Multiobjective Optimization: Improved FPTAS for Shortest Paths and Non-linear Objectives with Applications *

George Tsaggouris and Christos Zaroliagis

Computer Technology Institute, Patras University Campus, 26500 Patras, Greece;
and Dept of Computer Eng & Informatics, University of Patras, 26500 Patras, Greece.
{tsaggour,zaro}@ceid.upatras.gr

Abstract. We provide an improved FPTAS for multiobjective shortest paths, a fundamental (NP-hard) problem in multiobjective optimization, along with a new generic method for obtaining FPTAS to *any* multiobjective optimization problem with *non-linear* objectives. We show how these results can be used to obtain better approximate solutions to three related problems that have important applications in QoS routing and in traffic optimization.

1 Introduction

Multiobjective shortest paths (MOSP) is a core problem in the area of multiobjective optimization [3, 4] with numerous applications. Informally, the problem consists in finding a set of paths that captures not a single optimum but the trade-off among $d > 1$ objective functions in a digraph whose edges are associated with d-dimensional attribute (cost) vectors. In general, an instance of a multiobjective optimization problem is associated with a set of feasible solutions Q and a d-vector function $\mathbf{f} = [f_1, \ldots, f_d]^T$ (d is typically a constant) associating each feasible solution $q \in Q$ with a d-vector $\mathbf{f}(q)$ (w.l.o.g. we assume that all objectives f_i, $1 \leq i \leq d$, are to be minimized). In a multiobjective optimization problem, we are interested not in finding a single optimal solution, but in computing the trade-off among the different objective functions, called the *Pareto set or curve* $\mathcal{P}$, which is the set of all feasible solutions in Q whose vector of the various objectives is *not* dominated by any other solution (a solution p *dominates* another solution q iff $f_i(p) \leq f_i(q)$, $\forall 1 \leq i \leq d$). Multiobjective optimization problems are usually NP-hard (as indeed is the case for MOSP). This is due to the fact that the Pareto curve is typically exponential in size (even in the case of two objectives). On the other hand, even if a decision maker is armed with the entire Pareto curve, s/he is left with the problem of which is the "best" solution for the application at hand. Consequently, three natural approaches to solve multiobjective optimization problems are to: (i) study approximate versions of the Pareto curve; (ii) optimize one objective while bounding the rest

* This work was partially supported by the FET Unit of EC (IST priority – 6th FP), under contracts no. FP6-021235-2 (ARRIVAL) and no. IST-2002-001907 (DELIS).

(*constrained approach*); and (iii) proceed in a normative way and choose the "best" solution by introducing a utility (typically non-linear) function on the objectives (*normalization approach*). In this paper, we investigate all of them for MOSP.

Multiobjective Shortest Paths. Despite so much research in multiobjective optimization [3, 4], only recently a systematic study of the complexity issues regarding the construction of approximate Pareto curves has been initiated [11, 14]. Informally, an $(1+\varepsilon)$-*Pareto curve* $\mathcal{P}_\varepsilon$ is a subset of feasible solutions such that for any Pareto optimal solution, there exists a solution in $\mathcal{P}_\varepsilon$ that is no more than $(1+\varepsilon)$ away in all objectives. Papadimitriou and Yannakakis show in a seminal work [11] that for any multiobjective optimization problem there exists a $(1+\varepsilon)$-Pareto curve $\mathcal{P}_\varepsilon$ of (polynomial) size $|\mathcal{P}_\varepsilon| = O((4B/\varepsilon)^{d-1})$, where B is the number of bits required to represent the values in the objective functions (bounded by some polynomial in the size of the input); $\mathcal{P}_\varepsilon$ can be constructed by $O((4B/\varepsilon)^d)$ calls to a GAP routine that solves (in time polynomial in the size of the input and $1/\varepsilon$) the following problem: given a vector of values $\mathbf{a}$, either compute a solution that dominates $\mathbf{a}$, or report that there is no solution better than $\mathbf{a}$ by at least a factor of $1+\varepsilon$ in all objectives.

For the case of MOSP (and some other problems with linear objectives), it is shown in [11] how a GAP routine can be constructed (based on a pseudopolynomial algorithm for computing exact paths), and consequently a FPTAS is provided. Note that FPTAS for MOSP were already known in the case of two objectives [8], as well as in the case of multiple objectives in directed acyclic graphs (DAGs) [15]. In particular, the 2-objective case has been extensively studied [4], while for $d > 2$ very little has been achieved; actually, the results in [11, 15] are the only and currently best FPTAS known (see Table 1). Let C^{max} denote the ratio of the maximum to the minimum edge weight (in any dimension), and let n (resp. m) be the number of nodes (resp. edges) in a digraph.

Our first contribution in this work (Section 3) is a new and remarkably simple FPTAS for constructing a set of approximate Pareto curves (one for every node) for the *single-source* version of the MOSP problem in *any* digraph. For any $d > 1$, our algorithm runs in time $O(nm(\frac{n\log(nC^{max})}{\varepsilon})^{d-1})$ for general digraphs, and in $O(m(\frac{n\log(nC^{max})}{\varepsilon})^{d-1})$ for DAGs. Table 1 summarizes the comparison of our results with the best previous ones. Our results improve significantly upon previous approaches for general digraphs [11, 14] and DAGs [14, 15], for all $d > 2$. For $d = 2$, our running times depend on ε^{-1}; those in [14] are based on repeated applications of a stronger variant of the GAP routine, like a FPTAS for the restricted shortest path (RSP) problem (see e.g., [9], and thus depend on ε^{-2}. Hence, our algorithm gives always better running times for DAGs, while for general digraphs we improve the dependence on $1/\varepsilon$.

Non-linear Objectives. Our second contribution in this work concerns two fundamental problems in multiobjective optimization: (i) Construct a FPTAS for the normalized version of a multiobjective optimization problem when the utility function is *non-linear*. (ii) Construct a FPTAS for a multiobjective optimization problem with *non-linear* objectives.

		Best previous	This work
General digraphs	$d=2$	$O\left(nm\frac{1}{\varepsilon}\log\left(nC^{max}\right)\left(\log\log n+\frac{1}{\varepsilon}\right)\right)$ [14]	$O\left(n^2m\frac{1}{\varepsilon}\log\left(nC^{max}\right)\right)$
	$d>2$	$O((\log(nC^{max})/\varepsilon)^d\cdot T_{GAP})$ [11]	$O\left(nm\left(\frac{n\log(nC^{max})}{\varepsilon}\right)^{d-1}\right)$
DAGs	$d=2$	$O\left(nm\frac{1}{\varepsilon}\log n\log\left(nC^{max}\right)\right)$ [15] $O\left(nm\frac{1}{\varepsilon^2}\log\left(nC^{max}\right)\right)$ [14]	$O\left(nm\frac{1}{\varepsilon}\log\left(nC^{max}\right)\right)$
	$d>2$	$O\left(nm(\frac{n\log(nC^{max})}{\varepsilon})^{d-1}\log^{d-2}(\frac{n}{\varepsilon})\right)$ [15]	$O\left(m\left(\frac{n\log(nC^{max})}{\varepsilon}\right)^{d-1}\right)$

Table 1. Comparison of new and previous results for MOSP. T_{GAP} denotes the time of a GAP routine, which is polynomial in the input and $1/\varepsilon$ (but exponential in d).

An algorithm for the first problem was given in [12] (earlier version of this work) for $d \geq 2$ objectives and polynomial utility function, and independently in [1] for $d = 2$ objectives and quasi-polynomially bounded utility function. Let $T(1/\varepsilon, m')$ denote the time to generate an $(1+\varepsilon)$-Pareto curve for an instance of a multiobjective optimization problem of size m'. The algorithm in [1] provides a FPTAS with time complexity $T(\Lambda_1/\varepsilon^2, m')$, where Λ_1 is polylogarithmic on the maximum cost in any dimension.

We show in Section 4 that we can construct a FPTAS for the normalized version of *any* multiobjective optimization problem with $d \geq 2$ objectives and quasi-polynomially bounded utility function in time $T(\Lambda_2/\varepsilon, m')$, where $\Lambda_2 < \Lambda_1$ is polylogarithmic on the maximum cost in any dimension. Our results are based on a *novel* and simple analysis, and improve upon those in [1] both w.r.t. the running time (better dependence on $1/\varepsilon$ and $\Lambda_2 < \Lambda_1$) and the number of objectives – as well as upon those in [12] w.r.t. the class of utility functions.

The only generic method known for addressing the second problem is that in [11], which assumes the existence of a GAP routine. Such routines for the case of non-linear objectives are not known. The GAP routines given in [11] concern problems with linear objectives only.

We show in Section 4 that a FPTAS for *any* multiobjective optimization problem $\mathcal{M}'$ with quasi-polynomially bounded non-linear objective functions can be constructed from a FPTAS for a much simpler version $\mathcal{M}$ of the problem. $\mathcal{M}$ has the same feasible solution set with $\mathcal{M}'$ and objectives the *identity functions* on the attributes of the non-linear objective functions of $\mathcal{M}'$. In other words, our result suggests that restricting the study of approximate Pareto curves to identity (on the attributes) objectives suffices for treating the non-linear case. Our approach constitutes the first generic method for obtaining FPTAS for any multiobjective optimization problem with quasi-polynomial non-linear objectives.

Applications. The following problems play a key role in several domains.

Multiple Constrained (Optimal) Paths. One of the key issues in networking [10] is how to determine paths that satisfy QoS constraints, a problem known as QoS routing or constraint-based routing. The two most fundamental problems in QoS routing are the *multiple constrained optimal path* (MCOP) and the *multiple constrained path* (MCP) problems (see e.g., [7, 10]). In MCOP, we are given a d-vector of costs $\mathbf{c}$ on the edges and a $(d-1)$-vector $\mathbf{b}$ of QoS-bounds. The objective is to find an s-t path p that minimizes $c_d(p) = \sum_{e\in p} c_d(e)$, and obeys

the QoS-bounds, i.e., $c_i(p) = \sum_{e \in p} c_i(e) \leq b_i, \forall 1 \leq i \leq d-1$. MCOP is NP-hard, even when $d = 2$ in which case it is known as the restricted shortest path problem and admits a FPTAS (see e.g., [9]). In MCP, the objective is to find an s-t path p that simply obeys a d-vector $\mathbf{b}$ of QoS-bounds, i.e., $c_i(p) = \sum_{e \in p} c_i(e) \leq b_i$, $\forall 1 \leq i \leq d$. MCP is NP-complete. For both problems, the case of $d = 2$ objectives has been extensively studied and there are also very efficient FPTAS known (see e.g., [9]). For $d > 2$, apart from the generic approach in [11], only heuristic methods and pseudopolynomial time algorithms are known [10]. We are able to show how (quality guaranteed) approximate schemes to both MCOP and MCP can be constructed that have the same complexity with MOSP, thus improving upon all previous approaches for any $d > 2$.

Non-Additive Shortest Paths. In this problem (NASP), we are given a digraph whose edges are associated with d-dimensional cost vectors and the task is to find a path that minimizes a certain d-attribute non-linear utility function. NASP is a fundamental problem in several domains [5, 6], the most prominent of which is finding traffic equilibria [5]. NASP is an NP-hard problem. By virtue of the results in [1, 12], there exists a FPTAS for $d = 2$ and quasi-polynomial utility function [1], and a FPTAS for any $d \geq 2$ and polynomial utility function [12].

In Section 5, we show how our FPTAS for MOSP, along with our generic framework for dealing with non-linear objectives, can be used to obtain a FPTAS for NASP for *any* $d > 1$ and a larger than quasi-polynomially bounded family of utility functions. Our results improve considerably upon those in [1, 12] w.r.t. time (dependence on $1/\varepsilon$), number of objectives, and class of utility functions.

2 Preliminaries

Recall that an instance of a multiobjective optimization problem is associated with a set of feasible solutions Q and a d-vector function $\mathbf{f} = [f_1, \ldots, f_d]^T$ associating each feasible solution $q \in Q$ with a d-vector $\mathbf{f}(q)$. The *Pareto set or curve* $\mathcal{P}$ of Q is defined as the set of all undominated elements of Q. Given a vector of approximation ratios $\boldsymbol{\rho} = [\rho_1, \ldots, \rho_d]^T$ ($\rho_i \geq 1$, $1 \leq i \leq d$), a solution $p \in Q$ $\boldsymbol{\rho}$*-covers* a solution $q \in Q$ iff it is as good in each objective i by at least a factor ρ_i, i.e., $f_i(p) \leq \rho_i \cdot f_i(q)$, $1 \leq i \leq d$. A set $\Pi \subseteq Q$ is a $\boldsymbol{\rho}$*-cover* of Q iff for all $q \in Q$, there exists $p \in \Pi$ such that p $\boldsymbol{\rho}$-covers q (note that a $\boldsymbol{\rho}$-cover may contain dominated solutions). A $\boldsymbol{\rho}$-cover is also called $\boldsymbol{\rho}$*-Pareto set*. If all entries of $\boldsymbol{\rho}$ are equal to ρ, we also use the terms ρ-cover and ρ-Pareto set.

A *fully polynomial time approximation scheme* (FPTAS) for computing the Pareto set of an instance of a multiobjective optimization problem is a family of algorithms that, for any fixed constant $\varepsilon > 0$, contains an algorithm that always outputs an $(1 + \varepsilon)$-Pareto set and runs in time polynomial in the size of the input and $1/\varepsilon$. W.l.o.g. we make the customary assumption that $\varepsilon \leq 1$, yielding $\ln(1 + \varepsilon) = \Theta(\varepsilon)$, which will be used throughout the paper.

If $\mathbf{a} = [a_1, a_2, \cdots, a_d]^T$ is a d-dimensional vector and λ a scalar, then we denote by $\mathbf{a}^\lambda = [a_1^\lambda, a_2^\lambda, \cdots, a_d^\lambda]^T$. A vector with all its elements equal to zero is denoted by $\mathbf{0}$.

3 Single-Source Multiobjective Shortest Paths

In the multiobjective shortest path problem, we are given a digraph $G = (V, E)$ and a d-dimensional function vector $\mathbf{c} : E \to [\mathbb{R}^+]^d$ associating each edge e with a cost vector $\mathbf{c}(e)$. We extend the cost function vector to handle paths by extending the domain to the powerset of E, thus considering the function $\mathbf{c} : 2^E \to [\mathbb{R}^+]^d$, where the cost vector of a path p is the sum of the cost vectors of its edges, i.e., $\mathbf{c}(p) = \sum_{e \in p} \mathbf{c}(e)$. Given two nodes v and w, let $P(v, w)$ denote the set of all v-w paths in G. In the *multiobjective shortest path* problem, we are asked to compute the Pareto set of $P(v, w)$ w.r.t. $\mathbf{c}$. In the *single-source multiobjective shortest path* (SSMOSP) problem, we are given a node s and the task is to compute the Pareto sets of $P(s, v)$ w.r.t. $\mathbf{c}$, $\forall v \in V$.

Given a vector $\boldsymbol{\varepsilon} = [\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_{d-1}]^T$ of error parameters ($\varepsilon_i > 0$, $1 \leq i \leq d-1$) and a source node s, we present below an algorithm that computes, for each node v, a $\boldsymbol{\rho}$-cover of $P(s, v)$, where $\boldsymbol{\rho} = [1+\varepsilon_1, 1+\varepsilon_2, \cdots, 1+\varepsilon_{d-1}, 1]^T$. Note that we can be *exact* in one dimension (here w.l.o.g. the d-th one), without any impact on the running time. In the following, let $c_i^{min} \equiv \min_{e \in E} c_i(e)$, $c_i^{max} \equiv \max_{e \in E} c_i(e)$, and $C_i = \frac{c_i^{max}}{c_i^{min}}$, for all $1 \leq i \leq d$. Let also $P^i(v, w)$ denote the set of all v-w paths in G with no more than i edges; clearly, $P^{n-1}(v, w) \equiv P(v, w)$.

3.1 The SSMOSP algorithm

Our algorithm resembles the classical (label correcting) Bellman-Ford method. Previous attempts to straightforwardly apply such an approach [2–4] had a very poor (exponential) performance, since all undominated solutions (exponentially large sets of labels) have to be maintained. The key idea of our method is that we can implement the label sets as arrays of polynomial size by relaxing the requirements for strict Pareto optimality to that of $\boldsymbol{\rho}$-covering.

We represent a path $p = (e_1, e_2, \cdots, e_{k-1}, e_k)$ by a label that is a tuple $(\mathbf{c}(p), \text{pred}(p), \text{lastedge}(p))$, where $\mathbf{c}(p) = \sum_{e \in p} \mathbf{c}(e)$ is the d-dimensional cost vector of the path, $\text{pred}(p) = \boldsymbol{q}$ is a pointer to the label of the subpath $q = (e_1, e_2, \cdots, e_{k-1})$ of p, and $\text{lastedge}(p) = e_k$ points to the last edge of p. An empty label is represented by $(\mathbf{0}, null, null)$, while a single edge path has a *null* pred pointer. This representation allows us to retrieve the entire path, without implicitly storing its edges, by following the pred pointers. Let $\mathbf{r} = [r_1, \ldots, r_{d-1}, 1]$ be a vector of approximation ratios. The algorithm proceeds in rounds. In each round i and for each node v the algorithm computes a set of labels Π_v^i, which is an $\mathbf{r}^i$-cover of $P^i(s, v)$. We implement these sets of labels using $(d-1)$-dimensional arrays $\Pi_v^i[0..\lfloor \log_{r_1}(nC_1) \rfloor, 0..\lfloor \log_{r_2}(nC_2) \rfloor, \cdots, 0..\lfloor \log_{r_{d-1}}(nC_{d-1}) \rfloor]$, and index these arrays using $(d-1)$-vectors. This is done by defining a function $\mathbf{pos} : 2^E \to [\mathbb{N}_0]^{d-1}$. For a path p, $\mathbf{pos}(p) = [\lfloor \log_{r_1} \frac{c_1(p)}{c_1^{min}} \rfloor, \lfloor \log_{r_2} \frac{c_2(p)}{c_2^{min}} \rfloor, \cdots, \lfloor \log_{r_{d-1}} \frac{c_{d-1}(p)}{c_{d-1}^{min}} \rfloor]^T$ gives us the position in Π_v^i corresponding to p. The definition of **pos** along with the fact that for any path p we have $c_i(p) \leq (n-1)c_i^{max}$, $\forall 1 \leq i \leq d$, justifies the size of the arrays.

Initially, $\Pi_v^0 = \emptyset$, for all $v \in V - \{s\}$, and Π_s^0 contains only the trivial empty path. For each round $i \geq 1$ and for each node v the algorithm computes Π_v^i as follows. Initially, we set Π_v^i equal to Π_v^{i-1}. We then examine the incoming edges of v, one by one, and perform an *Extend-&-Merge* operation for each edge examined. An *Extend-&-Merge* operation takes as input an edge $e = (u, v)$ and the sets Π_u^{i-1} and Π_v^i. It extends all labels $p \in \Pi_u^{i-1}$ by e, and merges the resulting set of s-v paths with Π_v^i. Since each extension results in a new label (path) $q = (\mathbf{c}(p) + \mathbf{c}(e), \boldsymbol{p}, e)$ whose $\mathbf{pos}(q)$ leads to an array position which may not be empty, the algorithm maintains in each array position the (at most one) path that covers all other paths with the same $\mathbf{pos}(\cdot)$ value, which turns out to be the path with the smallest c_d cost. This keeps the size of the sets polynomially bounded. In particular, q is inserted in the position $\mathbf{pos}(q) = [\lfloor \log_{r_1} \frac{c_1(q)}{c_1^{min}} \rfloor, \lfloor \log_{r_2} \frac{c_2(q)}{c_2^{min}} \rfloor, \cdots, \lfloor \log_{r_{d-1}} \frac{c_{d-1}(q)}{c_{d-1}^{min}} \rfloor]^T$ of Π_v^i, unless this position is already filled in with a label q' for which $c_d(q') \leq c_d(q)$.

The next two lemmas establish the algorithm's correctness and complexity.

Lemma 1. *For all $v \in V$ and for all $i \geq 0$, after the i-th round Π_v^i $\mathbf{r}^i$-covers $P^i(s, v)$.*

Proof. It suffices to prove that for all $p \in P^i(s, v)$, there exists $q \in \Pi_v^i$ such that $c_\ell(q) \leq r_\ell^i c_\ell(p)$, $\forall 1 \leq \ell \leq d$. We prove this by induction.

For the basis of the induction ($i = 1$) consider a single edge path $p \equiv (e) \in P^1(s, v)$. At each round all incoming edges of v are examined and an *Extend-&-Merge* operation is executed for each edge. After the first round and due to the **if** condition of the *Extend-&-Merge* operation, position $\mathbf{pos}(p)$ of Π_v^1 contains a path q for which: (i) $\mathbf{pos}(q) = \mathbf{pos}(p)$; and (ii) $c_d(q) \leq c_d(p)$. From (i) it is clear that for all $1 \leq \ell \leq d-1$, we have $\lfloor \log_{r_\ell} \frac{c_\ell(q)}{c_\ell^{min}} \rfloor = \lfloor \log_{r_\ell} \frac{c_\ell(p)}{c_\ell^{min}} \rfloor$, and therefore $\log_{r_\ell} \frac{c_\ell(q)}{c_\ell^{min}} - 1 \leq \log_{r_\ell} \frac{c_\ell(p)}{c_\ell^{min}}$. This, along with (ii) and the fact that $r_d = 1$, implies that $c_\ell(q) \leq r_\ell c_\ell(p)$, $\forall 1 \leq \ell \leq d$.

For the induction step consider a path $p \equiv (e_1, e_2, \ldots, e_k = (u, v)) \in P^i(s, v)$, for some $k \leq i$. The subpath $p' \equiv (e_1, e_2, \ldots, e_{k-1})$ of p has at most $i - 1$ edges and applying the induction hypothesis we get that there exists a path $q' \in \Pi_u^{i-1}$ such that $c_\ell(q') \leq r_\ell^{i-1} c_\ell(p')$, $1 \leq \ell \leq d$. Let now $\bar{q}$ be the concatenation of q' with edge e_k. Then, we have:

$$c_\ell(\bar{q}) \leq r_\ell^{i-1} c_\ell(p), \quad 1 \leq \ell \leq d \tag{1}$$

It is clear by our algorithm that during the *Extend-&-Merge* operation for edge e_k in the i-th round $\bar{q}$ was examined. Moreover, at the end of the i-th round and due to the **if** condition of the *Extend-&-Merge* operation, position $\mathbf{pos}(\bar{q})$ of Π_v^i contains a path q for which: (iii) $\mathbf{pos}(q) = \mathbf{pos}(\bar{q})$; and (iv) $c_d(q) \leq c_d(\bar{q})$. From (iii) it is clear that $\lfloor \log_{r_\ell} c_\ell(q) \rfloor = \lfloor \log_{r_\ell} c_\ell(\bar{q}) \rfloor$, $\forall\ 1 \leq \ell \leq d-1$, and therefore $\log_{r_\ell} c_\ell(q) - 1 \leq \log_{r_\ell} c_\ell(\bar{q})$, $\forall\ 1 \leq \ell \leq d-1$, which implies that

$$c_\ell(q) \leq r_\ell c_\ell(\bar{q}), \quad 1 \leq \ell \leq d-1. \tag{2}$$

Since $r_d = 1$, combining now (iv) and (2) with (1), we get that $c_\ell(q) \leq r_\ell^i c_\ell(p)$, $\forall\ 1 \leq \ell \leq d$. □

Lemma 2. *Algorithm SSMOSP computes, for all $v \in V$, an $\mathbf{r}^{n-1}$-cover of $P(s,v)$ in total time $O(nm\prod_{j=1}^{d-1}(\lfloor \log_{r_j}(nC_j)\rfloor + 1))$.*

Proof. From Lemma 1, it is clear that, for any $v \in V$, Π_v^{n-1} is an $\mathbf{r}^{n-1}$-cover of $P^{n-1}(s,v) \equiv P(s,v)$, since any path has at most $n-1$ edges. The algorithm terminates after $n-1$ rounds. In each round it examines all of the m edges and performs an *Extend-&-Merge* operation. The time of this operation is proportional to the size of the arrays used, which equals $\prod_{j=1}^{d-1}(\lfloor \log_{r_j}(nC_j)\rfloor + 1)$ and therefore the total time complexity is $O(nm\prod_{j=1}^{d-1}(\lfloor \log_{r_j}(nC_j)\rfloor + 1))$. □

Applying Lemma 2 with $\mathbf{r} = [(1+\varepsilon_1)^{\frac{1}{n-1}}, (1+\varepsilon_2)^{\frac{1}{n-1}}, \cdots, (1+\varepsilon_{d-1})^{\frac{1}{n-1}}, 1]$, and taking into account that $\ln(1+\delta) = \Theta(\delta)$ for small δ, yields our main result.

Theorem 1. *Given a vector $\boldsymbol{\varepsilon} = [\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_{d-1}]^T$ of error parameters and a source node s, there exists an algorithm that computes, for all $v \in V$, a $\boldsymbol{\rho}$-cover of $P(s,v)$ (set of all s-v paths), where $\boldsymbol{\rho} = [1+\varepsilon_1, 1+\varepsilon_2, \cdots, 1+\varepsilon_{d-1}, 1]^T$, in total time $O(n^d m\prod_{j=1}^{d-1}(\frac{1}{\varepsilon_j}\log(nC_j)))$.*

Let $C^{max} = \max_{1\le j\le d-1} C_j$. In the special case, where $\varepsilon_i = \varepsilon, \forall 1 \le i \le d-1$, we have the following result.

Corollary 1. *For any error parameter $\varepsilon > 0$, there exists a FPTAS for the single-source multiobjective shortest path problem with d objectives on a digraph G that computes $(1+\varepsilon)$-Pareto sets (one for each node of G) in total time $O(nm(\frac{n\log(nC^{max})}{\varepsilon})^{d-1})$.*

Further improvements can be obtained in the case of DAGs; see [13].

4 Non-linear Objectives

In this section, we present two generic methods to construct a FPTAS for the normalized version of any multiobjective optimization problem with a non-linear utility function, as well as a FPTAS for any multiobjective optimization problem with non-linear objectives, for a quite general family of non-linear functions. The only precondition is the existence of a FPTAS for a much simpler version of the problems.

Let $\mathcal{M}$ be (an instance of) a multiobjective optimization problem with set of feasible solutions Q and vector of objective functions $\mathbf{c} = [c_1, \ldots, c_d]^T$, associating each feasible solution $q \in Q$ with a d-vector of attributes $\mathbf{c}(q)$; i.e., the i-th objective is the identity function of the i-th attribute.

Let $\mathcal{N}$ be the normalized version of $\mathcal{M}$ w.r.t. a non-decreasing, non-linear utility function $\mathcal{U} : [\mathbb{R}^+]^d \to \mathbb{R}$; i.e., the objective of $\mathcal{N}$ is $\min_{q\in Q}\mathcal{U}(\mathbf{c}(q))$. We will show that a FPTAS for $\mathcal{M}$ can provide a FPTAS for $\mathcal{N}$. To obtain such a FPTAS, we consider a quite general family of non-linear functions $\mathcal{U}(\mathbf{x})$.

A multiattribute function $\mathcal{U}(\mathbf{x})$ is called *quasi-polynomially bounded* (see e.g., [1]) if there exist some constants γ and δ such that $\frac{\frac{\partial \mathcal{U}}{\partial x_i}(\mathbf{x})}{\mathcal{U}(\mathbf{x})} \le \gamma\frac{1}{x_i}\prod_{k=1}^{d}\ln^{\delta} x_k$,

$1 \leq i \leq d$. For instance, the function $\mathcal{U}([x_1, x_2]^T) = x_1^{polylog(x_1)} + x_2^{polylog(x_2)}$ is quasi-polynomially bounded, while the function $\mathcal{U}([x_1, x_2]^T) = 2^{x_1^\mu} + 2^{x_2^\mu}$, for some $\mu > 0$, is not. Note also that this class includes all non-decreasing polynomials.

Let $\mathcal{C}_i = \max_{q \in Q} c_i(q)$ be the maximum cost in the i-th dimension, and let $\log \mathcal{C}_i$ be polynomial to the input size (as indeed is the case for MOSP and other problems, like the multiobjective versions of spanning tree, perfect matching, knapsack, etc). We can prove the following.

Theorem 2. *Let the objective function $\mathcal{U}$ of $\mathcal{N}$ be quasi-polynomially bounded. If there exists a FPTAS for $\mathcal{M}$ with time complexity $T(1/\varepsilon, m')$, then there exists a FPTAS for $\mathcal{N}$ with complexity $T(\Lambda/\varepsilon, m')$, where m' is the input size of $\mathcal{M}$ and $\Lambda = \gamma d \prod_{i=1}^{d} \ln^{\delta} \mathcal{C}_i$.*

Proof. We construct an $(1+\varepsilon')$-Pareto set Π for $\mathcal{M}$, where ε' will be chosen later. Pick $q = \text{argmin}_{p \in \Pi}(\mathcal{U}(\mathbf{c}(p)))$. Let p^* denote the optimal solution with cost vector $\boldsymbol{c}^* = \boldsymbol{c}(p^*)$. By the definition of Π, we know that there exists some $p' \in \Pi$ such that $c_i(p') \leq \min\{(1+\varepsilon')c_i^*, \mathcal{C}_i\}$. By the choice of q we have that $\mathcal{U}(\mathbf{c}(q)) \leq \mathcal{U}(\mathbf{c}(p'))$, thus it suffices to bound $\frac{\mathcal{U}(\boldsymbol{c}(p'))}{\mathcal{U}(\boldsymbol{c}(p^*))}$.

Let $\boldsymbol{c'}$ be the vector whose elements are given by $c_i' = \min\{(1+\varepsilon')c_i^*, \mathcal{C}_i\}, \forall 1 \leq i \leq d$. Since $\mathcal{U}(\cdot)$ is non-decreasing, $\frac{\mathcal{U}(\boldsymbol{c}(p'))}{\mathcal{U}(\boldsymbol{c}(p^*))} \leq \frac{\mathcal{U}(\boldsymbol{c'})}{\mathcal{U}(\boldsymbol{c}^*)} = \exp\left[\ln \mathcal{U}(\boldsymbol{c'}) - \ln \mathcal{U}(\boldsymbol{c}^*)\right]$. We write the exponent as a telescopic sum $\ln \mathcal{U}(\boldsymbol{c'}) - \ln \mathcal{U}(\boldsymbol{c}^*) = \sum_{k=1}^{d} \left[F_k(c_k') - F_k(c_k^*)\right]$, where $F_k(x) = \ln \mathcal{U}([c_1', \ldots, c_{k-1}', x, c_{k+1}^*, \ldots, c_d^*]^T)$. On each term k of the sum, we apply the well-known Mean Value Theorem[1] for $F_k(x)$ on the interval (c_k^*, c_k'). Hence, $\forall 1 \leq k \leq d$, there exists some ζ_k with $c_k^* < \zeta_k < c_k'$ such that $F_k(c_k') - F_k(c_k^*) = F_k'(\zeta_k)(c_k' - c_k^*) \leq \frac{\frac{\partial \mathcal{U}}{\partial x_k}(\boldsymbol{c}^{[k]})}{\mathcal{U}(\boldsymbol{c}^{[k]})} \varepsilon' c_k^*$, where $\boldsymbol{c}^{[k]}$ are vectors with $c_i^{[k]} = \begin{cases} c_i' \text{ if } 1 \leq i < k \\ \zeta_k \text{ if } i = k \\ c_i^* \text{ if } k < i \leq d \end{cases}$. Consequently, $\frac{\mathcal{U}(\boldsymbol{c'})}{\mathcal{U}(\boldsymbol{c}^*)} \leq \exp\left[\varepsilon' \sum_{k=1}^{d} \left[\frac{\frac{\partial \mathcal{U}}{\partial x_k}(\boldsymbol{c}^{[k]})}{\mathcal{U}(\boldsymbol{c}^{[k]})} c_k^*\right]\right]$. Observe now that the term $\sum_{k=1}^{d} \left[\frac{\frac{\partial \mathcal{U}}{\partial x_k}(\boldsymbol{c}^{[k]})}{\mathcal{U}(\boldsymbol{c}^{[k]})} c_k^*\right]$ is bounded by $\Lambda = \gamma d \prod_{i=1}^{d} \ln^{\delta} \mathcal{C}_i$. Hence, choosing $\varepsilon' = \frac{\ln(1+\varepsilon)}{\Lambda}$, yields an $1+\varepsilon$ approximation in time $T(\Lambda/\varepsilon, m')$. □

The above result improves upon that of [1] both w.r.t. d (number of objectives) and time; the time in [1] ($d = 2$) is $T(\Lambda'/\varepsilon^2, m')$, where $\Lambda' = \gamma 2^{\delta+4} \prod_{i=1}^{2} \ln^{\delta+1} \mathcal{C}_i$.

Now, let $\mathcal{M}'$ be a multiobjective optimization problem, defined on the same with $\mathcal{M}$ set of feasible solutions Q, but having a vector of objective functions $\mathbf{U} = [U_1, \ldots, U_h]^T$ associating each $q \in Q$ with an h-vector $\mathbf{U}(q)$. These objective functions are defined as $U_i(q) = \mathcal{U}_i(\mathbf{c}(q))$, $1 \leq i \leq h$, where $\mathcal{U}_i : [\mathbb{R}^+]^d \to \mathbb{R}$ are non-linear, non-decreasing, quasi-polynomially bounded functions. By working similarly to Theorem 2, we can show the following (details in [13]).

[1] **Mean Value Theorem:** Let $f(x)$ be differentiable on (a, b) and continuous on $[a, b]$. Then, there is at least one point $c \in (a, b)$ such that $f'(c) = (f(b) - f(a))/(b - a)$.

Theorem 3. *Let the objective functions of $\mathcal{M}'$ be quasi-polynomially bounded. If there exists a FPTAS for $\mathcal{M}$ with time complexity $T(1/\varepsilon, m')$, then there exists a FPTAS for $\mathcal{M}'$ with complexity $T(\Lambda/\varepsilon, m')$, where m' is the input size of $\mathcal{M}$ and $\Lambda = \gamma d \prod_{i=1}^{d} \ln^{\delta} \mathcal{C}_i$.*

5 Applications

Multiple Constrained (Optimal) Paths. Let $\boldsymbol{\rho} = [1+\varepsilon_1, 1+\varepsilon_2, \cdots, 1+\varepsilon_{d-1}, 1]^T$ and let Π be a $\boldsymbol{\rho}$-cover Π of $P(s,t)$, constructed using the SSMOSP algorithm as implied by Theorem 1. For MCOP, choose $p' = \text{argmin}_{p\in\Pi}\{c_d(p); c_i(p) \leq (1+\varepsilon_i)b_i,\ \forall 1 \leq i \leq d-1\}$. This provides a so-called *acceptable* solution in the sense of [7] by slightly relaxing the QoS-bounds; that is, the path p' is at least as good as the MCOP-optimum and is nearly feasible, violating each QoS-bound b_i, $1 \leq i \leq d-1$, by at most an $1+\varepsilon_i$ factor. For MCP, choose a path $p' \in \Pi$ that obeys the QoS-bounds, or answer that there is no path p in $P(s,t)$ for which $c_i(p) \leq b_i/(1+\varepsilon_i)$, $\forall 1 \leq i < d$. In the latter case, if a feasible solution for MCP exists, then (by the definition of Π) we can find a solution in Π that is nearly feasible (i.e., it violates each QoS-bound b_i, $1 \leq i \leq d-1$, by at most an $1+\varepsilon_i$ factor). By Theorem 1, the required time for both cases is $O(n^d m \prod_{j=1}^{d-1}(\frac{1}{\varepsilon_j}\log(nC_j))$, which can be reduced to $O(n^d m \prod_{j=1}^{d-1}(\frac{1}{\varepsilon_j}\log(\min\{nC_j, b_j/c_j^{min}\}))$ by observing that it is safe to discard any path p for which $c_j(p) > (1+\varepsilon_j)b_j$ for some $1 \leq j \leq d-1$ (thus reducing the size of the Π_v^i arrays).

Non-Additive Shortest Paths. In this problem (NASP) we are given a digraph $G = (V, E)$ and a d-dimensional function vector $\mathbf{c} : E \to [\mathbb{R}^+]^d$ associating each edge e with a vector of attributes $\mathbf{c}(e)$ and a path p with a vector of attributes $\mathbf{c}(p) = \sum_{e\in p} \mathbf{c}(e)$. We are also given a d-attribute non-decreasing and *non-linear* utility function $\mathcal{U} : [\mathbb{R}^+]^d \to \mathbb{R}$. The objective is to find a path p^*, from a specific source node s to a destination t, that minimizes the objective function, i.e., $p^* = \text{argmin}_{p\in P(s,t)}\mathcal{U}(\mathbf{c}(p))$. (It is easy to see that in the case where $\mathcal{U}$ is linear, NASP reduces to the classical single-objective shortest path problem.) For the general case of non-linear $\mathcal{U}$, it is not difficult to see that NASP is NP-hard.

Theorem 2 suggests that our FPTAS for MOSP yields an (improved w.r.t. [1, 12]) FPTAS for NASP for the case of quasi-polynomially bounded functions. We show that we can do better by taking advantage of the fact that our FPTAS for MOSP is *exact* in one dimension (w.l.o.g. the d-th). This allows us to provide a FPTAS for an even more general (than quasi-polynomial) family of functions. Specifically, we consider d-attribute functions for which there exist some constants γ and δ such that $\frac{\frac{\partial \mathcal{U}}{\partial x_i}(\mathbf{x})}{\mathcal{U}(\mathbf{x})} \leq \gamma \frac{1}{x_i}\prod_{k=1}^{d}\ln^{\delta} x_k$, $1 \leq i \leq d-1$. The fact that we do not require that this condition holds for the d-th attribute allows $\mathcal{U}$ to be even *exponential* on x_d; e.g., $\mathcal{U}([x_1, x_2]^T) = x_1^{polylog(x_1)} + 2^{x_2^{\mu}}$, for any $\mu > 0$. Note that this does not contradict the inapproximability result in [1], which applies to functions of the form $\mathcal{U}([x_1, x_2]^T) = 2^{x_1^{\mu}} + 2^{x_2^{\mu}}$, for $\mu > 0$.

Our result makes the gap between NASP approximability and inapproximability even tighter. Let $\mathcal{C}_i$ denote the maximum path cost in the i-th dimension, i.e., $\mathcal{C}_i = (n-1)\max_{e \in E} c_i(e)$. We can show the following (see [13]).

Theorem 4. *Let $\mathcal{U}$ be a non-decreasing function for which $\frac{\frac{\partial \mathcal{U}}{\partial x_i}(\mathbf{x})}{\mathcal{U}(\mathbf{x})} \leq \gamma \frac{1}{x_i} \prod_{k=1}^{d} \ln^{\delta} x_k$, $1 \leq i \leq d-1$. Then, for any $\varepsilon > 0$, there exists an algorithm that computes in time $O(n^d m (\frac{\log(nC^{max})\Lambda}{\varepsilon})^{d-1})$ an $(1+\varepsilon)$-approximation to the NASP optimum w.r.t. $\mathcal{U}(\mathbf{x})$, where $\Lambda = \gamma(d-1)\prod_{i=1}^{d} \ln^{\delta} \mathcal{C}_i$.*

References

1. H. Ackermann, A. Newman, H. Röglin, and B. Vöcking, "Decision Making Based on Approximate and Smoothed Pareto Curves", in *Algorithms and Computation* – ISAAC 2005, LNCS Vol. 3827 (Springer 2006), pp. 675-684; full version as Tech. Report AIB-2005-23, RWTH Aachen, December 2005.
2. H. Corley and I. Moon, "Shortest Paths in Networks with Vector Weights", *Journal of Optimization Theory and Applications*, 46:1(1985), pp. 79-86.
3. M. Ehrgott, *Multicriteria Optimization*, Springer, 2000.
4. M. Ehrgott and X. Gandibleux (Eds), *Multiple Criteria Optimization – state of the art annotated bibliographic surveys*, Kluwer Academic Publishers, Boston, 2002.
5. S. Gabriel and D. Bernstein, "The Traffic Equilibrium Problem with Nonadditive Path Costs", *Transportation Science* 31:4(1997), pp. 337-348.
6. S. Gabriel and D. Bernstein, "Nonadditive Shortest Paths: Subproblems in Multi-Agent Competitive Network Models", *Computational & Mathematical Organization Theory* 6(2000), pp. 29-45.
7. A. Goel, K. G. Ramakrishnan, D. Kataria, and D. Logothetis, "Efficient Computation of Delay-Sensitive Routes from One Source to All Destinations", in *Proc. IEEE Conf. Comput. Commun.* – INFOCOM 2001.
8. P. Hansen, "Bicriterion Path Problems", *Proc. 3rd Conf. Multiple Criteria Decision Making – Theory and Applications*, LNEMS Vol. 117 (Springer, 1979), pp. 109-127.
9. D.H. Lorenz and D. Raz, "A simple efficient approximation scheme for the restricted shortest path problem", *Operations Res. Lett.*, 28 (2001) pp.213-219.
10. P. Van Mieghem, F.A. Kuipers, T. Korkmaz, M. Krunz, M. Curado, E. Monteiro, X. Masip-Bruin, J. Sole-Pareta, and S. Sanchez-Lopez, "Quality of Service Routing", Chapter 3 in *Quality of Future Internet Services*, LNCS Vol. 2856 (Springer-Verlag, 2003), pp. 80-117.
11. C. Papadimitriou and M. Yannakakis, "On the Approximability of Trade-offs and Optimal Access of Web Sources", in *Proc. 41st Symp. on Foundations of Computer Science* – FOCS 2000, pp. 86-92.
12. G. Tsaggouris and C. Zaroliagis, "Improved FPTAS for Multiobjective Shortest Paths with Applications", CTI Techn. Report TR-2005/07/03, July 2005.
13. G. Tsaggouris and C. Zaroliagis, "Multiobjective Optimization: Improved FPTAS for Shortest Paths and Non-linear Objectives with Applications", CTI Techn. Report TR-2006/03/01, March 2006.
14. S. Vassilvitskii and M. Yannakakis, "Efficiently Computing Succinct Trade-off Curves", in *Automata, Languages, and Programming* – ICALP 2004, LNCS Vol. 3142 (Springer, 2004), pp. 1201-1213.
15. A. Warburton, "Approximation of Pareto Optima in Multiple-Objective Shortest Path Problems", *Operations Research* 35(1987), pp. 70-79.