Yugoslav Journal of Operations Research
15 (2005), Number 1, 53-63

LARGE NEIGHBORHOOD LOCAL SEARCH FOR THE P-MEDIAN PROBLEM

Yuri KOCHETOV, Ekaterina ALEKSEEVA
Tatyana LEVANOVA, Maxim LORESH
Sobolev Institute of Mathematics, Russia

Presented at XXX Yugoslav Simposium on Operations Research
Received: January 2004 / Accepted: January 2005

Abstract: In this paper we consider the well known *p*-median problem. We introduce a new large neighborhood based on ideas of S.Lin and B.W. Kernighan for the graph partition problem. We study the behavior of the local improvement and Ant Colony algorithms with new neighborhood. Computational experiments show that the local improvement algorithm with the neighborhood is fast and finds feasible solutions with small relative error. The Ant Colony algorithm with new neighborhood as a rule finds an optimal solution for computationally difficult test instances.

Keywords: Large neighborhood, Lagrangean relaxations, ant colony, *p*-median, benchmarks.

1. INTRODUCTION

In the *p*-median problem we are given a set $I=\{1,\ldots,m\}$ of m potential locations for p facilities, a set $J=\{1,\ldots,n\}$ of n customers, and a $n\times m$ matrix (g_{ij}) of transportation costs for servicing the customers by the facilities. If a facility i can not serve a customer j then we assume $g_{ij} = +\infty$. Our gain is to find a feasible subset $S \subset I$, $|S| = p$ such that minimizes the objective function

$$F(S) = \sum_{j\in J} \min_{i\in S} g_{ij}.$$

This problem is NP-hard in strong sense. So, the metaheuristics such as Ant Colony, Variable Neighborhood Search and others [7] are the most appropriate tools for the problem.

In this paper we introduce a new large neighborhood based on ideas of S.Lin and B.W. Kernighan for the graph partition problem [9]. We study the behavior of the local improvement algorithm with different starting points: optimal solutions of

Lagrangean relaxation randomized rounding of optimal solution for the linear programming relaxation, and random starting points. Computational experiments show that the local improvement algorithm with new neighborhood is fast and finds feasible solutions with small relative error for all starting points. Moreover, the Ant Colony heuristic with new neighborhood as a rule finds an optimal solution for computational difficult test instances.

The paper is organized as follows. In section 2 we describe Swap, *k*-Swap and Lin-Kernighan neighborhoods for the *p*-median problem. Section 3 presents Lagrangean relaxation and randomized rounding procedures for selecting of starting points for the local improvement algorithm. The framework of Ant Colony heuristic is considered in section 4. Finally, the difficult test instances and computational results are discussed in sections 5 and 6. In section 7 we give conclusions and further research directions.

2. ADAPTIVE NEIGHBORHOODS

Standard local improvement algorithm starts from an initial solution and moves to a better neighboring solution until it terminates at a local optimum. For a subset S the Swap neighborhood contains all subsets S', $|S'| = p$, with Hamming distance from S' to S at most 2:

$$\text{Swap }(S) = \{S' \subset I \mid |S'| = p,\ d(S,S') \le 2\}.$$

By analogy, the *k*-Swap neighborhood is defined as follows:

$$k\text{-Swap }(S) = \{S' \subset I \mid |S'| = p,\ d(S,S') \le 2k\}.$$

Finding the best element in the *k*-Swap neighborhood requires high efforts for large *k*. So, we introduce a new neighborhood which is a part of the *k*-Swap neighborhood and based on the greedy strategy [1].

Let us define the Lin-Kernighan neighborhood (*LK*) for the *p*-median problem. For the subset S it consists of k elements, $k \le n - p$, and can be described by the following steps.

Step 1. Choose two facilities $i_{ins} \in I \setminus S$ and $i_{rem} \in S$ such that $F(S \cup \{i_{ins}\} \setminus \{i_{rem}\})$ is minimal even if it greater than $F(S)$.
Step 2. Perform exchange of i_{rem} and i_{ins}.
Step 3. Repeat steps 1, 2 k times so that a facility can not be chosen to be inserted in S if it has been removed from S in one of the previous iterations of step 1 and step 2.

The sequence $\{(i^{\tau}_{ins}, i^{\tau}_{rem})\}_{\tau \le k}$ defines k neighbors S_τ for the subset S. The best element in the Swap neighborhood can be found in $O(nm)$ time [12]. Hence, we can find the best element in the *LK*-neighborhood in $O(knm)$ time. We say that S is a local minimum with respect to the *LK*-neighborhood if $F(S) \le F(S_\tau)$ for all $\tau \le k$. Any local minimum with respect to the *LK*-neighborhood is a local minimum with respect to the Swap neighborhood and may be not a local minimum with respect to the *k*-Swap neighborhood.

3. STARTING POINTS

Let us rewrite the *p*-median problem as a 0-1 program:

$$\min \sum_{i\in I}\sum_{j\in J} g_{ij} y_{ij} \tag{1}$$

s.t.

$$\sum_{i\in I} y_{ij} = 1, \quad j\in J \tag{2}$$

$$x_i \ge y_{ij} \ge 0, \quad i\in I,\ j\in J \tag{3}$$

$$\sum_{i\in I} x_i = p \tag{4}$$

$$y_{ij}, x_i \in \{0,1\},\ \ i\in I,\ \ j\in J. \tag{5}$$

In this formulation x_i=1 if $i\in S$ and x_i= 0 otherwise. Variables y_{ij} define a facility that serves the customer *j*. We may set y_{ij}=1 for a facility *i* that achieves $min_{i\in S}\, g_{ij}$ and set y_{ij}= 0 otherwise. Lagrangean relaxation with multipliers u_j which correspond to equations (2) is the following program:

$$L(u) = \min \sum_{i\in I}\sum_{j\in J} (g_{ij} - u_j) y_{ij} + \sum_{j\in J} u_j$$

s.t. (3), (4), (5).

It is easy to find an optimal solution *x*(*u*), *y*(*u*) of the problem in polynomial time [6].

The dual problem

$$\max_u L(u)$$

can be solved by subgradient optimization methods, for example, by the Volume algorithm [2,3]. It produces a sequence of Lagrangean multipliers u_j^t, *t* =1,2,…,*T*, as well as a sequence of optimal solutions $x(u^t)$, $y(u^t)$ of the problem $L(u^t)$. Moreover, the algorithm allows us to get an approximation $\bar{x}, \bar{y}$ of the optimal solution for the linear programming relaxation (1)–(4). In order to get starting points for the local improvement algorithm we use optimal solutions $x(u^t)$ or apply the randomized rounding procedure to the fractional solution $\bar{x}$.

4. ANT COLONY OPTIMIZATION

The Ant Colony algorithm (AC) was initially proposed by Colorni *et al.* [5, see also 7]. The main idea of the approach is to use the statistical information of previously obtained results to guide the search process into the most promising parts of the feasible domain. It is iterative procedure. At the each iteration, we construct a prescribed number of solutions by the following Randomized Drop heuristic (RD):

Randomized Drop Heuristic

```
1. Put $S := I$
2. While $|S| > p$ do
    2.1 Select $i \in S$ at random
    2.2 Update $S := S \setminus \{i\}$
3. Apply the local improvement algorithm to $S$.
```

The step 2.1 is crucial in the heuristic. To select an element *i* we should bear in mind the variation of the objective function $\Delta F_i = F(S) - F(S \setminus \{i\})$ and additional information about attractiveness of the element *i* from the point of view a set of local optima obtained at the previous iterations. To realize the strategy, we define a candidate set by the following:

$$S(\lambda) = \{i \in S \mid \Delta F_i \le (1-\lambda)\min_{l \in S} \Delta F_l + \lambda \max_{l \in S} \Delta F_l\} \text{ for } \lambda \in (0,1).$$

At the step 2.1, the element i is selected in $S(\lambda)$ instead of the set S. Probability p_i to draw an element i depends on the variation ΔF_i and a value α_i that expresses a priority of i to remove from the set S. More exactly, the probability p_i is defined as follows:

$$p_i = \frac{\alpha_i(\max_{l \in S} \Delta F_l - \Delta F_i + \varepsilon)}{\sum_{k \in S(\lambda)} \alpha_k(\max_{l \in S} \Delta F_l - \Delta F_k + \varepsilon)}, \quad i \in S(\lambda),$$

where ε is a small positive number. To define α_i we present the framework of AC.

AC algorithm $(\alpha^0, T, K, \bar{K})$

```
1. Put $\alpha_i := 1, i \in I, F^* := +\infty$
2. While $t < T$ do
    2.1 Compute local optima $S_1, \ldots, S_K$ by the RD heuristic
    2.2 Select $\bar{K}$ minimal local optima: $F(S_1) \le F(S_2) \le \ldots \le F(S_{\bar{K}})$,  $\bar{K} < K$
    2.3 Update $\alpha_i, i \in I$ using $S_1, \ldots, S_{\bar{K}}$
    2.4 If $F^* > F(S_1)$ then
        2.4.1 $F^* := F(S_1)$
        2.4.2 $S^* := S_1$
        2.4.3 $\alpha_i := \alpha^0, i \in S^*$
3. Return $S^*$
```

The algorithm has four control parameters:

α^0 : the minimal admissible value of $\alpha_i, i \in I$;

T : the maximal number of the iterations;

K : the number of local optima obtained with fixed values of α_i, $i \in I$;

$\bar{K}$: the number of local optima which used to update of α_i, $i \in I$.

At the step 2.3, we have the local optima $S_1, \ldots, S_{\bar{K}}$ and compute the frequency γ_i of opening facility i in the solutions $S_1, \ldots, S_{\bar{K}}$. If $\gamma_i = 0$ then the facility i is closed in all solutions $S_1, \ldots, S_{\bar{K}}$. To modify α_i we use the following rule:

$$\alpha_i := \frac{\alpha^0 + q^{\gamma_i}(\alpha_i - \alpha^0)}{\beta}, \quad i \in I ,$$

where control parameters $0 < q < 1$, $0 < \beta < 1$ are used to manage the adaptation.

5. COMPUTATIONALLY DIFFICULT INSTANCES

5.1. Polynomially solvable instances

Let us consider a finite projective plane of order k [8]. It is a collection of $n = k^2 + k + 1$ points $p_1, \ldots, p_n$ and lines $L_1, \ldots, L_n$. An incidence matrix A is an $n \times n$ matrix defining the following: $a_{ij} = 1$ if $p_j \in L_i$ and $a_{ij} = 0$ otherwise. The incidence matrix A satisfying the following properties:

- A has constant row sum $k + 1$;
- A has constant column sum $k + 1$;
- the inner product of any two district rows of A is 1;
- the inner product of any two district columns of A is 1.

These matrices exist if k is a power of prime. A set of lines $B_j = \{L_i \mid p_j \in L_i\}$ is called a bundle for the point p_j. Now we define a class of instances for the p-median problem. Put $I = J = \{1, \ldots, n\}$, $p = k + 1$ and

$$g_{ij} = \begin{cases} \xi, & \text{if } a_{ij} = 1, \\ +\infty & \text{otherwise,} \end{cases}$$

where ξ is a random number taken from the set {0, 1, 2, 3, 4} with uniform distribution. We denote the class of instances by FPP_k . From the properties of the matrix A we can get that an optimal solution for FPP_k corresponds to a bundle. Hence, an optimal solution for the corresponding p-median problem can be found in polynomial time.

Every bundle of the plane accords with a feasible solution of the p-median problem and vice versa. For any feasible solution S, the (k-1)-Swap neighborhood has one element only. So, the landscape for the problem with respect to the neighborhood is a collection of isolated vertices. This case is hard enough for the local search methods if k is sufficiently large.

5.2. Instances with exponential number of strong local optima

Let us consider two classes of instances where number of strong local optima grows exponentially as dimension increases. The first class uses the binary perfect codes with code distance 3. The second class is constructed with help a chess board.

5.2.1. Instances based on perfect codes

Let B^k be a set of words or vectors of length k over an alphabet $\{0, 1\}$. A binary code of length k is an arbitrary nonempty subset of B^k. Perfect binary code with distance 3 is a subset $C \subseteq B^k$, $|C|=2^k/(k+1)$ such that Hamming distance $d(c_1,c_2) \geq 3$ for all $c_1, c_2 \in C$, $c_1 \neq c_2$. These codes exist for $k=2^r-1$, $r > 1$, integer.

Put $n = 2^k$, $I = J = \{1,\dots, n\}$, and $p=|C|$. Every element $i \in I$ corresponds to a vertex $v(i)$ of the binary hyper cube $\mathbf{Z}_2^k$. Therefore, we may use Hamming distance $d(v(i), v(j))$ for any two elements $i, j \in I$. Now we define

$$g_{ij} = \begin{cases} \xi, & \text{if } d(v(i), v(j)) \leq 1, \\ +\infty & \text{otherwise,} \end{cases}$$

where ξ is a random number taken in the set $\{0, 1, 2, 3, 4\}$ with uniform distribution. The number of perfect codes $\aleph(k)$ grows exponentially as k increases. The best known lower bound [10] is

$$\aleph(k) \geq 2^{2^{\frac{k+1}{2}-\log_2(k+1)}} \cdot 3^{2^{\frac{k-3}{4}}} \cdot 2^{2^{\frac{k+5}{4}-\log_2(k+1)}}.$$

Each feasible solution of the p-median problem corresponds to a binary perfect code with distance 3 and vice versa. The minimal distance between two perfect codes or feasible solutions is at least $2^{(k+1)/2}$. We denote the class of benchmarks by PC_k.

5.2.2. Instance based on a chess board

Let us glue boundaries of the $3k\times 3k$ chess board so that we get a torus. Put $r = 3k$. Each cell of the torus has 8 neighboring cells. For example, the cell (1,1) has the following neighbors: (1,2), (1,r), (2,1), (2,2), (2,r), (r,1), (r,2), (r,r). Define $n = 9k^2$, $I = J = \{1,\dots,n\}$, $p = k^2$, and

$$g_{ij} = \begin{cases} \xi, & \text{if the cells } i, j \text{ are neighbors} \\ +\infty & \text{otherwise,} \end{cases}$$

where ξ is a random number taken from the set $\{0, 1, 2, 3, 4\}$ with uniform distribution. The torus is divided into k^2 squares by 9 cells in each of them. Every cover of the torus by k^2 squares corresponds to a feasible solution for the p-median problem and vise versa. The total number of feasible solutions is $2\cdot 3^{k+1}-9$. The minimal distance between them is $2k$. We denote the class of benchmarks by CB_k.

5.3. Instances with large duality gap

Let the $n \times n$ matrix (g_{ij}) has the following property: each row and column have the same number of non-infinite elements. We denote this number by l. The value l/n is called the density of the matrix. Now we present an algorithm to generate random matrices (g_{ij}) with the fixed density.

Random matrix generator (*l*,*n*)

```
1. $J \leftarrow \{1,\ldots,n\}$
2. Column $[j] \leftarrow 0$ for all $j \in J$
3. $g[i,j] \leftarrow +\infty$ for all $i, j \in J$
4. for $i \leftarrow 1$ to $n$
5.     do $l_0 \leftarrow 0$
6.      for $j \leftarrow 1$ to $n$
7.           do if $n - i + 1 = l -$ Column $[j]$
8.                then $g[i, j] \leftarrow \xi$
9.                     $l_0 \leftarrow l_0+1$
10.                    Column $[j] \leftarrow$ Column $[j]+1$
11.                    $J \leftarrow J \setminus j$
12.     select a subset $J' \subset J$, $|J'| = l - l_0$ at random and
        put $g[i,j] \leftarrow \xi$ for $j \in J'$
```

The array Column $[j]$ keeps the number of *small* elements in j-th column of the generating matrix. Variable l_0 is used to count the columns where small elements must be located in i-th row. These columns are detected in advance (line 7) and removed from the set J (line 11). Note that we may get random matrices with exactly l small elements for each row only if we remove lines 6–11 from the algorithm. By transposing we get random matrices with this property for columns only. Now we introduce three classes of benchmarks:

Gap-A: each column of the matrix (g_{ij}) has exactly l small elements
Gap-B: each row of the matrix (g_{ij}) has exactly l small elements
Gap-C: each column and row of the matrix (g_{ij}) has exactly l small elements.

For each instance we define p as a minimal value of facilities which can serve all customers. In computational experiments we put $l = 10$, $n = m = 100$ and $p = 12 \div 15$.

The instances have significant duality gap:

$$\delta = \frac{F_{opt} - F_{LP}}{F_{opt}} \cdot 100\%,$$

where F_{LP} is an optimal solution for the linear programming relaxation. In average, we observe $\delta \approx 35.5\%$ for class *Gap-A*, $\delta \approx 37.6\%$ for class *Gap-B*, $\delta \approx 41.5\%$ for class *Gap-C*. For comparison, $\delta \approx 9.84\%$ for class FPP_{11}, $\delta \approx 14.9\%$ for class CB_4, $\delta \approx 1.8\%$ for class PC_7.

6. COMPUTATIONAL EXPERIMENTS

All algorithms were coded and tested on instances taken from the electronic benchmarks libraries: the well-known OR Library [4] and new library "Discrete Location Problems" available by address: http://www.math.nsc.ru/AP/bench-marks/P-median/p-med_eng.html. All instances are random generated and were solved exactly. For OR Library instances, the elements g_{ij} are Euclidean distances between random points on two dimensional Euclidean plane. The density of the matrices is 100 %. The problem parameters range from instances with $n = m = 100$, $p = 5$, 10 and up to instances with $n = m = 900$, $p = 5$. Our computational experiments show that the instances are quite easy. The local improvement algorithm with random starting points and simple restart strategy with 100 trials finds an optimal solution for instances with $n = 100 \div 700$ if we use Swap neighborhood. For the *LK*-neighborhood, the algorithm finds an optimal solution for all OR Library instances.

The new library "Discrete Local Problems" contains more complicated instances for the p-median problem. For every class discussed above, 30 test instances are available. The density of matrices (gij) is small, about 10 % – 16 %. We study the behavior of the local improvement and Ant Colony algorithms for these tests. Three variants of local improvement algorithm are considered:

LR: Local improvement with starting points $x(u^t)$.

RR: Local improvement with starting points generated by the randomized rounding procedure applied to the fractional solution $\bar{x}$.

Rm: Local improvement with random starting points.

In computational experiments every algorithm finds 120 local optima. The best of them is returned.

Table 1: Average relative error for the algorithms with Swap neighborhood

Benchmarks	*n, p*	*RR*	*LR*	*Rm*	*AC*
Gap-A	100, 12-13	1.31	1.34	1.12	0.00
Gap-B	100, 14-15	4.79	4.48	5.45	0.00
Gap-C	100, 14	6.53	5.19	8.65	0.00
FPP_{11}	133, 12	0.09	0.07	0.15	0.00
PC_7	128, 16	0.07	0.05	3.49	0.00
CB_4	144, 16	1.32	1.32	0.96	0.00
Uniform	100, 12	0.11	0.05	0.01	0.00

Table 1 presents the average relative error for the algorithms with Swap neighborhood. For comparison, we include additional Uniform class of test instances. The elements g_{ij} are taken in interval $[0, 10^4]$ at random with uniform distribution. The density of the matrices is 100% and $p = 12$.

Table 2: The percent of trials when no feasible solutions obtained by Swap neighborhood

Benchmarks	*n, p*	*RR*	*LR*	*Rm*	*AC*
Gap-A	100, 12-13	33.50	19.17	15.50	16.11
Gap-B	100, 14-15	37.22	37.22	36.11	27.44
Gap-C	100, 14	34.40	38.07	26.93	20.88
FPP_{11}	133, 12	0.00	0.00	0.00	0.00
PC_7	128, 16	0.00	0.00	0.33	0.00
CB_4	144, 16	6.17	0.50	0.00	0.00
Uniform	100, 12	0.00	0.00	0.00	0.00

Table 2 presents a percent of trials when no feasible solutions can be obtained. By the experiments we may conclude that Ant Colony approach shows the best results. As a rule, it finds optimal solutions. The local improvement algorithm is weaker. Nevertheless, it can find feasible solutions with small relative error. It is interesting to note that *LR* and *RR* algorithms [3] without local improvement procedure can not find feasible solutions for difficult test instances. So, the stage of local improvement is very important for the *p*-median problem.

Table 3: Average relative error for the algorithms with *LK*- neighborhood

Benchmarks	*n, p*	*RR*	*LR*	*Rm*	*AC*
Gap-A	100, 12-13	0.33	0.51	0.20	0.00
Gap-B	100, 14-15	1.08	1.16	0.97	0.00
Gap-C	100, 14	1.69	1.44	1.61	0.00
FPP_{11}	133, 12	0.09	0.07	0.09	0.00
PC_7	128, 16	0.05	0.04	2.35	0.00
CB_4	144, 16	0.13	0.09	0.00	0.00
Uniform	100, 12	0.00	0.00	0.00	0.00

Table 4. The percent of trials when no feasible solutions obtained by *LK*- neighborhood

Benchmarks	*n, p*	*RR*	*LR*	*Rm*	*AC*
Gap-A	100, 12-13	10.67	5.78	3.78	4.33
Gap-B	100, 14-15	9.67	7.56	7.56	13.44
Gap-C	100, 14	0.00	0.00	0.00	0.00
FPP_{11}	133, 12	0.00	0.00	0.00	0.00
PC_7	128, 16	0.00	0.00	0.00	0.00
CB_4	144, 16	0.00	0.00	0.00	0.00
Uniform	100, 12	0.00	0.00	0.00	0.00

Table 5: The average number of steps by the Swap neighborhood to reach a local optimum

Benchmarks	*n, p*	*RR*	*LR*	*Rm*	*AC*
Gap-A	100, 12-13	6.76	7.70	10.69	5.11
Gap-B	100, 14-15	10.24	11.24	11.92	5.85
Gap-C	100, 14	9.78	10.74	10.74	5.80
FPP_{11}	133, 12	7.52	7.89	9.24	5.02
PC_7	128, 16	4.77	7.72	12.17	6.39
CB_4	144, 16	7.19	8.02	13.93	7.64
Uniform	100, 12	6.01	6.25	13.08	5.12

Table 6: The average number of steps by the *LK*- neighborhood to reach a local optimum

Benchmarks	*n, p*	*RR*	*LR*	*Rm*	*AC*
Gap-A	100, 12-13	7.85	7.81	9.28	0.67
Gap-B	100, 14-15	11.84	11.68	2.59	0.84
Gap-C	100, 14	10.78	10.50	11.70	0.89
FPP_{11}	133, 12	0.39	0.34	0.29	0.017
PC_7	128, 16	0.4	0.85	2.04	0.19
CB_4	144, 16	11.00	10.95	15.72	0.58
Uniform	100, 12	4.57	4.18	7.49	0.38

Table 3 and 4 show results for the *LK*-neighborhood. Comparison these tables and two previous ones persuade that the *LK*-neighborhood allows to improve the performance of the algorithms indeed. We get feasible solutions more often. The relative error decreases. Tables 5 and 6 present average number of steps by Swap and *LK*-neighborhoods to reach a local optimum. As we can see, a path from starting points to local optima is shot enough.

7. CONCLUSIONS

In this paper we have introduced a new promising neighborhood for the *p*-median problem. It contains at most *n–p* elements and allows the local improvement algorithm to find near optimal solutions for difficult test instances and optimal solutions for Euclidean instances with middle dimensions. We hope this new neighborhood will be useful for more powerful meta-heuristics [7]. For example, the Ant Colony algorithm with *LK*-neighborhood shows excellent results for all test instances considered.

Another interesting direction for research is computational complexity of the local search procedure with Swap and *LK*-neighborhoods for the *p*-median problem. It seems plausible that the problem is *PLS*-complete from the point of view of the worst case analysis [15], but is solvable polynomially in average case [14].

Acknowledgment: This research was supported by the Russian Foundation for Basic Research, grant 03-01-00455.

REFERENCES

[1] Ahuja, R.K., James, O.E., Orlin, B., and Punnen, A.P., "A survey of very large-scale neighborhood search techniques", *Discrete Applied Mathematics*, 123 (2002) 75–102.

[2] Barahona, F., and Anbil, R., "The volume algorithm: producing primal solutions with a subgradient method", *Mathematical Programming*, *Ser.A,* 87 (2000) 385–399.

[3] Barahona, F., and Chudak, F.A., "Solving large scale uncapacitated facility location problems", in: Pardalos P.M. (ed.), *Approximation and Complexity in Numerical Optimization: Continuous and Discrete Problems*, Kluwer Academic Publishers, 1999.

[4] Beasley, J.E., "OR-Library: Distributing test problems by electronic mail", *Journal of the Operational Research Society*, 41(11) (1990) 1069-1072.

[5] Colorny, A., Dorigo, M., and Maniezzo, V., "Distributed optimization by ant colonies", *Proceedings of the 1991 European Conference on Artificial Life*, 1991, 134-142.

[6] Erlenkotter, D., "A dual-based procedure for uncapacitated facility location", *Operations Research*, 26 (1978) 992–1009.

[7] Hansen, P., and Mladenović, N., "An introduction to variable neighborhood search", in: S.Voss et al. (eds.), *Metaheuristics, Advances and Trends in Local Search Paradigms for Optimization*, Kluwer Academic Publishers, Dorchester, 1998.

[8] Hall, M. Jr., *Combinatorial Theory*, Blaisdell, Waltham, MA, 1967.

[9] Kernighan, B.W., and Lin, S., "An effective heuristic procedure for partitioning graphs", *Bell System Technical Journal*, 49 (1970) 291–307.

[10] Krotov, D.S., "Lower bounds for number of *m*- quasi groups of order 4 and number of perfect binary codes", *Discrete Analysis and Operations Research, Ser.1*, 7(2) (2000) 47-53. (in Russian)

[11] Mirchandani, P.B., and Francis, R.L. (eds.), *Discrete Location Theory*, Wiley-Interscience, 1990.

[12] Resende, M.G.C., and Werneck, R.F., "On the implementation of a swap-based local search procedure for the *p*-median problem", in: R.E. Lander (ed.), *Proceedings of the Fifth Workshop on Algorithm Engineering and Experiments (ALENEX'03)*, SIAM, 2003, 119-127,

[13] Nemhauser, G.L., and Wolsey, L.A., *Integer and Combinatorial Optimization*, John Wiley & Sons, 1988.

[14] Tovey, C.A., "Local improvement on discrete structures", in: E.H.L. Aarts, J.K. Lenstra (eds.), *Local Search in Combinatorial Optimization*, Wiley, Chichester, 1997.

[15] Yannakakis, M., "Computational complexity", in: E.H.L.Aarts, J.K.Lenstra (eds.), *Local Search in Combinatorial Optimization*, Wiley, Chichester, 1997.