Applications of Discrepancy Theory in Multiobjective Approximation

Christian Glaßer, Christian Reitwießner, and Maximilian Witek

Julius-Maximilians-Universität Würzburg
Institut für Informatik
Lehrstuhl für Theoretische Informatik
Am Hubland, 97074 Würzburg, Germany
{glasser,reitwiessner,witek}@informatik.uni-wuerzburg.de

Abstract

We apply a multi-color extension of the Beck-Fiala theorem to show that the multiobjective maximum traveling salesman problem is randomized $1/2$-approximable on directed graphs and randomized $2/3$-approximable on undirected graphs. Using the same technique we show that the multiobjective maximum satisfiabilty problem is $1/2$-approximable.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Discrepancy Theory, Multiobjective Optimization, Satisfiability, Traveling Salesman

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2011.55

1 Introduction

We study multiobjective variants of the traveling salesman problem and the satisfiability problem.

- The k-objective maximum traveling salesman problem: Given is a directed / undirected complete graph with edge weights from $\mathbb{N}^k$. Find a Hamiltonian cycle of maximum weight.
- The k-objective maximum weighted satisfiability problem: Given is a Boolean formula in conjunctive normal form and for each clause a non-negative weight in $\mathbb{N}^k$. Find a truth assignment that maximizes the sum of the weights of all satisfied clauses.

In general we cannot expect to find a single solution that is optimal with respect to all objectives. Instead we are interested in the *Pareto set* which consists of all optimal solutions in the sense that there is no solution that is at least as good in all objectives and better in at least one objective. Typically, the Pareto set has exponential size, and this particularly holds for the traveling salesman and the satisfiability problems considered here. We are hence interested in computing an approximation of the Pareto set.

A popular strategy for approximating single-objective traveling salesman and single-objective satisfiability is to compute two or more alternatives out of which one chooses the best one:

- For each cycle in a maximum cycle cover of a graph, remove the edge with *the lowest weight*, and connect the remaining paths to a Hamiltonian cycle.
- For some formula, take an arbitrary truth assignment and its complementary truth assignment, and return the one with *the highest weight* of satisfied clauses.

However, in the presence of multiple objectives, these alternatives can be incomparable and hence we need an argument that allows to *appropriately combine* incomparable alternatives.

31st Int'l Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2011).
Editors: Supratik Chakraborty, Amit Kumar; pp. 55–65

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

While previous work focused on problem-specific properties to construct solutions of good quality, we show that the Beck-Fiala theorem [3] from discrepancy theory and its multi-color extension due to Doerr and Srivastav [5] provide a general and simple way to combine alternatives appropriately. Its application leads to simplified and improved approximation algorithms for the k-objective maximum traveling salesman problem on directed and undirected graphs and the k-objective maximum weighted satisfiability problem.

2 Preliminaries

2.1 Multiobjective Optimization

Let $k \geq 1$ and consider some k-objective maximization problem $\mathcal{O}$ that consists of a set of instances $\mathcal{I}$, a set of solutions $S(x)$ for each instance $x \in \mathcal{I}$, and a function w assigning a k-dimensional weight $w(x,s) \in \mathbb{N}^k$ to each solution $s \in S(x)$ depending also on the instance $x \in \mathcal{I}$. If the instance x is clear from the context, we also write $w(s) = w(x,s)$. The components of w are written as w_i. For weights $a = (a_1, \ldots, a_k)$, $b = (b_1, \ldots, b_k) \in \mathbb{N}^k$ we write $a \geq b$ if $a_i \geq b_i$ for all i.

Let $x \in \mathcal{I}$. The Pareto set of x, the set of all optimal solutions, is the set $\{s \in S(x) \mid \neg\exists s' \in S(x)\ (w(x,s') \geq w(x,s) \text{ and } w(x,s') \neq w(x,s))\}$. For solutions $s, s' \in S(x)$ and $\alpha < 1$ we say s is α-approximated by s' if $w_i(s') \geq \alpha \cdot w_i(s)$ for all i. We call a set of solutions α-approximate Pareto set of x if every solution $s \in S(x)$ (or equivalently, every solution from the Pareto set) is α-approximated by some s' contained in the set.

We say that some algorithm is an α-approximation algorithm for $\mathcal{O}$ if it runs in polynomial time and returns an α-approximate Pareto set of x for all inputs $x \in \mathcal{I}$. We call it randomized if it is allowed to fail with probability at most $1/2$. An algorithm is a PTAS (polynomial-time approximation scheme) for $\mathcal{O}$, if on input x and $0 < \varepsilon < 1$ it computes a $(1-\varepsilon)$-approximate Pareto set of x and for each fixed ε, its runtime is polynomial in the length of x. If there is a single polynomial in $1/\varepsilon + \text{length}(x)$ that bounds the algorithm's runtime, we call it FPTAS (fully polynomial-time approximation scheme). The randomized variants are called PRAS (polynomial-time randomized approximation scheme) and FPRAS (fully polynomial-time randomized approximation scheme).

2.2 Graph Prerequisites

An *$\mathbb{N}^k$-labeled directed (undirected) graph* is a tuple $G = (V, E, w)$, where V is some finite set of vertices, $E \subseteq V \times V$ ($E \subseteq \binom{V}{2}$) is a set of directed (undirected) edges, and $w\colon E \to \mathbb{N}^k$ is a k-dimensional weight function. If $E = (V \times V) \setminus \{(i,i) \mid i \in V\}$ ($E = \binom{V}{2}$) then G is called *complete*. We denote the i-th component of w by w_i and extend w to sets of edges by taking the sum over the weights of all edges in the set. A *cycle (of length $m \geq 1$)* in G is an alternating sequence of vertices and edges $v_0, e_1, v_1, \ldots e_m, v_m$, where $v_i \in V$, $e_j \in E$, $e_j = (v_{j-1}, v_j)$ ($e_j = \{v_{j-1}, v_j\}$) for all $0 \leq i \leq m$ and $1 \leq j \leq m$, neither the sequence of vertices $v_0, v_1, \ldots, v_{m-1}$ nor the sequence of edges $e_1, \ldots, e_m$ contains any repetition, and $v_m = v_0$. A cycle in G is called *Hamiltonian* if it visits every vertex in G. A set of cycles in G is called *cycle cover* if for every vertex $v \in V$ it contains exactly one cycle that visits v. For simplicity we interpret cycles and cycle covers as sets of edges and can thus (using the above mentioned extension of w to sets of edges) write $w(C)$ for the (multidimensional) weight of a cycle cover C of G.

2.3 Approximating Cycle Covers

We will consider approximation algorithms for the multiobjective traveling salesman problem using a multiobjective version of the maximum cycle cover problem. For directed input graphs we have the following problem definition.

k-Objective Maximum Directed Edge-Fixed c-Cycle Cover (k-c-MaxDCC$_{\mathrm{F}}$)
Instance: $\mathbb{N}^k$-labeled complete directed graph (V, E, w) and $F \subseteq E$
Solution: Cycle cover $C \subseteq E$ with at least c edges per cycle and $F \subseteq C$
Weight: $w(C)$

For undirected input graphs we analogously define the k-objective maximum undirected edge-fixed c-cycle cover problem (***k*-*c*-MaxUCC$_{\mathbf{F}}$**, for short). Let ***k*-*c*-MaxUCC** (***k*-*c*-MaxDCC**) denote the problems we obtain from k-c-MaxDCC$_{\mathrm{F}}$ (k-c-MaxUCC$_{\mathrm{F}}$) if we require $F = \emptyset$. Using this notation we obtain the usual cycle cover problems ***k*-MaxDCC** as k-0-MaxDCC and ***k*-MaxUCC** as k-0-MaxUCC.

Manthey and Ram [14] show by a reduction to matching that there is an FPRAS for k-objective *minimum* cycle cover problems. The same technique can be used to show that there are FPRAS for k-MaxDCC and k-MaxUCC [12]. We show that there are FPRAS for k-2-MaxDCC$_{\mathrm{F}}$ and k-3-MaxUCC$_{\mathrm{F}}$ by a reduction to k-MaxDCC and k-MaxUCC.

▶ **Theorem 1.** *For every $k \geq 1$, k-2-MaxDCC$_{\mathrm{F}}$ and k-3-MaxUCC$_{\mathrm{F}}$ admit an FPRAS.*

Proof. For every $l \geq 1$, let l-MaxDCC-Approx (l-MaxUCC-Approx) denote the FPRAS for l-MaxDCC (l-MaxUCC). We begin with the directed case.

Let $k \geq 1$. On input of the $\mathbb{N}^k$-labeled complete directed graph $G = (V, E, w)$ and $F \subseteq E$, let $G' = (V, E, w')$, where $w' \colon E \to \mathbb{N}^{k+1}$ such that for all $e \in E$,

$$w'_i(e) = w_i(e) \qquad \text{for } 1 \leq i \leq k \qquad \text{and} \qquad w'_{k+1}(e) = \begin{cases} 1 & \text{if } e \in F \\ 0 & \text{otherwise.} \end{cases}$$

For $\varepsilon > 0$, apply $(k+1)$-MaxDCC-Approx to G' with approximation ratio $\varepsilon' = \min\{\varepsilon, 1/(r+1)\}$, where $r := \#F$ and return the obtained set of cycle covers that contain all edges from F.

Let C be some (arbitrary) cycle cover with $F \subseteq C$. If no such cycle cover exists, we are done. Otherwise, we have $w'_{k+1}(C) = r$, and with probability at least $1/2$ the FPRAS must have returned some cycle cover C' that ε'-approximates C. By $\varepsilon' \leq 1/(r+1)$ we have $w'_{k+1}(C') \geq (1 - \varepsilon') \cdot w'_{k+1}(C) \geq (1 - 1/(r+1)) \cdot r = r - r/(r+1) > r - 1$ and hence $F \subseteq C'$. Moreover, by $\varepsilon' \leq \varepsilon$ we have $w_i(C') = w'_i(C') \geq (1-\varepsilon') \cdot w'_i(C) \geq (1-\varepsilon) \cdot w'_i(C) = (1-\varepsilon) \cdot w_i(C)$ for all $1 \leq i \leq k$. Since an arbitrary cycle in a complete directed graph has length at least two, the assertion is proved.

The proof for the undirected case is very similar, as we call $(k+1)$-MaxUCC-Approx instead. Since in a complete undirected graph every cycle has length at least three, the assertion follows. ◀

2.4 Boolean Formulas

We consider formulas over a finite set of propositional variables V, where a *literal* is a propositional variable $v \in V$ or its negation $\overline{v}$, a *clause* is a finite, nonempty set of literals, and a *formula in conjunctive normal form* (**CNF**, for short) is a finite set of clauses. A *truth assignment* is a mapping $I \colon V \to \{0, 1\}$. For some $v \in V$, we say that I *satisfies the literal* v if $I(v) = 1$, and I *satisfies the literal* $\overline{v}$ if $I(v) = 0$. We further say that I *satisfies the clause* C and write $I(C) = 1$ if there is some literal $l \in C$ that is satisfied by I.

3 Multi-Color Discrepancy

Suppose we have a list of items with (single-objective) weights and want to find a subset of these items with about half of the total weight. The exact version of this problem is of course the NP-complete problem Partition [7], and hence it is unlikely that an exact solution can be found in polynomial time. If we allow a deviation in the order of the largest weight, this problem can be solved in polynomial time, though. Surprisingly, this is still true if the weights are not single numbers but vectors of numbers, which follows from a classical result in discrepancy theory known as the Beck-Fiala theorem [3]. It is important to note that the allowed deviation is independent of the number of vectors since this enables us to use this result in multiobjective approximation for balancing out multiple objectives at the same time with an error that does not depend on the input size.

In the Beck-Fiala theorem and the task discussed above, we have to decide for each item to either include it or not. In some situations in multiobjective optimization, though, a more general problem needs to be solved: There is a constant number of weight vectors for each item, out of which we have to choose exactly one. Doerr and Srivastav [5] showed that the Beck-Fiala theorem generalizes to this so-called multi-color setting with almost the same deviation. Their proof implicitly shows that this choice can be computed in polynomial time.

For a vector $x \in \mathbb{Q}^m$ let $||x||_\infty = \max_i |x_i|$, and for a matrix $A \in \mathbb{Q}^{m\times n}$ let $||A||_1 = \max_j \sum_i |a_{ij}|$. For $c \geq 2$, $n \geq 1$ let $\overline{M_{c,n}} = \{x \in (\mathbb{Q} \cap [0,1])^{cn} \mid \sum_{k=0}^{c-1} x_{cb-k} = 1 \text{ for all } b \in \{1,\dots,n\}\}$ and $M_{c,n} = \overline{M_{c,n}} \cap \{0,1\}^{cn}$.

▶ **Theorem 2.** *(Doerr, Srivastav [5]) There is a polynomial-time algorithm that on input of some $A \in \mathbb{Q}^{m\times cn}$, $m,n \in \mathbb{N}$, $c \geq 2$ and $p \in \overline{M_{c,n}}$ finds a coloring $\chi \in M_{c,n}$ such that $||A(p-\chi)||_\infty \leq 2||A||_1$.*

▶ **Corollary 3.** *There is a polynomial-time algorithm that on input of a set of vectors $v^{j,r} \in \mathbb{Q}^m$ for $1 \leq j \leq n$, $1 \leq r \leq c$ computes a coloring $\chi\colon \{1,\dots,n\} \to \{1,\dots,c\}$ such that for each $1 \leq i \leq m$ it holds that*

$$\left| \frac{1}{c} \sum_{j=1}^{n} \sum_{r=1}^{c} v_i^{j,r} - \sum_{j=1}^{n} v_i^{j,\chi(j)} \right| \leq 2m \max_{j,r} |v_i^{j,r}|.$$

Proof. The result is obvious for $c = 1$. For $c \geq 2$, we use Theorem 2. Because the error bound is different for each row, we need to scale the rows of the vectors. Let $\delta_i = \max_{j,r} |v_i^{j,r}|$ for $1 \leq i \leq m$. Let $A = (a_{i,j'}) \in \mathbb{Q}^{m\times cn}$ where $a_{i,(c(j-1)+r)} = \frac{1}{\delta_i} v_i^{j,r}$ (if $\delta_i = 0$, set it to 0) and $p \in \mathbb{Q}^{cn}$ such that $p_i = \frac{1}{c}$ for all $1 \leq i \leq cn$. We obtain a coloring $\chi \in \{0,1\}^{cn}$ such that for each $1 \leq j \leq n$ there is exactly one $1 \leq r \leq c$ such that $\chi_{c(j-1)+r} = 1$ and it holds that $||A(p-\chi)||_\infty \leq 2||A||_1$. Note that because of the scaling, the largest entry in A is 1 and thus we have $||A||_1 \leq m$. Define $\chi'\colon \{1,\dots,n\} \to \{1,\dots,c\}$ by $\chi'(j) = r \iff \chi_{c(j-1)+r} = 1$. For each $1 \leq i \leq m$ we obtain

$$\begin{aligned} 2m\delta_i &\geq 2||\delta_i\, A||_1 \geq |(\delta_i\, A(p-\chi))_i| \\ &= \left| \sum_{j'=1}^{cn} \delta_i a_{ij'} (p_{j'} - \chi_{j'}) \right| = \left| \sum_{j=1}^{n} \sum_{r=1}^{c} \frac{1}{c} v_i^{j,r} - \sum_{j=1}^{n} v_i^{j,\chi'(j)} \right|. \end{aligned}$$

◀

4 Approximating Multiobjective Maximum Traveling Salesman

4.1 Definition

Given some complete $\mathbb{N}^k$-labeled graph as input, our goal is to find a Hamiltonian cycle of maximum weight. For directed graphs this problem is called k-objective maximum asymmetric traveling salesman (k-MaxATSP), while for undirected graphs it is called k-objective maximum symmetric traveling salesman (k-MaxSTSP). Below we give the formal definition of k-MaxATSP, the problem k-MaxSTSP is defined analogously.

k-Objective Maximum Asymmetric Traveling Salesman (k-MaxATSP)
Instance: $\mathbb{N}^k$-labeled directed complete graph (V, E, w)
Solution: Hamiltonian cycle C
Weight: $w(C)$

4.2 Previous Work

In 1979, Fisher, Nemhauser and Wolsey [6] gave a $1/2$-approximation algorithm for single-objective maximum asymmetric traveling salesman (1-MaxATSP) by removing the lightest edge from each cycle of a maximum cycle cover and connecting the remaining paths to a Hamiltonian cycle. Since undirected cycles always contain at least three edges, this also showed that single-objective maximum symmetric traveling salesman (1-MaxSTSP) is $2/3$-approximable. Since then, many improvements were achieved, and currently, the best known approximation ratios of $2/3$ for 1-MaxATSP and $7/9$ for 1-MaxSTSP are due to Kaplan et al. [9] and Paluch, Mucha and Madry [15].

Most single-objective approximation algorithms do not directly translate to the case of multiple objectives, and hence we need more sophisticated algorithms. For k-MaxATSP and k-MaxSTSP, where $k \geq 2$, the currently best known approximation algorithms are due to Manthey, who showed a randomized $(1/2-\varepsilon)$-approximation of k-MaxATSP and a randomized $(2/3-\varepsilon)$-approximation of k-MaxSTSP [12]. Recently, Manthey also gave a deterministic $(1/2k-\varepsilon)$-approximation of k-MaxSTSP and a deterministic $(1/(4k-2)-\varepsilon)$-approximation of k-MaxATSP [13].

4.3 Our Results

We show that k-MaxATSP is randomized $1/2$-approximable and k-MaxSTSP is randomized $2/3$-approximable using the following idea. We choose a suitable number l depending only on k and try all sets of at most l edges F using brute force. For each such F we apply the FPRAS for k-2-MaxDCC$_\mathrm{F}$ (k-3-MaxUCC$_\mathrm{F}$), which exists by Theorem 1, fixing the edges in F. For all cycle covers thus obtained, we select two (three) edges from each cycle and compute a 2-coloring (3-coloring) of the cycles with low discrepancy with regard to the weight vectors of the selected edges. Using this coloring, we remove exactly one edge from each cycle and connect the remaining simple paths to a single cycle in an arbitrary way. Since the coloring has low discrepancy, we only remove about one half (one third) of the weight in each objective. The introduced error is absorbed by choosing suitable heavy edges F at the beginning. The described procedure generally works for arbitrary c-cycle covers.

▶ **Lemma 4.** *Let $c \geq 2$ and $k \geq 1$. If there exists an FPRAS for k-c-MaxDCC$_\mathrm{F}$ (k-c-MaxUCC$_\mathrm{F}$, resp.), then the algorithm* Alg-k-MaxTSP *computes a randomized $(1-1/c)$-approximation for k-MaxATSP (k-MaxSTSP, resp.).*

Algorithm: Alg-k-MaxTSP(V, E, w) with parameter $c \geq 2$

Input : $\mathbb{N}^k$-labeled directed/undirected complete graph $G = (V, E, w)$
Output : set of Hamiltonian cycles of G

1 `foreach` $F_H, F_L \subseteq E$ with $\#F_H \leq 3\,c\,k^2$, $\#F_L \leq c\#F_H$ `do`
2 let $\delta \in \mathbb{N}^k$ with $\delta_i = \max\{n \in \mathbb{N} \mid$ there are $3\,c\,k$ edges $e \in F_H$ with $w_i(e) \geq n\}$;
3 `foreach` $e \in E \setminus F_H$ `do`
4 `if` $w(e) \not\leq \delta$ `then` set $w(e) = 0$ for the current iteration of line 1;
5 compute $(1 - 1/\#V)$-approx. $\mathcal{S}$ of k-c-MaxDCC$_{\mathrm{F}}$ / k-c-MaxUCC$_{\mathrm{F}}$ on $(G, F_H \cup F_L)$;
6 `foreach` cycle cover $S \in \mathcal{S}$ `do`
7 let $C_1, \ldots, C_r$ denote the cycles in S;
8 `if` for each $i \in \{1, \ldots, r\}$, $C_i \setminus F_H$ contains a path of length c `then`
9 `foreach` $i \in \{1, \ldots, r\}$ `do` choose path $e_{i,1}, \ldots, e_{i,c} \in C_i \setminus F_H$ arbitrarily;
10 compute some coloring $\chi\colon \{1, \ldots, r\} \to \{1, \ldots, c\}$ such that

$$\sum_{i=1}^{r} w(e_{i,\chi(i)}) \quad \leq \quad 2k \cdot \delta + \frac{1}{c} \sum_{i=1}^{r} \sum_{j=1}^{c} w(e_{i,j})$$

  and remove the edges $\{e_{i,\chi(i)} \mid 1 \leq i \leq r\}$ from S;
11 output the remaining edges, arbitrarily connected to a Hamiltonian cycle;

Proof. Let $k \geq 1$, $c \geq 2$, and $G = (V, E, w)$ be some $\mathbb{N}^k$-labeled (directed or undirected) input graph with $m = \#V$ sufficiently large.

We will first argue that the algorithm terminates in time polynomial in the length of G. Since there are only polynomially many subsets $F_H, F_L \subseteq E$ with cardinality bounded by a constant, the loop in line 1 is executed polynomially often. In each iteration the FPRAS on $G = (V, E, w)$ and $F_H \cup F_L \subseteq E$ terminates in time polynomial in the length of G and $F_H \cup F_E$, which means that the set $\mathcal{S}$ contains only polynomially many cycle covers. Hence, for each iteration of the loop in line 1, the loop in line 6 is also executed at most polynomially many times, and overall we have polynomially many nested iterations. In each nested iteration where each cycle of the cycle cover contains a path as required, we compute a coloring of $\{1, \ldots, r\}$ with low discrepancy. By Corollary 3 this can be done in polynomial time. Observe that all further steps require at most polynomial time, and hence the algorithm terminates after polynomially many steps.

Next we argue that the algorithm will succeed with probability at least $1/2$. Observe that the only randomized parts of the algorithm are the calls to the randomized cycle cover approximation algorithm in line 5. Using amplification we can assume that the probability that all the calls to this algorithm succeed is at least $1/2$.

It remains to show that if the algorithm Alg-k-MaxTSP succeeds, it outputs some $(1-1/c)$-approximate set of Hamiltonian cycles. Hence, for the remainder of the proof, let us assume that the algorithm and hence all calls to the internal FPRAS succeed. Furthermore, let $R \subseteq E$ be some Hamiltonian cycle of G. We will argue that there is some iteration where the algorithm outputs an $(1 - 1/c)$-approximation of R.

For each $1 \leq i \leq k$, let $F_{H,i} \subseteq R$ be some set of $3\,c\,k$ heaviest edges of R in the i-th component, breaking ties arbitrarily. Let $F_H = \bigcup_{i=1}^{k} F_{H,i}$. We define $F_L \subseteq R$ such that $F_L \cap F_H = \emptyset$ and each edge in F_H is part of a path in $F_L \cup F_H$ that contains c edges from F_L. This is always possible as long as R is large enough. We now have $\#F_H \leq 3\,c\,k^2$ and $\#F_L \leq c\#F_H$. Hence in line 1 there will be some iteration that chooses F_H and F_L. We fix

this iteration for the remainder of the proof.

Let $\delta \in \mathbb{N}^k$ as defined in line 2 and observe that $\delta_i = \min\{w_i(e) \mid e \in F_{H,i}\}$ for all i, which means that for all edges $e \in R \setminus F_H$ we have $w(e) \leq \delta$. Hence the loop in line 3 sets the weights of all edges $e \in E \setminus R$ that do not fulfill $w(e) \leq \delta$ to zero, and these are the only weights that are modified. In particular, this does not affect edges in R, hence $w(R)$ remains unchanged. Note that since we do not increase the weight of any edge and do not change the weight of the edges in R, it suffices to show that the algorithm computes an approximation with respect to the changed weights.

Next we obtain a $(1-1/\#V)$-approximate set $\mathcal{S}$ of c-cycle covers of G that contain $F_H \cup F_L$. Since R is a c-cycle cover of G with $F_H \cup F_L \subseteq R$, there must be some c-cycle cover $S \in \mathcal{S}$ with $F_H \cup F_L \subseteq S$ that $(1 - 1/\#V)$-approximates R. Hence in line 6 there will be some iteration that chooses this S. Again we fix this iteration for the remainder of the proof.

As in line 7, let $C_1, \ldots, C_r$ denote the cycles in S. Note that each cycle contains at least c edges. Since each edge in F_H is part of a path in $F_H \cup F_L$ with at least c edges from F_L, we even know that each cycle contains at least c edges not from F_H and thus the condition in line 8 is fulfilled. Let these edges $e_{i,j}$ be defined as in the algorithm. Note that since $e_{i,j} \notin F_H$ we have $w(e_{i,j}) \leq \delta$ for all i, j, because the weight function was changed accordingly.

In line 10 we compute some $\chi\colon \{1, \ldots, r\} \to \{1, \ldots, c\}$ such that

$$\sum_{i=1}^{r} w(e_{i,\chi(i)}) \quad \leq \quad 2\,k \cdot \delta + \frac{1}{c} \sum_{i=1}^{r} \sum_{j=1}^{c} w(e_{i,j}) \leq 2\,k \cdot \delta + \frac{1}{c} \cdot w(S \setminus F_H).$$

Recall that by Corollary 3 such a coloring exists and can be computed in polynomial time. Removing the chosen edges breaks the cycles into simple paths, which can be arbitrarily connected to a Hamiltonian cycle R'. For the following estimation note that $\delta \leq \frac{w(F_H)}{3\,c\,k}$ and $w(F_H) \geq \frac{3\,c\,k}{m} w(R)$ and recall that $m = \#V = \#R$.

$$\begin{aligned} w(R') \quad &\geq \quad w(S) - \sum_{i=1}^{r} w(e_{i,\chi(i)}) \geq w(S) - 2\,k \cdot \delta - \frac{1}{c} \cdot w(S \setminus F_H) \\ &= \quad \left(1 - \frac{1}{c}\right) w(S) + \frac{1}{c} w(F_H) - 2\,k \cdot \delta \geq \left(1 - \frac{1}{c}\right) w(S) + \frac{1}{3\,c} w(F_H) \\ &\geq \quad \left(1 - \frac{1}{c}\right) \left(1 - \frac{1}{m}\right) w(R) + \frac{k}{m} w(R) \geq \left(1 - \frac{1}{c}\right) w(R) \end{aligned}$$

This proves the assertion. ◀

It is known that 1-c-MaxDCC is APX-hard for all $c \geq 3$ [4] and that 1-c-MaxUCC is APX-hard for $c \geq 5$ [11]. This means that, unless P = NP, there is no PTAS for these problems (and especially not for the variants with fixed edges). Furthermore, the existence of an FPRAS or PRAS for these problems implies NP = RP and thus a collapse of the polynomial-time hierarchy, which is seen as follows.

If an APX-hard problem has a PRAS, then all problems in APX have a PRAS and hence MAX-3SAT has one. There exists an $\varepsilon > 0$ and a polynomial-time computable f mapping CNF formulas to 3-CNF formulas such that if $x \in$ SAT, then $f(x) \in$ 3SAT; and if $x \notin$ SAT, then there is no assignment satisfying more than a fraction of $1 - \varepsilon$ of $f(x)$'s clauses [1, Theorem 10.1]. The PRAS for MAX-3SAT allows us to compute probabilistically a $(1 - \varepsilon/2)$-approximation for $f(x)$ which in turn tells us whether or not $x \in$ SAT. Since this procedure has no false negatives we get RP = NP, which implies a collapse of the polynomial-time hierarchy [10, 17].

So it seems unlikely that there is a PRAS for 1-c-MaxDCC where $c \geq 3$ and 1-c-MaxUCC where $c \geq 5$. However, this does not necessarily mean that the above algorithm is useless for parameters $c \geq 3$ in the directed and $c \geq 5$ in the undirected case: The algorithm could still benefit from a constant-factor approximation for k-c-MaxUCC$_{\mathrm{F}}$ or k-c-MaxDCC$_{\mathrm{F}}$. A simple change in the estimation shows that if the cycle cover algorithm has an approximation ratio of α, the above algorithm provides an approximation with ratio $\alpha(1 - 1/c)$.

▶ **Theorem 5.** *Let $k \geq 1$.*

1. *k-MaxATSP is randomized $1/2$-approximable.*
2. *k-MaxSTSP is randomized $2/3$-approximable.*

Proof. We combine Theorem 1 and Lemma 4. ◀

5 Approximating Multiobjective Maximum Satisfiability

5.1 Definition

Given a formula in CNF and a function that maps each clause to a k-objective weight, our goal is to find truth assignments that maximize the sum of the weights of all satisfied clauses. The formal definition is as follows.

k-Objective Maximum Weighted Satisfiability (k-MaxSAT)
Instance: Formula H in CNF over a set of variables V, weight function $w\colon H \to \mathbb{N}^k$
Solution: Truth assignment $I\colon V \to \{0,1\}$
Weight: Sum of the weights of all clauses satisfied by I, i.e., $w(I) = \sum_{\substack{C \in H \\ I(C)=1}} w(C)$

5.2 Previous Work

The first approximation algorithm for maximum satisfiability is due to Johnson [8], whose greedy algorithm showed that the single-objective 1-MaxSAT problem is $1/2$-approximable. Further improvements on the approximation ratio followed, and the currently best known approximation ratio of 0.7846 for 1-MaxSAT is due to Asano and Williamson [2].

Only little is known about k-MaxSAT for $k \geq 2$. Santana et. al. [16] apply genetic algorithms to a version of the problem that is equivalent to k-MaxSAT with polynomially bounded weights. To our knowledge, the approximability of k-MaxSAT for $k \geq 2$ has not been investigated so far.

5.3 Our Results

We show that k-MaxSAT is $1/2$-approximable mainly by transferring the idea that for any truth assignment, the assignment itself or its complementary assignment satisfies at least one half of all clauses to multidimensional objective functions. We choose some suitable parameter $l \in \mathbb{N}$ depending only on the number of objectives. For a given formula in CNF we try all possible partial truth assignments for each set of at most l variables using brute force and extend each partial assignment to a full assignment in the following way: For each remaining variable v we compute two vectors roughly representing the weight gained by the two possible assignments for v. We then compute a 2-coloring of those weight vectors with low discrepancy which completes the partial assignment to a truth assignment whose weight is at least one half of the total weight of the remaining satisfiable clauses minus some error.

This error can be compensated by choosing l large enough such that the partial assignment already contributes a large enough weight. This results in a $1/2$-approximation for k-MaxSAT.

For a set of clauses H and a variable v let $H[v=1]=\{C\in H \mid v\in C\}$ be the set of clauses that are satisfied if this variable is assigned one, and analogously $H[v=0]=\{C\in H \mid \overline{v}\in C\}$ be the set of clauses that are satisfied if this variable is assigned zero. This notation is extended to sets of variables V by $H[V=i]=\bigcup_{v\in V} H[v=i]$ for $i=0,1$.

Algorithm: Alg-k-MaxSAT(H,w)

Input : Formula H in CNF over the variables $V=\{v_1,\dots,v_m\}$, k-objective weight function $w\colon H\to\mathbb{N}^k$

Output : Set of truth assignments $I\colon V\to\{0,1\}$

1 `foreach` disjoint $V^0,V^1\subseteq V$ with $\#(V^0\cup V^1)\leq 4k^2$ `do`

2 $\quad G:=H\setminus(H[V^0=0]\cup H[V^1=1])$;

3 $\quad \hat{V}^{(1-i)}:=\{v\in V\setminus(V^0\cup V^1)\mid 4k\cdot w(G[v=i])\not\leq w(H\setminus G)\}$, $i=0,1$;

4 $\quad$ `if` $\hat{V}^0\cap\hat{V}^1=\emptyset$ `then`

5 $\quad\quad V':=V\setminus(V^0\cup V^1\cup\hat{V}^0\cup\hat{V}^1)$, $L':=V'\cup\{\overline{v}\mid v\in V'\}$;

6 $\quad\quad G':=(G[V'=0]\cup G[V'=1])\setminus(G[\hat{V}^0=0]\cup G[\hat{V}^1=1])$;

7 $\quad\quad$ for $v_j\in V'$ let $x^{j,i}=\sum\{\frac{w(C)}{\#(C\cap L')}\mid C\in G'[v_j=i]\}$ for $i=0,1$;

8 $\quad\quad$ compute some coloring $\chi\colon V'\to\{0,1\}$ such that

$$\sum_{v_j\in V'} x^{j,\chi(j)}\geq\frac{1}{2}\sum_{v_j\in V'}(x^{j,0}+x^{j,1})-2k\delta$$

$\quad\quad$ where $\delta_r=\max\{x_r^{j,i}\mid v_j\in V', i\in\{0,1\}\}$;

9 $\quad\quad$ let $I(v):=i$ for $v\in V^i\cup\hat{V}^i\cup\chi^{-1}(\{i\})$, $i=0,1$;

10 $\quad\quad$ output I

▶ **Theorem 6.** *k-MaxSAT is $1/2$-approximable for any $k\geq 1$.*

Proof. We show that this approximation is realized by Alg-k-MaxSAT. First note that this algorithm runs in polynomial time since k is constant and the coloring in line 8 can be computed in polynomial time using Corollary 3. For the correctness, let (H,w) be the input where H is a formula over the variables $V=\{v_1,\dots,v_m\}$ and $w\colon H\to\mathbb{N}^k$ is the k-objective weight function. Let $I_o\colon V\to\{0,1\}$ be an optimal truth assignment. We show that there is a loop iteration of Alg-k-MaxSAT(H,w) that outputs a truth assignment I such that $w(I)\geq w(I_o)/2$.

We first note that there are sets V^0 and V^1 with a bounded cardinality of at most $4k^2$ that define a partial truth assignment that contributes a large weight.

▶ **Claim 7.** *There are sets $V^i\subseteq I_o^{-1}(\{i\})$, $i=0,1$ with $\#(V^0\cup V^1)\leq 4k^2$ such that for $G=H\setminus(H[V^0=0]\cup H[V^1=1])$ and any $v\in V\setminus(V^0\cup V^1)$ it holds that*

$$w(G[v=I_o(v)])\leq\frac{1}{4k}w(H\setminus G). \tag{1}$$

Proof Sketch. The set $V^0\cup V^1$ is obtained by iteratively choosing variables such that (one component of) the weight of the remaining clauses that get satisfied if the variable is set to its value under I_o is high, while the components are chosen in a round-robin fashion. Since $V^0\cup V^1$ contains the $4k$ most "influential" variables per objective, none of the remaining variables can have high "influence" on the remaining clauses, because otherwise one of them would have been chosen to belong to $V^0\cup V^1$. ◀

We choose the iteration of the algorithm where V^0 and V^1 equal the sets whose existence is guaranteed by Claim 7. In the following, we use the variables as they are defined in the algorithm. Observe that by the claim it holds that $I_o(v) = i$ for all $v \in \hat{V}^i$ for $i = 0, 1$ and thus $\hat{V}^0 \cap \hat{V}^1 = \emptyset$. Note that

$$\begin{aligned}\sum_{v_j \in V'} x^{j,0} + x^{j,1} &= \sum_{v_j \in V'} \sum_{i \in \{0,1\}} \sum_{C \in G'[v_j = i]} \frac{w(C)}{\#(C \cap L')} \\ &= \sum_{C \in G'} \#(C \cap L') \frac{w(C)}{\#(C \cap L')} \\ &= w(G').\end{aligned}$$

Furthermore, for all $v_j \in V'$ and $i = 0, 1$, we have the bound $x^{j,i} \leq w(G'[v_j = i]) \leq w(G[v_j = i]) \leq \frac{1}{4k} w(H \setminus G)$ because of the definition of V' and $\hat{V}^{(1-i)}$. By Corollary 3, we find a coloring $\chi \colon V' \to \{0, 1\}$ such that for each $1 \leq i \leq k$ it holds that

$$\left| \frac{1}{2} \sum_{v_j \in V'} \sum_{r=0}^{1} x_i^{j,r} - \sum_{v_j \in V'} x_i^{j,\chi(v_j)} \right| \leq 2k \max_{j,r} |x_i^{j,r}| \leq 2k \frac{1}{4k} w_i(H \setminus G) = \frac{1}{2} w_i(H \setminus G)$$

and hence

$$\sum_{v_j \in V'} x^{j,\chi(v_j)} \geq \frac{1}{2} \sum_{v_j \in V'} (x^{j,0} + x^{j,1}) - \frac{1}{2} w(H \setminus G) = \frac{1}{2}(w(G') - w(H \setminus G)).$$

For I being the truth assignment generated in this iteration it holds that

$$w(\{C \in G' \mid I(C) = 1\}) \geq \sum_{v_j \in V'} x^{j,\chi(v_j)} \geq \frac{1}{2}(w(G') - w(H \setminus G)). \tag{2}$$

Furthermore, since I and I_o coincide on $V \setminus V'$, we have

$$\begin{aligned} w(\{C \in H \setminus G' \mid I(C) = 1\}) &= w(\{C \in H \setminus G' \mid I_o(C) = 1\}) && (3) \\ &\geq w(\{C \in H \setminus G \mid I_o(C) = 1\}) \\ &= w(\{H \setminus G\}). && (4)\end{aligned}$$

Thus we finally obtain

$$\begin{aligned} w(I) &= w(\{C \in H \setminus G' \mid I(C) = 1\}) + w(\{C \in G' \mid I(C) = 1\}) \\ &\overset{(2)}{\geq} w(\{C \in H \setminus G' \mid I(C) = 1\}) + \tfrac{1}{2}(w(G') - w(H \setminus G)) \\ &\overset{(3)}{=} w(\{C \in H \setminus G' \mid I_o(C) = 1\}) + \tfrac{1}{2}(w(G') - w(H \setminus G)) \\ &\overset{(4)}{\geq} \tfrac{1}{2} w(\{C \in H \setminus G' \mid I_o(C) = 1\}) + \tfrac{1}{2} w(G') \\ &\geq \tfrac{1}{2} w(I_o). \end{aligned}$$

◀

References

1 S. Arora and C. Lund. Hardness of approximations. In D. Hochbaum, editor, *Approximation Algorithms for NP-hard Problems*. PWS Publishing Company, Boston, 1997.

2 T. Asano and D. P. Williamson. Improved approximation algorithms for MAX SAT. *Journal of Algorithms*, 42(1):173–202, 2002.

3 J. Beck and T. Fiala. "Integer-Making" Theorems. *Discrete Applied Mathematics*, 3(1):1–8, 1981.

4 M. Bläser and B. Manthey. Approximating maximum weight cycle covers in directed graphs with weights zero and one. *Algorithmica*, 42(2):121–139, 2005.

5 B. Doerr and A. Srivastav. Multicolour discrepancies. *Combinatorics, Probability & Computing*, 12(4):365–399, 2003.

6 M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. An analysis of approximations for finding a maximum weight Hamiltonian circuit. *Operations Research*, 27(4):799–809, 1979.

7 M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

8 D. S. Johnson. Approximation algorithms for combinatorial problems. *Journal of Computer System Sciences*, 9(3):256–278, 1974.

9 H. Kaplan, M. Lewenstein, N. Shafrir, and M. Sviridenko. Approximation algorithms for asymmetric TSP by decomposing directed regular multigraphs. *Journal of the ACM*, 52(4):602–626, 2005.

10 C. Lautemann. BPP and the polynomial hierarchy. *Information Processing Letters*, 17:215–217, 1983.

11 B. Manthey. On approximating restricted cycle covers. *SIAM J. Comput.*, 38(1):181–206, 2008.

12 B. Manthey. On approximating multi-criteria TSP. In S. Albers and J.-Y. Marion, editors, *26th Intern. Symposium on Theoretical Aspects of Computer Science, STACS 2009*, pages 637–648. Dagstuhl Research Online Publication Server, 2009.

13 B. Manthey. Deterministic algorithms for multi-criteria TSP. In *Proceedings of the International Conference on Theory and Applications of Models of Computation*, volume 6648 of *Lecture Notes in Computer Science*, pages 264–275. Springer Verlag, 2011.

14 B. Manthey and L. S. Ram. Approximation algorithms for multi-criteria traveling salesman problems. *Algorithmica*, 53(1):69–88, 2009.

15 K. Paluch, M. Mucha, and A. Madry. A 7/9 - approximation algorithm for the maximum traveling salesman problem. In I. Dinur, K. Jansen, J. Naor, and J. Rolim, editors, *Proceedings of APPROX/RANDOM*, volume 5687 of *Lecture Notes in Computer Science*, pages 298–311. Springer Berlin / Heidelberg, 2009.

16 R. Santana, C. Bielza, J. A. Lozano, and P. Larrañaga. Mining probabilistic models learned by EDAs in the optimization of multi-objective problems. In *GECCO '09: Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, pages 445–452, New York, NY, USA, 2009. ACM.

17 M. Sipser. A complexity theoretic approach to randomness. In *Proceedings of the 15th Symposium on Theory of Computing*, pages 330–335, 1983.