Variants of the Shortest Path Problem

Lara Turner[*]

Abstract

The shortest path problem in which the (s,t)-paths P of a given digraph $G = (V, E)$ are compared with respect to the sum of their edge costs is one of the best known problems in combinatorial optimization. The paper is concerned with a number of variations of this problem having different objective functions like bottleneck, balanced, minimum deviation, algebraic sum, k-sum and k-max objectives, (k_1, k_2)-max, (k_1, k_2)-balanced and several types of trimmed-mean objectives. We give a survey on existing algorithms and propose a general model for those problems not yet treated in literature. The latter is based on the solution of resource constrained shortest path problems with equality constraints which can be solved in pseudo-polynomial time if the given graph is acyclic and the number of resources is fixed. In our setting, however, these problems can be solved in strongly polynomial time. Combining this with known results on k-sum and k-max optimization for general combinatorial problems, we obtain strongly polynomial algorithms for a variety of path problems on acyclic and general digraphs.

Keywords: Shortest path problem, universal objective function, resource constrained shortest path problem, strongly polynomial-time algorithm.

1 Introduction

Shortest path problems (SPPs) are classical problems in combinatorial optimization with various applications in theory and practice. Given a directed graph $G = (V, E)$ with node set V of cardinality n, edge set E of cardinality m and costs $c(e) \in \mathbb{R}$ for all edges $e \in E$, the single-source single-sink version

[*] Department of Mathematics, University of Kaiserslautern, P. O. Box 3049, 67653 Kaiserslautern, Germany. Email address: turner@mathematik.uni-kl.de.

of the *sum shortest path problem (sum SPP)* finds a path from source s to sink t which minimizes the sum of the edge costs, i.e.

$$\min_{P \in \mathcal{P}_{st}} \sum_{e \in P} c(e) \tag{1}$$

where $\mathcal{P}_{st}$ is the set of all elementary (s,t)-paths defined as sequences $P = (s = i_0, e_1, i_1, \ldots, i_{l(P)-1}, e_{l(P)}, i_{l(P)} = t)$ of nodes $i_k \in V$ and edges $e_k = (i_{k-1}, i_k) \in E$ with the property that no nodes (and thus no edges) are repeated. As usual, the length $l(P)$ denotes the number of edges in path P. It is well-known that sum SPP is NP-hard, but can be solved in polynomial time if there are no negative dicycles (paths with the same start- and endnode and negative costs) in graph G. The currently best strongly polynomial-time algorithms are the label-setting algorithm of Dijkstra in its Fibonacci heap implementation (for non-negative costs) and the label-correcting algorithm of Bellman and Ford (for arbitrary costs) which have complexity $\mathcal{O}(m+n\log n)$ and $\mathcal{O}(nm)$, respectively. These algorithms can, for instance, be found in the books of Ahuja et al. [1] and Schrijver [33]. For recent surveys on shortest path algorithms we refer the reader to Zwick [42] and Festa [8].

In this paper, we study several variants of the shortest path problem with source s and sink t. These problems have the same feasible set as the classical sum SPP, but the sum objective is replaced by another criteria like minimizing the largest edge cost (bottleneck SPP) or the difference between the largest and smallest edge cost (balanced SPP). Throughout the paper, we only focus on objective functions which arise as special cases of the more general *universal shortest path problem (Univ-SPP)* introduced in [35]. In its sequential definition, this problem is solved as a sequence of $n-1$ cardinality constrained path problems, Univ-SPP(l),

$$\min_{P \in \mathcal{P}_{st}:\, l(P)=l} f_{\lambda^l}(P) = \sum_{i=1}^{l} \lambda_i^l c_{(i)}(P) \tag{2}$$

where $c_{(i)}(P)$ is the i^{th}-largest edge cost in path P, $\lambda_i^l \in \mathbb{R}$, $i = 1, \ldots, l$, are universal weight coefficients and $l \in \{1, \ldots, n-1\}$. Besides the shortest path problem with sum objective for which we set $\lambda_i^l = 1$ for all $i = 1, \ldots, l$, the bottleneck and balanced shortest path problems can be formulated in this setting with $\lambda_1^l = 1$ and $\lambda_i^l = 0$ else for bottleneck SPP, $\lambda_1^l = 1$, $\lambda_l^l = -1$ and $\lambda_i^l = 0$ else for balanced SPP with length $l \neq 1$ and $\lambda_1^1 = 0$ for balanced SPP with length $l = 1$. Many other problems can be modelled as Univ-SPPs which illustrates the potential of the universal approach. These problems are addressed in Sections 2 and 4 and include, among others, k-sum and k-max, (k_1, k_2)-max and (k_1, k_2)-balanced as well as trimmed-mean shortest path problems.

The remainder of the paper is organized as follows. In Section 2 we review shortest path problems dealt with in literature. We consider special algorithms which have been designed for path problems only and general algorithms for arbitrary combinatorial optimization problems which can be applied to path problems. New variants of shortest path problems on acyclic and general digraphs are discussed in Section 4. The idea to tackle these problems is to fix one or several edges as k^{th}-largest or k^{th}-smallest cost edges of the feasible paths and to solve the resulting constrained path problem. This resource constrained shortest path problem (with sum objective and equality constraints) is analysed in Section 3 and we show that two pseudo-polynomial dynamic programming algorithms which solve the problem on acyclic graphs with a fixed number of resources are strongly polynomial in our case. As a consequence, the path problems of Section 4 are solvable in strongly polynomial time, too. A summary of our results will be given in Section 5.

2 Shortest Path Problems in Literature

Using appropriate weight coefficients λ_i^l, all path problems considered in this paper can be modelled as universal shortest path problems. Although the latter problem is in general NP-hard (it contains as special case the classical sum SPP with cardinality constraints), polynomial-time algorithms for many objective functions can be obtained by solving the unconstrained version of the problem.

This holds for the classical sum shortest path problem (see Section 1) and the *bottleneck shortest path problem* which minimizes the largest of the edge costs in path P:

$$\min_{P \in \mathcal{P}_{st}} \max_{e \in P} c(e). \tag{3}$$

This problem which is also known as *maximum capacity path problem* can be solved by modifying the algorithms for sum SPP, see e.g. Pollack [27]. So, Dijkstra's algorithm with Fibonacci heaps yields a complexity of $\mathcal{O}(m+n \log n)$. Another strongly polynomial-time algorithm for bottleneck SPP is obtained by applying the binary search version of the standard threshold algorithm of Edmonds and Fulkerson [6] which consists of solving $\log m$ many feasibility problems. Using breadth-first search to determine in $\mathcal{O}(n+m)$ time if there is a path from s to t using only edges with costs less than a given threshold (see e.g. Krumke and Noltemeier [22]), it has a performance of $\mathcal{O}(n \log m + m \log m) = \mathcal{O}(n \log n + m \log n)$. This holds because the number of edges in digraph $G = (V, E)$ is less than n^2, i.e. $\mathcal{O}(\log m) = \mathcal{O}(\log n)$. Hu [18] developed an algorithm to find the subset of edges containing the bottleneck paths between all pairs of nodes and Gabow and Tarjan [9] established an algorithm for the single-source single-sink case derived from their algorithm for bottleneck directed spanning trees. The latter algorithm has a complexity of $\mathcal{O}(\min\{m + n \log n, m \log^* n\})$ where $\log^* n := \min\{i : \log^{(i)} n \leq 1\}$ and $\log^{(i)} n$ is iteratively defined as $\log^{(0)} n := n$ and $\log^{(i+1)} n := \log \log^{(i)} n$ else. Recently, bottleneck shortest path problems have been addressed by Kaibel and Peinhardt [21].

A related problem is the *balanced shortest path problem* in which the difference between the largest and smallest edge cost is minimized:

$$\min_{P \in \mathcal{P}_{st}} \left(\max_{e \in P} c(e) - \min_{e \in P} c(e) \right). \tag{4}$$

We use the algorithm of Martello et al. [24] which has been developed for balanced combinatorial optimization problems. It is similar to the threshold algorithm of [6] and solves a sequence of at most m feasibility problems such that a balanced shortest path can be found in $\mathcal{O}(nm + m^2)$ time. Duin and Volgenant [4] suggested a unified approach to tackle balanced and minimum deviation problems simultaneously. For general combinatorial problems where all feasible solutions have the same cardinality minimum deviation problems were introduced by Gupta and Punnen [16]. The *minimum deviation shortest path problem* is defined as

$$\min_{P \in \mathcal{P}_{st}} \sum_{e_i \in P} \left(\max_{e \in P} c(e) - c(e_i) \right) \tag{5}$$

where all paths $P \in \mathcal{P}_{st}$ have the same length $l(P)$. Solving optimization instead of feasibility problems, the algorithms of [16] and [4] applied to minimum deviation SPP have a running time of $\mathcal{O}(mT)$ where T is the time needed to solve a sum SPP with modified (possibly negative) costs. Using the algorithms of Dijkstra or Bellman-Ford to compute these shortest paths (if there are no negative-cost cycles), this is $\mathcal{O}(m^2 + nm \log n)$ or $\mathcal{O}(nm^2)$. Relaxing the assumption of fixed path lengths, the problem can be formulated in terms of Univ-SPP setting $\lambda_1^l = l - 1$ and $\lambda_i^l = -1$ else for $l \neq 1$ and $\lambda_1^1 = 0$ for $l = 1$. On acyclic directed graphs, where the cardinality-constrained shortest path problem can be solved efficiently for all $l = 1, \ldots, n-1$, it is thus solvable in strongly polynomial time.

The same time complexity is attained for problems with combined min-max min-sum objective function. Such "algebraic sum" problems with a bottleneck and sum objective function (that are usually based on different cost functions) were considered by Minoux [25] and Punnen [28]. The solution algorithms are similar to those for balanced and minimum deviation problems. If there is a single cost function for the bottleneck and sum objective, the algebraic sum version of SPP, the *algebraic sum shortest path problem*, is

$$\min_{P \in \mathcal{P}_{st}} \left(\max_{e \in P} c(e) + \sum_{e \in P} c(e) \right) \tag{6}$$

and can be modelled as Univ-SPP with $\lambda_1^l = 2$ and $\lambda_i^l = 1$ else for $l \neq 1$ and $\lambda_1^1 = 2$ for $l = 1$.

In literature, there exist algorithms for two other types of shortest path problems, the k-sum SPP and the k-max SPP, in which we minimize the sum of the k largest edge costs or the k^{th}-largest edge cost, respectively. These path problems will be studied in more detail since we will need them in Section 4 where new variants of shortest path problems, not yet treated in literature, will be discussed.

k-sum optimization problems have been investigated in Gupta and Punnen [17] and Punnen and Aneja [29] for general combinatorial optimization problems and in Garfinkel et al. [12] for shortest path problems. In [12], the authors introduced two versions of the k-sum SPP, here called k-centrum SPP, which are both shown to be NP-hard: One version in which all paths $P \in \mathcal{P}_{st}$ of length $l(P) < k$ are assumed to be infeasible and another more flexible one which accepts (s,t)-paths P of length less than k and assigns to them the sum of their edge costs as objective value. In case that all paths from s to t have length at least k, they have proposed a strongly polynomial-time algorithm of complexity $\mathcal{O}(n^2m^2)$ for the first version of k-centrum SPP. This algorithm is recursion-based and determines a k-centrum shortest walk (i.e. a non-elementary or non-simple path allowing repetition of nodes or edges) which is reducible to a k-centrum shortest path. If there are no negative dicycles in graph G, the algorithm can be modified to solve the second version of k-centrum SPP, too. Another approach to solve k-sum SPP is to adapt the algorithm of Punnen and Aneja [29] for general combinatorial problems with k-sum objective. In this setting, the cost coefficients are assumed to be non-negative and the k-sum objective value is assumed to be the ordinary sum if a solution has less than k elements. Using this definition, the *k-sum shortest path problem* is

$$\min_{P \in \mathcal{P}_{st}} \sum_{i=1}^{\min\{k,l(P)\}} c_{(i)}(P) \tag{7}$$

where $l(P)$ is the length of path P and $c_{(i)}(P)$ denotes its i^{th}-largest edge cost. As such, k-sum SPP fits into the framework of Univ-SPP choosing $\lambda_1^l = \ldots = \lambda_k^l = 1$ and $\lambda_i^l = 0$ else if $l > k$ and $\lambda_i^l = 1$ for all $i = 1, \ldots, l$ if $l \leq k$. In the algorithm given by [29], the set $\mathcal{P}_{st}$ is partitioned into sets

$$\mathcal{P}_{st}(c_i) := \{P \in \mathcal{P}_{st} : c_{(k)}(P) = c_i\} \tag{8}$$

of those (s,t)-paths P with cost c_i as k^{th}-largest edge cost and the set

$$\mathcal{P}_{st}(0) := \{P \in \mathcal{P}_{st} : l(P) < k\} \cup \{P \in \mathcal{P}_{st} : c_{(k)}(P) = 0\} \tag{9}$$

containing all (s,t)-paths P with length $l(P) < k$ and the (s,t)-paths P with cost 0 as k^{th}-largest edge cost (if $c(e) = 0$ for some edge $e \in E$). Sorting the values c_i in $\{c(e) : e \in E\} \cup \{0\}$ in increasing order, a k-sum shortest path can be found among the optimal paths in $\mathcal{P}_{st}(c_i)$. For each $\mathcal{P}_{st}(c_i)$, $c_i \neq 0$, such an optimal path is obtained by solving a sum SPP with edge costs

$$c_{c_i}(e) := \begin{cases} c(e) - c_i & \text{if } c(e) \geq c_i \\ 0 & \text{if } c(e) < c_i \end{cases} \tag{10}$$

and an optimal path in $\mathcal{P}_{st}(0)$, - or a path in $\mathcal{P}_{st}(c_i)$ with smaller objective value than all paths in $\mathcal{P}_{st}(0)$ - , is obtained by solving a sum SPP with costs $c(e)$. Since a k-sum shortest path with less than k edges can alternatively be computed by solving a resource constrained shortest path problem (see Section 3), the algorithm is still correct on digraphs without negative-cost cycles. Using a different cost modification scheme

$$c_{c_i}(e) := \begin{cases} c(e) & \text{if } c(e) > c_i \\ c_i & \text{if } c(e) \leq c_i \end{cases} \tag{11}$$

a similar algorithm was proposed in a preceding paper of Gupta and Punnen [17]. However, this is only valid if all (s,t)-paths P have the same length $l(P)$. Both algorithms terminate in $\mathcal{O}(m^2 + mT)$ time which is $\mathcal{O}(m^2 + nm \log n)$ or $\mathcal{O}(nm^2)$ if the algorithms of Dijkstra or Bellman-Ford are applied to solve the corresponding sum SPPs of complexity $\mathcal{O}(T)$.

The k-max objective function which generalizes the bottleneck objective function by minimizing the k^{th}-largest cost coefficient was considered in Gorski and Ruzika [15]. They presented a bisection search algorithm which is applicable to general combinatorial optimization problems and solve the problem in (strongly) polynomial time whenever a related sum problem with binary cost coefficients can be solved in (strongly) polynomial time. This holds true for shortest paths such that the *k-max shortest path problem*

$$\min_{P \in \mathcal{P}_{st}} c_{(k)}(P) \tag{12}$$

can be solved in $\mathcal{O}(n \log n \log m + m \log m) = \mathcal{O}(n(\log n)^2 + m \log n)$ time provided that any (s,t)-path P has length $l(P) \geq k$. Using bisection, it is tested in each iteration if there exists a path from s to t whose k^{th}-largest cost edge has costs smaller than $c(e_j)$ for a given edge $e_j \in E$, $j \in \{1, \ldots, m\}$. Sorting the edges by non-decreasing costs, i.e.

$$c(e_1) \leq \ldots \leq c(e_m), \tag{13}$$

this is done iteratively by solving a sum SPP

$$\min_{P \in \mathcal{P}_{st}} \sum_{e_i \in P} d_j(e_i) \tag{14}$$

with binary (and thus non-negative) costs defined as

$$d_j(e_i) := \begin{cases} 0 & \text{if } i \leq j \\ 1 & \text{if } i > j. \end{cases} \tag{15}$$

Note that the k-max objective function is a universal objective function if we set $\lambda_k^l = 1$ and $\lambda_i^l = 0$ else for all $l \geq k$. The *k-min shortest path problem*

$$\min_{P \in \mathcal{P}_{st}} c_{(l(P)-k+1)}(P) \tag{16}$$

minimizes the k^{th}-smallest instead of the k^{th}-largest edge cost. If we define costs

$$d_j(e_i) := \begin{cases} 1 & \text{if } i < j \\ 0 & \text{if } i \geq j \end{cases} \tag{17}$$

it can be solved in the same way as k-max SPP where problem (14) turns into a sum shortest path problem of maximization type. Since this longest path problem can only be solved in polynomial time on acyclic graphs, the same holds for k-min SPP which is universal if $\lambda_{l-k+1}^l = 1$ and $\lambda_i^l = 0$ else for all $l \geq k$.

3 Resource Constrained Shortest Path Problems

For other types of shortest path problems than those listed in Section 2, we give a general solution concept which is based on resource constrained shortest paths. The idea is to pick in each iteration an edge $e_{j_k} \in E$ as k^{th}-largest or k^{th}-smallest cost edge and to solve associated sum SPPs where the set of feasible paths is restricted to those paths $P \in \mathcal{P}_{st}$ with edge e_{j_k} as k^{th}-largest or k^{th}-smallest cost edge, respectively. To this end, we choose an approach which has been discussed for so-called generalized balanced optimization problems [36] and specialize it to the case of shortest paths. A related approach using concepts of multicriteria optimization to handle different objective functions in the context of general combinatorial optimization problems was independently suggested by Gorski [13, 14].

We sort the edges $E = \{e_1, \ldots, e_m\}$ of digraph G by non-increasing costs such that

$$c(e_1) \geq \ldots \geq c(e_m). \tag{18}$$

According to this order, we define edge costs

$$d_{j_k}(e_i) := \begin{cases} 1 & \text{if } i < j_k \\ 0 & \text{if } i \geq j_k \end{cases} \tag{19}$$

or

$$d_{j_k}(e_i) := \begin{cases} 0 & \text{if } i \leq j_k \\ 1 & \text{if } i > j_k \end{cases} \tag{20}$$

as in Gorski and Ruzika [15] which are 0 or 1 depending on the index of edge $e_i \in E$ compared with that of a previously chosen edge e_{j_k}, $j_k \in \{1, \ldots, m\}$. The paths $P \in \mathcal{P}_{st}$ with edge e_{j_k} as k^{th}-largest or k^{th}-smallest cost edge can be characterized as follows (see Theorem 3 of [36]).

Theorem 1. *For an edge $e_{j_k} \in E$, $j_k \in \{1, \ldots, m\}$, and a path $P \in \mathcal{P}_{st}$, it holds:*

(a) e_{j_k} is the k^{th}-largest cost edge in P if and only if $\sum_{e_i \in P} d_{j_k}(e_i) = k - 1$ and $\sum_{e_i \in P} d_{j_k+1}(e_i) = k$ where the costs $d_{j_k}(e_i)$ and $d_{j_k+1}(e_i)$ are defined as in (19).

(b) e_{j_k} is the k^{th}-smallest cost edge in P if and only if $\sum_{e_i \in P} d_{j_k-1}(e_i) = k$ and $\sum_{e_i \in P} d_{j_k}(e_i) = k - 1$ where the costs $d_{j_k-1}(e_i)$ and $d_{j_k}(e_i)$ are defined as in (20).

The cost $c(e_{j_k})$ associated with edge e_{j_k} is then said to be the k^{th}-largest or k^{th}-smallest edge cost of path P, respectively.

Proof. It suffices to prove claim (a) since claim (b) can be shown analogously. By definition of the costs $d_{j_k}(e_i)$ and $d_{j_k+1}(e_i)$, the sums $\sum_{e_i \in P} d_{j_k}(e_i)$ and $\sum_{e_i \in P} d_{j_k+1}(e_i)$ count the number of edges in path P having an index which is strictly smaller than j_k or $j_k + 1$.
If $\sum_{e_i \in P} d_{j_k}(e_i) < k - 1$, there are at most $k - 2$ edges in path P with index smaller than j_k and these edges have costs greater or equal than $c(e_{j_k})$. Since path P is elementary, edge e_{j_k} occurs at most once and cannot be its k^{th}-largest cost edge. The same reasoning applies if $\sum_{e_i \in P} d_{j_k}(e_i) > k - 1$, $\sum_{e_i \in P} d_{j_k+1}(e_i) < k$ or $\sum_{e_i \in P} d_{j_k+1}(e_i) > k$.
Conversely, if $\sum_{e_i \in P} d_{j_k}(e_i) = k - 1$ and $\sum_{e_i \in P} d_{j_k+1}(e_i) = k$, path P has exactly $k-1$ edges with index smaller than j_k and k edges with index smaller than j_k+1. Thus, edge e_{j_k} is contained in path P and has k^{th}-largest cost. □

Observe that Theorem 1 is not correct for directed walks in which edge e_{j_k} might be used repeatedly.

Shortest path problems with sum objective and constraints as given in Theorem 1 (a) and (b) can be interpreted as resource constrained shortest path problems with equality constraints.

Definition 1. For a directed graph G with K resources we define the *resource constrained shortest path problem with equality constraints* as

$$\min \quad \sum_{e \in P} c(e) \tag{21a}$$

$$\text{s.t.} \quad \sum_{e \in P} r^k(e) = R^k \qquad \forall\ k = 1, \dots, K \tag{21b}$$

$$P \in \mathcal{P}_{st} \tag{21c}$$

where $r^k(e) \in \mathbb{Z}_0^+$ are the units of resource k consumed along edge $e \in E$ and $R^k \in \mathbb{Z}^+$ is the required total consumption of resource k on path P with $k = 1, \dots, K$.

Problem (21) is a variation of the *resource* or *weight constrained shortest path problem* in which the equality constraints (21b) are replaced by inequality constraints

$$\sum_{e \in E} r^k(e) \leq R^k \qquad \forall\ k = 1, \dots, K \tag{21d}$$

which give an upper bound on the consumption of resources $k = 1, \dots, K$. The problem with constraints (21d) is widely studied. Early work concerned with “route” problems of that type where nodes and edges may be repeated can be found in Joksch [20]. If there are no dicycles of negative cost, an optimal solution to such a walk problem will be an elementary path and solves the corresponding path problem. Besides dynamic programming algorithms (see e.g. Lawler [23]), other solution concepts for this problem (which is, in general, strongly NP-hard and even weakly NP-hard on acyclic digraphs with only one resource [11], [38]) include labeling algorithms, path ranking procedures and relaxation methods. For a summary on available literature, we refer to the monographs of Ziegelmann [41], Dumitrescu [5], Zhu [39] and Garcia [10]. A comprehensive survey on the generalized problem with resource windows defined for the nodes of graph G is given in Irnich and Desaulniers [19]. For directed graphs with negative-cost cycles, the *elementary resource constrained shortest path problem* in which node and edge repetition are explicitly forbidden has been considered in Beasley and Christofides [2], Dumitrescu [5], Feillet et al. [7] or Righini and Salani [31]. This problem is NP-hard in the strong sense [3].

In contrast, there is only few literature on the resource constrained shortest path problem with equality constraints. Research started with a paper of Saigal [32] on the corresponding walk problem. The path problem appeared as special case of the resource constrained shortest path problem with lower and upper resource limits, see Ribeiro and Minoux [30] or Beasley and Christofides [2]. Since the Hamiltonian path problem is obtained for $K = 1$ resource with $R^K = n - 1$, it is strongly NP-hard (compare Garey and Johnson [11]). Recently, Zhu and Wilhelm [40] proposed a three stage approach to tackle the problem on acyclic graphs which is a subproblem in column generation. In this case, the resource constrained shortest path problem with equality constraints and a single resource can be proved to be weakly NP-hard (compare Wang and Crowcroft [38]). In the following, we study a standard dynamic programming approach to tackle this problem.

For an acyclic digraph G with source s and sink t, we sort the nodes in topological order such that $i < j$ for all $(i, j) \in E$. For any node $j \in V$, we define

$$c_j(r^1, \dots, r^K) \tag{22}$$

to be the cost of a sum shortest path from s to j with resource consumption $r^k \in \{0, \dots, R^k\}$ for each resource k, $k = 1, \dots, K$. If the nodes are examined in topological order, these values can be computed recursively as

$$c_j(r^1, \dots, r^K) := \min_{(i,j)\in E:\, r^k_{ij} \le r^k} \{c_i(r^1 - r^1_{ij}, \dots, r^K - r^K_{ij}) + c_{ij}\} \tag{23}$$

starting with $c_j(0,\dots,0)$ which is the cost of a shortest path from source s to node j with total resource consumption equal to 0 or infinity if a path with this property does not exist. The correctness of this first algorithm follows from the principle of optimality.

Theorem 2. *On acyclic digraphs, the dynamic programming algorithm solves the resource constrained shortest path problem with equality constraints in $\mathcal{O}(T + mKR^1 \cdots R^K)$ time where T is the time needed to solve a one-to-all sum shortest path problem.*

Proof. The algorithm is correct since a cost-minimal path from s to j with resource consumption $r^1,\dots,r^K$ is the concatenation of a cost-minimal path from s to any predecessor node i of node j with resource consumption $r^1 - r^1_{ij},\dots,r^K - r^K_{ij}$ and edge $(i,j) \in E$ with cost c_{ij} and resource consumption $r^1_{ij},\dots,r^K_{ij}$. The optimal cost is $c_t(R^1,\dots,R^K)$ which is infinity if there are no resource feasible paths from s to t. The corresponding resource constrained shortest path $P^* \in \mathcal{P}_{st}$ can be found by backtracking along predecessor labels $pred(j;r^1,\dots,r^K)$ where we store node i in which the minimum of (23) is attained.
A topological order of the nodes in digraph G can be obtained in $\mathcal{O}(n+m)$ (see e.g. Krumke and Noltemeier [22]). The initialization steps can be done in $\mathcal{O}(T + mK)$ where T is the time to compute the shortest paths from s to all other nodes after deleting those edges $e \in E$ with $r^k(e) > 0$ for some resource k, $k = 1,\dots,K$. Since every edge $e \in E$ has to be considered only once, we need $\mathcal{O}(mK)$ time to determine $c_j(r^1,\dots,r^K)$ and the predecessors $pred(j;r^1,\dots,r^K)$ for all nodes $j \in V$. For all $r^k \in \{0,\dots,R^k\}$, $k = 1,\dots,K$, these values are obtained in a total of $\mathcal{O}(mKR^1 \cdots R^K)$ time. The resource constrained shortest path from s to t is constructed in at most $n-1$ steps using backtracking. □

An alternative algorithm in which no topological sorting is needed uses the following recursion

$$c^l_j(r^1,\dots,r^K) := \min \Bigg\{ c^{l-1}_j(r^1,\dots,r^K), \\ \min_{(i,j)\in E:\, r^k_{ij} \le r^k} \{c^{l-1}_i(r^1 - r^1_{ij},\dots,r^K - r^K_{ij}) + c_{ij}\} \Bigg\}. \tag{24}$$

For resource consumption $r^1,\dots,r^K$, it computes the cost of a sum shortest path from s to j which has no more than l edges. Since any path consists of at most $n-1$ edges, the cost of an optimal path from s to t is $c^{n-1}_t(R^1,\dots,R^K)$ where $c^0_j(r^1,\dots,r^K)$ has been initialized as

$$c^0_s(0,\dots,0) := 0 \tag{25}$$

and

$$c_j^0(r^1, \dots, r^K) := \infty \tag{26}$$

else. As in the proof of Theorem 2 we can show that this algorithm solves the resource constrained shortest path problem on acyclic graphs in time $\mathcal{O}(n^2 R^1 \cdots R^K + nmKR^1 \cdots R^K)$. In general graphs recursion (24) usually finds a shortest (s,t)-walk which cannot be reduced to an elementary (s,t)-path satisfying the resource constraints.

If the number of resources K is fixed, both dynamic programming algorithms have pseudo-polynomial time complexity depending on the size of the resource consumption $R^1, \dots, R^K$. If the resource constraints are as in Theorem 1 (a) or (b) there are $K = 2$ equality constraints with $R^1 = k - 1$ and $R^2 = k$ or $R^1 = k$ and $R^2 = k-1$, respectively, and the resulting resource constrained shortest path problem is even solvable in strongly polynomial time. Note that problem (27) can be reduced to a resource constrained shortest path problem with only one equality constraint if $k = 1$.

Corollary 1. *Using the first dynamic programming algorithm, problem*

$$\begin{aligned} \min \quad & \sum_{e \in E} c(e) && \text{(27a)} \\ \text{s.t.} \quad & \sum_{e_i \in P} d_{j_k}(e_i) = k - 1 && \text{(27b)} \\ & \sum_{e_i \in P} d_{j_k+1}(e_i) = k && \text{(27c)} \\ & P \in \mathcal{P}_{st} && \text{(27d)} \end{aligned}$$

where $d_{j_k}(e_i)$ and $d_{j_k+1}(e_i)$ are as in (19) *can be solved in $\mathcal{O}(n^2 m)$ time on acyclic directed graphs. The same holds for constraints $\sum_{e_i \in P} d_{j_k-1}(e_i) = k$ and $\sum_{e_i \in P} d_{j_k}(e_i) = k - 1$ where $d_{j_k-1}(e_i)$ and $d_{j_k}(e_i)$ are as in* (20).

Proof. For some fixed k the constraints ensure that edge e_{j_k} is the k^{th}-largest or k^{th}-smallest cost edge of the paths $P \in \mathcal{P}_{st}$. Since any (s,t)-path has at most $n - 1$ edges we have that $k \in \{1, \dots, n-1\}$ which implies that R^1, $R^2 \leq n$. By Theorem 2 these resource constrained shortest path problems are solvable in at most $\mathcal{O}(n^2 m)$ time if we apply the first version of the dynamic programming algorithm and the algorithms of Dijkstra or Bellman-Ford to find the sum shortest paths from source s to all nodes with resource consumption 0. □

4 New Variants of Shortest Path Problems

In the following three subsections, we investigate some new shortest path problems which are generalizations or extensions of the balanced, k-sum and k-max shortest path problems presented in Section 2. Some of the objective functions have already been discussed in the context of continuous and discrete location problems (see e.g. Nickel and Puerto [26] or Velten [37]), but the solution approach for path problems is different. We use the results of Section 3 and solve a sequence of resource constrained shortest path problems of type (27) with appropriately defined costs. For simplicity, we assume that the paths $P \in \mathcal{P}_{st}$ have "sufficient" length $l(P)$ such that the objective functions are well-defined for all (s, t)-paths in graph G. Furthermore, we suppose that the edges $e \in E$ are already sorted by non-increasing costs.

4.1 $(\boldsymbol{k_1}, \boldsymbol{k_2})$-Max and $(\boldsymbol{k_1}, \boldsymbol{k_2})$-Balanced SPP

The (k_1, k_2)-max and (k_1, k_2)-balanced shortest path problem are related path problems where the sum of the k_1^{st}-largest and k_2^{nd}-largest edge cost or the difference of the k_1^{st}-largest and k_2^{nd}-smallest edge cost, respectively, is minimized. The (k_1, k_2)-balanced objective function has not been considered before.

Definition 2. (a) If $k_1 < k_2 \leq l(P)$ for all paths $P \in \mathcal{P}_{st}$, the (k_1, k_2)-*max SPP* is defined as

$$\min_{P \in \mathcal{P}_{st}} \big(c_{(k_1)}(P) + c_{(k_2)}(P)\big). \tag{28}$$

(b) If $k_1 + k_2 \leq l(P)$ for all paths $P \in \mathcal{P}_{st}$, the (k_1, k_2)-*balanced SPP* is defined as

$$\min_{P \in \mathcal{P}_{st}} \big(c_{(k_1)}(P) - c_{(l(P)-k_2+1)}(P)\big). \tag{29}$$

These problems generalize the k-max and balanced SPP introduced in Section 1 and can be stated as Univ-SPPs setting $\lambda^l_{k_1} = \lambda^l_{k_2} = 1$ and $\lambda^l_i = 0$ else for (k_1, k_2)-max SPP with $l \geq k_2$ or $\lambda^l_{k_1} = 1$, $\lambda^l_{l-k_2+1} = -1$ and $\lambda^l_i = 0$ else for (k_1, k_2)-balanced SPP with $l \geq k_1 + k_2$. They can be solved by a single solution approach using k-max shortest paths.

Theorem 3. *On acyclic directed graphs, the (k_1, k_2)-max and (k_1, k_2)-balanced SPP can be solved in $\mathcal{O}(n^2 m^2 \log n)$ time by solving at most m constrained k_1-max SPPs.*

Proof. Let

$$\mathcal{P}_{st}(e_{j_{k_2}}) := \{P \in \mathcal{P}_{st} : \, e_{j_{k_2}} \text{ is the } k_2^{nd}\text{-largest cost edge in } P\} \tag{30}$$

or

$$\mathcal{P}_{st}(e_{j_{k_2}}) := \{P \in \mathcal{P}_{st} : \, e_{j_{k_2}} \text{ is the } k_2^{nd}\text{-smallest cost edge in } P\} \tag{31}$$

be the set of paths $P \in \mathcal{P}_{st}$ with edge $e_{j_{k_2}} \in E$ as k_2^{nd}-largest or k_2^{nd}-smallest cost edge. For each $j_{k_2} \in \{k_2, \ldots, m\}$ or $j_{k_2} \in \{k_1 + 1, \ldots, m - k_2 + 1\}$ (note that $\mathcal{P}_{st}(e_{j_{k_2}}) = \emptyset$ if index j_{k_2} is smaller than k_2 or larger than $m - k_2 + 1$) we solve

$$\min_{P \in \mathcal{P}_{st}(e_{j_{k_2}})} c_{(k_1)}(P) \tag{32}$$

which is a standard k_1-max shortest path problem with additional constraints

$$\sum_{e_i \in P} d_{j_{k_2}}(e_i) = k_2 - 1, \sum_{e_i \in P} d_{j_{k_2}+1}(e_i) = k_2 \tag{33}$$

or

$$\sum_{e_i \in P} d_{j_{k_2}-1}(e_i) = k_2, \sum_{e_i \in P} d_{j_{k_2}}(e_i) = k_2 - 1 \tag{34}$$

depending on the definition of $\mathcal{P}_{st}(e_{j_{k_2}})$. Applying the algorithm of Gorski and Ruzika [15] (for edges sorted by non-increasing costs), a k_1-max shortest path $P^*_{j_{k_2}} \in \mathcal{P}_{st}(e_{j_{k_2}})$ can be computed by solving $\log m$ many resource constrained shortest path problems of type (27) with binary edge costs. Path $P^*_{j_{k_2}}$ is obviously optimal for the (k_1, k_2)-max and (k_1, k_2)-balanced SPP with feasible set $\mathcal{P}_{st}(e_{j_{k_2}})$ since the objective function values are

$$f_\lambda(P) = c_{(k_1)}(P) \pm c(e_{j_{k_2}}) \tag{35}$$

and

$$c_{(k_1)}(P^*_{j_{k_2}}) \le c_{(k_1)}(P) \tag{36}$$

for all $P \in \mathcal{P}_{st}(e_{j_{k_2}})$. An overall optimal solution can thus be found as

$$P^* := \operatorname*{argmin}_{j_{k_2} \in \{k_2, \ldots, m\} / \{k_1+1, \ldots, m-k_2+1\}} f_\lambda(P^*_{j_{k_2}}) \tag{37}$$

where $f_\lambda(P^*_{j_{k_2}}) = \infty$ if an (s,t)-path P with edge $e_{j_{k_2}}$ as k_2^{nd}-largest or k_2^{nd}-smallest cost edge does not exist. The running time follows since we solve at most m constrained k_1-max SPPs each of which has complexity $\mathcal{O}(n^2 m \log m) = \mathcal{O}(n^2 m \log n)$. □

Using the following property of the (k_1, k_2)-max and (k_1, k_2)-balanced objective function, the corresponding path problems cannot only be solved on acyclic graphs, but also on general graphs. To evaluate the objective function value of an (s, t)-walk W we assume that the edge costs in walk W are counted with multiplicities.

Lemma 1. *For an (s, t)-walk W and its associated path $P \in \mathcal{P}_{st}$ obtained by eliminating all dicycles in W it holds that*

$$f_\lambda(P) \leq f_\lambda(W) \tag{38}$$

where $f_\lambda(\cdot)$ is the (k_1, k_2)-max or (k_1, k_2)-balanced objective function.

Proof. Let

$$c_{(1)}(W) \geq \ldots \geq c_{(l(W))}(W) \tag{39}$$

and

$$c_{(1)}(P) \geq \ldots \geq c_{(l(P))}(P) \tag{40}$$

be the edge costs of walk W and path P which have been sorted in non-increasing order. It holds that

$$c_{(k_1)}(P) \leq c_{(k_1)}(W) \tag{41}$$

and

$$c_{(k_2)}(P) \leq c_{(k_2)}(W) \tag{42}$$

but the k_2^{nd}-smallest edge cost becomes larger if we remove the dicycles in walk W, i.e.

$$c_{(l(P)-k_2+1)}(P) \geq c_{(l(W)-k_2+1)}(W). \tag{43}$$

This implies that

$$\begin{aligned} f_\lambda(P) &= c_{(k_1)}(P) + c_{(k_2)}(P) \\ &\leq c_{(k_1)}(W) + c_{(k_2)}(W) = f_\lambda(W) \end{aligned} \tag{44}$$

and

$$\begin{aligned} f_\lambda(P) &= c_{(k_1)}(P) - c_{(l(P)-k_2+1)}(P) \\ &\leq c_{(k_1)}(W) - c_{(l(W)-k_2+1)}(W) = f_\lambda(W) \end{aligned} \tag{45}$$

for the (k_1, k_2)-max and (k_1, k_2)-balanced objective function $f_\lambda(\cdot)$. □

Corollary 2. *On general digraphs, a (k_1, k_2)-max or (k_1, k_2)-balanced shortest path can be found in strongly polynomial time.*

Proof. In general graphs where no topological sorting exists we apply the second dynamic programming algorithm to tackle the resource constrained shortest path problems which are solved in each iteration of the constrained k_1-max shortest path problems (32). Therefore, the optimal solution W^* which is found as in Theorem 3 might be a non-elementary (with repeated nodes) or non-simple (with repeated edges) path from s to t. By Lemma 1, the corresponding (s,t)-path P^* that is obtained by deleting the dicycles in walk W^* satisfies

$$f_\lambda(P^*) \le f_\lambda(W^*) = \min_{j_{k_2}\in\{k_2,\dots,m\}/\{k_1+1,\dots,m-k_2+1\}} f_\lambda(P^*_{j_{k_2}}) \le f_\lambda(P^*_{j^*_{k_2}}) \le f_\lambda(P^*) \tag{46}$$

where the solutions $P^*_{j_{k_2}}$ of problems (32) might be walks and edge $e_{j^*_{k_2}}$ is the k_2^{nd}-largest or k_2^{nd}-smallest edge cost of path P^*. So P^* is an optimal path for (k_1,k_2)-max or (k_1,k_2)-balanced SPP with $f_\lambda(P^*) = f_\lambda(W^*)$. Using recursion (24) we need $\mathcal{O}(n^4 m \log n + n^3 m^2 \log n)$ time for solving m constrained k_1-max SPPs. □

A special case of (k_1,k_2)-balanced SPP is the *k-balanced shortest path problem* where the difference between the largest and k^{th}-smallest edge cost or the k^{th}-largest and smallest edge cost is as small as possible, i.e.

$$\min_{P\in\mathcal{P}_{st}} \left(\max_{e\in P} c(e) - c_{(l(P)-k+1)}(P) \right) \tag{47}$$

or

$$\min_{P\in\mathcal{P}_{st}} \left(c_{(k)}(P) - \min_{e\in P} c(e) \right). \tag{48}$$

These problems are solvable as described in Theorem 3 or Corollary 2. An alternative approach which is valid for any combinatorial optimization problem solves a sequence of at most m maximization problems of k-min type where $c(e) \le c_i$ for all $e \in E$ or m minimization problems of k-max type where $c(e) \ge c_i$ for all $e \in E$. The prescribed maximum or minimum edge cost c_i varies between $c(e_1),\dots,c(e_m)$ and both, the k-min and k-max problem, can be solved sequentially by problems

$$\min_{P\in\mathcal{P}_{st}} \sum_{e_i\in P} d_{j_k}(e_i) \tag{49}$$

with binary edge costs $d_{j_k}(e_i)$ as given in (20) or (19). This approach has a worst case complexity of $\mathcal{O}(nm(\log n)^2 + m^2 \log n)$.

4.2 Trimmed-Mean SPP and Related Problems

For a path $P \in \mathcal{P}_{st}$, the (k_1, k_2)-trimmed-mean objective function ignores its k_1 largest and k_2 smallest cost edges and adds the costs of the remaining edges in P. Conversely, if the costs of the k_1 largest and k_2 smallest edges are added and all other edge costs are ignored, this is called a (k_1, k_2)-anti-trimmed-mean objective function.

Definition 3. Let $k_1 + k_2 \leq l(P)$ for all paths $P \in \mathcal{P}_{st}$. We define

(a) the (k_1, k_2)-*trimmed-mean SPP* as

$$\min_{P \in \mathcal{P}_{st}} \sum_{i=k_1+1}^{l(P)-k_2} c_{(i)}(P) \tag{50}$$

(b) and the (k_1, k_2)-*anti-trimmed-mean SPP* as

$$\min_{P \in \mathcal{P}_{st}} \left(\sum_{i=1}^{k_1} c_{(i)}(P) + \sum_{i=l(P)-k_2+1}^{l(P)} c_{(i)}(P) \right). \tag{51}$$

Both objective functions generalize the k-sum objective function (see Section 2) and are "universal" by choosing $\lambda^l_{k_1+1} = \ldots = \lambda^l_{l-k_2} = 1$ or $\lambda^l_1 = \ldots = \lambda^l_{k_1} = \lambda^l_{l-k_2+1} = \ldots = \lambda^l_l = 1$ and $\lambda^l_i = 0$ else for all $l \geq k_1 + k_2$.

Trimmed-mean objectives are known from location theory, but we give the first algorithm for shortest path problems of this type. We use the ideas of Gupta and Punnen [17] and Punnen and Aneja [29] to solve a sequence of "easier" sum optimization problems with modified costs. As in the previous subsection, additional constraints to fix an edge as k^{th}-largest or k^{th}-smallest cost edge are needed.

For $j_{k_1}, j_{k_2} \in \{1, \ldots, m\}$, we define the sets $\mathcal{P}_{st}(e_{j_{k_1}})$ and $\mathcal{P}_{st}(e_{j_{k_2}})$ containing all (s,t)-paths P with edge $e_{j_{k_1}}$ as k_1^{st}-largest cost edge and edge $e_{j_{k_2}}$ as k_2^{nd}-smallest cost edge, respectively. In addition, we set

$$\begin{aligned} \mathcal{P}_{st}(e_{j_{k_1}}, e_{j_{k_2}}) :=& \{P \in \mathcal{P}_{st} : e_{j_{k_1}} \text{ is the } k_1^{st}\text{-largest cost edge in } P \\ & \text{and } e_{j_{k_2}} \text{ is the } k_2^{nd}\text{-smallest cost edge in } P\} \end{aligned} \tag{52}$$

where $\mathcal{P}_{st}(e_{j_{k_1}}, e_{j_{k_2}}) = \emptyset$ if the edges $e_{j_{k_1}}, e_{j_{k_2}} \in E$ cannot be the k_1^{st}-largest and k_2^{nd}-smallest cost edges of a path $P \in \mathcal{P}_{st}$. Furthermore, let us define edge costs $c_{j_{k_1},j_{k_2}}(e_i)$ as

$$c_{j_{k_1},j_{k_2}}(e_i) := \begin{cases} 0 & \text{if } i \leq j_{k_1} \\ c(e_i) & \text{if } j_{k_1} < i < j_{k_2} \\ c(e_{j_{k_2}}) & \text{if } i \geq j_{k_2} \end{cases} \tag{53}$$

for (k_1, k_2)-trimmed-mean SPP or

$$c_{j_{k_1},j_{k_2}}(e_i) := \begin{cases} c(e_i) - c(e_{j_{k_1}}) & \text{if } i \leq j_{k_1} \\ 0 & \text{if } j_{k_1} < i < j_{k_2} \\ c(e_i) & \text{if } i \geq j_{k_2} \end{cases} \tag{54}$$

for (k_1, k_2)-anti-trimmed-mean SPP.

Lemma 2. *Let $e_{j_{k_1}}$, $e_{j_{k_2}} \in E$.*

(a) For the (k_1, k_2)-trimmed-mean objective function $f_\lambda(\cdot)$ it holds that

$$\text{1. } \sum_{e_i \in P} c_{j_{k_1},j_{k_2}}(e_i) \geq f_\lambda(P) + k_2 c(e_{j_{k_2}}) \text{ for } P \in \mathcal{P}_{st}(e_{j_{k_1}}), \tag{55}$$

$$\text{2. } \sum_{e_i \in P} c_{j_{k_1},j_{k_2}}(e_i) = f_\lambda(P) + k_2 c(e_{j_{k_2}}) \text{ for } P \in \mathcal{P}_{st}(e_{j_{k_1}}, e_{j_{k_2}}). \tag{56}$$

(b) For the (k_1, k_2)-anti-trimmed-mean objective function $f_\lambda(\cdot)$ it holds that

$$\text{1. } \sum_{e_i \in P} c_{j_{k_1},j_{k_2}}(e_i) \geq f_\lambda(P) - k_1 c(e_{j_{k_1}}) \text{ for } P \in \mathcal{P}_{st}(e_{j_{k_2}}), \tag{57}$$

$$\text{2. } \sum_{e_i \in P} c_{j_{k_1},j_{k_2}}(e_i) = f_\lambda(P) - k_1 c(e_{j_{k_1}}) \text{ for } P \in \mathcal{P}_{st}(e_{j_{k_1}}, e_{j_{k_2}}). \tag{58}$$

Proof. We only prove the lemma for the (k_1, k_2)-trimmed-mean objective function $f_\lambda(\cdot)$. Let P be a path with edge $e_{j_{k_1}}$ as k_1^{st}-largest cost edge. We may assume that the cost values of the edges in path P which contribute to the right hand side of inequality (55)

$$f_\lambda(P) + k_2 c(e_{j_{k_2}}) \tag{59}$$

are 0 for the k_1 largest cost edges, $c(e_{j_{k_2}})$ for the k_2 smallest cost edges and unchanged otherwise. By definition of $c_{j_{k_1},j_{k_2}}(e_i)$, the k_1 largest cost edges of P with index $i \leq j_{k_1}$ have costs $c_{j_{k_1},j_{k_2}}(e_i) = 0$ as well. The costs of the remaining edges in P, however, are

$$c_{j_{k_1},j_{k_2}}(e_i) = c(e_i) \geq c(e_{j_{k_2}}) \tag{60}$$

if $j_{k_1} < i < j_{k_2}$ or

$$c_{j_{k_1},j_{k_2}}(e_i) = c(e_{j_{k_2}}) \geq c(e_i) \tag{61}$$

if $i \geq j_{k_2}$ due to the non-increasing sorting of the edge costs. It follows that

$$c_{j_{k_1},j_{k_2}}(e_i) \geq c(e_{j_{k_2}}) \tag{62}$$

for the k_2 smallest cost edges of path P and

$$c_{j_{k_1},j_{k_2}}(e_i) \geq c(e_i) \tag{63}$$

for the edges in between. This shows claim 1.
Claim 2 follows since the costs $c_{j_{k_1},j_{k_2}}(e_i)$ are equal to 0, $c(e_{j_{k_2}})$ and $c(e_i)$ for the k_1 largest cost edges, the k_2 smallest cost edges and all other edges of any path $P \in \mathcal{P}_{st}(e_{j_{k_1}}, e_{j_{k_2}})$. Inequality (55) is thus satisfied at equality. □

Theorem 4. *On acyclic directed graphs, the* (k_1, k_2)*-trimmed-mean and* (k_1, k_2)*-anti-trimmed-mean SPP can be solved in* $\mathcal{O}(n^2m^3)$ *time by solving at most* m^2 *resource constrained shortest path problems with equality constraints.*

Proof. Consider the (k_1, k_2)-trimmed-mean SPP. For each pair of edges $e_{j_{k_1}}$, $e_{j_{k_2}} \in E$ we solve a sum SPP with costs $c_{j_{k_1},j_{k_2}}(e_i)$ where the set of feasible paths is $\mathcal{P}_{st}(e_{j_{k_1}})$. This is a resource constrained shortest path problem of type (27) which is solvable in $\mathcal{O}(n^2m)$. By Lemma 2 (a), the corresponding resource constrained shortest path $P^*_{j_{k_1},j_{k_2}} \in \mathcal{P}_{st}(e_{j_{k_1}})$ satisfies

$$\begin{aligned} f_\lambda(P^*_{j_{k_1},j_{k_2}}) + k_2 c(e_{j_{k_2}}) &\leq \sum_{e_i \in P^*_{j_{k_1},j_{k_2}}} c_{j_{k_1},j_{k_2}}(e_i) \\ &\leq \sum_{e_i \in P} c_{j_{k_1},j_{k_2}}(e_i) = f_\lambda(P) + k_2 c(e_{j_{k_2}}) \end{aligned} \tag{64}$$

for all $P \in \mathcal{P}_{st}(e_{j_{k_1}}, e_{j_{k_2}})$. Subtracting $k_2 c(e_{j_{k_2}})$ we get

$$f_\lambda(P^*_{j_{k_1},j_{k_2}}) \leq f_\lambda(P). \tag{65}$$

An optimal solution to the (k_1, k_2)-trimmed-mean shortest path problem is

$$P^* := \underset{j_{k_1} \in \{k_1,\dots,m-k_2\},\, j_{k_2} \in \{k_1+1,\dots,m-k_2+1\}}{\operatorname{argmin}} f_\lambda(P^*_{j_{k_1},j_{k_2}}). \tag{66}$$

It can be determined in a total of at most $\mathcal{O}(n^2m^3)$ time. □

If we assume that the (original) edge costs are non-negative, the (k_1, k_2)-trimmed-mean shortest path problem can even be solved on general graphs. This is not correct for its counterpart, the (k_1, k_2)-anti-trimmed-mean shortest path problem.

Corollary 3. *In general digraphs with $c(e) \geq 0$ for all $e \in E$, the (k_1, k_2)-trimmed-mean SPP is solvable in strongly polynomial time.*

Proof. As indicated in the proof of Corollary 2 an optimal solution W^* found by recursion (24) might be a walk. Reducing walk W^* to its associated path P^* yields

$$l(P^*) < l(W^*) \tag{67}$$

and

$$c_{(i)}(P^*) \leq c_{(i)}(W^*) \tag{68}$$

for all $i = 1, \ldots, l(P^*)$. It follows that

$$\sum_{i=k_1+1}^{l(P^*)-k_2} c_{(i)}(P^*) \leq \sum_{i=k_1+1}^{l(P^*)-k_2} c_{(i)}(W^*) \tag{69}$$

and

$$\begin{aligned} f_\lambda(P^*) &= \sum_{i=k_1+1}^{l(P^*)-k_2} c_{(i)}(P^*) \\ &\leq \sum_{i=k_1+1}^{l(P^*)-k_2} c_{(i)}(W^*) + \sum_{i=l(P^*)-k_2+1}^{l(W^*)-k_2} c_{(i)}(W^*) = f_\lambda(W^*) \end{aligned} \tag{70}$$

because all costs are non-negative. As in Corollary 2, path P^* is optimal and can be found in $\mathcal{O}(n^4m^2 + n^3m^3)$ time. □

The following problem which we denote as (k_1, k_2)-*anti-trimmed-mean-balanced shortest path problem* and in which we compute the difference between the k_1 largest and k_2 smallest edge costs where $k_1 + k_2 \leq l(P)$ for all paths $P \in \mathcal{P}_{st}$

$$\min_{P \in \mathcal{P}_{st}} \left(\sum_{i=1}^{k_1} c_{(i)}(P) - \sum_{i=l(P)-k_2+1}^{l(P)} c_{(i)}(P) \right) \tag{71}$$

can be seen as combination of the balanced and (k_1, k_2)-anti-trimmed-mean SPP (we have $\lambda_1^l = \ldots = \lambda_{k_1}^l = 1$, $\lambda_{l-k_2+1}^l = \ldots = \lambda_l^l = -1$ and $\lambda_i^l = 0$ else for all $l \geq k_1 + k_2$). But in contrast to the latter, it can be solved by classical sum shortest path problems without resource constraints.

Theorem 5. *The (k_1, k_2)-anti-trimmed-mean-balanced SPP can be solved in strongly polynomial time on general directed graphs.*

Proof. We define a sum SPP with non-negative costs

$$c_{j_{k_1},j_{k_2}}(e_i) := \begin{cases} c(e_i) - c(e_{j_{k_1}}) & \text{if } i \leq j_{k_1} \\ 0 & \text{if } j_{k_1} < i < j_{k_2} \\ c(e_{j_{k_2}}) - c(e_i) & \text{if } i \geq j_{k_2} \end{cases} \tag{72}$$

for which an optimal path $P^*_{j_{k_1},j_{k_2}} \in \mathcal{P}_{st}$ can be found in $\mathcal{O}(m + n \log n)$ time applying the label-setting algorithm of Dijkstra. As in the proof of Theorem 4, we can argue that

$$\sum_{e_i \in P} c_{j_{k_1},j_{k_2}}(e_i) \geq f_\lambda(P) - k_1 c(e_{j_{k_1}}) + k_2 c(e_{j_{k_2}}) \tag{73}$$

for all paths $P \in \mathcal{P}_{st}$ since

$$c_{j_{k_1},j_{k_2}}(e_i) \geq c(e_i) - c(e_{j_{k_1}}) \tag{74}$$

$$c_{j_{k_1},j_{k_2}}(e_i) \geq c(e_{j_{k_2}}) - c(e_i) \tag{75}$$

and

$$c_{j_{k_1},j_{k_2}}(e_i) \geq 0 \tag{76}$$

for the k_1 largest, the k_2 smallest and the remaining cost edges. For the paths $P \in \mathcal{P}_{st}(e_{j_{k_1}}, e_{j_{k_2}})$ one has

$$\sum_{e_i \in P} c_{j_{k_1},j_{k_2}}(e_i) = f_\lambda(P) - k_1 c(e_{j_{k_1}}) + k_2 c(e_{j_{k_2}}) \tag{77}$$

and thus

$$f_\lambda(P) \geq f_\lambda(P^*_{j_{k_1},j_{k_2}}) \tag{78}$$

such that an optimal solution for (k_1, k_2)-anti-trimmed-mean-balanced SPP is among the sum shortest paths $P^*_{j_{k_1},j_{k_2}}$. Since less than m^2 sum shortest path problems have to be solved we get a complexity of $\mathcal{O}(m^3 + m^2 n \log n)$ time. □

An analogous result can be obtained for general combinatorial optimization problems if the corresponding problem with sum objective is solvable in (strongly) polynomial time.

4.3 Univ-SPP with Non-Negative Weight Coefficients and Weight Coefficients in "Blocks"

The path problems considered so far are universal shortest path problems whose weight coefficients λ_i^l are in $\{0, \pm 1\}$. In this section, we consider further variants where the universal weight coefficients are arbitrary or non-negative real numbers.

The results of Subsections 4.1 and 4.2 remain valid if the weight coefficients 1 or -1 are multiplied by α_1, $\alpha_2 \in \mathbb{R}_0^+$. Hence path problems with $\lambda_{k_1}^l = \alpha_1$, $\lambda_{k_2}^l = \alpha_2$ and $\lambda_i^l = 0$ else or $\lambda_{k_1}^l = \alpha_1$, $\lambda_{l-k_2+1}^l = -\alpha_2$ and $\lambda_i^l = 0$ else which generalize the (k_1, k_2)-max or (k_1, k_2)-balanced SPP can be solved in strongly polynomial time on acyclic and general directed graphs. For the problems of Subsection 4.2 the corresponding generalizations are

$$\min_{P \in \mathcal{P}_{st}} \sum_{i=k_1+1}^{l(P)-k_2} \alpha c_{(i)}(P) \tag{79}$$

where $\alpha \in \mathbb{R}_0^+$ and

$$\min_{P \in \mathcal{P}_{st}} \left(\sum_{i=1}^{k_1} \alpha_1 c_{(i)}(P) \pm \sum_{i=l(P)-k_2+1}^{l(P)} \alpha_2 c_{(i)}(P) \right). \tag{80}$$

To solve them, it suffices to modify the edge costs $c_{j_{k_1}, j_{k_2}}(e_i)$ defined in (53), (54) and (72).

Another class of universal objective functions to which our model applies are objective functions $f_\lambda(\cdot)$ where the universal weight coefficients λ_i^l can be divided into "blocks". Assuming that any path $P \in \mathcal{P}_{st}$ has length at least k_p, this means e.g. that

$$\begin{aligned}
\lambda_1^l &= \ldots = \lambda_{k_1-1}^l &&= \alpha_0 \\
\lambda_{k_1}^l &= \ldots = \lambda_{k_2-1}^l &&= \alpha_1 \\
\vdots & && \vdots \\
\lambda_{k_p}^l &= \ldots = \lambda_l^l &&= \alpha_p
\end{aligned} \tag{81}$$

or

$$\begin{aligned}
\lambda_1^l &= \ldots = \lambda_{l-k_p+1}^l = \alpha_p \\
\vdots & \qquad\qquad\quad \vdots \\
\lambda_{l-k_2+2}^l &= \ldots = \lambda_{l-k_1+1}^l = \alpha_1 \\
\lambda_{l-k_1+2}^l &= \ldots = \lambda_l^l \qquad = \alpha_0
\end{aligned} \tag{82}$$

where $p \in \{1, \ldots, n-2\}$, $1 < k_1 < \ldots < k_p \leq l$ and $\alpha_0, \alpha_1, \ldots, \alpha_p \in \mathbb{R}$. On acyclic digraphs, such problems can be solved in strongly polynomial time if p is fixed. For suitable $e_{j_{k_1}}, \ldots, e_{j_{k_p}} \in E$, we choose these edges as k_1^{st}-,…,k_p^{th}-largest or k_1^{st}-,…,k_p^{th}-smallest cost edges of path P and solve resource constrained shortest path problems with $2p$ constraints and costs defined as

$$c_{j_{k_1},\ldots,j_{k_p}}(e_i) := \begin{cases} \alpha_0 c(e_i) & \text{if } i < j_{k_1} \\ \alpha_1 c(e_i) & \text{if } j_{k_1} \leq i < j_{k_2} \\ \vdots & \vdots \\ \alpha_p c(e_i) & \text{if } i \geq j_{k_p} \end{cases} \tag{83}$$

or

$$c_{j_{k_1},\ldots,j_{k_p}}(e_i) := \begin{cases} \alpha_p c(e_i) & \text{if } i \leq j_{k_p} \\ \vdots & \vdots \\ \alpha_1 c(e_i) & \text{if } j_{k_2} < i \leq j_{k_1} \\ \alpha_0 c(e_i) & \text{if } i > j_{k_1} \end{cases} \tag{84}$$

respectively. Special cases are e.g. the minimum deviation and algebraic sum SPP (see Section 2). Others are the so-called cent-dian or anti-cent-dian objectives which are well-known in location theory and for which the universal weight coefficients are set to $\lambda_1^l = 1$ and $\lambda_i^l = \alpha$ else or $\lambda_l^l = -1$ and $\lambda_i^l = -\alpha$ else. Finally the problems in which we minimize the smallest or the k smallest edge costs belong to this problem class and can be modelled using 0-1 weight coefficients $\lambda_l^l = 1$ or $\lambda_l^l = \ldots = \lambda_{l-k+1}^l = 1$ and the remaining $\lambda_i^l = 0$.

5 Conclusion

We have provided strongly polynomial-time algorithms for a series of shortest path problems. In addition to the problems with sum, bottleneck or k-sum objective for which specially-designed shortest path algorithms exist, balanced, minimum deviation, algebraic sum and k-max shortest path problems could be addressed by algorithms which had been developed for general combinatorial optimization problems. To handle other objective functions we have solved equality constrained shortest path problems. Algorithms were given for the following problems on acyclic digraphs: (k_1, k_2)-max and (k_1, k_2)-balanced SPP, (k_1, k_2)-trimmed-mean and (k_1, k_2)-anti-trimmed-mean SPP, variants of these SPPs such as k-balanced or (k_1, k_2)-anti-trimmed-mean-balanced SPP, Univ-SPPs with non-negative universal weight coefficients and Univ-SPPs with universal weight coefficients in "blocks". The (k_1, k_2)-max and (k_1, k_2)-balanced SPP, the (k_1, k_2)-trimmed-mean SPP (provided that $c(e) \geq 0$), the k-balanced and (k_1, k_2)-anti-trimmed-mean-balanced SPP and their generalizations discussed in Section 4.3 were actually solvable on general digraphs.

Unlike the classical sum shortest path problem which can also be solved as linear program (see e.g. Ahuja et al. [1]), the path problems considered in this paper and their generalization, the universal shortest path problem, cannot be addressed by linear programming since the additional sorting problem makes the objective function non-linear. IP formulations for Univ-SPP are proposed and analysed in [35] and [34].

It is worth investigating the relationship of our approach and that of Gorski [13, 14] and combining them to improve the results in both papers. This will be done in the forthcoming thesis of Turner [34].

The ideas of Section 4 can be used to tackle the corresponding shortest walk problems where an optimal walk (with repeated nodes or edges) instead of an optimal elementary path is sought for. In the case where edges may be repeated and there are no negative-cost cycles with total resource consumption equal to 0, an edge $e_{j_k} \in E$ is guaranteed to be the k^{th}-largest or k^{th}-smallest cost edge in walk W if we require that

$$\sum_{e_i \in W} d_{j_k}(e_i) \leq k-1, \quad \sum_{e_i \in W} d_{j_k+1}(e_i) \geq k \tag{85}$$

or

$$\sum_{e_i \in W} d_{j_k-1}(e_i) \geq k, \quad \sum_{e_i \in W} d_{j_k}(e_i) \leq k-1, \tag{86}$$

respectively, where the costs $d_{j_k}(e_i)$, $d_{j_k+1}(e_i)$ or $d_{j_k-1}(e_i)$, $d_{j_k}(e_i)$ are defined as in (19) or (20). The resulting resource constrained walk problems have lower and upper resource limits (see e.g. Beasley and Christofides [2]) in the problem size.

6 Acknowledgements

This research was in part supported by Deutsche Forschungsgemeinschaft, Graduiertenkolleg "Mathematik und Praxis".

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network flows: Theory, algorithms, and applications.* Prentice Hall, 1993.

[2] J. E. Beasley and N. Christofides. An algorithm for the resource constrained shortest path problem. *Networks*, 19(4):379–394, 1989.

[3] M. Dror. Note on the complexity of the shortest path models for column generation in VRPTW. *Operations Research*, 42(5):977–978, 1994.

[4] C. W. Duin and A. Volgenant. Minimum deviation and balanced optimization: A unified approach. *Operations Research Letters*, 10(1):43–48, 1991.

[5] I. Dumitrescu. *Constrained path and cycle problems.* PhD thesis, University of Melbourne, 2002.

[6] J. Edmonds and D. R. Fulkerson. Bottleneck extrema. *Journal of Combinatorial Theory*, 8(3):299–306, 1970.

[7] D. Feillet, P. Dejax, M. Gendreau, and C. Gueguen. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44(3):216–229, 2004.

[8] P. Festa. Shortest path algorithms. In M. G. C. Resende and P. M. Pardalos, editors, *Handbook of optimization in telecommunications*, pages 185–210. Springer, 2006.

[9] H. N. Gabow and R. E. Tarjan. Algorithms for two bottleneck optimization problems. *Journal of Algorithms*, 9(3):411–417, 1988.

[10] R. Garcia. *Resource constrained shortest paths and extensions.* PhD thesis, Georgia Institute of Technology, 2009.

[11] M. R. Garey and D. S. Johnson. *Computers and intractability: A guide to the theory of NP-completeness.* W. H. Freeman and Company, 1979.

[12] R. Garfinkel, E. Fernández, and T. J. Lowe. The k-centrum shortest path problem. *Top*, 14(2):279–292, 2006.

[13] J. Gorski, 2010. Personal communication.

[14] J. Gorski. *Multiple objective optimization and implications for single objective optimization*. PhD thesis, University of Wuppertal, 2010.

[15] J. Gorski and S. Ruzika. On k-max-optimization. *Operations Research Letters*, 37(1):23–26, 2009.

[16] S. K. Gupta and A. P. Punnen. Minimum deviation problems. *Operations Research Letters*, 7(4):201–204, 1988.

[17] S. K. Gupta and A. P. Punnen. k-sum optimization problems. *Operations Research Letters*, 9(2):121–126, 1990.

[18] T. C. Hu. The maximum capacity route problem. *Operations Research*, 9(6):898–900, 1961.

[19] S. Irnich and G. Desaulniers. Shortest path problems with resource constraints. In G. Desaulniers, J. Desrosiers, and M. M. Solomon, editors, *Column generation*, pages 33–65. Springer, 2005.

[20] H. C. Joksch. The shortest route problem with constraints. *Journal of Mathematical Analysis and Applications*, 14(2):191–197, 1966.

[21] V. Kaibel and M. A. F. Peinhardt. On the bottleneck shortest path problem. ZIB-Report 06-22, Zuse Institute Berlin, 2006.

[22] S. O. Krumke and H. Noltemeier. *Graphentheoretische Konzepte und Algorithmen*. Vieweg+Teubner, 2009.

[23] E. L. Lawler. *Combinatorial optimization: Networks and matroids*. Dover Publications, 2001.

[24] S. Martello, W. R. Pulleyblank, P. Toth, and D. de Werra. Balanced optimization problems. *Operations Research Letters*, 3(5):275–278, 1984.

[25] M. Minoux. Solving combinatorial problems with combined min-max-min-sum objective and applications. *Mathematical Programming*, 45(1):361–372, 1989.

[26] S. Nickel and J. Puerto. *Location theory: A unified approach*. Springer, 2005.

[27] M. Pollack. The maximum capacity through a network. *Operations Research*, 8(5):733–736, 1960.

[28] A. P. Punnen. On combined minmax-minsum optimization. *Computers & Operations Research*, 21(6):707–716, 1994.

[29] A. P. Punnen and Y. P. Aneja. On k-sum optimization. *Operations Research Letters*, 18(5):233–236, 1996.

[30] C. C. Ribeiro and M. Minoux. A heuristic approach to hard constrained shortest path problems. *Discrete Applied Mathematics*, 10(2):125–137, 1985.

[31] G. Righini and M. Salani. New dynamic programming algorithms for the resource constrained elementary shortest path problem. *Networks*, 51(3):155–170, 2008.

[32] R. Saigal. A constrained shortest route problem. *Operations Research*, 16(1):205–209, 1968.

[33] A. Schrijver. *Combinatorial optimization: Polyhedra and efficiency.* Springer, 2003.

[34] L. Turner. *Universal combinatorial optimization.* PhD thesis, University of Kaiserslautern, 2011. To appear.

[35] L. Turner and H. W. Hamacher. Universal shortest paths. Report in Wirtschaftsmathematik 128, University of Kaiserslautern, 2010.

[36] L. Turner, A. P. Punnen, Y. P. Aneja, and H. W. Hamacher. On generalized balanced optimization problems. *Mathematical Methods of Operations Research*, 2010. Online first, DOI: 10.1007/s00186-010-0331-4.

[37] S. Velten. *Discrete location problems with flexible objectives.* PhD thesis, Saarland University, 2008.

[38] Z. Wang and J. Crowcroft. Quality-of-service routing for supporting multimedia applications. *IEEE Journal on Selected Areas in Communications*, 14(7):1228–1234, 1996.

[39] X. Zhu. *The dynamic, resource-constrained shortest path problem on an acyclic graph with application in column generation and a literature review on sequence-dependent scheduling.* PhD thesis, Texas A&M University, 2005.

[40] X. Zhu and W. E. Wilhelm. Three-stage approaches for optimizing some variations of the resource constrained shortest-path sub-problem in a column generation context. *European Journal of Operational Research*, 183(2):564–577, 2007.

[41] M. Ziegelmann. *Constrained shortest paths and related problems.* PhD thesis, Saarland University, 2001.

[42] U. Zwick. Exact and approximate distances in graphs - A survey. In *Algorithms - ESA 2001*, volume 2161 of *Lecture Notes in Computer Science*, pages 33–48. Springer, 2001.