Approximability of Connected Factors*

Kamiel Cornelissen[1], Ruben Hoeksma[1], Bodo Manthey[1],
N. S. Narayanaswamy[2], and C. S. Rahul[2]

[1] University of Twente, Enschede, The Netherlands
{k.cornelissen, r.p.hoeksma, b.manthey}@utwente.nl
[2] Indian Institute of Technology Madras, Chennai, India
{swamy, rahulcs}@cse.iitm.ac.in

Abstract. Finding a d-regular spanning subgraph (or d-factor) of a graph is easy by Tutte's reduction to the matching problem. By the same reduction, it is easy to find a minimal or maximal d-factor of a graph. However, if we require that the d-factor is connected, these problems become NP-hard – finding a minimal connected 2-factor is just the traveling salesman problem (TSP).
Given a complete graph with edge weights that satisfy the triangle inequality, we consider the problem of finding a minimal connected d-factor. We give a 3-approximation for all d and improve this to an $(r+1)$-approximation for even d, where r is the approximation ratio of the TSP. This yields a 2.5-approximation for even d. The same algorithm yields an $(r+1)$-approximation for the directed version of the problem, where r is the approximation ratio of the asymmetric TSP. We also show that none of these minimization problems can be approximated better than the corresponding TSP.
Finally, for the decision problem of deciding whether a given graph contains a connected d-factor, we extend known hardness results.

1 Introduction

The traveling salesman problem (Min-TSP) is one of the basic combinatorial optimization problems: 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 [18, 24]. In general, d-regular, spanning subgraphs (also called d-factors) of minimum weight can be found in polynomial time using Tutte's reduction [18, 24] to the matching problem. Cheah and Corneil [6] 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

* A full version with all proofs is available at http://arxiv.org/abs/1310.2387.

d. While one might think at first glance that the problem cannot become easier for larger d, finding (minimum-weight) connected d-factors is easy for $d \geq n/2$, where $n = |V|$, as in this case any d-factor is already connected. This poses the question for which values of d (as a function of n) the problem becomes tractable. In this paper, we analyze the complexity and approximability of the problem of finding a d-factor of minimum weight.

1.1 Problem Definitions and Preliminaries

In the following, n is always the number of vertices. To which graph n refers will be clear from the context.

All problems defined below deal with undirected graphs, unless stated otherwise. For any d, d-RCS is the following decision problem: Given an arbitrary undirected graph G, does G have a connected d-factor? Here, d can be a constant, but also a function of the number n of vertices of the input graph G. 2-RCS is just the Hamiltonian cycle problem.

Just as Min-TSP is the optimization variant of 2-RCS, we consider the optimization variant of d-RCS, which we call Min-d-RCS: As an instance, we are given an undirected complete graph $G = (V, E)$ and non-negative edge weights w that satisfy the triangle inequality, i.e., $w(\{x, z\}) \leq w(\{x, y\}) + w(\{y, z\})$ for every $x, y, z \in V$. The goal of Min-d-RCS is to find a connected d-factor of G of minimum weight. Min-2-RCS is just Min-TSP.

A *bridge edge* of a graph is an edge whose removal increases the number of components of the graph. A graph G is called *2-edge connected* if G is connected and does not contain bridge edges. For even d, any connected d-factor is also 2-edge-connected, i.e., does not contain bridge edges. This is not true for odd d. If we require 2-edge-connectedness also for odd d, we obtain the problem Min-d-R2CS, which is defined as Min-d-RCS, but asks for a 2-edge-connected d-factor. For consistency, Min-d-R2CS is also defined for even d, although it is then exactly the same problem as Min-d-RCS.

Finally, we also consider the asymmetric variant of the problem: given a directed complete graph $G = (V, E)$, find a spanning connected subgraph of G that is d-regular. Here, d-regular means that every vertex has indegree d and outdegree d. We denote the corresponding minimization problem by Min-d-ARCS. Min-1-ARCS is just the asymmetric TSP (Min-ATSP).

Max-d-RCS and Max-d-ARCS are the maximization variants of Min-d-RCS and Min-d-ARCS, respectively. For Max-d-RCS and Max-d-ARCS we do not require that the edge weights satisfy the triangle inequality. In the same way as for the minimization variants, Max-2-RCS is the maximum TSP (Max-TSP) and Max-1-ARCS is the maximum ATSP (Max-ATSP).

If the graph and its edge weights are clear from the context, we abuse notation by also denoting by d-RCS a minimum-weight connected d-factor, by d-R2CS a minimum-weight 2-edge-connected d-factor, and by d-ARCS a minimum-weight connected d-regular subgraph of a directed graph.

In the same way, let d-F denote a minimum-weight d-factor (no connectedness required) of a graph and let d-AF denote a minimum-weight d-factor of a directed graph. Let MST denote a minimum-weight spanning tree, and let TSP and ATSP denote minimum-weight (asymmetric) TSP tours. We have 2-RCS = TSP and 1-ARCS = ATSP. Furthermore, 2-F is the undirected cycle cover problem and 1-AF is the directed cycle cover problem.

We note that d-factors do not exist for all combinations of d and n. If both n and d are odd, then no n-vertex graph possesses a d-factor. For all other combinations of n and d with $d \leq n-1$, there exist d-factors in n-vertex graphs, at least in the complete graph.

In the following, K_n denotes the undirected complete graph on n vertices. A vertex v of a graph G is called a *cut vertex* if removing v increases the number of components of G.

1.2 Previous Results

Requiring connectedness in addition to some other combinatorial property has already been studied for dominating sets [13] and vertex cover [8]. For problems such as minimum s-t vertex separator, which are known to be solvable in polynomial time, the connectedness condition makes it NP-hard, and recent results have studied the parameterized complexity of finding a connected s-t vertex separator [19]. Also finding connected graphs with given degree sequences that are allowed to be violated only slightly has been well-studied [5, 23].

As far as we are aware, so far only the maximization variant Max-d-RCS of the connected factor problem has been considered for $d \geq 3$. Baburin, Gimadi, and Serdyukov proved that Max-d-RCS can be approximated within a factor of $1-\frac{2}{d\cdot(d+1)}$ [2,12]. A slightly better approximation ratio can be achieved if the edge weights are required to satisfy the triangle inequality [3]. Baburin and Gimadi also considered approximating both Max-d-RCS and Min-d-RCS (both without triangle inequality) for random instances [3,4]. For $d = 2$, we inherit the approximation results for Min-TSP of 3/2 [26, Section 2.4] and Max-TSP of 7/9 [20]. For $d = 1$, we inherit the $O(\log n/\log\log n)$-approximation for Min-ATSP [1] and 2/3 for Max-ATSP [14]. As far as we know, no further polynomial-time approximation algorithms with worst-case guarantees are known for Min-d-RCS. Like for Min-TSP [26, Section 2.4], the triangle inequality is crucial for approximating Min-d-RCS and Min-d-ARCS – otherwise, no polynomial-time approximation algorithm is possible, unless P = NP. Baburin and Gimadi [2, 3] claimed that Max-d-RCS is APX-hard because it generalizes Max-TSP. However, this is only true if we consider d as part of the input, as then $d = 2$ corresponds to Max-TSP.

1.3 Our Results

Table 1 shows an overview of previous results and our results. Our main contributions are a 3-approximation algorithm for Min-d-RCS for any d and a 2.5-approximation algorithm for Min-d-RCS for even d (Section 3). The latter is in fact an $(r + 1)$-approximation algorithm for Min-d-RCS, where r is the factor within which Min-TSP can be approximated. This result can be extended

problem	*result*	*reference*
d-RCS	in P for $d \geq \frac{n}{2} - 1$	trivial for $d \geq n/2$, Section 5.2
	NP-complete for constant d	Cheah and Corneil [6]
	and d of any growth rate up to $O(n^{1-\varepsilon})$	Section 4.2
Min-d-RCS	$(r+1)$-approximation for even d	Section 3.2
	3-approximation for odd d	Section 3.1
	2-approximation for $d \geq n/3$	Section 5.1
	no better approximable than Min-TSP	Section 4.1
Min-d-R2CS	3-approximation	Section 3.1
	no better approximable than Min-TSP	Section 4.1
Min-d-ARCS	$(r+1)$-approximation	Section 3.2
	no better approximable than Min-ATSP	Section 4.1
Max-d-RCS	$(1 - \frac{2}{d \cdot (d+1)})$-approximation	Baburin and Gimadi [2]
Max-d-ARCS	$(1 - \frac{1}{d \cdot (d+1)})$-approximation	Section 5.3

Table 1. Overview of the complexity and approximability of finding (optimal) connected d-factors. We left out that all optimization variants are polynomial-time solvable for $d \geq n/2$ and APX-hard according to Sections 4.1 and 4.2. Here, r is the approximation ratio of Min-TSP or Min-ATSP.

to Min-d-ARCS, where r is now the approximation ratio of Min-ATSP. Our approximation algorithms, in particular for the maximization variants, are in the spirit of the classical approximation algorithm of Fisher et al. [10] for Max-TSP: compute a non-connected structure, and then remove and add edges to make it connected.

As lower bounds, we prove that Min-d-RCS and Min-d-ARCS cannot be approximated better than Min-TSP and Min-ATSP, respectively (Section 4). In particular, this implies the APX-hardness of the problems.

We prove some structural properties of connected d-factors and their relation to TSP, MST, and d-factors without connectedness requirement (Section 2). Some of these properties are needed for the approximation algorithms and some might be interesting in their own right or were initially counterintuitive to us.

Our algorithms work for all values of d, even when d is part of the input. The hardness results are extended to the case where d grows with n. In Section 5, we improve our approximation guarantee for $d \geq n/3$, prove that $(\frac{n}{2} - 1)$-RCS $\in$ P, and generalize Baburin and Gimadi's algorithm [2] to directed instances.

2 Structural Properties

In the following two lemmas, we make statements about the relationship between the weights of optimal solutions of the different minimization problems. We call an inequality $A \leq c \cdot B$ *tight* if, for every $\varepsilon > 0$, replacing c by $c - \varepsilon$ does not yield a valid statement for all instances.

Lemma 2.1 (undirected comparison).

1. *$w(\mathsf{MST}) \le w(d\text{-}\mathsf{RCS}) \le w(d\text{-}\mathsf{R2CS})$ for all d and all undirected instances, and this is tight.*
2. *$w(d\text{-}\mathsf{F}) \le w(d\text{-}\mathsf{RCS})$ for all d and all undirected instances, and this is tight.*
3. *$w(d\text{-}\mathsf{R2CS}) \le 3 \cdot w(d\text{-}\mathsf{RCS})$ for all odd d and all undirected instances, and this is tight for all odd d.*
4. *$w(\mathsf{TSP}) \le w(d\text{-}\mathsf{RCS})$ for all even d and all undirected instances, and this is tight.*
5. *$w(\mathsf{TSP}) \le 2 \cdot w(d\text{-}\mathsf{RCS})$ for all odd d and all undirected instances, and this is tight for all odd d.*
6. *$w(\mathsf{TSP}) \le \frac{4}{3} \cdot w(3\text{-}\mathsf{R2CS})$ for all undirected instances, and this is tight.*
7. *For all odd d, there are instances with $w(\mathsf{TSP}) \ge (\frac{4}{3} - o(1)) \cdot w(d\text{-}\mathsf{R2CS})$.*
8. *$w((d-2)\text{-}\mathsf{F}) \le \frac{d-2}{d} \cdot w(d\text{-}\mathsf{F})$ and $w((d-2)\text{-}\mathsf{RCS}) \le w(d\text{-}\mathsf{RCS})$ for all even $d \ge 4$ and all undirected instances, and both inequalities are tight.*
9. *Monotonicity does not hold for odd d: for every odd $d \ge 5$, there exist instances with $w((d-2)\text{-}\mathsf{RCS}) \ge \frac{d+2}{d} \cdot w(d\text{-}\mathsf{RCS})$.*

Lemma 2.2 (directed comparison).

1. *$w(d\text{-}\mathsf{AF}) \le w(d\text{-}\mathsf{ARCS})$ for all d and all directed instances, and this is tight.*
2. *$w(\mathsf{ATSP}) \le w(d\text{-}\mathsf{ARCS})$ for all d and all directed instances, and this is tight.*
3. *$w((d-1)\text{-}\mathsf{AF}) \le \frac{d-1}{d} \cdot w(d\text{-}\mathsf{AF})$ and $w((d-1)\text{-}\mathsf{ARCS}) \le w(d\text{-}\mathsf{ARCS})$ for all $d \ge 2$ and all directed instances, and both inequalities are tight.*

3 Approximation Algorithms

3.1 3-Approximation for Min-d-RCS and Min-d-R2CS

The 3-approximation that we present in this section works for all d, odd or even. It also works for d growing as a function of n. An interesting feature of this algorithm, and possibly an indication that a better approximation ratio is possible for Min-d-RCS, is that the same algorithm provides an approximation ratio of 3 for both Min-d-RCS and Min-d-R2CS. In fact, we compute a 2-edge-connected d-regular graph that weighs at most three times the weight of the optimal connected d-regular graph.

First we make some preparatory observations on 2-edge-connectedness. Given a connected graph $G = (V, E)$, we can create a tree $T(G)$ as follows: We have a vertex for every maximal subgraph of G that is 2-edge-connected (called a 2-edge-connected component), and two such vertices are connected if the corresponding components are connected in G. In this case, they are connected by a bridge edge. Now consider a leaf of tree $T(G)$ and its corresponding 2-edge-connected component C. Since C is a leaf in $T(G)$, it is only incident to a single bridge edge e in G. Now assume that G is d-regular with $d \ge 3$ odd (for $d = 2$, any connected graph is also 2-edge-connected). Let u be the vertex of C that is incident to e. Then u must be incident to $d - 1$ other vertices in C. Thus, C has at least d

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

1 compute a minimum-weight d-factor d-F of G;
2 $k \leftarrow k(d\text{-}\mathsf{F})$
3 $Q \leftarrow \{e_1, \ldots, e_k\}$ with $e_i = e_i(d\text{-}\mathsf{F}) = \{u_i, v_i\}$
4 compute MST of G;
5 duplicate each edge of MST and take shortcuts to obtain a Hamiltonian cycle H
6 take shortcuts to obtain from H a Hamiltonian cycle H' through $\{u_1, \ldots, u_k\}$, assume w.l.o.g. that H' traverses the vertices in the order $u_1, \ldots, u_k, u_1$
7 obtain R from d-F by adding the edges $\{u_i, v_{i+1}\}$ (with $k + 1 = 1$) and removing Q

Algorithm 1: 3-approximation for Min-d-RCS and Min-d-R2CS.

vertices. Since the $d - 1$ neighbors of u are not incident to bridge edges, they must be adjacent to other vertices in C. Since G is d-regular, C has at least $d+1$ vertices and more than $d^2/2 > d$ edges. Therefore, there exists an edge e' in C that is not incident to u, i.e., e' does not share an endpoint with a bridge edge.

If G is not connected, we have exactly the same properties with "tree" replaced by "forest".

To simplify notation in the algorithm, let $k = k(G)$ denote the number of 2-edge-connected components of G that are leaves in the forest described above, and let $L_1(G), \ldots, L_k(G)$ denote the 2-edge-connected components of a graph G that correspond to leaves in the tree described above. For such an $L_i(G)$, let $e_i(G)$ denote an edge that is not adjacent to a bridge edge in G. The choice of $e_i(G)$ is arbitrary.

We prove that Algorithm 1 is a 3-approximation for both Min-d-RCS and Min-d-R2CS by a series of lemmas. Since the set of vertices is fixed, we sometimes identify graphs with their edge set. In particular, R denotes both the connected d-factor that we compute and its edge set.

Lemma 3.1. *Assume that R is computed as in Algorithm 1. Then R is a d-regular spanning subgraph of G.*

Lemma 3.2. *Assume that R is computed as in Algorithm 1. Then R is 2-edge-connected.*

Lemma 3.3. *Assume that R is computed as in Algorithm 1. Then $w(R) \leq 3 \cdot w(d\text{-RCS}) \leq 3 \cdot w(d\text{-R2CS})$.*

The following theorem is an immediate consequence of the lemmas above.

Theorem 3.4. *For all d, Algorithm 1 is a polynomial-time 3-approximation for* Min-d-RCS *and* Min-d-R2CS*. This includes the case that d is a function of n.*

Remark 3.5. If we are only interested in a 3-approximation for Min-d-RCS and not for Min-d-R2CS, then we can simplify Algorithm 1 a bit: we only pick one non-bridge edge for each component and not for every 2-edge-connected component.

The rest of the algorithm and its analysis remain the same. However, this does not seem to improve the worst-case approximation ratio.

Remark 3.6. The analysis is tight in the following sense: By Lemma 2.1(3), a minimum-weight 2-edge-connected d-factor can be three times as heavy as a minimum-weight connected d-factor. Thus, any algorithm that outputs a 2-edge-connected d-factor cannot achieve an approximation ratio better than 3. Furthermore, since $w(\mathsf{MST}) \leq w(d\text{-}\mathsf{R2CS})$ and $w(d\text{-}\mathsf{F}) \leq w(d\text{-}\mathsf{R2CS})$ are tight (Lemma 2.1(1) and (2)), the analysis is essentially tight. If we only require connectedness and not 2-edge-connectedness, we see that the analysis cannot be improved since $w(\mathsf{TSP}) \leq 2w(d\text{-}\mathsf{RCS})$ and $w(d\text{-}\mathsf{F}) \leq w(d\text{-}\mathsf{RCS})$ are tight.

However, it is reasonable to assume that not all these inequalities can be tight at the same time and, in addition, taking shortcuts in the duplicated MST to obtain a TSP tour through $u_1, \ldots, u_k$ does not yield any improvement. Therefore, it might be possible to improve the analysis and show that Algorithm 1 achieves a better approximation ratio than 3.

Remark 3.7. Lines 4 and 5 of Algorithm 1 are in fact the double-tree heuristic for Min-TSP [26, Section 2.4]. One might be tempted to construct a better tour using Christofides' algorithm [26, Section 2.4], which achieves a ratio of 3/2 instead of only 2. However, in the analysis we compare the optimal solution for Min-d-RCS to the MST, and we know that $w(\mathsf{MST}) \leq w(d\text{-}\mathsf{RCS}) \leq w(d\text{-}\mathsf{R2CS})$. If we use Christofides' algorithm directly, we have to compare a TSP tour to the minimum-weight connected d-factor. In particular for odd d, we have that for some instances $w(\mathsf{TSP}) \geq (\frac{4}{3} - o(1)) \cdot w(d\text{-}\mathsf{R2CS}) \geq (\frac{4}{3} - o(1)) \cdot w(d\text{-}\mathsf{RCS})$ (Lemma 2.1(7)). Even if this is the true bound – as it is for $d = 3$ (Lemma 2.1(6)) –, the TSP tour constructed contributes with a factor 3/2 times 4/3, which equals 2, to the approximation ratio, which is no improvement.

3.2 $(r + 1)$-Approximation

In this section, we give an $(r+1)$-approximation for Min-d-RCS for even values of d and Min-d-ARCS for all values of d. Here, r is the ratio within which Min-TSP (for Min-d-RCS) or Min-ATSP (for Min-d-ARCS) can be approximated. This means that we currently have $r = 3/2$ for the symmetric case by Christofides' algorithm [26, Section 2.4] and, for the asymmetric case, we have either $r = O(\log n / \log \log n)$ if we use the randomized algorithm by Asadpour et al. [1] or $r = \frac{2}{3} \cdot \log_2 n$ if we use Feige and Singh's deterministic algorithm [9]. Although the algorithm is a simple modification of Algorithm 1, we summarize it as Algorithm 2 for completeness.

Theorem 3.8. *If* Min-TSP *can be approximated in polynomial time within a factor of r, then Algorithm 2 is a polynomial-time $(r + 1)$-approximation for* Min-d-RCS *for all even d.*

If Min-ATSP *can be approximated in polynomial time within a factor of r, then Algorithm 2 is a polynomial-time $(r+1)$-approximation for* Min-d-ARCS *for all d.*

The results still hold if d is part of the input.

input : undirected or directed complete graph $G = (V, E)$, edge weights w, d
output: connected d-factor R of G

compute a minimum-weight d-factor C of G
let $C_1, \ldots, C_k$ be the connected components of C, and let $e_i = (u_i, v_i)$ be any edge of C_i
compute a TSP tour H using an approximation algorithm with ratio r
take shortcuts to obtain from H a TSP tour H' through $\{u_1, \ldots, u_k\}$, assume w.l.o.g. that H' traverses the vertices in the order $u_1, \ldots, u_k, u_1$
obtain R from C by adding the edges (u_i, v_{i+1}) (with $k + 1 = 1$) and removing $e_1, \ldots, e_k$

Algorithm 2: $(r + 1)$-approximation for Min-d-RCS for even d and Min-d-ARCS.

4 Hardness Results

4.1 TSP-Inapproximability

In this section, we prove that Min-d-RCS cannot be approximated better than Min-TSP.

Theorem 4.1. *For every $d \geq 2$, if* Min-d-RCS *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.*

The same construction as in the proof of Theorem 4.1 yields the same result for Min-d-R2CS. A similar construction yields the same result for Min-d-ARCS.

Corollary 4.2. *For every $d \geq 2$, if* Min-d-R2CS *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.*

Corollary 4.3. *For every $d \geq 2$, if* Min-d-ARCS *can be approximated in polynomial time within a factor of r, then* Min-ATSP *can be approximated in polynomial time within a factor of r.*

Min-TSP, Min-ATSP, Max-TSP, and Max-ATSP are APX-hard [22]. Furthermore, the reduction from Min-TSP to Min-d-RCS is in fact an L-reduction [21] (see also Shmoys and Williamson [26, Section 16.2]). This proves the APX-hardness of Min-d-RCS for all d. The reductions from Min-TSP to Min-d-R2CS and from Min-ATSP to Min-d-ARCS work in the same way. Furthermore, by reducing from Max-TSP and Max-ATSP in a similar way (here, the edges between the copies of a vertex have high weight), we obtain APX-hardness for Max-d-RCS and Max-d-ARCS as well.

Corollary 4.4. *For every fixed $d \geq 2$, the problems* Min-d-RCS, Min-d-R2CS, *and* Max-d-RCS *are* APX*-complete. For every fixed $d \geq 1$,* Min-d-ARCS *and* Max-d-ARCS *are* APX*-complete.*

4.2 Hardness for Growing d

In this section, we generalize the NP-hardness proof for d-RCS by Cheah and Corneil [6] to the case that d grows with n. Furthermore, we extend Theorem 4.1 and Corollaries 4.2 and 4.3 and the APX-hardness of the minimization variants (Corollary 4.4) to growing d. The APX-hardness of Max-d-RCS and Max-d-ARCS does not transfer to growing d – both can be approximated within a factor of $1 - O(1/d^2)$, which is $1 - o(1)$ for growing d.

Let us consider Cheah and Corneil's [6, Section 3.2] reduction from 2-RCS, i.e., the Hamiltonian cycle problem, to d-RCS. 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 $G \in$ 2-RCS 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 $G \in$ 2-RCS if and only if $R_d(G) \in d$-RCS.

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 4.5. *For every fixed $\varepsilon > 0$, there is a function $f = \Theta(n^{1-\varepsilon})$ that maps to even integers such that f-RCS is NP-hard.*

For every fixed $\varepsilon > 0$, there is a function $f = \Theta(n^{\frac{1}{2}-\varepsilon})$ that maps to odd integers such that f-RCS is NP-hard.

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 4.1, and the proof follows the line of the proof of Theorem 4.5. Here, however, we do not have to distinguish between odd and even d for the symmetric variant, as the reduction in Section 4.1 is the same for both cases.

Theorem 4.6. *For every fixed $\varepsilon > 0$, there is a function $f = \Theta(n^{1-\varepsilon})$ such that* Min-f-RCS *and* Min-f-R2CS *are* APX-*hard and cannot be approximated better than* Min-TSP.

For every fixed $\varepsilon > 0$, there is a function $f = \Theta(n^{1-\varepsilon})$ such that Min-f-ARCS *is* APX-*hard and cannot be approximated better than* Min-ATSP.

5 Further Algorithms

5.1 2-Approximation for $d \geq n/3$

If $d \geq n/3$, then we easily get a better approximation algorithm for Min-d-R2CS and Min-d-RCS. In this case, d-F consists either of a single component – then we are done – or of two components C_1 and C_2 with $C_i = (V_i, E_i)$. In the latter case, we proceed as follows: first, find the lightest edge $e = \{u, v\}$ with $u \in V_1$ and $v \in V_2$. Second, choose any edges $\{u, u'\} \in E_1$ and $\{v, v'\} \in E_2$. Third, remove $\{u, u'\}$ and $\{v, v'\}$ and add $\{u, v\}$ and $\{u', v'\}$. The increase in weight is at most $2 \cdot w(\{u, v\})$ by the triangle inequality.

The resulting graph is clearly d-regular. It is connected since C_1 and C_2 are 2-edge-connected: they both consist of at most $\frac{2n}{3} - 1$ vertices and are d-regular with $d \geq n/3$. Thus, they are even Hamiltonian by Dirac's theorem [25]. Furthermore, any connected d-regular graph must have at least two edges connecting V_1 and V_2: If d is even, then this follows by 2-edge-connectedness. If d is odd, then $|V_1|$ and $|V_2|$ are even and, thus, an even number of edges must leave either of them. Thus, $w(\{u, v\}) \leq \frac{1}{2} \cdot w(d\text{-RCS})$. Since we add at most $2 \cdot w(\{u, v\})$ and also have $w(d\text{-F}) \leq w(d\text{-RCS})$, we obtain the following theorem.

Theorem 5.1. *For $d \geq n/3$, there is a polynomial-time 2-approximation for* Min-d-RCS.

5.2 Decision Problem for $d = \lceil \frac{n}{2} \rceil - 1$

For $d \geq n/2$, any d-factor is immediately connected and also the minimization variant can be solved efficiently. In this section, we slightly extend this to the case of $d \geq \frac{n}{2} - 1$.

We assume that the input graph G is connected. To show that the case $d = \lceil \frac{n}{2} \rceil - 1$ is in P, we compute a d-factor. If none exists or we obtain a connected d-factor, then we are done. Otherwise, we have a d-factor consisting of two components C_1 and C_2 which are both cliques of size $n/2$. If G contains a cut vertex, say, $u \in C_1$, then this is the only vertex with neighbors in C_2. In this case, G does not contain a connected d-factor. If G does not contain a cut vertex, there are two disjoint edges $e = \{u, v\}$, $e' = \{u', v'\}$ with $u, u' \in C_1$ and $v, v' \in C_2$. Adding e and e' and removing $\{u, u'\}$ and $\{v, v'\}$ yields a connected d-factor.

Theorem 5.2. d-RCS *is in* P *for every d with $d \geq \frac{n}{2} - 1$.*

5.3 Approximating Max-d-ARCS

The approximation algorithm for Max-d-RCS [2] can easily be adapted to work for Max-d-ARCS: We compute a directed d-factor of maximum weight. Any component consists of at least $d + 1$ vertices, thus at least $d \cdot (d + 1)$ arcs. We remove the lightest arc of every component and connect the resulting (still at least weakly connected) components arbitrarily to obtain a connected d-factor. Since we have removed at most a $\frac{1}{d \cdot (d+1)}$-fraction of the weight, we obtain the following result.

Theorem 5.3. *For every d,* Max-d-ARCS *can be approximated within a factor of $1 - \frac{1}{d \cdot (d+1)}$.* □

6 Open Problems

An obvious open problem is to improve the approximation ratios. Apart from this, let us mention two open problems: First, is it possible to achieve constant factor approximations for minimum-weight k-edge-connected or k-vertex-connected d-regular graphs? Without the regularity requirement, the problem of computing minimum-weight k-edge-connected graphs can be approximated within a factor of 2 [17] and the problem of computing minimum-weight k-vertex-connected graphs can be approximated within a factor of $2 + 2 \cdot \frac{k-1}{n}$ for metric instances [15] and still within a factor of $O(\log k)$ if the instances are not required to satisfy the triangle inequality [7]. We refer to Khuller and Raghavachari [16] for a concise survey.

Second, we have seen that $(\lceil \frac{n}{2} \rceil - 1)$-$\mathsf{RCS} \in \mathsf{P}$, but we do not know if Min-$(\lceil \frac{n}{2} \rceil - 1)$-$\mathsf{RCS}$ can be solved in polynomial time as well. In addition, we conjecture that also $(\lceil \frac{n}{2} \rceil - k)$-$\mathsf{RCS}$ is in P for any constant k.

References

1. Asadpour, A., Goemans, M.X., Madry, A., Gharan, S.O., Saberi, A.: An $O(\log n/ \log \log n)$-approximation algorithm for the asymmetric traveling salesman problem. In: Proc. of the 21st Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA). pp. 379–389. SIAM (2010)
2. Baburin, A.E., Gimadi, E.K.: Approximation algorithms for finding a maximum-weight spanning connected subgraph with given vertex degrees. In: Operations Research Proceedings 2004. pp. 343–351 (2005)
3. Baburin, A.E., Gimadi, E.K.: Polynomial algorithms for some hard problems of finding connected spanning subgraphs of extreme total edge weight. In: Operations Research Proceedings 2006. pp. 215–220 (2007)
4. Baburin, A.E., Gimadi, E.K.: An approximation algorithm for finding a d-regular spanning connected subgraph of maximum weight in a complete graph with random weights of edges. Journal of Applied and Industrial Mathematics 2(2), 155–166 (2008)
5. Chan, Y.H., Fung, W.S., Lau, L.C., Yung, C.K.: Degree bounded network design with metric costs. SIAM Journal on Computing 40(4), 953–980 (2011)
6. Cheah, F., Corneil, D.G.: The complexity of regular subgraph recognition. Discrete Applied Mathematics 27(1-2), 59–68 (1990)
7. Cheriyan, J., Vempala, S., Vetta, A.: An approximation algorithm for the minimum-cost k-vertex connected subgraph. SIAM Journal on Computing 32(4), 1050–1055 (2003)
8. Escoffier, B., Gourvès, L., Monnot, J.: Complexity and approximation results for the connected vertex cover problem in graphs and hypergraphs. Journal of Discrete Algorithms 8(1), 36–49 (2010)
9. Feige, U., Singh, M.: Improved approximation ratios for traveling salesperson tours and paths in directed graphs. In: Proc. of the 10th Int. Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX). Lecture Notes in Computer Science, vol. 4627, pp. 104–118. Springer (2007)
10. Fisher, M.L., Nemhauser, G.L., Wolsey, L.A.: An analysis of approximation for finding a maximum weight Hamiltonian cycle. Operations Research 27(4), 799–809 (1979)

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. Gimadi, E.K., Serdyukov, A.I.: A problem of finding the maximal spanning connected subgraph with given vertex degrees. In: Operations Reserach Proceedings 2000. pp. 55–59. Springer (2001)
13. Guha, S., Khuller, S.: Improved methods for approximating node weighted steiner trees and connected dominating sets. Information and Computation 150(1), 57–74 (1999)
14. Kaplan, H., Lewenstein, M., Shafrir, N., Sviridenko, M.I.: Approximation algorithms for asymmetric TSP by decomposing directed regular multigraphs. Journal of the ACM 52(4), 602–626 (2005)
15. Khuller, S., Raghavachari, B.: Improved approximation algorithms for uniform connectivity problems. Journal of Algorithms 21(2), 434–450 (1996)
16. Khuller, S., Raghavachari, B.: Graph connectivity. In: Kao, M.Y. (ed.) Encyclopedia of Algorithms. Springer (2008)
17. Khuller, S., Vishkin, U.: Biconnectivity approximations and graph carvings. Journal of the ACM 41(2), 214–235 (1994)
18. Lovász, L., Plummer, M.D.: Matching Theory, North-Holland Mathematics Studies, vol. 121. Elsevier (1986)
19. Marx, D., O'sullivan, B., Razgon, I.: Finding small separators in linear time via treewidth reduction. ACM Transactions on Algorithms 9(4), 30:1–30:35 (2013)
20. Paluch, K., Mucha, M., Madry, A.: A 7/9 approximation algorithm for the maximum traveling salesman problem. In: Proc. of the 12th Int. Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX). Lecture Notes in Computer Science, vol. 5687, pp. 298–311. Springer (2009)
21. Papadimitriou, C.H., Yannakakis, M.: Optimization, approximation, and complexity classes. Journal of Computer and System Sciences 43(3), 425–440 (1991)
22. Papadimitriou, C.H., Yannakakis, M.: The traveling salesman problem with distances one and two. Mathematics of Operations Research 18(1), 1–11 (1993)
23. Singh, M., Lau, L.C.: Approximating minimum bounded degree spanning trees to within one of optimal. In: Proc. of the 39th Ann. Int. Symp. on Theory of Computing (STOC). pp. 661–670. ACM (2007)
24. Tutte., W.T.: A short proof of the factor theorem for finite graphs. Canadian Journal of Mathematics 6, 347–352 (1954), http://dx.doi.org/10.4153/CJM-1954-033-3
25. West, D.B.: Introduction to Graph Theory. Prentice Hall (2001)
26. Williamson, D.P., Shmoys, D.B.: The Design of Approximation Algorithms. Cambridge University Press (2011)