $V$ using Christofides' algorithm
6      take shortcuts to get a tour $T'$ on $u_1, \ldots, u_s$ (without loss of generality in this order)
7      $S \leftarrow \big\{\{u_i, v_{i+1}\} \mid 1 \leq i \leq s\big\}$ (arithmetic modulo $s$)
8      $F_2 \leftarrow F_0 - Q + S$
9  else
10     $F_2 \leftarrow F_0$
11 end
12 for $p \leftarrow 3, \ldots, k$ do
13     if $F_{p-1}$ is not $p$-edge-connected then
14         apply Algorithm 3 to obtain $F_p$
15     else
16         $F_p \leftarrow F_{p-1}$
17     end
18 end
19 $R \leftarrow F_k$
```

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

**Lemma 3.17** *Let $k \geq 1$ be arbitrary, and let $p \in \{0, 2, 3, 4, \ldots, k-1, k\}$. Let $F_p$ be computed by Algorithm 2. Then $F_p$ is $d$-regular and $p$-edge-connected.*

*Proof* By construction of $F_0$ and $F_2, \ldots, F_k$, all these graphs are $d$-regular. The graph $F_2$ is 2-edge-connected by Lemma 3.16 or $F_0$ was already 2-edge-connected. Now assume that the lemma holds for some $p-1$ for $p \geq 3$. We know that $F_{p-1}$ is $(p-1)$-edge-connected and $d$-regular by induction hypothesis.

Either $F_{p-1}$ is already $p$-edge-connected. In this case, $F_p = F_{p-1}$ (line 16). Or $F_{p-1}$ is not $p$-edge-connected. Then we apply Algorithm 3. By Lemma 3.11, $F_p$ is $p$-edge-connected. All degrees are maintained. Thus, $F_p$ is $d$-regular. □

**Algorithm 3** Increasing the edge-connectivity of a graph by one while maintaining $d$-regularity

```
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
         $2\lceil p/2 \rceil$
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 nodes $u_i, v_i \in L_i$ for all $i \in \{1, \ldots, s\}$ as in Lemma 3.8; $Q \leftarrow \big\{\{u_i, v_i\} \mid 1 \leq i \leq s\big\}$
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  $F_p \leftarrow F_{p-1} - Q + S$
```

In our approximation algorithm, we use Christofides' algorithm [29, Section 2.4] to compute TSP tours. 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 [30]. It relates the weight of the tour computed by Christofides' algorithm to the objective value of the relaxation of the integer linear program for $k$-edge-connected graphs of minimum weight. Fukunaga and Nagamochi state the following result only for the case $k = 2$. We obtain the result below that we need for our purposes by scaling the right-hand side of their linear program by a factor of $k/2$.

**Lemma 3.18** (Fukunaga, Nagamochi [8, Theorem 2]) *Let $T$ be the TSP tour obtained from Christofides' algorithm. Then $w(T) \leq \frac{3}{k} \cdot w(OptE^k)$.*

We also need the following lemma. (Otherwise, we have to replace Christofides' algorithm by the spanning tree heuristic for the case of $k = 1$.)

**Lemma 3.19** *Let $T$ be the TSP tour obtained from Christofides' algorithm. Then $w(T) \leq 2 \cdot w(MST)$.*

*Proof* The weight $w(T)$ can be bounded from above by the sum of an MST plus the weight of a minimum-weight perfect matching on the odd-degree nodes of this tree. The weight of this minimum-weight perfect matching can be bounded from above by the weight of the MST using the triangle inequality. □

**Lemma 3.20** *If, in Algorithm 2, we enter line 14 and call Algorithm 3, then*

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

*Proof* Let $Q$, $S$, $T$, and $T'$ be as in the corresponding call of Algorithm 3. We have $w(F_p) = w(F_{p-1}) - w(Q) + w(S)$. By construction, the triangle inequality, Lemma 3.18, and $w(\mathrm{OptE}^k) \leq w(\mathrm{optEF}_d^k)$, we have

$$w(T') \leq w(T) \leq \frac{3}{k} \cdot w(\mathrm{optEF}_d^k).$$

Since $w(\{u_i, v_{i+1}) \leq w(\{u_i, u_{i+1}\}) + w(\{u_{i+1}, v_{i+1}\})$, we have $w(S) \leq w(T') + w(Q)$. The overall weight added is $w(S) - w(Q)$, thus at most $w(T')$. Hence, the lemma holds. □

**Lemma 3.21** *In Algorithm 2, we have $w(F_2) \leq w(F_0) + \frac{3}{k} \cdot w(optEF_d^k)$.*

*Proof* The proof is almost identical to the proof of Lemma 3.20. □

**Lemma 3.22** *If Algorithm 2 calls Algorithm 3 $q$ times, then*

$$w(F_k) \leq \frac{3q}{k} \cdot w(optEF_d^k) + w(F_2).$$

*Proof* The lemma follows by immediately applying Lemma 3.20. □

**Theorem 3.23** *For $k \geq 1$ and $d \geq 2\lceil k/2 \rceil$, Algorithm 2 is a polynomial-time approximation algorithm for* Min-$d$Reg-$k$Edge. *It achieves an approximation ratio of*

- *2.5 for even $d$,*
- $4 - \frac{3}{k}$ *for odd $d$ and $k \geq 2$, and*
- *3 for odd $d$ and $k = 1$.*

*Proof* First, let $d$ be even. We can restrict ourselves to consider even $k$ as discussed in Section 1.1.2. Algorithm 2 calls Algorithm 3 at most $k/2 - 1$ times as every call increases the connectivity by at least two. The approximation ratio follows from Lemmas 3.21 and 3.22 and $w(F_0) \leq w(\text{optEF}_d^k)$.

Second, let $d$ be odd and $k \geq 2$. We need at most $k - 2$ calls of Algorithm 3 to obtain a $k$-edge-connected $d$-factor. Thus, we obtain $w(F_k) \leq \frac{3k-6}{k} \cdot w(\text{optEF}_d^k) + w(F_2) \leq \left(4 - \frac{3}{k}\right) \cdot w(\text{optEF}_d^k)$ by Lemmas 3.21 and 3.22.

Finally, consider odd $d$ and $k = 1$. In this case, the approximation ratio follows from Lemma 3.19. (Lemma 3.20 yields a worse bound for this case.) □

Algorithm 2 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].

*Remark 3.24* We observe that even for Min-$d$Reg-1Edge for odd $d$, Algorithm 2 always outputs a 2-edge-connected graph that weighs at most $3 \cdot w(\text{OptEF}_d^1)$. This limits the approximation ratio of the approximation algorithm as there are instances with $3 \cdot w(\text{OptEF}_d^1) = w(\text{OptEF}_d^2)$ (see Proposition 3.5 below). Directly computing a 1-edge-connected solution might result in an improved approximation ratio, but we do not see how to achieve this.

**Proposition 3.25** *For every odd $d \geq 3$, there are instances with $3 \cdot w(OptEF_d^1) = w(OptEF_d^2)$.*

*Proof* The set of vertices of the instance consists of a vertex $v$ plus $d$ sets $V_1, \ldots, V_d$ that consist of $d + 2$ vertices each. We set the distance of $v$ to each of the other vertices equal to 1. The distance between each pair of vertices from the same set $V_i$ is 0. Finally, the distance between all pairs of vertices from different sets $V_i$ and $V_j$ is 2. The triangle inequality is satisfied.

An optimal connected $d$-factor connects $v$ to one vertex of each $V_i$. Since each $V_i$ has an odd number of vertices and $d$ is odd as well, we can complete the connected $d$-factor without any further cost. Thus, the total cost is $d$.

An optimal 2-edge-connected $d$-factor has a weight of $3d$: Because each $V_i$ has an odd number of vertices and $d$ is odd, any 2-edge-connected $d$-factor must have at least three edges leaving each set $V_i$. If such an edge $e$ is incident with $v$, then we charge its weight to $V_i$. The other possibility is that $e$ is incident with a vertex from some $V_j$, where $j \neq i$. In this case $e$ has a weight of 2 and we charge a weight of 1 to both $V_i$ and $V_j$. The total charge of all sets $V_i$ equals the total weight of the

2-edge-connected $d$-factor. Since each $V_i$ is charged at least 3, the total weight of any 2-edge-connected $d$-factor is at least $3d$. □

## 4 Generalization to Arbitrary Degree Sequences

Both algorithms of Sections 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-$k$Vertex and Min-$d$Gen-$k$Edge, where we have a degree requirement of at least $d$ for each vertex.

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 2.3.) 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-$(2k-1)$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* $5 + \frac{2k-2}{n}$.

**Theorem 4.2** *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* $4 - \frac{3}{k}$.

Min-2Gen-1Edge *can be approximated in polynomial time with an approximation ratio of 3.*

## 5 Hardness Results

### 5.1 TSP-Inapproximability

In this section, we prove that Min-$d$Reg-1Edge cannot be approximated better than Min-TSP.

**Theorem 5.1** *For every $d \geq 2$, if* Min-$d$Reg-1Edge *can be approximated in polynomial time within a factor of $r$, then* Min-TSP *can be approximated in polynomial time within a factor of $r$.*

*Proof* We show that Min-$d$Reg-1Edge can be used to approximate Min-TSP. Let the instance of Min-TSP be given by a complete graph $G = (V, E)$ and edge weights $w = (w_e)_{e \in E}$ that satisfy the triangle inequality. Let $n = |V|$. We construct an instance of Min-$d$Reg-1Edge as follows: The instance consists of a complete graph $H = (V', E')$. Here $V' = \bigcup_{v \in V} V_v$, where $V_v = \{v_1, v_2, \ldots, v_{d+1}\}$, i.e., $H$ contains $(d+1) \cdot n$ vertices. We assign edge weights $\tilde{w}$ as follows:

- $\tilde{w}_{\{v_i, v_j\}} = 0$ for all $v \in V$, $i \neq j$,

- $\tilde{w}_{\{u_i,v_j\}} = w_{\{u,v\}}$ for all $u \neq v$, $i$ and $j$.

Every TSP tour $T$ of $G$ maps to a connected $d$-factor $R$ of $H$ of the same weight: We give $T$ an orientation. For an edge from $u$ to $v$ in $T$, we include $\{u_1, v_2\}$ in $R$. Adding all edges except $\{v_1, v_2\}$ to $R$ within each $V_v$ yields a connected $d$-factor $R$. Clearly, $\tilde{w}(F) = w(T)$.

Now assume that we have a connected $d$-factor $R$ of $H$. We claim that we can construct a TSP tour $T$ of $G$ with $w(T) \leq \tilde{w}(R)$. We construct a multiset $T'$ of edges of $G$ as follows: For each edge $\{u_i, v_j\}$ of $R$, if $u \neq v$, we add an edge $\{u, v\}$ to $T'$. Otherwise, if $u = v$, we ignore the edge. The sum of the degrees in $R$ of all vertices in each set $V_v$ is equal to $(d+1)d$ and is therefore even. Thus, for each $v$, the number of edges leaving $V_v$ in $R$, which equals the number of edges incident to $v$ in $T'$ by construction, is even as well. Since $R$ is connected, the multigraph $G' = (V, T')$ is connected as well. By construction, $w(T') = \tilde{w}(R)$. Since $G'$ is connected and all its vertices have even degree, $G'$ is Eulerian. Therefore, we can obtain a TSP tour $T$ from $T'$ by taking shortcuts. By the triangle inequality, $w(T) \leq w(T') = \tilde{w}(R)$. □

The same construction as in the proof of Theorem 5.1 yields the same inapproximability result for Min-$d$Reg-2Edge.

Min-TSP is APX-hard [25]. Furthermore, the reduction from Min-TSP to Min-$d$Reg-1Edge (and also to Min-$d$Reg-2Edge) is in fact an L-reduction [24] (see also Shmoys and Williamson [29, Section 16.2]). This proves the APX-hardness of Min-$d$Reg-1Edge and Min-$d$Reg-2Edge for all $d \geq 2$.

## 5.2 Hardness for Growing *d*

In this section, we generalize the NP-hardness proof by Cheah and Corneil [2] for the decision problem if a graph contains a connected $d$-factor to the case that $d$ grows with $n$. We also extend Theorem 5.1 and the APX-hardness to growing $d$.

Let us consider Cheah and Corneil's [2, Section 3.2] reduction from the Hamiltonian cycle problem. Crucial for their reduction is the notion of the $d$-expansion of a vertex $v$, which is obtained as follows:

1. We construct a gadget $G_{d+1}$ by removing a matching of size $\lceil \frac{d}{2} \rceil - 1$ from a complete graph on $d+1$ vertices.
2. We connect each vertex whose degree has been decreased by one to $v$.

The reduction itself takes a graph $G$ for which we want to test if $G$ contains a Hamiltonian cycle and maps it to a graph $R_d(G)$ as follows: For even $d$, $R_d(G)$ is the graph obtained by performing a $d$-expansion for every vertex of $G$. For odd $d$, the graph $R_d(G)$ is obtained by doing the following for each vertex $v$ of $G$: add vertices $u_1, u_2, \ldots, u_{d-2}$; connect $v$ to $u_1, \ldots, u_{d-2}$; perform a $d$-expansion on $u_1, \ldots, u_{d-2}$. We have that $G$ contains a Hamiltonian cycle if and only if $R_d(G)$ contains a connected $d$-factor.

We note that $R_d(G)$ has $(d+2) \cdot n$ vertices for even $d$ and $\Theta(d^2 n)$ vertices for odd $d$ and can easily be constructed in polynomial time since $d < n$.

**Theorem 5.2** *For every fixed $\delta \in [0, 1)$, there is a function $d = d(n) = \Theta(n^\delta)$ that maps to even integers such that checking if an $n$-vertex graph contains a connected $d(n)$-factor is* NP*-hard.*

*For every fixed $\delta \in [0, \frac{1}{2})$, there is a function $d = d(n) = \Theta(n^\delta)$ that maps to odd integers such that checking if an $n$-vertex graph contains a $d(n)$-factor is* NP*-hard.*

*Proof* We first present the proof for the case that we map to even integers. After that, we briefly point out the difference for odd integers.

Let $f = f(n) = 2\lceil n^{\frac{\delta}{1-\delta}} \rceil$ and apply $R = R_f(G)$. The graph $R$ has $g(n) = n \cdot (f + 2)$ vertices. since $f$ is even. We have $g = \Theta(n^{\frac{1}{1-\delta}})$. Now we determine $d$. We require $d(g(n)) = f(n)$. This can be achieved because $g = \omega(n)$ is an injective function. From this, $d = \Theta(n^\delta)$ follows. For natural numbers that are not images of $g$, we interpolate $f$ to maintain the growth bound.

Let us now point out the differences for functions $f$ mapping to odd integers. In this case, since the reduction for $d$ maps to graphs of size $\Theta(d^2 n)$, we have to choose $d = \Theta(n^{\frac{\delta}{1-2\delta}})$. This, however, works only up to $\delta < 1/2$. □

In the same way as the NP-completeness, the inapproximability can be transferred. The reduction creates graphs of size $(d + 1) \cdot n$. The construction is the same as in Section 5.1, and the proof follows the line of the proof of Theorem 5.2. Here, however, we do not have to distinguish between odd and even $d$ for the symmetric variant, as the reduction in Section 5.1 is the same for both cases.

**Theorem 5.3** *For every fixed $\delta \in [0, 1)$, there exists a function $d = \Theta(n^\delta)$ such that* Min-$d$Reg-1Edge *and* Min-$d$Reg-2Edge *are* APX*-hard and cannot be approximated better than* Min-TSP.

Complementing Theorems 5.2 and 5.3, Min-$d$Reg-1Edge admits a PTAS for $d \geq n/3$ and finding a connected $d$-factor can be done in polynomial time for $d \geq n/3$ [23].

## 6 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.erek.G.: The complexity of regular subgraph recognition. Discret. 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. Discret. 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.) Proc. of the 11th Workshop on Approximation and Online Algorithms (WAOA 2013), volume 8447 of Lecture Notes in Computer Science, pp. 120–131. Springer (2014)
6. Czumaj, A., Lingas, A.: Minimum $k$-connected geometric networks. In: Kao, M.-Y. (ed.) Encyclopedia of Algorithms, pp. 536–539. Springer (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. Algo. **24**(2), 310–324 (1997)
8. Fukunaga, T., Nagamochi, H.: Network design with edge-connectivity and degree constraints. Theory Comput. Syst. **45**(3), 512–532 (2009)
9. Fukunaga, T., Nagamochi, H.: Network design with weighted degree constraints. Discret. Optim. **7**(4), 246–255 (2010)
10. Fukunaga, T., Ravi, R.: Iterative rounding approximation algorithms for degree-bounded node-connectivity network design. In: Proc. of the 53rd Ann. IEEE Symp. 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 (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: Methodological Foundations, volume 3418 of Lecture Notes in Computer Science, pp. 143–177. Springer (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 (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.) Proc. of the 17th Int. Conf. on Integer Programming and Combinatorial Optimization (IPCO), volume 8494 of Lecture Notes in Computer Science, pp. 369–380. Springer (2014)
21. Lovász, L., Plummer, M.D.: Matching Theory, volume 121 of North-Holland Mathematics Studies. Elsevier (1986)
22. Manthey, B., Waanders, M.: Approximation algorithms for $k$-connected graph factors. In: Sanitá, L., Skutella, M. (eds.) Proc. of the 13th Workshop on Approximation and Online Algorithms (WAOA 2015), volume 9499 of Lecture Notes in Computer Science, pp. 1–12. Springer (2016)
23. Narayanaswamy, N.S., Rahul, C.S.: Approximation and exact algorithms for special cases of connected $f$-factors. Proc. of the 10th Int. Computer Science Symp. in Russia (CSR), volume 9139 of Lecture Notes in Computer Science, pp. 350–363. Springer (2015)
24. Papadimitriou, C.H., Yannakakis, M.: Optimization approximation, and complexity classes. J. Comput. Syst. Sci. **43**(3), 425–440 (1991)
25. Papadimitriou, C.H., Yannakakis, M.: The traveling salesman problem with distances one and two. Math. Oper. Res. **18**(1), 1–11 (1993)
26. 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)
27. Tutte, W.T.: A short proof of the factor theorem for finite graphs. Can. J. Math. **6**, 347–352 (1954)
28. West, D.B.: Introduction to Graph Theory, 2nd edn. Prentice-Hall (2001)
29. Williamson, DP., Shmoys, DB.: The Design of Approximation Algorithms. Cambridge University Press (2011)
30. Wolsey, L.A.: Heuristic analysis, linear programming and branch and bound. In: Rayward-Smith, V.J. (ed.) Combinatorial Optimization II, volume 13 of Mathematical Programming Studies, pp. 121–134. Springer (1980)