Discrete Applied Mathematics 157 (2009) 1470–1480

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Minimum-weight cycle covers and their approximability ☆

Bodo Manthey *

Saarland University, Computer Science, Postfach 151150, 66041 Saarbrücken, Germany

ARTICLE INFO

Article history:
Received 19 September 2007
Received in revised form 5 September 2008
Accepted 17 October 2008
Available online 28 November 2008

Keywords:
Approximation algorithms
Cycle covers
Inapproximability
Non-constructive algorithms

ABSTRACT

A cycle cover of a graph is a set of cycles such that every vertex is part of exactly one cycle. An L-cycle cover is a cycle cover in which the length of every cycle is in the set $L \subseteq \mathbb{N}$.

We investigate how well L-cycle covers of minimum weight can be approximated. For undirected graphs, we devise non-constructive polynomial-time approximation algorithms that achieve constant approximation ratios for all sets L. On the other hand, we prove that the problem cannot be approximated with a factor of $2 - \varepsilon$ for certain sets L.

For directed graphs, we devise non-constructive polynomial-time approximation algorithms that achieve approximation ratios of $O(n)$, where n is the number of vertices. This is asymptotically optimal: We show that the problem cannot be approximated with a factor of $o(n)$ for certain sets L.

To contrast the results for cycle covers of minimum weight, we show that the problem of computing L-cycle covers of maximum weight can, at least in principle, be approximated arbitrarily well.

1. Introduction

A cycle cover of a graph is a spanning subgraph that consists solely of cycles such that every vertex is part of exactly one cycle. Cycle covers are an important tool for the design of approximation algorithms for different variants of the traveling salesman problem [3,5,6,9–12,21], for the shortest common superstring problem from computational biology [8,28], and for vehicle routing problems [18].

In contrast to Hamiltonian cycles, which are special cases of cycle covers, cycle covers of minimum weight can be computed efficiently. This is exploited in the above-mentioned algorithms, which in general start by computing a cycle cover and then join cycles to obtain a Hamiltonian cycle (this technique is called *subtour patching* [14]).

Short cycles limit the approximation performances achieved by such algorithms. Roughly speaking, the longer the cycles in the initial cover, the better the approximation ratio. Thus, we are interested in computing cycle covers without short cycles. Moreover, there are algorithms that perform particularly well if the cycle covers computed do not contain cycles of odd length [5]. Finally, some vehicle routing problems [18] require covering vertices with cycles of bounded length. Therefore, we consider *restricted cycle covers*, where cycles of certain lengths are ruled out a priori: For a set $L \subseteq \mathbb{N}$, an *L-cycle cover* is a cycle cover in which the length of each cycle is in L.

Unfortunately, computing L-cycle covers is NP-hard for almost all sets L [20,23]. Thus, in order to fathom the possibility of designing approximation algorithms based on computing cycle covers, our aim is to find out how well L-cycle covers can be approximated.

Beyond being a basic tool for approximation algorithms, cycle covers are interesting in their own right. Matching theory and graph factorization are important topics in graph theory. The classical matching problem is the problem of finding one-factors, i. e., spanning subgraphs in which every vertex is incident to exactly one edge. Cycle covers of undirected graphs are

☆ A preliminary version of this work has been presented at the 33rd Workshop on Graph-Theoretic Concepts in Computer Science (WG 2007).

* Tel.: +49 681 3025502; fax: +49 681 3025576.
E-mail address: manthey@cs.uni-sb.de.

0166-218X/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2008.10.005

also called two-factors since every vertex is incident to exactly two edges in a cycle cover. Both structural properties of graph factors and the complexity of finding graph factors have been the topic of a considerable amount of research (cf. Lovász and Plummer [22] and Schrijver [27]).

1.1. Preliminaries

Let $G = (V, E)$ be a graph with vertex set V and edge set E. If G is undirected, then a **cycle cover** of G is a subset $C \subseteq E$ of the edges of G such that all vertices in V are incident to exactly two edges in C. If G is a directed graph, then a cycle cover of G is a subset $C \subseteq E$ such that all vertices are incident to exactly one incoming and one outgoing edge in C. Thus, the graph (V, C) consists solely of vertex-disjoint cycles. The length of a cycle is the number of edges it consists of. We are concerned with simple graphs, i. e., the graphs that do not contain multiple edges or loops. Thus, the shortest cycles of undirected and directed graphs are of length three and two, respectively. We call a cycle of length λ a **$\boldsymbol{\lambda}$-cycle** for short.

An ***L*-cycle cover** of an undirected graph is a cycle cover in which the length of every cycle is in the set $L \subseteq \mathcal{U} = \{3, 4, 5, \ldots\}$. An L-cycle cover of a directed graph is analogously defined except that $L \subseteq \mathcal{D} = \{2, 3, 4, \ldots\}$. A special case of L-cycle covers are ***k*-cycle covers**, which are $\{k, k+1, \ldots\}$-cycle covers. Let $\overline{L} = \mathcal{U} \setminus L$ in the case of undirected graphs, and let $\overline{L} = \mathcal{D} \setminus L$ in the case of directed graphs (whether we consider undirected or directed cycle covers will be clear from the context).

Given edge weights $w : E \to \mathbb{N}$, the **weight $\boldsymbol{w(C)}$** of a subset $C \subseteq E$ of the edges of G is $w(C) = \sum_{e \in C} w(e)$. In particular, this defines the weight of a cycle cover since we view cycle covers as sets of edges.

Min-*L*-UCC is the following optimization problem: Given an undirected complete graph with non-negative edge weights that satisfy the triangle inequality ($w(\{u, v\}) \leq w(\{u, x\}) + w(\{x, v\})$ for all $u, x, v \in V$) find an L-cycle cover of minimum weight. **Min-*k*-UCC** is defined for $k \in \mathcal{U}$ like Min-L-UCC except that k-cycle covers rather than L-cycle covers are sought. The triangle inequality is not only a natural restriction, it is also necessary: If finding L-cycle covers in graphs is NP-hard, then Min-L-UCC without the triangle inequality does not allow for any approximation at all. This can be seen by reduction from the decision problem whether a graph contains an L-cycle cover (the proof is similar to the inapproximability of the traveling salesman problem without triangle inequality [26]): Given an instance $G = (V, E)$ for which we want to decide whether it contains an L-cycle cover, create a complete graph on V with weights $w(e) = 1$ if $e \in E$ and $w(e) = \alpha$ for some large $\alpha \gg n$. If G possesses an L-cycle cover, then the new graph possesses an L-cycle cover of weight n. Otherwise, any L-cycle cover of the new graph has a weight of at least α.

Min-*L*-DCC and **Min-*k*-DCC** are defined for directed graphs like Min-L-UCC and Min-k-UCC for undirected graphs except that $L \subseteq \mathcal{D}$ and $k \in \mathcal{D}$ and the triangle inequality is of the form $w(u, v) \leq w(u, x) + w(x, v)$. Again, the triangle inequality is mandatory for the existence of approximation algorithms.

Finally, **Max-*L*-UCC** , **Max-*k*-UCC** , **Max-*L*-DCC** , and **Max-*k*-DCC** are analogously defined except that cycle covers of maximum weight are sought and that the edge weights do not have to fulfill the triangle inequality.

1.2. Previous results

Min-$\mathcal{U}$-UCC , i. e., the undirected cycle cover problem without any restrictions, can be solved in polynomial time via Tutte's reduction to the classical perfect matching problem [22]. By a modification of an algorithm of Hartvigsen [17], also 4-cycle covers of minimum weight in graphs with edge weights one and two can be computed efficiently. For Min-k-UCC restricted to graphs with edge weights one and two, there exists a factor 7/6 approximation algorithm for all k [7]. Hassin and Rubinstein [19] presented a randomized approximation algorithm for Max-$\{3\}$-UCC that achieves an approximation ratio of $83/43 + \epsilon$. Max-L-UCC admits a factor 2 approximation algorithm for arbitrary sets L [23]. Goemans and Williamson [15] showed that Min-k-UCC and Min-$\{k\}$-UCC can be approximated with a factor of 4. Min-L-UCC is NP-hard and APX-hard if $\overline{L} \not\subseteq \{3\}$, i. e., for all but a finite number of sets L [20,23,29]. This means that for almost all L, these problems are unlikely to possess polynomial-time approximation schemes (PTAS, see Ausiello et al. [2] for a definition).

Min-$\mathcal{D}$-DCC , which is also known as the *assignment problem*, can be solved in polynomial time by a reduction to the minimum weight perfect matching problem in bipartite graphs [1]. The only other L for which Min-L-DCC can be solved in polynomial time is $L = \{2\}$. For all $L \subseteq \mathcal{D}$ with $L \neq \{2\}$ and $L \neq \mathcal{D}$, Min-L-DCC and Max-L-DCC are APX-hard and NP-hard, even if only two different edge weights are allowed [23]. There is a 4/3 approximation algorithm for Max-3-DCC [6] as well as for Min-k-DCC for $k \geq 3$ with the restriction that the only edge weights allowed are one and two [4]. Max-L-DCC can be approximated with a factor of 8/3 for all L [23].

1.3. New results

While L-cycle covers of *maximum* weight allow for constant factor approximations, only little is known so far about the approximability of computing L-cycle covers of *minimum* weight. Our aim is to close this gap.

We prove that approximation algorithms exist for Min-L-UCC for all sets $L \subseteq \mathcal{U}$. The approximation ratios achieved are constant; they depend only on the set L (Section 2.1). More specifically, we present an algorithm into which a finite set

$L' \subseteq \mathcal{U}$ is "hardwired" that achieves constant approximation ratio for Min-L'-UCC. Given a set L, our algorithm, equipped with an appropriate $L' \subseteq L$, yields also an approximation algorithm for Min-L-UCC.

On the other hand, we show that the problem cannot be approximated with a factor of $2 - \varepsilon$ for general L (Section 2.2).

Then we transfer our results to Min-L-DCC, for which we achieve a ratio of $O(n)$, where n is the number of vertices (Section 3.1). This is asymptotically optimal: There exist sets L for which no algorithm can approximate Min-L-DCC with a factor of $o(n)$ (Section 3.2).

Finally, to contrast our results for Min-L-UCC and Min-L-DCC, we show that Max-L-UCC and Max-L-DCC can be approximated arbitrarily well at least in principle (Section 4).

2. Approximability of Min-*L*-UCC

2.1. An approximation algorithm for Min-L-UCC

The aim of this section is to prove the existence of approximation algorithms for Min-L-UCC for all sets $L \subseteq \mathcal{U}$. The catch is that for most L it is impossible to decide whether some cycle length is in L since there are uncountably many sets L: If, for instance, L is not a recursive set, then deciding if a cycle cover is an L-cycle cover is impossible. One option would be to restrict ourselves to sets L such that the unary language $\{1^\lambda \mid \lambda \in L\}$ is in P. For such L, Min-L-UCC and Min-L-DCC are NP optimization problems (see Ausiello et al. [2] for a definition). Another possibility for circumventing the problem would be to include the permitted cycle lengths in the input. While such restrictions are mandatory if we want to compute optimum solutions, they are not needed for our approximation algorithms.

A complete n-vertex graph contains an L-cycle cover as a spanning subgraph if and only if there exist (not necessarily distinct) lengths $\lambda_1, \ldots, \lambda_k \in L$ for some $k \in \mathbb{N}$ with $\sum_{i=1}^{k} \lambda_i = n$. We call such an n ***L*-admissible** and define $\langle L \rangle = \{n \mid n \text{ is } L\text{-admissible}\}$. Although L might not be a recursive set, $\langle L \rangle$ allows efficient membership testing according to the following lemma.

Lemma 2.1 (*Manthey [23, Lem. 3.1]*). *For all $L \subseteq \mathbb{N}$, there exists a finite set $L' \subseteq L$ with $\langle L' \rangle = \langle L \rangle$.*

In the following, L will always be the set of cycle lengths we are actually interested in, while $L' \subseteq L$ will be a finite set according to the lemma above. Unfortunately, there is no effective way of obtaining a finite set L' from L. In this sense, the proof of approximability is nonconstructive, similar to the nonconstructive proof that any minor-closed family of graphs can be decided in polynomial time [13]. But at least for many "natural" sets L, an appropriate finite subset L' can be found easily: If L itself is finite, then, of course, $L = L'$ will do. If $\overline{L}$ is finite, then L' can also be found easily. More generally, if the set L contains exactly all multiples of a certain number g above a certain threshold p (it can contain any subset of the numbers smaller than p), then L' can also be computed easily.

To cope with this problem, we always assume that the finite set L' is given and hardwired into our algorithm. Since there are only countably many finite sets L', we obtain a countable number of approximation algorithms for an uncountable number of optimization problems. Then we prove that this algorithm achieves a constant approximation ratio for Min-L-UCC for any $L \supseteq L'$ with $\langle L \rangle = \langle L' \rangle$.

Let g_L be the greatest common divisor of all numbers in L. Then $\langle L \rangle$ is a subset of the set of natural numbers divisible by g_L. The proof of Lemma 2.1 shows that there exists a minimum $p_L \in \mathbb{N}$ such that $\eta g_L \in \langle L \rangle$ for all $\eta > p_L$. The number p_L is the Frobenius number [25] of the set $\{\lambda \mid g_L \lambda \in L\}$, which is L scaled down by g_L. For instance, if $L = \{8, 10\}$, then $g_L = 2$ and $p_L = 11$ since the Frobenius number of $\{4, 5\}$ is 11.

In the remainder of this section, we will allow 2-cycles, where an undirected 2-cycle consisting of vertices u and v contains the edge $\{u, v\}$ twice. (It also contributes twice its weight to the weight of the cycle cover.) We allow 2-cycles in order to be prepared for the directed variant of the problem (Section 3.1).

In the following, $L \subseteq \mathcal{U} \cup \{2\} = \mathcal{D}$ will be arbitrary, and $L' \subseteq L$ will be chosen so as to fulfill Lemma 2.1. Note that $p_L = p_{L'}$ and $g_L = g_{L'}$. We compare the weight of the L'-cycle cover computed to the weight of an optimal $\langle L' \rangle$-cycle cover to bound the approximation ratio. Every L'-cycle cover is also an L-cycle cover. Furthermore, $L \subseteq \langle L \rangle = \langle L' \rangle$. Thus, the weight of an optimum $\langle L' \rangle$-cycle cover is no greater than the weight of an optimum L-cycle cover. Thus, the ratio of the weight of the cycle cover computed and the weight of the optimum $\langle L' \rangle$-cycle cover will provide an upper bound for the approximation ratio for Min-L-UCC.

Goemans and Williamson have presented a technique for approximating constrained forest problems [15], which we will exploit. Let $G = (V, E)$ be an undirected graph, and let $w : E \to \mathbb{N}$ be non-negative edge weights. Let 2^V denote the power set of V. A function $f : 2^V \to \{0, 1\}$ is called a **proper function** if it satisfies

- $f(S) = f(V \setminus S)$ for all $S \subseteq V$ (symmetry),
- if A and B are disjoint, then $f(A) = f(B) = 0$ implies $f(A \cup B) = 0$ (disjointness), and
- $f(V) = 0$.

The aim is to find a set F of edges such that there is an edge connecting S to $V \setminus S$ for all $S \subseteq V$ with $f(S) = 1$. (The name "constrained forest problems" comes from the fact that it suffices to consider forests as solutions; cycles only increase the weight of a solution.) For instance, the minimum spanning tree problem corresponds to the proper function f with $f(S) = 1$ for all S with $\emptyset \subsetneq S \subsetneq V$.

Goemans and Williamson have presented an approximation algorithm [15, Fig. 1] for constrained forest problems that are characterized by proper functions. We will refer to their algorithm as GoeWill.

Theorem 2.2 (*Goemans, Williamson [15, Thm. 2.4]*). *Let ℓ be the number of vertices v with $f(\{v\}) = 1$. Then* GoeWill *is a $(2 - \frac{2}{\ell})$-approximation for the constrained forest problem defined by a proper function f.*

In particular, the function $f_{L'}$ given by

$$f_{L'}(S) = \begin{cases} 1 & \text{if } |S| \not\equiv 0 \pmod{g_{L'}} \text{ and} \\ 0 & \text{if } |S| \equiv 0 \pmod{g_{L'}} \end{cases}$$

is proper if $|V| = n$ is divisible by $g_{L'}$. (If n is not divisible by $g_{L'}$, then G does not contain an L'-cycle cover at all.) Given this function, a solution is a forest $H = (V, F)$ such that the size of every connected component of H is a multiple of $g_{L'}$. In particular, if $g_{L'} = 1$, then $f_{L'}(S) = 0$ for all S, and an optimum solution is n isolated vertices.

If the size of all components of the solution obtained are in $\langle L' \rangle$, we are done: By duplicating all edges, we obtain Eulerian components. Then we construct an $\langle L' \rangle$-cycle cover by traversing the Eulerian components and taking shortcuts whenever we come to a vertex that we have already visited. Finally, we divide each λ-cycle into paths of lengths $\lambda_1 - 1, \dots, \lambda_k - 1$ for some k such that $\lambda_1 + \cdots + \lambda_k = \lambda$ and $\lambda_i \in L'$ for all i. By connecting the respective endpoints of each path, we obtain cycles of lengths $\lambda_1, \dots, \lambda_k$. We perform this for all components to get an L'-cycle cover. A straightforward analysis yields an approximation ratio of 8. A more careful analysis shows that the actual ratio achieved is 4. The details for the special case of $L' = \{k\}$ are spelled out by Goemans and Williamson [15].

However, this procedure does not work for general sets L' since the sizes of some components may not be in $\langle L' \rangle$. This can happen if $p_{L'} > 0$ (for $L' = \{k\}$, for which the algorithm works, we have $p_{L'} = 0$). At the end of this section, we argue why it seems to be difficult to generalize the approach of Goemans and Williamson in order to obtain an approximation algorithm for Min-L-UCC whose approximation ratio is independent of L.

In the following, our aim is to add edges to the forest $H = (V, F)$ output by GoeWill such that the size of each component is in $\langle L' \rangle$. This will lead to an approximation algorithm for Min-L-UCC with a ratio of $4 \cdot (p_L + 4)$, which is constant for each L. Let F^* denote the set of edges of a minimum-weight forest such that the size of each component is in $\langle L \rangle$. The set F^* is a solution to G, w, and f_L, but not necessarily an optimum solution.

By Theorem 2.2, we have $w(F) \le 2 \cdot w(F^*)$ since $w(F^*)$ is at least the weight of an optimum solution to G, w, and f_L. Let $C = (V', F')$ be any connected component of F with $|V'| \notin \langle L \rangle$. The optimum solution F^* must contain an edge that connects V' to $V \setminus V'$. The weight of this edge is at least the weight of the minimum-weight edge connecting V' to $V \setminus V'$.

We will add edges until the sizes of all components is in $\langle L \rangle$. Our algorithm acts in phases as follows: Let $H = (V, F)$ be the graph at the beginning of the current phase, and let $C_1, \dots, C_a$ be its connected components, where V_i is the vertex set of C_i. We will construct a new graph $\tilde{H} = (V, \tilde{F})$ with $\tilde{F} \supseteq F$. Let $C_1, \dots, C_b$ be the connected components with $|V_i| \notin \langle L \rangle$. We call these components **illegal**. For $i \in \{1, \dots, b\}$, let e_i be the cheapest edge connecting V_i to $V \setminus V_i$. (Note that $e_i = e_j$ for $i \neq j$ is allowed.)

We add all these edges to F to obtain $\tilde{F} = F \cup \{e_1, \dots, e_b\}$. Since e_i is the cheapest edge connecting V_i to $V \setminus V_i$, the graph $\tilde{H} = (V, \tilde{F})$ is a forest. (If some e_i are not uniquely determined, cycles may occur. We can avoid these cycles by discarding some of the e_i to break the cycles. For the sake of simplicity, we ignore this case in the following analysis.) If $\tilde{H}$ still contains illegal components, we set H to be $\tilde{H}$ and iterate the procedure.

Lemma 2.3. *Let F and $\tilde{F}$ be as described above. Then $w(\tilde{F}) \le w(F) + 2 \cdot w(F^*)$.*

Proof. We observe that F^* contains at least one edge e_i^* connecting V_i to $V \setminus V_i$ for every $i \in \{1, \dots, b\}$. If $e_i^* = e_j^*$ for $i \neq j$, then $e_k^* \neq e_i^*$ for all $k \neq i, j$. This means that every edge occurs at most twice among $e_1^*, \dots, e_b^*$, which implies

$$\sum_{i=1}^{b} w(e_i^*) \le 2 \cdot w(F^*).$$

By the choice of e_i, we have $w(e_i) \le w(e_i^*)$. Putting everything together yields

$$w(\tilde{F}) \le w(F) + \sum_{i=1}^{b} w(e_i) \le w(F) + \sum_{i=1}^{b} w(e_i^*) \le w(F) + 2w(F^*). \quad \square$$

Let us bound the number of phases that are needed in the worst case.

Lemma 2.4. *After at most $\lfloor p_L/2 \rfloor + 1$ phases, $\tilde{H}$ does not contain any illegal components.*

Algorithm 1 ApxUndir$_{L'}$.

Input: undirected complete graph $G = (V, E)$, $|V| = n$; edge weights $w : E \to \mathbb{N}$ satisfying the triangle inequality
Output: an L'-cycle cover C^{apx} of G if n is L'-admissible, $\perp$ otherwise

if $n \notin \langle L \rangle'$ **then**
  return $\perp$
end if
run GoeWill using the function $f_{L'}$ described in the text to obtain $H = (V, F)$
while the size of some connected components of H is not in $\langle L' \rangle$ **do**
  let $C_1, \dots, C_a$ be the connected components of H, where V_i is the vertex set of C_i; let $C_1, \dots, C_b$ be its illegal components
  let e_i be the lightest edge connecting V_i to $V \setminus V_i$
  add $e_1, \dots, e_b$ to F
  while H contains cycles **do**
    remove one e_i to break a cycle
  end while
end while
duplicate each edge to obtain a multi-graph consisting of Eulerian components
for all components of the multi-graph **do**
  walk along an Eulerian cycle
  take shortcuts to obtain a Hamiltonian cycle
  discard edges to obtain a collection of paths, the number of vertices of each of which is in L'
  connect the two endpoints of every path in order to obtain cycles
end for
the union of all cycles constructed forms C^{apx}; return C^{apx}

Proof. In the beginning, all components of $H = (V, F)$ contain at least g_L vertices. If $g_L \in L'$, no phases are needed at all. Thus, we can assume that $\min(L') \geq 2g_L$.

To bound the number of phases needed, we will estimate the size of the smallest illegal component. Consider any of the smallest illegal components before some phase t, and let s be the number of its vertices. In phase t, this component will be connected either to another illegal component, which results in a component with a size of at least $2s$, or to a legal component, which results in a component with a size of at least $s+2g_L$. (It can happen that more than two illegal components are connected to a single component in one phase.)

In either case, except for the first phase, the size of the smallest illegal component increases by at least $2g_L$ in every step. Thus, after at most $\lfloor p_L/2 \rfloor + 1$ phases, the size of every illegal component is at least $(p_L + 1)g_L$. Hence, there are no more illegal components since components that consist of at least $(p_L + 1)g_L$ vertices are not illegal. □

Eventually, we obtain a forest that consists solely of components whose sizes are in $\langle L' \rangle$. We call this forest $\tilde{H} = (V, \tilde{F})$. Then we proceed as already described above: We duplicate each edge, thus obtaining Eulerian components. After that, we take shortcuts to obtain an $\langle L' \rangle$-cycle cover, which is also a $\langle L \rangle$-cycle cover. Finally, we break edges and connect the endpoints of each path to obtain an L'-cycle cover, which is also an L-cycle cover since $L \supseteq L'$. The weight of this L'-cycle cover is at most $4 \cdot w(\tilde{F})$.

Overall, for the set L', we obtain ApxUndir$_{L'}$ (Algorithm 1) and the following theorem.

Theorem 2.5. *Let $L \subseteq \mathcal{U} \cup \{2\} = \mathcal{D}$ be arbitrary and $L' \subseteq L$ be chosen according to Lemma 2.1. Then* ApxUndir$_{L'}$ *is a factor $(4 \cdot (p_L + 4))$ approximation algorithm for Min-L-UCC. Its running-time is $O(n^2 \log n)$.*

Proof. Let C^* be a minimum-weight $\langle L' \rangle$-cycle cover. The weight of $\tilde{F}$ is bounded from above by

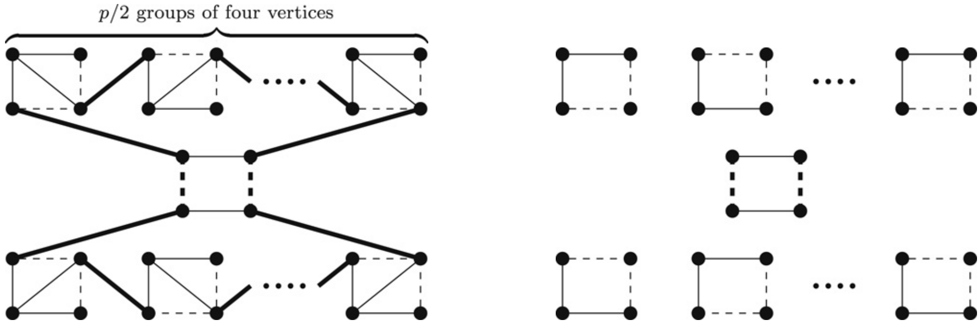
$$w(\tilde{F}) \leq \left(\left\lfloor \frac{p_L}{2} \right\rfloor + 1 \right) \cdot 2 \cdot w(F^*) + 2 \cdot w(F^*) \leq (p_L + 4) \cdot w(C^*).$$

Combining this with $w(C^{\text{apx}}) \leq 4 \cdot w(\tilde{F})$ yields the approximation ratio.

Executing GoeWill takes time $O(n^2 \log n)$. All other operations can be implemented to run in time $O(n^2)$, where the O hides a constant that depends on L'. □

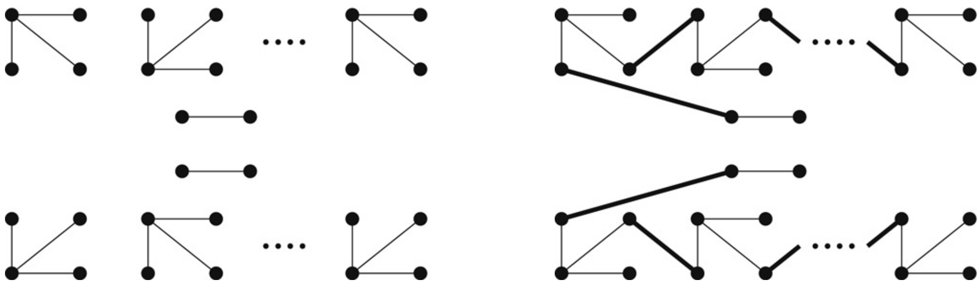
We conclude the analysis of this algorithm by providing an example that shows that the approximation ratio of the algorithm depends indeed linearly on p_L. To do this, let $p \in \mathbb{N}$ be even. We choose $L = \{4, 2p+2, 2p+4, 2p+6, \dots, 4p+4\}$. Thus, $g_L = 2$ and $p_L = p - 1$. Since L is finite, we can choose $L' = L$. Fig. 1 shows the graph that we consider and its optimal L-cycle cover. The graph consists of $4p + 4$ vertices. The weights of the edges, which satisfy the triangle inequality, are as follows:

- Solid bold edges have a weight of 1.
- Dashed bold edges have a weight of $1 + \varepsilon$, where $\varepsilon > 0$ can be made arbitrarily small.

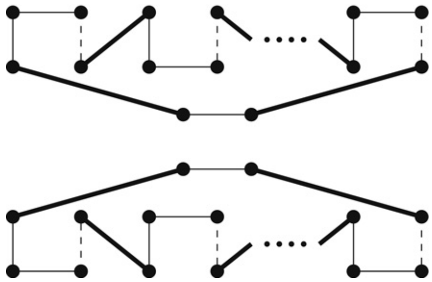


(a) The graph. (b) The optimal L-cycle cover.

Fig. 1. An example on which ApxUndir$_{L'}$ achieves only a ratio of roughly $p_L/2$.



(a) The output of GoeWill. (b) The final forest.



(c) The L-cycle cover C^{apx}.

Fig. 2. How ApxUndir$_{L'}$ computes an L-cycle cover of the graph of Fig. 1(a).

- Solid non-bold edges have a weight of ε.
- Dashed non-bold edges have a weight of 2ε.
- The weight of the edges not drawn is given by the shortest path between the respective vertices.

The weight of the optimum L-cycle cover is $2 + (6p + 4)\varepsilon$: The four central vertices contribute $2 + 4\varepsilon$, and each of the p remaining 4-cycles contributes 6ε. By decreasing ε, the weight of the optimum L-cycle cover can get arbitrarily close to 2.

Fig. 2 shows what ApxUndir$_{L'}$ computes. Let us assume that GoeWill returns the optimum L-forest shown in Fig. 2(a). GoeWill might also return a different forest of the same weight: Instead of creating a component of size four, it can take, e. g., two vertical edges of weights ε and 2ε. However, the resulting L-cycle covers will be equal.

Starting with the output of GoeWill, ApxUndir$_{L'}$ chooses greedily the bold edges, which have a weight of 1, rather than the two edges of weight $1 + \varepsilon$ (Fig. 2(b)). From the forest thus obtained, it constructs an L-cycle cover (Fig. 2(c)). The weight of this L-cycle cover is $2(p/2+1)+(4p+2)\varepsilon$. For sufficiently small ε, this is approximately $p+2 = p_L+3$, which is roughly $p_L/2 + 3/2$ times as large as the weight of the optimum L-cycle cover.

Of course, it would be desirable to have an approximation algorithm with a ratio that does not depend on L. Directly adapting the technique of Goemans and Williamson [15] does not seem to work: The function $f(S) = 1$ if and only if $|S| \notin \langle L\rangle$ is not proper because it violates symmetry. To force it to be symmetric, we can modify it to $f'(S) = 1$ if and only if $|S| \notin \langle L\rangle$ or $|V \setminus S| \notin \langle L\rangle$. But f' does not satisfy disjointness. There are generalizations of Goemans and Williamson's approximation technique to larger classes of functions [16]. However, it seems that L-cycle covers can hardly be modeled even by these more general functions.

2.2. Unconditional inapproximability of Min-L-UCC

In this section, we provide a lower bound for the approximability of Min-L-UCC as a counterpart to the approximation algorithm of the previous section. We show that the problem cannot be approximated with a factor of $2-\varepsilon$. This inapproximability result is unconditional, i. e., it does not rely on complexity theoretic assumptions like $\mathsf{P} \neq \mathsf{NP}$.

The key to the inapproximability of Min-L-UCC are **immune sets** [24]: An infinite set $L \subseteq \mathbb{N}$ is called an immune set if L does not contain an infinite recursively enumerable subset. Such sets exist. Our result limits the possibility of designing general approximation algorithms for L-cycle covers. To obtain algorithms with a ratio better than 2, we have to design algorithms tailored to specific sets L.

Finite variations of immune sets are again immune sets: If a finite number of elements is added to or removed from an immune set, the resulting set is still immune. Thus for every $k \in \mathbb{N}$, there exist immune sets L containing no number smaller than k.

Theorem 2.6. *Let $\varepsilon > 0$ be arbitrarily small. Let $k > 2/\varepsilon$, and let $L \subseteq \{k, k+1, \ldots\}$ be an immune set. Then Min-L-UCC cannot be approximated with a factor of $2-\varepsilon$.*

Proof. Let G_n be an undirected complete graph with vertices $1, 2, \ldots, n$. The weight of an edge $\{i, j\}$ for $i < j$ is $\min\{j-i, n+i-j\}$. This means that the vertices are ordered along an undirected cycle, and the distance from i to j is the number of edges that have to be traversed in order to get from i to j. These edge weights fulfill the triangle inequality.

For all $n \in L$, the optimal L-cycle cover of G_n is a Hamiltonian cycle of weight n. Furthermore, the weight of every cycle c that traverses $\ell \leq n/2$ vertices has a weight of at least $2\ell - 2$: Let i and j be two vertices of c that are farthest apart according to the edge lengths of G_n. Assume that $i < j$. By the triangle inequality, the weight of c is at least $2 \cdot \min\{j-i, n+i-j\}$. Since $\ell \leq n/2$ and by the choice of i and j, we have $\min\{j-i, n+i-j\} \geq \ell - 1$, which proves $w(c) \geq 2\ell - 2$.

Consider any approximation algorithm Approx for Min-L-UCC. We run Approx on G_n for $n \in \mathbb{N}$. By outputting the cycle lengths occurring in the L-cycle cover of G_n for all n, we obtain an enumeration of a subset $S \subseteq L$. Since L is immune, S must be a finite set, and $s = \max(S)$ exists. Let $n \geq 2s$. The L-cycle cover output for G_n consists of cycles whose lengths are at most $s \leq n/2$. Since $\min(L) \geq k$, we also have $\min(S) \geq k$ and the L-cycle cover output for G_n consists of at most n/k cycles. Hence, the weight of the cycle cover computed by Approx is at least $\frac{n}{k} \cdot (2k-2)$. For $n \in L$, this is a factor of $2 - \frac{2}{k} > 2 - \varepsilon$ away from the optimum solution. $\square$

Theorem 2.6 is tight since L-cycle covers can be approximated with a factor of 2 by L'-cycle covers for every set $L' \subseteq L$ with $\langle L' \rangle = \langle L \rangle$ as we will prove now. Let $\min_L(G, w)$ denote the weight of a minimum-weight L-cycle cover of G with edge weights w.

Theorem 2.7. *Let $L \subseteq \mathcal{U}$ be a non-empty set, and let $L' \subseteq L$ with $\langle L' \rangle = \langle L \rangle$. Then we have $\min_{L'}(G, w) \leq 2 \cdot \min_L(G, w)$ for all undirected complete graphs G with edge weights w that satisfy the triangle inequality.*

Proof. Consider an arbitrary L-cycle cover C and any of its cycles c of length $\lambda \in L$. To prove the theorem, we show how to obtain an L'-cycle cover C' from C with $w(C') \leq 2 \cdot w(C)$. Consider any cycle c of C that has a length of λ. If $\lambda \in L'$, we simply put c into C'. Otherwise, since $\langle L' \rangle = \langle L \rangle \supseteq L$, there exist $\lambda_1, \ldots, \lambda_k \in L'$ for some $k \in \mathbb{N}$ such that $\sum_{i=1}^{k} \lambda_i = \lambda$. We remove k edges from c to obtain k paths consisting of $\lambda_1, \ldots, \lambda_k$ vertices. No additional weight is incurred in this way. Then we connect the respective endpoints of each path to obtain k cycles of lengths $\lambda_1, \ldots, \lambda_k$. By the triangle inequality, the weight of an edge added to close a cycle is at most the weight of the corresponding path. By performing this for every cycle of C, we obtain an L'-cycle cover C' as claimed. $\square$

An immediate consequence of Theorem 2.7 is that approximation algorithms for L'-cycle covers for finite L' can be turned into approximation algorithms for arbitrary L by losing only a factor of 2 in the approximation performance.

Corollary 2.8. *Let $L \subseteq \mathcal{U}$ be a non-empty set, and let $L' \subseteq L$ with $\langle L \rangle = \langle L' \rangle$. If Min-$L'$-UCC can be approximated with a factor of r, then Min-L-UCC can be approximated with a factor of $2r$.*

Proof. Let (G, w) be an instance of Min-L-UCC and Min-L'-UCC. Let C' be the L'-cycle cover of G output by the r approximation for Min-L'-UCC. Clearly, C' is also an L-cycle cover. Furthermore, $w(C') \leq r \cdot \min_{L'}(G, w) \leq 2r \cdot \min_L(G, w)$. $\square$

3. Approximability of Min-*L*-DCC

3.1. An approximation algorithm for Min-L-DCC

In this section, we prove the existence of approximation algorithms for Min-L-DCC for all sets $L \subseteq \mathcal{D}$. Again, we provide an algorithm ApxDir$_{L'}$ that contains a particular set $L' \subseteq \mathcal{D}$ hardwired into it. This algorithm will then serve as approximation algorithm for Min-L-DCC for sets $L \supseteq L'$ with $\langle L \rangle = \langle L' \rangle$. The algorithm ApxDir$_{L'}$ exploits ApxUndir$_{L'}$ to achieve an approximation ratio of $O(n)$. The hidden factor depends on $p_{L'}$ again. This result matches asymptotically the lower bound of Section 3.2 and shows that Min-L-DCC can be approximated at least to some extent.

Algorithm 2 $\textsc{ApxDir}_{L'}$.

Input: directed complete graph $G = (V, E)$, $|V| = n$; edge weights $w : E \to \mathbb{N}$ satisfying the triangle inequality
Output: an L'-cycle cover C^{apx} of G if n is L'-admissible, $\perp$ otherwise

```
if n ∉ ⟨L⟩' then
    return ⊥
end if
construct an undirected complete graph G_U = (V, E_U) with edge weights w_U({u, v}) = w(u, v) + w(v, u)
run ApxUndir_{L'} on G_U and w_U to obtain C_U^apx
for all cycles c_U of C_U^apx do
    c_U corresponds to a cycle of G that can be oriented in two ways; put the orientation c that yields less weight into C^apx
end for
return C^apx
```

In order to approximate Min-L-DCC, we reduce the problem to a variant of Min-L-UCC, where also 2-cycles are allowed (now it pays off that we included 2 in the possible cycle lengths in Section 2.1): We obtain a 2-cycle of an undirected graph by taking an edge $\{u, v\}$ twice. Let $G = (V, E)$ be a directed complete graph with n vertices and edge weights $w : E \to \mathbb{N}$ that fulfill the triangle inequality. The corresponding undirected complete graph $G_U = (V, E_U)$ has weights $w_U : E_U \to \mathbb{N}$ with $w_U(\{u, v\}) = w(u, v) + w(v, u)$.

Let C be any cycle cover of G. The corresponding cycle cover C_U of G_U is given by $C_U = \{\{u, v\} \mid (u, v) \in C\}$. Note that we consider C_U as a multiset: If both (u, v) and (v, u) are in C, i. e., u and v form a 2-cycle, then $\{u, v\}$ occurs twice in C_U. Let us bound the weight of C_U in terms of the weight of C.

Lemma 3.1. *For every cycle cover C of G, we have $w_U(C_U) \leq n \cdot w(C)$.*

Proof. Consider any edge $e = (u, v) \in C$, and let c be the cycle of length λ that contains e. By the triangle inequality, we have $w_U(\{u, v\}) = w(u, v) + w(v, u) \leq w(c)$. Let c_U be the cycle of C_U that corresponds to c. Since c consists of λ edges, we obtain $w_U(c_U) \leq \lambda \cdot w(c) \leq n \cdot w(c)$. Summing over all cycles of C completes the proof. □

Our algorithm computes an L'-cycle cover for some finite $L' \subseteq L$ with $\langle L' \rangle = \langle L \rangle$. As in Section 2.1, the weight of the cycle cover computed is compared to an optimum $\langle L \rangle$-cycle.

Let C_U^{apx} be the L'-cycle cover output by $\textsc{ApxUndir}_{L'}$ on G_U. We transfer C_U^{apx} into an L'-cycle cover C^{apx} of G. For every cycle c_U of C_U^{apx}, we can orient the corresponding directed cycle c in two directions. We take the orientation that yields less weight, thus $w(C^{\text{apx}}) \leq w_U(C_U^{\text{apx}})/2$. Overall, we obtain $\textsc{ApxDir}_{L'}$ (Algorithm 2), which achieves an approximation ratio of $O(n)$.

Theorem 3.2. *Let $L \subseteq \mathcal{D}$ be arbitrary, and let $L' \subseteq L$ be chosen according to Lemma 2.1. Then $\textsc{ApxDir}_{L'}$ is a factor $(2n \cdot (p_L + 4))$ approximation algorithm for Min-L-DCC. Its running-time is $O(n^2 \log n)$.*

Proof. We start by estimating the approximation ratio. Theorem 2.5 yields $w_U(C_U^{\text{apx}}) \leq 4 \cdot (p_L + 4) \cdot w_U(C_U^*)$, where C_U^* is an optimal $\langle L \rangle$-cycle cover of G_U. Now consider an optimum $\langle L \rangle$-cycle cover C^* of G. Lemma 3.1 yields $w_U(C_U^*) \leq n \cdot w(C^*)$. Overall,

$$w(C^{\text{apx}}) \leq \frac{1}{2} \cdot w_U(C_U^{\text{apx}}) \leq 2 \cdot (p_L + 4) \cdot w_U(C_U^*) \leq 2 \cdot (p_L + 4) \cdot n \cdot w(C^*).$$

The running-time is dominated by the time needed to execute $\textsc{GoeWill}$ in $\textsc{ApxUndir}_{L'}$, which is $O(n^2 \log n)$. □

3.2. Unconditional inapproximability of Min-L-DCC

For undirected graphs, both Max-L-UCC and Min-L-UCC can be approximated to within constant factors in polynomial time. Surprisingly, in case of directed graphs, this holds only for the maximization variant of the directed L-cycle cover problem. Min-L-DCC cannot be approximated with a factor of $o(n)$ for certain sets L, where n is the number of vertices of the input graph. In particular, $\textsc{ApxDir}_{L'}$ achieves asymptotically optimal approximation ratios for Min-L-DCC. Similar to the case of Min-L-UCC, this result shows that to find approximation algorithms, specific properties of the sets L have to be exploited. A general algorithm with a good approximation ratio for all sets L does not exist.

Theorem 3.3. *Let $L \subseteq \mathcal{U}$ be an immune set. Then no approximation algorithm for Min-L-DCC achieves a ratio of $o(n)$, where n is the number of vertices of the instance.*

Proof. Let G_n be a directed complete graph with n vertices $\{1, 2, \ldots, n\}$. The weight of an edge (i, j) is $(j - i) \bmod n$. This means that the vertices are ordered along a directed cycle, and the distance from i to j is the number of edges that have to be traversed in order to get from i to j. These edge weights fulfill the triangle inequality.

For all $n \in L$, the optimal L-cycle cover of G_n is a Hamiltonian cycle of weight n. Furthermore, the weight of every cycle that traverses some of G_n's vertices has a weight of at least n: Let i and j be two traversed vertices with $i < j$. By the triangle inequality, the path from i to j has a weight of at least $j - i$ while the path from j to i has a weight of at least $i - j + n = (i - j) \bmod n$.

Consider any approximation algorithm Approx for Min-L-DCC. We run Approx on G_n for $n \in \mathbb{N}$. By outputting the cycle lengths occurring in the L-cycle cover of G_n for all $n = 1, 2, \ldots$, we obtain an enumeration of a subset $S \subseteq L$. Since L is immune, S is a finite set, and $s = \max(S)$ exists. Thus, the L-cycle cover output for G_n consists of at least n/s cycles and has a weight of at least n^2/s. For $n \in L$, this is a factor of n/s away from the optimum solution, where s is a constant that depends only on Approx. Thus, no recursive algorithm can achieve an approximation ratio of $o(n)$. $\square$

Assume that we can approximate Min-L'-DCC with a ratio of r for every finite set L'. Theorem 3.4 shows that then Min-L-DCC can be approximated for all L with a ratio of $\varepsilon n r$, where ε can be made arbitrarily small. This is the directed counterpart of Theorem 2.7 and Corollary 2.8, and it shows that Theorem 3.3 is tight.

Theorem 3.4. *For every L and every $\varepsilon > 0$, there exists a finite set $L' \subseteq L$ with $\langle L' \rangle = \langle L \rangle$ such that $\min_{L'}(G, w) \leq \varepsilon n \cdot \min_L(G, w)$ for all directed complete graphs G with edge weights w.*

For the proof of the theorem, we need the following lemma, which we will also use for Theorem 4.1.

Lemma 3.5. *For every $L \subseteq \mathbb{N}$ and every $\varepsilon > 0$, there exists a finite set $L' \subseteq L$ with $\langle L' \rangle = \langle L \rangle$ and the following property: For every $\lambda \in L \setminus L'$, there exist $\lambda_1, \ldots, \lambda_z \in L'$ with $z \leq \varepsilon\lambda$ such that $\sum_{i=1}^{z} \lambda_i = \lambda$.*

Proof. If L is finite, we simply choose $L' = L$. So we assume that L is infinite. Let again g_L denote the greatest common divisor of all numbers of L. Let us first describe how to proceed if $g_L \in L$. After that we deal with the case that $g_L \notin L$.

Let $L' = \{\lambda \in L \mid \lambda \leq m\}$, and let $\ell \in L'$. If m is sufficiently large, then $\langle L' \rangle = \langle L \rangle$ (this follows from the proof of Lemma 2.1 [23, Lem. 3.1] and also implicitly from this proof). We will specify ℓ and m, which depend on ε, later on.

Let $\lambda \in L \setminus L'$. Thus, $\lambda > m$. Let $r = \bmod(\lambda, \ell)$. Since λ and ℓ are divisible by g_L, also r is divisible by g_L. Since $\lambda \notin L'$, we have to find $\lambda_1, \lambda_2, \ldots \in L'$ that add up to λ. We have $\lambda = \lfloor \lambda/\ell \rfloor \cdot \ell + (r/g_L) \cdot g_L$. Now we choose $\lambda_1 = \cdots = \lambda_{\lfloor \lambda/\ell \rfloor} = \ell$ and $\lambda_{\lfloor \lambda/\ell \rfloor + 1} = \cdots = \lambda_{\lfloor \lambda/\ell \rfloor + r/g_L} = g_L$. What remains is to show that $\lfloor \lambda/\ell \rfloor + r/g_L \leq \varepsilon\lambda$. To do this, we choose $\ell > 1/\varepsilon$. Since r/g_L is bounded from above by ℓ/g_L, which does not depend on λ, we obtain $\lfloor \lambda/\ell \rfloor + r/g_L \leq \varepsilon\lambda$ for all $\lambda > m$ for some sufficiently large m.

The case that $g_L \notin L$ remains to be considered. There exist $\pi_1, \ldots, \pi_p \in L$ and $\xi_1, \ldots, \xi_p \in \mathbb{Z}$ for some $p \in \mathbb{N}$ with $g_L = \sum_{i=1}^{p} \xi_i \pi_i$. Without loss of generality, we assume that $\xi_1 = \min_{1 \leq i \leq p} \xi_i$. We have $\xi_1 < 0$ since $g_L \notin L$.

As above, let $L' = \{\lambda \in L \mid \lambda \leq m\}$, and let $\ell \in L'$. Let $\ell^* = -\xi_1 \ell \cdot \sum_{i=1}^{p} \pi_i > 0$. We choose m to be larger than ℓ^*. Let $\lambda > m$, and let $r = \bmod(\lambda - \ell^*, \ell)$. Then

$$\lambda = \left\lfloor \frac{\lambda - \ell^*}{\ell} \right\rfloor \cdot \ell + r + \ell^* = \left\lfloor \frac{\lambda - \ell^*}{\ell} \right\rfloor \cdot \ell + \frac{r}{g_L} \cdot \sum_{i=1}^{p} \pi_i \xi_i - \xi_1 \ell \cdot \sum_{i=1}^{p} \pi_i$$
$$= \left\lfloor \frac{\lambda - \ell^*}{\ell} \right\rfloor \cdot \ell + \sum_{i=1}^{p} \pi_i \cdot \left(\frac{r\xi_i}{g_L} - \xi_1 \ell \right).$$

We have $\rho_i = \frac{r\xi_i}{g_L} - \xi_1 \ell \geq 0$: Since $\xi_1 < 0$, we have $-\xi_1 \ell > 0$. If $\xi_i > 0$, then of course $\rho_i \geq 0$. If $\xi_i < 0$, then $-\xi_i \leq -\xi_1$, and $\rho_i \geq 0$ follows from $r < \ell$. According to the deliberations above, we choose $\lambda_1 = \cdots = \lambda_{\lfloor (\lambda - \ell^*)/\ell \rfloor} = \ell$. In addition, for $1 \leq i \leq p$, we set ρ_i of the λ_j's equal to π_i. It remains to be shown that $\lfloor (\lambda - \ell^*)/\ell \rfloor + \sum_{i=1}^{p} \rho_i \leq \varepsilon\lambda$. This follows from the fact that $\rho_i \leq \ell \cdot (\xi_i/g_L - \xi_1)$ for all i, which is independent of λ. Again, we choose $\ell > 1/\varepsilon$ and m sufficiently large to complete the proof. $\square$

Proof of Theorem 3.4. Let $\varepsilon > 0$ and $L \subseteq \mathcal{D}$ be given. We choose $L' \subseteq L$ as described in the proof of Lemma 3.5. In order to prove the theorem, let G be a directed complete graph, and let C be an L-cycle cover of minimum weight of G. We show that we can find an L'-cycle cover C' with $w(C') \leq \varepsilon n \cdot w(C)$.

The L'-cycle cover C' contains all cycles of C whose lengths are in L'. Now consider any cycle c of length $\lambda \in L \setminus L'$. According to Lemma 3.5, there exist $\lambda_1, \ldots, \lambda_z \in L'$ with $\sum_{i=1}^{z} \lambda_i = \lambda$ and $z \leq \varepsilon\lambda$. We decompose c into z cycles of length $\lambda_1, \ldots, \lambda_z$. By the triangle inequality, the weight of each of these new cycles is at most $w(c)$. Thus, the total weight of all z cycles is at most $z \cdot w(c) \leq \varepsilon\lambda \cdot w(c) \leq \varepsilon n \cdot w(c)$. By performing this for all cycles of C, we obtain an L'-cycle cover C' with $\min_{L'}(G, w) \leq w(C') \leq \varepsilon n \cdot w(C) = \varepsilon n \cdot \min_L(G, w)$. $\square$

4. Properties of maximum-weight cycle covers

To contrast our results for Min-L-UCC and Min-L-DCC, we show that their maximization counterparts Max-L-UCC and Max-L-DCC can, at least in principle, be approximated arbitrarily well; their inapproximability is solely due to their APX-hardness and not to the difficulties arising from undecidable sets L. In other words, the lower bounds for Min-L-UCC and

Min-L-DCC presented in this paper are based on the hardness of deciding if certain lengths are in L. The inapproximability of Max-L-UCC and Max-L-DCC is based on the difficulty of finding good L-cycle covers rather than testing if they are L-cycle covers.

Let $\max_L(G, w)$ be the weight of a maximum-weight L-cycle cover of G with edge weights w. The edge weights w do not have to fulfill the triangle inequality. We will show that $\max_L(G, w)$ can be approximated arbitrarily well by $\max_{L'}(G, w)$ for finite sets $L' \subseteq L$ with $\langle L' \rangle = \langle L \rangle$. Thus, any approximation algorithm for Max-L'-UCC or Max-L'-DCC for finite sets L' immediately yields an approximation algorithm for general sets L with an only negligibly worse approximation ratio. The following theorem for directed cycle covers contains the case of undirected graphs as a special case.

Theorem 4.1. *Let $L \subseteq \mathcal{D}$ be any non-empty set, and let $\varepsilon > 0$. Then there exists a finite subset $L' \subseteq L$ with $\langle L' \rangle = \langle L \rangle$ such that $\max_{L'}(G, w) \geq (1 - \varepsilon) \cdot \max_L(G, w)$ for all directed complete graphs G with edge weights w.*

Proof. Let $\varepsilon > 0$ be given. Depending on L and ε, we choose L' according to Lemma 3.5. Let us compare $\max_{L'}(G, w)$ to $\max_L(G, w)$. Therefore, let C be an optimum L-cycle cover. We show how to obtain an L'-cycle cover C' from C. The L'-cycle cover C' contains all cycles of C whose lengths are in L'. Let us consider any cycle c of length $\lambda \in L \setminus L'$. There exist $\lambda_1, \ldots, \lambda_z \in L'$ for some $z \leq \varepsilon\lambda$ that sum up to λ. We break z edges of c to obtain a collection of paths of lengths $\lambda_1 - 1, \ldots, \lambda_z - 1$. By doing this, we remove at most an ε fraction of c's weight: Let $e_1, \ldots, e_\lambda$ be the edges of c in that order, where e_1 is chosen uniformly at random from c's edges. Then we break $e_{\lambda_1}, e_{\lambda_1+\lambda_2}, \ldots, e_{\lambda_1+\cdots+\lambda_z}$. In this way, we obtain a collection of paths consisting of $\lambda_1 - 1, \lambda_2 - 1, \ldots, \lambda_z - 1$ edges, each of which can be closed to form a cycle whose length is in L'. By the random choice of e_1 and since $z \leq \varepsilon\lambda$ edges are broken, every edge is removed with a probability of at most ε. Thus, the expected total weight of the paths is at least $(1 - \varepsilon) \cdot w(c)$. Hence, we can choose e_1 deterministically such that at most an ε fraction of the weight is removed.

We have lost at most $\varepsilon \cdot w(c)$ of the weight of every cycle c of C, thus $\max_{L'}(G, w) \geq w(C') \geq (1 - \varepsilon) \cdot w(C) = (1 - \varepsilon) \cdot \max_L(G, w)$. □

5. Concluding remarks

First of all, we would like to know if there is a general upper bound for the approximability of Min-L-UCC : Does there exists an r (independent of L) such that Min-L-UCC can be approximated with a factor of r? If such an algorithm works also for the slightly more general problem Min-L-UCC with $2 \in L$ (see Section 3.1), then we would obtain a factor $rn/2$ approximation for Min-L-DCC as well.

While the problem of computing L-cycle cover of minimum weight can be approximated efficiently in the case of undirected graphs, the directed variant seems to be much harder. We are interested in developing approximation algorithms for Min-L-DCC for particular sets L or for certain classes of sets L. For instance, how well can Min-L-DCC be approximated if L is a finite set? Are there non-constant lower bounds for the approximability of Min-L-DCC, for instance bounds depending on $\max(L)$? Because of the similarities between Min-L-DCC and ATSP, an answer to either questions would hopefully also shed some light on the approximability of the ATSP.

Acknowledgements

The author's work done in part at the Department of Computer Science at Yale University (P. O. Box 208285, New Haven, CT 06520-8285, USA) supported by the Postdoc-Program of the German Academic Exchange Service (DAAD).

References

[1] Ravindra K. Ahuja, Thomas L. Magnanti, James B. Orlin, Network Flows: Theory, Algorithms, and Applications, Prentice-Hall, 1993.
[2] Giorgio Ausiello, Pierluigi Crescenzi, Giorgio Gambosi, Viggo Kann, Alberto Marchetti-Spaccamela, Marco Protasi, Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties, Springer, 1999.
[3] Markus Bläser, A 3/4-approximation algorithm for maximum ATSP with weights zero and one, in: Klaus Jansen, Sanjeev Khanna, José D.P. Rolim, Dana Ron (Eds.), Proc. of the 7th Int. Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX, in: Lecture Notes in Computer Science, vol. 3122, Springer, 2004, pp. 61–71.
[4] Markus Bläser, Bodo Manthey, Approximating maximum weight cycle covers in directed graphs with weights zero and one, Algorithmica 42 (2) (2005) 121–139.
[5] Markus Bläser, Bodo Manthey, Jiří Sgall, An improved approximation algorithm for the asymmetric TSP with strengthened triangle inequality, Journal of Discrete Algorithms 4 (4) (2006) 623–632.
[6] Markus Bläser, L. Shankar Ram, Maxim I. Sviridenko, Proc. of the 9th Workshop on Algorithms and Data Structures, WADS, in: Frank Dehne, Alejandro López-Ortiz, Jörg-Rüdiger Sack (Eds.), in: Lecture Notes in Computer Science, vol. 3608, Springer, 2005, pp. 350–359.
[7] Markus Bläser, Bodo Siebert, Computing cycle covers without short cycles, in: Friedhelm Meyer auf der Heide (Ed.), Proc. of the 9th Ann. European Symp. on Algorithms, ESA, in: Lecture Notes in Computer Science, vol. 2161, Springer, 2001, pp. 368–379. Bodo Siebert is the birth name of Bodo Manthey.
[8] Avrim L. Blum, Tao Jiang, Ming Li, John Tromp, Mihalis Yannakakis, Linear approximation of shortest superstrings, Journal of the ACM 41 (4) (1994) 630–647.
[9] Hans-Joachim Böckenhauer, Juraj Hromkovič, Ralf Klasing, Sebastian Seibert, Walter Unger, Approximation algorithms for the TSP with sharpened triangle inequality, Information Processing Letters 75 (3) (2000) 133–138.
[10] L. Sunil Chandran, L. Shankar Ram, On the relationship between ATSP and the cycle cover problem, Theoretical Computer Science 370 (1–3) (2007) 218–228.

[11] Zhi-Zhong Chen, Takayuki Nagoya, Improved approximation algorithms for metric MaxTSP, Journal of Combinatorial Optimization 13 (4) (2007) 321–336.
[12] Zhi-Zhong Chen, Yuusuke Okamoto, Lusheng Wang, Improved deterministic approximation algorithms for Max TSP, Information Processing Letters 95 (2) (2005) 333–342.
[13] Michael R. Fellows, Michael A. Langston, Nonconstructive tools for proving polynomial-time decidability, Journal of the ACM 35 (3) (1988) 727–739.
[14] Paul C. Gilmore, Eugene L. Lawler, David B. Shmoys, Well-solved special cases, in: Eugene L. Lawler, Jan Karel Lenstra, Alexander H.G. Rinnooy Kan, David B. Shmoys (Eds.), The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization, John Wiley & Sons, 1985, pp. 87–143.
[15] Michel X. Goemans, David P. Williamson, A general approximation technique for constrained forest problems, SIAM Journal on Computing 24 (2) (1995) 296–317.
[16] Michel X. Goemans, David P. Williamson, The primal-dual method for approximation algorithms and its application to network design problems, in: Dorit S. Hochbaum (Ed.), Approximation Algorithms for NP-hard Problems, PWS Publishing Company, 1997, pp. 144–191.
[17] David Hartvigsen, An extension of matching theory, Ph.D. Thesis, Department of Mathematics, Carnegie Mellon University, Pittsburgh, PA, USA, September 1984.
[18] Refael Hassin, Shlomi Rubinstein, On the complexity of the k-customer vehicle routing problem, Operations Research Letters 33 (1) (2005) 71–76.
[19] Refael Hassin, Shlomi Rubinstein, Erratum to "An approximation algorithm for maximum triangle packing" [Discrete Applied Mathematics 154 (2006) 971–979], Discrete Applied Mathematics 154 (18) (2006) 2620.
[20] Pavol Hell, David G. Kirkpatrick, Jan Kratochvíl, Igor Kríz, On restricted two-factors, SIAM Journal on Discrete Mathematics 1 (4) (1988) 472–484.
[21] Haim Kaplan, Moshe Lewenstein, Nira Shafrir, Maxim I. Sviridenko, Approximation algorithms for asymmetric TSP by decomposing directed regular multigraphs, Journal of the ACM 52 (4) (2005) 602–626.
[22] László Lovász, Michael D. Plummer, Matching Theory, in: North-Holland Mathematics Studies, vol. 121, Elsevier, 1986.
[23] Bodo Manthey, On approximating restricted cycle covers, SIAM Journal on Computing 38 (1) (2008) 181–206.
[24] Piergiorgio Odifreddi, Classical Recursion Theory, in: Studies in Logic and The Foundations of Mathematics, vol. 125, Elsevier, 1989.
[25] Jorge L. Ramírez Alfonsín, The Diophantine Frobenius Problem, in: Oxford Lecture Series in Mathematics and Its Applications, vol. 30, Oxford University Press, 2005.
[26] Sartaj Sahni, Teofilo Gonzalez, P-complete approximation problems, Journal of the ACM 23 (3) (1976) 555–565.
[27] Alexander Schrijver, Combinatorial Optimization: Polyhedra and Efficiency, in: Algorithms and Combinatorics, vol. 24, Springer, 2003.
[28] Z. Sweedyk, A $2\frac{1}{2}$-approximation algorithm for shortest superstring, SIAM Journal on Computing 29 (3) (1999) 954–986.
[29] Oliver Vornberger, Easy and hard cycle covers, Technical Report, Universität/Gesamthochschule Paderborn, 1980.