Approximating Multi-Criteria Max-TSP

Markus Bläser, Bodo Manthey, and Oliver Putz

Saarland University, Computer Science
Postfach 151150, 66041 Saarbrücken, Germany
blaeser/manthey@cs.uni-sb.de, oli.putz@gmx.de

Abstract. We present randomized approximation algorithms for multi-criteria Max-TSP. For Max-STSP with $k > 1$ objective functions, we obtain an approximation ratio of $\frac{1}{k} - \varepsilon$ for arbitrarily small $\varepsilon > 0$. For Max-ATSP with k objective functions, we obtain a ratio of $\frac{1}{k+1} - \varepsilon$.

1 Multi-Criteria Traveling Salesman Problem

1.1 Traveling Salesman Problem

The traveling salesman problem (TSP) is one of the most fundamental problems in combinatorial optimization. Given a graph, the goal is to find a Hamiltonian cycle of minimum or maximum weight. We consider finding Hamiltonian cycles of maximum weight (Max-TSP).

An instance of Max-TSP is a complete graph $G = (V, E)$ with edge weights $w : E \to \mathbb{N}$. The goal is to find a Hamiltonian cycle of maximum weight. The weight of a Hamiltonian cycle (or, more general, of a subset of E) is the sum of the weights of its edges. If G is undirected, we speak of Max-STSP (symmetric TSP). If G is directed, we have Max-ATSP (asymmetric TSP).

Both Max-STSP and Max-ATSP are NP-hard and APX-hard. Thus, we are in need of approximation algorithms. The currently best approximation algorithms for Max-STSP and Max-ATSP achieve approximation ratios of 61/81 and 2/3, respectively [3, 5].

Cycle covers are an important tool for designing approximation algorithms for the TSP. A cycle cover of a graph is a set of vertex-disjoint cycles such that every vertex is part of exactly one cycle. Hamiltonian cycles are special cases of cycle covers that consist of just one cycle. Thus, the weight of a maximum-weight cycle cover is an upper bound for the weight of a maximum-weight Hamiltonian cycle. In contrast to Hamiltonian cycles, cycle covers of maximum weight can be computed efficiently using matching algorithms [1].

1.2 Multi-Criteria Optimization

In many optimization problems, there is more than one objective function. Consider buying a car: We might want to buy a cheap, fast car with a good gas mileage. How do we decide which car suits us best? With multiple criteria involved, there is no natural notion of a best choice. Instead, we have to be content

Proc. 16th European Symposium on Algorithms (ESA 2008) – © Springer-Verlag

with a trade-off. The aim of multi-criteria optimization is to cope with this problem. To transfer the concept of an optimal solution to multi-criteria optimization problems, the notion of *Pareto curves* was introduced (cf. Ehrgott [4]). A Pareto curve is a set of solutions that can be considered optimal.

More formally, a k-criteria optimization problem consists of instances I, solutions $\mathrm{sol}(X)$ for every instance $X \in I$, and k objective functions $w_1, \ldots, w_k$ that map $X \in I$ and $Y \in \mathrm{sol}(X)$ to $\mathbb{N}$. Throughout this paper, our aim is to maximize the objective functions. We say that a solution $Y \in \mathrm{sol}(X)$ *dominates* another solution $Z \in \mathrm{sol}(X)$ if $w_i(Y, X) \geq w_i(Z, X)$ for all $i \in [k] = \{1, \ldots, k\}$ and $w_i(Y, X) > w_i(Z, X)$ for at least one i. This means that Y is strictly preferable to Z. A *Pareto curve* contains all solutions that are not dominated by another solution. Unfortunately, Pareto curves cannot be computed efficiently in many cases: They are often of exponential size, and they are often NP-hard to compute even for otherwise easy problems. Thus, we have to be content with approximate Pareto curves.

For simpler notation, let $w(Y, X) = (w_1(Y, X), \ldots, w_k(Y, X))$. We will omit the instance X if it is clear from the context. Inequalities are meant componentwise. A set $\mathcal{P} \subseteq \mathrm{sol}(X)$ of solutions is called an α *approximate Pareto curve* for $X \in I$ if the following holds: For every solution $Z \in \mathrm{sol}(X)$, there exists a $Y \in \mathcal{P}$ with $w(Y) \geq \alpha w(Z)$. We have $\alpha \leq 1$, and a 1 approximate Pareto curve is a Pareto curve. (This is not precisely true if there are several solutions whose objective values agree. However, in our case this is inconsequential, and we will not elaborate on this for the sake of clarity.) An algorithm is called an α *approximation algorithm* if, given the instance X, it computes an α approximate Pareto curve. It is called a randomized α approximation algorithm if its success probability is at least $1/2$. This success probability can be amplified to $1 - 2^{-m}$ by executing the algorithm m times and taking the union of all sets of solutions. (We can remove dominated solutions from this union, but this is not required by the definition of an approximate Pareto curve.)

Papadimitriou and Yannakakis [8] showed that $(1 - \varepsilon)$ approximate Pareto curves of size polynomial in the instance size and $1/\varepsilon$ exist. The technical requirement for the existence is that the objective values of solutions in $\mathrm{sol}(X)$ are bounded from above by $2^{p(N)}$ for some polynomial p, where N is the size of X. This is fulfilled in most optimization problems and in particular in our case.

A *fully polynomial time approximation scheme* (FPTAS) for a multi-criteria optimization problem computes $(1-\varepsilon)$ approximate Pareto curves in time polynomial in the size of the instance and $1/\varepsilon$ for all $\varepsilon > 0$. Papadimitriou and Yannakakis [8] showed that multi-criteria minimum-weight matching admits a *randomized FPTAS*, i. e., the algorithm succeeds in computing a $(1 - \varepsilon)$ approximate Pareto curve with constant probability. This yields also a randomized FPTAS for the multi-criteria maximum-weight cycle cover problem [7], which we will use in the following.

Manthey and Ram [6, 7] designed randomized approximation algorithms for several variants of multi-criteria Min-TSP. However, they leave it as an open problem to design any approximation algorithm for Max-TSP.

1.3 New Results

We devise the first approximation algorithm for multi-criteria Max-TSP. For k-criteria Max-STSP, we achieve an approximation ratio of $\frac{1}{k} - \varepsilon$ for arbitrarily small $\varepsilon > 0$. For k-criteria Max-ATSP, we achieve $\frac{1}{k+1} - \varepsilon$. Our algorithm is randomized. Its running-time is polynomial in the input size and $1/\varepsilon$ and exponential in the number k of criteria. However, the number of different objective functions is usually a small constant. The main ingredient for our algorithm is a decomposition technique for cycle covers and a reduction from k-criteria instances to $(k-1)$-criteria instances.

Due to lack of space, some proofs are omitted. For complete proofs, we refer to the full version of this paper [2].

2 Outline and Idea

A straight-forward 1/2 approximation for mono-criterion Max-ATSP is the following: First, we compute a maximum-weight cycle cover C. Then we remove the lightest edge of each cycle, thus losing at most half of C's weight. In this way, we obtain a collection of paths. Finally, we add edges to connect the paths to get a Hamiltonian cycle. For Max-STSP, the same approach yields a 2/3 approximation since the length of every cycle is at least three.

Unfortunately, this does not generalize to multi-criteria Max-TSP for which "lightest edge" is usually not well defined: If we break an edge that has little weight with respect to one objective, we might lose a lot of weight with respect to another objective. Based on this observation, the basic idea behind our algorithm and its analysis is the following case distinction:

Light-weight edges: If all edges of our cycle cover contribute only little to its weight, then removing one edge does not decrease the overall weight by too much. Now we choose the edges to be removed such that no objective loses too much of its weight.

Heavy-weight edges: If there is one edge that is very heavy with respect to at least one objective, then we take only this edge from the cycle cover. In this way, we have enough weight for one objective, and we proceed recursively on the remaining graph with $k-1$ objectives.

In this way, the approximation ratio for k-criteria Max-TSP depends on two questions: First, how well can we decompose a cycle cover consisting solely of light-weight edges? Second, how well can $(k-1)$-criteria Max-TSP be approximated? We deal with the first question in Section 3. In Section 4, we present and analyze our approximation algorithms, which also gives an answer to the second question. Finally, we give evidence that the analysis of the approximation ratios is tight and point out some ideas that might lead to better approximation ratios (Section 5).

3 Decompositions

Let $\alpha \in (0,1]$, and let C be a cycle cover. We call a collection $P \subseteq C$ of paths an *α-decomposition of C* if $w(P) \geq \alpha w(C)$ (recall that all inequalities are meant component-wise). In the following, our aim is to find α-decompositions of cycle covers consisting solely of light-weight edges, that is, $w(e) \leq \alpha w(C)$ for all $e \in C$.

Of course, not every cycle cover possesses an α-decomposition for every α. For instance, a single directed cycle of length two, where each edge has a weight of 1 shows that $\alpha = 1/2$ is best possible for a single objective function in directed graphs. On the other hand, by removing the lightest edge of every cycle, we obtain a 1/2-decomposition.

For undirected graphs and $k = 1$, $\alpha = 2/3$ is optimal: We can find a 2/3-decomposition by removing the lightest edge of every cycle, and a single cycle of length three, where each edge weight is 1, shows that this is tight.

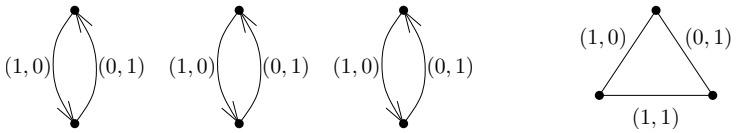
More general, we define $\alpha_k^d \in (0,1]$ to be the maximum number such that every directed cycle cover C with $w(e) \leq \alpha_k^d \cdot w(C)$ for all $e \in C$ possesses an α_k^d-decomposition. Analogously, $\alpha_k^u \in (0,1]$ is the maximum number such that every undirected cycle cover C with $w(e) \leq \alpha_k^u \cdot w(C)$ possesses an α_k^u-decomposition. We have $\alpha_1^d = \frac{1}{2}$ and $\alpha_1^u = \frac{2}{3}$, as we have already argued above. We also have $\alpha_k^u \geq \alpha_k^d$ and $\alpha_k^u \leq \alpha_{k-1}^u$ as well as $\alpha_k^d \leq \alpha_{k-1}^d$.

3.1 Existence of Decompositions

In this section, we investigate for which values of α such α-decompositions exist. In the subsequent section, we show how to actually find good decompositions. We have already dealt with α_1^u and α_1^d. Thus, $k \geq 2$ remains to be considered in the following theorems. In particular, only $k \geq 2$ is needed for the analysis of our algorithms.

Let us first normalize our cycle covers to make the proofs in the following a bit easier. For directed cycle covers C, we can restrict ourselves to cycles of length two: If we have a cycle c of length ℓ with edges $e_1, \dots, e_\ell$, we replace it by $\lfloor \ell/2 \rfloor$ cycles (e_{2j-1}, e_{2j}) for $j = 1, \dots, \lfloor \ell/2 \rfloor$. If ℓ is odd, then we add a edge $e_{\ell+1}$ with $w(e_{\ell+1}) = 0$ and add the cycle $(e_\ell, e_{\ell+1})$. (Strictly speaking, edges are 2-tuples of vertices, and we cannot simply reconnect them. What we mean is that we remove the edges of the cycle and create new edges with the same names and weights together with appropriate new vertices.) We do this for all cycles of length at least three and call the resulting cycle cover C'. Now any α-decomposition P' of the new cycle cover C' yields an α-decomposition P of the original cycle cover C by removing the newly added edges $e_{\ell+1}$: In C, we have to remove at least one edge of the cycle c to obtain a decomposition. In C', we have to remove at least $\lfloor \ell/2 \rfloor$ edges of c, thus at least one. Furthermore, if $w(e) \leq \alpha \cdot w(C)$ for every $e \in C$, then also $w(e) \leq \alpha \cdot w(C')$ for every $e \in C'$ since we kept all edge weights. This also shows $w(P) = w(P')$.

We are interested in α-decompositions that work for all cycle covers with k objective functions. Thus in particular, we have to be able to decompose C'.



(a) $\alpha_2^d \leq 1/3$. (b) $\alpha_2^u \leq 1/2$.

Fig. 1. Examples that limit the possibility of decomposition.

The consequence is that if every directed cycle cover consisting solely of cycles of length two possesses an α-decomposition, then all directed cycle covers do so.

For undirected cycle covers, we can restrict ourselves to cycles of length three: We replace a cycle $c = (e_1, \ldots, e_\ell)$ by $\lfloor \ell/3 \rfloor$ cycles $(e_{3j-2}, e_{3j-1}, e_{3j})$ for $1 \leq j \leq \lfloor \ell/3 \rfloor$. If ℓ is not divisible by three, then we add one or two edges $e_{\ell+1}, e_{\ell+2}$ to form a cycle of length three with the remaining edge(s). Again, every α-decomposition of the new cycle cover yields an α-decomposition of the original cycle cover.

The following two theorems are proved using the probabilistic method.

Theorem 1. *For all $k \geq 2$, we have $\alpha_k^u \geq 1/k$.*

For undirected graphs and $k = 2$, we do not need the assumption that the weight of each edge is at most α_2^u times the weight of the cycle cover. Lemma 1 below immediately yields a $(1/2 - \varepsilon)$ approximation for bi-criteria Max-STSP: First, we compute a Pareto curve of cycle covers. Second, we decompose each cycle cover to obtain a collection of paths, which we then connect to form Hamiltonian cycles. The following lemma can also be generalized to arbitrary k that do not contain cycles of length at most k.

Lemma 1. *For every undirected cycle cover C with edge weights $w = (w_1, w_2)$, there exists a collection $P \subseteq C$ of paths with $w(P) \geq w(C)/2$.*

For directed cycle covers, our aim is again to show that the probability of having not enough weight in one component is less than $1/k$. Hoeffding's inequality works only for $k \geq 7$. We use a different approach, which immediately gives us the desired result for $k \geq 6$, and which can be tweaked to work also for small k.

Theorem 2. *For all $k \geq 2$, we have $\alpha_k^d \geq 1/(k+1)$.*

Figure 1 shows that Theorems 1 and 2, respectively, are tight for $k = 2$. Due to these limitations for $k = 2$, proving larger values for α_k^u or α_k^d does not immediately yield better approximation ratios (see Section 5). However, for larger values of k, Hoeffding's inequality yields the existence of $\Omega(1/\log k)$-decompositions. Together with a different technique for heavy-weight cycle covers, this might lead to improved approximation algorithms for larger values of k.

Lemma 2. *We have $\alpha_k^u, \alpha_k^d \in \Omega(1/\log k)$.*

$P \leftarrow$ DECOMPOSE(C, w, k, α)
input: cycle cover C, edge weights w, $k \geq 2$, $w(e) \leq \alpha \cdot w(C)$ for all $e \in C$
output: a collection P of paths
1: obtain w' from w by scaling each component such that $w'_i(C) = 1/\alpha$ for all i
2: normalize C to C' as described in the text such that C' consists solely of cycles of length three (undirected) or two (directed)
3: **while** there are cycles c and c' in C' with $w'(c) \leq 1/2$ and $w'(c') \leq 1/2$ **do**
4: combine c and c' to $\tilde{c}$ with $w'(\tilde{c}) = w'(c) + w'(c')$
5: replace c and c' by $\tilde{c}$ in C'
6: try all possible combinations of decompositions
7: choose one P' that maximizes $\min_{i \in [k]} w'_i(P)$
8: translate $P' \subseteq C'$ back to obtain a decomposition $P \subseteq C$
9: return P

Algorithm 1: A deterministic algorithm for finding a decomposition.

3.2 Finding Decompositions

While we know that decompositions exist due to the previous section, we have to find them efficiently in order to use them in our approximation algorithm. We present a deterministic algorithm and a faster randomized algorithm for finding decompositions.

DECOMPOSE (Algorithm 1) is a deterministic algorithm for finding a decomposition. The idea behind this algorithm is as follows: First, we scale the weights such that $w(C) = 1/\alpha$. Then $w(e) \leq 1$ for all edges $e \in C$. Second, we normalize all cycle covers such that they consist solely of cycles of length two (in case of directed graphs) or three (in case of undirected graphs). Third, we combine very light cycles as long as possible. More precisely, if there are two cycles c and c' such that $w'(c) \leq 1/2$ and $w'(c') \leq 1/2$, we combine them to one cycle $\tilde{c}$ with $w'(\tilde{c}) \leq 1$. The requirements for an α-decomposition to exist are still fulfilled. Furthermore, any α-decomposition of C' immediately yields an α-decomposition of C.

Lemma 3. *Let $k \geq 2$. Let C be an undirected cycle cover and $w_1, \ldots, w_k$ be edge weights such that $w(e) \leq \alpha_k^u \cdot w(C)$. Then* DECOMPOSE$(C, w, k, \alpha_k^u)$ *returns a collection P of paths with $w(P) \geq \alpha_k^u \cdot w(C)$.*

Let C be a directed cycle cover and $w_1, \ldots, w_k$ be edge weights such that $w(e) \leq \alpha_k^d \cdot w(C)$. Then DECOMPOSE$(C, w, k, \alpha_k^d)$ *returns a collection P of paths with $w(P) \geq \alpha_k^d \cdot w(C)$.*

Let us also estimate the running-time of DECOMPOSE. The normalization in lines 1 to 5 can be implemented to run in linear time. Due to the normalization, the weight of every cycle is at least $1/2$ with respect to at least one w'_i. Thus, we have at most $2k/\alpha_k^u$ cycles in C' in the undirected case and at most $2k/\alpha_k^d$ cycles in C' in the directed case. In either case, we have $O(k^2)$ cycles. All of these cycles are of length two or of length three. Thus, we find an optimal decomposition, which in particular is an α_k^u or α_k^d-decomposition, in time linear in the input size and exponential in k.

$P \leftarrow$ RandDecompose(C, w, k, α)
input: cycle cover C, edge weights $w = (w_1, \ldots, w_k)$, $k \geq 2$, $w(e) \leq \alpha \cdot w(C)$ for all $e \in C$
output: a collection P of paths with $w(P) \geq \alpha \cdot w(C)$
1: **if** $k \geq 6$ **then**
2: **repeat**
3: randomly choose one edge of every cycle of C
4: remove the chosen edges to obtain P
5: **until** $w(P) \geq \alpha \cdot w(C)$
6: **else**
7: $P \leftarrow$ Decompose(C, w, k, α)

Algorithm 2: A randomized algorithm for finding a decomposition.

By exploiting the probabilistic argument of the previous section, we can find a decomposition much faster with a randomized algorithm. RandDecompose (Algorithm 2) does this: We choose the edges to be deleted uniformly at random for every cycle. The probability that we obtain a decomposition as required is positive and bounded from below by a constant. Furthermore, as the proofs of Theorems 1 and 2 show, this probability tends to one as k increases. For $k \geq 6$, it is at least approximately 0.7 for undirected cycle covers and at least $1/4$ for directed cycle covers. For $k < 6$, we just use our deterministic algorithm, which has linear running-time for constant k. The following lemma follows from the considerations above.

Lemma 4. *Let $k \geq 2$. Let C be an undirected cycle cover and $w_1, \ldots, w_k$ be edge weights such that $w(e) \leq \alpha_k^u \cdot w(C)$. Then* RandDecompose$(C, w, k, \alpha_k^u)$ *returns a collection P of paths with $w(P) \geq \alpha_k^u \cdot w(C)$.*

Let C be a directed cycle cover and $w_1, \ldots, w_k$ be edge weights such that $w(e) \leq \alpha_k^d \cdot w(C)$. Then RandDecompose$(C, w, k, \alpha_k^d)$ *returns a collection P of paths with $w(P) \geq \alpha_k^d \cdot w(C)$.*

The expected running-time of RandDecompose *is $O(|C|)$.*

4 Approximation Algorithms

Based on the idea sketched in Section 2, we can now present our approximation algorithms for multi-criteria Max-ATSP and Max-STSP. However, in particular for Max-STSP, some additional work has to be done if heavy edges are present.

4.1 Multi-Criteria Max-ATSP

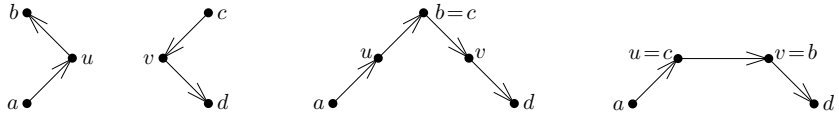
We first present our algorithm for Max-ATSP (Algorithm 3) since it is a bit easier to analyze.

First of all, we compute a $(1-\varepsilon)$ approximate Pareto curve $\mathcal{C}$ of cycle covers. Then, for every cycle cover $C \in \mathcal{C}$, we decide whether it is a light-weight cycle cover or a heavy-weight cycle cover (line 7). If C has only light-weight edges,

$\mathcal{P}_{\text{TSP}} \leftarrow$ MC-MaxATSP(G, w, k, ε)
input: directed complete graph $G = (V, E)$, $k \geq 1$, edge weights $w : E \to \mathbb{N}^k$, $\varepsilon > 0$
output: approximate Pareto curve $\mathcal{P}_{\text{TSP}}$ for k-criteria Max-TSP

if $k = 1$ **then**
  compute a 2/3 approximation $\mathcal{P}_{\text{TSP}}$
else
  compute a $(1 - \varepsilon)$ approximate Pareto curve $\mathcal{C}$ of cycle covers
  $\mathcal{P}_{\text{TSP}} \leftarrow \emptyset$
  for all cycle covers $C \in \mathcal{C}$ **do**
    if $w(e) \leq \alpha_k^d \cdot w(C)$ for all edges $e \in C$ **then**
      $P \leftarrow$ Decompose(C, w, k)
      add edges to P to form a Hamiltonian cycle H; add H to $\mathcal{P}_{\text{TSP}}$
    else
      let $e = (u, v) \in C$ be an edge with $w(e) \not\leq \alpha_k^d \cdot w(C)$
      for all $a, b, c, d \in V$ such that $P^e_{a,b,c,d}$ is legal **do**
        for $i \leftarrow 1$ **to** k **do**
          obtain G' from G by contracting the paths of $P^e_{a,b,c,d}$
          obtain w' from w by removing the ith objective
          $\mathcal{P}'_{\text{TSP}} \leftarrow$ MC-MaxATSP$(G', w', k - 1, \varepsilon)$
          for all $H' \in \mathcal{P}'_{\text{TSP}}$ **do**
            form a Hamilton cycle from H' plus $P^e_{a,b,c,d}$; add it to $\mathcal{P}_{\text{TSP}}$
            form a Hamilton cycle from H' plus (u, v); add it to $\mathcal{P}_{\text{TSP}}$

Algorithm 3: Approximation algorithm for k-criteria Max-ATSP.



(a) Two disjoint paths. (b) With $b = c$. (c) Including edge e.

Fig. 2. The three possibilities of $P^e_{a,b,c,d}$. Symmetrically to (b), we also have $a = d$. Symmetrically to (c), we also have $v = a$ and $u = d$.

we decompose it to obtain a collection P of paths. Then we add edges to P to obtain a Hamiltonian cycle H, which we then add to $\mathcal{P}_{\text{TSP}}$.

If C contains a heavy-weight edge, then there exists an edge $e = (u, v)$ and an i with $w_i(e) > \alpha_k \cdot w_i(C)$. We pick one such edge. Then we iterate over all possible vertices a, b, c, d (including equalities and including u and v). We denote by $P^e_{a,b,c,d}$ the graph with vertices u, v, a, b, c, d and edges (a, u), (u, b), (c, v), and (v, d). We call $P^e_{a,b,c,d}$ *legal* if it can be extended to a Hamiltonian cycle: $P^e_{a,b,c,d}$ is legal if and only if it consists of one or two vertex-disjoint directed paths. Figure 2 shows the different possibilities.

For every legal $P^e_{a,b,c,d}$, we contract the paths as follows: We remove all outgoing edges of a and c, all incoming edges of b and d, and all edges incident to u or v. Then we identify a and b as well as c and d. If $P^e_{a,b,c,d}$ consists of a single path, then we remove all vertices except the two endpoints of this path, and we identify these two endpoints.

In this way, we obtain a slightly smaller instance G'. Then, for every i, we remove the ith objective to obtain w', and recurse on G' with only $k-1$ objectives w'. This yields a approximate Pareto curves $\mathcal{P}'_{\text{TSP}}$ of Hamiltonian cycles of G'. Now consider any $H' \in \mathcal{P}'_{\text{TSP}}$. We expand the contracted paths to obtain H. Then we construct two tours: First, we just add $P^e_{a,b,c,d}$ to H, which yields a Hamiltonian cycle by construction. Second, we observe that no edge in H is incident to u or v. We add the edge (u, v) to H as well as some more edges such that we obtain a Hamiltonian cycle. We put the Hamiltonian cycles thus constructed into $\mathcal{P}_{\text{TSP}}$.

We have not yet discussed the success probability. Randomness is needed for computing the approximate Pareto curves of cycle covers and the recursive calls of MC-MaxATSP with $k-1$ objectives. Let N be the size of the instance at hand, and let $p_k(N, 1/\varepsilon)$ is a polynomial that bounds the size of a $(1-\varepsilon)$ approximate Pareto curve from above. Then we need at most $N^4 \cdot p_k(N, 1/\varepsilon)$ recursive calls of MC-MaxATSP. In total, the number of calls of randomized algorithms is bounded by some polynomial $q_k(N, 1/\varepsilon)$. We amplify the success probabilities of these calls such that the probability is at least $1 - \frac{1}{2 \cdot q_k(N,1/\varepsilon)}$. Thus, the probability that one such call is not successful is at most $q_k(N, 1/\varepsilon) \cdot \frac{1}{2 \cdot q_k(N,1/\varepsilon)} \leq 1/2$ by a union bound. Hence, the success probability of the algorithm is at least $1/2$.

Instead of Decompose, we can also use RandDecompose. We modify RandDecompose such that the running-time is guaranteed to be polynomial and that there is only a small probability that RandDecompose errs. Furthermore, we have to make the error probabilities of the cycle cover computation as well as the recursive calls of MC-MaxATSP slightly smaller to maintain an overall success probability of at least $1/2$.

The running-time of MC-MaxATSP is polynomial in the input size and $1/\varepsilon$, which can be seen by induction on k: We have a polynomial time approximation algorithm for $k = 1$. For $k > 1$, the approximate Pareto curve of cycle covers can be computed in polynomial time, yielding a polynomial number of cycle covers. All further computations can also be implemented to run in polynomial time since MC-MaxATSP for $k-1$ runs in polynomial time by induction hypothesis.

Theorem 3. MC-MaxATSP *is a randomized* $\frac{1}{k+1} - \varepsilon$ *approximation for multi-criteria Max-ATSP. Its running-time is polynomial in the input size and* $1/\varepsilon$.

Proof. We have already discussed the error probabilities and the running-time. Thus, it remains to consider the approximation ratio, and we can assume in the following, that all randomized computations are successful. We prove the theorem by induction on k. For $k = 1$, this follows since mono-criterion Max-ATSP can be approximated with a factor $2/3 > 1/2$.

Now assume that the theorem holds for $k - 1$. We have to prove that, for every Hamiltonian cycle $\hat{H}$, there exists a Hamiltonian cycle $H \in \mathcal{P}_{\text{TSP}}$ with $w(H) \geq \left(\frac{1}{k+1} - \varepsilon\right) \cdot w(\hat{H})$. Since every Hamiltonian cycle is in particular a cycle cover, there exists a $C \in \mathcal{C}$ with $w(C) \geq (1 - \varepsilon) \cdot w(\hat{H})$. Now we distinguish two cases. The first case is that C consists solely of light-weight edges, i. e., $w(e) \leq \frac{1}{k+1} \cdot w(C)$, then Decompose returns a collection P of paths with

$w(P) \geq \frac{1}{k+1} \cdot w(C) \geq \left(\frac{1}{k+1} - \varepsilon\right) \cdot w(\hat{H})$, which yields a Hamiltonian cycle H with $w(H) \geq w(P) \geq \left(\frac{1}{k+1} - \varepsilon\right) \cdot w(\hat{H})$ as claimed.

The second case is that C contains at least one heavy-weight edge $e = (u, v)$. Let (a, u), (u, b), (c, v), and (v, d) be the edges in $\hat{H}$ that are incident to u or v. (We may have some equalities among the vertices as shown in Figure 2.) Note that $\hat{H}$ does not necessarily contain the edge e. We consider the corresponding $P^e_{a,b,c,d}$ and divide the second case into two subcases.

The first subcase is that there is a $j \in [k]$ with $w_j(P^e_{a,b,c,d}) \geq \frac{1}{k+1} \cdot w_j(\hat{H})$, i. e., at least a $\frac{1}{k+1}$ fraction of the jth objective is concentrated in $P^e_{a,b,c,d}$. (We can have $j = i$ or $j \neq i$.) Let $J \subseteq [k]$ be the set of such j.

We fix one $j \in J$ arbitrarily and consider the graph G' obtained by removing the jth objective and contracting the paths (a, u, b) and (c, v, d). A fraction of $1 - \frac{1}{k+1} = \frac{k}{k+1}$ of the weight of $\hat{H}$ is left in G' with respect to all objectives but those in J. Thus, G' contains a Hamiltonian cycle $\hat{H}'$ with $w_\ell(\hat{H}') \geq \frac{k}{k+1} \cdot w_\ell(\hat{H})$ for all $\ell \in [k] \setminus J$. Since $(k-1)$-criteria Max-ATSP can be approximated with a factor of $\frac{1}{k} - \varepsilon$ by assumption, $\mathcal{P}'_{\text{TSP}}$ contains a Hamiltonian cycle H' with $w_\ell(H') \geq (\frac{1}{k} - \varepsilon) \cdot \frac{k}{k+1} \cdot w_\ell(\hat{H}) \geq \left(\frac{1}{k+1} - \varepsilon\right) \cdot w_\ell(\hat{H})$ for all $\ell \in [k] \setminus J$. Together with $P^e_{a,b,c,d}$, which contributes enough weight to the objectives in J, we obtain a Hamiltonian cycle H with $w(H) \geq \left(\frac{1}{k+1} - \varepsilon\right) \cdot w(\hat{H})$, which is as claimed.

The second subcase is that $w_j(P^e_{a,b,c,d}) \leq \frac{1}{k+1} \cdot w_j(H)$ for all $j \in [k]$. Thus, at least a fraction of $\frac{k}{k+1}$ of the weight of $\hat{H}$ is outside of $P^e_{a,b,c,d}$. We consider the case with the ith objective removed. Then, with the same argument as in the first subcase, we obtain a Hamiltonian cycle H' of G' with $w_\ell(H') \geq \left(\frac{1}{k+1} - \varepsilon\right) \cdot w_\ell(\hat{H})$ for all $\ell \in [k] \setminus \{i\}$. To obtain a Hamiltonian cycle of G, we take the edge $e = (u, v)$ and connect its endpoints appropriately. (For instance, if a, b, c, d are distinct, then we add the path (a, u, v, d) and the edge (c, b).) This yields enough weight for the ith objective in order to obtain a Hamiltonian cycle H with $w(H) \geq \left(\frac{1}{k+1} - \varepsilon\right) \cdot w(\hat{H})$ since $w_i(e) \geq \frac{1}{k+1} \cdot w(C) \geq \left(\frac{1}{k+1} - \varepsilon\right) \cdot w(\hat{H})$.

4.2 Multi-Criteria Max-STSP

MC-MaxATSP works of course also for undirected graphs, for which it achieves an approximation ratio of $\frac{1}{k+1} - \varepsilon$. But we can do better for undirected graphs.

Our algorithm MC-MaxSTSP for undirected graphs (Algorithm 4) starts by computing an approximate Pareto curve of cycle covers just as MC-MaxATSP did. Then we consider each cycle cover C separately. If C consists solely of light-weight edges, then we can decompose C using Decompose. If C contains one or more heavy-weight edges, then some more work has to be done than in the case of directed graphs. The reason is that we cannot simply contract paths – this would make the new graph G' (and the edge weights w') asymmetric.

So assume that a cycle cover $C \in \mathcal{C}$ contains a heavy-weight edge $e = \{u, v\}$. Let $i \in [k]$ be such that $w_i(e) \geq w_i(C)/k$. In a first attempt, we remove the ith objective to obtain w'. Then we set $w'(f) = 0$ for all edges f incident to u or v. We recurse with $k - 1$ objectives on G with edge weights w'. This yields a

$\mathcal{P}_{\text{TSP}} \leftarrow$ MC-MaxSTSP(G, w, k, ε)
input: undirected complete graph $G = (V, E)$, $k \geq 2$, edge weights $w : E \to \mathbb{N}^k$, $\varepsilon > 0$
output: approximate Pareto curve $\mathcal{P}_{\text{TSP}}$ for k-criteria Max-TSP

```
 1: compute a (1 − ε) approximate Pareto curve C of cycle covers
 2: P_TSP ← ∅
 3: if k = 2 then
 4:     for all C ∈ C do
 5:         P ← Decompose(C, w, k)
 6:         add edges to P to form a Hamiltonian cycle H; add H to P_TSP
 7: else
 8:     for all cycle covers C ∈ C do
 9:         if w(e) ≤ w(C)/k for edges e ∈ C then
10:             P ← Decompose(C, w, k)
11:             add edges to P to form a Hamiltonian cycle H; add H to P_TSP
12:         else
13:             let i ∈ [k] and e = {u, v} ∈ C with w_i(e) > w_i(C)/k
14:             for all ℓ ∈ {0, ..., 4k}, distinct x_1, ..., x_ℓ ∈ V \ {u, v}, and k ∈ [k] do
15:                 U ← {x_1, ..., x_ℓ, u, v}
16:                 obtain w' from w by removing the jth objective
17:                 set w'(f) = 0 for all edges f incident to U
18:                 P_TSP^{U,j} ← MC-MaxSTSP(G, w', k − 1, ε)
19:                 for all H ∈ P_TSP^{U,j} do
20:                     remove all edges f from H with f ⊆ U to obtain H'
21:                     for all H_U such that H' ∪ H_U is a Hamiltonian cycle do
22:                         add H' ∪ H_U to P_TSP
```

Algorithm 4: Approximation algorithm for k-criteria Max-STSP.

tour H' on G. Now we remove all edges incident to u or v of H' and add new edges including e. In this way, we get enough weight with respect to objective i. Unfortunately, there is a problem if there is an objective j and an edge f incident to u or v such that f contains almost all weight with respect to w_j: We cannot guarantee that this edge f is included in H without further modifying H'. To cope with this problem, we do the following: In addition to u and v, we set the weight of all edges incident to the other vertex of f to 0. Then we recurse. Unfortunately, there may be another objective j' that now causes problems. To solve the whole problem, we iterate over all $\ell = 0, \ldots, 4k$ and over all additional vertices $x_1, \ldots, x_\ell \neq u, v$. Let $U = \{x_1, \ldots, x_\ell, u, v\}$. We remove one objective $i \in [k]$ to obtain w', set the weight of all edges incident to U to 0, and recurse with $k - 1$ objectives. Although the time needed to do this is exponential in k, we maintain polynomial running-time for fixed k.

As in the case of directed graphs, we can make the success probability of every randomized computation small enough to maintain a success probability of at least $1/2$.

The base case is now $k = 2$: In this case, every cycle cover possesses a $1/2$ decomposition, and we do not have to care about heavy-weight edges. Overall, we obtain the following result.

Theorem 4. MC-MaxSTSP *is a randomized* $\frac{1}{k} - \varepsilon$ *approximation for multi-criteria Max-STSP. Its running-time is polynomial in the input size and* $1/\varepsilon$.

5 Remarks

The analysis of the approximation ratios of our algorithms is essentially optimal: Our approach can at best lead to approximation ratios of $\frac{1}{k+c}$ for some $c \in \mathbb{Z}$. The reason is as follows: Assume that $(k-1)$-criteria Max-TSP can be approximated with a factor of τ_k. If we have a k-criteria instance, we have to set the threshold for heavy-weight edges somewhere. Assume for the moment that this threshold α_k be arbitrary. Then the ratio for k-criteria Max-TSP is $\min\{\alpha_k, (1-\alpha_k) \cdot \tau_{k-1}\}$. Choosing $\alpha_k = \frac{\tau_{k-1}}{\tau_{k-1}+1}$ maximizes this ratio. Thus, if $\tau_{k-1} = 1/T$ for some T, then $\tau_k \leq \frac{\tau_{k-1}}{\tau_{k-1}+1} = \frac{1}{T+1}$. We conclude that the denominator of the approximation ratio increases by at least 1 if we go from $k-1$ to k.

For undirected graphs, we have obtained a ratio of roughly $1/k$, which is optimal since $\alpha_2^u = 1/2$ implies $c \geq 0$. Similarly, for directed graphs, we have a ratio of $\frac{1}{k+1}$, which is also optimal since $\alpha_2^d = 1/3$ implies $c \geq 1$.

Due to the existence of $\Omega(1/\log k)$-decompositions, we conjecture that both k-criteria Max-STSP and k-criteria Max-ATSP can in fact be approximated with factors of $\Omega(1/\log k)$. This, however, requires a different approach or at least a new technique for heavy-weight edges.

References

1. Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, 1993.
2. Markus Bläser, Bodo Manthey, and Oliver Putz. Approximating multi-criteria Max-TSP. Computing Research Repository, arXiv:0806.3668 [cs.DS], 2008.
3. Zhi-Zhong Chen, Yuusuke Okamoto, and Lusheng Wang. Improved deterministic approximation algorithms for Max TSP. *Information Processing Letters*, 95(2):333–342, 2005.
4. Matthias Ehrgott. *Multicriteria Optimization*. Springer, 2005.
5. Haim Kaplan, Moshe Lewenstein, Nira Shafrir, and Maxim I. Sviridenko. Approximation algorithms for asymmetric TSP by decomposing directed regular multigraphs. *Journal of the ACM*, 52(4):602–626, 2005.
6. Bodo Manthey. Approximate pareto curves for the asymmetric traveling salesman problem. Computing Research Repository cs.DS/0711.2157, arXiv, 2007.
7. Bodo Manthey and L. Shankar Ram. Approximation algorithms for multi-criteria traveling salesman problems. *Algorithmica*, to appear.
8. Christos H. Papadimitriou and Mihalis Yannakakis. On the approximability of trade-offs and optimal access of web sources. In *Proc. of the 41st Ann. IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 86–92. IEEE Computer Society, 2000.