An efficient variable neighborhood search for solving a robust dynamic facility location problem in emergency service network

Stefan Mišković [a,1,2] Zorica Stanimirović [a,3] Igor Grujičić [b,4]

[a] *Faculty of Mathematics, University of Belgrade, Belgrade, Serbia*

[b] *Institute of Computer Graphics and Algorithms, Vienna University of Technology, Vienna, Austria*

Abstract

In this study, we propose a robust variant of a dynamic facility location problem that arises from optimizing the emergency service network of Police Special Forces Units (PSFUs) in the Republic of Serbia. We present for the first time a mathematical programming formulation of the problem under consideration. We further propose a Variable Neighborhood Search (VNS) method with an efficient local search procedure for solving real-life problem instances that remained out of reach of CPLEX solver. The results presented in this paper may help in optimizing the network of PSFUs and other security networks as well.

Keywords: Variable neighborhood search, Dynamic facility location, Emergency service network.

[1] This research was supported by Serbian Ministry of Education, Science and Technological Development under the grants no. 174010 and 47017

[2] Email: `stefan@matf.bg.ac.rs`

[3] Email: `zoricast@matf.bg.ac.rs`

[4] Email: `grujicic@ads.tuwien.ac.at`

1 Problem formulation

The study presented in this paper introduces a dynamic variant of the emergency location problem considered in [1], whose main concern is to establish and optimally utilize the network of PSFUs in the Republic of Serbia. It is defined as follows. Let $J = \{1, ..., m\}$ be the set of potential locations for locating a PSFU and $I = \{1, ..., n\}$ be the set of locations of cities, where $I \subseteq J$, $n \leq m$. The distance matrix is denoted by $D = [d_{ij}]$, $i, j \in J$, and it is assumed that for every two nodes i and j from J, there exists a direct link between them. For each city $i \in I$ and a potential PSFU location $j \in J$, we define $C_{ij} = \{k \in J | d_{ik} \leq d_{ij}\}$. For each $j \in J$ and a given constant c, we define $S_j = \{i \in I | d_{ij} \leq c\}$ as the set of cities $i \in I$ that lie within the range c from the location $j \in J$, and D_j is defined as the complement of S_j. Constant $c > 0$ is chosen as the maximal distance between a PSFU at location j and the city at i, such that a PSFU at j is able to reach the city at i in a timely manner.

Furthermore, we extend the problem from [1] by introducing the given set of time periods $T = \{1, ..., p\}$ and thus obtain the dynamic variant. The motivation lies in the fact that police units usually have 8-hours or 12-hours work shifts, and that the number of criminal acts during day and night shifts may vary significantly. By introducing time periods, a decision maker has a greater degree of flexibility in security planing. For each city $i \in I$ and time period $t \in T$, we assign the value of $f_{it} \geq 0$ that represents the average number of serious criminal acts in the city $i \in I$ for the time period $t \in T$.

In practice, a limited number of PSFUs is available to establish a security network, and therefore, in our study, the number of established PSFUs in all time periods is limited to $k_{max} > 0$. In addition, we limit the difference in the number of established PSFUs in two subsequent time periods $t - 1$ and t to a constant m_t, $2 \leq t \leq |T|$. A single allocation scheme is used, meaning that in each time period each city is allocated to exactly one established PSFU. Since we deal with emergency situations, each city is assigned to its nearest located PSFU.

Two sets of binary decision variables are used in the model. Variable y_{jt} is equal to 1 if a PSFU is established at location j in time period t, and 0 otherwise. Variable x_{ijt} takes the value of 1 in the case that the city at i is assigned to a PSFU at j in time period t, and 0 otherwise. A continuous, non-negative decision variable $L_{max} \geq 0$, which represents the maximum load of located PSFUs through all time periods.

Following the idea from [1], the load of an established PSFU location at node $j \in J$ in a time period $t \in T$ is calculated as $\sum_{i\in S_j} f_{it}x_{ijt} + \sum_{i\in D_j} f_{it}(1+p_{ij})x_{ijt}$, and it depends on the distance that a PSFU needs to travel to a crime scene and the number of heavy criminal acts in the assigned city in the given time period. Parameter p_{ij} is used as a penalty in the case that a city $i \in D_j$ is assigned to a PSFU at location j. The value of parameter p_{ij} is defined as $p_{ij} = \min\{|\frac{d_{ij}-c}{c}|, 1\}$, for $j \in J$ and $i \in D_j$. The goal of the problem is to minimize the maximum load of a located PSFU unit through all time periods, while preserving the efficiency of the emergency system.

Using the notation mentioned above, we intoduce the mathematical formulation of the problem as follows:

$$\min \; L_{max} \tag{1}$$

subject to:

$$\sum_{j\in J} x_{ijt} = 1 \quad \forall i \in I \quad \forall t \in T, \tag{2}$$

$$x_{ijt} \le y_{jt} \quad \forall i \in I \quad \forall j \in J \quad \forall t \in T, \tag{3}$$

$$y_{jt} \le \sum_{k\in C_{ij}} x_{ikt} \quad \forall i \in I \quad \forall j \in J \quad \forall t \in T, \tag{4}$$

$$\sum_{t\in T}\sum_{j\in J} y_{jt} \le k_{max}, \tag{5}$$

$$-m_t \le \sum_{j\in J}(y_{jt} - y_{j,t-1}) \le m_t, \quad \forall t \in T, t \ge 2 \tag{6}$$

$$\sum_{i\in S_j} f_{it}x_{ijt} + \sum_{i\in D_j} f_{it}(1+p_{ij})x_{ijt} \le L_{max} \quad \forall j \in J \quad \forall t \in T, \tag{7}$$

$$x_{ijt} \in \{0,1\} \quad \forall i \in I \quad \forall j \in J \quad \forall t \in T, \tag{8}$$

$$y_{jt} \in \{0,1\} \quad \forall j \in J \quad \forall t \in T, \tag{9}$$

$$L_{max} \ge 0. \tag{10}$$

The objective function (1) combined with constraint (7) minimizes the maximum load of established emergency units in all time periods, such that the difference between the number of established locations for all consecutive time periods $t-1$ and t is limited to m_t – constraint (6). Constraints (2) and (3) ensure that each city is assigned to exactly one, previously located PSFU. By combining constraint (4) with (3), each city is allocated to its nearest located PSFU [4]. By constraint (5), the number of established PSFUs in all time periods is limited to k_{max}. Constraints (8)–(9) denote that variables x_{ijt} and y_{jt} are binary, while (10) reflects the non-negativity of continuous variable L_{max}.

The key challenge in the considered problem of locating PSFUs is the uncertainty of criminal attacks. Robust optimization showed to be an efficient strategy to involve data uncertainty in cases when distribution of input parameters is not known, see [2,3]. In robust variant of our problem, we assume that the input parameters f_{it}, $i \in I$, $t \in T$ are subject to uncertainty. Each f_{it}, $i \in I$, $t \in T$ is modeled as independent, symmetric and bounded random variable with unknown distribution, denoted as $\tilde{f}_{it}$. The variable $\tilde{f}_{it}$ takes values from $[f_{it} - \hat{f}_{it}, f_{it} + \hat{f}_{it}]$, where $\hat{f}_{it} \geq 0$ represents a deviation from nominal coefficient f_{it}. Without loss of generality, we may assume that $\tilde{f}_{it} \in [f_{it}, f_{it} + \hat{f}_{it}]$, $i \in I$, $t \in T$, since by linear substitutions the case of symmetric interval may be reduced to the non-symmetric one.

In practice, it is unlikely that all of the $\hat{f}_{it}$ will change. Therefore, it is assumed that only a subset of the coefficients f_{it} will change in order to adversely affect the solution. We introduce protection level parameters Γ_t, $t \in T$ that take values from the interval $[0, |K_t|]$, where $K_t = \{i \in I \mid \hat{f}_{it} > 0\}$. The role of the parameters Γ_t is to control the level of robustness in the objective. Our goal is to be protected against all cases where up to Γ_t of coefficients f_{it} are allowed to change. In the case of $\Gamma_t = 0$, we completely ignore the influence of cost deviations, while in the case of $\Gamma_t = |K_t|$, we consider all possible cost deviations, which is indeed most conservative. In general, a higher value of Γ_t increases the level of robustness at the expense of higher nominal cost [3]. Following the results presented in [3], the discrete dynamic model is extended to a robust optimization model as follows:

$$\min \ L_{max} \tag{11}$$

subject to (2)–(6), (8)–(10) and

$$\sum_{i \in S_j} f_{it} x_{ijt} + \sum_{i \in D_j} f_{it}(1 + p_{ij}) x_{ijt} + z_t \Gamma_t + \sum_{i \in K_t} r_{it} \leq L_{max} \quad \forall j \in J \quad \forall t \in T, \tag{12}$$

$$z_t + r_{it} \geq \hat{f}_{it} x_{ijt} \quad \forall i \in K_t \quad \forall j \in J \quad \forall t \in T, \tag{13}$$

$$r_{it} \geq 0 \quad \forall i \in K_t \quad \forall t \in T, \tag{14}$$

$$z_t \geq 0 \quad \forall t \in T. \tag{15}$$

2 Proposed VNS method

The proposed VNS method for solving the considered problem consists of two phases: the Multi-start Reduced Variable Neighborhood Search (MRVNS) and the Basic Variable Neighborhood Search (VNS) [5]. The idea is to use the MRVNS method to quickly find a good initial solution for the basic VNS.

Regarding the nature of the considered problem, the solution's code consists of p binary segments of length m, where each segment corresponds to a time period and each bit in a code's segment represents a potential location for establishing a PSFU in the corresponding time period. If the bit on the position $j \in J$ in the segment $t \in T$ takes the value of 1, it means that a PSFU is located at node j in time period t. If not, the bit on the j-th position in the segment t has the value of 0. Neighborhood structures that are used in the proposed VNS method are defined as follows. Let sol be a solution of the given problem, $code(sol)$ its binary code. We will say that

- $sol' \in N_1(sol)$, if within a segment $t \in T$ of $code(sol)$, two randomly chosen positions $i, j \in J$ with different bit values are exchanged and $code(sol')$ is obtained;
- $sol' \in N_2(sol)$, if in two different segments $t_1, t_2 \in T$ of $code(sol)$, two randomly chosen positions $i \in J$ (from segment t_1) and $j \in J$ (from segment t_2) having different bit values are exchanged and $code(sol')$ is obtained;
- $sol' \in N_3(sol)$, if we find bit positions $i, j \in J$ that have different values in all segments $t \in T$ of $code(sol)$ (if such positions exist); By exchanging the corresponding bit values in all segments $t \in T$ of $code(sol)$, we obtain $code(sol')$;
- $sol' \in N_4(sol)$, if we invert the bit value on a randomly chosen position $i \in J$ in a segment $t \in T$, and $code(sol')$ is obtained.

The set of initial solutions sol_i, $i = 1, 2, ...40$ for the MRVNS phase is generated randomly. Then, in the main MRVNS loop, we iteratively try to improve a current solution sol_i by changing the neighborhoods and randomly choosing a solution $sol'_i \in N_k(sol_i)$, $k = 1, 2, 3, 4$, $i = 1, 2, .., 40$. The MRVNS stops with improving each of the sol_i if the maximal number of 5000 iterations is reached.

The best solution from the MRVNS phase $bestSol$ is taken as the initial solution for the VNS phase. The VNS part consists of three steps: *shaking, local search* and *neighborhood exchange*, which are repeated until maximal number of 10 000 iterations is reached. Note that the first improvement strategy is used within the local search phase. In order to improve the efficiency of the objective function calculation of the neighbor sol' of the current solution sol, we apply the following strategies.

Algorithm 1. Pseudocode of the proposed VNS-based algorithm

```
1: **MRVNS part of the algorithm:**
2: randomly generate initial solutions $sol_i$, $i = 1, 2, .., 40$, for the VNS
3: for each $sol_i$, $i = 1, 2, .., 40$ do
4:     while maximal number of iteration is less than 5 000 do
5:         $k \leftarrow 1$
6:         while $k \leq 4$ do
7:             Generate a random $sol'_i \in N_k(sol_i)$
8:             if $Value(sol'_i) < Value(sol_i)$ then
9:                 $sol_i \leftarrow sol'_i$
10:                $k \leftarrow 1$
11:            else
12:                $k \leftarrow k + 1$
13:            end if
14:        end while
15:    end while
16: end for
17: Basic VNS part of the algorithm:
18: $bestSol = \arg\min_{i=1,2,..,40} Value(sol_i)$
19: while maximal number of iteration is less than 10 000 do
20:     $k \leftarrow 1$
21:     while $k \leq 4$ do
22:         Generate a random $bestSol' \in N_k(bestSol)$
23:         $bestSol'' \leftarrow FirstImprovement(bestSol')$
24:         if $Value(bestSol'') < Value(bestSol)$ then
25:             $bestSol \leftarrow bestSol''$
26:             $k \leftarrow 1$
27:         else
28:             $k \leftarrow k + 1$
29:         end if
30:     end while
31: end while
```

- If a bit value at position $j \in J$ in segment $t \in T$ of $code(sol)$ changed its value from 1 to 0, it means that the PSFU is closed on location j in time period t. For the new solution sol', the corresponding load for location $j \in J$ in time period t is set to 0, while all cities $i \in I$ that were previously allocated to $j \in J$ are now being allocated to their nearest PSFU located in period t, and their corresponding loads are being updated.

- If a bit value at position $j \in J$ in segment $t \in T$ of $code(sol)$ changed its value from 0 to 1, it means that the PSFU is opened on location j in time period t. In the new solution sol', for each city $i \in I$ we check if the newly opened PSFU location $j \in J$ is closer compared to its closest opened PSFU location $k \in J$ from sol in time period t. If it is the case, we re-allocate city $i \in I$ to $j \in J$ and update the loads of both $j \in J$ aand $k \in J$ for the time period t.

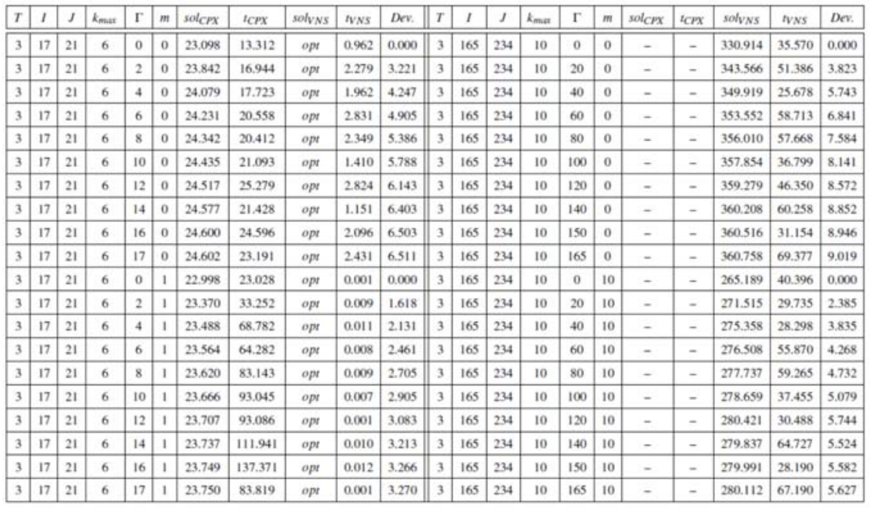
3 Computational results

All computational tests were performed on an Intel Core i5-2430M on 2.4 GHz with 8GB RAM memory under Windows 7 operating system. The optimization package CPLEX 12.1, was used to obtain optimal solutions on considered instances (if possible). The proposed VNS was implemented in C++ programming language. Computational experiments were performed on the set of real-life problem instances from [1], which are modified for the problem under consideration. The description of instances can be found at *http://poincare.matf.bg.ac.rs/ stefan/vns_psfu/*. We have considered $T = 2$ and $T = 3$ time periods, and different values of protection parameter Γ. Conducted computational experiments show the efficiency of the proposed VNS method for solving all instances of the considered problem. In this paper, we present only the results obtained for instance $i12$ with 17 cities and 21 possible PSFU locations and the largest instance *i_all* including whole territory of Serbia. In Table 1, we present optimal solutions obtained by CPLEX solver and the best solutions obtained by the proposed VNS method, together with the corresponding CPU times (in seconds) and percentage of the deviation from the nominal objective ($\Gamma = 0$). In cases when CPLEX produced no solution within 2 hours, a dash is written in the corresponding column.

As it can be seen from Table 1, for all considered cases of the largest instance *i_all*, the CPLEX 12.1 was not able to find the solution. The proposed VNS approach reached all optimal solutions for instance $i12$ and provided best solutions for instance *i_all* in short CPU time. The total VNS running time was under 68 seconds for the largest instance *i_all*. From column *Dev.* in Table 1, we can notice that that deviation of solution increases as Γ increases, and also that for smaller number of possible PSFU reallocations between time periods we obtain larger deviations. Having in mind that the average objective value is smaller when this number is smaller, the proposed approach is better compared to the case when the number of PSFU locations is equal in all time periods.

The obtained results indicate the possibility that the proposed VNS method may be adapted for solving similar emergency network problems. The results presented in this paper may help in optimizing emergency service network of PSFUs and in identifying a sustainable security strategy. The proposed robust dynamic facility location model and VNS approach may be applicable in designing and management of other emergency-service networks as well.

Table 1
The results of the RVNS-VNS algorithm for instances i12 i i_all



T	I	J	k_{max}	Γ	m	sol_{CPX}	t_{CPX}	sol_{VNS}	t_{VNS}	$Dev.$	T	I	J	k_{max}	Γ	m	sol_{CPX}	t_{CPX}	sol_{VNS}	t_{VNS}	$Dev.$
3	17	21	6	0	0	23.098	13.312	*opt*	0.962	0.000	3	165	234	10	0	0	–	–	330.914	35.570	0.000
3	17	21	6	2	0	23.842	16.944	*opt*	2.279	3.221	3	165	234	10	20	0	–	–	343.566	51.386	3.823
3	17	21	6	4	0	24.079	17.723	*opt*	1.962	4.247	3	165	234	10	40	0	–	–	349.919	25.678	5.743
3	17	21	6	6	0	24.231	20.558	*opt*	2.831	4.905	3	165	234	10	60	0	–	–	353.552	58.713	6.841
3	17	21	6	8	0	24.342	20.412	*opt*	2.349	5.386	3	165	234	10	80	0	–	–	356.010	57.668	7.584
3	17	21	6	10	0	24.435	21.093	*opt*	1.410	5.788	3	165	234	10	100	0	–	–	357.854	36.799	8.141
3	17	21	6	12	0	24.517	25.279	*opt*	2.824	6.143	3	165	234	10	120	0	–	–	359.279	46.350	8.572
3	17	21	6	14	0	24.577	21.428	*opt*	1.151	6.403	3	165	234	10	140	0	–	–	360.208	60.258	8.852
3	17	21	6	16	0	24.600	24.596	*opt*	2.096	6.503	3	165	234	10	150	0	–	–	360.516	31.154	8.946
3	17	21	6	17	0	24.602	23.191	*opt*	2.431	6.511	3	165	234	10	165	0	–	–	360.758	69.377	9.019
3	17	21	6	0	1	22.998	23.028	*opt*	0.001	0.000	3	165	234	10	0	10	–	–	265.189	40.396	0.000
3	17	21	6	2	1	23.370	33.252	*opt*	0.009	1.618	3	165	234	10	20	10	–	–	271.515	29.735	2.385
3	17	21	6	4	1	23.488	68.782	*opt*	0.011	2.131	3	165	234	10	40	10	–	–	275.358	28.298	3.835
3	17	21	6	6	1	23.564	64.282	*opt*	0.008	2.461	3	165	234	10	60	10	–	–	276.508	55.870	4.268
3	17	21	6	8	1	23.620	83.143	*opt*	0.009	2.705	3	165	234	10	80	10	–	–	277.737	59.265	4.732
3	17	21	6	10	1	23.666	93.045	*opt*	0.007	2.905	3	165	234	10	100	10	–	–	278.659	37.455	5.079
3	17	21	6	12	1	23.707	93.086	*opt*	0.001	3.083	3	165	234	10	120	10	–	–	280.421	30.488	5.744
3	17	21	6	14	1	23.737	111.941	*opt*	0.010	3.213	3	165	234	10	140	10	–	–	279.837	64.727	5.524
3	17	21	6	16	1	23.749	137.371	*opt*	0.012	3.266	3	165	234	10	150	10	–	–	279.991	28.190	5.582
3	17	21	6	17	1	23.750	83.819	*opt*	0.001	3.270	3	165	234	10	165	10	–	–	280.112	67.190	5.627

References

[1] Grujičić, I. and Z. Stanimirović, *Variable neighborhood search method for optimizing the emergency service network of police special forces units*, Electronic Notes in Discrete Mathematics **39** (2012), pp. 185–192.

[2] Ben-Tal, A. and A. Nemirovski, *Robust solutions of linear programming problems contaminated with uncertain data*, Mathematical programming **88** (2000), pp. 411–424.

[3] Bertsimas, D. and M. Sim, *Robust discrete optimization and network flows*, Mathematical programming **98** (2003), pp. 49–71.

[4] Espejo, I., A. Marín and A. M. Rodríguez-Chía, *Closest assignment constraints in discrete location problems*, European Journal of Operational Research **219(1)** (2012), pp. 49–58.

[5] Hansen, P., N. Mladenović, *Chapter 6: Variable neighborhood search*, In: Glover, P., G. A. Kochenberger (Eds), "Handbook of Metaheuristics", Kluwer Academic Publishers (2003), pp. 145–185.