Int. J. Appl. Math. Comput. Sci., 2017, Vol. 27, No. 1, 133–155
DOI: 10.1515/amcs-2017-0010

A RELATION OF DOMINANCE FOR THE BICRITERION BUS ROUTING PROBLEM

JACEK WIDUCH [a]

[a] Institute of Informatics
Silesian University of Technology, ul. Akademicka 16, 44-100 Gliwice, Poland
e-mail: jacek.widuch@polsl.pl

A bicriterion bus routing (BBR) problem is described and analysed. The objective is to find a route from the start stop to the final stop minimizing the time and the cost of travel simultaneously. Additionally, the time of starting travel at the start stop is given. The BBR problem can be resolved using methods of graph theory. It comes down to resolving a bicriterion shortest path (BSP) problem in a multigraph with variable weights. In the paper, differences between the problem with constant weights and that with variable weights are described and analysed, with particular emphasis on properties satisfied only for the problem with variable weights and the description of the influence of dominated partial solutions on non-dominated final solutions. This paper proposes methods of estimation a dominated partial solution for the possibility of obtaining a non-dominated final solution from it. An algorithm for solving the BBR problem implementing these estimation methods is proposed and the results of experimental tests are presented.

Keywords: multicriteria optimization, set of non-dominated solutions, bicriterion shortest path problem, variable weights, label correcting algorithm, transportation problem.

1. Introduction

The shortest path (SP) problem is one of the studied issues of graph theory and one of great importance in many information systems and applications. For example, transportation problems where the goal is to determine the shortest path (the path with the minimal length, or with the minimal time or the cost of travel, etc.) between two given points can be described as a SP problem. It is solved by determining a path with minimal weight between two given vertices in the graph with a single weight function. There are well-known algorithms for finding the path with minimal weight, like the Dijkstra, Bellman–Ford, Floyd–Warshall and Johnson ones (Jungnickel, 1999). In many cases using a graph with a single weight function is insufficient because it does not describe precisely the problem considered. For example, we want to determine the path where the cost and the time of travel are considered and minimized simultaneously. Thus, a graph with $k > 1$ weight functions is used and the problem is called the multicriteria shortest path (MSP) one. A special case of the MSP problem is the BSP one, where $k = 2$ weight functions are considered.

The MSP and BSP problems are known to be NP-complete by transformation from a 0-1 knapsack problem (Garey and Johnson, 1990; Hansen, 1980; Skriver and Andersen, 2000a). Many algorithms for solving both problems are known, and they fall into the following categories: label correcting algorithms (Brumbaugh-Smith and Shier, 1989; Corley and Moon, 1985; Daellenbach and De Kluyver, 1980; Skriver and Andersen, 2000b), label setting algorithms (Hansen, 1980; Martins, 1984; Tung and Chew, 1988), k-th shortest path algorithms (Climaco and Martins, 1982), two-phase algorithms (Mote *et al.*, 1991) and others (Chen and Nie, 2013; Dell'Olmo *et al.*, 2005; Machuca *et al.*, 2009; Mandow and Pérez de la Cruz, 2008; Martí *et al.*, 2009; Raith and Ehrgott, 2009). The MSP problem is also solved using the weighed linear scalarization method, where a single-objective function is formulated and an optimal solution to a single-objective function is determined (Carraway *et al.*, 1990). All of the algorithms mentioned assume constant weights of the arcs of the graph, i.e., the value of the weight function does not change for the given arc.

For the first time the BBR problem was defined and described by Widuch (2012). The bus network is represented by a directed multigraph with two weight functions standing for the cost and the time of travel, respectively. For a given arc the weights take variable values because they are calculated during the process of finding the paths in the multigraph. The goal of the problem is to determine a path between two given vertices minimizing the time and the cost of travel simultaneously. Additionally, the time of starting travel at the start vertex is given. In the work of Widuch (2012) an analysis of the problem and a label correcting algorithm with deleting partial solutions were presented. In the algorithm, during the process of finding the solutions only a single partial solution is stored and it represents a path from the start vertex to the given vertex v_i. The new vertices are added to the current partial solution by visiting the vertices of the multigraph representing the bus network using the depth first search and the backtracking method. Each vertex v_i stores the list of pairs (t_i, c_i), which constitute a set of non-dominated solutions, where t_i and c_i are equal to the time and the cost of travel of the partial solutions which have already been analysed. These values are used for estimating the partial solution after adding a new vertex if it is possible to obtain a non-dominated final solution from it. If the estimation is negative, then the partial solution is not analysed and the backtrack is performed. Otherwise, the partial solution is analysed until the final vertex has been added to it.

BBR was modified by adding the next criterion, and in the work of Widuch (2013) the multiple-criteria bus routing (MBR) problem is described, where additionally the length of the path is taken into consideration. Thus, in the MBR problem we determine the path minimizing three criteria, i.e., the time and the cost of travel and the length of the path, simultaneously. There are important differences between the properties of the paths and the methods used to solve the MBR and BBR problems. The set of non-dominated solutions contains only loopless paths. The methods of estimating the partial solutions are different and we cannot use the same methods in both the problems.

In this paper a new algorithm for solving the BBR problem is presented. The work contains theoretical analysis of the BBR problem with reference to graph theory with particular emphasis on differences between that with constant weights and the problem with variable weights. The properties satisfied only for the latter problem are described. In particular, a possibility of obtaining a non-dominated final solution from a dominated partial solution is precisely analysed. It has been proved theoretically and experimentally confirmed. There are defined necessary terms for the final solution obtained from a dominated partial solution, which are not presented in the work of Widuch (2012). All relationships between the final solutions, with one of them obtained from a dominated partial solution, are defined. The proposed representation of a partial solution and more effective estimation of partial solutions used in the algorithm influence the number of computed and analysed partial solutions which are fewer in comparison with the method presented by Widuch (2012).

The bus routing problem has gained the attention of many researchers and have been intensively studied in the last few decades. In the work of Huang *et al.* (2014) the problem of optimal bus routing is studied. It is formulated as linking a series of bus stops in a certain order, aiming at minimizing the total cost, which includes user and supplier costs. Thus, a single-criterion problem is considered and ant colony optimization is used to determine an optimal solution.

In 1969 the school bus routing problem (SBRP) was formulated (Newton and Thomas, 1969). It is a problem in the management of school bus fleet and seeks to plan an efficient schedule for a fleet of school buses that pick up students from various bus stops and deliver them to the school by satisfying various constraints, such as the bus capacity, where all students are picked and each student must be assigned to a particular bus. The objective of bus route planing is to visiting all bus stops minimizing the number of school used buses and the total bus travel distance while satisfying service qualities such as student maximum riding time on a bus. The problem is widely studied and a review of papers on SPRP solutions is presented by Park and Kim (2010). The work on solving the problem has continued by the adaptation of various methods such as the branch-and-cut algorithm (Riera-Ledesma and Salazar-González, 2012), ant colony optimization (Addor *et al.*, 2013; Arias-Rojas *et al.*, 2012; Bronshtein and Vagapova, 2015; Yigit and Unsal, 2016), simulated annealing (Manumbu *et al.*, 2014), the genetic algorithm (Sghaier *et al.*, 2013), tabu search (Pacheco *et al.*, 2013), the GRASP (greedy randomized adaptative search procedure) metaheuristic (Siqueira *et al.*, 2016), the time saving heuristic (Worwa, 2014), the harmony search heuristic (Kim and Park, 2013), or the column-generation-based algorithm (Caceres *et al.*, 2014). In the work of Chen *et al.* (2015) two algorithms for solving the SBRP are proposed: an exact method of mixed integer programming (MIP) and hybrid simulated annealing with the local search metaheuristic.

The SBRP is modified and many variants of the problem have been studied. One variant of the SBRP is the school bus routing problem with time windows (SBRPTW). It takes into account that buses must arrive to pick up students before some specific time (lower bound of the time window), and they can arrive before another specific time (the upper bound of the time window). In addition, the students were not picked up before the beginning of the time window. A hybrid column generation method (López and Romero, 2015) and a branch-and-bound algorithm (Kim *et al.*, 2012) are proposed to solve the problem. The next studied variant of the SBRP is the school bus routing problem with bus stop selection (SBRPBSS). Here a set of potential stops is determined first in such a way that each student lives within a given distance of at least one stop. Routes are then determined for school buses so that all students are picked up at a stop they can reach. Thus, determining the set of visited bus stops is a part of the problem. The following methods of resolving the SBRPBSS are proposed: a genetic algorithm (Díaz-Parra *et al*., 2012; Kang *et al*., 2015), a column-generation-based algorithm (Kinable *et al*., 2014; Riera-Ledesma and Salazar-González, 2013), a GRASP + VND (variable neighborhood descent) matheuristic (Schittekat *et al*., 2013), an artificial ant colony with a variable neighborhood local search algorithm (Euchi and Mraihi, 2012), continuous approximation (Ellegood *et al*., 2015). In the work of Chalkia *et al*. (2014) the SBRPBSS is modified and the safety of the bus stop (the size and location of the waiting area, the quality of the ground in the waiting area, and the visibility of the stop for approaching drivers, pedestrian crossing, etc.) is in addition considered.

The paper consists of four sections and it is organized as follows. In Section 2, the BBR problem is described. It contains the formulation of the mathematical model, the analysis of the BBR problem and the algorithm for solving it. The influence of dominated partial solutions on non-dominated final solutions is precisely analysed and the conditions, whose fulfillment makes it possible to obtain a non-dominated final solution from a dominated partial solution are presented. In Section 3 experimental test results are presented. Finally, some conclusions are drawn in Section 4.

2. Bicriterion bus routing problem

2.1. Formulation of the problem. The BBR problem belongs to the group of problems where the goal is to choose the means of transport and to find a route of travel between two given points for a given time of starting travel. The bus network is represented by a directed weighted multigraph $G = (V, E)$. The multigraph G contains $|V| = n$ vertices $v_1, \ldots, v_n$ and $|E| = m$ arcs $e_1, \ldots, e_m$ ($e_i = (v_j, v_k)$; $v_j \neq v_k$; $v_j, v_k \in V$). The vertices represent the bus stops, thus a vertex expression with reference to the multigraph G representing the bus network determines the bus stop of the bus network. In the network buses of M bus lines numbered from 1 to M are run. The network is divided into zones and determines the cost of travel.

For each bus line i ($i = 1, \ldots, M$) the route is defined and consists of a sequence of stops through which the bus runs from a start stop, represented by vertex v_s^i, to a final stop, represented by vertex v_e^i, of the line. The travel of the bus of a given bus line is directed, i.e., if it runs from v_a^i to v_b^i ($v_a^i \neq v_b^i$), this does not imply that the bus runs in the opposite direction. The bus can run in both directions but the routes can be different. Bus stops belonging to the route of the bus line are different except for the start and final stops, which can be the same. If the start and final stops are identical, then we have called a circular bus line.

Let the route of the i-th bus line ($i = 1, \ldots, M$) be represented by a sequence of the following vertices:

$$\langle v_0^i, v_1^i, \ldots, v_{k-1}^i, v_k^i \rangle, \tag{1}$$

where $v_0^i = v_s^i$ represents the start stop and $v_k^i = v_e^i$ represents the final stop of the line. The bus runs between stops belonging to the route represented by (1) with a given frequency. It runs from v_0^i at time T_0^i, passes through $v_1^i, \ldots, v_{k-1}^i$ at times $T_0^i + \delta_0^i, \ldots, T_0^i + \delta_{k-2}^i$, respectively, and reaches v_k^i at time $T_0^i + \delta_{k-1}^i$. The bus starts the next course at time T_1^i ($T_1^i = T_0^i + \beta_0^i$), and therefore it reaches $v_1^i, \ldots, v_k^i$ at times $T_1^i + \delta_0^i, \ldots, T_1^i + \delta_{k-1}^i$. It executes p^i courses and leaves v_0^i at the following times: $T_0^i, \ldots, T_{p^i-1}^i$ ($T_0^i < \ldots < T_{p^i-1}^i$), where $T_j^i = T_0^i + \beta_{j-1}^i$ ($j = 1, \ldots, p^i - 1$). The timetable of the bus of the i-th line ($i = 1, \ldots, M$) defines the values T_0^i, $\beta_0^i, \ldots, \beta_{p^i-2}^i$, $\delta_0^i, \ldots, \delta_{k-1}^i$ ($0 < \beta_0^i < \ldots < \beta_{p^i-2}^i$; $0 < \delta_0^i < \ldots < \delta_{k-1}^i$).

The frequency of the bus courses depends on the time of day. For example, during peak hours it is greater than in the evening. Thus the timetable defines the parameters $\beta_0^i, \ldots, \beta_{p^i-2}^i$ for the given bus line. The time of day also influences the time of travel between two given bus stops, i.e., it may be greater during peak hours than in the evening. In addition, it depends on the way of travel between the pair of stops, i.e., if the travel is directed or the travel through other stops. Therefore the parameters $\delta_0^i, \ldots, \delta_{k-1}^i$ are defined for each bus line. We assume a simplified model of the bus network where the times of getting on and off the bus by passengers are omitted.

The BBR problem is stated as follows. Given the bus network structure, the bus line routes and the timetable, the start stop represented by the start vertex v_s and the final stop represented by the final vertex v_e between which we want to travel, and the time T_s of starting travel at v_s. The goal is to find a route from the start stop to the final stop minimizing the time and the cost of travel. The stops belonging to the route, the stops of changes, the times of departure from all stops belonging to the route, the bus lines along which the buses run between stops should be determined.

The time of travel is the sum of the time of waiting at the start stop, the times of waiting for changes and the travel times between stops belonging to the route. The travel times between stops are defined by the timetable of bus lines. The cost of travel depends on the location of the stops in the area of zones, the number of changes in the route and the type of bus line, i.e., whether it is a regular or a fast line. Travel by a fast line is faster than by a regular line, and the cost of travel is twice as large as the cost of travel by a regular line. The cost of a single travel, i.e., travel without a bus change, by a bus of a regular line is calculated as follows. A ticket for travel within the area of a single zone equals c_1 $(0 < c_1)$ units, within two zones it equals c_2 $(c_1 < c_2)$ units and within the confines of more than two zones it equals c_3 $(c_2 < c_3)$ units. Therefore the cost of travel from the start stop to the final stop equals the sum of costs of travel between the stops of bus changes. In the examples the following costs of a ticket are assumed: $c_1 = 2.0$, $c_2 = 2.3$ and $c_3 = 2.6$ units.

2.2. Mathematical model of the BBR problem. The mathematical model of the BBR problem is formulated as follows (Table 1 shows the symbols used in the model):

$$\min \quad T(p) = \sum_{l=1}^{M} \sum_{i \in V} \sum_{j \in V} t_{ijl} \cdot x_{ijl}, \tag{2}$$

$$\min \quad C(p) = \sum_{l=1}^{M} \sum_{i \in V} \sum_{j \in V} c_{ijl} \cdot x_{ijl}, \tag{3}$$

subject to

$$\sum_{l=1}^{M} \sum_{\{j|(i,j)\in E\}} x_{ijl} - \sum_{l=1}^{M} \sum_{\{j|(j,i)\in E\}} x_{jil} = \begin{cases} 1 & \text{if } i = v_s, \\ 0 & \text{if } i \neq v_s, v_e, \\ -1 & \text{if } i = v_e, \end{cases} \tag{4}$$

$$\sum_{l=1}^{M} \sum_{j \in V} x_{ijl} \leq 2, \quad \forall i \in V, \tag{5}$$

$$T_{p,a}^k = \begin{cases} T_s, & k = 0, \\ T_s + \sum_{q=0}^{k-1} t_{abl}, & a = v_p^q, b = v_p^{q+1}, \\ & l = l_p^{q,q+1}, \\ & 0 < k < \text{len}(p), \end{cases} \tag{6}$$

$$T_{p,a}^k \leq T_{p,d}^k \ \wedge \ T_{p,d}^k \in D[a;l],$$
$$k = 0, \ldots, \text{len}(p) - 2, \quad a = v_p^k, \quad l = l_p^{k,k+1}. \tag{7}$$

The objective functions (2) and (3) minimize the time and the cost of travel, respectively. The constraints (4) yield a directed path from the start vertex v_s to the final vertex v_e. Constraints (5) state that each vertex is visited at most two times. The constraints (6) state the time of arrival to each vertex belonging to the path. Finally, the constraints (7) force that the time of departure from a vertex is not earlier than the time of arrival to this vertex and the time of departure is in line with the timetable.

Table 1. Symbols used in the mathematical model of the BBR problem.

Parameters	
$V = \{1, \ldots, n\}$	set of vertices representing bus stops
E	set of all arcs between vertices
M	number of bus lines
$D[1 \ldots, n; 1, \ldots, M]$	timetable (times of departures of each bus line from each bus stop)
t_{ijl}	time of travel from v_i to v_j by bus of line l, it includes potential time of waiting at v_i
c_{ijl}	cost of travel from v_i to v_j by bus of line l
v_s, v_e	start and final vertices
T_s	time of starting travel at v_s
Additional symbols	
p	path from v_s to v_e
len(p)	number of vertices belonging to p
v_p^k	vertex in k-th position in path p
$l_p^{k,k+1}$	line number of bus which runs from vertex in k-th position to vertex in $(k+1)$-th position in p
Decision variables	
x_{ijl}	1 if bus of line l traverses arc from v_i to v_j, 0 otherwise
$T_{p,a}^k$	time of arrival to vertex in k-th position in path p
$T_{p,d}^k$	time of departure from vertex in k-th position in path p

2.3. Analysis of the BBR problem. The BBR problem can be modeled in graph theoretical terms as follows. A directed weighted multigraph $G = (V, E)$ represents the bus network. The vertices $v_1, \ldots, v_n$ represent the bus stops. Each arc $e_i = (v_j, v_k)$, $(v_j \neq v_k;\ v_j, v_k \in V)$ corresponds to a specific bus line whose buses run directly from the stop represented by v_j to the stop represented by v_k. Direct travel from v_j to v_k means that the route does not include other vertices. Between a pair of stops buses of many bus lines can run. Therefore the multigraph can contain parallel arcs. The arc e_i has a single label $l(e_i)$ and two weights: $t(e_i)$ and $c(e_i)$.

The label $l(e_i)$ takes a value from the range $1, \ldots, M$ and represents the line number of the bus which runs from v_j to v_k. During the process of finding the solutions it is used to determine if a change at the given bus stop is done, and to determine the cost of travel.

The weight $t(e_i)$ takes a positive value and it equals the difference between the time of arrival T_k to v_k and the time of arrival T_j to v_j, i.e., $t(e_i) = T_k - T_j$. Thus it is a sum of the time of waiting at v_j and the travel time from v_j to v_k. The travel time from v_j to v_k is constant

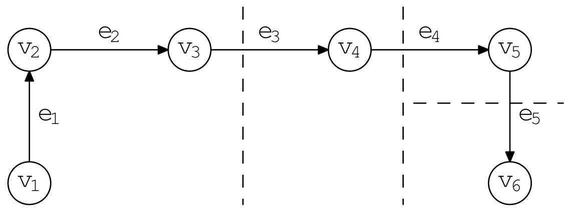

Fig. 1. Route of the bus line. The dashed line denotes the border of zones.

Table 2. Values of the weight c of arcs $e_1, \ldots, e_5$ depending on the start vertex v_s for travel to the final vertex v_6 by a bus of the line whose route is presented in Fig. 1.

v_s	$c(e_1)$	$c(e_2)$	$c(e_3)$	$c(e_4)$	$c(e_5)$
v_1	c_1	0	$c_2 - c_1$	$c_3 - c_2$	0
v_2	–	c_1	$c_2 - c_1$	$c_3 - c_2$	0
v_3	–	–	c_2	$c_3 - c_2$	0
v_4	–	–	–	c_2	$c_3 - c_2$
v_5	–	–	–	–	c_2

and defined by the timetable, but the time of waiting at v_j is variable and it depends on the time of arrival T_j to v_j. Therefore the value of $t(e_i)$ is variable and determined by the time of waiting at v_j.

The weight $c(e_i)$ takes a non-negative value and equals $c(e_i) = c_k - c_j$, where c_k is the cost of travel from v_s to v_k and c_j is the cost of travel from v_s to v_j. The weight $c(e_i)$ like the weight $t(e_i)$, is variable. The value of $c(e_i)$ depends on a possible change at v_j and location the vertices v_j and v_k in the same zone or of in different zones. We should consider the following cases to determine the value of $c(e_i)$. First, travel with a change at v_j is considered. If v_j and v_k are located in the same zone, then $c(e_i)$ equals c_1 and c_2 otherwise.[1] Second, we travel without a change at the vertex v_j. If v_j and v_k are located in the same zone, then this does not increase the cost of travel and $c(e_i) = 0$. If we cross a zone, then the value of $c(e_i)$ depends on the number of zones which we crossed since the last change while travelling to v_j. If we did not cross any zone, then $c(e_i) = c_2 - c_1$. It equals $c(e_i) = c_3 - c_2$ if we crossed a single zone, and if we crossed two or more zones it equals $c(e_i) = 0$.

The determination of the value of the weight c is illustrated with an example of travel to the vertex v_6 by a bus of the line whose route is presented in Fig. 1. The value of the weight c of arcs $e_1, \ldots, e_5$ depends on the start vertex v_s (Table 2). Let us consider the start vertex $v_s = v_1$. The vertices v_1 and v_2 are located in the same zone, therefore $c(e_1) = c_1$. We do not cross a zone while travelling from v_2 to v_3. Therefore it does not increase the cost of travel and the weight c of arc e_2 equals $c(e_2) = 0$. The vertices v_1, v_2 and v_3 are located in the same zone; and therefore the cost of travel from v_1 to v_2 equals the cost of travel from v_1 to v_3 and is equal to c_1. We crossed two zones while travelling from v_1 to v_5; therefore the next crossed zone while travelling from v_5 to v_6 does not increase the cost of travel and $c(e_5) = 0$. Table 2 shows that the weight $c(e_i)$ is not constant. For example, the weight $c(e_5)$ takes 3 different values.

The bus route from the start stop represented by the start vertex v_s to the final stop represented by the final vertex v_e is given by the path

$$p_{v_s,v_e} = \langle v_0 = v_s, e_1, \ldots, v_{k-1}, e_k, v_k = v_e \rangle \quad (8)$$

from v_s to v_e in the multigraph G representing the bus network. For each vertex v_i $(i = 0, \ldots, k-1)$ belonging to the path p_{v_s,v_e}, the time of departure T_i is stored. Thus a path expression with reference to the multigraph G representing the bus network determines the bus route in the network.

Definition 1. A partial solution is called the *path* p_{v_s,v_i} in the multigraph G from the start vertex v_s to any vertex v_i, where $v_i \neq v_s$ and $v_i \neq v_e$. The path p_{v_s,v_e} from the start vertex v_s to the final vertex v_e is called the *final solution*.

Definition 2. A path p_{v_i,v_j} containing a subsequence of vertices and arcs from v_i to v_j belonging to p_{v_s,v_e} is called a *subpath* of p_{v_s,v_e} and it is denoted as follows:

$$p_{v_i,v_j} = \mathrm{sub}_{p_{v_s,v_e}}(v_i, v_j).$$

Definition 3. Assume that paths p_{v_s,v_i} and p_{v_i,v_e} containing a sequence of vertices and arcs described by

$$\begin{aligned} p_{v_s,v_i} &= \langle v'_0, e'_1, \ldots, e'_j, v_i \rangle, \\ p_{v_i,v_e} &= \langle v''_0, e''_1, \ldots, e''_k, v_e \rangle, \end{aligned}$$

are given, where $v'_0 = v_s$ and $v''_0 = v_i$. The start vertex of p_{v_i,v_e} and the final vertex of p_{v_s,v_i} are identical, and it follows that it is possible to obtain the path p_{v_s,v_e} as a concatenation of p_{v_s,v_i} and p_{v_i,v_e}:

$$p_{v_s,v_e} = p_{v_s,v_i} \oplus p_{v_i,v_e}. \quad (9)$$

The path p_{v_s,v_e} consists of a sequence of vertices and arcs belonging to the paths p_{v_s,v_i} and p_{v_i,v_e}:

$$p_{v_s,v_e} = \langle v_s, e'_1, \ldots, e'_j, v_i, e''_1, \ldots, v''_{k-1}, e''_k, v_e \rangle.$$

The length of the path equals the number of arcs belonging to the path. The path (8) has two weights $T(p_{v_s,v_e})$ and $C(p_{v_s,v_e})$ that represent the time and the cost of travel from v_s to v_e. These weights are equal to the sum of the corresponding weights of the arcs belonging to

[1]Travel by a bus of a regular line is assumed, otherwise the value of $c(e_i)$ must be multiplied by 2.

p_{v_s,v_e}, i.e.,

$$T(p_{v_s,v_e}) = \sum_{i=1}^{k} t(e_i), \tag{10}$$

$$C(p_{v_s,v_e}) = \sum_{i=1}^{k} c(e_i). \tag{11}$$

Additionally, the number of crossed zones in p_{v_s,v_e} is denoted by $Z(p_{v_s,v_e})$.

The time of travel of the path p_{v_s,v_e} (9) obtained as a concatenation of p_{v_s,v_i} and p_{v_i,v_e} equals the sum of the times of travel:

$$T(p_{v_s,v_e}) = T(p_{v_s,v_i}) + T(p_{v_i,v_e}),$$

where the time of starting travel at v_i equals $T_i = T_s + T(p_{v_s,v_i})$.

The cost of travel $C(p_{v_s,v_e})$ of the path p_{v_s,v_e} (9) depends on a possible change at the vertex v_i. If the travel through v_i is done with a change (Fig. 2(a)) then it is necessary to validate a new ticket and the cost of travel $C(p_{v_s,v_e})$ equals the sum of the costs of travel:

$$C(p_{v_s,v_e}) = C(p_{v_s,v_i}) + C(p_{v_i,v_e}).$$

Otherwise (Fig. 2(b)) it is not necessary to validate a new ticket, the sum of costs of travel should be decreased by Δc and equals

$$C(p_{v_s,v_e}) = C(p_{v_s,v_i}) + C(p_{v_i,v_e}) - \Delta c.$$

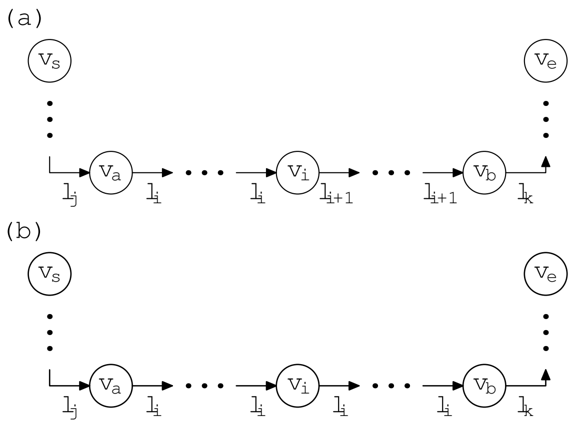

Fig. 2. Paths from the start vertex v_s to the final vertex v_e: with a change at the vertex v_i (a), without a change at the vertex v_i (b). The label on each arc represents the line number of the bus which runs between bus stops represented by the vertices connected by the arc.

The value of decreasing the cost Δc depends on the vertex v_i and the number of crossed zone borders Z_a since the last change at v_a while travelling to v_i in p_{v_s,v_i} and the number of crossed zone borders Z_b to the next change at v_b while travelling from v_i in p_{v_i,v_e} (Fig. 2(a)). It equals

$$\Delta c = C(\mathrm{sub}_{p_{v_s,v_e}}(v_a, v_i)) + C(\mathrm{sub}_{p_{v_s,v_e}}(v_i, v_b)) - C(\mathrm{sub}_{p_{v_s,v_e}}(v_a, v_b)),$$

where $C(\mathrm{sub}_{p_{v_s,v_e}}(v_a, v_i))$ and $C(\mathrm{sub}_{p_{v_s,v_e}}(v_i, v_b))$ are respectively the cost of travel from v_a to v_i and the cost of travel from v_i to v_b, and $C(\mathrm{sub}_{p_{v_s,v_e}}(v_a, v_b))$ equals the cost of travel from v_a to v_b. The values of Δc are shown in Table 3.

Table 3. Values of decreasing the cost Δc depending on the number of crossed zone borders Z_a and Z_b being a result of travel through the vertex v_i without a change.

	$Z_b = 0$	$Z_b = 1$	$Z_b \geq 2$
$Z_a = 0$	c_1	c_1	c_1
$Z_a = 1$	c_1	$2 \times c_2 - c_3$	c_2
$Z_a \geq 2$	c_1	c_2	c_3

The objective of the BBR problem is to find, in the multigraph G representing the bus network, the path p_{v_s,v_e} minimizing (10) and (11) simultaneously.

The BBR problem is an example of a multiple-criteria optimization (MO) problem, where k ($k > 1$) minimized or maximized criteria f_i ($i = 1, \ldots, k$) are given. In most cases, there does not exist a single solution for which all the critera take optimum values, because in order to improve the value of any of the functions we need to degrade those of other functions. Therefore the solution of the MO is a set of solutions called the set of non-dominated (Pareto optimal) solutions (Ehrgott, 2000; Pareto, 1896).

Definition 4. Assume that there are k ($k > 1$) minimized criteria f_i ($i = 1, \ldots, k$) and two solutions A and B. The solution A is said to *dominate* the solution B, which is denoted as $A \succ B$, if the following conditions are satisfied:

$$\forall i \in \{1, \ldots, k\} : f_i(A) \leq f_i(B),$$
$$\exists j \in \{1, \ldots, k\} : f_j(A) < f_j(B).$$

Solving the BBR problem consists in solving the bicriterion shortest path (BSP) problem between v_s and v_e vertices in a multigraph with variable weights. The solution of the BBR problem consists of a set of paths in the multigraph G, representing the bus network, forming the set of non-dominated solutions. The weights defined by (10) and (11) are the criteria to be minimized.

The set of non-dominated solutions can contain many paths with the same values of the weights (10) and (11) (Widuch, 2012). According to Definition 4 these paths are non-dominated solutions.

Let the paths p'_{v_s,v_e} and p''_{v_s,v_e} belong to the set of non-dominated solutions and

$$T(p'_{v_s,v_e}) = T(p''_{v_s,v_e}), \quad (12)$$
$$C(p'_{v_s,v_e}) = C(p''_{v_s,v_e}). \quad (13)$$

One of the following properties is satisfied:

1. The paths p'_{v_s,v_e} and p''_{v_s,v_e} differ from each other in vertices or arcs belonging to these paths.

2. The paths p'_{v_s,v_e} and p''_{v_s,v_e} are identical, i.e., they contain the same sequence of vertices and arcs, and differ from each other in the times of departure from all vertices belonging to these paths.

A necessary condition for a non-dominated solution with the second property is determined by Lemma 1.

Lemma 1. (Widuch, 2013) *Consider paths p'_{v_s,v_e} and p''_{v_s,v_e} which consist of the same sequence of vertices and arcs but differ from each other in the times of departure from vertices belonging to these paths. Let (12) and (13) be satisfied. Then both the paths belong to the set of non-dominated solutions if we change at least once in these paths.*

The multigraph G representing the bus network contains many paths with the same sequence of vertices and arcs that differ from each other in times of departure from vertices. The property of a path belonging to the set of non-dominated solutions is described by Lemma 2.

Lemma 2. *Let P_T be the set of all paths from v_s to v_e containing the same sequence of vertices and arcs that differ from each other in the times of departure from vertices. For the path $p_{v_s,v_e} \in P_T$, let Δt equal the total time of waiting for changes in p_{v_s,v_e}. If p_{v_s,v_e} belongs to the set of non-dominated solutions, then*

$$\forall p'_{v_s,v_e} \in P_T : \Delta t \leq \Delta t', \quad (14)$$

where $\Delta t'$ equals the total time of waiting for changes in p'_{v_s,v_e}.

Proof. The paths belonging to the set P_T contain the same sequence of vertices and arcs, and it follows that

$$\forall p'_{v_s,v_e} : C(p_{v_s,v_e}) = C(p'_{v_s,v_e}). \quad (15)$$

The time of travel $T(p_{v_s,v_e})$ equals the sum of the travel times between vertices and the total time of waiting for changes. The total travel times between vertices in these paths are identical, and the time of travel $T(p_{v_s,v_e})$ depends on the total time of waiting for changes. If p_{v_s,v_e} is a non-dominated solution, then it follows that (14) and

$$\forall p'_{v_s,v_e} \in P_T : T(p_{v_s,v_e}) \leq T(p'_{v_s,v_e})$$

are satisfied, otherwise it is a dominated solution. ■

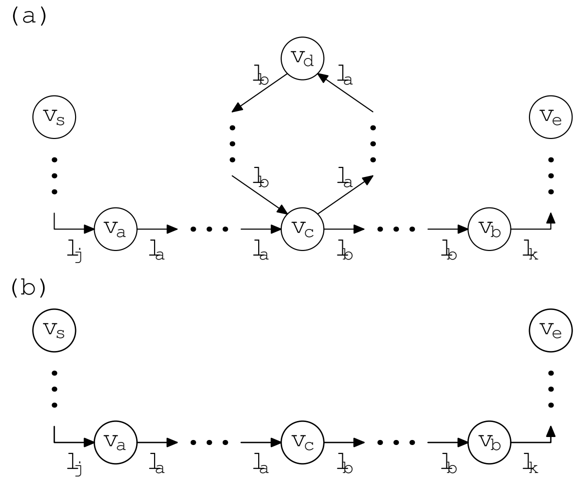

Fig. 3. Part of paths from the start vertex v_s to the final vertex v_e: a non-loopless path (a), a loopless path (b).

The set of non-dominated solutions can contain paths which are not loopless (Widuch, 2012). This property is satisfied only in a multigraph with variable weights of arcs, like the multigraph G representing the bus network. It has been shown that the set of non-dominated solutions contains only loopless paths if weights of arcs are non-negative and constant, and at least one is positive (Henig, 1985; Tung and Chew, 1988; 1992).

It should be pointed out that for each non-loopless path p_{v_s,v_e} there exists a loopless path p'_{v_s,v_e} which contains the same sequence of vertices and arcs like p_{v_s,v_e} but is devoid of the cycle. The properties of the non-loopless path p_{v_s,v_e} belonging to the set of non-dominated solutions are defined by Theorem 1.

Theorem 1. *Consider a non-loopless path p_{v_s,v_e} containing the cycle p_{v_c,v_c} (Fig. 3(a)) and a loopless path p'_{v_s,v_e} which contains the same sequence of vertices and arcs like p_{v_s,v_e} but is devoid of the cycle (Fig. 3(b)). The path p_{v_s,v_e} belongs to the set of non-dominated solutions if the following conditions are satisfied:*

1. *The cycle p_{v_c,v_c} contains only a single change.*

2. *At the vertex v_c no change is made.*

3. *The sum of the time of travel through the cycle p_{v_c,v_c} and the total time of waiting for changes in vertices of the non-loopless path p_{v_s,v_e} equals the total time of waiting for changes in vertices of the loopless path p'_{v_s,v_e}.*

Proof. The weights c and t of arcs do not take negative values, and for that reason $C(p'_{v_s,v_e}) \leq C(p_{v_s,v_e})$ and $T(p'_{v_s,v_e}) \leq T(p_{v_s,v_e})$. The loopless path p'_{v_s,v_e} does not dominate p_{v_s,v_e} when the following conditions are satisfied:

$$C(p_{v_s,v_e}) = C(p'_{v_s,v_e}), \quad (16)$$
$$T(p_{v_s,v_e}) = T(p'_{v_s,v_e}). \quad (17)$$

Thus, it is necessary to prove the fulfillment of (16) and (17) and define conditions that guarantee this.

Consider two vertices of changes v_a and v_b, where v_a is the vertex of the last change while travelling to v_c and v_b is the vertex of the next change while travelling from v_c (Fig. 3). For the paths p_{v_s,v_e} and p'_{v_s,v_e} the conditions

$$\mathrm{sub}_{p_{v_s,v_e}}(v_s, v_a) = \mathrm{sub}_{p'_{v_s,v_e}}(v_s, v_a),$$
$$\mathrm{sub}_{p_{v_s,v_e}}(v_b, v_e) = \mathrm{sub}_{p'_{v_s,v_e}}(v_b, v_e)$$

are satisfied, and it follows that the costs of travel from v_s to v_a and from v_b to v_e are equal in both paths, i.e,

$$C(\mathrm{sub}_{p_{v_s,v_e}}(v_s, v_a)) = C(\mathrm{sub}_{p'_{v_s,v_e}}(v_s, v_a)),$$
$$C(\mathrm{sub}_{p_{v_s,v_e}}(v_b, v_e)) = C(\mathrm{sub}_{p'_{v_s,v_e}}(v_b, v_e)).$$

For this reason the cost of travel $C(p_{v_s,v_e})$ of the non-loopless path p_{v_s,v_e} depends on the cost of travel from v_a to v_b, and (16) holds if

$$C(\mathrm{sub}_{p_{v_s,v_e}}(v_a, v_b)) = C(\mathrm{sub}_{p'_{v_s,v_e}}(v_a, v_b)). \quad (18)$$

The subpath $\mathrm{sub}_{p'_{v_s,v_e}}(v_a, v_b)$ of the loopless path contains a single change at the vertex v_c (Fig. 3(b)). The condition (18) is satisfied if the subpath $\mathrm{sub}_{p_{v_s,v_e}}(v_a, v_b)$ containing the cycle p_{v_c,v_c} also contains a single change at v_d and we do not change at v_c (Fig. 3(a)). With each next change it is necessary to validate a new ticket. This increases the cost of travel and $C(p'_{v_s,v_e}) < C(p_{v_s,v_e})$, and the non-loopless path p_{v_s,v_e} is dominated. This proves Properties 1 and 2 in the Theorem. In the next part of the proof, it is necessary to define the conditions of the fulfillment of (18).

The condition (18) holds if the costs of travel from v_a to the vertices v_d and v_c, where we change, and the costs of travel from these vertices to the vertex v_b are equal, i.e.,

$$C(\mathrm{sub}_{p_{v_s,v_e}}(v_a, v_d)) = C(\mathrm{sub}_{p'_{v_s,v_e}}(v_a, v_c)), \quad (19)$$
$$C(\mathrm{sub}_{p_{v_s,v_e}}(v_d, v_b)) = C(\mathrm{sub}_{p'_{v_s,v_e}}(v_c, v_b)). \quad (20)$$

The fulfillment of (19) depends on $Z(\mathrm{sub}_{p_{v_s,v_e}}(v_a, v_c))$ and $Z(\mathrm{sub}_{p_{v_s,v_e}}(v_c, v_d))$, where $Z(\mathrm{sub}_{p_{v_s,v_e}}(v_a, v_c))$ equals the number of crossed zones in the subpath from v_a to v_c and $Z(\mathrm{sub}_{p_{v_s,v_e}}(v_c, v_d))$ equals the number of crossed zones in the subpath from v_c to v_d. If $Z(\mathrm{sub}_{p_{v_s,v_e}}(v_a, v_c)) \geq 2$, then the next crossed zone does not increase the cost of travel. Thus, (19) is satisfied because the cost of travel $C(\mathrm{sub}_{p_{v_s,v_e}}(v_a, v_d))$ does not depend on $Z(\mathrm{sub}_{p_{v_s,v_e}}(v_c, v_d))$. Otherwise, i.e., $Z(\mathrm{sub}_{p_{v_s,v_e}}(v_a, v_c)) < 2$, the next crossed zone increase the cost of travel and (19) is satisfied if $Z(\mathrm{sub}_{p_{v_s,v_e}}(v_c, v_d)) = 0$.

Table 4. Timetable of the path $p_{1,7}$ from $v_s = 1$ to $v_e = 7$ containing a cycle.

Vertex/ zone	Arrival time	Departure time	Bus line	Cost of travel
1 / 1	12:00	12:05	1	0.0
2 / 1	12:08	12:08	1	2.0
3 / 1	12:11	12:11	1	2.0
4 / 1	12:15	12:25	2	2.0
5 / 1	12:28	12:28	2	4.0
2 / 1	12:32	12:32	2	4.0
6 / 1	12:35	12:45	3	4.0
7 / 1	12:50		3	6.0

The condition (20) is satisfied in similar cases. If $Z(\mathrm{sub}_{p_{v_s,v_e}}(v_c, v_b)) \geq 2$ then $Z(\mathrm{sub}_{p_{v_s,v_e}}(v_d, v_c))$ does not influence the cost of travel and (20) is satisfied. Otherwise, we have $Z(\mathrm{sub}_{p_{v_s,v_e}}(v_c, v_b)) < 2$, the condition (20) is satisfied if $Z(\mathrm{sub}_{p_{v_s,v_e}}(v_c, v_d)) = 0$.

In the second part of the proof, (17) will be demonstrated. The condition

$$\mathrm{sub}_{p_{v_s,v_e}}(v_s, v_c) = \mathrm{sub}_{p'_{v_s,v_e}}(v_s, v_c)$$

is satisfied. Then it follows that

$$T(\mathrm{sub}_{p_{v_s,v_e}}(v_s, v_c)) = T(\mathrm{sub}_{p'_{v_s,v_e}}(v_s, v_c))$$

is satisfied, too, and the total times of waiting for changes in vertices of the subpaths $\mathrm{sub}_{p_{v_s,v_e}}(v_s, v_{c-1})$ and $\mathrm{sub}_{p'_{v_s,v_e}}(v_s, v_{c-1})$ are equal, where v_{c-1} is the vertex which precedes v_c in the paths. If the time of travel through the cycle p_{v_c,v_c} equals the time of waiting for a change at v_c in p'_{v_s,v_e}, then the times of departure from v_c in p_{v_s,v_e} and p'_{v_s,v_e} are identical. It follows that

$$T(\mathrm{sub}_{p_{v_s,v_e}}(v_c, v_e)) = T(\mathrm{sub}_{p'_{v_s,v_e}}(v_c, v_e))$$

occurs and (17) is satisfied, and this proves Property 3 defined in the theorem.

When the time of travel through the cycle p_{v_c,v_c} is longer than the time of waiting for a change at v_c in p'_{v_s,v_e}, then (17) is satisfied if

$$T(p_{v_c,v_c}) + \Delta t_{v_c,v_e} = \Delta t'_{v_c,v_e},$$

where $\Delta t_{v_c,v_e}$ and $\Delta t'_{v_c,v_e}$ equal the total time of waiting for changes in the subpaths $\mathrm{sub}_{p_{v_s,v_e}}(v_c, v_e)$ and $\mathrm{sub}_{p'_{v_s,v_e}}(v_c, v_e)$, respectively. In this case, Property 3 defined in the Theorem is satisfied, too. ■

We illustrate Theorem 1 with an example of determining paths from $v_s = 1$ to $v_e = 7$. The path $p_{1,7}$ containing the cycle $2 \to 3 \to 4 \to 5 \to 2$ is presented in Table 4.[2] Table 5 shows the path $p'_{1,7}$, which has the

[2]The column "Cost of travel" contains the cost of travel from the start vertex $v_s = 1$ to the given vertex, and the column "Bus line" contains the bus line by which we leave the given vertex.

Table 5. Timetable of the path $p'_{1,7}$ from $v_s = 1$ to $v_e = 7$ without a cycle.

Vertex/ zone	Arrival time	Departure time	Bus line	Cost of travel
1 / 1	12:00	12:05	1	0.0
2 / 1	12:08	12:15	2	2.0
6 / 1	12:18	12:45	3	4.0
7 / 1	12:50		3	6.0

same sequence of vertices and arcs as $p_{1,7}$ but is devoid a cycle. All vertices belonging to the cycle are located in the same zone, and thus we do not cross a zone border while running through the cycle and the travel through the cycle does not increase the cost of travel. Therefore the costs of travel of the paths $p_{1,7}$ and $p'_{1,7}$ are equal and their value is 6.0 units. In the path $p'_{1,7}$ we change at the vertices 2 and 6, and the times of waiting for a change are equal to 7 and 27 minutes, respectively. The time of departure from the vertex 2 towards the vertex 6 in the path $p_{1,7}$ is later than in the path $p'_{1,7}$; thus the time of waiting for change at the vertex 6 in the path $p_{1,7}$ is shorter and it equals 10 minutes. The time of making the cycle in the path $p_{1,7}$ equals 24 minutes and is longer than the time of waiting for change at vertex 2 in the path $p'_{1,7}$. The sum of the time of making the cycle and the time of waiting for change at the vertex 6 in the path $p_{1,7}$ is 34 minutes. It equals the sum of times of waiting for a change at the vertices 2 and 6 in the path $p'_{1,7}$. Therefore the times of travel of the paths $p_{1,7}$ and $p'_{1,7}$ are equal and their value is 50 minutes. The path $p_{1,7}$ satisfies the conditions defined by Theorem 1. The cycle contains only a single change at the vertex 4, and the vertex 2 is passed without a change.

Let us consider the paths $p^t_{v_s,v_e}$ and $p^c_{v_s,v_e}$ with the minimal time and the minimal cost of travel from v_s to v_e, respectively, and $c_{\max} = C(p^t_{v_s,v_e})$ and $t_{\max} = T(p^c_{v_s,v_e})$. The values $t_{\max}$ and $c_{\max}$ determine the maximal time and the maximal cost of travel the path belonging to the set of non-dominated solutions. According to Definition 4, for the path p_{v_s,v_e}, if $C(p_{v_s,v_e}) > c_{\max}$ is satisfied, then $p^t_{v_s,v_e} \succ p_{v_s,v_e}$. Similarly, $p^c_{v_s,v_e} \succ p_{v_s,v_e}$ if $T(p_{v_s,v_e}) > t_{\max}$ occurs. The value of $t_{\max}$ makes it possible to determine the latest time of arrival $T^e_{\max}$ to the final vertex v_e in the path being a non-dominated solution, i.e.,

$$T^e_{\max} = T_s + t_{\max}. \tag{21}$$

2.4. Influence of dominated partial solutions on non-dominated final solutions.

The partial solution p_{v_s,v_i} can be extended to the final solution p_{v_s,v_e} by determining the path from v_i to v_e. There are several problems connected with this operation. Many partial solutions are determined for the given vertex v_i during the process of finding the solutions and these partial solutions can be compared to each other according to the time and the cost of travel. If the partial solution is dominated by another partial solution, it is necessary to decide whether it should be stored and analysed or if it may be omitted. In consequence, it is important to know whether it is possible to extend a dominated partial solution and obtain from it a non-dominated final solution. An answer to this question contains conditions required to obtain a non-dominated final solution from a dominated partial solution which are presented in this subsection. They take into account, *inter alia*, on whether the vertex v_i is passed with a change or without it. Therefore all possible cases are analysed. Next, it is necessary to define the conditions under which a dominated partial solution may be omitted because it is not possible to obtain a non-dominated final solution from it. The estimation is done based on the partial solutions already computed for the vertex v_i. This subsection resolves all of the mentioned problems and it contains all listed conditions.

Assume that there are two final solutions p_{v_s,v_e} and p'_{v_s,v_e} obtained from the partial solutions p_{v_s,v_i} and p'_{v_s,v_i}, where $p_{v_s,v_i} \succ p'_{v_s,v_i}$. If the weights of arcs are constant then it shown that the monotonicity assumption holds, i.e., a final solution p'_{v_s,v_e} obtained from a dominated partial solution p'_{v_s,v_i} is a dominated solution ($p_{v_s,v_e} \succ p'_{v_s,v_e}$) and p_{v_s,v_e} belongs to the set of non-dominated solutions solely if, for each v_i belonging to p_{v_s,v_e}, the subpath $\mathrm{sub}_{p_{v_s,v_e}}(v_s, v_i)$ is a non-dominated solution, too (Azevedo and Martins, 1991; Carraway *et al*., 1990; Martins *et al*., 1999; Mote *et al*., 1991).

Lemma 3. *Consider a weighed multigraph G, where the weights take non-negative and variable values, and two partial solutions p_{v_s,v_i} and p'_{v_s,v_i}, where $p_{v_s,v_i} \succ p'_{v_s,v_i}$. Then the monotonicity assumption does not hold, and it is possible to obtain a non-dominated final solution p'_{v_s,v_e} from a dominated partial solution p'_{v_s,v_i}.*

Proof. According to Definition 4, if $p_{v_s,v_i} \succ p'_{v_s,v_i}$, then (22) or (23) holds:

$$T(p_{v_s,v_i}) < T(p'_{v_s,v_i}) \wedge C(p_{v_s,v_i}) \le C(p'_{v_s,v_i}), \tag{22}$$

$$T(p_{v_s,v_i}) \le T(p'_{v_s,v_i}) \wedge C(p_{v_s,v_i}) < C(p'_{v_s,v_i}). \tag{23}$$

Let δt and δc be respectively the differences between the times and the costs of travel of the partial solutions p'_{v_s,v_i} and p_{v_s,v_i}:

$$\delta t = T(p'_{v_s,v_i}) - T(p_{v_s,v_i}),$$
$$\delta c = C(p'_{v_s,v_i}) - C(p_{v_s,v_i}).$$

In order to prove the theorem, it is necessary to consider all possible cases of obtaining the final solution on the basis of a partial one. The time and the cost of travel of the final solutions p'_{v_s,v_e} and p_{v_s,v_e} obtained from p'_{v_s,v_i}

and p_{v_s,v_i} will be analysed. Let us denote by p'_{v_s,v_e} and p_{v_s,v_e} the concatenation of paths:

$$p_{v_s,v_e} = p_{v_s,v_i} \oplus p_{v_i,v_e}, \quad (24)$$
$$p'_{v_s,v_e} = p'_{v_s,v_i} \oplus p'_{v_i,v_e}. \quad (25)$$

From the conditions (22) or (23) it follows that $\delta t \geq 0$ and $\delta c \geq 0$. The relationship between the time of travel is defined by one of the following conditions:

T1: $T(p_{v_i,v_e}) = T(p'_{v_i,v_e}) + \delta t \Rightarrow T(p_{v_s,v_e}) = T(p'_{v_s,v_e})$,

T2: $T(p_{v_i,v_e}) > T(p'_{v_i,v_e}) + \delta t \Rightarrow T(p_{v_s,v_e}) > T(p'_{v_s,v_e})$,

T3: $T(p_{v_i,v_e}) < T(p'_{v_i,v_e}) + \delta t \Rightarrow T(p_{v_s,v_e}) < T(p'_{v_s,v_e})$.

If $\delta t = 0$, then the relationship between the time of travel depends only on the time of travel from v_i to v_e. A similar relationship occurs between the cost of travel, i.e.,

C1: $C(p_{v_i,v_e}) - \Delta c + \delta c = C(p'_{v_i,v_e}) - \Delta c' \Rightarrow C(p_{v_s,v_e}) = C(p'_{v_s,v_e})$.

C2: $C(p_{v_i,v_e}) - \Delta c + \delta c > C(p'_{v_i,v_e}) - \Delta c' \Rightarrow C(p_{v_s,v_e}) > C(p'_{v_s,v_e})$,

C3: $C(p_{v_i,v_e}) - \Delta c + \delta c < C(p'_{v_i,v_e}) - \Delta c' \Rightarrow C(p_{v_s,v_e}) < C(p'_{v_s,v_e})$,

where Δc and $\Delta c'$ are equal to the decrease in the cost of travel of the paths (24) and (25) when the vertex v_i is passed without a change. According to Definition 4, the following conditions are satisfied:

$$\delta c = 0 \quad \Rightarrow \quad \delta t > 0,$$
$$\delta t = 0 \quad \Rightarrow \quad \delta c > 0.$$

On the basis of T1–T3 and C1–C3 it is possible to determine nine relationships between the final solutions p_{v_s,v_e} and p'_{v_s,v_e}, denoted by R1–R9:

R1: C1 $\wedge$ T1 $\Rightarrow p_{v_s,v_e}$ and p'_{v_s,v_e} are non-dominated,

R2: C1 $\wedge$ T2 $\Rightarrow p'_{v_s,v_e} \succ p_{v_s,v_e}$,

R3: C1 $\wedge$ T3 $\Rightarrow p_{v_s,v_e} \succ p'_{v_s,v_e}$,

R4: C2 $\wedge$ T1 $\Rightarrow p'_{v_s,v_e} \succ p_{v_s,v_e}$,

R5: C2 $\wedge$ T2 $\Rightarrow p'_{v_s,v_e} \succ p_{v_s,v_e}$,

R6: C2 $\wedge$ T3 $\Rightarrow p_{v_s,v_e}$ and p'_{v_s,v_e} are non-dominated,

R7: C3 $\wedge$ T1 $\Rightarrow p_{v_s,v_e} \succ p'_{v_s,v_e}$,

R8: C3 $\wedge$ T2 $\Rightarrow p_{v_s,v_e}$ and p'_{v_s,v_e} are non-dominated,

R9: C3 $\wedge$ T3 $\Rightarrow p_{v_s,v_e} \succ p'_{v_s,v_e}$.

A possible change at the vertex v_i (Fig. 2) determines whether a given relationship of R1–R9 is actually satisfied or just only theoretically and never occurs. For that reason the following cases are considered:

1. passing v_i with a change in p_{v_s,v_e} and p'_{v_s,v_e},
2. passing v_i with a change only in p'_{v_s,v_e},
3. passing v_i with a change only in p_{v_s,v_e},
4. passing v_i without a change in p_{v_s,v_e} and p'_{v_s,v_e}.

In the first case, travel with a change at v_i in p_{v_s,v_e} and p'_{v_s,v_e} is considered. It follows that $\Delta c = \Delta c' = 0$, and the time and the cost of travel of p_{v_s,v_e} and p'_{v_s,v_e} only depend on p_{v_i,v_e} and p'_{v_i,v_e}, which are subpaths of p_{v_s,v_e} and p'_{v_s,v_e}, respectively:

$$p_{v_i,v_e} = \mathrm{sub}_{p_{v_s,v_e}}(v_i, v_e),$$
$$p'_{v_i,v_e} = \mathrm{sub}_{p'_{v_s,v_e}}(v_i, v_e).$$

If p_{v_i,v_e} and p'_{v_i,v_e} are identical, then

$$T(p_{v_i,v_e}) = T(p'_{v_i,v_e}),$$
$$C(p_{v_i,v_e}) = C(p'_{v_i,v_e}),$$

and according to (22) and (23), the following conditions are satisfied:

$$T(p_{v_s,v_e}) \leq T(p'_{v_s,v_e}), \quad (26)$$
$$C(p_{v_s,v_e}) \leq C(p'_{v_s,v_e}). \quad (27)$$

It follows that $p'_{v_s,v_i} \succ p_{v_s,v_i}$ never occurs and the relationships R2, R4 and R5 are not satisfied.

It should be pointed out that it is not possible to obtain p'_{v_s,v_e} for which (26) or (27) is not satisfied. This would mean that p_{v_i,v_e} and p'_{v_i,v_e} are different and one of the following conditions is satisfied:

$$T(p_{v_i,v_e}) > T(p'_{v_i,v_e}),$$
$$C(p_{v_i,v_e}) > C(p'_{v_i,v_e}).$$

In this case, p_{v_s,v_e} would be created as a concatenation of paths: $p_{v_s,v_e} = p_{v_s,v_i} \oplus p'_{v_i,v_e}$, then the subpaths $\mathrm{sub}_{p_{v_s,v_e}}(v_i, v_e)$ and $\mathrm{sub}_{p'_{v_s,v_e}}(v_i, v_e)$ are identical and the conditions (26) and (27) occur.

In the second case, a change at v_i in p'_{v_s,v_e} is but in p_{v_s,v_e} it is passed without a change. It follows that $\Delta c' = 0$ and $\Delta c \neq 0$. If the subpaths $\mathrm{sub}_{p_{v_s,v_e}}(v_i, v_e)$ and $\mathrm{sub}_{p'_{v_s,v_e}}(v_i, v_e)$ are identical, then

$$T(p_{v_s,v_e}) \leq T(p'_{v_s,v_e}),$$
$$C(p_{v_s,v_e}) < C(p'_{v_s,v_e}).$$

Therefore, $p_{v_s,v_i} \succ p'_{v_s,v_i}$ and only the relationship R7 or R9 can be satisfied. When the subpaths $\mathrm{sub}_{p_{v_s,v_e}}(v_i, v_e)$ and $\mathrm{sub}_{p'_{v_s,v_e}}(v_i, v_e)$ are different, then it is not possible to determine the relationship between the time and the cost of travel of the paths p_{v_s,v_e} and p'_{v_s,v_e}, thus any of the relationships R1–R9 can be fulfilled.

Table 6. Possibility of fulfillment of the relationships R1–R9 between the final solutions p_{v_s,v_e} and p'_{v_s,v_e} passing the vertex v_i with or without a change and the subpaths $\mathrm{sub}_{p_{v_s,v_e}}(v_i, v_e)$ and $\mathrm{sub}_{p'_{v_s,v_e}}(v_i, v_e)$ identical or different.

	Relationship between the final solutions p_{v_s,v_e} and p'_{v_s,v_e}								
	R1	R2	R3	R4	R5	R6	R7	R8	R9
vertex v_i is passed with change in p_{v_s,v_e} and p'_{v_s,v_e}, subpaths $\mathrm{sub}_{p_{v_s,v_e}}(v_i, v_e)$ and $\mathrm{sub}_{p'_{v_s,v_e}}(v_i, v_e)$ are identical	+	−	+	−	−	+	+	+	+
vertex v_i is passed with change in p'_{v_s,v_e} but in p_{v_s,v_e} it is passed without change, subpaths $\mathrm{sub}_{p_{v_s,v_e}}(v_i, v_e)$ and $\mathrm{sub}_{p'_{v_s,v_e}}(v_i, v_e)$ are identical	−	−	−	−	−	−	+	−	+
vertex v_i is passed without change in p'_{v_s,v_e} but in p_{v_s,v_e} it is passed with or without change or v_i is passed with change in p'_{v_s,v_e} but in p_{v_s,v_e} it is passed without change, subpaths $\mathrm{sub}_{p_{v_s,v_e}}(v_i, v_e)$ and $\mathrm{sub}_{p'_{v_s,v_e}}(v_i, v_e)$ are different	+	+	+	+	+	+	+	+	+
vertex v_i is passed without change in p'_{v_s,v_e} but in p_{v_s,v_e} it is passed with or without a change, subpaths $\mathrm{sub}_{p_{v_s,v_e}}(v_i, v_e)$ and $\mathrm{sub}_{p'_{v_s,v_e}}(v_i, v_e)$ are identical	+	−	+	+	−	+	+	−	+

In the next case, v_i is passed without a change in p'_{v_s,v_e} obtained from the dominated partial solution p'_{v_s,v_i} and a change is performed in p_{v_s,v_e}, thus $\Delta c' \neq 0$ and $\Delta c = 0$. When the subpaths $\mathrm{sub}_{p_{v_s,v_e}}(v_i, v_e)$ and $\mathrm{sub}_{p'_{v_s,v_e}}(v_i, v_e)$ are identical, then

$$T(p_{v_s,v_e}) \leq T(p'_{v_s,v_e}), \tag{28}$$

but the cost of travel depends on $\Delta c'$ and δc, and one of the following conditions is satisfied:

$$\Delta c' = \delta c \Rightarrow C(p_{v_s,v_e}) = C(p'_{v_s,v_e}), \tag{29}$$

$$\Delta c' > \delta c \Rightarrow C(p_{v_s,v_e}) > C(p'_{v_s,v_e}), \tag{30}$$

$$\Delta c' < \delta c \Rightarrow C(p_{v_s,v_e}) < C(p'_{v_s,v_e}). \tag{31}$$

According to (28)–(31), the relationships R2, R5 and R8 never occur. If the subpaths $\mathrm{sub}_{p_{v_s,v_e}}(v_i, v_e)$ and $\mathrm{sub}_{p'_{v_s,v_e}}(v_i, v_e)$ are different, then any of the relationships T1–T3 and C1–C3 can occur, thus any of the relationships R1–R9 can be fulfilled.

In the last case, travel without a change at v_i in p_{v_s,v_e} and p'_{v_s,v_e} is considered, thus $\Delta c \neq 0$ and $\Delta c' \neq 0$. First, the case when the subpaths $\mathrm{sub}_{p_{v_s,v_e}}(v_i, v_e)$ and $\mathrm{sub}_{p'_{v_s,v_e}}(v_i, v_e)$ are identical is considered. It should be pointed out that v_i in the partial solutions p_{v_s,v_i} and p'_{v_s,v_i} is reached by a bus of the same line. The conditions (22) or (23) is satisfied and it follows that

$$T(p_{v_s,v_e}) \leq T(p'_{v_s,v_e}).$$

The cost of travel depends on Δc, $\Delta c'$ and δc, therefore one of the following conditions can be fulfilled:

$$\Delta c' = \Delta c + \delta c \Rightarrow C(p_{v_s,v_e}) = C(p'_{v_s,v_e}),$$

$$\Delta c' > \Delta c + \delta c \Rightarrow C(p_{v_s,v_e}) > C(p'_{v_s,v_e}),$$

$$\Delta c' < \Delta c + \delta c \Rightarrow C(p_{v_s,v_e}) < C(p'_{v_s,v_e}).$$

For that reason the relationships R2, R5 and R8 never occur.

If the subpaths $\mathrm{sub}_{p_{v_s,v_e}}(v_i, v_e)$ and $\mathrm{sub}_{p'_{v_s,v_e}}(v_i, v_e)$ are different then any of the relationships T1–T3 and C1–C3 can be satisfied and any of the relationships R1–R9 can be fulfilled.

To complete the proof, the possibility of the fulfillment of the relationships R1–R9 between the final solutions p_{v_s,v_e} and p'_{v_s,v_e} in each case is presented in Table 6. If the given relationship never occurs, then it is denoted by the symbol "−", otherwise, i.e., it can be satisfied, it is denoted by the symbol "+". ■

According to Lemma 3, a final solution obtained from a dominated partial solution can be a non-dominated one. Thus, during the process of finding the solutions, it is necessary to analyse dominated partial solutions and they cannot be omitted. For a given partial solution p_{v_s,v_i} one can estimate whether the final solution obtained from it will be a dominated one. This can be estimated on the basis of the values $t_{\max}$, $c_{\max}$, $t^i_{\min}$, $c^i_{\min}$ and Δc, where $t^i_{\min}$ and $c^i_{\min}$ are equal respectively the minimal time and the minimal cost of travel from v_i to the final vertex v_e. The time of travel from v_i to v_e, depends on the time of arrival to v_i, therefore the value $t^i_{\min}$ equals the minimal time of travel from v_i to v_e which guarantees arrival to v_e no later than the time $T^e_{\max}$ (21). As mentioned in Section 2.3, if the time of arrival to v_e is later than $T^e_{\max}$, then the solution is dominated.

Theorem 2. (Widuch, 2012) *Let p_{v_s,v_i} be a partial solution representing a path from the start vertex v_s to the vertex v_i. It is possible to obtain a non-dominated final solution from p_{v_s,v_i} even when the following condition is satisfied:*

$$C(p_{v_s,v_i}) + c^i_{\min} - \Delta c > c_{\max}.$$

It can be estimated that the final solution obtained from the partial solution p_{v_s,v_i} will be a dominated one. The partial solution p_{v_s,v_i} can be omitted if it is not possible to obtain a non-dominated final solution from it. When it is estimated on the basis of the cost of travel, then the possibility of travel through the vertex v_i without a change and travel to any vertex which is a successor of v_i in the path of this line should be taken into account. Thus we must assume the value $\Delta c_{\max}$ being a maximal value of the decreasing of the cost of travel which takes into account the number of zones Z_a that we crossed since the last change while travelling to v_i (Table 3).

The partial solution p_{v_s,v_i} can be omitted when

$$C(p_{v_s,v_i}) + c^i_{\min} - \Delta c_{\max} > c_{\max} \tag{32}$$

occurs. If the vertex v_i is the final vertex of the line by whose bus we arrive at v_i in p_{v_s,v_i}, then we must change at v_i and in this case $\Delta c_{\max} = 0.0$ is assumed. The partial solution can be estimated on the basis of the time of travel and can be omitted if condition

$$T(p_{v_s,v_i}) + t^i_{\min} > t_{\max} \tag{33}$$

is satisfied.

The partial solutions already computed can be used to estimate, given the partial solution p_{v_s,v_i}, whether the final solution obtained from it will be a dominated one. We can stop analysing a dominated partial solution and it can be omitted when the conditions determined by Theorem 3 are satisfied. Otherwise we have to continue analysing it.

Theorem 3. *Consider partial solutions p_{v_s,v_i} and p'_{v_s,v_i}. The final solution obtained from p_{v_s,v_i} will be a dominated one if*

$$T(p_{v_s,v_i}) \geq T(p'_{v_s,v_i}), \tag{34}$$

$$C(p_{v_s,v_i}) - \Delta c_{\max} > C(p'_{v_s,v_i}). \tag{35}$$

Therefore p_{v_s,v_i} can be omitted.

Proof. Let us consider the final solutions p_{v_s,v_e} and p'_{v_s,v_e} respectively obtained from p_{v_s,v_i} and p'_{v_s,v_i},

$$p_{v_s,v_e} = p_{v_s,v_i} \oplus p_{v_i,v_e},$$
$$p'_{v_s,v_e} = p'_{v_s,v_i} \oplus p'_{v_i,v_e}.$$

From (34) it follows that the time of arrival to v_i in p_{v_s,v_i} is not earlier than the time of arrival to v_i in p'_{v_s,v_i}, and the time of departure from v_i can be earlier in p'_{v_s,v_i}. Therefore the subpaths p_{v_i,v_e} and p'_{v_i,v_e} from v_i to the final vertex v_e in the paths p_{v_s,v_e} and p'_{v_s,v_e} can be different. This case is not considered for the following reason. The subpaths p_{v_i,v_e} and p'_{v_i,v_e} can be identical and this case decides on negative estimation of the partial solution p_{v_s,v_i}.

If the subpaths p_{v_i,v_e} and p'_{v_i,v_e} are identical, then from (34) it follows that

$$T(p_{v_s,v_e}) \geq T(p'_{v_s,v_e}). \tag{36}$$

If (35) holds, then

$$C(p_{v_s,v_e}) > C(p'_{v_s,v_e}) \tag{37}$$

regardless of whether a change at v_i is performed it is passed without a change.

From (36) and (37) it follows that p_{v_s,v_e} is a dominated solution and it is not possible to obtain a non-dominated partial solution from p_{v_s,v_i} when (34) and (35) are satisfied. ■

Consider the vertex v_i and the list *LPS*$[v_i]$ containing k partial solutions $p^1_{v_s,v_i}, \ldots, p^k_{v_s,v_i}$ determined for the vertex v_i. Let $t^{pi}_{\min}$, $t^{pi}_{\max}$ and $c^{pi}_{\max}$ be respectively the minimal and the maximal time of travel and the maximal cost of travel from among all partial solutions in the list *LPS*$[v_i]$:

$$t^{pi}_{\min} = \min\{T(p^1_{v_s,v_i}), \ldots, T(p^k_{v_s,v_i})\},$$
$$t^{pi}_{\max} = \max\{T(p^1_{v_s,v_i}), \ldots, T(p^k_{v_s,v_i})\},$$
$$c^{pi}_{\max} = \max\{C(p^1_{v_s,v_i}), \ldots, C(p^k_{v_s,v_i})\}.$$

Let $c^{pi}_{\min}$ signify the minimal cost of travel decreased by the value of $\Delta c^i_{\max}$ $(i = 1, \ldots, k)$ from among all partial solutions, i.e.,

$$c^{pi}_{\min} = \max\{C(p^1_{v_s,v_i}) - \Delta c^1_{\max}, \ldots, C(p^k_{v_s,v_i}) - \Delta c^k_{\max}\}.$$

The estimation of the partial solution p_{v_s,v_i} can be done on the basis of the values of $t^{pi}_{\max}$ and $c^{pi}_{\max}$. According to Theorem 3, the partial solution p_{v_s,v_i} can be omitted when

$$T(p_{v_s,v_i}) \geq t^{pi}_{\max} \wedge C(p_{v_s,v_i}) - \Delta c_{\max} > c^{pi}_{\max} \tag{38}$$

is satisfied. The final solution obtained from it will be a dominated one. Additionally, the partial solutions stored in the list *LPS*$[v_i]$ can be estimated on the basis of p_{v_s,v_i} and the values of $c^{pi}_{\min}$ and $t^{pi}_{\min}$. If

$$T(p_{v_s,v_i}) \leq t^{pi}_{\min} \wedge C(p_{v_s,v_i}) < c^{pi}_{\min}, \tag{39}$$

then all partial solutions can be removed from the list *LPS*$[v_i]$. These partial solutions can be omitted because it is not possible to obtain a non-dominated final solution from any of them. The estimation is done only by checking the (38) and (39) conditions without comparing p_{v_s,v_i} with solutions stored in the list *LPS*$[v_i]$. Thus, it is done in $O(1)$ time and does not depend on the number of partial solutions in the list *LPS*$[v_i]$.

2.5. Algorithm for solving the BBR problem. The algorithm for finding all non-dominated paths from the start vertex v_s to the final vertex v_e and for the time of starting travel T_s at v_s is presented as Algorithm 1. It belongs to the group of label correcting algorithms with storing partial solutions and implements the methods of estimation of partial solutions presented in Section 2.4. During the process of finding the solutions, the following data structures are used:

- $\mathit{LPS}[v_i]$: the list of computed partial solutions for the vertex $v_i \in V$; for the final vertex v_e it contains the final solutions formed as the set of non-dominated solutions;
- *LFS*: the list of final solutions constitute the set of non-dominated solutions, where each solution represents the path from v_s to v_e for the time of starting travel equal to T_s;
- Q: the queue containing vertices for which the partial solutions have been computed.

The solutions stored in the list $\mathit{LPS}[v_i]$ are represented by a record $(t_{si}, c_{si}, v_k, l_i, T_k, \mathit{LPP}_k)$ containing the following data:

- t_{si}, c_{si}: the time and the cost of travel from v_s to v_i,
- v_k: the vertex which precedes v_i in the path,
- l_i: the bus line by which we arrive from v_k to v_i,
- T_k: the time of departure from v_k towards v_i,
- LPP_k: the list of pointers to the partial solutions for the vertex v_k.

In the initial part of the algorithm (lines 1–7) the values $t_{\max}, c_{\max}, t^i_{\min}$ and $c^i_{\min}$ are computed by the procedure FINDTC. The values $t_{\max}$ and $c_{\max}$ are computed by determining adequately the path of the minimal cost and that of the minimal time of travel from v_s to v_e. The path of the minimal time of travel is determined in the multigraph $G = (V, E)$ using the Dijkstra algorithm. It is not possible to find the path of the minimal cost of travel in the multigraph G using this algorithm or the other shortest path one (Widuch, 2012). Thus it is necessary to create the multigraph $G' = (V, E')$, $E \subset E'$, obtained from G, by adding additional arcs for each bus line. Let the path of the i-th bus line contains following sequence of vertices:

$$\langle v^i_0, v^i_1, \ldots, v^i_{k-1}, v^i_k \rangle.$$

For each pair of vertices v^i_a and v^i_b $(a = 0, \ldots, k-2,$ $b = a+2, \ldots, k)$ an arc (v^i_a, v^i_b) is added to the multigraph G'. The path of the minimal cost of travel is determined in the multigraph $G' = (V, E')$ using the Dijkstra algorithm.

Algorithm 1. Algorithm SOLVEBBR for finding all non-dominated paths from the start vertex v_s to the final vertex v_e and for the time of starting travel T_s.

Input: $v_s, v_e, T_s, G = (V, E)$
Output: *LFS*
1: $t_{\max}, c_{\max}, t^i_{\min}, c^i_{\min} \leftarrow$ FINDTC(v_s, v_e, T_s, G);
2: $G \leftarrow$ MODIFYG$(G, t^i_{\min}, t_{\max}, c^i_{\min}, c_{\max})$;
3: **for all** $v_i \in V$ **do**
4:    $\mathit{LPS}[v_i] \leftarrow \emptyset$;
5: **end for**
6: $S_s \leftarrow (0, 0, 0, 0, T_s, \emptyset)$; $\mathit{LPS}[v_s]$.ADDTOLIST(S_s);
7: $Q \leftarrow \emptyset$; Q.PUSH(v_s);
8: **while not** Q.EMPTY() **do**
9:    $v_k \leftarrow Q$.POP();
10:    **for all** $S_k \in \mathit{LPS}[v_k]$ has not yet been analysed **do**
11:       **for all** $(v_k, v_i, l_i) \in \mathit{out}(v_k)$; $v_i \neq v_s$ **do**
12:          **if not** INPATH(S_k, v_i) **then**
13:             $t_{si}, c_{si} \leftarrow$ the time and the cost of travel from v_s to v_i;
14:             $T_k \leftarrow$ the time of departure from v_k;
15:             $\Delta c_{max} \leftarrow$ the maximal decreasing of the cost of travel from v_s to v_i;
16:             $S_i \leftarrow (t_{si}, c_{si}, v_k, l_i, T_k, \&S_k)$;
17:             **if** estimation of S_i is positive **then**
18:                **if** $v_i = v_e$ **then**
19:                   $\mathit{LPS}[v_i] \Leftarrow$ ADDFSOL$(\mathit{LPS}[v_i], S_i)$;
20:                **else**
21:                   $\mathit{LPS}[v_i] \Leftarrow$ ADDPSOL$(\mathit{LPS}[v_i], S_i, \Delta c_{\max}, t_{\max}, c_{\max}, t^i_{\min}, c^i_{\min})$;
22:                   Q.PUSH(v_i);
23:                **end if**
24:             **end if**
25:          **end if**
26:       **end for**
27:    **end for**
28: **end while**
29: $\mathit{LFS} \Leftarrow$ CREATEFULLPATHS$(\mathit{LPS}[v_e], v_e)$;
30: $\mathit{LFS} \Leftarrow$ PATHSDIFFERTIMES(LFS, G);
31: $\mathit{LFS} \Leftarrow$ PATHSCONTAININGCYCLES(LFS, G);
32: **return** *LFS*;

The Dijkstra algorithm and the multigraphs G and G' are also used in the procedure FINDTC for determining the minimal times and the minimal costs of travel from each vertex $v_i \in V$ to v_e, and these values are stored in $t^i_{\min}$ and $c^i_{\min}$. The value $t^i_{\min}$ guarantees arrival to v_e no later than the time $T^e_{\max}$.

The values of $t_{\max}, c_{\max}, t^i_{\min}$ and $c^i_{\min}$ make it possible to determine a set of vertices that do not belong to a path being a non-dominated solution. If one of the following conditions is satisfied:

$$t^i_{\min}[v_i] \geq t_{\max}, \tag{40}$$

$$c^i_{\min}[v_i] - \Delta c^i_{\max} > c_{\max}, \tag{41}$$

then a path containing the vertex v_i does not belong to the set of non-dominated solutions. Thus v_i can be removed from the multigraph G. Therefore G is modified (line 2 of Algorithm 1) by the procedure MODIFYG, which removes from G all vertices v_i for which (40) or (41) is satisfied. Additionally, all arcs incoming into v_i and all arcs outgoing from v_i are removed from G.

In the next steps of the initial part of the algorithm, for each vertex $v_i \in V$ $(v_i \neq v_s)$ the list of partial solutions *LPS*$[v_i]$ is initialised as empty (lines 3–5). The list *LPS*$[v_s]$ of the start vertex v_s is initialised by an initial solution (line 6) from which the algorithm starts computation and v_s is inserted into the queue Q (line 7).

The paths from v_s to v_e belonging to the set of non-dominated solutions are computed in the **while**-loop (lines 8–28) by visiting vertices of the multigraph G using a modified breadth first search method (Jungnickel, 1999). There are computed all paths belonging to the set of non-dominated solutions except for those containing cycles, and paths differ from each other only in the times of departure from vertices. These paths are computed by the procedures PATHSCONTAININGCYCLES and PATHSDIFFERTIMES (lines 30–31) based on the paths computed in the **while**-loop.

In a single iteration of the **while**-loop the following operations are executed. The first vertex v_k from the queue Q is taken (line 9). In the **for**-loop (lines 10–27) we try to extend each unanalyzed partial solution S_k in the list *LPS*$[v_k]$ (each partial solution is analysed only once) and obtain new solutions for all adjacent vertices of the vertex v_k. For this purpose, in the **for**-loop all outgoing arcs (v_k, v_i, l_i) of the vertex v_k are analysed (lines 11–26). The arc (v_k, v_i, l_i) corresponds to the l_i bus line whose buses run directly from v_k to v_i. If v_i does not belong to the path represented by the partial solution S_k (it is checked by the procedure INPATH in line 12), then the time t_{si} and the cost c_{si} of travel from v_s to v_i as well as the time of departure T_k from v_k are computed (lines 13–14). Additionally, a maximal decrease in the cost of travel $\Delta c_{\max}$ is determined (line 15) and a new solution S_i is created (line 16). The solution S_i represents a path from v_s to v_i and the vertex v_k precedes v_i in the path.

The solution S_i is estimated if it is possible to obtain a non-dominated final solution from it. It is omitted if (32) or (33) is satisfied, which is checked in line 17. Otherwise, S_i is inserted into the list *LPS*$[v_i]$. If $v_i = v_e$, then S_i is inserted into *LPS*$[v_i]$ by the procedure ADDFSOL (line 19), otherwise the procedure ADDPSOL is used and v_i is inserted into the queue Q (lines 21–22).

In the last part of the algorithm, for all computed solutions in the list *LPS*$[v_e]$ full paths are determined by the procedure CREATEFULLPATHS (line 29), and these paths are stored in the list *LFS*. The full path is determined based on values stored in the record representing a solution in the list *LPS*$[v_e]$. The record contains the vertex v_k, which precedes the vertex v_e in the path, and contains the list *LPP*$_k$, which stores pointers to the partial solutions for the vertex v_k. Thus, the path is created from v_e by visiting the vertices which precede the current vertex until the start vertex v_s is reached. The full path is determined on the basis of the vertex v_k and the list of pointers *LPP*$_k$ stored in the record representing the solution.

Next, paths that differ from each other only in the times of departure from vertices belonging to the path are determined by the procedure PATHSDIFFERTIMES (line 30). According to Lemma 1, we change at least once in these paths. Thus in the procedure each path p_{v_s,v_e} stored in the list *LFS* is analysed but only the paths containing at least single vertex of change are taken into consideration. Let p_{v_s,v_e} contain k $(k > 0)$ vertices of changes: $v_0, \ldots, v_k$, where $v_0 = v_s$. Let us assume the times of departure from $v_0, \ldots, v_k$ are equal respectively to $T_0, \ldots, T_k$. If one can leave v_j $(j = 0, \ldots, k-1)$ at any time later than the time T_j and this does not increase the time of travel $T(p_{v_s,v_e})$, then a new path p'_{v_s,v_e} is created and added to the list *LFS*. The path p'_{v_s,v_e} differs from the path p_{v_s,v_e} only in times of departure from vertices belonging to these paths, and hence these paths differ in times of waiting for changes on vertices $v_0, \ldots, v_k$. The times of travel in both paths are equal, i.e., $T(p_{v_s,v_e}) = T(p'_{v_s,v_e})$, and it follows that the total times of waiting for changes in these paths are also equal.

In the last step, paths containing cycles are determined by the procedure PATHSCONTAININGCYCLES (line 31). A necessary condition for a non-dominated solution containing cycles is determined by Theorem 1, and therefore the paths containing at least a single vertex of change are analysed. Let us consider the path p_{v_s,v_e} (Fig. 3(b)), where the change is made at v_c and we arrive at it on a bus of the line l_a and leave it on a bus of the line l_b. The paths $p^a_{v_0^a,v_k^a}$ and $p^a_{v_0^b,v_k^b}$ of the buses of l_a and l_b contain the following sequence of vertices:

$$p^a_{v_0^a,v_j^a} = \langle v_0^a, \ldots, v_p^a = v_c, \ldots, v_j^a \rangle,$$
$$p^a_{v_0^b,v_k^b} = \langle v_0^b, \ldots, v_q^b = v_c, \ldots, v_k^b \rangle,$$

and these paths have a common vertex v_c. If

$$\exists x \in \{p+1, \ldots, j\} \wedge \exists y \in \{0, \ldots, q-1\} : v_x^a = v_y^b,$$

then the paths $p^a_{v_0^a,v_k^a}$ and $p^a_{v_0^b,v_k^b}$ have also a common vertex v_d, where $v_d = v_x^a = v_y^b$, and there exists the cycle p_{v_c,v_c} which starts and ends at v_c and contains v_d. If passing a cycle does not increase the time of travel and the conditions defined by Theorem 1 are satisfied, then a new path p'_{v_s,v_e} (Fig. 3(a)) containing the cycle p_{v_c,v_c} is created and inserted into the list *LFS*.

The procedure ADDPSOL is described by Algorithm 2. It tries to insert S_i into the list of partial solutions *LPS*. In the first part of the procedure

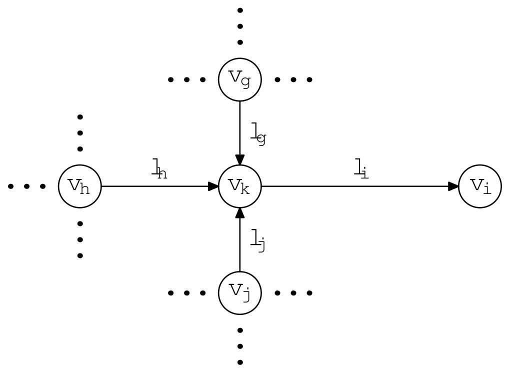

Fig. 4. Ways of arriving to the vertex v_k.

(lines 1–5) the solution S_i and solutions in the list *LPS* are estimated on the basis of values $t^{pi}_{\max}$, $c^{pi}_{\max}$, $t^{pi}_{\min}$ and $c^{pi}_{\min}$ without comparing S_i with solutions stored in *LPS*. First, in line 1 the condition (38) is checked, and if it is satisfied then the final solution obtained from S_i will be a dominated solution. Therefore S_i is omitted and the function returns an unmodified list *LPS* (line 2). The next, according to (39) condition it is estimated (line 3) if the final solutions obtained from partial solutions stored in *LPS* will be dominated. If the estimation is positive then all solutions are removed from *LPS* and S_i is inserted into *LPS* (line 4). In this case, the procedure returns the list *LPS* containing only single solution S_i (line 5).

In the second part of the procedure, in the **for**-loop the partial solution S_i is compared with each solution S stored in *LPS* (lines 7–14). According to Theorem 3 there are checked the conditions (34) and (35) (lines 9 and 11). If the condition in line 9 is satisfied, then it is not possible to obtain non-dominated final solutions from S_i; therefore S_i is omitted, the procedure ends its execution and returns *LPS* (line 10). If the condition in line 11 is met, then a final solution obtained from S stored in *LPS* will be a dominated solution, and therefore it is removed from *LPS* (line 12).

If the estimation of the partial solution S_i is positive, i.e., it is possible to obtain from it a non-dominated final solution, then it is inserted into the list *LPS* (lines 15–19). It should be pointed out that the vertex v_k, which precedes the vertex v_i in the path, can be reached from many vertices by buses of different lines. In Fig. 4 it is reached from v_g, v_h, v_j, and each case represents a different path from v_s to v_k. These paths are represented by solutions S_g, S_h, S_j stored in the list *LPS*$[v_k]$. The solutions S_g, S_h, S_j are extended by adding the vertex v_i to obtain new solutions S'_g, S'_h, S'_j for v_i. The new solutions represent different paths from v_s to v_i, where v_i is preceded by v_k in these paths and v_i is reached by a bus of the line l_i. If the times and the costs of travel are identical, i.e., $S'_g.t_{si} = S'_h.t_{si} = S'_j.t_{si}$ and $S'_g.c_{si} = S'_h.c_{si} = S'_j.c_{si}$, then only a single solution S' is created and stored in the

Algorithm 2. Procedure ADDPSOL of adding the partial solution S_i to the list *LPS*.

Input: *LPS*$, S_i, \Delta c_{\max}, t_{\max}, c_{\max}, t^i_{\min}, c^i_{\min}$
Output: *LPS*

```
1: if S_i.t_si ≥ LPS.t^pi_max and S_i.c_si − Δc_max > LPS.c^pi_max then
2:   return LPS;
3: else if S_i.t_si ≤ LPS.t^pi_min and S_i.c_si < LPS.c^pi_min then
4:   LPS.CLEAR(); LPS.ADDTOLIST(S_i);
5:   return LPS;
6: else
7:   for all S ∈ LPS do
8:     Δc'_max ← the maximal decrease in the cost of travel in the solution S;
9:     if S_i.t_si ≥ S.t_si and S_i.c_si − Δc_max > S.c_si then
10:      return LPS;
11:    else if S.t_si ≥ S_i.t_si and S.c_si − Δc'_max > S_i.c_si then
12:      LPS.REMOVE(S);
13:    end if
14:  end for
15:  if ∃S ∈ LPS: S.t_si = S_i.t_si and S.c_si = S_i.c_si and S.l_si = S_i.l_si then
16:    S.LPP_k.ADDTOLIST(S_i.LPP_k);
17:  else
18:    LPS.ADDTOLIST(S_i);
19:  end if
20:  return LPS;
21: end if
```

Algorithm 3. Procedure ADDFSOL of adding the final solution S_i to the list *LPS*.

Input: *LPS*$, S_i$
Output: *LPS*

```
if S_i.t_si > LPS.t^pi_max and S_i.c_si > LPS.c^pi_max then
  return LPS;
else if S_i.t_si < LPS.t^pi_min and S_i.c_si < LPS.c^pi_min then
  LPS.CLEAR(); LPS.ADDTOLIST(S_i);
  return LPS;
else
  for all S ∈ LPS do
    if S ≻ S_i then
      return LPS;
    else if S_i ≻ S then
      LPS.REMOVE(S);
    end if
  end for
  LPS.ADDTOLIST(S_i);
  return LPS;
end if
```

list *LPS*$[v_i]$. In the solution S' the list LPP_k stores pointers to the solutions S_g, S_h, S_j, which decreases the number of stored and analysed solutions as well as the time of computation. But in the procedure CREATEFULLPATHS (line 29 of Algorithm 1), from a single solution S' three full paths are determined. In these paths the vertex v_i is preceded by the vertices v_g, v_h, v_j, respectively.

In line 15 of the procedure ADDPSOL it is checked if the list *LPS* contains a solution S in which the vertex v_i is reached by a bus of the same line as in the partial solution S_i and the time and the cost of travel of S and S_i are equal. If S exists, then the pointer to the partial solution for the vertex v_k is added to the list LPP_k stored in S (line 16) and the new partial solution S_i is not inserted into the list *LPS*. Otherwise S_i is inserted into *LPS* (line 18).

The procedure ADDFSOL is described by Algorithm 3. It tries to insert the new solution S_i to the list *LPS* storing solutions for the final vertex v_e. The first part of the procedure (lines 1–5) is similar to lines 1–5 of the procedure ADDPSOL. The solution S_i and solutions in the list *LPS* are estimated on the basis of the values t^{pi}_{max}, $c^{pi}_{\max}$, $t^{pi}_{\min}$ and $c^{pi}_{\min}$ without comparing S_i with solutions stored in *LPS*. During the estimation the solution S_i or all solutions in the list *LPS* are omitted as dominated ones (lines 2 and 4).

In the second part of the procedure (lines 7–15) the solution S_i is compared with each solution S stored in the list *LPS*. If S_i dominates S (line 10), then S is removed from the list *LPS* (line 11), or if S dominates S_i (line 8), then S_i is omitted and the procedure ends its execution (line 9). If S_i is a non-dominated solution, then it is inserted into the list *LPS* (line 14).

The time and memory complexities of the algorithm SOLVEBBR depend on the number of paths belonging to the set of non-dominated solutions. The latter is a variant of the BSP problem. The BSP problem is known to be NP-complete and, as shown by Serafini (1987) or Skriver and Andersen (2000b), in the worst case, that of solutions of the problem grows exponentially with the number of vertices representing the bus stops. Therefore any algorithm that attempts to solve it is also exponential in terms of worst-case time and memory complexities. The worst case occurs when all possible paths between a given pair of vertices belong to the set of non-dominated solutions and the number of paths equals $\prod_{i=1}^{n} x_i \cdot d_i$, where d_i is equal to the outdegree of the vertex v_i; $x_i = 1$ if $v_i \neq v_e$ and $x_i = 0$ otherwise (we do not examine arcs outgoing from the final vertex v_e and the path does not contain these arcs). Therefore the pessimistic memory complexity equals $O(d^n)$, where $d = \max_{i=1,\dots,n}\{x_i \cdot d_i\}$.

Time complexity is clearly dominated by execution of the **while**-loop (lines 8–28 of the algorithm SOLVEBBR). Each path determined in the **while**-loop contains at most n vertices. For each of q_k solutions determined for the vertex v_k and stored in the list *LPS*$[v_k]$ (line 10), all of d_k outgoing arcs from v_k are examined (line 11). Thus, in the worst case it is necessary to examine $\prod_{i=1}^{n} x_i \cdot q_i \cdot d_i$ arcs, and pessimistic time complexity equals $O(r^n)$, where $r = \max_{i=1,\dots,n}\{x_i \cdot q_i \cdot d_i\}$.

Table 7. Parameters of the multigraphs G and G' representing the bus network.

	multigraph G	multigraph G'
number of vertices	1211	1211
number of arcs	5549	35610
minimum indegree of vertex	1	6
minimum outdegree of vertex	1	6
minimum degree of vertex	2	13
maximum indegree of vertex	13	122
maximum outdegree of vertex	13	122
maximum degree of vertex	26	244

3. Experimental test results

The SOLVEBBR algorithm was implemented in C++ and the test experiments were carried out on a 2.9 GHz Intel(R) Core(TM) i5-4570S CPU computer with 4 GB of RAM, running Windows 7 Enterprise x64. The results of the tests are compared with those obtained by the algorithm denoted by WID12 and presented in (Widuch, 2012). In the tests, a random generated bus network consisting of 1211 stops divided into 26 zones was used. In the network, buses of 500 bus lines are run: the shortest length of the route of a bus line equals 6 and the longest length of the route equals 29. The parameters of the multigraphs G and G' representing the bus network are presented in Table 7.[3] The performed tests had the following goals:

- investigate the properties of computed non-dominated final solutions, i.e., the number of computed final solutions containing cycles, the number of computed final solutions differing from each other only in the times of departure from vertices belonging to the paths, and the number of computed final solutions obtained from dominated partial solutions;
- compare the computation time and the number of computed partial solutions using the tested algorithms; all computed partial solutions inserted into the list *LPS*$[v_i]$ (line 21 of the procedure SOLVEBBR) during a single experiment were counted; additionally, all analysed and omitted partial solutions for which the estimation is negative

[3] An indegree, an outdegree and a degree of a vertex are defined by Jungnickel (1999).

(this means that the final solutions obtained from them are dominated) were counted;

- count the number of computed non-dominated final solutions;
- count the number of computed final solutions satisfying the relationships R1-R9 defined in Section 2.4.

We carried out 7,326,550 test experiments using the SOLVEBBR and WID12 algorithms. The aim of a single test was to find all paths belonging to the set of non-dominated solutions for the given pair of the start v_s and the final v_e vertices and the time of staring travel T_s at v_s. The tests were carried out for all pairs of vertices and for the times of starting travel equal to $T_s =$ 0:00, 8:00, 12:00, 16:00 and 20:00, respectively. The results of the tests are presented depending on the number of changes in the path and the length of the path.

The results of test experiments using the algorithms are presented in Tables 8 and 9. Here we show

- the number of all computed non-dominated final solutions during all experiments and the maximal number of computed non-dominated final solutions in a single experiment,
- the maximal number of computed and omitted partial solutions in a single experiment,
- maximal computation time in milliseconds.

Additionally, in charts (Fig. 5) we show

- the percentage number of non-dominated solutions containing cycles,
- the percentage number of solutions with the same time and cost of travel and the same path, differing from each other only in the times of departure from the vertices,
- the percentage number of solutions obtained from dominated partial solutions.

The number of solutions differing from each other only in the times of departure and the of solutions obtained from dominated partial solutions were computed by mutual comparison of solutions obtained in a single experiment. According to Theorem 1 and Lemma 1, the paths containing cycles and those differing from each other only in the times of departure belong to the set of non-dominated solutions if we change at least once in these paths. Therefore, there was no such path without a change. The percentage number of these determined solutions grows with the length of the path. The maximum length of a determined path equals 95, and it should be pointed out that all paths of this length contain a cycle. About 80% of the determined paths of the maximum length differ from each other only in the times of departure from vertices.

The computation time depends on the number of computed partial and final non-dominated solutions. With an increase in the length of the path and the number of changes, the searched space of solutions grows and so does the number of computed partial solutions. The maximal computation time using the SOLVEBBR algorithm equals 343 milliseconds and was a result of determining the paths in which the bus change 6 and 7 times was made while the length of the path equals 41–70 vertices. In these cases the number of computed partial solutions is maximal and equal to 10,121.

The SOLVEBBR algorithm applies conditions for estimation of partial solutions described in Sections 2.3 and 2.4, but the WID12 algorithm does not apply them. For that reason, a larger space of solutions by the WID12 algorithm than by the SOLVEBBR algorithm is searched, the WID12 algorithm computes a larger number of partial solutions than the SOLVEBBR algorithm and a larger number of solutions are analysed by the WID12 algorithm. Therefore, the computation time is larger for the WID12 algorithm than the for SOLVEBBR algorithm.

The number of computed partial solutions also depends on their representation. The partial solutions stored in the list *LPS*$[v_i]$ of the vertex v_i contain the list of pointers *LPP*$_k$ to the partial solutions for the vertex v_k, which precedes v_i in the path from the start vertex v_s to v_i (see Section 2.5). Thus the partial solution represents many paths from v_s to v_i. It is an important difference to the WID12 algorithm, wherein the partial solution represents a single path from v_s to v_i. Therefore, the number of computed partial solutions by the SOLVEBBR algorithm is smaller than the number of partial solutions computed and analysed by the WID12 algorithm. In the best case, it computes 66 times less partial solutions. For that reason, the computation time is shorter for the SOLVEBBR algorithm than for the WID12 algorithm, and in the best case it is 47 times less. It depends on checking by the procedure INPATH if the vertex v_i belongs to the path represented by the partial solution S_k (line 12 of the SOLVEBBR algorithm). It is executed in $O(k)$ time, where k equals the length of the path, but in the WID12 algorithm it is checked in $O(1)$ time.

The representation of a partial solution and conditions used for estimation of partial solutions have also influence the number of omitted partial solutions. It should be pointed out that each omitted partial solution was analysed, but due to negative estimation it was not inserted into the list *LPS*$[v_i]$. Therefore the conditions for estimation of partial solutions applied by the SOLVEBBR algorithm and the proposed representation of the partial solution decrease the number of analysed and omitted partial solutions. In the best case it is 214 times less.

Table 8. Number of all computed non-dominated final solutions during all experiments, the maximal number of computed non-dominated final solutions in a single experiment, the maximal number of computed and omitted partial solutions in a single experiment and maximal computation time for the SOLVEBBR and WID12 algorithms. They depend on the number of changes.

Number of changes	Maximal number of non-dominated final solutions computed in single experiment	Number of non-dominated final solutions computed during all experiments	Maximal number of partial solutions computed by algorithm		Maximal number of partial solutions omitted by algorithm		Maximal computation time (in ms) for algorithm	
			SOLVEBBR	WID12	SOLVEBBR	WID12	SOLVEBBR	WID12
0	3	161251	962	9789	18249	356123	16	16
1	52	2018583	3375	70275	46119	2224413	16	16
2	1966	10022638	5435	165285	75672	4299398	16	94
3	4772	28648372	7596	416546	80712	12184704	32	140
4	23917	57132301	9171	576074	108322	16522381	62	499
5	127259	96012598	10121	622354	118343	17852038	140	733
6	529039	58530211	10121	622354	120729	17852038	343	749
7	228614	7889988	10121	622354	120729	17852038	343	780
8	4149	1516711	10121	564170	120729	16549834	312	780
9	1410	375808	10121	556555	119128	16549834	312	780
10	300	113357	10121	525733	113660	14866546	172	780
11	270	28950	10121	568959	87476	16396951	172	670
12	180	5587	9421	622354	83366	17852038	31	702
13	72	860	6281	208329	76607	6343552	16	749
14	32	32	5557	208329	52898	4655803	16	276

Table 9. Number of all computed non-dominated final solutions during all experiments, the maximal number of computed non-dominated final solutions in a single experiment, the maximal number of computed and omitted partial solutions in a single experiment and maximal computation time for the SOLVEBBR and WID12 algorithms. They depend on the length of the path.

Length of path	Maximal number of non-dominated final solutions computed in single experiment	Number of non-dominated final solutions computed during all experiments	Maximal number of partial solutions computed by algorithm		Maximal number of partial solutions omitted by algorithm		Maximal computation time (in ms) for algorithm	
			SOLVEBBR	WID12	SOLVEBBR	WID12	SOLVEBBR	WID12
1–5	106	295599	946	6601	17091	214944	16	16
6–10	545	1776642	1828	28007	28896	860096	16	16
11–15	2822	4792685	3344	67973	46990	2061928	16	32
16–20	6648	9626750	4323	202677	79282	5748026	16	78
21–25	8447	16043672	6012	202677	85086	5748026	32	219
26–30	20207	22892227	7644	337508	99581	9955280	62	249
31–35	36037	27976184	9114	496767	108817	14218868	63	374
36–40	90347	32729054	9171	622354	118343	17852038	296	608
41–45	106932	37001418	10121	622354	118343	17852038	343	749
46–50	290418	38250850	10121	622354	118343	17852038	343	780
51–55	288822	31927564	10121	622354	120729	17852038	343	780
56–60	228044	20002773	10121	622354	120729	17852038	343	780
61–65	49336	11397158	9905	622354	120729	17852038	343	780
66–70	117832	5403238	9905	622354	120729	17852038	343	749
71–75	12346	1780694	9905	568959	120729	16396951	219	749
76–80	7166	511479	9905	444956	119128	13732192	78	733
81–85	2490	45222	9905	438201	117127	13732192	47	530
86–90	240	3825	7375	265206	107377	6889943	31	436
91–95	60	213	6063	265206	80152	1082443	16	109

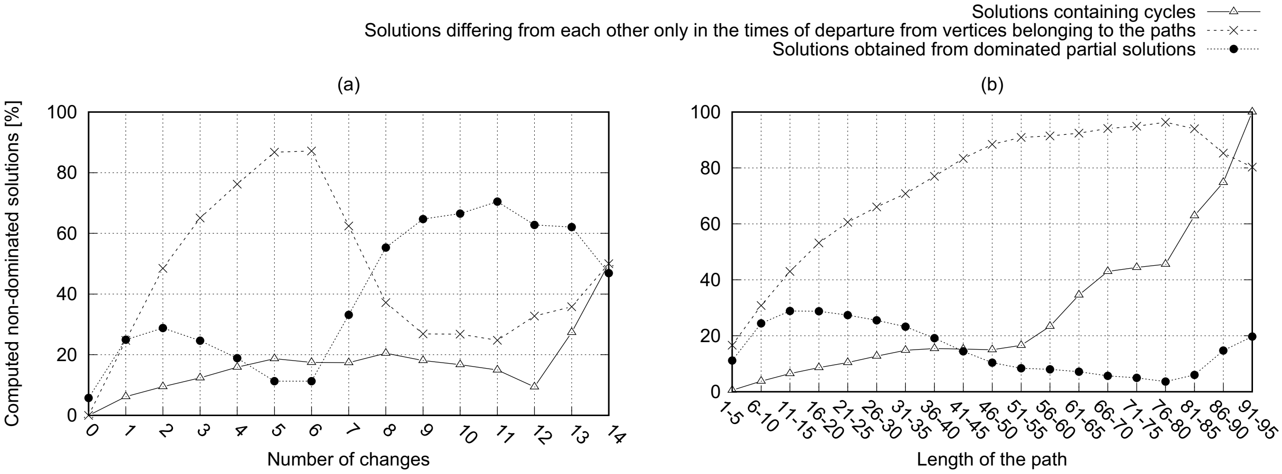

Fig. 5. Properties of all computed non-dominated solutions by the SOLVEBBR algorithm depending on the number of changes (a) and the length of the path (b).

In the single experiment it was not determined more than 3 non-dominated final solutions where we travel without a bus change (Table 8). The maximal number of bus changes equals 14, and there were determined 32 paths with this property, all in a single experiment. During all experiments most paths were determined with lengths from the range 46–50 vertices, and 38,250,850 paths were determined (Table 9). The average number of computed non-dominated final solutions during a single experiment equals 36, where 28 solutions have the same paths and differ from each other only in the times of departure from all vertices belonging to these paths while 6 solutions contain a cycle. The average numbers of partial solutions computed and omitted by the SOLVEBBR algorithm are equal to 305 and 6,918, respectively. This is about 7 and 11 times less than for the WID12 algorithm. This implies that the SOLVEBBR algorithm searches a smaller space of solutions than the WID12 algorithm.

The next goal of the tests was to count the number of determined final solutions satisfying the relationships R1–R9 defined in Section 2.4. It was counted by mutual comparison all pairs of paths belonging to a determined set of non-dominated solutions. The paths that differ from each other only in the times of departure from vertices belonging to these paths are not counted because each path contains at least one vertex v_i for which one of the relationships R1–R9 is satisfied due to the time of travel from v_s to v_i. Consider a pair of paths

$$\begin{aligned} p_{v_s,v_e} &= \langle v_0 = v_s, \dots, v_j, \dots, v_x = v_e \rangle, \\ p'_{v_s,v_e} &= \langle v'_0 = v_s, \dots, v'_k, \dots, v'_y = v_e \rangle, \end{aligned}$$

belonging to the set of non-dominated solutions. The paths are compared if the following conditions are satisfied:

1. These paths have a common vertex v_i, i.e.,

$$\begin{aligned} &\exists j \in \{1, \dots, x-1\} \wedge \\ &\exists k \in \{1, \dots, y-1\} \ : \ v_j = v'_k = v_i. \end{aligned}$$

If there exist many j and k, then the minimal values from among j and k are taken into account.

2. The partial solution p_{v_s,v_i} dominates the partial solution p'_{v_s,v_i}, i.e., $p_{v_s,v_i} \succ p'_{v_s,v_i}$, where

$$\begin{aligned} p_{v_s,v_i} &= \mathrm{sub}_{p_{v_s,v_e}}(v_s, v_i), \\ p'_{v_s,v_i} &= \mathrm{sub}_{p'_{v_s,v_e}}(v_s, v_i). \end{aligned}$$

If the solution satisfies several relationships, then all these cases are counted, and if satisfies the relationship many times, then it is counted only once. The results of the comparison of the solutions are presented in plots (Fig. 6). The determined non-dominated final solutions are compared, therefore only relationships R1, R6 and R8 are satisfied. To check satisfaction of other relationships, it is necessary to determine all possible paths from the start vertex v_s to the final vertex v_e. Thus the dominated solutions cannot be omitted during computation in the SOLVEBBR algorithm. This kind of test is not possible to be carried out within reasonable time due to memory complexity. The plots contain results for all cases considered in Section 2.4, i.e.,

- when v_i is passed without a change in both the compared paths p_{v_s,v_e} and p'_{v_s,v_e} (Figs. 6(a) and (b)),
- when v_i is passed without a change in p_{v_s,v_e} and we change at v_i in p'_{v_s,v_e} (Figs. 6(c) and (d)),
- when we change at v_i in p_{v_s,v_e} and it is passed without a change in p'_{v_s,v_e} (Figs. 6(e) and (f)),

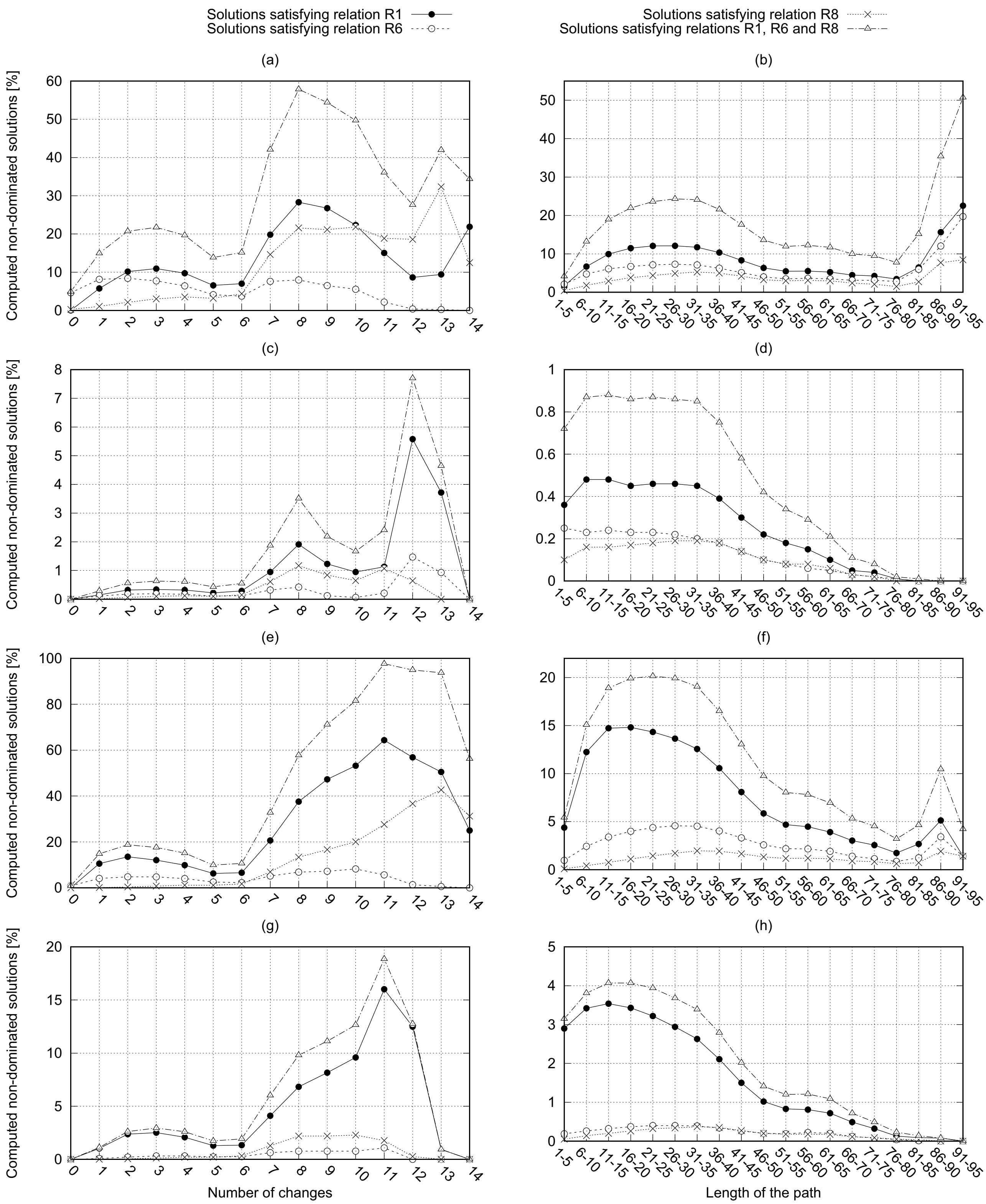

Fig. 6. Percentage number of all computed non-dominated solutions by the SOLVEBBR algorithm satisfying R1, R6 and R8 in relation to the number of changes and the path length.

- when we change at v_i in both the compared paths p_{v_s,v_e} and p'_{v_s,v_e} (Figs. 6(g) and (h)).

The number of solutions satisfying the relationships considered decreases with the length of the path, except for the case when the vertex v_i is passed without a change in both paths p_{v_s,v_e} and p'_{v_s,v_e} (Fig. 6(b)). In this case it grows from the length of 76.

In most cases the relationship R1 is satisfied by solutions if the time and the cost of travel fulfill the conditions

$$T(p_{v_s,v_e}) = T(p'_{v_s,v_e}) \wedge C(p_{v_s,v_e}) = C(p'_{v_s,v_e}). \quad (42)$$

The largest difference between the number of solutions satisfying the relationship R1 and the number of solutions satisfying the relationships R6 and R8 occurs when we change at v_i in p_{v_s,v_e}. This is due to the fact that in most cases the subpath $\mathrm{sub}_{p_{v_s,v_e}}(v_i, v_e)$ is the same as the subpath $\mathrm{sub}_{p'_{v_s,v_e}}(v_i, v_e)$ and the time of travel $T(p_{v_s,v_e}) = T(p'_{v_s,v_e})$. Both paths belong to the set of non-dominated solutions, therefore the cost of travel $C(p_{v_s,v_e}) = C(p'_{v_s,v_e})$ and (42) occurs.

Most solutions satisfy the relationships in the case when v_i is passed without a change in p'_{v_s,v_e} obtained from the dominated partial solution p'_{v_s,v_i} (Figs. 6(a), (b), (e) and (f)). When v_i is passed without a change, the cost of travel $C(p'_{v_s,v_e})$ equals the sum of costs of travel $C(p'_{v_s,v_i})$ and $C(\mathrm{sub}_{p'_{v_s,v_e}}(v_i, v_e))$ decreased by $\Delta c'$. This increases the possibility of obtaining a non-dominated solution from a dominated partial one. Otherwise, when we change at v_i in p'_{v_s,v_i}, we do not decrease the cost of travel $C(p'_{v_s,v_i})$ by $\Delta c'$. Therefore in many cases a final solution obtained from a dominated partial one was a dominated solution. Thus, at most 20% of non-dominated final solutions were obtained from a dominated partial solutions (Figs. 6(c), (d), (g) and (h)).

For each number of changes and each range of the path length a solution obtained from a dominated partial solution exists. Thus, these solutions would not have been determined as a result of omitting dominated partial solutions as in the case of the multigraph with constant weights (Skriver and Andersen, 2000b). It can be seen that the number of solutions obtained from a dominated partial solution decreases with the path length. It should be noted that about 70% solutions where in the path a change is performed 11 times and about 5% solutions where the journey is performed without a change were obtained from dominated partial solutions. Additionally, in each case there exists pair of solutions p_{v_s,v_e} and p'_{v_s,v_e} satisfying any relationship,

$$T(p_{v_s,v_e}) < T(p'_{v_s,v_e}) \wedge C(p_{v_s,v_e}) > C(p'_{v_s,v_e}),$$
$$T(p_{v_s,v_e}) = T(p'_{v_s,v_e}) \wedge C(p_{v_s,v_e}) = C(p'_{v_s,v_e}),$$
$$T(p_{v_s,v_e}) > T(p'_{v_s,v_e}) \wedge C(p_{v_s,v_e}) < C(p'_{v_s,v_e}),$$

that takes place between the time and the cost of travel, where p_{v_s,v_e} is obtained from the partial solution p_{v_s,v_i} and p'_{v_s,v_e} is obtained from the dominated partial solution p'_{v_s,v_i}, i.e., $p_{v_s,v_i} \succ p'_{v_s,v_i}$.

For 1,838,360 tests all non-dominated solutions were obtained from non-dominated partial solutions, and this is about 25% of tests carried out. Thus about 75% of conducted tests contain at least one solution obtained from a dominated partial solution, and for 311,309 tests (which is about 4% of all executed tests) all non-dominated solutions were obtained from dominated partial solutions. The results demonstrate that the monotonicity assumption does not hold in the BBR problem and the proposed estimation of partial solutions applied by the SolveBBR algorithm makes it possible to determine these solutions. Otherwise, i.e., when dominated partial solutions are omitted, it would not determine any of these solutions.

The maximal number of computed non-dominated final solutions in the single test equals 529,183. There are only 145 solutions differing from each other in vertices or arcs belonging to these paths, but 143 solutions satisfying the relationships R1 and R6 from among them. Thus, 143 were obtained from dominated partial solutions. The maximal number of non-dominated solutions obtained from dominated partial ones in a single test equals 2,627 and these solutions satisfying the relationships R1, R6 and R8. The number of all non-dominated solutions determined in this test equals 4,962, and all solutions obtained from dominated partial ones differ from each other in vertices or arcs belonging to these paths. In these two tests the omission of dominated partial solutions would cause a failure in finding valid non-dominated final solutions.

4. Conclusions

In this paper the bicriterion bus routing (BBR) problem was considered and its theoretical analysis was presented. It was shown that a non-dominated final solution can be obtained from a dominated partial one. This is an important property used during the process of determining solutions. We presented a detailed analysis of the conditions that must be fulfilled in order to obtain a non-dominated solution from a dominated partial one.

We proposed a new label correcting algorithm for solving the BBR problem. The proposed representation of a partial solution and methods of estimating partial solutions decrease the number of computed and analysed partial solutions in comparison with the algorithm presented by Widuch (2012). The methods of estimating partial solutions make it possible to determine all non-dominated solutions, which was confirmed by experimental tests. The tests were carried out for all pairs of vertices in a multigraph representing the bus network and for 5 different times of starting travel.

The experimental tests showed that about 75% of the conducted tests contain a non-dominated final solution obtained from a dominated partial one and about 4% of conducted tests contain all non-dominated final solutions obtained from a dominated partial solution.

The problem is known to be NP-complete, and in the worst case the number of solutions grows exponentially with the number of vertices representing the bus stops. On the basis of the test results we found that our procedure exhibits a reasonable execution time for a bus network containing about 1200 stops. Additionally, a test results demonstrate that the number of non-dominated solutions is not exponential in reality. The set of non-dominated solutions determined by the algorithm may be a basis for choosing by a passenger or an application a single bus route. It is chosen on the basis of additional criteria, for example, the length of the route, the number of changes, the total waiting time at bus stops, etc.

References

Addor, J.A., Amponsah, S.K., Annan, J. and Sebil, C. (2013). School bus routing: A case study of wood bridge school complex, Sekondi-Takoradi, Ghana, *International Journal of Business and Social Research* **3**(12): 26–36.

Arias-Rojas, J.S., Jiménez, J.F. and Montoya-Torres, J.R. (2012). Solving of school bus routing problem by ant colony optimization, *Revista EIA* **9**(17): 193–208.

Azevedo, J.A. and Martins, E.Q.V. (1991). An algorithm for the multiobjective shortest path problem on acyclic networks, *Investigação Operacional* **11**(1): 52–69.

Bronshtein, E.M. and Vagapova, D.M. (2015). Comparative analysis of application of heuristic and metaheuristic algorithms to the school bus routing problem, *Informatics and Its Applications* **9**(2): 56–62.

Brumbaugh-Smith, J. and Shier, D. (1989). An empirical investigation of some bicriterion shortest path algorithms, *European Journal of Operational Research* **43**(2): 216–224.

Caceres, H., Batta, R. and He, Q. (2014). School bus routing with stochastic demand and duration constraints, *Transportation Research Board 93rd Annual Meeting, Washington, DC, USA*, pp. 1–23.

Carraway, R.L., Morin, T.L. and Moskowitz, H. (1990). Generalized dynamic programming for multicriteria optimization, *European Journal of Operational Research* **44**(1): 95–104.

Chalkia, E., Grau, J.M.S., Bekiaris, E., Ayfandopoulou, G., Ferarini, C. and Mitsakis, E. (2014). Routing algorithms for the safe transportation of pupils to school using school buses, *Transport Research Arena (TRA) 5th Conference: Transport Solutions from Research to Deployment, Paris, France*, pp. 1–10.

Chen, P. and Nie, Y.M. (2013). Bicriterion shortest path problem with a general nonadditive cost, *Transportation Research B: Methodological* **57**: 419–435.

Chen, X., Kong, Y., Dang, L., Hou, Y. and Ye, X. (2015). Exact and metaheuristic approaches for a bi-objective school bus scheduling problem, *PLoS ONE* **10**(7): 1–20. DOI:10.1371/journal.pone.0132600.

Climaco, J.C. and Martins, E.Q.V. (1982). A bicriterion shortest path algorithm, *European Journal of Operational Research* **11**(4): 399–404.

Corley, H.W. and Moon, I.D. (1985). Shortest paths in networks with vector weights, *Journal of Optimization Theory and Application* **46**(1): 79–86.

Daellenbach, H.G. and De Kluyver, C.A. (1980). Note on multiple objective dynamic programming, *Journal of the Operational Research Society* **31**(7): 591–594.

Dell'Olmo, P., Gentili, M. and Scozzari, A. (2005). On finding dissimilar Pareto-optimal paths, *European Journal of Operational Research* **162**(1): 70–82.

Díaz-Parra, O., Ruiz-Vanoye, J.A., Buenabad-Arias, A. and Cocón, F. (2012). A vertical transfer algorithm for the school bus routing problem, *4th World Congress on Nature and Biologically Inspired Computing (NaBIC), Mexico City, Mexico*, pp. 66–71.

Ehrgott, M. (2000). *Multicriteria Optimization*, Springer-Verlag, Berlin.

Ellegood, W. A., Campbell, J. F. and North, J. (2015). Continuous approximation models for mixed load school bus routing, *Transportation Research B* **77**: 182–198.

Euchi, J. and Mraihi, R. (2012). The urban bus routing problem in the Tunisian case by the hybrid artificial ant colony algorithm, *Swarm and Evolutionary Computation* **2**: 15–24.

Garey, M. and Johnson, D. (1990). *Computers and Intractibility: A Guide to the Theory of NP-Completeness*, W.H. Freeman & Co., New York, NY.

Hansen, P. (1980). Bicriterion path problems, *in* G. Fandel and T. Gal (Eds.), *Multiple Criteria Decision Making: Theory and Application*, Springer-Verlag, Berlin, pp. 109–127.

Henig, M.I. (1985). The shortest path problem with two objective functions, *European Journal of Operational Research* **25**(2): 281–291.

Huang, L.C., Guan, W. and Xiong, J. (2014). Routing design optimization of bus joint for passenger transfer centers, *in* M. Sun and Y. Zhang (Eds.), *Renewable Energy and Environmental Technology*, Applied Mechanics and Materials, Vol. 448, Trans Tech Publications, Zurich, pp. 4140–4149.

Jungnickel, D. (1999). *Graphs, Networks and Algorithms, 2nd Edition*, Springer-Verlag, Berlin.

Kang, M., Kim, S.K., Felan, J.T., Choi, H.R. and Cho, M. (2015). Development of a genetic algorithm for the school bus routing problem, *International Journal of Software Engineering and Its Applications* **9**(5): 107–126.

Kim, B.I., Kim, S. and Park, J. (2012). A school bus scheduling problem, *European Journal of Operational Research* **218**(2): 577–585.

Kim, T. and Park, B.J. (2013). Model and algorithm for solving school bus problem, *Journal of Emerging Trends in Computing and Information Sciences* **4**(8): 596–600.

Kinable, J., Spieksma, F.C.R. and Berghe, G.V. (2014). School bus routing—a column generation approach, *International Transactions in Operational Research* **21**(3): 453–478.

López, E.R. and Romero, J. (2015). A hybrid column generation and clustering approach to the school bus routing problem with time windows, *Ingeniería* **20**(1): 111–127.

Machuca, E., Mandow, L. and Pérez de la Cruz, J.L. (2009). An evaluation of heuristic functions for bicriterion shortest path problems, *Proceedings of the 14 Portuguese Conference on Artificial Inteligence (EPIA 2009), Aveiro, Portugal*, pp. 205–216.

Mandow, L. and Pérez de la Cruz, J.L. (2008). Path recovery in frontier search for multiobjective shortest path problems, *Journal of Intelligent Manufacturing* **21**(1): 89–99.

Manumbu, D.M., Mujuni, E. and Kuznetsov, D. (2014). A simulated annealing algorithm for solving the school bus routing problem: A case study of Dar es Salaam, *Computer Engineering and Intelligent Systems* **5**(8): 44–58.

Martí, R., González Velarde, J.L. and Duarte, A. (2009). Heuristics for the bi-objective path dissimilarity problem, *Computers & Operations Research* **36**(11): 2905–2912.

Martins, E.Q.V. (1984). On a multicriteria shortest path problem, *European Journal of Operational Research* **16**(2): 236–245.

Martins, E.Q.V., Pascoal, M.M.B., Rasteiro, D.M.L.D. and Santos, J.L.E. (1999). The optimal path problem, *Investigação Operacional* **19**(1): 43–60.

Mote, J., Murthy, I. and Olson, D.L. (1991). A parametric approach to solving bicriterion shortest path problems, *European Journal of Operational Research* **53**(1): 81–82.

Newton, R.M. and Thomas, W.H. (1969). Design of school bus routes by computer, *Socio-Economic Planning Sciences* **3**(1): 75–85.

Pacheco, J., Caballero, R., Laguna, M. and Molina, J. (2013). Bi-objective bus routing: An application to school buses in rural areas, *Transportation Science* **47**(3): 397–411.

Pareto, V. (1896). *Course d'Economie Politique*, F. Rouge, Lausanne.

Park, J. and Kim, B.-I. (2010). The school bus routing problem: A review, *European Journal of Operational Research* **202**(2): 311–319.

Raith, A. and Ehrgott, M. (2009). A comparison of solution strategies for biobjective shortest path problems, *Journal Computers and Operations Research* **36**(4): 1299–1331.

Riera-Ledesma, J. and Salazar-González, J.J. (2012). Solving school bus routing using the multiple vehicle traveling purchaser problem: A branch-and-cut approach, *Computers & Operations Research* **39**(2): 391–404.

Riera-Ledesma, J. and Salazar-González, J.J. (2013). A column generation approach for a school bus routing problem with resource constraints, *Computers & Operations Research* **40**(2): 566–583.

Schittekat, P., Kinable, J., Sörensen, K., Sevaux, M. and Spieksma, F. (2013). A metaheuristic for the school bus routing problem with bus stop selection, *European Journal of Operational Research* **229**(2): 518–528.

Serafini, P. (1987). Some considerations about computational complexity for multi objective combinatorial problems, *in* J. Jahn and W. Krabs (Eds.), *Recent Advances and Historical Development of Vector Optimization*, Lecture Notes in Economics and Mathematical Systems, Vol. 294, Springer, Berlin/Heidelberg, pp. 222–232.

Sghaier, S.B., Guedria, N.B. and Mraihi, R. (2013). Solving school bus routing problem with genetic algorithm, *International Conference on Advanced Logistics and Transport (ICALT'2013), Sousse, Tunisia*, pp. 7–12.

Siqueira, V.S., Silva, F.J.E.L., Silva, E.N., Silva, R.V.S. and Rocha, M.L. (2016). Implementation of the metaheuristic grasp applied to the school bus routing problem, *International Journal of e-Education, e-Business, e-Management and e-Learning* **6**(2): 137–145.

Skriver, A.J.V. and Andersen, K.A. (2000a). A classification of bicriteria shortest path (BSP) algorithms, *Asia-Pacific Journal of Operational Research* **17**(2): 199–212.

Skriver, A.J.V. and Andersen, K.A. (2000b). A label correcting approach for solving bicriterion shortest-path problems, *Computers & Operations Research* **27**(6): 507–524.

Tung, C.T. and Chew, K.L. (1988). A bicriterion Pareto-optimal path algorithm, *Asia-Pacific Journal of Operational Research* **5**(2): 166–172.

Tung, C.T. and Chew, K.L. (1992). A bicriterion Pareto-optimal path algorithm, *European Journal of Operational Research* **62**(2): 203–209.

Widuch, J. (2012). A label correcting algorithm for the bus routing problem, *Fundamenta Informaticae* **118**(3): 305–326.

Widuch, J. (2013). A label correcting algorithm with storing partial solutions to solving the bus routing problem, *Informatica* **24**(3): 461–484.

Worwa, K. (2014). A case study in school transportation logistics, *Research in Logistics & Production* **4**(1): 45–54.

Yigit, T. and Unsal, O. (2016). Using the ant colony algorithm for real-time automatic route of school buses, *International Arab Journal of Information Technology* **13**(5): 559–565.

Jacek Widuch was awarded the PhD degree at the Silesian University of Technology in 2008. At present he is an assistant professor in the Institute Informatics there. His main research interests are connected with multicriteria optimization algorithms in transportation problems and parallel computing. He is a member of the Upper Silesian Regional Committee of the Polish School Contest in Informatics.

Received: 25 January 2016
Revised: 13 July 2016
Accepted: 10 November 2016