IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 20, 2021, accepted August 14, 2021, date of publication August 24, 2021, date of current version September 29, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3107676

Public Service System Design With Conflicting Criteria

JAROSLAV JANACEK AND RENE FABRICIUS
Faculty of Management Science and Informatics, University of Žilina, 010 26 Žilina, Slovakia

Corresponding author: Jaroslav Janacek (jaroslav.janacek@fri.uniza.sk)

This work was supported in part by the Research Grant VEGA 1/0089/19 (''Data analysis methods and decisions support tools for service systems supporting electric vehicles''), and in part by the Slovak Research and Development Agency under Contract APVV-19-0441.

ABSTRACT Multi-criteria optimization problems represent a crucial task for any designer of a public service system due to conflicting criteria, which have to be faced. Possible solution of this almost unsolvable situation can be seen in obtaining a series of solutions, where it is impossible to improve one of the criteria without worsening some of the others. Full set of such non-dominated solutions is called Pareto front. The multi-criteria optimization problem can be solved by submitting the Pareto front or its approximation on contracting authority's board for possible negotiation with representative of public. This paper deals with methods, which are able to produce a series of non-dominated solutions of bi-criteria public service system design problem. The first criterion considered is average response time and the second one corresponds to the number of users located behind a given limit of response time. We suggest two approaches to the problem. The first of them is an exact approach, which produces Pareto front of the bi-criteria problem. The second approach makes use of an evolutionary hybridized algorithm, which uses so called elite set, to save temporary non-dominated solutions. In the computational study, we try to verify the hypothesis that it is possible to approximate the Pareto front by utilizing the evolutionary algorithm. We also suggest a way of hybridization and tuning the algorithm to obtain a good approximation in acceptable computational time.

INDEX TERMS Public service system, multi-objective problem, Pareto optimization, exact approach, hybridized genetic algorithm.

I. INTRODUCTION

Public service systems are created and operated on public sources to make public's life safer and more comfortable. The systems provide inhabitants of a concerned geographical region with various kinds of necessary service and they are subjected to quality evaluation by addressed groups of concerned inhabitants. As population of a geographical region is heterogeneous from the points of view of age, profession, dwelling place size and location, various criteria are applied to the system functioning evaluation, especially, when the system is to be designed or updated. Some of the criteria may be conflicting, i.e., system changes which improve one of the criteria cause impairment of the other ones. Due to impossibility to compare the criteria on a quantitative base, the selection of the final system design is matter of a public consensus. To enable the associated negotiation of the potential system founder, we concentrate our effort on the ways of obtaining a representative set of public service system designs in the form of so called non-dominated solutions [1], [2].

The public service system design problem has been studied as a task of selection of p center locations from a set of possible service center locations so that a given criterion value is minimal. The original approaches [3]–[7] evaluated quality of the p-center deployment by the average response time of the system on a user's service request. These approaches modelled the problems as the weighted p-median problem and made use of exact and heuristic tools for the problem solving. An important success in large-size p-median problem solving was achieved by application of the radial approach [8]–[12]. The min-sum weighted p-median problem formulation was generalized to capture characteristic of a real public service system with random emerging demands and limited capacity of the service centers by introducing probability values of the situations, when the nearest, second nearest and so on up to r-nearest service center is the nearest available center due to occupation of the closer centers by earlier demands [13].

The associate editor coordinating the review of this manuscript and approving it for publication was Juan Liu.

The radial approach to this generalized p-median problem was developed and tested by [14].

The above-mentioned approaches include usage of an exact solving tools based mostly on branch-and-bound searching strategy, which consumes enormous portion of computational time to verify optimality of the best-found-solution.

To avoid this computational burden, an attention of professionals was directed to heuristic approaches based on imitation of biological processes [15]–[18]. To obtain a good set of non-dominated feasible solutions, an evolutionary metaheuristic seems to be a convenient tool, because an evolutionary metaheuristic can build up a set of non-dominated solutions continuously in the form of an elite set [2].

At the same time, p-location problems with so called criteria of fairness were studied starting with p-center problem up to the strongest fairness scheme applicable to the public service systems [19]. The strongest formulation [20] is called the lexicographic min-max criterion. A weaker, but more frequent, fair criterion is the number of system users, which are situated behind a given response time limit.

Within this paper, we will study approaches to the two-objective problem of the public service system design with randomly emerging users' demands and limited capacity of the service centers. The main goal of the study is to compare an exact approach to a heuristic one from the point of efficiency of obtaining a set of non-dominated solutions of the two-objective problem, where the primary objective corresponds with the average response time and the secondary one is defined as the number of users located behind the given time limit. The exact approach is based on usage of a special bisection procedure, which employs the radial formulation of the generalized weighted p-median problem. The studied heuristic approach has been developed from a genetic algorithm with elite set, which is hybridized with a simple improving swap heuristic.

The remainder of the paper is organized as follows: section II is devoted to the radial formulation of the min-sum problem with limited min-max criterion. The exact approach to Pareto front determination is described in section III and metaheuristic approach is explained in section IV. The associated numerical experiments are reported in section V. The results and findings are summarized in Section VI.

II. ICONFLICTING CRITERIA AND RADIAL FORMULATION OF THE PROBLEM

A. CONFLICTING CRITERIA OF THE PUBLIC SERVICE SYSTEMS

The public service system design problem can be formulated as a task to deploy p facilities in a set of m possible locations so that an objective expressing a disutility perceived by serviced population is minimal. Disutility can take different forms and it can be modelled by many ways depending on system users' point of view and also on accuracy, with which the real service system characteristics are described. Within this paper, we will study two of the often-mentioned objectives. The primal objective is an average response time of the system and the secondary objective is expressed by the number of inhabitants behind a given limit D of generally accepted response time. To formulate the associated models, we introduce the following denotations and assumptions. We assume that the serviced population of a given geographical area is concentrated in n dwelling places and the number of inhabitants of j-th dwelling place is denoted by the symbol b_j. It is supposed that the value b_j is proportional to the number of average service visits at the dwelling place and that a visit will be provided by a vehicle starting from a located service center. The time-distance between a possible service center location i and a dwelling place j is denoted by d_{ij}.

As a real public service system with randomly emerging demands for service operates as a queuing system, the situation must be considered that the nearest service center is occupied by servicing an earlier demand and thus the current demand can be serviced by the second, third or r-th nearest center depending on their current occupancy. To model this situation, we introduce the values $q_1, q_2, \ldots, q_r$, which express probabilities that the nearest, the second nearest, and up to r-th nearest center is the first available one, which can service the current demand.

To model decisions on service center deployment, a binary variable $y_i \in \{0, 1\}$ will be introduced for each possible service center location $i = 1,.., m$. The variable y_i takes the value of one, if a service center is to be located at location i and it takes the value of zero otherwise. This way, the vector $\boldsymbol{y}$, consisting of the zero-one components, corresponds with a vertex of a unit hypercube in m-dimensional space. The set $\boldsymbol{Y}$ of all feasible solutions of the public service system design problem is a sub-set of the unit hypercube vertices and it is described by (1).

$$\boldsymbol{Y} = \{\boldsymbol{y} \in \{0, 1\}^m, \sum_{i=1}^{m} y_i = p\} \tag{1}$$

The primary objective f_1 proportional to the average response time is modelled by (2).

$$f_1(\boldsymbol{y}) = \sum_{j=1}^{n} b_j \sum_{k=1}^{r} q_k min_k\{d_{ij} : i = 1, \ldots, m, y_i = 1\} \tag{2}$$

The result of operator $\min_k\{a_1, a_2, \ldots, a_m\}$ used in (2) gives the k-th minimal value of a finite set of real values $a_1, a_2, \ldots, a_m$, where $k < m$.

As the input set of the operator consists of all distances from dwelling place j to located service centers, the operator produces the distance from the dwelling place to the k-th nearest center. This distance is multiplied by probability q_k of the case that the k-th nearest center will be the first available one and thus the sum of these products gives the average response time of the system to a demand at the dwelling place j.

The second objective f_2 is defined by the equation (3) for a given value D of a response time threshold.

$$f_2(\boldsymbol{y}) = \sum_{j=1}^{n} b_j max\{0, sign(min\{d_{ij} : i = 1, \ldots, m, y_i = 1\} - D)\} \quad (3)$$

The operator $\max\{0, \text{sign}(\min\{d_{ij}: i = 1, \ldots, m, y_i = 1\} - D)\}$ gives the value of one if the distance between the user location j and the nearest service center is greater than the threshold D. In the other cases, the resulting value equals to zero. This way, the quantities of inhabitants of dwelling places distant at most D from the nearest service center will not be included in the sum, which determines the value of $f_2(\boldsymbol{y})$ for given feasible solution $\boldsymbol{y}$.

It can be easily seen that the criteria (2) and (3) are in conflict. Let us demonstrate it by the "toy-example" depicted in Fig. 1, where two dwelling places are considered and they are depicted by circles. The values inside the circles give the numbers of inhabitants, i.e., $b_1 = 100$ and $b_2 = 10$. For bigger simplicity of the presented example, the parameters $p = 1$ and $r = 1$ are considered and only two possible center locations are taken into account. The possible center locations are depicted as squares and the distances between pairs of locations are placed under the links, which connect the locations. Let the threshold D take the value of two. It can be seen that the objective function values are $f_1(\boldsymbol{y}^1) = 130$ and $f_2(\boldsymbol{y}^1) = 10$ for the solution $\boldsymbol{y}^1 = (1, 0)$, whereas the solution $\boldsymbol{y}^2 = (0, 1)$ has the objective values $f_1(\boldsymbol{y}^2) = 220$ and $f_2(\boldsymbol{y}^2) = 0$.

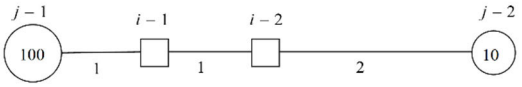

FIGURE 1. Simple example of the solved problem. Circles represent dwelling places with their population and squares represent possible center locations.

If the bi-criteria public service system design problem is studied with the goal to choose some representative solution, it can be found that it is impossible to choose a unique solution based on exact approach. That is why, the notion of non-dominated solution was formulated and a set of non-dominated solutions is offered as a representative output instead of a single solution.

We say that a solution $\boldsymbol{y} \in \boldsymbol{Y}$ dominates a solution $\boldsymbol{x} \in \boldsymbol{Y}$ if $f_1(\boldsymbol{y}) \neq f_1(\boldsymbol{x})$ or $f_2(\boldsymbol{y}) \neq f_2(\boldsymbol{x})$ and the following couple of inequalities holds $f_1(\boldsymbol{y}) \leq f_1(\boldsymbol{x})$ and $f_2(\boldsymbol{y}) \leq f_2(\boldsymbol{x})$. This definition assumes that the both objective functions are to be minimized. The definition considers solutions $\boldsymbol{y}$ and $\boldsymbol{x}$ equivalent from the point of dominancy if they do not differ in both objective function values. After these preliminaries, we can define non-dominated solution of the problem as a solution, which is not dominated by any feasible problem solution. One of possible mathematical definitions follows.

A solution $\boldsymbol{y} \in \boldsymbol{Y}$ is denoted as a non-dominated solution of the discussed problem if every other solution $\boldsymbol{x} \in \boldsymbol{Y}$ satisfying $f_1(\boldsymbol{y}) \neq f_1(\boldsymbol{x})$ or $f_2(\boldsymbol{y}) \neq f_2(\boldsymbol{x})$ meets the following clause: $f_1(\boldsymbol{y}) < f_1(\boldsymbol{x})$ or $f_2(\boldsymbol{y}) < f_2(\boldsymbol{x})$.

As the set $\boldsymbol{Y}$ of all feasible solutions is finite, the set of non-dominated solutions must be also finite and non-empty taking into account that optimal solutions for each of the objectives must exist.

Relations among dominated and non-dominated solutions are depicted in Fig. 2, where a feasible solution $\boldsymbol{y}$ is represented by a pair $(f_1(\boldsymbol{y}), f_2(\boldsymbol{y}))$.

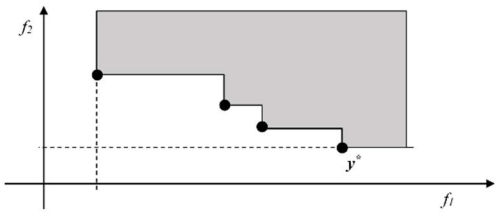

FIGURE 2. Non-dominated solutions are depicted by small black circles and the area of dominated solutions is filled by gray color.

The right most non-dominated solution $\boldsymbol{y}$ can be determined by the following process. First, solve the problem (4) and obtain optimal solution $\boldsymbol{x}^*$ with objective function value $f_2(\boldsymbol{x}^*)$.

$$min\{f_2(\boldsymbol{y}) : \boldsymbol{y} \in \boldsymbol{Y}\} \quad (4)$$

Second, solve the problem (5) and the resulting solution $\boldsymbol{y}^*$ corresponds to the right-most non-dominated solution.

$$min\{f_1(\boldsymbol{y}) : \boldsymbol{y} \in \boldsymbol{Y}, f_2(\boldsymbol{y}) \leq f_2(\boldsymbol{x}^*)\} \quad (5)$$

In the case of finite set of all feasible solutions, a solving method of the problem (5) can be used to obtain the sub-set of all non-dominated solutions. Such a sub-set is called Pareto front.

B. RADIAL MODEL OF RESTRICTED PUBLIC SERVICE SYSTEM DESIGN PROBLEM

As the solving methods of the problem (5) may play a significant role in searching for non-dominated solution, we introduce a radial formulation of the problem. The radial model in connection with a common IP-solver may constitute an important solving tool of the problem. To complete the radial model for the generalized objective function f_1, we start with several assumptions on constants described in the previous sub-section. It is generally supposed for the probability values that $q_1 > q_2 > \ldots > q_r$. We assume that the time-distance d_{ij} between service center location i and user location j is integer and M denotes the maximal value of all distances. We will use the earlier introduced location variables $y_i \in \{0, 1\}$ for $i = 1, \ldots, m$ and, in addition, we introduce a series of auxiliary zero-one variables $x_{jsk} \in \{0, 1\}$ for $j = 1, \ldots, n$, $s = 0, \ldots, M - 1$ and $k = 1, \ldots, r$, where subscripts j corresponds to user location, s stands for a value of distance and k denotes order of the first available service center in the sequence of the nearest, second nearest and so on up to

r-th nearest located center to the user location j. The variable will take the value of one, if the k-th nearest center to a user j is more distant than s and it will take the zero value otherwise. Then, the sum $x_{j0k} + x_{j1k} + \cdots + x_{jM-1k}$ represents the k-th smallest distance from user j to a located center, due to the following reason.

Assuming that the integer time-distance from location j to the k-th nearest center is t, each variable x_{jsk} of the sum, middle subscript of which is less than t, contributes by one unit to the sum. As the remaining variables contribute by zero values, the sum equals to t. It must be noted that this substitution approximates the real time distance with preciseness of one unit.

To be able to complete the model, we introduce constants a_{ijs} for each $i = 1, \ldots, m$, $j = 1, \ldots, n$, and $s = 0, \ldots, M-1$ so that $a_{ijs} = 1$ if and only if $d_{ij} \leq s$ and $a_{ijs} = 0$ otherwise.

In addition, we introduce series of zero-one variables $z_j \in \{0, 1\}$ for $j = 1, \ldots, n$, where z_j takes the value of one, if the distance from j to the nearest located service center exceeds the threshold D and it takes the zero value otherwise

Using the above introduced structures and decision variables, we constitute the following model [11].

$$Minimize\ f_1(\mathbf{y}, \mathbf{x}, \mathbf{z}) = \sum_{j=1}^{n} b_j \sum_{k=1}^{r} q_k \sum_{s=0}^{M-1} x_{jsk} \quad (6)$$

$$Subject\ to \quad \sum_{i=1}^{m} y_i = p \quad (7)$$

$$\sum_{k=1}^{r} x_{jsk} + \sum_{i=1}^{m} a_{ijs} y_i \geq r \ for\ j = 1, \ldots, n, \quad s = 0, \ldots, M-1 \quad (8)$$

$$\sum_{j=1}^{n} b_j z_j \leq B \quad (9)$$

$$z_j + \sum_{i=1}^{m} a_{ijD} y_i \geq 1 \ for\ j = 1, \ldots n \quad (10)$$

$$y_i \in \{0, 1\} \quad for\ i = 1, \ldots, m \quad (11)$$

$$x_{jsk} \in \{0, 1\} \quad for\ j = 1, \ldots, n, \quad s = 0, \ldots, M-1,\ k = 1, \ldots, r \quad (12)$$

$$z_j \in \{0, 1\} \quad for\ j = 1, \ldots, n \quad (13)$$

The objective function (6) expresses the expected sum of response time distances and it is denoted as generalized disutility. Constraint (7) limits the number of located service centers by the constant p. Each constraint of the series (8) gives relation between location variables y_i and those auxiliary variables x_{jsk}, which are connected with distance s from users' location j to r-nearest located centers. It can be noticed that if each center is located outside the radius s from the user's location j then the second sum of the left-hand-side of (8) equals to zero and thus the first sum has to equal to r. In general, the sum of auxiliary variables x_{jsk} equals at least to the positive difference of r and the number of service centers, which are located in the radius s from the user's location j.

As the minimized objective function (6) can be seen as a positively weighted double sum of formulae $q_1 x_{js1}{}^{+}{}^{+} q_r x_{jsr}$ and the sequence $\{q_k\}$ is decreasing, the case $x_{js1}{}^{+}{}^{+} x_{jsr} < r$ will cause that the integer value of the sum $x_{js1}{}^{+}{}^{+} x_{jsr}$ will deployed among the individual zero-one variables so that the variables with the lowest subscript k will be given by zero value, to minimize the expression $q_1\ x_{js1}{}^{+}{}^{+} q_r x_{jsr}$. This mode of deployment ensures that $x_{j0k} + x_{j1k}{}^{+} \cdots {}^{+} x_{jM-1k} \leq x_{j0k+1} + x_{j1k+1}{}^{+} \cdots {}^{+} x_{jM-1k+1}$ holds for $k = 1, \ldots, r-1$ and thus $x_{j0k} + x_{j1k}{}^{+} \cdots {}^{+} x_{jM-1k}$ corresponds to the distance from location j to the k-th nearest service center in the optimal solution of the problem.

The constraints (9) and (10) assure that the number of inhabitants behind the threshold D is less than a given limit B. More specifically, the constraint (10) assures the implication that if no center is located in radius D from the dwelling place j, then z_j must equal to one. It causes that quantity b_j will be included in the sum of the left-hand-side of (9).

The model (6) – (13) imitates performance of a real emergency service system with bigger accuracy than the previous models, which were based on the assumption that a user's demand is satisfied from the nearest service center. The bigger accuracy is balanced by bigger complexity of the suggested model and thus it is questionable, which value of r should be applied to the modelled situation to keep computational time of the associated optimization process in an acceptable limit. Fortunately, thanks to sharp decrease of the probability values q_k in the real systems, the relevant value of r is about three [22].

III. AN EXACT APPROACH TO PARETO FRONT DETERMINATION

Let us denote $F_1(u)$ the optimal value of (6) obtained by solving the problem (6) – (13) for the value of u substituted for the right-hand-side B in (9). Let $\mathbf{y}(u)$ be the associated optimal solution of the problem (6)-(13). It holds $f_1(\mathbf{y}(u)) = F_1(u)$ and $f_2(\mathbf{y}(u)) \leq u$. As diminishing of B makes the set of feasible solutions of (6) – (13) smaller, then $F_1(U) \leq F_1(u)$ for every pair of reals U and u, which satisfies $U > u$. It can be easily derived from the above-mentioned definition of non-dominated solution that if $F_1(U) = F_1(u)$ holds, then there is no non-dominated solution $\mathbf{y}$ such that $U \geq f_2(\mathbf{y}) > u$. In addition, if $F_1(U) < F_1(u)$ holds, then there is at least one non dominated solution $\mathbf{y}$, for which $f_2(\mathbf{y}(U)) \geq f_2(\mathbf{y}) > u$ and $F_1(U) \leq f_1(\mathbf{y}) < F_1(u)$.

To substantiate correctness of the following algorithm, we draw attention to some features of the solved problem. As mentioned above, the set $\boldsymbol{Y}$ of all feasible solutions of the p-location problem is finite and thus the set of all non-dominated solutions must be finite as well.

As the quantities b_j of inhabitants of the individual dwelling places are integer, the sum at the left-hand-side of the constraint (9) is also integer and thus only integer values can be taken into account, when range of $F_1(u)$ is inspected. In other words, two different members of Pareto front have to differ at least by unit in the value of f_2. These features justify the proposition that an unknown non-dominated solution $\mathbf{y}$ satisfying $f_2(\mathbf{y}(U)) \geq f_2(\mathbf{y}) > u$ and $F_1(U) \leq f_1(\mathbf{y}) < F_1(u)$ can be determined by a finite bisection process performed in the interval $(u, U]$. It must be underlined that the half-open

interval $(u, U]$ denotes here only set of integers of the interval and, furthermore, u and U are also integers.

The suggested bisection process tries to determine such non-dominated solution $\boldsymbol{y}$, for which the difference $f_2(\boldsymbol{y}(U))$ - $f_2(\boldsymbol{y})$ is minimal. To perform the basic trial, if $u < v$ the value of v is determined by $v = \lfloor (U + u) / 2 \rfloor$ and values $F_1(v)$ and $f_2(\boldsymbol{y}(v))$ are computed, otherwise $U = u+ 1$ and the bisection process terminates.

Then, the following propositions can be made:

If $F_1(U) = F_1(v)$, then there is no non-dominated solution $\boldsymbol{y}$ with $f_2(\boldsymbol{y}) \in (v, U]$ and thus the interval can be reduced to $(u, v]$. In this case $F_1(v) < F_1(u)$ holds.

If $F_1(U) < F_1(v)$, then the interval $(u, U]$ can be reduced to $(v, U]$.

As each determination of $F_1(v)$ asks for considerable amount of computational time necessary for solving (6)-(13), we suggested the following process of Pareto front determination, which is based on the bisection, but which exploits the information of all performed computations of the function $F_1(v)$. The suggested process produces step-by-step the series of non-dominated solutions starting from the solution with the lowest value of f_1 and the highest value of f_2. The starting value of U enters the process as an upper bound of f_2 of all non-dominated solutions.

The process processes a stack L, which is operated as a first-in-last-out list (FILO). The k-th record of the list consists of three elements $L_1(k)$, $L_2(k)$, $L_3(k)$, where $L_3(k)$ is a solution $\boldsymbol{y}$, $L_1(k)$ is the value $f_1(\boldsymbol{y})$ and $L_2(k)$ is the value $f_2(\boldsymbol{y})$. The subscript noL points at the top of the stack, i.e., it points at the most recently saved record. The stack L contains the solutions, which are candidates for being a non-dominated solution. The searching process performs according to the following steps.

0. {Initialization}
   Initialize U by the input value and compute $F_1(U)$, $\boldsymbol{y}(U)$ and $f_2(\boldsymbol{y}(U))$ according to (6)-(13). Initialize the stack L by noL $= 1$, $L_1(\text{noL}) = f_1(\boldsymbol{y})$, $L_2(\text{noL}) = f_2(\boldsymbol{y})$ and $L_3(\text{noL}) = \boldsymbol{y}$, where $\boldsymbol{y}$ is the input non-dominated solution with minimal value of f_2. Initialize u by the value of $L_2(\text{noL})$. Comment: It is assumed that $F_1(U) < F_1(u)$. Otherwise, the only non-dominated solution of the problem is $L_3(1)$.
1. {Bisection}
   Determine $v = \lfloor (U + u) / 2 \rfloor$ and compute $F_1(v)$, $\boldsymbol{y}(v)$ and $f_2(\boldsymbol{y}(v))$ solving the problem (6)-(13).
2. {Decision on reduction of interval $(u, U]$ to $(u, v]$ due to no non-dominated solution between U and v}
   If $F_1(U) = F_1(v)$, then update $U = v$, $F_1(U) = F_1(v)$, $f_2(\boldsymbol{y}(U)) = f_2(\boldsymbol{y}(v))$, $\boldsymbol{y}(U) = \boldsymbol{y}(v)$ and go to 1. Otherwise go to 3.
3. {Decision on $\boldsymbol{y}(U)$ being a non-dominated solution}
   If $U > v+ 1$, then if $F_1(U) < F_1(v)$, then update noL = noL + 1 and insert the triple $F_1(v), f_2(\boldsymbol{y}(v)), \boldsymbol{y}(v)$ into L as $L_1(\text{noL})$, $L_2(\text{noL})$ and $L_3(\text{noL})$ respectively. Update $u = v$, $F_1(u) = F_1(v)$, $f_2(\boldsymbol{y}(u)) = f_2(\boldsymbol{y}(v))$, $\boldsymbol{y}(u) = \boldsymbol{y}(v)$ and go to 1.
   Otherwise $\{U = v+ 1$, i.e., $\boldsymbol{y}(U)$ is a non-dominated solution$\}$ insert the solution $\boldsymbol{y}(U)$ into the set of non-dominated solutions and go to 4.
4. {Decision on reduction of interval $(u, U]$ to $(u, v]$ due to the only non-dominated solution $\boldsymbol{y}(U)$ has been revealed.}
   If $F_1(v) < F_1(u)$ then update $U = v$, $F_1(U) = F_1(v)$, $f_2(\boldsymbol{y}(U)) = f_2(\boldsymbol{y}(v))$, $\boldsymbol{y}(U) = \boldsymbol{y}(v)$ and go to 1. Otherwise go to 5.
5. {Decision on searching process termination.}
   If noL > 1, then update $U = L_2(\text{noL})$, $F_1(U) = L_1(\text{noL})$, $\boldsymbol{y}(U) = L_3(\text{noL})$ and noL = noL $- 1$. Update $u = L_2(\text{noL})$, $F_1(u) = L_1(\text{noL})$, $\boldsymbol{y}(u) = L_3(\text{noL})$ and go to 1.
   Otherwise insert solution $L_3(1)$ into the set of non-dominated solutions and terminate.

The basic operation of the above algorithm is represented by solving of integer programming problem (6) – (13) by the branch-and-bound method. Even if the p-location problem is the polynomial one for fixed number of located centers, its enormous complexity is $O(m^p)$, what approves using a common IP-solver equipped with the branch-and-bound method, which does not ensure the polynomial complexity, but, thanks to the mentioned radial approach, it reaches optimal solution in acceptable computational time, when applied once. The situation considerably changes, when the basic operation is embedded in the presented bi-section algorithm determined for Pareto front constituting.

If one performance of the basic operation is considered as a unit, then the above algorithm finds the most right member of the Pareto front with complexity $log_2(N)$, where N is the length of the interval $(u, U]$ and only integers are taken into account. The number of necessary basic operations for detecting the whole Pareto front depends on unknown number of the Pareto front members. It can be easily derived that if the number of Pareto front members is k and $k = 2^a$, then $k(1 + log_2 I(\text{N}/k)) - 1$ basic operations is enough to determine the whole Pareto front in the worst case. Nevertheless, if $k \approx N$, then N operations are necessary.

IV. HYBRIDIZED GENETIC ALGORITHM FOR DETERMINATION OF A SET OF NON-DOMINATED SOLUTIONS

A. GENETIC ALGORITHM FOR BI-CRITERIA P-LOCATION PROBLEM

Genetic algorithm (GA) is a metaheuristic inspired by Darwinian evolution theory. It works with populations of individuals, where each individual usually corresponds to a feasible solution of the solved problem. In the course of the algorithm execution, these populations are periodically exchanged. Candidates for the entry into the new population are created by performing the operations of crossover and mutation with the individuals of the previous population. The new population is then built from the candidates by operation called selection. Due to the specific properties of the solved problem, we designed our own implementation

of GA, which allowed us to significantly speed up the routines used.

The set of non-dominated solutions is maintained in the form of an elite set. This set is updated only when a new non-dominated solution is found or if an element of the set becomes dominated by a newly found solution, otherwise it passes through the populations unchanged.

In the implemented algorithm we use several subroutines. First of them is the basic operation of the genetic algorithm – crossover. Input into crossover are two individuals called parents and output are also two individuals called offspring. Each individual is represented by a list of located centers. Operation of the subroutine is expressed in the following scheme.

Crossover(y_1, y_2):

0. {Initialization}
   Initialize o_1, o_2 as empty lists.
1. {Common locations}
   Create a list c of center locations that are present in both parents y_1, y_2.
   Create another list d of center locations that are present in one parent, but not in the other.
2. {Offspring creation}
   Insert all the center locations from c into both o_1, and o_2.
   Randomly shuffle locations in d. Divide d into two equal halves d_1 and d_2. Insert locations from d_1 into o_1 and locations from d_2 into o_2.
3. {Output}
   Return offspring o_1, o_2.

Second basic subroutine of the genetic algorithm is the operation mutation. Input into mutation is a single individual m and a hyper parameter noM giving the number of exchanges performed during the mutation. Output is the modified individual. This subroutine is expressed by the following steps.

Mutation(y, noM):

0. {Initialization}
   Create a list c of all the centers not located in y.
1. {Mutation}
   Repeat noM times:
   Randomly choose index i in the list y and index j in the list c.
   Exchange center locations at $y[i]$ and $c[j]$.
2. {Output}
   Return modified individual y.

Using these two subroutines, we can express the schema of the implemented genetic algorithm by the following steps.

0. {Initialization}
   Initialize starting population Pop. Compute the objective functions f_1 and f_2 for each Pop member and find the non-dominated solutions. Place these solutions into the elite set El. Set the starting population Pop as the current population cPop.
1. {Termination}
   If the termination rule is met, terminate. Otherwise continue with step 2.
2. {Creation of candidates}
   Create required number of candidates by repeating the following steps: Select two individuals m_1, m_2 from the cPop. Perform the Crossover(m_1, m_2), producing offspring o_1, o_2. If the condition for mutation is met, then apply o_1: = Mutation(o_1), o_2: = Mutation(o_2). Compute the objective functions f_1 and f_2 of the offspring o_1, o_2 and place them into the set of candidates Cas.
3. {Elite set update}
   Update the elite set El with non-dominated solutions from the set of candidates Cas.
4. {New population creation}
   Create the new population Pop by inserting the whole elite set El and filling the remaining gaps with individuals from the set of candidates Cas, until the required population size is reached. Set the new population Pop as the current population cPop and go to step 1.

In the step 0, the starting population is initialized by randomly generated feasible solutions until the required size is achieved. Random generation of a feasible solution involves random selection of p locations from m possible center locations.

The termination rule, in the step 1, is based on the time elapsed during the execution of the algorithm. When the specified time limit is reached, the current elite set is returned as the output of the algorithm.

Both subroutines Crossover and Mutation have, in our implementation, computational complexity of $O(m)$, where m is the number of possible center locations. Subroutine computing the values of objective functions f_1 and f_2 has computational complexity of $O(mpr)$, where p is the number of deployed centers and r is a parameter of generalized disutility objective function, in our experiments it has the value of 3. However, since the GA as a whole has a time-based termination rule, computation complexity of the GA would not pose a meaningful information.

There are two points in the algorithm, where individuals from the population (members to perform crossover on) or from the set of candidates (when filling in gaps in the new population) are selected. This selection is based on the fitness of the individuals. Fitness represents the quality of solutions and is therefore dependent on the values of objective functions f_1 and f_2. Fitness of a feasible solution $\boldsymbol{x}$ is given by (14). The parameter α allows us to manage the selection process towards prioritizing one or the other objective function. Values of the objective functions in (14) are scaled to the interval [0, 1].

$$g(\alpha, \boldsymbol{x}) = \alpha * f_1(\boldsymbol{x}) + (1 - \alpha) * f_2(\boldsymbol{x}) \tag{14}$$

We suggested and tested three strategies for setting the value of coefficient α.

The first strategy consists in setting α at the beginning of the algorithm and keeping it constant during the whole execution.

The second strategy divides the execution time into three parts. In the first part, α is set to the value of 1, and therefore the algorithm considers only the objective function f_1 in the operation of selection. In the second part, α is set to the value of 0, and therefore the fitness value depends only on objective function f_2. Finally, in the third part, α is set to the value of 0.5, which causes that the selection considers both objective functions equally.

The third strategy is similar to the previous one and it differs only in the third part of the execution. In this case, the value of α is modified dynamically, based on the current elite set. These adaptive changes enable to manage the search in that direction, in which the greatest potential of finding a new non-dominated solution is expected. Determination of the direction is based on the assumption that the elite set is a subset of Pareto set. Under this hypothesis, it can be seen in Fig. 3 that the only space, where missing Pareto set solutions could be situated, corresponds to the union of rectangles formed by adjacent elite set solutions. For each rectangle, its top left corner is given by one elite set solution and its bottom right corner by the following elite set solution. Then it can be expected that the greatest potential to find a missing Pareto set solution of the Pareto front is in the part, where the area of these rectangles is the largest.

B. HYBRIDIZATION OF GENETIC ALGORITHM

Genetic algorithms are usually good in performing global search and finding promising regions of the solution space. But they may be slow in converging to the best solution in these regions. The two processes are called the exploration and the exploitation. It is possible to hybridize a genetic algorithm with the use of simple heuristics, which can accelerate the exploitation process [15]. In this setting, such arbitrary heuristic is called ''meme''. While the processes of a genetic algorithm are inspired by the biological evolution, memes are more similar to the cultural evolution [21].

The memes employed in our implementation of GA are exchange heuristics based on neighborhood search. The neighborhood of a current solution consists of all the solutions which differ from the current solution in exactly one service center location. We used two exchange heuristics, namely the *best admissible* and the *first admissible*.

The *best admissible* heuristic searches the whole neighborhood of the current solution and takes the neighbor with the best value of fitness. If fitness of this neighbor is better than the fitness of the current solution, then the neighbor is declared as the current solution and the process repeats.

The *first admissible* heuristic also searches the neighborhood of the current solution. The difference from the *best admissible* heuristic is such that if it encounters a neighbor with better fitness value than the current solution, then this neighbor is immediately declared as the current solution and the search is restarted with the new current solution.

In both heuristics, we limit the maximal number of the neighborhood search repetitions by a hyperparameter. That enables us to limit the amount of computational resources used by the memes. The memes are applied to solutions of the elite set and the probability that a meme will be applied to a solution is given by another hyperparameter.

C. TOOLS OF TUNING

Performance of a GA or other evolutionary metaheuristics in general depends on the choice of hyperparameter values. When the hyperparameter value is to be set, interactions among other hyperparameters must be considered. A simple but expensive approach to hyperparameter tuning is an exhaustive grid search. The grid search means that each combination of relevant values of different hyperparameters is tested.

First a set is formed from combinations of relevant hyperparameter values and then, the algorithm is executed with each of these combinations on some representative test problem. Preliminary experience with tuning similar algorithms has provided us with some reasonable limits. Metaheuristics are generally stochastic algorithms; therefore, it is a common practice to perform several runs with each configuration of hyperparameters and to take the mean of a chosen *performance quality criterion*. If combinations of values for hyperparameters are chosen carefully, grid search is able to provide a good hyperparameter configuration for the test problem. However, it is possible that the determined hyperparameter configuration will not perform well for different problems. Substantial disadvantage of the grid search is its enormous time complexity, which is not practical for case of an algorithm with many hyperparameters.

Much simpler approach to hyperparameter tuning also begins with choosing a set of reasonable values for each hyperparameter. Then, an order of hyperparameters and a starting combination of hyperparameter values are chosen. The process of hyperparameter configurations testing is executed step by step, where each step corresponds with one of the hyperparameters in the chosen order. For each hyperparameter, iteration over the set of relevant values of the processed hyperparameter is done so that configuration with each of these values is tested. Values of the other hyperparameters are kept constant during this process. Value, for which the best average performance quality criterion is attained, is incorporated into the configuration and processing of the next hyperparameter follows. The whole process can be restarted again with the obtained hyperparameter configuration used as a starting configuration, to ensure its stability. We call this hyperparameter tuning approach ''one-by-one''. Authors in the paper [2] tested both mentioned approaches to the task of tuning hyperparameters of a two-objective GA. Their results show that in most cases there is no statistically significant difference between the solutions obtained by these approaches.

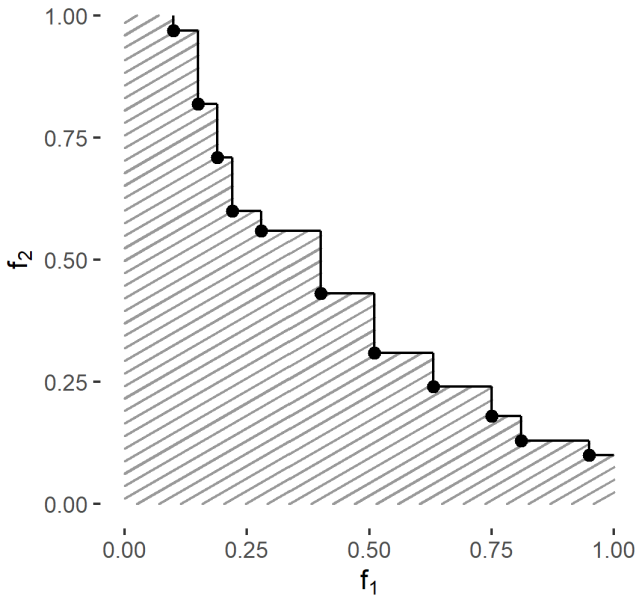

FIGURE 3. Quality of the non-dominated set is given by the hatched area under the black line connecting the non-dominated solutions.

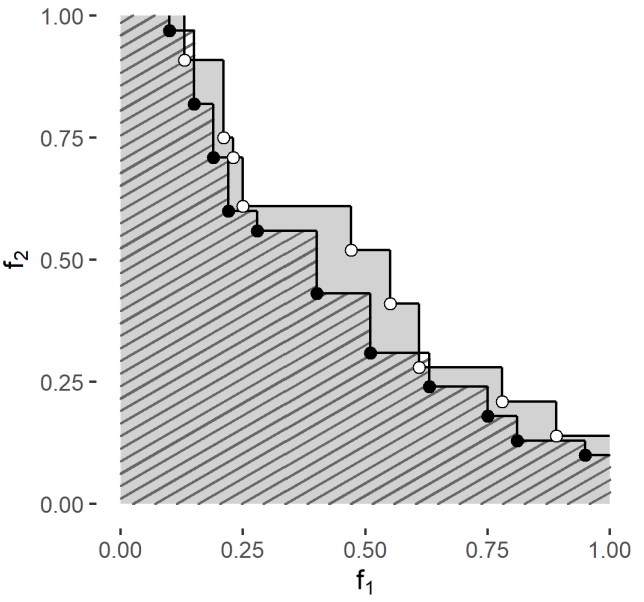

FIGURE 4. Comparison of two sets of non-dominated solutions. Set displayed by solid black points has smaller area under the line, therefore it is considered better.

Based on the results of the study [2], we decided to use for our GA the one-by-one tuning approach. This is mainly because of a large number of hyperparameters such as population size, number of candidates created in population exchanges, mutation probability, number of changes in a single mutation, strategy for setting the α parameter from equation (14), type of used meme and others. Practical execution of the hyperparameter tuning process is described in section V-B.

During the process of hyperparameter tuning, we assess the quality of a hyperparameter configuration by a *performance quality criterion*. This criterion incorporates both the speed of convergence during the algorithm execution and the quality of obtained non-dominated set.

Measure of quality of non-dominated set is displayed in Fig. 3. It is given by the area under the line connecting the non-dominated solutions plotted as points in two-dimensional space with coordinates given by the values of objective functions. Both objective functions of these solutions are scaled and shifted for their values to lie in the interval [0, 1]. Since we minimize both objective functions, it naturally follows that smaller area represents better solution. We refer to this measure as an *area criterion*. Comparison of two sets of non-dominated solutions is demonstrated in Fig. 4.

Measure of algorithm convergence speed can also be visualized geometrically. Comparison of two algorithm executions is displayed in the Fig. 5. Time elapsed since the beginning of the execution of the algorithm is displayed on the horizontal axis. On the vertical axis is the

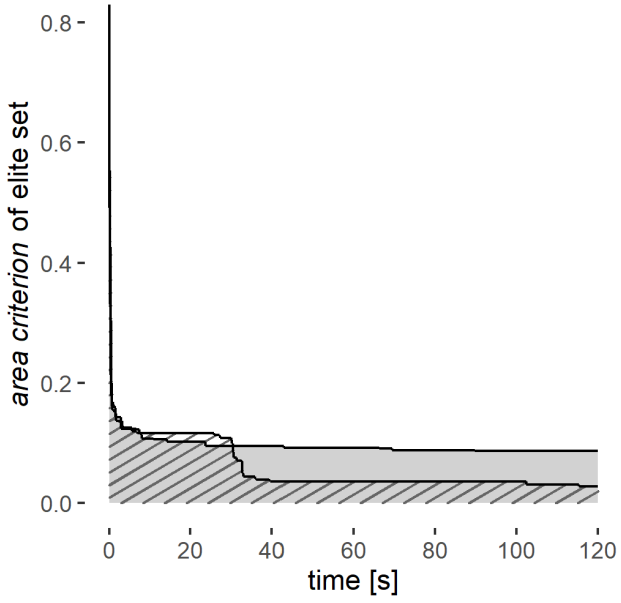

FIGURE 5. Comparison of two executions of the algorithm by the means of performance quality criterion. Criterion is given by the area under the plot. The execution corresponding to the hatched area is considered better because it has a lower value of the criterion.

above-mentioned *area criterion* of the elite set achieved in that particular time. *Performance quality criterion* is then given by the area under the plot. Algorithm execution corresponding to the hatched area in Fig. 5 is considered to be better because its area is smaller.

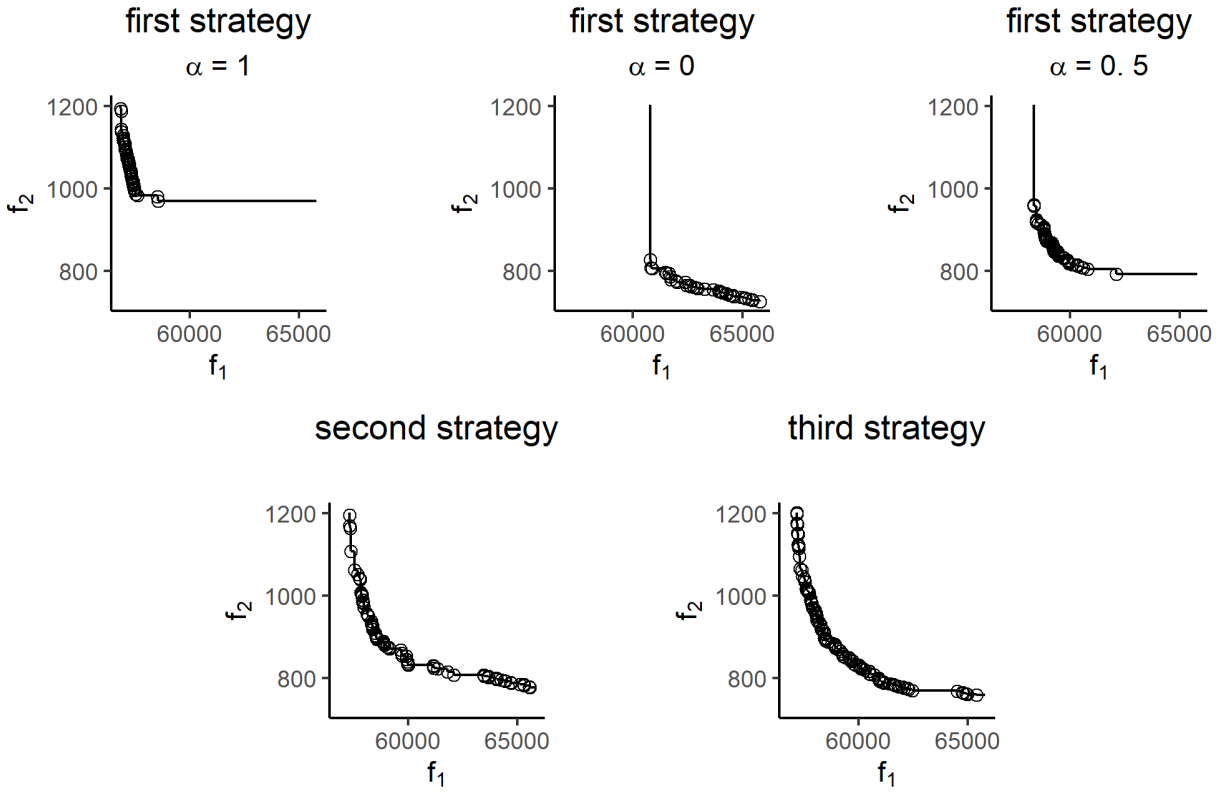

FIGURE 6. Resulting non-dominated sets when employing different strategies for modification of parameter α. The top line shows from left to right the results of the first strategy for α values fixed at 1, 0, 0,5 respectively. The second row displays results of using the second and third strategy, respectively.

V. COMPUTATIONAL STUDY

A. BENCHMARKS

This section is devoted to the computational study aimed at studying performance of the suggested genetic algorithm and the exact bisection process from the viewpoint of computational time demand and the solution accuracy.

The real instances of p-location problems were obtained from the road network of Slovak Republic and the emergency health care systems organized in particular self-governing regions. The individual instances are named by the names of capital cities of the regions. The list of the eight instances consists of Bratislava (BA, 87, 14), Banská Bystrica (BB, 515, 36), Košice (KE, 460, 32), Nitra (NR, 350, 27), Prešov (PO, 664, 32), Trenčín (TN, 276, 21), Trnava (TT, 249, 18) and Žilina (ZA, 315, 29). The triples attached to each instance name contains the following instance characteristics: The first member is the further used abbreviation of the instance name. The second member gives the number m of all possible center locations and the last member stands for the number p of deployed centers.

All cities and villages denoted as communities were taken into account. The coefficient b_j corresponds to the number of inhabitants of a community j given in hundreds.

The set of possible service center locations and the set of system users' locations are identical and correspond to the set of communities.

Here, the problem with objective function (2) was solved for $r = 3$. As discussed in [22], three nearest service centers are enough to model real emergency medical service system with satisfactory solution accuracy. The associated coefficients q_k for $k = 1, \ldots, r$ were set in percentage as follows: $q_1 = 77.063\%$, $q_2 = 16.476\%$ and $q_3 = 100\% - q_1 - q_2$. These values were obtained from a simulation model of existing system in Slovakia as described in [22]. The threshold D was set up at the value of 10.

An individual experiment was organized so that the exact solution of the underlying p-location problem was computed using the radial approach described in [10] and [11]. To obtain the exact solution of the problem, the optimization software FICO Xpress 7.3 was used, and the experiments were run on a PC equipped with the Intel® Core™ i7 5500U processor with clock speed 2.4 GHz and with 16 GB RAM.

The genetic algorithm including the above-described hybridization was programmed in programming language JAVA in NetBeans IDE 7.3 and the associated experiments were run on a PC equipped with the Intel® Core™ i7-860 processor with clock speed 2.8 GHz and with 8 GB RAM installed.

RAM usage on both computers remained below 8 GB during the algorithm execution, so the main difference was in processor clock speed. The computer on which GA was executed has approximately 17% higher clock speed.

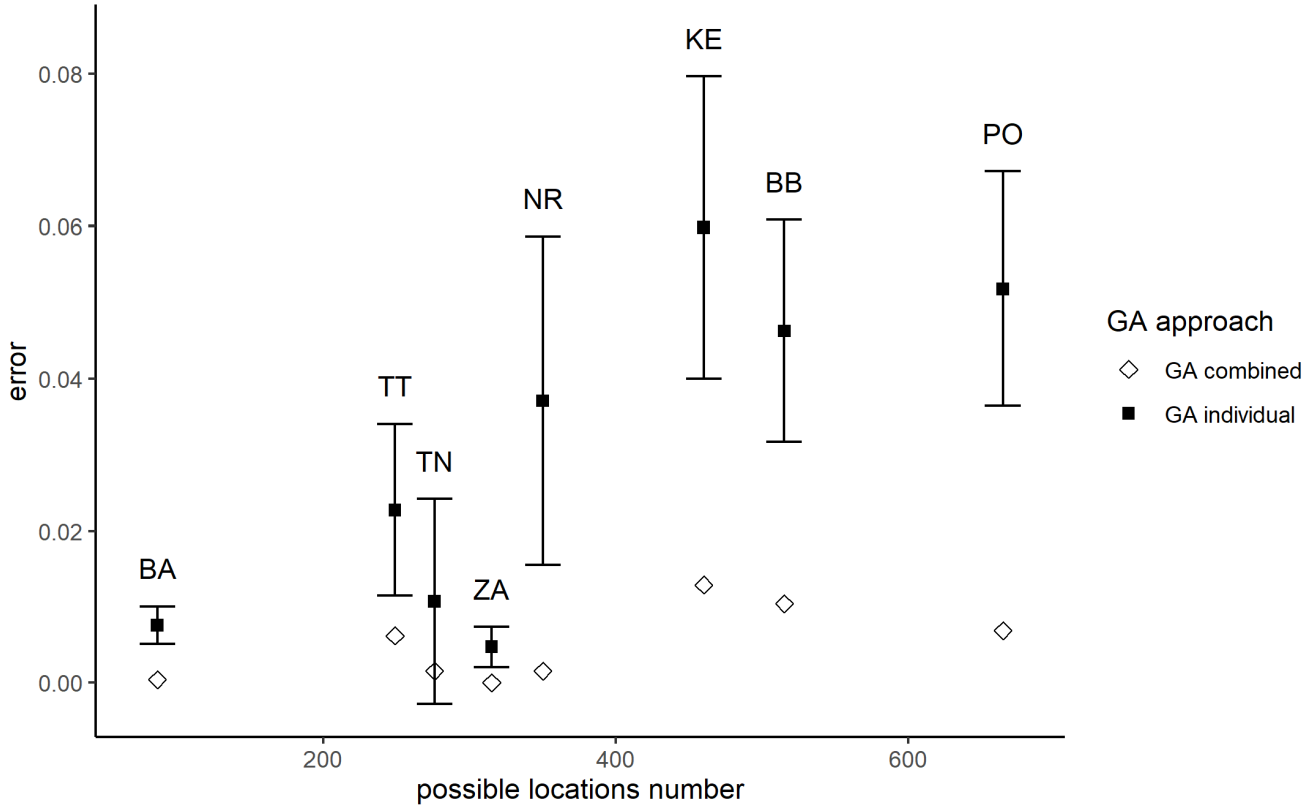

FIGURE 7. Relation of the error in area criterion of proposed GA approaches to the size of the problem instance. Size of the problem instance is represented by the number of possible center locations. In the case of GA individual, the whiskers display single standard deviation.

TABLE 1. Results of the exact approach.

Test instance	Pareto front size	Computational time [s]
BA	34	141
BB	229	33560
KE	262	12227
NR	106	5497
PO	271	47566
TN	98	1331
TT	64	1831
ZA	97	2127

Results of the exact approach characterized by the size of Pareto front and the associated computational time.

Differences in computation time between GA and exact approach are large enough, so that this computational difference does not influence any conclusions made in the following sections.

The bisection algorithm described in Section 3 was implemented in the language "mosel" of the solver Xpress using its exact optimization procedures to obtain solution of the problem (6)-(13) for a given value of B. The Pareto front of solutions for objectives (2) and (3) was obtained for each instance (self-governing region) mentioned above. The number of Pareto front members and computational time necessary for the associated bisection process for each instance is referred in Table 1.

B. HEURISTICS APPROACH

The first series of experiments was focused on tuning the parameters of the implemented GA. We used the procedure explained in section 4.3, with relevant values for each parameter determined by several initial experiments or from previous experiments with similar algorithms. Visually interesting were the effects of different strategies for modifying the α coefficient from the equation of the fitness criterion (14) in section IV-A. on the shape of the resulting non-dominated set.

Non-dominated sets obtained by the different strategies on the problem instance Prešov (PO, 664, 32) are displayed in Fig. 6. For the first strategy, we can observe set members clustered in that region of objective function space which the current value of α in the fitness criterion (14) favors most. In the case of $\alpha = 1$ objective function f_1 is minimized which drives the solutions into the upper left part of the objective function space. In the case of $\alpha = 0$ we can observe similar effect for objective function f_2. For $\alpha = 0.5$ both objective functions are considered equally which produces a cluster in the central part of the objective function space. Second strategy, which during its execution once employs all three above-mentioned values for the α coefficient produces set members more evenly spread across the whole spectrum of objective functions values approximating Pareto front. Even better spread reflected in the superior value of the used

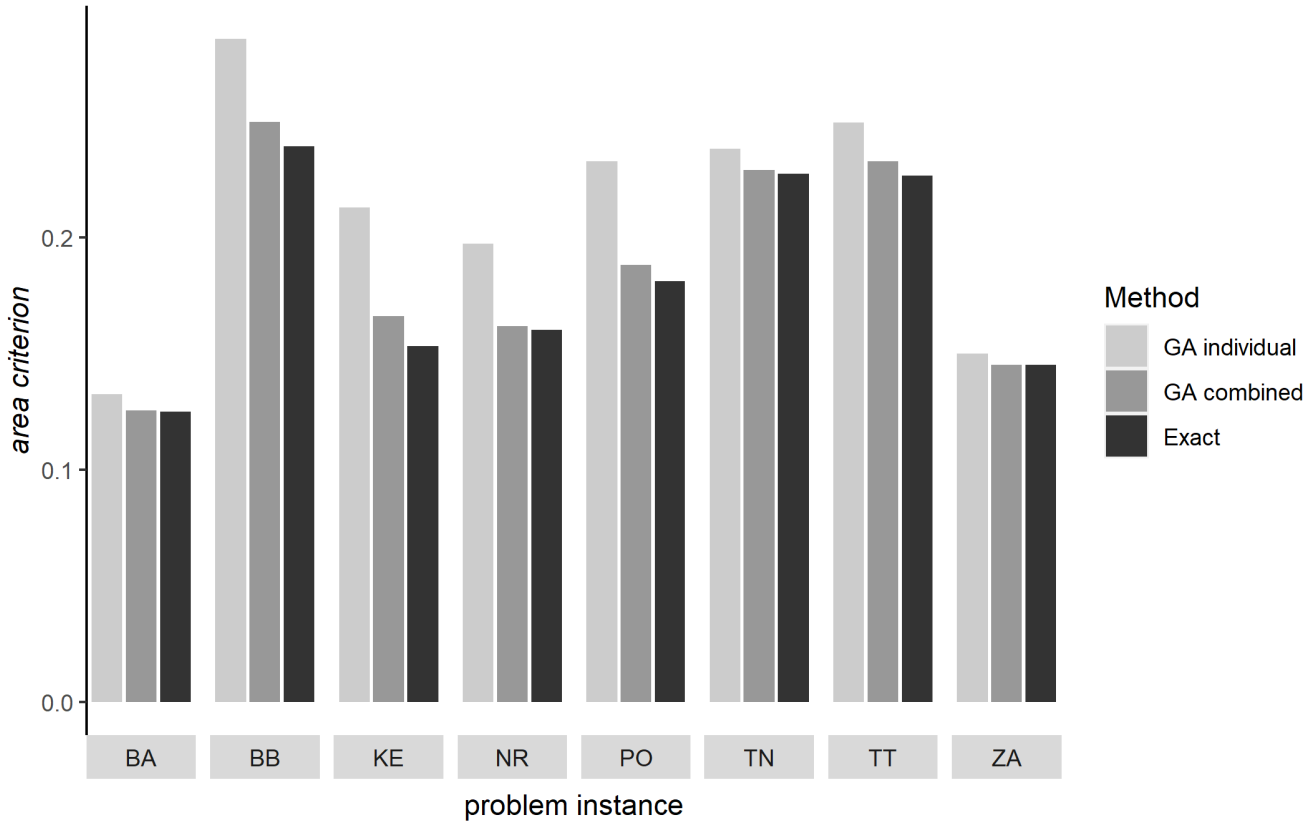

FIGURE 8. Comparison of the area criterion for the results of exact algorithm, combination of 50 GA runs and the average over 50 GA runs.

performance quality criterion achieves third strategy, which changes the value of α adaptively.

After determining good values for all parameters of the basic algorithm, we turned to testing the hybridized version. We tuned the parameters of the memes in a similar fashion that we used for the basic algorithm. The *first admissible* heuristic proved to provide better results than the *best admissible*. Having tuned the hybridized algorithm, we focused on examination of hybridization effects on the algorithm performance. We tested both the basic and hybridized versions of the algorithm and statistically evaluated the results. The hybridized version achieved statistically significantly better performance than the basic version in five out of eight problem instances. The difference between both versions was not statistically significant in the remaining three instances.

C. COMPARISON

To evaluate the results of the hybridized genetic algorithm introduced in section 4, we used Pareto front obtained by the exact algorithm described in section 3. Each run of the hybridized genetic algorithm was restricted by the time limit of two minutes and each instance solution was repeated fifty times with fixed parameter values. As the genetic algorithm operations are based on random trials, each run of the algorithm produces different results, i.e., different sets of the non-dominated solutions. These results can be observed individually, or the sets of non-dominated solutions can be merged, and the resulting combined set of the non-dominated solutions can be considered as the algorithm output. In the following text, we denote the results concerning the individual GA executions as ''GA individual'' and the results concerning the combined outputs of the fifty individual GA executions as ''GA combined''.

Comparison criterion used in the following experiments is the *area criterion* as described in section IV-C. Using this criterion, we examined accuracy and precision of the proposed genetic algorithm. We also observed the number of Pareto set members for each problem instance as well as

TABLE 2. Area criterion comparison .

Test instance	Pareto front	GA combined	GA average
BA	0.125	0.1254	0.1326
BB	0.2393	0.2497	0.2855
KE	0.1533	0.1661	0.2131
NR	0.1602	0.1618	0.1973
PO	0.1812	0.1881	0.233
TN	0.2276	0.2291	0.2383
TT	0.2267	0.2329	0.2495
ZA	0.1453	0.1453	0.1500

Comparison of *area criterion* for the results of exact algorithm, combination of fifty GA runs and the average over fifty GA runs.

TABLE 3. Nondominated solutions comparison.

	BA	BB	KE	NR	PO	TN	TT	ZA
Pareto set size	34	229	262	106	271	98	64	97
GA comb. Pareto optimal count	30	114	121	86	128	88	58	96
GA comb. elite set size	31	173	200	99	190	95	65	96
GA avg. Pareto optimal count	19.74	8.66	8.62	15.42	9.56	47.14	44.58	56.72
GA avg. elite set size	26.42	96.12	98.82	82.28	82.40	82.06	54.76	77.80

Comparison of number of solutions and of Pareto optimal solutions obtained by different methods.

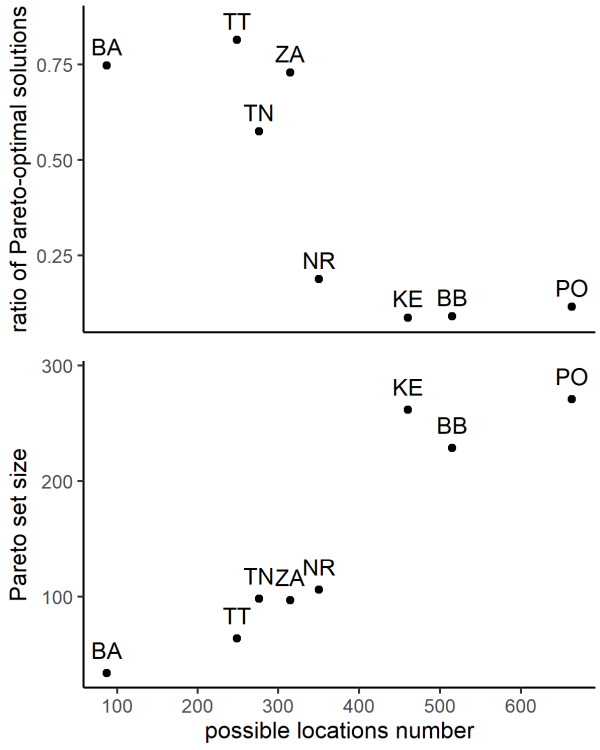

FIGURE 9. In the bottom plot is displayed the relation between the size of Pareto set and the number of possible center locations. In the top plot is displayed the relation between the average ratio of Pareto- optimal solutions among the elite sets produced by GA individual and the number of possible center locations.

the number of Pareto set solutions, which were found by the metaheuristic.

A member of a set of non-dominated solutions obtained by the genetic algorithm is referred to as Pareto-optimal, if it is a member of the Pareto set obtained by the exact approach.

In all comparisons with individual GA executions, the value of respective criterion is averaged over the fifty executions.

We evaluated the performance of both the fifty individual GA execution results and the combination of these results by the means of computing the difference in *area criterion* between solutions obtained by these GA approaches and the exact solution. We denote this statistic as the *error,* and it is a representation of the inaccuracy of the proposed GA. For GA individual we used the average of *area criterion* over the fifty obtained solutions. To examine the imprecision of the GA individual approach, we computed the *standard deviation* (*sd*) in the *area criterion* over the fifty executions. Relation between these two closeness statistics and the number of possible center locations, in the model denoted as m, is displayed in the Fig. 7. As we can see, the *error* for both GA individual and GA combined has a raising trend for increasing number of possible center locations. Problem instance ZA stands out in this plot, it has the smallest *error* for both GA approaches, although it is a medium sized benchmark. Raising trend of the *error* is less prominent for the GA combined approach, this could suggest that this combining approach could be better suited for countering the increasing *error* of individual executions for larger problem instances. This hypothesis is further backed by the generally increasing trend of *standard deviation* for larger problem instances. As we can see in the Fig. 7, improvement in *error* in GA combined over GA individual has a good correspondence to *standard deviation* of individual solutions.

More detailed comparison of the *area criterion* is displayed in Fig. 8 and the associated data are plotted in Table 2.

Another comparison was carried out for the number of Pareto-optimal solutions contained in resulting elite sets produced by GA individual and GA combined. For GA individual we plotted the relation between average ratio of Pareto-optimal solutions among the resulting elite set and the number of possible center locations. This relation is displayed in the top plot of Fig. 9. We can observe a sharp decrease in the ratio of Pareto-optimal solutions in outputs for larger problem instances. When we consider the increasing size of Pareto sets for larger problem instances, displayed in the bottom plot of Fig. 9, we can conclude significantly increased difficulty of finding members of Pareto set for larger problems. This suggests that for larger problem instances it is not necessary to increase the size of the elite set and to rather experiment with longer execution times.

Complete results concerning Pareto-optimal solutions in outputs of GA are plotted in Fig. 10 and displayed in Table 3.

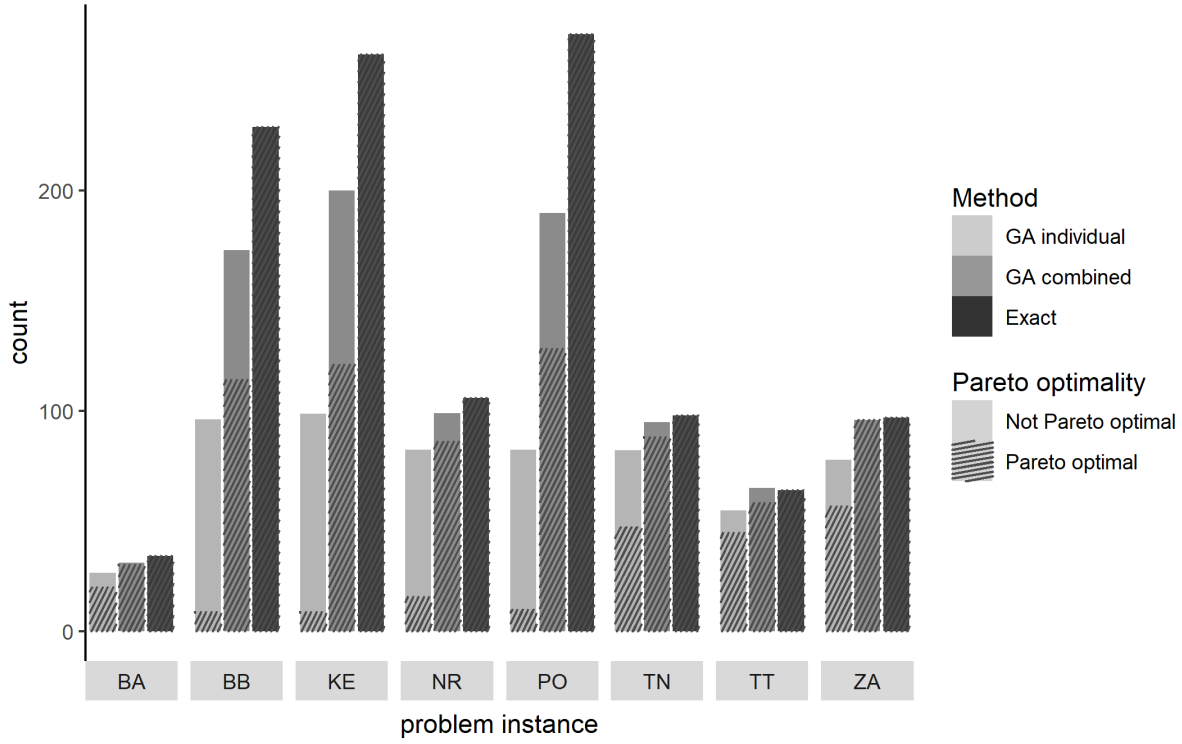

FIGURE 10. Comparison of number of solutions and of Pareto optimal solutions obtained by different methods.

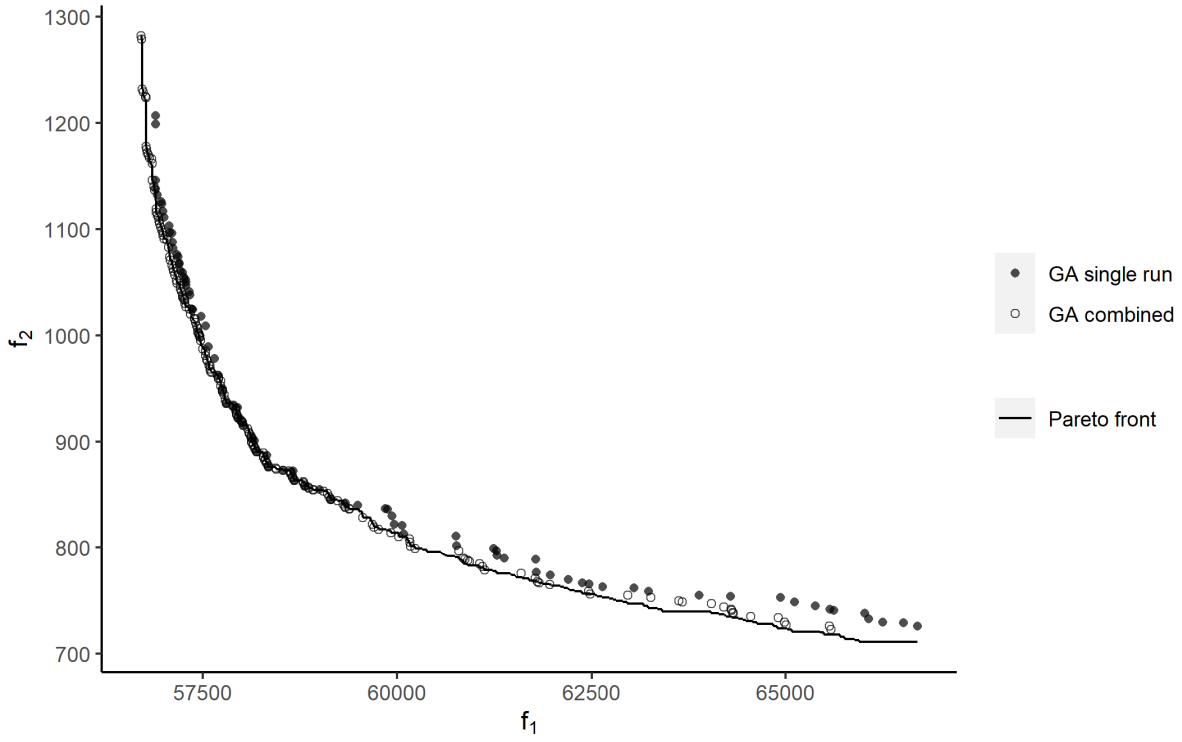

FIGURE 11. Comparison of a Pareto front with a combination of fifty GA runs and with a single GA run for the problem instance PO.

Comparing the test results obtained for the individual benchmarks, the combined set of non-dominated solutions of instance ZA contains all Pareto set solutions but one. On the other hand, the least portion of Pareto set created by merging the fifty results of genetic algorithm was obtained for the instance KE and contained only 46.18% of Pareto

set solutions. Notably, both these problem instances are among the middle sized by the number of possible center locations and among the larger by the number of deployed centers. The average portion of Pareto set discovered by an individual GA run was the greatest one for the test instances TT and ZA. The least average portion was obtained for the test instance KE.

Execution time for different algorithms was also compared. Single GA execution time was limited to 120 s, therefore execution times of **GA individual (120 s)** and **GA combined (6000 s)** are constant for all test instances. Execution times for exact method are reported in Table 1. As can be seen in the table, execution times of a single GA run are much shorter than the execution times of exact method in all tested instances of the problem.

It is worth to notice that fifty GA runs were several times faster than the exact method in solving the largest test instances such as KE, BB or PO.

To show that good results can be obtained by GA even in the largest test instance PO, we plotted a comparison of the Pareto front with the combination of fifty GA runs and with a single selected GA run. This comparison is shown in Fig. 11. The selected single GA run contains ten Pareto-optimal solutions and the size of the resulting elite set is 76. As can be seen in Fig. 11, even the single GA run provides a relatively good set of choices for the system design, which are spread over the whole spectrum of the Pareto optimal solutions. The combination of the fifty GA runs approximates the Pareto front even better.

VI. CONCLUSION

The presented research was devoted to the approaches, which can find either the Pareto front or its satisfactory approximation for the bi-criteria public service system design problem. An exact bi-section method was suggested to be able to produce the exact Pareto front for the problem, which includes generalized disutility as one of the considered criteria. The second criterion corresponds with one of the possible fairness criteria. The further studied approach was based on an evolutionary metaheuristic represented by a hybridized genetic algorithm. Within this paper, the hypothesis was verified that the hybridized genetic algorithm can approximate the Pareto front of solutions satisfactorily in acceptable computational time. Nevertheless, the computational study showed that only one run of the evolutionary process is not enough to obtain a substantial part of the Pareto front even if the genetic algorithm is carefully tuned and hybridized with improving meme. We proved that the key to success is repeating the suggested algorithm run and merging the obtained results.

The achieved results prove that the suggested heuristic approach can provide a good approximation of the Pareto front in the sense of the measure of quality of non-dominated sets. For practical reasons, the presented method can replace the exact approach, which is very time consuming and often fails, when large problem instances are solved.

However, we observed increasing gap between solutions obtained by GA and exact solutions for larger test instances. Tests we performed suggest that for solving larger problems, longer execution times of individual GA solutions may be required and it's possible that the GA will not be able to reach accuracy as high as it did for the tested problem instances. Memory requirements of the suggested GA approach are increasing linearly with rising number of deployed centers and with the number of possible center locations and therefore should not pose any limitation in practical settings.

REFERENCES

[1] J. E. C. Arroyo, P. M. dos Santos, M. S. Soares, and A. G. Santos, "A multi-objective genetic algorithm with path relinking for the p-median problem," in *Proc. 12th Ibero-Amer. Conf. Adv. Artif. Intell.*, 2010, pp. 70–79.

[2] D. Grygar and R. Fabricius, "An efficient adjustment of genetic algorithm for Pareto front determination," *Transp. Res. Proc.*, vol. 40, pp. 1335–1342, Jan. 2019, doi: 10.1016/j.trpro.2019.07.185.

[3] P. Avella, A. Sassano, and I. Vasil'ev, "Computational study of large-scale p-median problems," *Math. Program.*, vol. 109, no. 1, pp. 89–114, Jan. 2007, doi: 10.1007/s10107-005-0700-6.

[4] L. Brotcorne, G. Laporte, and F. Semet, "Ambulance location and relocation models," *Eur. J. Oper. Res.*, vol. 147, pp. 451–463, Apr. 2003, doi: 10.1016/S0377-2217(02)00364-8.

[5] K. F. Doerner, W. J. Gutjahr, R. F. Hartl, M. Karall, and M. Reimann, "Heuristic solution of an extended double-coverage ambulance location problem for Austria," *Central Eur. J. Oper. Res.*, vol. 13, no. 4, pp. 325–340, Dec. 2005.

[6] A. Ingolfsson, S. Budge, and E. Erkut, "Optimal ambulance location with random delays and travel times," *Health Care Manage. Sci.*, vol. 11, no. 3, pp. 262–274, Sep. 2008, doi: 10.1007/s10729-007-9048-1.

[7] L. Janosikova, "Emergency medical service planning," *Commun. Sci. Lett. Univ. Zilina*, vol. 9, no. 2, pp. 64–68, Mar. 2007. [Online]. Available: https://komunikacie.uniza.sk/index.php/communications/article/view/1133

[8] S. Elloumi, M. Labbé, and Y. Pochet, "A new formulation and resolution method for the p-center problem," *INFORMS J. Comput.*, vol. 16, no. 1, pp. 84–94, 2004, doi: 10.1287/ijoc.1030.0028.

[9] S. García, M. Labbé, and A. Marín, "Solving large p-median problems with a radius formulation," *INFORMS J. Comput.*, vol. 23, no. 4, pp. 546–556, Nov. 2011, doi: 10.1287/ijoc.1100.0418.

[10] J. Janáček, "Approximate covering models of location problems," in *Proc. Lect. Notes Manage. Sci., 1st Int. Conf. ICAOR*. Teheran, Iran: Tadbir Institute for Operational Research, System Design and Financial Services, Sep. 2008, pp. 53–61.

[11] M. Kvet, "Advanced radial approach to resource location problems," in *Developments and Advances in Intelligent Systems and Applications*. Cham, Switzerland: Springer, 2018, pp. 29–48, doi: 10.1007/978-3-319-58965-7.

[12] D. Sayah and S. Irnich, "A new compact formulation for the discrete p-dispersion problem," *Eur. J. Oper. Res.*, vol. 256, no. 1, pp. 62–67, Jan. 2017, doi: 10.1016/j.ejor.2016.06.036.

[13] T. Drezner and Z. Drezner, "The gravity p-median model," *Eur. J. Oper. Res.*, vol. 179, no. 3, pp. 1239–1251, Jun. 2007, doi: 10.1016/j.ejor.2005.04.054.

[14] M. Kvet, "Computational study of radial approach to public service system design with generalized utility," in *Proc. 10th Int. Conf. Digit. Technol.*, Jul. 2014, pp. 187–197, doi: 10.1109/DT.2014.6868713.

[15] G. Gopal, "Hybridization in genetic algorithms," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 3, pp. 403–409, Apr. 2013.

[16] C. Reeves, "Genetic algorithms," in *Handbook of Metaheuristics*, vol. 146. New York, NY, USA: Springer, 2010, pp. 109–139, doi: 10.1007/978-1-4419-1665-5.

[17] A. Rybickova, A. Burketova, and D. Mockova, "Solution to the location-routing problem using a genetic algorithm," in *Proc. Smart Cities Symp. Prague (SCSP)*, May 2016, pp. 1–6, doi: 10.1109/SCSP.2016.7501016.

[18] A. Rybickova, D. Mocková, and D. Teichmann, "Genetic algorithm for the continuous location-routing problem," *Neural Netw. World*, vol. 29, pp. 173–187, Apr. 2019, doi: 10.14311/NNW.2019.29.012.
[19] D. Bertsimas, V. F. Farias, and N. Trichakis, "The price of fairness," *Oper. Res.*, vol. 59, no. 1, pp. 17–31, Apr. 2011, doi: 10.1287/opre.1100.0865.
[20] W. Ogryczak and T. Śliwiński, "On direct methods for lexicographic min-max optimization," in *Proc. Comput. Sci. Appl. ICCSA*. Berlin, Germany: Springer, 2006, pp. 802–811, doi: 10.1007/11751595-85.
[21] P. Moscato, "On evolution, search, optimization, genetic algorithms and martial arts—Towards memetic algorithms," Caltech Concurrent Comput. Program, California Inst. Technol., Pasadena, CA, USA, C3P Rep. 826, 1989.
[22] P. Jankovic, "Calculating reduction coefficients for optimization of emergency service system using microscopic simulation model," in *Proc. IEEE 17th Int. Symp. Comput. Intell. Informat. (CINTI)*, Nov. 2016, pp. 163–168, doi: 10.1109/CINTI.2016.7846397.

JAROSLAV JANACEK was born in Ústí nad Labem, Czech Republic, in 1949. He received the M.S. degree in theoretical from Charles University, Prague, Czech Republic, in 1972, and the Ph.D. degree in technical cybernetics from the Slovak Academy of Sciences, Bratislava, Slovakia, in 1979.

From 1973 to 1984, he was an Assistant Professor with the Department of Cybernetics in Transport and Communications, Technical University of Transport and Communications, Žilina, Slovakia. From 1985 to 1998, he was an Associate Professor. He has been a Full Professor with the University of Žilina, since 1999. He is the author of 15 textbooks and monographs and has published more than 80 research articles. His research interest includes the field of discrete optimization, especially location problems.

RENE FABRICIUS was born in Brezno, Slovakia, in 1994. He received the B.S. and M.S. degrees in informatics from the University of Žilina, Žilina, Slovakia, in 2018 and 2020, respectively, where he is currently pursuing the Ph.D. degree in applied informatics.

Since 2021, he has been a Research Assistant with the Research Centre, University of Žilina. His research interests include image classification using ensemble models of neural networks and evolutionary metaheuristics.