Hyperheuristics for Determination of Non-dominated Set of Public Service System Designs

Marek Kvet, Jaroslav Janáček
University of Žilina
Žilina, Slovakia
{marek.kvet, jaroslav.janacek}@fri.uniza.sk

***Abstract*—A design of a public service system subjects to various objectives, which are usually in conflict. The most known pair of conflicting criteria is the system and fair criterion, where the system criterion expresses utility or disutility of an average system user and the criterion of fairness takes into account the access of the worst situated minority of the system users to service. A series of non-dominated system designs is important especially for the decision maker responsible form the final form of the system. In this contribution, we concentrated on study and construction of hyperheuristics assigned to the efficient determination of a non-dominated set of public service system designs, where the system and fair criteria are taken into account. The suggested hyperheuristic disposes with a list of subordinate heuristics with dynamically updated ranks depending on their previous success in improving quality of the non-dominated solution set. To explore properties of the employed subordinate heuristic, a series of numerical experiments with real-sized benchmarks has been performed and the obtained results are presented and discussed.**

I. Introduction

There are host of heuristic approaches developed to solve complex combinatorial problems in an acceptable time limit. The one of the family of the computational time demanding problems is determination of the Pareto-front of public service system designs with two conflicting objectives [11, 13]. Just a simple public service system design problem with one objective represents a hard combinatorial problem, exact solution of which asks for unpredictable computational time [3, 12, 21, 23]. The objective imposed on a public service system can be divided into two classes, when the first one comprises so called system objectives and the second one contains so called fair objectives. The system objective usually expresses average disutility perceived by the system users and its value is computed as a sum of individual disutility contributions. As the sum is to be minimized, the associated problems are called min-sum problems. Contrary to the system objectives, a fair objective evaluates perceived disutility of the worst situated minority of the system users. The associated problems are called min-max problems, due to maximal disutility perceived by the users of the worst situated minority is minimized. Criteria belonging in the two different classes are naturally in conflict, what means that if one of them is minimized, the other takes a bigger value.

The basic tool for Pareto-front determination consists of usage a minimization of one of the criteria subject to the conflicting one is restricted by an upper limit. This tool has been used in both exact and heuristic approaches [11, 13]. Both Pareto-front determination and Pareto-front approximation asks for a long series of problem solving, where the limit of the conflicting criterion is changed. The necessity of repeated optimization processes raises the question of which of the simple incremental heuristics brings the greatest benefit for improving the objective function. This decision can be avoided by usage of a hyperheuristic, which makes use of a set of simpler incrementing heuristics and tries to answer the question by testing the individual heuristics and evaluating their efficiency during the computation process.

This paper is devoted to the design of a hyperheuristic designed to build a good approximation of the Pareto front of public service system designs, where a design is evaluated according two conflicting criteria. The first of them is represented by a min-sum objective function, which value is proportional to the average response time of the system to its users. The second fair criterion counts the number of users' demands located outside a given radius from the nearest service center. The considered public service system provides its users with service from a given number of service centers and the service center deployment in a finite set of possible service center locations corresponds to a system design [1, 2, 3, 5, 7, 8, 9, 10, 17, 18, 19]. The simple heuristics embedded into the suggested hyperheuristic are based on neighborhood search, where the neighborhood of a current design is given by all designs, which differ from the current design in location of exactly one service center. These swapping heuristics operate under different constraints and exhibit different efficiencies when applied to a given default solution. Behavior of the suggested hyperheuristic is studied under series of benchmarks [11, 13, 14, 15, 22] in the concluding part of this paper.

II. Scheme of the Perturbation Selective Hyperheuristic

The classical perturbation selective hyperheuristic with learning disposes with a stack A of simpler heuristic and each heuristic – routine $a \in A$ has its own rank denoted $Rank(a)$. Let $\boldsymbol{y}$ be an input solution and $a(\boldsymbol{y})$ be the resulting solution of the routine applied to $\boldsymbol{y}$. Let $f(\boldsymbol{y})$ denote the objective function value of the solution $\boldsymbol{y}$. Let t^{max} be the maximal computational time permitted for the hyperheuristic and let $Proh$ be a list of temporary prohibited routines. Then the scheme of the minimization perturbation selective hyperheuristic with learning can be briefly described by the following steps.

0. Initialize the best found solution $\boldsymbol{y}^{best}$ by an input solution $\boldsymbol{y}$ and $f^{best} = f(\boldsymbol{y}^{best})$. Initialize rank $Rank(a) = 0$ for each $a \in A$ and empty the list $Proh = \emptyset$.

1. While CPU ≤ t^{max} repeat the following steps, otherwise terminate and return $\boldsymbol{y}^{best}$ and f^{best} as the result.

2. Choose the routine with the highest rank, which is not prohibited, i.e. $a \notin Proh$, Perform the chosen routine a on the current solution $\boldsymbol{y}$.

3. If $f(a(\boldsymbol{y})) < f^{best}$, then set $Rank(a) = Rank(a) + 1$, $\boldsymbol{y} = a(\boldsymbol{y})$, $\boldsymbol{y}^{best} = \boldsymbol{y}$, $f^{best} = f(\boldsymbol{y}^{best})$ and $Proh = \emptyset$. Otherwise set $Rank(a) = Rank(a) - 1$, $Proh = Proh \cup \{a\}$ and perform random trial of accepting the solution $a(\boldsymbol{y})$. If result of the trial is true, then set $\boldsymbol{y} = a(\boldsymbol{y})$ and $Proh = \emptyset$. Continue with step 1.

The above hyperheuristic includes a learning characteristic represented by the structure *Rank* and the way of its updating. Our experience so far with this hyperheuristic applied to public service design has shown that the used *Rank* management causes the routine that achieved success first to be stably selected in subsequent steps and to maintain its leading position until the end of the algorithm run. Furthermore, the below described process of improving Pareto-front approximation consists of series of simple heuristic applications to different starting solutions. That is why, our newly suggested hyperheuristic has to be embedded into the process as a subroutine and, in addition, it must enable transition of experience obtained during one run of the routine to the next runs.

III. PARETO-FRONT APPROXIMATION BY GRADUAL REFINEMENT

The public service system design problem is defined as a task to minimize an objective function f on the finite set Y of feasible solutions. Each element $\boldsymbol{y}$ of Y corresponds to the deployment of exactly p service centers in the set of m possible service center locations. The deployment $\boldsymbol{y}$ is represented by an m – dimensional vector, whose components can take the values one or zero. The value one at the i - th position means that a service center is located at at the i - th possible service center location. The zero value means that the associated position stays unoccupied. Obviously, any feasible solution $\boldsymbol{y}$ contains exactly p units.

In our paper, we consider two conflicting objective functions f_1 and f_2, where $f_1(\boldsymbol{y})$ gives the value proportional to the average response time of the system to its users. Each user $j \in \{1, \ldots, n\}$ is characterized by frequency b_j of his demand occurrence. Let t_{ij} denote traversing time from i – th possible service center location to position of j – th user. Furthermore, we consider that a temporary occupancy of the nearest service center may prevent the system from providing a current demand with service and then the demand must be satisfied from a more distant center. To comprise this situation in the objective function, we introduce probabilities $q_1, \ldots, q_r$ so that probability q_k is assigned to the case, when k – th nearest service center is the nearest one, which is available. Then (1) can define the objective function f_1.

$$f_1(\mathbf{y}) = \sum_{j=1}^{n} b_j \sum_{k=1}^{r} q_k . \min_k \left\{ t_{ij} : i = 1, \ldots, m,\ y_i = 1 \right\} \qquad (1)$$

Operation $\min_k\{v_1, \ldots, v_m\}$ returns the k-th minimal value of the set $\{v_1, \ldots, v_m\}$.

The fair objective function $f_2(\boldsymbol{y})$ is defined as the number of users' demands located outside the radius T from the nearest service center [4, 6]. This function can be described by (2).

$$f_2(\mathbf{y}) = \sum_{j=1}^{n} b_j . \max \left\{ \begin{array}{l} 0, \\ sign\left(\min\{ t_{ij} : i = 1, \ldots, m,\ y_i = 1 \} - T \right) \end{array} \right\} \qquad (2)$$

These objective functions are in conflict. If one of the functions is minimized, then the value of the other is increased. This property enables to introduce notion of dominancy for two feasible solutions $\boldsymbol{y}, \boldsymbol{x} \in Y$. We say that solution $\boldsymbol{y}$ dominates $\boldsymbol{x}$, if $f_1(\boldsymbol{y}) \le f_1(\boldsymbol{x})$ and $f_2(\boldsymbol{y}) \le f_2(\boldsymbol{x})$ and if $f_1(\boldsymbol{y}) < f_1(\boldsymbol{x})$ or $f_2(\boldsymbol{y}) < f_2(\boldsymbol{x})$. As a dominate solution can be excluded from the set of sensible designs, only solutions, which are non-dominated by any other solution of Y must be taken into account. The set of all non-dominated solutions is called Pareto-front. The explanation of Pareto front can be easily understood from visualization illustrated in Fig. 1.

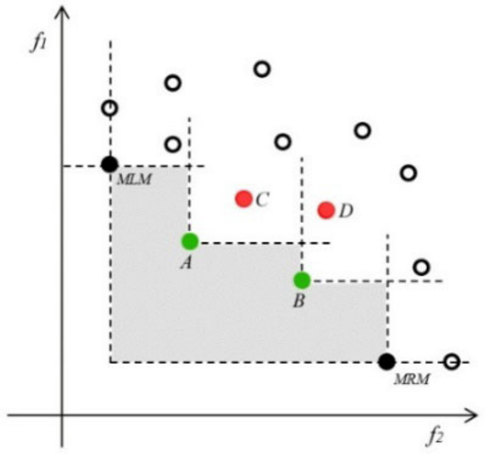

Fig. 1. Pareto front of non-dominated solutions

If we look at Fig. 1, we can see that the green solutions A and B are members of the Pareto front, because they are not dominated by any other solution. It means that none of the Pareto fronts members is equally good or better in both quality criteria. On the contrary, the red solutions C and D do not belong to the Pareto front. The particle A dominates solution C and the solution D is dominated by both particles A and B. For completeness, let us note that *MLM* and *MRM* represent the bordering members of the Pareto front. While *MLM* denotes the most left member, symbol *MRM* is used to denote the most right one. The bordering members can be obtained by optimizing only one of given criteria.

As mentioned in previous Sections, exact determination of Pareto-front is very computational time demanding task. That is why; we devote the remainder of the paper to finding a good approximation of the Pareto-front. The approximation will be denoted by *NDSS* and it will consist of *noNDSS* mutually non-dominated solutions ordered according to increasing values of f_2. Assuming that $\boldsymbol{y}^1$ and $\boldsymbol{y}^{noNDSS}$ are near to the first and last members of the Pareto-front, then proximity of an approximation *NDSS* to the Pareto-front can be measures by the difference of *NDSS-Area* and area of the Pareto-front. The *Area* can be computed according to (3).

$$Area = \sum_{i=1}^{noNDSS-1} (f_1(\mathbf{y}^i) - f_1(\mathbf{y}^{noNDSS})).(f_2(\mathbf{y}^{i+1}) - f_2(\mathbf{y}^i)) \quad (3)$$

The gradual refinement approach minimizing step-by-step the *NDSS-Area* starts from the initial *NDSS* formed by two bordering solutions $\boldsymbol{y}^1$ and $\boldsymbol{y}^{noNDSS}$ of the Pareto-front. The basic step of the refinement process proceeds the *k*-th member of the current *NDSS*. The basic process starts with initial solution $\boldsymbol{y}^k$ and inspects the neighborhood of the current solution. Each inspected solution is evaluated by functions f_1 and f_2 and used to update the current *NDSS*, what means that it is either refused, if it is dominated by a current member of *NDSS*, or it is inserted into *NDSS* and, then, the solutions, which are dominated by it are excluded. Regardless of the *NDSS* management, the basic process evaluates a surrogate objective function and if a better solution than the current one is found, it moves to this solution and declares it as a new current solution. Then the process continues with proceeding the neighborhood of the new current solution until a termination rule is met. As the *NDSS* is changing during the basic step, the solution of the *k*-th position of *NDSS* is compared to the starting $\boldsymbol{y}^k$ and, if they differs, the basic process is repeated with the updated $\boldsymbol{y}^k$. Otherwise, the refinement process continues with the solution $\boldsymbol{y}^{k+1}$. If $\boldsymbol{y}^{noNDSS-1}$ has been processed, the refinement process can be repeated from $\boldsymbol{y}^k$ to $\boldsymbol{y}^{k+1}$ of the resulting *NDSS*. The basic step, i.e. processing of $\boldsymbol{y}^k$, may be constructed in many ways. In this paper, we study variants belonging to the family of directed searches.

IV. Directed search routines

Directed search routine is a swapping algorithm with strategy the best admissible, which minimizes a surrogate objective function formulated as linear combination $w_1 f_1(\boldsymbol{y}) + w_2 f_2(\boldsymbol{y})$, where coefficients w_1 and w_2 determine the direction in the space $f_1 \times f_2$ in which the minimization is performed. If the neighborhood solutions are restricted, the restriction has the form of inequality $a_1 f_1(\boldsymbol{y}) + a_2 f_2(\boldsymbol{y}) \leq a_3$. In the suggested hyperheuristic, five directed search routines are used. Each routine has its run limited by computational time *aTime*. The list of directed search routines follows:

Routine1: {The simple radial search} The routine performs the swapping algorithm with initial solution $\boldsymbol{y}^k$ and with parameters $w_1 = f_2(\boldsymbol{y}^{k+1}) - f_2(\boldsymbol{y}^k)$, $w_2 = f_1(\boldsymbol{y}^k) - f_1(\boldsymbol{y}^{k+1})$, $a_1 = 0$, $a_2 = 0$ and $a_3 = 0$ (no restriction is applied).

Routine2: {The restricted simple radial search} The routine performs the swapping algorithm with initial solution $\boldsymbol{y}^k$ and with parameters $w_1 = f_2(\boldsymbol{y}^{k+1}) - f_2(\boldsymbol{y}^k)$, $w_2 = f_1(\boldsymbol{y}^k) - f_1(\boldsymbol{y}^{k+1})$, $a_1 = 0$, $a_2 = 1$ and $a_3 = f_2(\boldsymbol{y}^{k+1})$.

Routine3: {The double vertical and horizontal search} First, the routine performs the swapping algorithm under time limit *aTime*/2 with initial solution $\boldsymbol{y}^k$ and with parameters $w_1 = 1$, $w_2 = 0$, $a_1 = 0$, $a_2 = 1$ and $a_3 = f_2(\boldsymbol{y}^{k+1})$. Then the routine performs the swapping algorithm with the same initial solution $\boldsymbol{y}^k$ and with parameters $w_1 = 0$, $w_2 = 1$, $a_1 = 1$, $a_2 = 0$ and $a_3 = f_1(\boldsymbol{y}^k)$.

Routine4: {The double composed vertical and horizontal search} The routine works similarly to the *Routine3* with one difference, which consists in the initial solution in the second phase, where this phase starts with the resulting solution of the first phase.

Routine5: {The double composed vertical and radial search} The routine works like *Routine*4, where the first phase uses parameters $w_1 = 1$, $w_2 = 0$, and the second phase works with $w_1 = f_2(\boldsymbol{y}^{k+1}) - f_2(\boldsymbol{y}^k)$, $w_2 = f_1(\boldsymbol{y}^k) - f_1(\boldsymbol{y}^{k+1})$.

V. Proposal of specific hyperheuristic for gradual refinement

The suggested hyperheuristic was embedded into the broader frame of the gradual refinement process at the place of the basic step mentioned in Section3. The hyperheuristic chooses one of the directed search routines based on its evaluation performed in the previous heuristic applications, then the hyperheuristic performs the hosen routine, evaluates its performance and updates the structure *Score*, which is used here instead of *Rank*. The structure *Score*(*r*) is initialized at the beginning of the gradual refinement by a constant and it is updated according to the rule *Score*(*r*) = *Score*(*r*) + (*Area*0 – *Area*1)/*InitArea* during the process. The *Area*0 is *NDSS-Area* before the routine performance and *Area*1 is *NDSS-Area* after the routine performance. Let *noR* denote the number of the directed search routines, then the scheme of the hyperheuristic follows:

Hyperheuristic(*Score*, *NDSS*, *noR*, $\boldsymbol{y}$, *InitArea*)

0. Compute *Area*0 = *NDSS-Area*. Determine *Pr*(*r*) = *Score*(*r*)/sum(*k*=1..*noR*) *Score*(*k*) and define *CumPr*(1) = *Pr*(1) and *CumPr*(*r*) = *CumPr*(*r*-1) + *Pr*(*r*) for *r* = 2, …, *noR*.

1. {Roulette wheel choice} Generate a random number *rn* from interval [0, 1] with uniform probability distribution and find the biggest subscript *r* so that *rn* ≤ *CumPr*(*r*).

2. Perform the *r-th* routine with starting solution $\boldsymbol{y}$.

3. Compute *Area*0 = *NDSS-Area*.

4. Update *Score*(*r*) = *Score*(*r*) + (*Area*0 – *Area*1)/*InitArea*.

VI. Numerical experiments

Efficiency of any heuristic and hyperheuristic as well depends on proper setting of the parameters, which determine heuristic performance. Performance of the presented hyperheuristic is influenced namely by two parameters *aTime* and *InitialScore*. The parameter *aTime* causes prematurely termination of applied neighborhood search routine, what may issue in two consequences. On one side, the neighborhood search is shorter and thus the spared computational time can be used for application of the other routines on other starting solutions. On the other side, the time limit may prevent the restricted routine from finding a better solution or candidates for *NDSS* update. Dependence of resulting *NDSS* quality on the value *aTime* is the first goal of the performed numerical experiments. The range of studied values starts from 16 milliseconds and ends with one second, when the complete time of hyperheuristic performance is set at 300 seconds. The individual values of *aTime* are $16*2^i$ for $i = 0, \ldots, 6$.

The second studied parameter *InitialScore* is assumed to influence the random choice of the currently applied routine. If the parameter value is too small, the probability of the next routine choices may be too influenced by the first choice result instead of overall admissibility of the individual routines. As the preliminary experiments showed that the total maximal increment of the score value varies from 1 to 4, the experiments were performed for the following vales of *InitialScore*: 1, 10 and 100.

Besides the final *NDSS_Area* (*FinArea*) the following parameters were recorded:

noNDSS – the number of non-dominated solutions found.

noTR – the number of time cycle runs performed by the hyperheuristic in the limit of 300 seconds. In one run of the time cycle, the *NDSS* improvement process is started with the initial two-member *NDSS*.

noOR – the total number of runs of outer cycle. The outer cycle run repeats the gradual refinement process with *NDSS* obtained by previously performed outer runs.

noBS – The total number of cases (in the 300 second interval), when the used routines stop their performance due to no solution better than the current one has been found.

noTS – The total number of cases (in the 300 second interval), when the used routines stop their performance due to the limit *aTime* has been reached.

An attention has been also paid to the final values of *Score*, which may indicate the most suitable routine. As the hyperheuristic performance depends on results of random trials, the computation was repeated ten times for each setting of the parameters *aTime* and *InitialScore* and average values of the studied characteristic are presented.

The initial study has been performed using the benchmarks derived from the Slovak self-governing regions, in which the Emergency Medical Service system is operated. Individual instances of the benchmarks are denoted by the following names: Bratislava (BA), Banská Bystrica (BB), Košice (KE), Nitra (NR), Prešov (PO), Trenčín (TN), Trnava (TT) and Žilina (ZA). The sizes of the individual benchmarks are determined by integers *m* and *p*. The number *m* gives the number of possible center locations and *p* gives the number of service centers to be located. The generalized disutility objective function used in the criterion (1) was computed for $r = 3$. The coefficients q_k for k=1 … r have been obtained from statistics presented in [16, 20] and their particular values are as follows: $q_1 = 77.063$, $q_2 = 16.476$ and $q_3 = 100 - q_1 - q_2$.

All numerical experiments were run on a PC equipped with the 11th Gen Intel® Core™ i7 11700KF processor with the parameters: 3,6 GHz and 16 GB RAM. The algorithms were implemented in the Java language and run in the IntelliJ Idea environment. Since the benchmarks used in this computational study were used also in previous research in our department, the complete Pareto fronts are available and the corresponding values of *Area* are known. The basic characteristics of the exact Pareto fronts are summarized in the right part of Table I. Each row of the table corresponds to one problem instance. The middle part of the table contains the size of solved problems, and the right part is devoted to the Pareto front descriptions. The column denoted by *NoS* gives the number of solutions forming the Pareto front. In the column denoted by *Area* we provide the readers with the size of *area* formed by all members of the set of non-dominated solutions as suggested by the expression (3).

TABLE I. BENCHMARKS CHARACTERISTICS AND THE EXACT PARETO FRONTS DESCRIPTIONS

Region	*m*	*p*	*NoS*	*Area*
BA	87	14	34	569039
BB	515	36	229	1002681
KE	460	32	262	1295594
NR	350	27	106	736846
PO	664	32	271	956103
TN	276	21	98	829155
TT	249	18	64	814351
ZA	315	29	97	407293

As mentioned in previous parts of the paper, proposed method may be sensitive to various parameters settings. The first portion of numerical experiments was aimed at studying the impact of various values of *aTime* on the results quality. For this portion of experiments, the parameter *InitialScore* was set to the value of 1. The following Table II and Table III contain the average values of *FinArea* computed according to (3) for each problem instance and each setting of *aTime*.

To make the results easier for quality evaluation, we will report the results also in the form of so-called *gaps*. Generally, *gap* is defined as a relative difference between two values. Here, the area of the complete Pareto front was taken as the base and then, the *gap* is evaluated in percentage. The average values of *gaps* are reported in Table IV and Table V.

TABLE II. AVERAGE *FINAREA* VALUES OF NUMERICAL EXPERIMENTS FOR *INITIALSCORE* = 1 – PART 1

	BA	BB	KE	NR
16 *msec*	577393.0	1012933.7	1334640.4	744174.0
32 *msec*	577393.0	1012193.8	1334978.9	744174.0
64 *msec*	577393.0	1013671.8	1340791.0	744240.8
128 *msec*	577393.0	1014407.5	1341352.9	748310.4
256 *msec*	577459.0	1011771.2	1341693.0	752577.5
512 *msec*	577393.0	1013917.7	1345004.5	766323.7
1024 *msec*	577393.0	1018715.9	1351231.3	768551.7

TABLE III. AVERAGE *FINAREA* VALUES OF NUMERICAL EXPERIMENTS FOR *INITIALSCORE* = 1 – PART 2

	PO	TN	TT	ZA
16 *msec*	967544.7	865429.0	815095.0	412704.9
32 *msec*	967544.5	865429.0	814807.9	412391.0
64 *msec*	964156.6	865429.0	814351.0	414034.4
128 *msec*	966459.7	865429.0	815264.8	413064.9
256 *msec*	967818.6	865429.0	815329.9	414662.2
512 *msec*	965624.0	865429.0	814882.3	410813.7
1024 *msec*	965607.9	865429.0	814956.7	413121.5

TABLE IV. AVERAGE *GAPS* FOR *INITIALSCORE* = 1 AND DIFFERENT SETTINGS OF *ATIME* – PART 1

	BA	BB	KE	NR
16 *msec*	1.47	1.02	3.01	0.99
32 *msec*	1.47	0.95	3.04	0.99
64 *msec*	1.47	1.10	3.49	1.00
128 *msec*	1.47	1.17	3.53	1.56
256 *msec*	1.48	0.91	3.56	2.13
512 *msec*	1.47	1.12	3.81	4.00
1024 *msec*	1.47	1.60	4.29	4.30

TABLE V. AVERAGE *GAPS* FOR *INITIALSCORE* = 1 AND DIFFERENT SETTINGS OF *ATIME* – PART 2

	PO	TN	TT	ZA
16 *msec*	1.20	4.37	0.09	1.33
32 *msec*	1.20	4.37	0.06	1.25
64 *msec*	0.84	4.37	0.00	1.66
128 *msec*	1.08	4.37	0.11	1.42
256 *msec*	1.23	4.37	0.12	1.81
512 *msec*	1.00	4.37	0.07	0.86
1024 *msec*	0.99	4.37	0.07	1.43

Simple analysis of the reported results lead to the finding that the best results were obtained for the second case, i.e., for *aTime* = 32 msec. Therefore, this setting was taken to the next series of numerical experiments, in which we studied the impact of *InitialScore* to the quality of the resulting set of non-dominated solutions. The obtained *FinArea* values are reported in Table VI and Table VII. The parameter *aTime* was set to 32 msec and the *InitialScore* took the values 1, 10 and 100.

TABLE VI. AVERAGE *FINAREA* VALUES OF NUMERICAL EXPERIMENTS FOR *ATIME* = 32 MSEC – PART I

InitialScore \Region	BA	BB	KE	NR
1	577393.0	1012193.8	1334978.9	744174.0
10	577393.0	1014074.5	1336271.2	744174.0
100	577393.0	1013510.5	1337504.7	744174.0

TABLE VII. AVERAGE *FINAREA* VALUES OF NUMERICAL EXPERIMENTS FOR *ATIME* = 32 MSEC – PART 2

InitialScore \Region	PO	TN	TT	ZA
1	967544.5	865429.0	814807.9	412391.0
10	964220.5	865429.0	814351.0	412391.0
100	966423.9	865429.0	814351.0	412391.0

As above, the absolute values of areas were recomputed to the form of gaps. The summary of gaps can be observed in Table VIII and Table IX respectively.

TABLE VIII. AVERAGE *GAPS* FOR *ATIME* = 32 MSEC – PART 1

InitialScore\Region	BA	BB	KE	NR
1	1.47	0.95	3.04	0.99
10	1.47	1.14	3.14	0.99
100	1.47	1.08	3.23	0.99

TABLE IX. AVERAGE *GAPS* FOR *ATIME* = 32 MSEC – PART 2

InitialScore\Region	PO	TN	TT	ZA
1	1.20	4.37	0.06	1.25
10	0.85	4.37	0.00	1.25
100	1.08	4.37	0.00	1.25

As we can see in previous tables, the best results were obtained for *InitialScore*=10. Thus, this combination of parameters settings (*aTime*=32 msec and *InitialScore* = 10) was used to perform the last portion of experiments, the results of which are reported in Table X and Table XI.

TABLE X. RESULTS OF NUMERICAL EXPERIMENTS FOR *INITIALSCORE* = 10 AND *ATIME* = 32 MSEC – PART 1

Region	BA	BB	KE	NR
noNDSS	33.0	205.2	248.9	100.0
FinArea	577393.0	1014074.5	1336271.2	744174.0
noTR	299.6	2.0	2.0	6.8
noOR	1271.4	13.2	15.0	59.0
noBS	69003.8	0.0	0.0	0.0
noTS	9.9	4360.8	6259.9	9502.5
*Score*1	31.7	10.3	10.2	10.9
*Score*2	30.9	10.3	10.5	10.8
*Score*3	74.5	10.3	10.2	10.8
*Score*4	83.3	10.4	10.3	11.6
*Score*5	90.7	10.3	10.5	11.5

TABLE XI. RESULTS OF NUMERICAL EXPERIMENTS FOR *INITIALSCORE* = 10 AND *ATIME* = 32 MSEC – PART 2

Region	PO	TN	TT	ZA
noNDSS	267.1	84.0	64.0	86.0
FinArea	964220.5	865429.0	814351.0	412391.0
noTR	2.0	20.1	28.5	10.9
noOR	7.4	80.0	137.0	84.0
noBS	0.0	0.0	22.4	0.0
noTS	3102.0	11463.3	13953.6	11850.4
*Score*1	10.4	11.7	12.4	11.5
*Score*2	10.3	11.7	12.3	11.5
*Score*3	10.3	13.7	14.1	11.2
*Score*4	10.3	13.5	15.8	12.1
*Score*5	10.3	14.8	17.4	12.9

As far as the quality of the obtained results is concerned, the hyperheuristic proved to be a very efficient tool for public service system design problem, in which two contradictory criteria are taken into account.

VII. CONCLUSION

Healthcare ambulances, fire brigades, police stations, public administration systems and many other types of public service systems are developed and established to keep, regulate and manage specific degree of safety, health and life quality of served people. In this study, we aimed at the operational research subfields that have applications in the public sector. The combinatorial nature of the aforementioned issues logically necessitates the use of various mathematical modeling techniques, software development expertise, or other high-level abilities. So, when making strategic decisions, the experts in operations research cannot be ignored. We are able to quickly and effectively solve significant problems because to the enormous and rapid development made in practically all relevant sectors.

To be more specific, this original research paper was focused on a special class of location problems, in which two conflicting criteria need to be taken into account. Since separate optimization of any of them leads to worsening the value of the second one, a Pareto front of solutions - a small set of nom-dominated system designs – is a suitable output of the optimization process. In this paper, the scientific content was aimed at studying and construction of hyperheuristics assigned to the efficient determination of a non-dominated set of public service system designs, which could produce a good approximation of the original set. The suggested hyperheuristic disposes with a list of subordinate heuristics with dynamically updated ranks depending on their previous success in improving quality of the non-dominated solution set. Based on performed case study on real-world problem instances we can conclude that the hyperheuristic is a very useful tool able to provide the decision-makers with a very accurate solution.

Further research in this area could be focused on the development of other exact or approximate methods for constructing a Pareto front of non-dominated solutions of such location problems, in which two or more criteria need to be optimized.

Acknowledgment

This work was supported by the research grants VEGA 1/0216/21 "Designing of emergency systems with conflicting criteria using tools of artificial intelligence", VEGA 1/0077/22 "Innovative prediction methods for optimization of public service systems" and VEGA 1/0654/22 "Cost-effective design of combined charging infrastructure and efficient operation of electric vehicles in public transport in sustainable cities and regions". This work was supported also by the Slovak Research and Development Agency under the Contract no. APVV-19-0441.

References

[1] Ahmadi-Javid, A., Seyedi, P. et al. (2017). A survey of healthcare facility location, Computers & Operations Research, 79, pp. 223-263.

[2] Arroyo, J. E. C., dos Santos, P. M., Soares, M. S. and Santos, A. G. (2010). A Multi-Objective Genetic Algorithm with Path Relinking for the p-Median Problem. In: Proceedings of the 12th Ibero-American Conference on Advances in Artificial Intelligence, 2010, pp. 70–79.

[3] Avella, P., Sassano, A., Vasil'ev, I. (2007). Computational study of large scale p-median problems. Mathematical Programming 109, pp. 89-114.

[4] Bertsimas, D., Farias, V. F., Trichakis, N. (2011). The Price of Fairness. In Operations Research, 59, 2011, pp. 17-31.

[5] Brotcorne, L, Laporte, G, Semet, F. (2003). Ambulance location and relocation models. Eur. Journal of Oper.Research, 147, pp. 451-463.

[6] Buzna, Ľ., Koháni, M., Janáček, J. (2013). Proportionally Fairer Public Service Systems Design. In: Communications - Scientific Letters of the University of Žilina 15(1), pp. 14-18.

[7] Current, J., Daskin, M. and Schilling, D. (2002). Discrete network location models, Drezner Z. et al. (ed) Facility location: Applications and theory, Springer, pp. 81-118.

[8] Doerner, K. F., Gutjahr, W. J., Hartl, R. F., Karall, M. and Reimann, M. (2005). Heuristic Solution of an Extended Double-Coverage Ambulance Location Problem for Austria. *Central European Journal of Operations Research*, 13(4), pp. 325-340.

[9] Drezner, T., Drezner, Z. (2007). The gravity p-median model. *European Journal of Operational Research* 179, pp. 1239-1251.

[10] Gopal, G. (2013). Hybridization in Genetic Algorithms. *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 3, pp. 403–409.

[11] Grygar, D., Fabricius, R. (2019). An efficient adjustment of genetic algorithm for Pareto front determination. In: TRANSCOM 2019: conference proceedings, Amsterdam: Elsevier Science, pp. 1335-1342.

[12] Ingolfsson, A., Budge, S., Erkut, E. (2008). Optimal ambulance location with random delays and travel times. Health care management science, 11(3), pp. 262-274.

[13] Janáček, J., Fabricius, R. (2021). Public service system design with conflicting criteria. In: IEEE Access: practical innovations, open solutions, ISSN 2169-3536, Vol. 9, pp. 130665-130679.

[14] Janáček, J., Kvet, M. (2021). Swap Heuristics for Emergency System Design with Multiple Facility Location. In: Proceedings of the 39th International Conference on Mathematical Methods in Economics, 2021, pp. 226-231.

[15] Janáček, J., Kvet, M. (2021). Emergency Medical System under Conflicting Criteria. In: SOR 2021 Proceedings, pp. 629-635.

[16] Jankovič, P. (2016). Calculating Reduction Coefficients for Optimization of Emergency Service System Using Microscopic Simulation Model. In: *17th International Symposium on Computational Intelligence and Informatics*, pp. 163-167.

[17] Jánošíková, Ľ. (2007). Emergency Medical Service Planning. In: Communications 9(2), pp. 64-68.

[18] Jánošíková, Ľ. and Žarnay, M. (2014). Location of emergency stations as the capacitated p-median problem. In: Quantitative Methods in Economics (Multiple Criteria Decision Making XVII). pp. 117-123.

[19] Jenelius, E. (2009). Network structure and travel patterns: explaining the geographical disparities of road network vulnerability, Journal of Transport Geography, 17, pp. 234-244.

[20] Kvet, M. (2014). Computational Study of Radial Approach to Public Service System Design with Generalized Utility. In: *Proceedings of International Conference DT 2014*, Žilina, Slovakia, pp. 198-208.

[21] Kvet, M. (2018). Advanced radial approach to resource location problems. In: Developments and advances in intelligent systems and applications. Cham: Springer International Publishing, 2018, Studies in computational intelligence, 718, pp. 29-48.

[22] Kvet, M., Janáček, J. (2021). Incrementing Heuristic for Non-Dominated Designs of Emergency Medical System. In: SOR 2021 Proceedings, pp. 429-474.

[23] Marianov, V. and Serra, D. (2002). Location problems in the public sector*, Facility location - Applications and theory* (Z. Drezner ed.), Berlin, Springer, pp 119-150.