Int. J. Appl. Math. Comput. Sci., 2007, Vol. 17, No. 2, 269–287
DOI: 10.2478/v10006-007-0023-2

SELECTED MULTICRITERIA SHORTEST PATH PROBLEMS: AN ANALYSIS OF COMPLEXITY, MODELS AND ADAPTATION OF STANDARD ALGORITHMS

ZBIGNIEW TARAPATA

Division of Operations Research and Decision Support
Cybernetics Faculty, Military University of Technology
ul. Kaliskiego 2, 00–908 Warsaw, Poland
e-mail: zbigniew.tarapata@wat.edu.pl

The paper presents selected multicriteria (multiobjective) approaches to shortest path problems. A classification of multiobjective shortest path (MOSP) problems is given. Different models of MOSP problems are discussed in detail. Methods of solving the formulated optimization problems are presented. An analysis of the complexity of the presented methods and ways of adapting of classical algorithms for solving multiobjective shortest path problems are described. A comparison of the effectiveness of solving selected MOSP problems defined as mathematical programming problems (using the CPLEX 7.0 solver) and multi-weighted graph problems (using modified Dijkstra's algorithm) is given. Experimental results of using the presented methods for multicriteria path selection in a terrain-based grid network are given.

Keywords: multiobjective shortest path, stochastic shortest path, algorithm complexity, routing problem, terrain-based modeling, approximation algorithm

1. Introduction

The problem of finding a shortest path from a specified origin node to another node has been considered, traditionally, in the framework of single objective optimization. More specifically, it is assumed that some value is associated to each arc (for example, the length or the travel time), and the goal is to determine a feasible path for which either the total distance or the total travel time is minimized. In many real applications it is often found that a single objective function is not sufficient to adequately characterize the problem. In such cases, multiobjective (multicriteria) shortest path (MOSP) problems are used.

There are many publications which deal with these problems in two frequently used domains: computer networks (Cidon *et al.*, 1997; 1999; Kerbache and Smith, 2000; Silva and Craveirinha, 2004) and transportation (Dial, 1979; Halder and Majumber, 1981; Rana and Vickson, 1988; Fujimura, 1996; Modesti and Sciomachen, 1998). For instance, in transportation networks, a typical situation that can be adequately represented only by considering multiple objectives is related to military route planning, where time, distance, or ability to camouflage on the path must be taken into account at the same time (Tarapata, 2003).

Another application in which it is important to deal with several factors is represented by path planning, where the goal is to find a navigation path for a mobile robot (Fujimura, 1996). In this case, the navigation path can be considered acceptable only if it satisfies multiple objectives, such as safety, time and energy consumption. In computer networks (as special cases of transportation networks), routing problems are most essential applications of MOSP problems.

The most often used criteria of route selection depend on the quality of service (QoS) (Silva and Craveirinha, 2004). These criteria are, for example, as follows: minimization of the number of lost packages, minimization of the maximal delay time of packages, minimization of the number of disjoint routes or minimization of the maximal transmission time for disjoint routes (in the case of disjoint routes), minimization of the overload, measured, e.g., by the mean value of traffic crossing by a link, minimization of the transmission time from a source to a destination, minimization of a route length, minimization of the probability of route unreliability or maximization of the probability of route reliability. Single-criterion formulations of routing problems use previously defined criteria. The choice of the appropriate method for solving defined problems depends on the answers to the following questions: Do we want to determine routes statically (algorithms such as: Dijkstra's, Ford-Bellmann's, PDM, A*) or dynamically (by adapting to the current load) (Djidjev *et al.*, 1995)? Are there stochastic dependencies in the network (Sigal *et al.*, 1980; Korzan, 1982; 1983a; 1983b; Loui, 1983; Tarapata, 1999; 2000)? Do we find a path for a single task or simultaneously for many tasks (e.g., through disjoint paths transmitting voice and picture or allocating channels in optical networks) (Li *et al.*, 1992; Schrijver and Seymour, 1992; Sherali *et al.*, 1998; Tarapata, 1999)? Do we plan to determine alternative paths (Golden and Skiscim, 1989)?

In many cases of routing problems, a single-criterion approach is not sufficient. There are many papers which deal with a description of practical examples of using many criteria in routing problems (Kerbache and Smith, 2000; Silva and Craveirinha, 2004). For example, Climaco *et al.* (2002) consider a bicriterion approach for routing problems in multimedia networks. In practical considerations, we often use contradicted criteria, e.g., fast and reliable access to the services (risk-profit) (Korzan, 1982; 1983a; 1983b; Loui, 1983; Tarapata, 1999; 2000). In such cases we can formulate and solve multicriteria optimization problems to support decisions of network designers (in computer or transportation networks) or administrators (traffic managers in transportation).

The aim of this paper is to analyse the complexity of MOSP problems and show how we can use modifications and advantages of fast implementations of Dijkstra's algorithm (using effective data structures, e.g., Fibonacci heaps, d-ary heaps) in order to effectively and optimally solve selected MOSP problems. As an additional result of this paper, a review of references and a categorization of MOSP problems are given.

2. State of the Art in Multiobjective Shortest Path (MOSP) Problems

MOSP problems are among the most tractable of NP-hard discrete optimization problems (Garey and Johson, 1979). In the work (Hansen, 1979a), the existence of a family of problems with an exponential number of optimal solutions was proved. This implies that any algorithm solving a multiobjective shortest path problem is, at least, exponential in the worst case analysis. On the other hand, some papers (Warburton, 1987; Vassilvitskii and Yannakakis, 2004; Tsaggouris and Zaroliagis, 2005) show that practical ε-approximate algorithms are generally limited either to problems having 2 or 3 criteria, or to problems requiring the ε-approximation of only certain restricted sets of efficient paths. One of the most popular methods of solving MOSP problems is the construction of approximate Pareto curves (Papadimitriou and Yannakakis, 2000; Vassilvitskii and Yannakakis, 2004). Informally, a $(1+\varepsilon)$-*Pareto curve* P_ε is a subset of feasible solutions such that for any Pareto optimal solution there exists a solution in P_ε that is no more than $(1+\varepsilon)$ away in all objectives.

Papadimitriou and Yannakakis (2000) show that for any multiobjective optimization problem there exists a $(1+\varepsilon)$-Pareto curve P_ε of (polynomial) size $|P_\varepsilon| = O((4B/\varepsilon)^{N-1})$, 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), which 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 $\boldsymbol{a}$, either compute a solution that dominates $\boldsymbol{a}$, or report that there is no solution better than $\boldsymbol{a}$ by at least a factor of $1+\varepsilon$ in all objectives. Extensions to this method to produce a constant approximation to the smallest possible $(1+\varepsilon)$-Pareto curve for the cases of 2 and 3 objectives are presented in (Vassilvitskii and Yannakakis, 2004), while for $N > 3$ objectives inapproximability results are shown for such a constant approximation. For the case of the MOSP (and some other problems with linear objectives), Papadimitriou and Yannakakis (2000) show how a "gap" routine can be constructed (based on a pseudopolynomial algorithm for computing exact paths) and, consequently, provide an FPTAS (Fully Polynomial Time Approximation Scheme) for this problem. Note that FPTASs for the MOSP were already known in the case of two objectives (Hansen, 1979a), as well as in the case of multiple objectives in directed acyclic graphs (DAGs) (Warburton, 1987). In particular, the biobjective case was extensively studied (Ehrgott and Gandibleux, 2002), while for $N > 2$ very little has been achieved; actually the results in (Warburton, 1987; Papadimitriou and Yannakakis, 2000; Tsaggouris and Zaroliagis, 2005) are the only and currently best FPTASs known.

Let $C^{\max}$ denote the ratio of the maximum to the minimum edge weights (in any dimension), V denote the number of nodes in a digraph, A denote the number of arcs (edges) and N be the number of criteria. For the case of DAGs and $N > 2$, the algorithm of (Warburton, 1987) runs in $O\left(VA\left(\frac{1}{\varepsilon}V(\log(VC^{\max}))\right)^{N-1}\left(\log\frac{V}{\varepsilon}\right)^{N-2}\right)$ time, while for $N = 2$ this improves to $O(VA\frac{1}{\varepsilon}\log V\log(nC^{\max}))$. For $N = 2$, an FPTAS can be created by repeated applications of a stronger variant of the "gap" routine—like an FPTAS for the restricted shortest path (RSPP) problem (Hassin, 1992; Lorenz and Raz, 2001; Ergun *et al.*, 2002). In (Vassilvitskii and Yannakakis, 2004), it is shown that this achieves a time of $O(VA|P_\varepsilon^*|(\log\log V + \frac{1}{\varepsilon}))$ for general digraphs and $O(VA|P_\varepsilon^*|/\varepsilon)$ for DAGs, where $|P_\varepsilon^*|$ is the size of the smallest possible $(1+\varepsilon)$-Pareto curve (and which can be as large as $\log_{1+\varepsilon} VC^{\max} \approx \frac{1}{\varepsilon}\ln(VC^{\max})$).

All these approaches deal typically with a single-pair version of the problem. Tsaggouris and Zaroliagis (2005) show a new and remarkably simple FPTAS for constructing a set of approximate Pareto curves for the *single-source* version of the MOSP problem in any digraph. For any $N > 1$, their algorithm runs in time $O\left(VA\left(\frac{1}{\varepsilon}V\log(VC^{\max})\right)^{N-1}\right)$ for general digraphs, and in $O\left(A\left(\frac{1}{\varepsilon}V\log(VC^{\max})\right)^{N-1}\right)$ for DAGs. These results improve significantly upon previous approaches for general digraphs (Golden and Skiscim, 1989; Hassin, 1992) and DAGs (Henig, 1985; Hassin, 1992), for all $N > 2$. For $N = 2$, their running times depend on ε^{-1}, while those based on repeated-RSPP applications (like in (Vassilvitskii and Yannakakis, 2004)) depend on ε^{-2}. Their approach to the MOSP, unlike previous methods based on converting pseudopolynomial time algorithms to an FPTAS using rounding and scaling techniques, builds upon a natural iterative process that extends and merges sets of node labels representing partial solutions, while keeping them small by discarding some solutions in an error controllable way.

One of the first papers to deal with MOSP problems was (Loui, 1983). The paper explores computationally tractable formulations of stochastic and multidimensional optimal path problems. A single formulation encompassing both problems is considered, in which a utility function defines preferences among candidate paths. The result is the ability to state explicit conditions for exact solutions using standard methods, and the applicability of well-understood approximation techniques.

Korzan wrote three papers (Korzan, 1982; 1983a; 1983b) which deal with the shortest path problem in unreliable networks. In the first one he presents methods of determining an optimal path in unreliable directed networks under various assumptions concerning the randomness of network elements. He assumes a vector objective function with two components: the path length (e.g., time) and some measure of unreliability (e.g., the probability of path “surviving”). An appriopriate multioptimization problem and a method for determining a compromise path for this problem are described. Some extensions of these problems and their solving methods included therein were discussed in two further papers (Korzan, 1983a; 1983b).

In the papers (Tarapata, 1999; 2000), an optimization problem of a few tasks in a parallel or distributed computing system in conditions of unreliability of computers and lines is considered. As a model of the system, a network is used with functions defined on its nodes (the time of task service at a node and the probability of node reliability) and arcs (time distances between nodes and the probability of arc (line) reliability during transmission). A damaging process of a network element (a node or an arc) is initiated: when a task starts its service in it (for a node) or its movement (for an arc) and it does not depend on the time which elapsed from the start time of the task sending (Tarapata, 1999), or when a task starts its service (or movement) in source nodes (Tarapata, 2000). In the second case, the “time-life” distribution of network elements depends on the time which elapsed from the start time of the task sending. This may be explained by the fact that, for example, the probability of damaging an element of a computer network is growing in time. In communication systems the probability of destroying system elements depends on the corresponding working time (the longer the system working time, the greater the possibility of system locating and, in consequence, the probability of annihilation of any system elements).

The problem of determining the best set of K disjoint paths in an unreliable network is formulated as a bicriteria optimization problem, in which the first criterion is the time of sending the slowest task (or the sum of sending times of all tasks) being minimized and the second is the probability of the reliability of all (K) paths being maximized. An approximate algorithm to solve the optimization problem is shown. The algorithm generalizes Dijkstra’s shortest path algorithm when we look for K $(K > 1)$ disjoint paths in the network with two functions (probabilities and distances) defined on the network nodes and arcs. Moreover, some conclusions concerning particular conditions which the paths should satisfy are given.

Generally, the multiobjective shortest path problem can be considered from the point of view of the following categories: number of criteria, type of problem (compromise solutions, lexicographic solutions, max-ordering problem, etc.), solution method (label setting or correcting, tabu search, simulated annealing and others). In Table 1 we classify MOSP problems (as a modification of the classification proposed in (Ehrgott and Gandibleux, 2000; 2002)) using the notation $X/Y/Z$, where X is the number and type of objective functions ($X = Q$ stands for an arbitrary number of objectives, e.g., 1-Sum$Q$-max denotes a problem with the sum and Q bottleneck objectives), Y denotes the problem type, Z denotes the type of solution method. The entries of Y are as follows: E – finding the efficient set, e – finding a subset of the efficient set, *SE* – finding supported efficient solutions, *Appr*(x) – finding an approximation of x, *lex* – solving the lexicographic problem (preemptive priorities), *MO* – max-ordering problem, U – optimizing a utility function, C/S – finding a compromise/satisfying solution, D – disjoint-path problem, *SCH* – stochastic problem. The entries of Z are as follows: *SP* – exact algorithm specifically designed for the problem, *LS*/*LC* – label setting or label correcting method, *DP* – algorithm based on dynamic programming, *BB* – algorithm based on branch and bound, *IA* – interactive method,

Table 1. Classification of Multiobjective Shortest Path (MOSP) problems.

Code of the problem	References
2-SUM/E/LC	(Tung and Chew, 1988; Brumbaugh-Smith and Shier, 1989; Skriver and Andersen, 2000)
2-SUM/E/LS	(Hansen, 1979a; 1979b)
2-SUM/E/2P,LC	(Mote *et al.*, 1991)
2-SUM/E/SP	(Martins and Climaco, 1981; Climaco and Martins, 1982; Huarng *et al.*, 1996)
2-SUM/E/DP	(Henig, 1985)
2-SUM/ Appr(E)/Appr	(Hansen, 1979a; 1979b)
1-SUM 1-max/E/SP	(Hansen, 1979a; 1979b; Pelegrin and Fernandez, 1998)
2-SUM/C/IA	(Current *et al.*, 1990)
2-SUM/U/SP	(Henig, 1985)
2-SUM/U/IA	(Murthy and Olsen, 1994)
2-SUM/e/IA	(Coutinho-Rodrigues *et al.*, 1999)
2-SUM/C,SCH/LS	(Korzan, 1982; 1983b)
2-SUM/lex,SCH/LS	(Korzan, 1983a; 1983b)
3-SUM/E/LC	(Gabrel and Vanderpooten, 1996)
3-SUM/C/IA	(Gabrel and Vanderpooten, 1996)
Q-SUM/SE/SP	(Henig, 1985; White, 1987)
Q-SUM/E/LS	(Martins, 1984)
Q-SUM/E/LC	(Tung and Chew, 1992; Corley and Moon, 1985; Cox, 1984)
Q-SUM/E/DP	(Hartley, 1985; Kostreva and Wiecek, 1993)
Q-SUM/Appr(E),Appr(MO)/Appr	(Warburton, 1987)
Q-SUM/C/IA	(Henig, 1994)
Q-SUM/U/DP	(Carraway *et al.*, 1990)
Q-SUM/U/SP	(Modesti and Sciomachen, 1998)
Q-SUM/MO/DP,BB	(Rana and Vickson, 1988)
Q-SUM/MO/LC	(Murthy and Her, 1992)
Q-SUM/U,SCH/Appr	(Loui, 1983)
Q-SUM/MO,D,C,lex,SCH/Appr,LS	(Tarapata, 1999; 2000)

$2P$ – two-phase method, *Appr* – approximation algorithm with worst case performance bound.

Other particular multiobjective path problems are presented in (Dial, 1979; Engberg *et al.*, 1983; Halder and Majumber, 1981; Sancho, 1988; Wijeratne *et al.*, 1993).

3. Model of the MOSP Problem

Let a directed graph $G = \langle V_G, A_G \rangle$ be given, where V_G is the set of graph nodes, $V_G = \{1, 2, \ldots, V\}$, A_G stands for the set of graph arcs, $A_G \subset \{\langle v, v' \rangle : v, v' \in V_G\}$, $|A_G| = A$. For example, in computer networks, we have routers as nodes of G and physical links between the routers as arcs of G. Generally, for each arc of G, we may define arc functions $f_n(v, v')$, $n = 1, \ldots, N$, which describe characteristics of the arc $\langle v, v' \rangle \in A_G$ such as the transmission time, distance, load, reliability, capacity, acceptable flows, etc. We assume that there are K tasks which we need to transport from the source nodes i^s to the destination ones i^d, $i^s = (i^s(1), i^s(2), \ldots, i^s(k), \ldots, i^s(K))$, $i^d = \big(i^d(1), i^d(2), \ldots, i^d(k), \ldots, i^d(K)\big)$. For $K = 1$, we have the classical case of routing for a single task. In some examples used in the paper we use a computer network model as G with a predefined matrix $\boldsymbol{t} = [\boldsymbol{t}_{v,v'}]_{V \times V}$, where $\boldsymbol{t}_{v,v'} = \big\langle t^1_{v,v'}, t^2_{v,v'}, \ldots, t^k_{v,v'}, \ldots, t^K_{v,v'} \big\rangle$ and $t^k_{v,v'}$ signifies a nonnegative value describing the transaction (transmission) time of the k-th task on the arc $\langle v, v' \rangle \in$

A_G (when $v \neq v'$). Moreover, let $I_k(i^s(k), i^d(k))$ describe a simple path and $T_k(i^s(k), i^d(k))$ describe achieving times of nodes belonging to the path for the k-th task as follows:

$$I_k\left(i^s(k), i^d(k)\right) = \left(i^0(k) = i^s(k), i^1(k), \ldots, i^r(k), \ldots, i^{R_k}(k) = i^d(k)\right),$$

$$T_k\left(i^s(k), i^d(k)\right) = \left(\tau^0(k), \tau^1(k), \ldots, \tau^r(k), \ldots, \tau^{R_k}(k)\right),$$

where $i^r(k)$ is the r-th node on the path for the k-th task and $\tau^r(k)$ stands for the achieving time of the r-th node on the path for the k-th task,

$$\tau^r(k) = \sum_{m=1}^{r} t^k_{i^{m-1}(k), i^m(k)}, \quad r = \overline{1, R_k}, \quad k = \overline{1, K}. \tag{1}$$

We adopt the convention that if $K = 1$, then we omit the index k (i.e., $i^r(1) \equiv i^r$, $\tau^r(1)) \equiv \tau^r$, etc.).

3.1. Formulation of the MOSP Problem

3.1.1. General Formulation of the Optimization Problem with a Vector Objective Function. We denote by $M(i^s, i^d)$ the set of acceptable K-dimensional vectors of paths in G from i^s to i^d, and by $I(i^s, i^d)$ – an element of $M(i^s, i^d)$. It can be observed that $I(i^s, i^d)$ is a vector whose components are simple paths for each k-th task. We also write $I \equiv I(i^s, i^d)$ (we omit i^s and i^d). We assume that we have an N-component vector $F(I) = \langle F_1(I), F_2(I), \ldots, F_N(I)\rangle$ of criteria functions estimating the vector of paths $I \in M(i^s, i^d)$. We have an arc function $f_n(v, v')$, $\langle v, v'\rangle \in A_G$, $n \in \{1, \ldots, N\}$, which will be used to compute $F_n(I)$ (e.g., as a sum of values of $f_n(v, v')$ for arcs belonging to the path I). Thus we can say that on the set $M(i^s, i^d)$ we defined a vector objective function as follows:

$$F(I) = \langle F_1(I), F_2(I), \ldots, F_N(I)\rangle, \quad I \in M(i^s, i^d). \tag{2}$$

The routing problem can be formulated as the multicriteria optimization problem:

$$\left\langle M(i^s, i^d), F(I), R^D \right\rangle, \tag{3}$$

where $R^D \subset Y^D(i^s, i^d) \times Y^D(i^s, i^d)$ is a domination relation in the criteria space

$$Y^D(i^s, i^d) = \left\{F(I) = \langle F_1(I), F_2(I), \ldots, F_N(I)\rangle : I \in M(i^s, i^d)\right\},$$

where

$$R^D = \Big\{ \langle F(I_m), F(I_z)\rangle \in Y^D(\cdot,\cdot) \times Y^D(\cdot,\cdot) : \Psi\left(F(I_m), F(I_z)\right) \Big\}, \tag{4}$$

$$\Psi\big(F(I_m), F(I_z)\big) = \begin{cases} 1 & \text{when } I_m \text{“is better” than } I_z, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

We can solve (3) using various methods of finding so-called nondominated solutions. The set of nondominated results equals

$$Y^{ND}(i^s, i^d) = \left\{ y(I) \in Y^D(\cdot,\cdot) :\sim \underset{\substack{z(I) \in Y^D(\cdot,\cdot) \\ z(I) \neq y(I)}}{\exists} \langle z(I), y(I)\rangle \in R^D \right\}. \tag{6}$$

The set of nondominated solutions (paths) is determined as the inverse image of $Y^{ND}(i^s, i^d)$ as follows:

$$M^{ND}(i^s, i^d) = \Big\{ I \in M(i^s, i^d) : y(I) \in Y^{ND}(\cdot,\cdot) \Big\}. \tag{7}$$

In order to solve MOSP problems, other approaches are also used, e.g., vector ε-domination (Warburton, 1987; Tsaggouris and Zaroliagis, 2005). The method of vector ε-domination uses the following definition:

Definition 1. (Warburton, 1987): We say that a vector $\boldsymbol{a} = \langle a_1, a_2, \ldots, a_N\rangle$ ε-dominates a vector $\boldsymbol{b} = \langle b_1, b_2, \ldots, b_N\rangle$ for fixed $\varepsilon \geq 0$ (we write $\boldsymbol{a} \overset{\varepsilon}{\leq} \boldsymbol{b}$) if

$$a_n \leq (1 + \varepsilon) b_n, \quad n = \overline{1, N}. \tag{8}$$

In some approaches it is additionally assumed that, for at least one $n \in \{1, \ldots, N\}$, e.g., n', we have $a_{n'} < (1 + \varepsilon) b_{n'}$. It can be observed that for $\varepsilon = 0$ this concept reduces to the usual notion of vector dominance. To use this approach, we have to replace the domination relation (4) by the ε-domination relation

$$R^D_\varepsilon = \Big\{ \langle F(I_m), F(I_z)\rangle \in Y^D(\cdot,\cdot) \times Y^D(\cdot,\cdot) : F(I_m) \overset{\varepsilon}{\leq} F(I_z) \Big\},$$

and we can solve the problem of finding an ε-shortest path which, according to (8), has a cost by no more than $(1+\varepsilon)$ away from the optimal values for all objectives. Warburton (1987) studies methods for approximating the set of Pareto optimal paths in multiple-objective, shortest path problems. He gives approximation methods that can estimate Pareto optima to any required degree (ε) of accuracy. The basis of his results is that the proposed methods are “fully polynomial”: they operate in time and space bounded by a polynomial in problem size and accuracy of approximation—the greater the accuracy, the longer the time required to reach a solution.

3.1.2. Exemplary Routing Problem Formulation as a Bicriteria Optimization Problem. In the example of the routing problem formulation as an MOSP problem, we assume that on each arc $\langle v, v' \rangle$ of the graph G we additionally define a function $q_{v,v'}(t)$ (identical for each task $k = \overline{1, K}$, so we omit k in the description of $q_{v,v'}(t)$), which describes the probability of arc reliability at least in time t:

$$q_{v,v'}(t) = \Pr\{\gamma_{v,v'} \geq t\},$$

where $\gamma_{v,v'}$ is a nonnegative random variable representing the "time-life" of the arc $\langle v, v' \rangle$. We assume that the random variables $\gamma_{v,v'}$ are nonnegative and independent for each pair $\langle v, v' \rangle$ of arcs. Then for each vector of the paths I in G we can define the following probability that all K paths will "survive":

$$P\left(I(i^s, i^d)\right) = \prod_{k=1}^{K} \prod_{r=1}^{R_k} q_{i^{r-1}(k), i^r(k)} \left(t^k_{i^{r-1}(k), i^r(k)}\right). \tag{9}$$

Next we also define the time of achieving destination nodes by all K tasks as the time of achieving the destination node by the most delayed task (10) or as a sum of achieving times of destination nodes (11):

$$T\left(I(i^s, i^d)\right) = \max_{k \in \{1,\dots,K\}} \tau^{R_k}(k) \tag{10}$$

or

$$T\left(I(i^s, i^d)\right) = \sum_{k \in \{1,\dots,K\}} \tau^{R_k}(k) \tag{11}$$

Then the vector objective function (2) has the form

$$F(I) = \langle T(I), P(I) \rangle, \quad I \in M(i^s, i^d),$$

i.e., $F_1(I) = T(I)$, $F_2(I) = P(I)$. The criteria space $Y^D(i^s, i^d)$ has the form

$$Y^D(i^s, i^d) = \left\{F(I) = \langle T(I), P(I) \rangle : I \in M(i^s, i^d)\right\},$$

and the function (5) (which makes the relation (4) a Pareto one):

$$\Psi\big(F(I_m), F(I_z)\big) = \begin{cases} 1 & \text{if } \ (T(I_m) < T(I_z) \wedge P(I_m) \geq P(I_z)) \\ & \quad \vee (T(I_m) \leq T(I_z) \wedge P(I_m) > P(I_z)), \\ 0 & \text{otherwise.} \end{cases}$$

We can equivalently define the problem formulated above as follows: Determine $I^*(i^s, i^d) \in M\left(i^s, i^d\right)$ for which

$$T^* = T\left(I^*(i^s, i^d)\right) = \min_{I(i^s,i^d) \in M(i^s,i^d)} T\left(I(i^s, i^d)\right),$$
$$P^* = P\left(I^*(i^s, i^d)\right) = \max_{I(i^s,i^d) \in M(i^s,i^d)} P\left(I(i^s, i^d)\right), \tag{12}$$

or

$$\hat{P}^* = \min_{I(i^s,i^d) \in M(i^s,i^d)} \hat{P}\left(I(i^s, i^d)\right) = \min_{I(i^s,i^d) \in M(i^s,i^d)} 1 - P\left(I(i^s, i^d)\right). \tag{13}$$

Generally, if the objective is to maximize one or more components of $F(I)$ from (2), MOSP algorithms can be applied to compute efficient paths only if G is acyclic (DAG). If G contains cycles and $N = 1$, we solve the NP-hard longest path problem (for $N > 1$ the problem is at least as difficult as for $N = 1$) (Garey and Johson, 1979). Therefore, we assume that all components of $F(I)$ are minimized and all have nonnegative values.

4. Methods of Solving MOSP Problems

4.1. Methods of Solving Single-Criterion Subproblems of the MOSP Problem. A method of determining T^* and P^* from (12) and (13) depends on the number K of tasks for which we determine paths. If $K = 1$, then we have a classical shortest paths problem in the graph G for fixed pairs of nodes (i^s, i^d) with an arc function $t_{v,v'}$. This problem could be solved for the criterion function $T(I(i^s, i^d))$ using, e.g., the following algorithms: Dijkstra's (based on effective data structures as Fibonacci's heaps (complexity $O(V \log V + A)$), d-ary heaps (complexity $O(A \log_d V)$, $d = \max\{2, \lceil A/V \rceil\}$)) (Schrijver, 2004), Ford-Bellman's, A^* (Djidjev *et al.*, 1995). When an arc function is nonadditive or nonlinear, we can use the approach described, e.g., by Bernstein and Kelly (1997), or we can formulate a nonlinear optimization problem and solve it using Kuhn-Tucker optimality conditions. For the function $\hat{P}(I(i^s, i^d))$, the approach presented, e.g., in (Korzan, 1983b) could be used. Even though the function $\hat{P}(I(i^s, i^d))$ from (13) is multiplicative (a product of probabilities), then it is possible to obtain its additive form as follows:

$$\tilde{\hat{P}}\left(I(i^s, i^d)\right) = \sum_{k=1}^{K} \sum_{r=1}^{R_k} \left| \ln q_{i^{r-1}(k), i^r(k)} \left(t^k_{i^{r-1}(k), i^r(k)}\right) \right|.$$

Defining the arc function as $f_1(v, v') = |\ln q_{v,v'}(t_{v,v'})|$, we can solve the problem (12)–(13) optimally using Dijkstra's algorithm (because the function $f_1(v, v')$ is additive and nonnegative). The obtained solutions (i.e., $I^*(i^s, i^d)$) both for $\hat{P}(I(i^s, i^d))$ and $\tilde{\hat{P}}(I(i^s, i^d))$ are identical. Other approaches to find the best path in stochastic graphs are considered in (Sigal *et al.*, 1980; Korzan, 1982; 1983a; Loui, 1983; Tarapata, 1999; 2000).

The situation is more complicated when $K > 1$. If we want to find disjoint routes for K tasks, then even for $K = 2$ and the function $T(I(i^s, i^d))$ the problem is NP-hard (Schrijver and Seymour, 1992; Schrijver, 2004). Li *et al.* (1992) gave a pseudopolynomial algorithm for an optimization version of the disjoint two-path problem in which the length of the longer path must be minimized. Eppstein (1999) considered the problem of finding pairs of node-disjoint paths in DAGs, either connecting two given nodes to a common ancestor, or connecting two given pairs of terminals. He demonstrated how to find K pairs with the shortest combined length in $O(AV + K)$ time. The papers (Suurballe and Tarjan, 1984; Li *et al.*, 1992; Sherali *et al.*, 1998) deal with problems and algorithms of disjoint paths for $K = 2$, but the papers (Schrijver and Seymour, 1992; Tarapata, 1999; 2000) deal with general problems of disjoint paths. In the papers (Tarapata, 1999; 2000), an optimization problem of several tasks (through disjoint paths) in a parallel or distributed computing system in the conditions of unreliability of computers and lines is considered. An approximation algorithm to solve the optimization problem is shown. The algorithm generalizes Dijkstra's shortest path algorithm to the case when we look for K $(K > 1)$ disjoint paths in a network.

In further deliberations we assume that $K = 1$. Let us note that for $K = 1$ the objective functions (10) and (11) are equivalent. We also assume that $i_1^s = s$, $i_1^d = t$.

4.2. Method of Compromise Solutions. To find a compromise solution with the parameter $p \geq 1$ we use the following metric ε_p, in the space $Y^D(\cdot,\cdot)$:

$$\varepsilon_p\big(h^*, h(I)\big) = \|h^*, h(I)\|_p = \sqrt[p]{\sum_{n=1}^{N} |h_n^* - h_n(I)|^p}. \tag{14}$$

For a compromise result h^0 the following condition is satisfied:

$$\varepsilon_p\big(h^*, h^0(I)\big) = \min_{I \in M(i^s, i^d)} \varepsilon_p\big(h^*, h(I)\big). \tag{15}$$

The compromise solution $I^c \in M(i^s, i^d)$ is such that (15) is satisfied. Note that the metric (14) defines different distances from an "ideal" solution, h^*:

- for $p = 1$ we obtain the sum of the absolute deviations from the ideal point (taxi distance);
- for $p = 2$ we obtain the Euclidean norm (in a two-dimensional space it amounts to the geometric distance between points)—it is the "best" compromise (Korzan, 1982; 1983b; Current *et al.*, 1990; Henig, 1994; Gabrel and Vanderpooten, 1996);
- for $p = \infty$ we obtain the Tchebycheff norm (minimization of maximal differences between "ideal" and actual values of criteria); this problem is also known as a max-ordering problem (Rana and Vickson, 1988; Warburton, 1987; Mote *et al.*, 1991).

To find a compromise solution with the parameter $p \geq 1$, we use the metric ε_1 while replacing $T(I)$ by $\overline{T}(I)$ and $P(I)$ by $\overline{P}(I)$. In order to find a compromise solution of the problem (3) with a vector objective function $F(I) = \langle T(I), P(I)\rangle$, we have to determine T^* and P^* described in the previous section. Having T^* and P^*, we can define

$$\overline{P}(I) = \frac{P(I)}{P^*}, \quad \overline{T}(I) = \frac{T(I)}{T^*},$$

thus obtaining the normalized vector objective function

$$h(I) = \left\langle \frac{T(I)}{T^*}, \frac{P(I)}{P^*} \right\rangle \tag{16}$$

under the assumption that $T^* \neq 0$ and $P^* \neq 0$. It can be observed that $\overline{T}(I) \geq 1$ and $\overline{P}(I) \leq 1$, $I \in M(\cdot,\cdot)$, so we obtain a normalized ideal point $h^* = (1, 1)$.

For example, for $p = 1$ we obtain

$$\varepsilon_1\big(h^*, h(I)\big) = \left|1 - \frac{T(I)}{T^*}\right| + \left|1 - \frac{P(I)}{P^*}\right|.$$

From the conditions

$$1 - \frac{T(I)}{T^*} \leq 0, \quad 1 - \frac{P(I)}{P^*} \geq 0$$

we get

$$\begin{aligned}\varepsilon_1\big(h^*, h(I)\big) &= \frac{T(I)}{T^*} - 1 + 1 - \frac{P(I)}{P^*} \\ &= \frac{T(I)}{T^*} - \frac{P(I)}{P^*}.\end{aligned}$$

For a compromise result h^0 the following condition is satisfied:

$$\begin{aligned}\varepsilon_1\big(h^*, h^0(I)\big) &= \min_{I \in M(i^s, i^d)} \varepsilon_1\big(h^*, h(I)\big) \\ &= \min_{I \in M(i^s, i^d)} \left[\frac{T(I)}{T^*} - \frac{P(I)}{P^*}\right].\end{aligned}$$

For a compromise solution $I^c \in M(i^s, i^d)$ (with $p = 1$) the above formula is satisfied.

However, since the function

$$\frac{T(I)}{T^*} - \frac{P(I)}{P^*}$$

has positive values, it is difficult to build an additive nonnegative arc function to compute it. This is very inconvenient because Dijkstra's algorithm (as a classical algorithm solving the shortest path problem) requires the values of the arc function to be nonnegative and additive (the function $\varepsilon_1(h^*, h(I))$ is nonadditive because of multiplications during the calculation of $P(I)/P^*$). Korzan (1982) shows that (for one task, i.e., $K = 1$), if the arc function $q_{v,v'}(t)$ is in the form $q_{v,v'}(t) = e^{-\lambda(v,v')\cdot t}$,

$\lambda(v,v') > 0$, that is, the probability function P from (9) equals

$$\begin{aligned} P\left(I(i^s,i^d)\right) &= \prod_{r=1}^{R_1} q_{i^{r-1},i^r}\left(t^1_{i^{r-1},i^r}\right) \\ &= \prod_{r=1}^{R_1} \exp(-\lambda(i^{r-1},i^r)t^1_{i^{r-1},i^r}) \\ &= \exp\Big(\sum_{r=1}^{R_1} -\lambda(i^{r-1},i^r)t^1_{i^{r-1},i^r}\Big), \end{aligned}$$

then the maximization of $P(I(i^s,i^d))$ is equivalent to the minimization of

$$\beta\left(I(i^s,i^d)\right) = \sum_{r=1}^{R_1} \lambda(i^{r-1},i^r)t^1_{i^{r-1},i^r}.$$

In this case we can define a new normalized vector objective function

$$\hat{h}(I) = \left\langle \frac{T(I)}{T^*}, \quad \frac{\beta(I)}{\beta^*} \right\rangle,$$

where

$$\hat{h}(I) = \langle \overline{T}(I),\ \overline{\beta}(I)\rangle, \quad \overline{T}(I) = \frac{T(I)}{T^*}, \quad \overline{\beta}(I) = \frac{\beta(I)}{\beta^*},$$

and the ideal point is $h^* = (1,1)$. Determining a new measure $\hat{\varepsilon}_1$, we obtain

$$\hat{\varepsilon}_1\big(h^*,\hat{h}(I)\big) = \left|1-\overline{T}(I)\right| + \left|1-\overline{\beta}(I)\right|.$$

But $1-\overline{T}(I) \le 0$ and $1-\overline{\beta}(I) \le 0$, so we obtain

$$\begin{aligned} \hat{\varepsilon}_1\big(h^*,\hat{h}(I)\big) &= \overline{T}(I) - 1 + \overline{\beta}(I) - 1 \\ &= \overline{T}(I) + \overline{\beta}(I) - 2. \end{aligned}$$

It can be observed that the function $\overline{T}(I)+\overline{\beta}(I)-2$ has a minimum value for the same I as the function $\overline{T}(I)+\overline{\beta}(I)$, so the component (-2) may be omitted and we have

$$\begin{aligned} \hat{\varepsilon}_1\big(h^*,\hat{h}^0(I)\big) &= \min_{I\in M(i^s,i^d)} \hat{\varepsilon}_1\big(h^*,\hat{h}(I)\big) \\ &= \min_{I\in M(i^s,i^d)} \left[\overline{T}(I)+\overline{\beta}(I)\right]. \end{aligned} \qquad (17)$$

The objective function from (17) is nonnegative and additive. Define a temporary function $H(I)$ as $H(I) = \overline{T}(I)+\overline{\beta}(I)$, so that

$$\begin{aligned} H(I) &= \frac{T(I)}{T^*} + \frac{\beta(I)}{\beta^*} \\ &= \frac{1}{T^*}\sum_{r=1}^{R_1} t^1_{i^{r-1},i^r} + \frac{1}{\beta^*}\sum_{r=1}^{R_1} \lambda(i^{r-1},i^r)t^1_{i^{r-1},i^r} \\ &= \sum_{r=1}^{R_1}\left(\frac{1}{T^*} + \frac{1}{\beta^*}\lambda(i^{r-1},i^r)\right)t^1_{i^{r-1},i^r}. \end{aligned}$$

In connection with the above, we can define the problem of finding a compromise path $I^c \in M(i^s,i^d)$ with $p=1$ as follows: Determine $I^c \in M(i^s,i^d)$ such that

$$H(I^c) = \min_{I\in M(i^s,i^d)} H(I). \qquad (18)$$

To optimally solve the problem (18) using Dijkstra's standard algorithm, we can use the following arc metafunction $mf(v,v')$:

$$mf(v,v') = \left(\frac{1}{T^*} + \frac{1}{\beta^*}\lambda(v,v')\right)t^1_{v,v'}, \quad \langle v,v'\rangle \in A_G.$$

The definition presented above has one more interesting property: If for each arc $\langle v,v'\rangle \in A_G$ we have $\lambda(v,v') = \lambda > 0$, then

$$\beta\left(I(i^s,i^d)\right) = \lambda\sum_{r=1}^{R_1} t^1_{i^{r-1},i^r}$$

and

$$\hat{h}(I) = \left\langle \frac{T(I)}{T^*}, \frac{\lambda T(I)}{\lambda T^*}\right\rangle,$$

so we solve a single-criterion problem with the criterion T.

Generally, if the arc functions $f_1, f_2, \dots, f_N$ are nonnegative, additive, i.e.,

$$F_n(I) = \sum_{r=0}^{R_1(I)-1} f_n\big(i_r(1), i_{r+1}(1)\big),$$

and all of them are minimized, then the measure ε_1 from (14) (for $p=1$) has the form

$$\begin{aligned} \varepsilon_1\big(h^*,h(I)\big) &= \sum_{n=1}^{N}\left|1-\frac{F_n(I)}{F_n^*}\right| \\ &= \sum_{n=1}^{N}\left|1-\frac{1}{F_n^*}\sum_{r=0}^{R_1-1} f_n(v_r,v_{r+1})\right|, \end{aligned}$$

where

$$\begin{aligned} F_n^* &= \min_{I\in M(i^s,i^d)} F_n(I), \\ h(I) &= \left\langle \frac{F_1(I)}{F_1^*}, \dots, \frac{F_N(I)}{F_N^*}\right\rangle, \end{aligned}$$

and

$$h^* = (\underbrace{1,1,\dots,1}_{N\text{times}}).$$

Because

$$1-\frac{F_n(I)}{F_n^*} \le 0$$

for all $n=\overline{1,N}$, we can write

$$\varepsilon_1\big(h^*,h(I)\big) = \sum_{n=1}^{N}\frac{F_n(I)}{F_n^*} - N.$$

It can be observed that the function

$$\sum_{n=1}^{N} \frac{F_n(I)}{F_n^*} - N$$

has a minimum value for the same I as the function

$$\sum_{n=1}^{N} \frac{F_n(I)}{F_n^*},$$

so the component $(-N)$ may be omitted. In this case, for a compromise result h^0 the following condition is satisfied (Problem $CS_{p=1}$):

$$\begin{aligned} \varepsilon_1\big(h^*, h^0(I)\big) &= \min_{I \in M(i^s, i^d)} \varepsilon_1\big(h^*, h(I)\big) \\ &= \min_{I \in M(i^s, i^d)} \sum_{n=1}^{N} \frac{F_n(I)}{F_n^*}. \end{aligned} \tag{19}$$

Thus we can solve Problem $CS_{p=1}$ optimally using Dijkstra's standard algorithm with the following arc metafunction (v, v'):

$$mf(v, v') = \sum_{n=1}^{N} \frac{f_n(v, v')}{F_n^*}, \quad \langle v, v' \rangle \in A_G. \tag{20}$$

The proof of the optimality of the resulting solution is presented in the next section (cf. Theorem 1). For $p > 1$ it is impossible to obtain a nonnegative, additive, linear form of the arc function, so it is rather impossible to solve the problem of finding a compromise solution optimally using Dijkstra's algorithm. In such cases, the problem can be formulated as a quadratic programming problem ($p = 2$) or a max-ordering problem ($p = \infty$) (Rana and Vickson, 1988; Warburton, 1987; Mote *et al.*, 1991). The method of compromise solutions with the parameter $1 \le p < \infty$ guarantees obtaining nondominated solutions, i.e., $I^c \in M^{ND}(i^s, i^d)$ (Ehrgott, 1997; Martins and Santos, 1999).

In Section 4.6 we define Problem $CS_{p=1}$ as linear programming problems *MOSP_LP1* and *MOSP_LP2*, Problem $CS_{p=2}$ as *MOSP_NP1*, and Problem $CS_{p=\infty}$ as *MOSP_NP2*.

4.3. Method with a Metacriterion Function. In this method we construct a function called the metacriterion function which "merges" all criteria. There are two main approaches to define this metacriterion function: in the first one the metacriterion function is in the form of a weighted average of criteria, and in the second one we minimize maximal deviations of criteria from their "ideal" values (some analogy to a compromise solution with the parameter $p = \infty$).

The metacriterion function (Type I) in the form of a weighted average of criteria with the weights α_n, $n = \overline{1, N}$ is defined as follows (under the assumptions that all criteria are minimized):

$$MF(I) = \sum_{n=1}^{N} \alpha_n F_n^*(I), \tag{21}$$

$$\begin{aligned} F_n^*(I) &= \frac{F_n(I)}{F_n^*} = \frac{F_n(I)}{\min\limits_{I \in M(i^s, i^d)} F_n(I)} \\ &= \frac{\sum\limits_{r=0}^{R_1 - 1} f_n(v_r, v_{r+1})}{\min\limits_{I \in M(i^s, i^d)} F_n(I)}, \quad n = \overline{1, N}, \end{aligned} \tag{22}$$

where $f_n(\cdot, \cdot)$ describes the n-th arc function of G, $f_n : A_G \to \mathbb{R}^+$, $n = \overline{1, N}$, R_1 stands for the number of nodes belonging to the path I. It is frequently assumed that the weights must satisfy

$$\alpha_n \in [0, 1], \quad n = \overline{1, N}, \quad \sum_{n=1}^{N} \alpha_n = 1.$$

This guarantees obtaining nondominated solutions, i.e., $I^{MF} \in M^{ND}(i^s, i^d)$ (Ehrgott, 1997; Martins and Santos, 1999).

The problem of finding an optimal solution (Problem *MF_I*) can be formulated as follows: Determine $I^{MF} \in M(i^s, i^d)$ such

$$MF(I^{MF}) = \min_{I \in M(i^s, i^d)} MF(I). \tag{23}$$

We can solve this problem using, e.g., Dijkstra's algorithm with the single arc metafunction

$$mf(v, v') = \sum_{n=1}^{N} \alpha_n \frac{f_n(v, v')}{F_n^*}, \quad \langle v, v' \rangle \in A_G, \tag{24}$$

and with the metacriterion function

$$MF(I) = \sum_{r=0}^{R_1 - 1} mf(v_r, v_{r+1}). \tag{25}$$

Theorem 1. *If the arc functions $f_1, f_2, \ldots, f_N$, $f_i : A_G \to \mathbb{R}^+$, $i = \overline{1, N}$ are additive, then we solve the problem (23) optimally using Dijkstra's algorithm with the arc meta-function (24). In this case, the metafunction (21) is equal to the metafunction (25).*

Proof. When the functions $f_1, f_2, \ldots, f_N$ are nonnegative, then the function (24) is nonnegative, and when the functions $f_1, f_2, \ldots, f_N$ are additive, then the cost of the path I is calculated as the sum of metacosts of arcs belonging to the path I. In this case, the assumptions of Dijkstra's algorithm regarding the arc function (nonnegativity and additivity) are satisfied, so we can use this function as the arc function in the algorithm. Now, using (25), we prove that $MF(I) = LF$ from (21) is equal to

$$\sum_{r=0}^{R_1-1} mf(v_r, v_{r+1}) = RG.$$

From (21) and (22) we obtain

$$\begin{aligned} LF = MF(I) &= \sum_{i=1}^{N} \alpha_i F_i^*(I) \\ &= \sum_{n=1}^{N} \alpha_n \frac{\sum_{r=0}^{R_1-1} f_n(v_r, v_{r+1})}{F_n^*} \\ &= \sum_{n=1}^{N} \sum_{r=0}^{R_1-1} \frac{\alpha_n}{F_n^*} f_n(v_r, v_{r+1}), \end{aligned}$$

and from (24) and (25) we obtain

$$\begin{aligned} RG &= \sum_{r=0}^{R_1(I)-1} mf(v_r, v_{r+1}) \\ &= \sum_{r=0}^{R_1-1} \sum_{n=1}^{N} \alpha_n \frac{f_n(v_r, v_{r+1})}{F_n^*} \\ &= \sum_{n=1}^{N} \sum_{r=0}^{R_1-1} \frac{\alpha_n}{F_n^*} f_n(v_r, v_{r+1}). \end{aligned}$$

Thus $LF = RG$. ■

Note that the arc function (20) is a special case of the arc function (24) (all $\alpha_i = 1$), and thus the problem (19) is a special case of the problem (23).

The complexity of the algorithm is the subject of Theorem 2.

Theorem 2. *The complexity of Dijkstra's modified algorithm (with Fibonacci's heaps) for solving the problem (23) using the arc metacriterion function (24) is equal to* $O(N(V \log V + A) + NA)$.

Proof. To evaluate the arc metafunction (24) for each arc, we must first solve the shortest path problem N times for each criterion: it takes time proportional to $O(N(V \log V + A))$ using Dijkstra's algorithm implemented with Fibonacci's heaps. Next, separately for each arc, we compute the value of the metafunction (24). For all arcs it takes time proportional to $\Theta(NA)$. Using Dijkstra's algorithm with arc metafunction (24), we compute the shortest path in time $O(V \log V + A)$, and thus the total time of the algorithm for solving the problem (23) is equal to $O(N(V \log V + A) + NA)$. ■

The metacriterion function (Type II) with the minimization of maximal deviations of criteria values from their "ideal" values can be defined using the following temporary function:

$$\begin{aligned} \overline{F}_n(I) &= \frac{F_n^*}{F_n(I)} = \frac{\min_{I \in M(i^s, i^d)} F_n(I)}{F_n(I)} \\ &= \frac{\min_{I \in M(i^s, i^d)} F_n(I)}{\sum_{r=0}^{R_1-1} f_n(v_r, v_{r+1})}, \quad n = \overline{1, N}. \end{aligned} \tag{26}$$

Note that $\overline{F}_n(I) \in (0, 1]$, $n = \overline{1, N}$, so the ideal point is equal to 1. Now, we can define the metacriterion function with the minimization of maximal deviations of criteria values from their "ideal" values (Problem *MF_2*) as follows:

$$u \to \min,$$

subject to

$$1 - \overline{F}_n(I) \leq u, \quad I \in M(i^s, i^d).$$

The additional variable u describes the maximal deviation of the values of the criteria functions $\overline{F}_n(I)$ from their "ideal" values (i.e., 1). From the condition $\overline{F}_n(I) \in (0, 1]$ it follows that $u \in [0, 1)$. In Section 4.6 we define this problem in detail as a mathematical programming problem (*MOSP_NP3*).

We shall show that Problem *MF_2* can be considered as that of finding a $(1 + \varepsilon)$-shortest path, $\varepsilon \geq 0$. The constraint $1 - \overline{F}_n(I) \leq u$ can be written as

$$F_n(I) \leq \frac{1}{1-u} F_n^*.$$

Taking into account the definition of the vector $(1 + \varepsilon)$-dominance (see (8)), we obtain

$$F_n(I) \leq (1 + \varepsilon) F_n^*,$$

that is,

$$\varepsilon = \frac{u}{1-u}.$$

Hence $u \to \min$ is equivalent to $\varepsilon \to \min$, because ε is an increasing function of u. Therefore, Problem *MF_2* can be solved by finding an $(1 + \varepsilon^*)$-shortest path, where ε^* is the smallest value of ε such that a $(1 + \varepsilon)$-shortest path exists (we use the following property of the $(1 + \varepsilon)$-shortest path: if any path I is a $(1 + \varepsilon)$-shortest path, then I is a $(1 + \varepsilon')$-shortest path for each $\varepsilon' \geq \varepsilon$). If we set the precision for u to m decimal places (m is a positive integer), then the algorithm *MF_2_half* is as follows:

Algorithm ***MF_2_half***

```
L:=0; R:=10^m; u*:=infinity;
while |L-R|>1 do
    u':= L + ceil ((R-L)/2);
    u:=u'/10^m;
```

```
    ε :=u/(1-u);
    Determine (1+ε)-shortest path
        from s to t;
    if (1+ε)-shortest path
        from s to t exists then
        R:=u'; u*:=u;
    else
        L:=u';
return u*;
```

If we denote by $T(\varepsilon)$ the complexity of an algorithm for finding a $(1+\varepsilon)$-shortest path between s and t (Warburton, 1987; Papadimitriou and Yannakakis, 2000), then Algorithm *MF_2_half* has complexity $O\left(\log_2 10^m T(\varepsilon)\right)$ (because the idea is similar to a binary-search for some value x in a sorted table with 10^m elements, where L and R denote the left and right indices of subtable ranges), respectively. For example, consider the weighted graph given in Fig. 1 with $s=1$, $t=5$. The "ideal" vector of criteria values is $c^* = (c_1^*, c_2^*, c_3^*) = (6,5,2)$. In the last column of Table 4, for each path I from $s=1$ to $t=5$ the smallest value of $(1+\varepsilon)$ such that

$$F(I) \overset{\varepsilon}{\leq} c^*$$

is calculated. Set $m=1$ (we want to calculate u with a precision of one decimal place) for Algorithm *MF_2_half*. In the first iteration, we get $L=0$, $R=10$, $u'=5$, $u=0.5$, $\varepsilon=1$. From Table 4 we see that a path (e.g., pA) for which $(1+\varepsilon)\leq 2$ exists. Hence this path is a $(1+(\varepsilon=1))$-shortest path from s to t and $R:=5$, $u^{:}=0.5$. In the second iteration, we get $L=0$, $R=5$, $u'=2$, $u=0.2$, $\varepsilon=0.25$. Because a path (e.g., pA) for which $(1+\varepsilon)\leq 1.25$ exists, it is a $(1+(\varepsilon=0.25))$-shortest path from s to t and $R:=2$, $u^*:=0.2$. In the third iteration, we get $L=0$, $R=2$, $u'=1$, $u=0.1$, $\varepsilon=1/9$. But no $(1+(\varepsilon=1/9))$-shortest path exists, and hence $L=1$, $R=2$ and we exit with $u^*=0.2$.

In Section 4.6 we define Problem *MF_1* as a linear programming problem (*MOSP_LP3*) and Problem *MF_2* as *MOSP_NP3*.

4.4. Method with the Hierarchization of Objective Functions. In this approach we order criteria functions according to their importance (in the set of criteria functions we set a lexicographic order), so that F_1 describes the most important criterion, F_2 – the second criterion with respect to importance, etc. A solution $I^h \in M_{j\leq N}(i^s,i^d) \subset M(i^s,i^d)$ is found by solving a sequence of single-criteria optimization problems starting from the most important criterion (with the index $j=1$, generating the set $M_1(i^s,i^d)$), and then taking into account the second criterion with respect to importance (the generating set $M_2(i^s,i^d)$), etc. The calculations are continued as long as we achieve M_N or at a previous stage $s\leq N$ we get $\overline{\overline{M}}_S = 1$. Each of the sets M_j tightens the previously obtained set M_{j-1} of acceptable solutions and is recurrently defined as

$$M_j(i^s,i^d) = \begin{cases} \left\{ I_j \in M_{j-1}(i^s,i^d) : F_j(I_j) = \min\limits_{I\in M_{j-1}(i^s,i^d)} F_j(I) \right\}, & j=\overline{1,N}, \\ M(i^s,i^d), & j=0. \end{cases}$$

The method of hierarchization of objective functions guarantees obtaining nondominated solutions, i.e., $I^h \in M^{ND}(i^s,i^d)$ (Ehrgott, 1997; Martins and Santos, 1999). For example, we consider a lexicographic solution (path) of the problem (3) with a vector objective function $F(I) = \langle T(I), P(I)\rangle$, where P is defined as follows:

$$P\big(I(i^s,i^d)\big) = \prod_{r=1}^{R_1} e^{-\lambda(i^{r-1},i^r)\sum\limits_{k=1}^{r} t^1_{i^{k-1},i^k}} = \prod_{r=1}^{R_1} q_{i^{r-1},i^r}\left(\sum_{k=1}^{r} t^1_{i^{k-1},i^k}\right),$$

$$q_{i^{r-1},i^r}(t) = e^{-\lambda(i^{r-1},i^r)t}.$$

There is one interesting question: How to find a solution following the order of the importance of the criteria (12) T, P? Korzan (1983a) proved (for $K=1$) that if inside the set $M^{ND}(i^s,i^d)$ there exist many shortest paths according to the criterion T with the same length T^*, then all of them have the same value of the P criterion. Acordingly, any node x with the same value of T on the part of the path from s can be considered at the next step of Dijkstra's algorithm. Hence, we can use Dijkstra's algorithm with the modifications presented in Table 2, where `d(x)` describes the value of the function T for the path from s to x, `c(x,y)` is equivalent to $c_{x,y}$, `p(x)` signifies the value of the function P for the path from s to x and `q(x,y,z)` is equivalent to $q_{x,y}(z)$.

A modification of Dijkstra's algorithm (*Dijkstra_Lex2*) has the same complexity as the original algorithm (with Fibonacci's heaps), that is, $O\left(V\log V + A\right)$. Generally, finding lexicographic solutions (paths) is NP-hard (Garey and Johson, 1979).

4.5. Method with Threshold Values of Some Criteria (Restricted Shortest Path Problem). Methods of threshold values (also known as restricted shortest path problems (RSPPs)) rely on the fact that some criteria functions have fixed critical values and they lighten the set of acceptable solutions. For example, the problem (12) could be written as follows: Determine $I^*(i^s,i^d) \in M\left(i^s,i^d\right)$ such that

$$P\left(I^*(i^s,i^d)\right) = \max_{I(i^s,i^d)\in M(i^s,i^d)} P\left(I(i^s,i^d)\right) \quad (27)$$

Table 2. Modification of Dijkstra's algorithm for finding lexicographic solution with the objectives T and P.

Dijkstra's standard algorithm	*Dijkstra_Lex2* algorithm

<pre>Dijkstra(
G = ⟨V_G, A_G⟩, [c(u,v)]_{V×V}, s, t)
 	
for each node v ∈ V_G do
  predecessor[v]:= null;
  d[v]:= +infinity;
 	
d[s]:= 0;
Q:= V_G;
while Q ≠ null do
   u:= Extract_Min(Q); /u is such node
               that d[u]= min{d[v]:v∈ Q}/
   Q:= Q \ u;
   if u=t then
     return;
   for each arc (u,v)∈ A_G starting from u
do
   if d[v] > d[u] + c(u,v) then
         d[v]:= d[u] + c(u,v);
         predecessor[v]:= u;</pre>
	<pre>Dijkstra_Lex2(
	G=⟨V_G, A_G⟩, [c(u,v)]_{V×V}, [q(u,v,z)]_{V×V×T}, s, t)
	
	for each node v ∈ V_G do
	   predecessor[v]:= null;
	   d[v]:= +infinity;
	   p[v]:= 0;
	p[s]:=1;
	d[s]:= 0;
	Q:= V_G;
	while Q ≠ null do
	   u:= Extract_Min(Q); /u is such node
	               that d[u]= min{d[v]:v∈ Q}/
	   Q:= Q \ u;
	   if u=t then
	        return;
	   for each arc (u,v) ∈ A_G starting from u
	do
	      if d[v] > d[u] + c(u,v) or
	         (d[v] = d[u] + c(u,v) and
	         p[v] < p[u] * q(u,v,d[v])) then
	           d[v]:= d[u] + c(u,v);
	           p[v]:= p[u] * q(u,v,d[v]);
	           predecessor[v]:= u;</pre>

with the additional restriction

$$T\left(I(i^s, i^d)\right) \le T_0,$$

where T_0 is a fixed threshold value of the criterion $T(\cdot)$. Warburton (1987) proposed an $O(V^2 Z \log V)$ algorithm for solving the RSPP problem for two objectives (with positive integers), where Z is an upper bound to the value of the second objective (the first objective is minimized). In Section 4.6 we define the RSPP problem as a mathematical programming one (*MOSP_LP4*).

4.6. Types of MOSP Problems Defined as Mathematical Programming Problems. For $K = 1$ we will use the formulation of the MOSP problem as the following linear programming one:

$$Cx \to \min \tag{28}$$

subject to

$$Bx = d, \quad x \ge 0. \tag{29}$$

Here $C = [c_{nj}]_{N\times A}$ is an objective matrix, $B = [b_{ij}]_{V\times A}$ is a transition matrix for the graph G, and $b_{ij} = 1$ when the j-th arc starts in the i-th node, $b_{ij} = -1$ when the j-th arc ends in the i-th node and $b_{ij} = 0$ otherwise. Furthermore, $d = [d_i]_{V\times 1}$ is a column vector which may have three values: $d_i = 1$ when $i = i^s$, $d_i = -1$ when $i = i^d$ and $d_i = 0$ otherwise. Moreover, $x = [x_j]_{A\times 1}$, $x_j \in \mathbb{R}^+ \subset \{0\}$, and 'min' describes the minimum in a vector sense (in the sense of the relation R^D). The i-th node has its equivalent in the set V_G, $i = \overline{1, V}$, and the j-th arc has its equivalent in the set A_G, $j = \overline{1, A}$. Each cost c_{nj} for the j-th arc has its equivalent in the value of the arc function $f_n(v, v')$, $\langle v, v'\rangle \in A_G$. For the case $N = 1$ we have the classical definition of the shortest path problem as a linear programming one (because of the total unimodularity of the matrix B and the vector d). Sometimes, we will use the following extended, equivalent form of the problem (28), (29):

$$\sum_{j=1}^{A} c_{nj} x_j \to \min, \quad n = \overline{1, N} \tag{30}$$

subject to

$$\begin{aligned} &\sum_{j=1}^{A} b_{ij} x_j = d_i, && i = \overline{1, V}, \\ &x_j \ge 0, && j = \overline{1, A}. \end{aligned} \tag{31}$$

The problem of finding a compromise solution with the parameter $p = 1$, however nonlinear in its nature, can be formulated as a linear programming one. Using the notation introduced in (30) and (31), the metric (14) can be written as follows:

$$\sum_{n=1}^{N}\left|1-\frac{1}{c_n^*}\sum_{j=1}^{A}c_{nj}x_j\right| \to \min,$$

where $c_n^* \equiv F_n^*$. We adopt the following notation:

$$\overline{z}_n = \max\left\{0, 1-\frac{1}{c_n^*}\sum_{j=1}^{A}c_{nj}x_j\right\}, \quad n=\overline{1,N},$$

$$\overline{\overline{z}}_n = \max\left\{0, \frac{1}{c_n^*}\sum_{j=1}^{A}c_{nj}x_j - 1\right\}, \quad n=\overline{1,N}.$$

Then for each $n=\overline{1,N}$ the following conditions are satisfied:

$$\left|1-\frac{1}{c_n^*}\sum_{j=1}^{A}c_{nj}x_j\right| = \overline{z}_n + \overline{\overline{z}}_n,$$

$$1-\frac{1}{c_n^*}\sum_{j=1}^{A}c_{nj}x_j = \overline{z}_n - \overline{\overline{z}}_n,$$

$$\overline{z}_n \geq 0, \quad \overline{\overline{z}}_n \geq 0, \quad \overline{z}_n \cdot \overline{\overline{z}}_n = 0.$$

For this reason we obtain the following linear programming problem (*MOSP_LP1*):

$$\sum_{n=1}^{N}\overline{z}_n + \overline{\overline{z}}_n \to \min$$

subject to

$$1-\frac{1}{c_n^*}\sum_{j=1}^{A}c_{nj}x_j = \overline{z}_n - \overline{\overline{z}}_n, \quad n=\overline{1,N},$$

$$\overline{z}_n \geq 0, \quad \overline{\overline{z}}_n \geq 0, \quad \overline{z}_n \cdot \overline{\overline{z}}_n = 0, \quad n=\overline{1,N},$$

and (31).

We omit the conditions $\overline{z}_n \cdot \overline{\overline{z}}_n = 0$, $n=\overline{1,N}$, but it can be shown that they do not extend the set of optimal solutions. The presented problem can be solved using, e.g, the simplex algorithm. But the problem can be of a large scale (the number of variables equals $N+A$, and the number of boundaries is $N+V$) and the effectiveness of solving this problem (using a simplex or an ellipsoidal algorithm) is rather unacceptable. According to the discussion of Section 4.2 and Eqn. (19), Problem $CS_{p=1}$ can be also defined as follows (*MOSP_LP2*):

$$\sum_{n=1}^{N}\frac{1}{c_n^*}\sum_{j=1}^{A}c_{nj}x_j \to \min$$

subject to (31).

Problem $CS_{p=2}$ of finding a compromise solution with the parameter $p=2$ (*MOSP_NP1*) is as follows:

$$\sum_{n=1}^{N}\left(1-\frac{1}{c_n^*}\sum_{j=1}^{A}c_{nj}x_j\right)^2 \to \min$$

subject to (31). Unfortunately, the criterion function makes it nonlinear.

Problem $CS_{p=\infty}$ of finding a compromise solution with the parameter $p=\infty$ (*MOSP_NP2*), known as the max-ordering problem, can be defined as follows:

$$\max_{n\in\{1,\dots,N\}}\left|1-\frac{1}{c_n^*}\sum_{j=1}^{A}c_{nj}x_j\right| \to \min$$

subject to (31). The notation "max" in the criterion function makes the problem nonlinear.

The method with a metacriterion function of Type I (*MOSP_LP3*) is defined as follows:

$$\sum_{n=1}^{N}\frac{\alpha_n}{c_n^*}\sum_{j=1}^{A}c_{nj}x_j \to \min$$

subject to

$$\sum_{n=1}^{N}\alpha_n = 1, \quad \alpha_n \geq 0, \quad n=\overline{1,N}$$

and (31).

To define the MOSP problem with a metacriterion function of Type II, note that the function $\overline{F}_n(I)$ from (26) is equivalent to $c_n^*/\sum_{j=1}^{A}c_{nj}x_j$. Hence we obtain Problem *MOSP_NP3*:

$$u \to \min$$

subject to

$$1-\frac{c_n^*}{\sum_{j=1}^{A}c_{nj}x_j} \leq u, \quad n=\overline{1,N}$$

and (31). The first type of constraints makes the problem nonlinear.

The method with critical values of criteria (*MOSP_LP4*), also known as the restricted shortest path problem, can be formulated as follows:

$$\sum_{j=1}^{A}c_{Lj}x_j \to \min$$

subject to

$$\sum_{j=1}^{A}c_{ij}x_j \leq g_i, \quad i=\overline{1,N}, \quad i \neq L$$

Table 3. Properties of MOSP problems formulated as mathematical programming ones.

Problem	Type of mathematical programming problem	Number of decision variables	Number of constraints
MOSP_LP1	Linear	$2N + A$	$V + N$
MOSP_LP2	Linear	A	V
MOSP_LP3	Linear	A	V
MOSP_LP4	Linear	A	$V + N - 1$
MOSP_NP1	Nonlinear	A	V
MOSP_NP2	Nonlinear	A	V
MOSP_NP3	Nonlinear	$A + 1$	$V + N$

and (31), where $g = (g_1, g_i, \ldots, g_N)_{i \neq L}$ collects threshold values of individual criteria, and L denotes the index of the criterion to be minimized. Note that if any component of g is not integer, then the constraint $x_j \geq 0$, $j = \overline{1, A}$ must be replaced by $x_j \in \{0, 1\}$, $j = \overline{1, A}$.

In Table 3 we present the properties of MOSP problems formulated as mathematical programming ones.

5. Numerical Examples

In Fig. 1 we present a graph which will be used as a running example of defined MOSP problems with a three-dimensional vector of costs (objectives). The values of all functions are minimized.

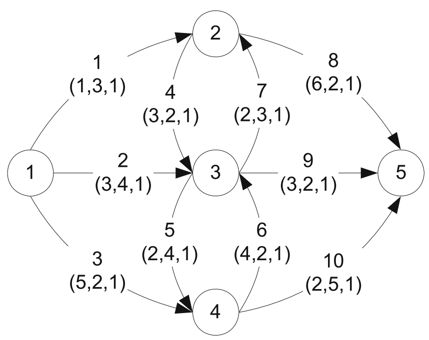

Fig. 1. Exemplary graph with multidimensional costs: on the top of each arc its label is given while on the bottom three-component arc cost are displayed.

In Table 4 we present the set of paths from $s = 1$ to $t = 5$ for the graph from Fig. 1 and their multidimensional properties. In the last row of the table, optimal costs for each of the objectives are presented ($c^* = (6, 5, 2)$). In the last column of Table 4, for each path I from $s = 1$ to $t = 5$ the smallest value of $(1 + \varepsilon)$ such that $F(I) \underset{\varepsilon}{\leq} c^*$ is calculated. For example, for pA we have

$$1 + \varepsilon = \max\left\{7/6, 5/5, 2/2\right\} = 7/6.$$

Table 4. Set of paths from $s = 1$ to $t = 5$ for the graph from Fig. 1 and their multidimensional properties.

Path name I	Path as sequence of nodes	Cost vector $F(I)$ of path	$1 + \varepsilon$
pA	1-2-5	(7, 5, 2)	7/6
pB	1-2-3-5	(7, 7, 3)	3/2
pC	1-2-3-4-5	(8, 14, 4)	14/5
pD	1-3-5	(6, 6, 2)	6/5
pE	1-3-2-5	(12, 8, 3)	12/6
pF	1-3-4-5	(7, 9, 3)	9/5
pG	1-4-5	(7, 7, 2)	7/5
pH	1-4-3-5	(12, 6, 3)	12/6
pI	1-4-3-2-5	(17, 9, 4)	17/6
	Vector of optimal costs:	$c_1^* = 6, \quad c_2^* = 5, \quad c_3^* = 2$	

Table 5 contains optimal multidimensional paths for the graph from Fig. 1 ($s = 1$, $t = 5$) using different types of MOSP problems.

In Figs. 2–4 we present weighted terrain-based grid graphs with the dimensions of 50×200 nodes (squares) representing the neighbourhood of Radom (a city in Poland). Each of the graphs has $A \approx 3,95V$ arcs, because only north-east-south-west moves are permitted from a node. Such graphs represent, e.g., a model of the battlefield in computer simulation games (Tarapata, 2003). For this example, each terrain square has a size of 200×200 m so that the graphs represent a piece of terrain with the dimension 10×40 km. Colours represent criteria values: c_1 for Fig. 2—the light colour of a node (square) describes open terrain (well passable), the dark colour describes obstacles (forests, lakes, rivers, buildings): the darkest colour represents the least passable terrain; c_2 for Fig. 3—the colour of a node (square) describes the ability to camouflage: the darker the colour, the smaller the ability to camouflage; c_3 for Fig. 4—the values of the criterion c_3 equal 1 for all nodes. The white colour in all

Table 5. Optimal multidimensional paths for the graph of Fig. 1 ($s = 1, t = 5$).

Problem	Optimal path	Cost of path
MF_1, $\alpha_n = 1/3,\ n = \overline{1,3} \Leftrightarrow$ *MOSP_LP3*	pA	1.045
MF_1, $\alpha_1 = 0.66,\ \alpha_2 = 0.17,\ \alpha_3 = 0.17 \Leftrightarrow$ *MOSP_LP3*	pD	1.034
$CS_{p=1} \Leftrightarrow$ *MOSP_LP2*	pA	3.167
RSPP $\Leftrightarrow$ *MOSP_LP4*, $L = 1,\ g_2 = 1.2c_2^*,\ g_3 = 1.2c_3^*$	pD	6.0
RSPP $\Leftrightarrow$ *MOSP_LP4*, $L = 1,\ g_2 = 1.1c_2^*,\ g_3 = 1.1c_3^*$	pA	7.0
RSPP $\Leftrightarrow$ *MOSP_LP4*, $L = 1, g_2 = c_2^*, g_3 = c_3^*$	Null	+infinity
$CS_{p=2} \Leftrightarrow$ *MOSP_NP1*	pA	0.139
$CS_{p=\infty} \Leftrightarrow$ *MOSP_NP2*	$pA,\ pD$	0.333
MOSP_NP3	pA	$u = 1/6$

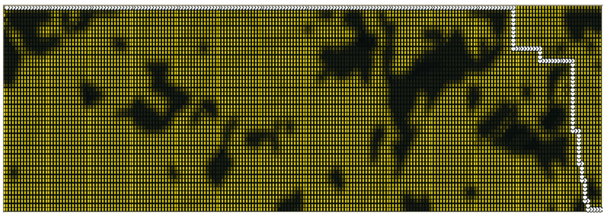

Fig. 2. Weighted terrain-based grid graph with 50×200 nodes (squares). Colours represent the values of the criterion c_1: the light colour of the nodes (square) corresponds to open terrain, and the dark colour describes obstacles (forests, lakes, rivers, buildings). The white colour describes an optimal path from the top-left corner to the bottom-right one.

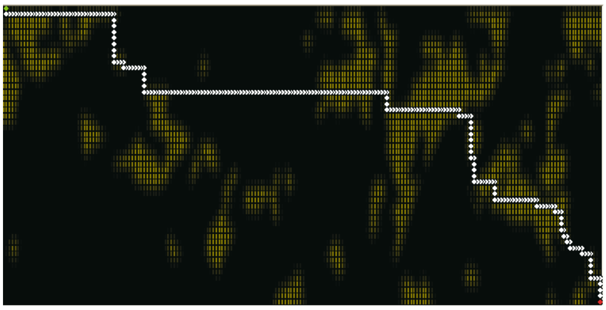

Fig. 3. Weighted terrain-based grid graph with 50×200 nodes (squares). The colour represents the values of the criterion c_2: the colour of the node (square) describes the ability to camouflage: the darker the colour, the lower the ability to camouflage. The white colour marks the optimal path from the top-left corner to the bottom-right one.

figures describes an optimal path from the top-left corner to the bottom-right one. Note that finding an optimal path in the sense of c_1 gives the fastest path, while c_2 gives the best "camouflaged" path, and c_3 yelds a shortest geometric path (with north-east-south-west moves from a node only). Without loss of generality, we can assume that the functions c_1, c_2, c_3 are described on the nodes

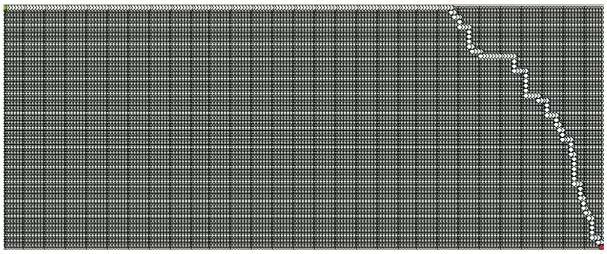

Fig. 4. Weighted terrain-based grid graph with 50×200 nodes (squares). All weights are identical (the value of the criterion c_3 equals 1). The white colour marks the optimal path from the top-left corner to the bottom-right one.

(squares) instead of the arcs. If it is necessary to obtain a graph with arc functions, we can construct a dual graph $GT = \langle V_{GT}, A_{GT} \rangle$ to the analysed graph $G = \langle V_G, A_G \rangle$, where $V_{GT} = A_G$ and each arc $(a, b) \in A_{GT} \subset A_G \times A_G$ is created when two arcs a, b in G have a common node (i.e., they are simultaneously incident with any node), then in GT the functions c_1, c_2, c_3 are described on arcs.

In Table 6 we present experimental results of average running times (in seconds) of Dijkstra's modified algorithm and CPLEX 7.0 for Problem *MF_1* ($\alpha_i = 1/N,\ i = 1, \dots, N$). Graphs with the numbers of nodes equal to $1000x$ ($x = 1, 2, \dots, 10$) are cut from the graph with 50×200 nodes (Figs. 2–4) and have $50 \times (20x)$ nodes. We can see a clear advantage of Dijkstra's modified algorithm with relation to CPLEX 7.0 solving Problem *MF_1* as the linear programming problem *MOSP_LP3*. Using Dijkstra's modified algorithm with its fast implementations is time effective. It is especially visible in Fig. 5, where we present the base 10 logarithm of the average running times (in milliseconds) of these two algorithms.

Figure 6 presents dependencies between the average running times (in milliseconds) of the CPLEX 7.0 solver and the *beta* coefficient for solving Problem *MOSP_LP4* for two graphs with $V = 1000(50 \times 20)$ and $V = 2000(50 \times 40)$ nodes. In Problem *MOSP_LP4* we min-

Table 6. Average running times (in seconds) of Dijkstra's modified algorithm and CPLEX 7.0 for Problem *MF_1* ($\alpha_i = 1/N, i = 1, \dots, N$).

Number of nodes (V)	Dijkstra's modified alg.			*MF_1* solved as *MOSP_LP3*		
	$N = 1$	$N = 2$	$N = 3$	$N = 1$	$N = 2$	$N = 3$
1000	0.03	0.08	0.11	0.76	2.31	4.39
2000	0.10	0.29	0.38	2.82	8.81	12.40
3000	0.25	0.71	0.96	6.52	21.20	29.14
4000	0.37	1.10	1.47	16.40	52.55	72.30
5000	0.59	1.74	2.33	30.41	98.12	136.22
6000	0.86	2.55	3.42	50.79	161.94	225.67
7000	1.16	3.44	4.59	74.61	238.27	333.80
8000	1.55	4.57	6.12	109.24	348.13	483.76
9000	1.96	5.82	7.77	134.78	432.47	620.94
10000	2.43	7.24	9.66	179.61	564.42	790.97

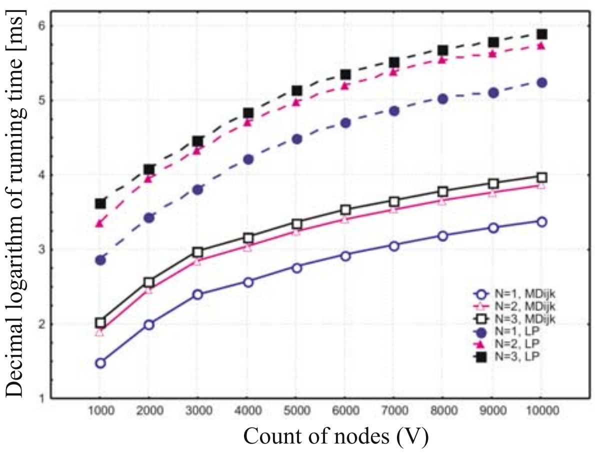

Fig. 5. Base 10 logarithm of the average running times (in milliseconds) of Dijkstra's modified algorithm (Problem *MF_1* ⇔ MDijk) and CPLEX 7.0 (Problem *MOSP_LP3* ⇔ LP) ($\alpha_i = 1/N$, $i = 1, \dots, N$).

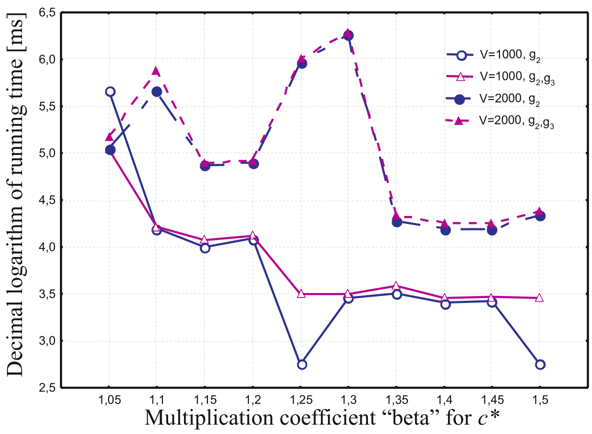

Fig. 6. Base 10 logarithm of the average running times (in milliseconds) of the CPLEX 7.0 solver solving Problem *MOSP_LP4* for two graphs with $V = 1000$ and $V = 2000$ nodes, $g_2 = beta \cdot c_2^*$ and (g_3 = infinity, $g_3 = beta \cdot c_3^*$).

imize the criterion c_1 subject to upper bounds (g_2 and g_3) on the values of the criteria c_2 and c_3 as follows: $g_2 = beta \cdot c_2^*$ and (g_3 = infinity, $g_3 = beta \cdot c_3^*$), where $c_2^* = 6964$, $c_3^* = 68$ for $V = 1000$ and $c_2^* = 6061$, $c_3^* = 88$ for $V = 2000$.

In Fig. 7 we present dependencies between the values of the objective function and the coefficient *beta* for Problem *MOSP_LP4*. Note that, generally, the greater the value of *beta*, the smaller the running time of the model in the CPLEX solver (and the smaller the value of the objective functions, Fig. 7) but the functions from Fig. 6 are not monotonic. The values of the running times for Problem *MOSP_LP4* are several times greater than those for Problem *MOSP_LP3* solved using the CPLEX solver (compare Figs. 6 and 5). For example, the running time for $V = 2000$ is about $10^5/10^{2.8}$ times greater than that for solving Problem *MOSP_LP3*. These results are clear: the smaller the restrictions on the criteria c_2 and c_3 (this means: the greater the value of *beta*), the smaller the running time. Moreover, the greater values of running time result from the fact that $g_i = beta \cdot c_i^*$ is not integer (except for $beta = 1.25$ and $beta = 1.5$ for $c_2^* = 6964$, V=1000), and *MOSP_LP4* (as a linear programming problem) becomes a binary programming problem which is harder to solve. For $beta \geq 1.35$ the value of the objective function (based on c_1) does not change because it achieves an optimal value ($c_1^* = 605$ for $V = 1000$, $c_1^* = 713$ for $V = 2000$).

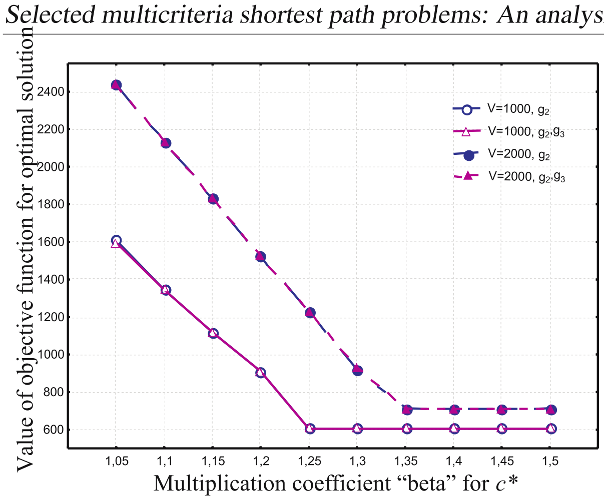

Fig. 7. Values of the objective functions for Problem *MOSP_LP4* for two graphs with $V = 1000$ and $V = 2000$ nodes, $g_2 = beta \cdot c_2^*$ and ($g_3 =$ infinity, $g_3 = beta \cdot c_3^*$).

6. Summary

Any algorithm solving the multiobjective shortest path problem is at least exponential in the worst case analysis, but we can use specific effective approaches for special MOSP problems. In the paper we focused on the analysis of the complexity of selected MOSP problems and showed how we can use modifications and advantages of fast implementations of Dijkstra's algorithm (using effective data structures such as, e.g., Fibonacci's heaps, d-ary heaps) in order to optimally solve them. Experimental results of the computational times for the presented approach (especially Dijkstra's modified algorithm) in Section 5 confirm their good effectiveness when solving selected MOSP problems. The models and methods described in the paper were chosen from numerous approaches. Problems such as determining disjoint paths (Li *et al.*, 1992, Schrijver and Seymour, 1992; Tarapata, 1999; 2000), stochastic network dependencies (Sigal *et al.*, 1980; Korzan, 1982; 1983a; 1983b; Loui, 1983), time-dependencies in networks (Bernstein *et al.*, 1997; Djidjev *et al.*, 1995; Sherali *et al.*, 1998) in the multicriteria context were only indicated here. These problems are often very complicated (computationally, too), and we use different methods for solving them, such as exact algorithms specifically designed for the problem, label setting or label correcting methods, algorithms based on dynamic programming, algorithms based on branch and bound, interactive methods, heuristics specifically designed for the problem, simulated annealing algorithms, tabu search algorithms, genetic or evolutionary algorithms, greedy randomized adaptative search procedures, goal programming, approximation algorithms with worst case performance bound, methods based on linear programming, etc.

References

Bernstein D. and Kelly S. (1997): *Solving a best path problem when the value of time function is nonlinear.* — Research paper, Princeton University.

Brumbaugh-Smith J. and Shier D. (1989): *An empirical investigation of some bicriterion shortest path algorithms.* — Europ. J. Oper. Res., Vol. 43, No. 2, pp. 216–224.

Carraway R.L., Morin T.L. and Moskovitz H. (1990): *Generalized dynamic programming for multicriteria optimization.* — Europ. J. Oper. Res., Vol. 44, No. 1, pp. 95–104.

Cidon I., Rom R. and Shavitt Y. (1997): *Multi-path routing combined with resource reservation.* — Proc. 16th Annual Joint Conf. IEEE *Computer and Communications Societies, INFOCOM'97*, Kobe, Japan, pp. 92–100.

Cidon I., Rom R. and Shavitt Y. (1999): *Analysis of multi-path routing.* — IEEE/ACM Trans. Netw., Vol. 7, No. 6, pp. 885–896.

Climaco J.C.M. and Martins E.Q.V. (1982): *A bicriterion shortest path algorithm.* — Europ. J. Oper. Res., Vol. 11, No. 1, pp. 399–404.

Climaco J., Craveirinha J. and Pascoal M. (2002): *A bicriterion approach for routing problems in multimedia networks.* — Networks, Vol. 41, No. 4, pp. 206–220.

Corley H.W. and Moon I.D. (1985): *Shortest paths in networks with vector weights.* — J. Optim. Th. Applic., Vol. 46, No. 1, pp. 79–86.

Coutinho-Rodrigues J.M., Climaco J.C.N. and Current J.R. (1999): *An intercative biobjective shortest path approach: Searching for unsupported nondominated solutions.* — Comput. Oper. Res., Vol. 26, No. 8, pp. 789–798.

Cox R.G. (1984): *Routing of hazardous material.* — Ph.D. thesis, School of Civil and Environmental Engineering, Cornell University, Ithaca, NY.

Current J.R., ReVelle C.S. and Cohon J.L. (1990): *An interactive approach to identify the best compromise solution for two objective shortest path problems.* — Comput. Oper. Res., Vol. 17, No. 2, pp. 187–198.

Dial R. (1979): *A model and algorithm for multicriteria route-mode choice.* — Transport. Res., Vol. 13B, No. 4, pp. 311–316.

Djidjev H., Pantziou G. and Zaroliagis C.D. (1995): *On-line and dynamic algorithms for shortest path problems.* — Lect. Notes Comput. Sci., Vol. 900, pp. 193–204.

Ehrgott M. (1997): *Multiple Criteria Optimization—Classification and Methodology.* — Aachen: Shaker Verlag.

Ehrgott M. and Gandibleux V. (2000): *A survey and annotated bibliography of multiobjective combinatorial optimization.* — OR Spektrum, Vol. 22, pp. 425–460.

Ehrgott M. and Gandibleux X. (Eds.) (2002): *Multiple Criteria Optimization—State of the Art Annotated Bibliographic Surveys.* — Boston, MA: Kluwer.

Engberg D., Cohon J. and ReVelle C. (1983): *Multiobjective siting of a natural gas pipeline*. — Proc. 11th Ann. Pittsburgh Conf. Modeling and Simulation, Pittsburgh, USA, pp. 137–145.

Eppstein D. (1999): *Finding the K shortest paths*. — SIAM J. Comput., Vol. 28, No. 2, pp. 652–673.

Ergun F., Sinha R. and Zhang L. (2002): *An improved FPTAS for restricted shortest path*. — Inf. Process. Lett., Vol. 83, No. 5, pp. 287–291.

Fujimura K. (1996): *Path planning with multiple objectives*. — IEEE Robot. Automat. Mag., Vol. 3, No. 1, pp. 33–38.

Gabrel V. and Vanderpooten D. (1996): *Generation and selection of efficient paths in a multiple criteria graph: The case of daily planning the shots taken by a satellite with an interactive procedure*. — Tech. Rep. 136, LAMSADE, Universitè Paris Dauphine, France.

Garey M. and Johnson D. (1979): *Computers and Intractability: A Guide to the Theory of NP Completeness*. — San Francisco, CA: W.H. Freeman.

Golden B.L. and Skiscim C.C. (1989): *Solving k-shortest and constrained shortest path problems efficiently*. — Netw. Optim. Appl. 20, Texas A&M University, College Station.

Halder D.K. and Majumber A. (1981): *A method for selecting optimum number of stations for a rapid transit line: An application in Calcutta tube rail*, In: Scientific Management of Transport Systems (N.K. Jaiswal, Ed.). — New York: North-Holland, pp. 97–108.

Hansen P. (1979a): *Bicriterion path problems*, In: Multiple Criteria Decision Making Theory and Application (G. Fandel and T. Gal, Ed.). — Berlin: Springer, pp. 109–127.

Hansen P. (1979b): *Bicriterion Path Problems*. — Proc. 3rd Conf. Multiple Criteria Decision Making—Theory and Applications, Lect. Notes Econ. Math. Syst., Vol. 117, pp. 109–127.

Hartley R. (1985): *Vector optimal routing by dynamic programming*, In: Mathematics of Multiobjective Optimization (P. Serahni, Ed.). — Vienna: Springer, pp. 215–224.

Hassin R. (1992): *Approximation schemes for the restricted shortest path problem*. — Math. Oper. Res., Vol. 17, No. 1, pp. 36–42.

Henig M.I. (1985): *The shortest path problem with two objective functions*. — Europ. J. Oper. Res., Vol. 25, No. 22, pp. 281–291.

Henig M.I. (1994): *Efficient interactive methods for a class of multiattribute shortest path problems*. — Manag. Sci., Vol. 40, No. 7, pp. 891–897.

Huarng F., Pulat P.S. and Shih L. (1996): *A computational comparison of some bicriterion shortest path algorithms*. — J. Chinese Inst. Ind. Eng., Vol. 13, No. 2, pp. 121–125.

Kerbache L. and Smith J. (2000): *Multi-objective routing within large scale facilities using open finite queueing networks*. — Europ. J. Oper. Res., Vol. 121, No. 1, pp. 105–123.

Korzan B. (1982): *Method of determining compromise paths in unreliable directed graphs*. — Bulletin of the Military University of Technology, Warsaw, Vol. XXXI, No. 7, pp. 21–36, (in Polish).

Korzan B. (1983a): *Method of determining nondominated paths in unreliable directed graphs*. — Bulletin of the Military University of Technology, Warsaw, Vol. XXXII, No. 11, pp. 21–33, (in Polish).

Korzan B. (1983b): *Paths optimization in unreliable directed graphs*. — Bulletin of the Military University of Technology, Warsaw, Vol. XXXII, No. 6, pp. 69–85, (in Polish).

Kostreva M.M. and Wiecek M.M. (1993): *Time dependency in multiple objective dynamic programming*. — J. Math. Anal. Appl., Vol. 173, No. 1, pp. 289–307.

Li C.L., McCormick S.T. and Simchi-Levi D. (1992): *Finding disjoint paths with different path-costs: Complexity and algorithms*. — Networks, Vol. 22, No. 7, pp. 653–667.

Lorenz D.H. and Raz D. (2001): *A simple efficient approximation scheme for the restricted shortest path problem*. — Oper. Res. Letters, Vol. 28, No. 1, pp. 213–219.

Loui R.P. (1983): *Optimal paths in graphs with stochastic or multidimensional weights*. — Comm. ACM, Vol. 26, No. 9, pp. 670–676.

Martins E.Q.V. (1984): *On a multicriteria shortest path problem*. — Europ. J. Oper. Res., Vol. 16, No. 1, pp. 236–245.

Martins E.Q.V. and Climaco J.C.N. (1981): *On the determination of the nondominated paths in a multiobjective network problem*. — Meth. Oper. Res., Vol. 40, pp. 255–258.

Martins E.Q.V. and Santos J.L.E. (1999): *The labelling algorithm for the multiobjective shortest path problem*. — Tech. Rep., Universidade de Coimbra, Portugal.

Modesti P. and Sciomachen A. (1998): *A utility measure for finding multiobjective shortest paths in urban multimodal transportation networks*. — Europ. J. Oper. Res., Vol. 111, No. 3, pp. 495–508.

Mote J., Murthy I. and Olson D. (1991): *A parametric approach to solving bicriterion shortest path problems*. — Europ. J. Oper. Res., Vol. 53, No. 1, pp. 81–92.

Murthy I. and Her S.S. (1992): *Solving min-max shortest-path problems on a network*. — Naval Res. Log., Vol. 39, No. 1, pp. 669–683.

Murthy I. and Olson D. (1994): *An interactive procedure using domination cones for bicriterion shortest path problems*. — Europ. J. Oper. Res., Vol. 72, No. 2, pp. 418–432.

Papadimitriou C. and Yannakakis M. (2000): *On the Approximability of Trade-offs and Optimal Access of Web Sources*. — Proc. 41st Symp. Foundations of Computer Science, FOCS, Redondo Beach, CA, pp. 86–92.

Pelegrin B. and Fernandez P. (1998): *On the sum-max bicriterion path problem*. — Comput. Oper. Res., Vol. 25, No. 12, pp. 1043–1054.

Rana K. and Vickson R.G. (1988): *A model and solution algorithm for optimal routing of a time-chartered containership*. — Transport. Sci., Vol. 22, No. 2, pp. 83–96.

Sancho N.G.F. (1988): *A new type of multiobjective routing problem*. — Eng. Optim., Vol. 14, No. 1, pp. 115–119.

Schrijver A. (2004): *Combinatorial Optimization*. — Berlin: Springer.

Schrijver A. and Seymour P. (1992): *Disjoint paths in a planar graph—A general theorem*. — SIAM J. Discr. Math., Vol. 5, No. 1, pp. 112–116.

Sherali H., Ozbay K. and Subramanian S. (1998): *The time-dependent shortest pair of disjoint paths problem: Complexity, models and algorithms*. — Networks, Vol. 31, No. 4, pp. 259–272.

Sigal E., Pritsker A. and Solberg J. (1980): *The stochastic shortest route problem*. — Oper. Res., Vol. 28, No. 5, pp. 1122–1129.

Silva R. and Craveirinha J. (2004): *An Overview of Routing Models for MPLS Networks*. — Proc. 1st Workshop *Multicriteria Modelling in Telecommunication Network Planning and Design*, Faculty of Economics of the University of Coimbra, Coimbra, Portugal, pp. 17–24.

Skriver A.J.V. and Andersen K.A. (2000): *A label correcting approach for solving bicriterion shortest path problems*. — Comput. Oper. Res., Vol. 27, No. 6, pp. 507–524.

Suurballe J.W. and Tarjan R.E. (1984): *A quick method for finding shortest pairs of disjoint paths*. — Networks, Vol. 14, No. 2, pp. 325–336.

Tarapata Z. (1999): *Optimization of many tasks sending in an unreliable parallel computing system*. — Proc. NATO Reg. Conf. *Military Communication and Information Systems*, Zegrze, Poland, Vol. III, pp. 245–254.

Tarapata Z. (2000): *Multi-paths optimization in unreliable time-dependent networks*. — Proc. NATO Reg. Conf. *Military Communication and Information Systems*, Zegrze, Poland, Vol. I, pp. 181–189.

Tarapata Z. (2003): *Military route planning in battlefield simulation: effectiveness problems and potential solutions*. — J. Telecomm. Inf. Technol., No. 4, pp. 47–56.

Tsaggouris G. and Zaroliagis Ch. (2005): *Improved FPTAS for multiobjective shortest paths with applications*. — Tech. Rep. No. TR 2005/07/03, Research Academic Computer Technology Institute, Petras, Greece.

Tung C.T. and Chew K.L. (1988): *A bicriterion Pareto-optimal path algorithm*. — Asia-Pacific J. Oper. Res., Vol. 5, No. 2, pp. 166–172.

Tung C.T. and Chew K.L. (1992): *A multicriteria Pareto-optimal path algorithm*. — Europ. J. Oper. Res., Vol. 62, No. 2, pp. 203–209.

Vassilvitskii S. and Yannakakis M. (2004): *Efficiently computing sufficient trade-off curves*. — Lect. Notes Comput. Sci., Vol. 3142, pp. 1201–1213.

Warburton A. (1987): *Approximation of Pareto optima in multiple-objective, shortest-path problems*. — Oper. Res., Vol. 35, No. 1, pp. 70–79.

White D.J. (1987): *The set of efficient solutions for multiple objective shortest path problems*. — Comput. Oper. Res., Vol. 9, No. 2, pp. 101–107.

Wijeratne A.B., Turnquist M.A. and Mirchandani P.B. (1993): *Multiobjective routing of hazardous materials in stochastic networks*. — Europ. J. Oper. Res., Vol. 65, No. 1, pp. 33–43.

Received: 23 November 2006
Revised: 3 April 2007