DOI: 10.15514/ISPRAS-2019-31(4)-8

Local search metaheuristics for Capacitated Vehicle Routing Problem: a comparative study

S.M. Avdoshin, ORCID: 0000-0001-8473-8077 <savdoshin@hse.ru>
E.N. Beresneva, ORCID: 0000-0001-6710-2843 <eberesneva@hse.ru>
Department of Software Engineering,
National Research University Higher School of Economics,
20, Myasnitskaya st., Moscow, 101000 Russia

Abstract. This study is concerned with local search metaheuristics for solving Capacitated Vehicle Routing Problem (CVRP). In this problem the optimal design of routes for a fleet of vehicles with a limited capacity to serve a set of customers must be found. The problem is NP-hard, therefore heuristic algorithms which provide near-optimal polynomial-time solutions are still actual. This experimental analysis is a continue of previous research on construction heuristics for CVRP. It was investigated before that Clarke and Wright Savings (CWS) heuristic is the best among constructive algorithms except for a few instances with geometric type of clients' distribution where Nearest Neighbor (NN) heuristic is better. The aim of this work is to make a comparison of best-known local search metaheuristics by criteria of error rate and running time with CWS or NN as initial algorithms because there were not found any such state-of-the-art comparative study. An experimental comparison is made using 8 datasets from well-known library because it is interesting to analyze "effectiveness" of algorithms depending on type of input data. Overall, five different groups of Pareto optimal algorithms are defined and described.

Keywords: capacitated vehicle routing problem; local search metaheuristics; LKH-3; variable record-to-record travel; simulated annealing; guided local search; tabu search

For citation: Avdoshin S.M., Beresneva E.N. Local search metaheuristics for Capacitated Vehicle Routing Problem: a comparative study. Trudy ISP RAN/Proc. ISP RAS, vol. 31, issue 4, 2019, pp. 121-138. DOI: 10.15514/ISPRAS-2019-31(4)-8

Эвристические методы конструирования маршрута для решения задачи маршрутизации с ограничением по грузоподъемности

С.М. Авдошин, ORCID: 0000-0001-8473-8077 <savdoshin@hse.ru>
Е.Н. Береснева, ORCID: 0000-0001-6710-2843 <eberesneva@hse.ru>
Департамент программной инженерии,
Национальный исследовательский университет «Высшая школа экономики», 101000,
Россия, г. Москва, ул. Мясницкая, д. 20.

Abstract. Задача маршрутизации – одна из широко известных задач комбинаторной оптимизации. Она состоит в отыскании оптимального множества маршрутов для транспортных средств с целью однократного обслуживания определенного множества клиентов. В данной работе исследуется подвид задачи маршрутизации – задача маршрутизации с ограничением по грузоподъемности, в которой каждое транспортное средство имеет свою грузоподъемность. Задача является NP-трудной, поэтому вместо точных алгоритмов решения исследуются только эвристические алгоритмы, позволяющие получить приближенные решения за полиномиальное время. Данная работа является продолжением исследования, посвященного алгоритмам конструирования маршрута; оно позволяет получить первоначальные решения для данной задачи. Было выяснено, что эвристика Clarke and Wright Savings (CWS) является одной из лучших, за исключением наборов данным с геометрическим расположением точек, для которых лучшим является алгоритм Nearest Neighbor (NN). Целью работы является сравнение локально-оптимальных метаэвристических алгоритмов решения задачи маршрутизации с ограничением по грузоподъемности по двум критериям: точность и время решения, для получения начального решения используются алгоритмы CWS и NN. Выявлено пять Парето оптимальных групп алгоритмов для различных типов входных данных. Интересно, что алгоритм «Lin, Kernighan and Helsgaun heuristic» (LKH-3), входящий во все Парето оптимальные группы для смежной задачи (задача коммивояжера), в данном случае вошел только в четыре группы из пяти.

Ключевые слова: задача маршрутизации с ограничением по грузоподъемности; локально-оптимальные метаэвристики; LKH-3; алгоритм переменного перемещения: метод отжига; алгоритм управляемого поиска; алгоритм поиска с запретами

Для цитирования: Авдошин С.М., Береснева Е.Н. Локально-оптимальные метаэвристики для решения задачи маршрутизации с ограничением по грузоподъемности. Труды ИСП РАН, том 31, вып. 4, 2019 г., стр. 121-138. DOI: 10.15514/ISPRAS-2019-31(4)-8

1. Introduction

The Vehicle Routing Problem (VRP) is one of the most widely known challenges in a class of combinatorial optimization problems. VRP is directly related to Logistics transportation problem and it is meant to be a generalization of the Travelling Salesman Problem (TSP). In contrast to TSP, VRP produces solutions containing some (usually, more than one) looped cycles, which are started and finished at the same point called a "depot". As in TSP, each customer must be visited only by one vehicle. The objective is to minimize the cost (time or distance) of all tours. Despite the fact that both problems belong to the class of NP-hard tasks, VRP has higher solving complexity than TSP for the identical types of input data.

This work is aimed at analysis of VRP subcase, which is called Capacitated Vehicle Routing Problem (CVRP), where the vehicles have a limited capacity. A new constraint is that the total sum of demands in a tour for any vehicle must not exceed its capacity. In the paper we will use CVRP abbreviation having in mind the mathematical formulation that was described in our previous work [1].

There are three types of algorithms that are used to solve any subcase of CVRP: exact algorithms, constructive heuristics and metaheuristics.

- *Exact algorithms.* These algorithms find an optimal solution but take a great time for solving large instances. State-of-the-art exact methods can provide optimal solution for some SCVRP instances with up to 100 nodes, but it takes 30-40 minutes at average [2]. Due to these restrictions, researchers all over the world concentrate on heuristic methods.
- *Constructive heuristics.* They build an approximate solution iteratively, but they do not include further improvement stage. These heuristics are usually used for generation of an initial solution for improvement algorithms. A lot of experiments show that classical heuristics work much faster than to exact methods. For example, an instance of 100-150 nodes can be solved up to a few (1-2) seconds [2].
- *Metaheuristics*. These algorithms take as input some approximate initial solution and try to iteratively improve it. According to [3], metaheuristics are divided into two groups: metaheuristics based on local search and metaheuristics based on population, or natural inspired ones. The first group look for new solutions by moving at each iteration from a current state to another state in its neighborhood, while the second group works on a basis of a population of solutions which may be combined together in the hope of generating better ones, like in nature. Certain limitations are inevitable in any research, hence in this work we concentrate only on the first group of metaheuristics, because one of the most perspective algorithms for TSP – LKH-3 – belongs to this group, and it is very interesting for us to compare it with its "closest" alternatives.

Capacitated vehicle routing problems CVRP form the core of logistics planning and are hence of great practical and theoretical interest. There is no doubt that actuality of research and development of heuristics algorithms for solving CVRP is on its top, because in a real word there can be up to one thousand clients in a delivery net, that is why it is especially important to explore heuristic algorithms that allow to quickly generate near optimal solution in a polynomial time.

There are a lot of articles related to CVRP local search metaheuristics, but no works were found which compare improvement heuristics using the same input data of different types and sizes. We will compare these algorithms under criteria of quality, or error rate, and running time. Under the error rate we mean the percentage of difference in the obtained value of the solution with the optimal (or best-known) solution for the problem.

The aim of this work is to make a comparison of best-known local search metaheuristics by criteria of solution quality and running time with CWS or NN as initial algorithms as there were not found any such state-of-the-art comparative study. In addition, it is important to define sets of Pareto optimal metaheuristics for different types of input data.

The paper is structured as follows. In the second part, a general local search approach is described. After that, in the third section, some notes on most popular local search metaheuristics are provided, including short description of chosen algorithms to be intercompared. The fourth part presents design of experiments on local search metaheuristics. The fifth and the sixth sections describe results of solution qualities and computing times of algorithms, respectively. The seventh part consists of definition of Pareto optimal metaheuristics and five such sets are presented. In the last part we summarize our findings and suggest areas for future work.

2. Local search approach

Local search algorithms take as input some approximate initial solution and try to iteratively upgrade it with local improvements. These changes can either improve a single route (intra-route optimization) or change more than one routes simultaneously in such a way that the overall solution is improved (inter-route optimization).

Intra-route and inter-route optimization strategies consist of different schemes, which are fully described in [5]. In this research we will use the most-known and simplest but still effective local improvement heuristics: 1-point and 2-point moves, 2-opt.

The set of all solutions that can be obtained by applying the local improvements on a solution s is called the neighborhood $N(s)$. Of course, the bigger neighborhood is, the more likely it contains a new solution that can improve current one. However, to have large neighborhoods means to have inevitably higher computational complexity since more solutions need to be generated and evaluated [6]. At the same time, local search methods must deal with the problem of being stuck in a local optimum. Thus, a lot of methods to escaping the local optimum are applied, they will be described in the next section.

So, basically, local search approach consists of the following main steps.

1) Taking as input some initial solution s.
2) Generation of a neighborhood$N(s)$.
3) Selection of the best solution s^* from $N(s)$ using some acceptance criteria.
4) Make a new s equal to s^*.
5) Checking for exceeding different limits. If stop criteria is satisfied then terminate, otherwise continue with the step 2.

3. CVRP local search metaheuristics

Local search metaheuristics are used to solve a wide range of combinatorial optimization problems. Among heuristic methods for solving TSP there is one, which is the best – it is local search

metaheuristic, proposed by Lin, Kernighan and Helsgaun [7]. Also, local search algorithms are key part of most known methods for solving most subcases of VRP [3] [8] [9] [10] etc.

The most well-known schemes for solving CVRP that include local search steps are different variants of tabu search, forms of deterministic and simulated annealing, variable neighborhood search, guided local search [11]. Recently a new adoption of LKH for TSP called LKH-3 was proposed by one of the original authors, Helsgaun [12] . Also, in a recent study it was stated that improved version of record-to-record travel heuristic analyzed [13] is "… a well-performing metaheuristic", which combines strategies of deterministic annealing, tabu search, variable neighborhood search and both intra-route and inter-route optimizations described later. It was proposed by Li and others [14].

Of course, there are a lot of other metaheuristics for finding CVRP solution, however it was decided to concentrate on a set of several reputed local search algorithms that were honorably mentioned in recent studies.

As it was stated earlier, for all metaheuristics an initial solution must be obtained. For most input problems Clarke and Wright Savings (CWS) heuristic [14] is used, which is the best among construction algorithms except for a few instances with geometric type of clients' distribution. For these especial input files Nearest Neighbor (NN) heuristic is applied instead of CWS.

Thus, in this study we will intercompare following local search metaheuristics for solving classical CVRP.

3.1 A set of optimization operators (OPT)

As it was stated above, in this research the most-known and simplest local improvement heuristics are used. They are 1-point move, 2-point move and 2-opt.

In a tour of N vertices 1-point move operator (or relocate heuristic) moves some vertex $v_i, i \in N$ after another vertex $v_j, j \in N, i \neq j$ at the same tour. Another 2-point move operator (or exchange heuristic) swaps locations of two different vertices $v_i, i \in N$ and $v_j, j \in N, i \neq j$. And the main idea of 2-opt heuristic is to remove two edges $(v_i, v_j), i, j \in N$ and $(v_x, v_y), x, y \in N$ from the solution and replace them with two new edges (v_i, v_x) and (v_j, v_y). It is important to note that all mentioned operators can be applied for each of intra-optimization and inter-optimization as it is shown in Fig. 1.

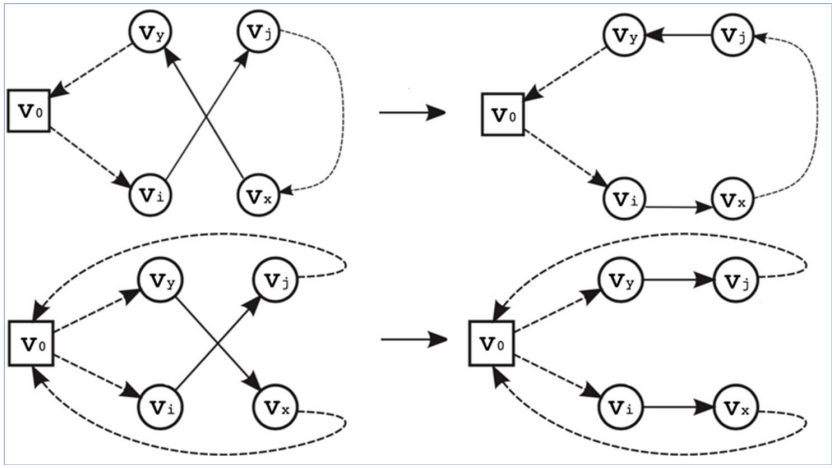

Fig. 1. Intra-route 2-opt (above) in compare to intre-route 2-opt (below).

3.2 Guided local search (GLS)

The guided local search metaheuristic is used to avoid local minima. It was initially proposed by [15] and later applied to CVRP by [16]. According to [17], this method is memory-based as it determines and penalizes "ineffective" edges by increasing its cost to a new $c^*(v_i, v_j) = c(v_i, v_j) + \lambda p(v_i, v_j)L$, where $p(v_i, v_j)$ counts the number of penalties of edge (v_i, v_j), L is a proxy for the average cost of an edge, computed as the costs of the starting solution divided by the number of customers, and λ controls the impact of penalties (authors suggest to always set λ = 0.1).

3.3 Tabu search (TS)

Tabu search originally was proposed by Glover and others [18]. The main idea of TS is as follows. If some solutions in $N(s)$ cannot be improved for several iterations or they violate the rules, they are made forbidden (or tabu) in order to prevent being in a stack of local minimum. Such forbidden solutions are put into a tabu-list, so this heuristic is also memory-based like GLS. The duration that a solution remains tabu is called its tabu-tenure and it can vary over different intervals of time.

We use recent TS algorithm that provides good results described in [19] as it was stated in [5].

3.4 Simulated annealing (SA)

This classical algorithm was developed in 1983 by [20]. It is based on an analogy from the annealing process of solids, where a solid is heated to a high temperature and gradually cooled in order for it to crystallize in a low energy configuration [21]. In this research we used adopted version of SA for CVRP by [22]. In SA some solution s^* from $N(s)$ at iteration i is chosen to be a new $s = \begin{cases} s^* \text{ with probability } P(s, s^*, i) \\ s \text{ with probability } 1 - P(s, s^*, i) \end{cases}$ accordingly to the probabilistic function $P(s, s^*, i) = \exp\left(-\frac{f(s) - f(s^*)}{Q_i}\right)$, where $f(s)$ is a length of solution s, Q_i is an element of an arbitrary decreasing, converging to zero, positive sequence, which specifies an analogue of the falling temperature in the crystal.

3.5 Lin-Kernighan-Helsgaun heuristic for CVRP (LKH-3)

LKH-3 is proposed by Helsgaun in 2017 [12]. The implementation of LKH-3 builds on the idea of transforming the problem into classical symmetric TSP. After that algorithm uses the principle of 2-opt algorithm and generalizes it. In this heuristic, the k-Opt, where $k = \overline{2..\sqrt{N}}$, is applied, so the switches of two or more edges are made in order to improve the tour. This method is adaptive, so decision about how many edges should be replaced is taken at each step [7] [23].

This algorithm was not developed by us as original source code of LKH-3 is free of charge for academic and non-commercial use and can be downloaded at [24].

3.6 Variable record-to-record travel heuristic (VRTR)

Li and others suggested a variable record-to-record travel heuristic, which is based on classical record-to-record travel algorithm (RTR). RTR combines approaches of deterministic annealing (which is a variant of simulated annealing heuristic) and tabu search. The main differences between VRTR and RTR are as follows. Firstly, VRTR considers 1-point, 2-point and 2-opt moves not only within individual routes as RTR does, but also between them. Secondly, "VRTR uses a variable-length neighbor list that should help focus the algorithm on promising moves and speed up the search procedure" [14].

4. Design of experiments

All algorithms from section III were implemented as sequential algorithms in C/C++, no multi-threading was explicitly utilized. They were executed on an Intel Core i5 clocked at 1.3GHz with 4 GB RAM running the macOS 10.14.3 operating system.

The computational testing of the solution methods for CVRP has been carried out by considering eight sets of test instances from the next well-known database [25]. Total number of instances in sets A, B, E, F, G, M, P, X is 211. All instances inside one set have its own characteristics and a way of generation: cluster-based / uniform / geometric distribution of clients, real-world / imitative cases etc. The integer Euclidean metric is used for all instances. The naming scheme and data format for each instance is described here [26]. Shortly, the first letter in names shows the name of used set, the figure after letter 'n' shows the number of nodes and the figure which stands after letter 'k' presents the number of vehicles.

Experiment starts with choice of a local search metaheuristic M from set {OPT, GLS, SA, TS, LKH-3, VRTR}. After one dataset D is selected from a list of all mentioned benchmark datasets, an instance file F from chosen dataset D is taken. Next, the following steps are repeated 51 times on instance F: chosen metaheuristic M is executed on a basis of initial solution obtained by CWS or NN (as it was explained in a previous chapter). During all iterations, except the first one, solution qualities $\varepsilon_{it}(M, F)$ and computing times $t_{it}(M, F)$ (in seconds) are calculated for algorithm M on test F. The first run is not taken into account in calculations because of specifity of C++ compiler. Solution quality (or percent above best-known, or gap) is calculated using the next formula [11]:

$$\frac{F(S^0) - F_{opt}(S)}{F_{opt}(S)} \cdot 100\%,$$

where $F(S^0)$ is a length of obtained solution and $F_{opt}(S)$ is a length of optimal solution or best-known one.

Also we calculate minimal value $\varepsilon_{\min}(M, F)$ among all figures $\varepsilon_{it}(M, F)$ and sample mean $\bar{X}_t(M, F) = \frac{1}{50}\sum_{it=1}^{50} t_{it}(M, F)$ among all figures $t_{it}(M, F)$. And finally, among all $\varepsilon_{\min}(M, F)$ from one dataset average sample mean $\bar{X}_\varepsilon(M, D) = \frac{1}{|D|}\sum \varepsilon_{\min}(M, F)$, $\forall F \in D$ is calculated, which shows average gap for algorithm M on dataset D. And among all $\bar{X}_t(M, F)$ from one dataset average sample mean $\bar{X}_t(M, D) = \frac{1}{|D|}\sum \bar{X}_t(M, F)$, $\forall F \in D$ is calculated, which shows average computing time of algorithm M on dataset D. $|D|$ is a number of input files in dataset D.

The plan of experiment on local search metaheuristics is described in fig. 2.

```
    Input: local search metaheuristics, datasets
1:  foreach local search metaheuristic M
2:      foreach dataset D from datasets
3:          foreach instance file F from D
4:              for it ∈ {0…50} // number of runs
5:                  init_sol = run CWS or NN on F
6:                  final_sol = run M on F with init_sol
7:                  if (it != 0) // if not the first run
8:                      calculate ε_it(M, F), t_it(M, F)
9:              calculate ε_min(M, F)
10:         calculate t_min(M, F)
11:         calculate t_max(M, F)
12:         calculate X̄_t(M, F)
13:         calculate s_t(M, F)
14:     calculate X̄_ε(M, D) // average gap on dataset
15:     calculate X̄_t(M, D) // average computing time
```

Fig. 2. Plan of experiment on constructive heuristics

Each metaheuristic is subsequently launched on all instances from every mentioned dataset, so no input file is missed.

5. Computational results on solution quality

Results about the best (= minimal) solution qualities $\varepsilon_{\min}(M,F)$ of metaheuristics available from experiments for set A are presented in fig. 3. The horizontal axis represents the name of instance data. The vertical axis shows the solution quality.

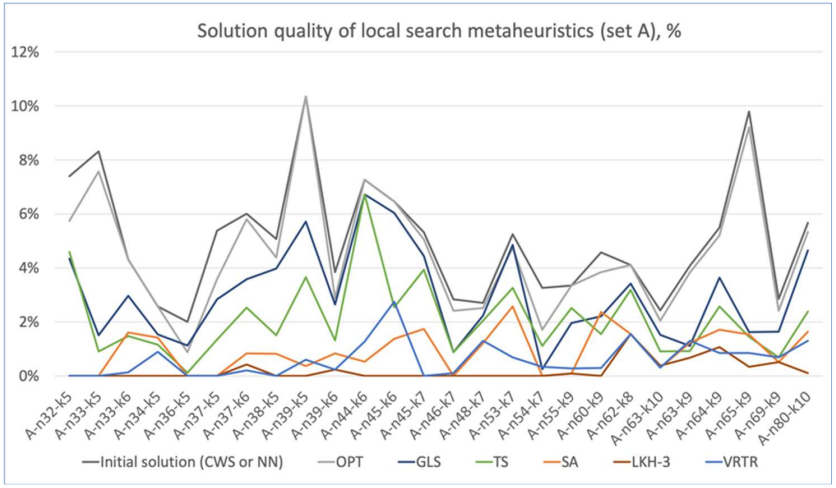

Fig. 3. Solution quality of local search metaheuristics, set A

Results for other sets cannot be presented here as detailed as for set A, because of its size, but they are aggregated in Table 1, where average gaps $\bar{X}_{\varepsilon}(M,D)$ for all metaheuristics and all datasets are given.

Table 1. Average gap $\bar{X}_{\varepsilon}(M,D)$ in the dataset, %.

Average gap $\bar{X}_{\varepsilon}(M,D)$ in the dataset		Local search metaheuristics						
		CWS or NN	OPT	GLS	TS	SA	LKH-3	VRTR
Set (its size)	A (26)	5,0%	4,5%	3,0%	2,1%	0,9%	0,2%	0,6%
	B (23)	4,4%	3,9%	3,1%	2,5%	1,0%	0,2%	0,6%
	E (11)	7,1%	5,3%	4,1%	3,6%	0,8%	0,4%	0,8%
	F (3)	4,4%	2,7%	3,3%	3,0%	1,8%	0,1%	1,9%
	G (20)	11%	10%	8,1%	9,7%	5,2%	2,1%	2,4%
	M (4)	4,7%	2,8%	2,6%	2,1%	0,5%	0,2%	0,5%
	P (24)	8,0%	6,6%	3,5%	4,5%	0,7%	0,5%	0,9%
	X(100)	5,9%	5,4%	4,9%	4,0%	4,1%	2,0%	1,7%

Fig. 4 is a visual representation of this table. These general figures can show an approximate overall effectiveness of algorithms by criterion of solution quality. Analysis of fig. 4 indicated a group of top-3 algorithms by criterion of solution quality: they are LKH-3, VRTR, SA. Let's take a closer look at their results.

It is clearly seen that in 7 out of 8 sets LKH-3 produces solutions with the least (= the best) solution qualities. Experiment results that are not shown here because of their large size reveal that LKH-3 produces not the best solutions in:

- 3 input files out of 26 for set A (≈12%);

- 1 input files out of 23 for set B (≈4%);
- 2 input files out of 11 for set E (≈18%);
- 1 input files out of 3 for set F (≈33%);
- 6 input files out of 20 for set G (30%);
- 0 input files out of 4 for set M (0%);
- 6 input files out of 24 for set P (≈25%);
- 61 input files out of 100 for set X (61%).

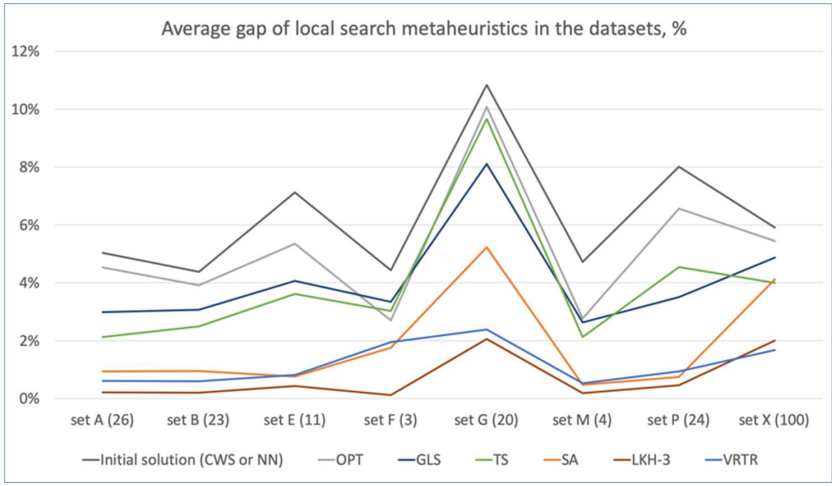

Fig. 4. Average gap of local search metaheuristics in the datasets (%)

On total, LKH-3 was not the best in quality criterion for 80 input files out of 211 (≈38%). Only 6 times SA was the best, while all other times VRTR was «the winner».

It is important to mention that LKH-3 tends to produce best solution for instances with no more than ≈100 clients in a delivery net regardless of type of distribution in the dataset. There are 86 instances with less than 102 clients, and solutions obtained using LKH-3 are the best in 86% (in 74 files).

In addition, it should be noted that LKH-3 is the best for input problems with cluster-based distribution of clients, when the number of clusters is a bit smaller than the amount of available vehicles (sets B and M). On the contrary, this algorithm is not the best for 61% of files from set X that consists of very different instances. Despite the fact that there are several files with cluster-based distribution, the amount of clusters is much less than the number of vehicles, and, as experiments showed, LKH-3 does not suits well for such cases.

VRTR nearly always takes «the second place in this race» except for set X. It can be noticed that VRTR works in a best way for instances with more than ≈320 clients in a delivery net for non-geometric distributions. There are 53 instances with more than 321 clients in set X, and solutions obtained using VRTR are the best in 89% (in 47 files). Nevertheless, if we take a range of ≈ [100; 320] clients, results show that either VRTR or LKH-3 are the best in nearly 50% of cases, so for this diapason both these algorithms can be admitted being equal.

For most of input files SA produces solutions which are nearly equal to other ones generated by VRTR. However, the situation is different for sets G and X, where SA is always worse than its closest "competitor". So, we can come to the conclusion that with SA it is better to use non-geometric input data with no more than 100 clients in a delivery net. Nevertheless, only 6 times out of 211 SA is better than LKH-3 – this fact shows superiority of LKH-3 over SA.

Other three local search metaheuristics – TS, GLS and OPT (listed in increasing size of average gaps) – were nearly always worse than top-3 group.

TS was better than LKH-3 only 3 times out of 211, better than VRTR or SA in 6 input files out of 26 from set A, in 2 input files out of 23 from set B, in 1 input file out of 24 from set P. Also, TS is slightly more effective than SA for set X, however, the difference in solution qualities between LKH-3 and TS is significant in general.

Roughly speaking, GLS is usually worse than TS, except for sets G and P. Also, GLS is better than LKH-3 only in one file, thus, it cannot compete with the algorithms from top-3 group seriously.

And the last one and the least effective by criterion of solution quality metaheuristic is OPT. It nearly always produces solutions which are better than ones obtained by simple initial algorithm but worse than other local search metaheuristics.

6. Experimental results on computing time

Results about average running times $\bar{X}_t(\mathrm{M},\mathrm{F})$ of metaheuristics available from experiments for set A are presented in fig. 5. The horizontal axis represents the name of instance data. The vertical axis shows the running time in seconds.

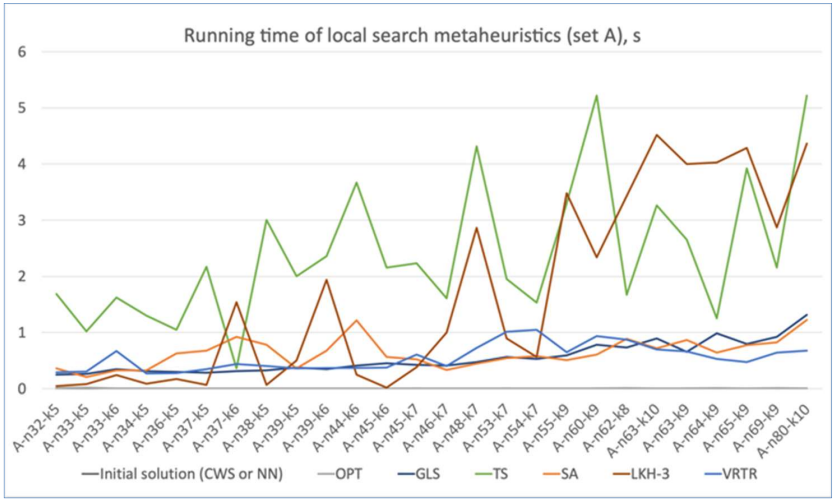

Fig. 5. Running time of local search metaheuristics, set A

Results for other sets cannot be presented here as detailed as for set A, because of its size, but they are aggregated in Table 2, where average computing times $\bar{X}_t(\mathrm{M},\mathrm{D})$ for all metaheuristics and all datasets are given.

Table 2. Average computing time $\bar{X}_t(M,D)$ in the dataset (in seconds)

Average gap $\bar{X}_\varepsilon(\mathrm{M},\mathrm{D})$ in the dataset		Local search metaheuristics						
		CWS or NN	OPT	GLS	TS	SA	LKH-3	VRTR
Set (its size)	A (26)	0,0003	0,004	0,540	2,411	0,635	1,692	0,554
	B (23)	0,0006	0,027	0,291	2,578	0,505	1,487	0,611
	E (11)	0,0008	0,016	0,311	2,172	0,857	1,930	0,851
	F (3)	0,0003	0,024	0,953	4,240	1,621	2,154	1,270
	G (20)	0,0468	0,569	7,979	16,34	15,78	8,563	9,748
	M (4)	0,0025	0,044	0,883	4,386	4,046	2,370	2,469
	P (24)	0,0008	0,023	0,367	2,216	0,566	2,166	0,554
	X (100)	0,041	0,62	7,088	65,15	56,74	37,41	9,998

Fig. 6 is a visual representation of this table. These general figures can show an approximate overall effectiveness of algorithms by criterion of running time.

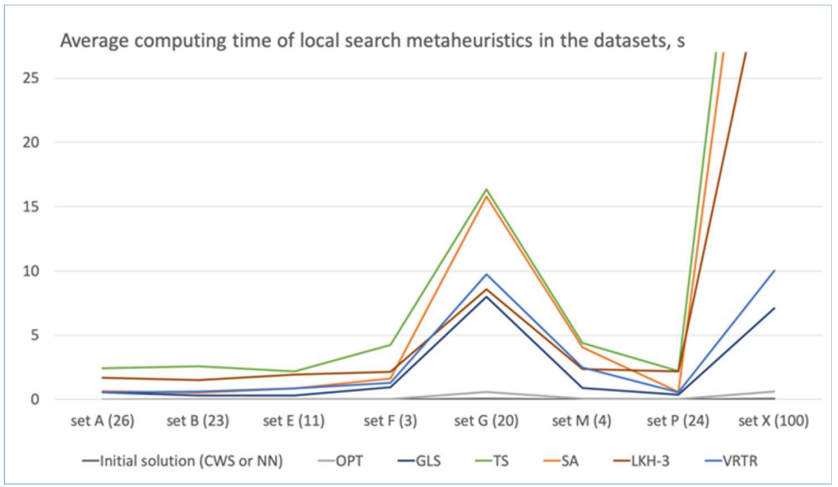

Fig. 6. Average computing time of local search metaheuristics in the datasets

It can be seen from Table 2 and fig. 4 that metaheuristics, which are listed in increasing size of average running times, are as follows: OPT, GLS, VRTR, SA, LKH-3, TS. Let us discuss their results in this order.

Analysis reveals that the «fastest» local search metaheuristic for all 211 instances is OPT as it was expected. Its running time is approximately 25 times bigger than computing time of initial algorithm, however, even for 1000 clients in a delivery net the maximum running time does not exceed 3 seconds.

Next algorithm is GLS, which is approximately 300 times slower than initial one. In sets B, E, F, P, M and X it nearly always (94%) ranks #2 just after OPT. Of course, there are cases when GLS works slightly slower than others, but the number of such situations is not very significant and the difference between obtained values does not exceed 1 second. However, in sets A and G GLS works with mixed results: computing times of GLS, SA, VRTR or LKH-3 (depending on dataset) are fluctuated and too close to each other, so it is impossible to find a leader.

Average computing time of VRTR steadily goes at the third plays, except for set G with geometric instances and several rare cases from other datasets. VRTR has smooth growth of speed, and no special aspects of its work are found apart from not very stable work with geometric-inspired instances.

Next one is SA. In comparison with VRTR, SA has bigger growth of speed and the plot of its running time is more fluctuated. In sets A, B, E, F and P this algorithm executes quicker than LKH-3 but slower than VRTR. However, in sets G, M and X it shows much worse effectivity, when the number of clients in a delivery net becomes more than 100. It means that SA is better to use for instances with up to one hundred delivery points.

LKH-3 is slower than other mentioned metaheuristics (except for TS) for all datasets apart from sets G, M and those instances from set X with 322 and more delivery points. The main unique feature of LKH-3 is its variability. Linear chart of running times of LKH-3 has a lot of drastic jumps and slumps. That is why this metaheuristic has not very positive computing time rate. Nevertheless, LKH-3 can work very quickly, especially when there are no more than 50-80 clients in a net.

Last one metaheuristic to be discussed concerning its computing time is TS. As LKH-3, linear chart of running times has a lot of drastic jumps and slumps. In average, this is the slowest algorithm in this group, however, in a third of cases it can compete with LKH-3 or SA but not very significant speeding up can be noticed.

7. Pareto optimal local search metaheuristics

The algorithm $m_0 \in \mathcal{M}$ is Pareto optimal if $(\forall m \in \mathcal{M})\left((m \neq m_0) \Rightarrow \left(\bar{X}_\varepsilon(m,D) > \bar{X}_\varepsilon(m_0,D)\right) \vee \left(\bar{X}_t(m,D) > \bar{X}_t(m_0,D)\right)\right)$. Thus, our aim is to find a sets of Pareto optimal algorithms for different types of input data.

Figures from 7 to 16 are plotted using values from Table 1 and Table 2. The horizontal axis represents average computing time $\bar{X}_t(\mathrm{M},\mathrm{D})$ in seconds. The vertical axis shows average solution quality $\bar{X}_\varepsilon(\mathrm{M},\mathrm{D})$.

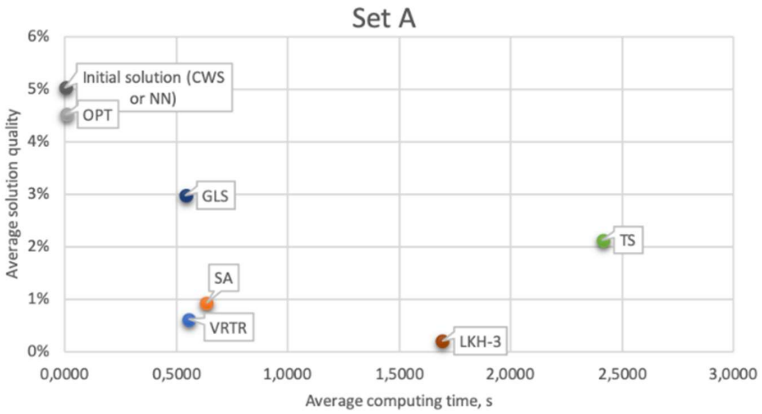

Fig. 7. Average solution quality vs. average running time, set A

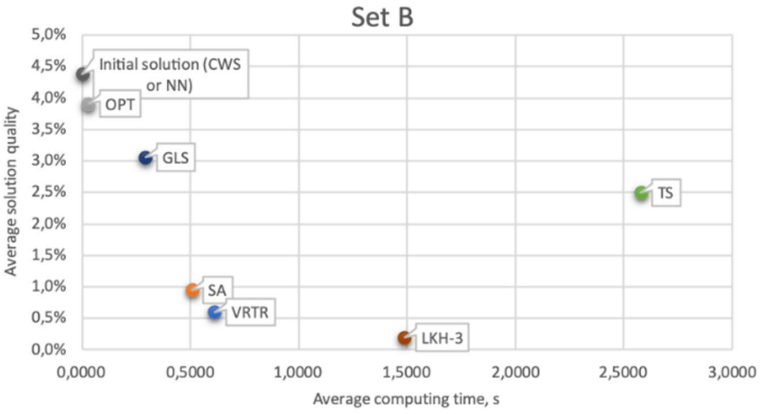

Fig. 8. Average solution quality vs. average running time, set B

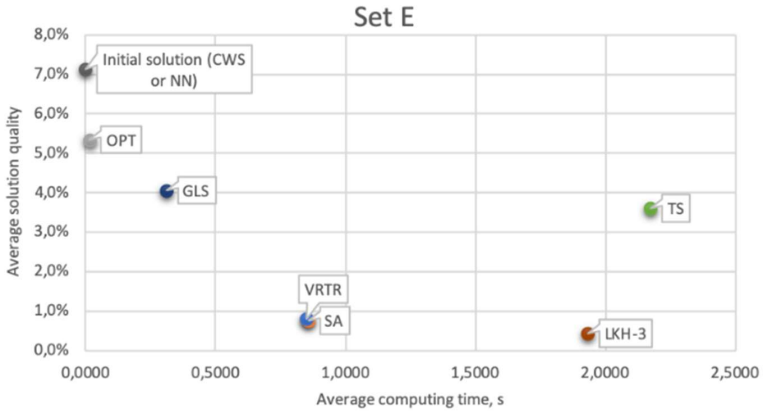

Fig. 9. Average solution quality vs. average running time, set E

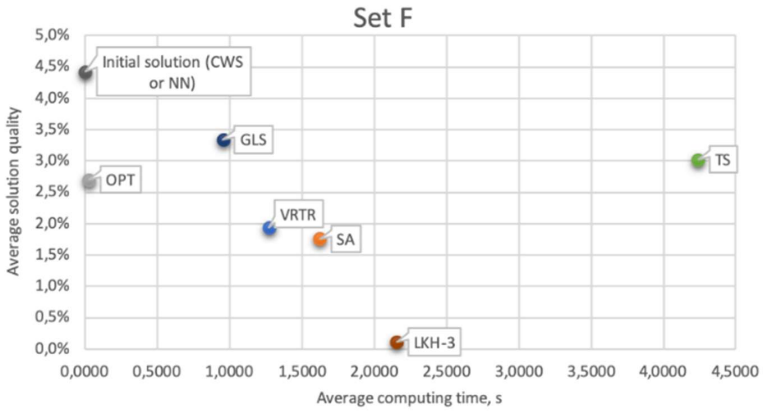

Fig. 10. Average solution quality vs. average running time, set F

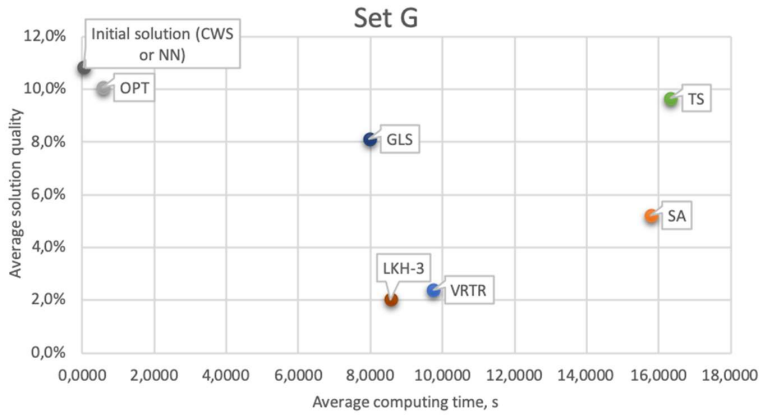

Fig. 11. Average solution quality vs. average running time, set G

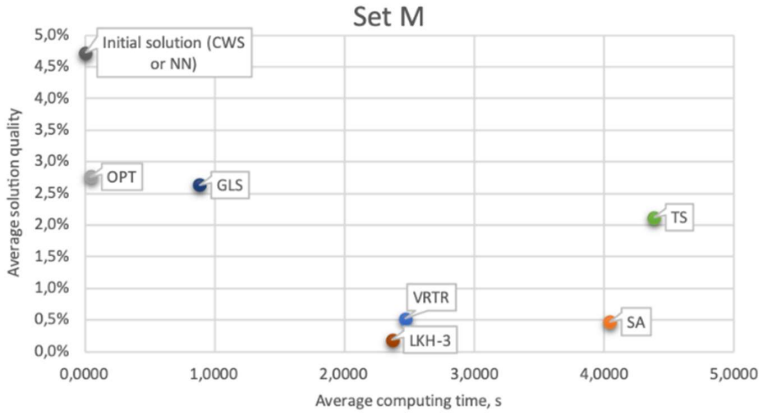

Fig. 12. Average solution quality vs. average running time, set M

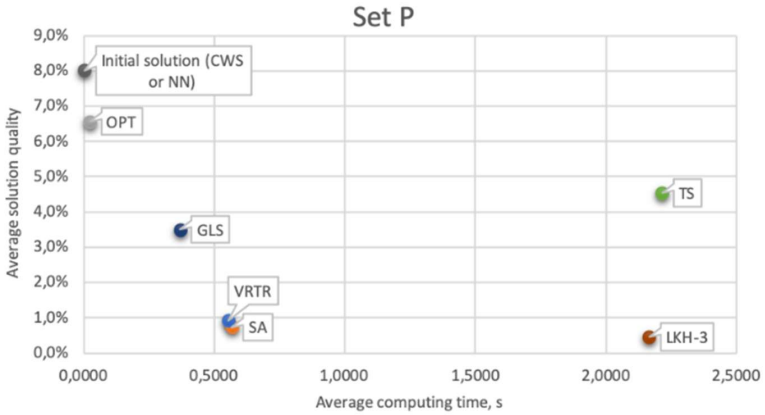

Fig. 13. Average solution quality vs. average running time, set P

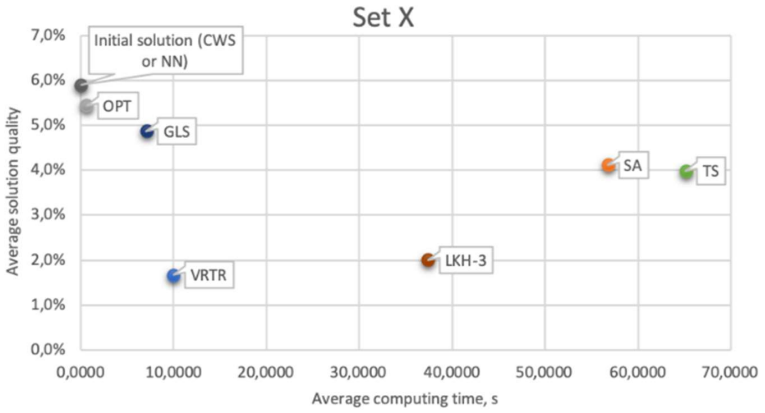

Fig. 14. Average solution quality vs. average running time, set X

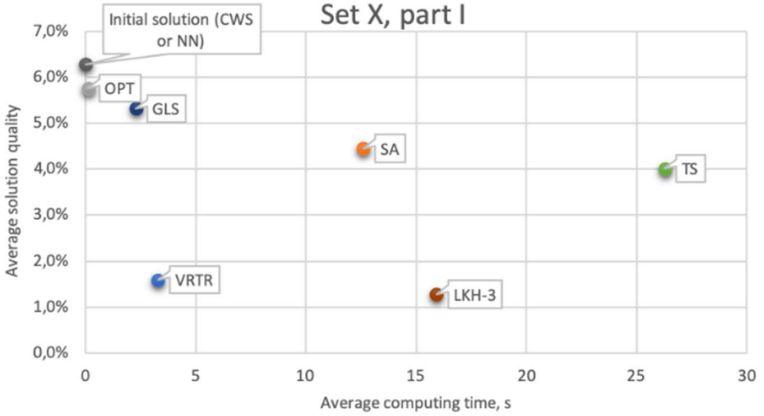

Fig. 15. Average solution quality vs. average running time, set X, part I

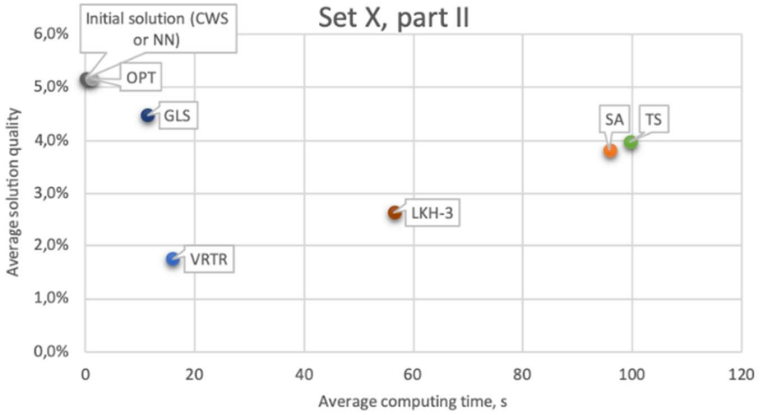

Fig. 16. Average solution quality vs. average running time, set X, part II

To sum up information presented in Figures from 7 to 16, Table 3 is formed.

Table 3. Involvement of algorithms in Pareto optimal groups for different datasets

Involvement of algorithms in Pareto optimal groups		Local search metaheuristics					
		OPT	GLS	TS	SA	LKH-3	VRTR
Set (its size)	A (26)	✓	✓		✓	✓	✓
	B (23)	✓	✓		✓	✓	✓
	E (11)	✓	✓		✓	✓	✓
	F (3)	✓			✓	✓	✓
	G (20)	✓	✓			✓	
	M (4)	✓	✓			✓	
	P (24)	✓	✓		✓	✓	✓
	X (100)	✓	✓				✓
	X, part I (47)	✓	✓			✓	✓
	X, part II (53)	✓	✓				✓

Table 3 shows involvement of local search metaheuristics in Pareto optimal groups for different datasets. Algorithms that belong to a group of Pareto optimal heuristics for some set are marked with a big tick. SA is marked by a small tick for set A as this metaheuristic can be in optimal by Pareto group as the difference between it and VRTR is too close, so it can be neglected. Also, it should be mentioned that two more rows are added – they are "set X, part I" which consists of instances with up to 322 clients and "set X, part II" which is vice versa has instances with 322 and more delivery points inside input files. This division is connected with the big size of set X and with the fact that was described in section V – VRTR works in a best way for instances with more than ≈320 clients in a delivery net for non-geometric distributions but LKH-3 produces good results for instances with no more than ≈100 clients regardless of type of distribution in the dataset.

The following conclusions are made on a base of results from Table 3.

1) Sets A, B, E and P can be aggregated together because a group of Pareto optimal algorithms is the same for all these sets. We will name this aggregation as *Group_1*. It represents two different types of inputs. The first one is with clients' coordinates and demands that are formed from a uniform distribution with some outlying cases. The second one is with nodes that are formed into clusters, and the number of clusters is equal or greater than number of available vehicles. Demands are also formed from a uniform distribution with some outlying cases. All input files from *Group_1* have 101 clients in a delivery net as maximum.
2) Set F forms a second group *Group_2* with only 3 instances obtained from real goods deliveries. Number of delivery points varies from 45 to 135.
3) *Group_3* is formed of sets G and M that also have the same Pareto optimal metaheuristics. Number of delivery points varies from 100 to 483. Set G has instances with locations in a form of concentric squares, pointed stars and rays, while set M consists of only 4 input files with locations that are grouped into clusters, and the number of clusters is equal or smaller by 1 than number of available vehicles.
4) *Group_4* is formed of the first part of set X. There are 47 instances in it, and the number of instances is up to 322 clients. *Group_4* is a mix of input data types: it has different combinations of demand distribution (unitary demands, small/large values, small/large variance), depot positioning (central, eccentric, random) and customer positioning (practical cases, uniform distribution, cluster-based).
5) *Group_5* is formed of the second part of set X. There are 53 instances in it, and the number of instances is from 322 to 1001 clients. Other characteristics of this group are the same as *Group_4* has.

Above-mentioned conclusions are outlined in Table 4 with information about involvement of algorithms in Pareto optimal groups depending on types of input data. All algorithms are listed in increasing order of average computing times (from best to worst) and in decreasing order of average solution qualities (from worst to best).

8. Conclusion

Overall, the next recommendation should be given to the problem which has described variant of mathematical model of CVRP. In general, for all types of clients' distribution the best algorithm to be applied is Clarke and Wright Savings, however, in case of having input data in form of concentric rays (like in Fig. 6) it is better to use Nearest Neighbor algorithm. Also, a few instances were solved best of all by Clarke and Wright Savings 2 algorithm, so it is important to have this algorithm in mind, however the difference between it and CWS is not very significant (no more than 1%).

One more conclusion is that it is unreasonable to use Sweep heuristic as it is not able to construct a set of routes without exceeding the number of vehicles for more than 50% of input files.

Finally, for our research it means that for all instances, except those 8 from set G, CWS heuristic will be used as initial algorithm for metaheuristic, otherwise – we will apply NN.

Table 4. Involvement of algorithms in Pareto optimal groups depending on types of input data

	Number of delivery points	**Distribution of delivery points**	**Distribution of demands**	**Pareto optimal algorithms**
Group_1	Up to 101	1. Uniform (with some outlying cases). 2. Cluster-based, the number of clusters is equal or greater than number of available vehicles.	Uniform (with some outlying cases)	OPT GLS SA VRTR LKH-3
Group_2	Up to 135	Real-world	Real-world	OPT SA VRTR LKH-3
Group_3	From 100 to 483	1. Geometric (concentric squares, pointed stars and rays). 2. Cluster-based, the number of clusters is equal or smaller by 1 than number of available vehicles.	Constant or uniform	OPT GLS LKH-3
Group_4	From 100 to 322	Mixed	Mixed	OPT GLS VRTR LKH-3
Group_5	From 322 to 1001	Mixed	Mixed	OPT GLS VRTR

As it was expected, unfortunately, there is no one universal metaheuristic that takes the first places by both criteria of solution quality and running time. Overall, the next recommendations should be given to people who are interested in metaheuristics solving CVRP.

1) For uniform (with some outlying cases) or cluster-based distribution of clients' locations, where the number of clusters is equal or greater than number of available vehicles it is better to apply Set of optimization operators, Guided local search, Simulated annealing, Variable record-to-record travel algorithm or Lin-Kernighan-Helsgaun heuristic for CVRP depending on desired solution quality and available time for calculations. Here and elsewhere the algorithms are listed in increasing order of average computing times (from best to worst) and in decreasing order of average solution qualities (from worst to best).
2) For real-world instances it is better to use following local search metaheuristics: Set of optimization operators, Simulated annealing, Variable record-to-record travel algorithm or Lin-Kernighan-Helsgaun heuristic for CVRP.
3) For geometric (with concentric squares, pointed stars and rays) or cluster-based distribution of clients' locations, where the number of clusters is equal or smaller by 1 than number of available vehicles, it is better to apply Set of optimization operators, Guided local search or Lin-Kernighan-Helsgaun heuristic for CVRP.
4) Finally, for mixed up combinations of demand distribution, depot positioning and customer positioning there are two recommendations. If there up to approximately 350 clients in a delivery net, it is better to use Set of optimization operators, Guided local search algorithm, Variable record-to-record travel algorithm or Lin-Kernighan-Helsgaun heuristic for CVRP. Otherwise, if there are nearly 320 delivery point and more then LKH-3 stops being so effective, so it is better to use following local search metaheuristics: Set of optimization operators, Guided local search algorithm or Variable record-to-record travelling algorithm.

Also experiments revealed absolute inefficiency of Tabu search as it is not in any of Pareto optimal groups.

In future we are planning to extend our study by conducting experiments using:

1) Population-based or nature-inspired metaheuristics as there are a lot of productive algorithms, too.

2) Other input data sets, including the most recent one with thousands of nodes in each instance. It is important to check local search metaheuristics on problems with extra-large dimensions to analyze their effectiveness.

References

[1]. E. Beresneva and S. Avdoshin. Analysis of Mathematical Formulations of Capacitated Vehicle Routing Problem and Methods for their Solution. Trudy ISP RAN/Proc. ISP RAS, vol. 30, no. 3, 2018, pp. 233-250. DOI: 10.15514/ISPRAS-2018-30(3)-17.

[2]. K. Braekers, K. Ramaekers, and I. Nieuwenhuyse. The vehicle routing problem: State of the art classification and review. Computers and Industrial Engineering, vol. 99, 2016, pp. 300-313.

[3]. B. Golden, S. Raghavan, and E. Wasil. The vehicle routing problem: Latest advances and new challenges. New York: Springer, 2008, 591 p.

[4]. P. Toth and D. Vigo. Vehicle Routing Problems, Methods, and Applications. Philadelphia: SIAM, 2014, 481 p.

[5]. F. Arnold, M. Gendreau, and K. Sorensen. Efficiently solving very large-scale routing problems. Computers and Operations Research, vol. 107, 2019, pp. 32-42.

[6]. K. Helsgaun. An effective implementation of the Lin–Kernighan traveling salesman heuristic. European Journal of Operational Research, vol. 12, issue 1, 2000, pp. 106-130.

[7]. E. Zachariadis and C. Kiranoudis. An open vehicle routing problem metaheuristic for examining wide solution neighborhoods. Computers and Operations Research, vol. 37, no. 4, 2010, pp. 712-723.

[8]. E. Taillard, G. Laporte and M. Gendreau. Vehicle routeing with multiple use of vehicles. Journal of the Operational Research Society, vol. 47, no. 8, 1996, pp. 1065-1070.

[9]. S. Ropke and D. Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transportation Science, vol. 40, no. 4, 2006, p. 455–472.

[10]. P. Toth and D. Vigo. An overview of vehicle routing problems. In The Vehicle Routing Problem, SIAM, 2002, pp. 1-26.

[11]. K. Helsgaun. An Extension of the Lin-Kernighan-Helsgaun TSP Solver for Constrained Traveling Salesman and Vehicle Routing Problems. Technical Report, Roskilde University, 2017, 60 p.

[12]. P. Schittekat and K. Sorensen. Deconstructing record-to-record travel for the capacitated vehicle routing problem. Operational Research and Management Science Letters, vol. 1, no. 1, 2018, pp. 17-27.

[13]. F. Li, B. Golden, and E. Wasil. Very large-scale vehicle routing: new test problems, algorithms, and results. Computers and Operations Research, vol. 32, issue 5, 2005, p. 1165–1179.

[14]. G. Laporte and F. Demet. Classical Heuristics for the Capacitated VRP. In The Vehicle Routing Problem, SIAM, 2002, pp. 109-128.

[15]. C. Voudouris, E. Tsang, and A. Alsheddy. Guided local search. In Handbook of metaheuristics, Springer, 2010, pp. 321-361.

[16]. D. Mester and O. Braysy. Active-guided evolution strategies for large-scale capacitated vehicle routing problems. Computers and Operations Research, vol. 34, no. 10, 2007, pp. 2964–2975.

[17]. F. Arnold and K. Sorensen. Knowledge-guided local search for the Vehicle Routing Problem. Computers and Operations Research, vol. 105, 2019, pp. 32-46.

[18]. F. T. E. Glover. A User's Guide to Tabu Search. Operations Research, vol. 41, no. 1, 1993, pp. 1-28.

[19]. E. Zachariadis and C. Kiranoudis. A strategy for Reducing the Computational Complexity of Local Search-Based Methods for the Vehicle Routing Problem. Computers and Operations Research, vol. 37, 2010, pp. 2089–2105.

[20]. S. Kirkpatrick, C. Gelatt, and M. Vecchi. Optimization by simulated annealing. Science, vol. 220, no. 4598, 1983, pp. 671–680.

[21]. NEO. Simulated Annealing. [Online]. Available: http://neo.lcc.uma.es/vrp/solution-methods/metaheuristics/simulated-annealing/. Accessed 27.02.2019.

[22]. I. H. Osman. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. Annals of Operations Research, vol. 41, 1993, pp. 421-451.

[23]. S. Avdoshin and E. Beresneva. The Metric Travelling Salesman Problem: The Experiment on Pareto-optimal Algorithms. Trudy ISP RAN/Proc. ISP RAS, vol. 29, no. 4, pp. 123-138, 2017. DOI: 10.15514/ISPRAS-2017-29(4)-8.

[24]. K. Helsgaun. LKH-3. [Online]. Available: http://akira.ruc.dk/~keld/research/LKH-3/. Accessed 01.2019.

[25]. I. Xavier. CVRPLIB. [Online]. Available: http://vrp.atd-lab.inf.puc-rio.br/index.php/en/. Accessed 09 05 2018.

[26]. Heidelberg University. TSPLIB. [Online]. Available: https://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/. Accessed 09 05 2018.

Информация об авторах / Information about authors

Екатерина Николаевна БЕРЕСНЕВА – с 2017 года преподаватель департамента программной инженерии НИУ ВШЭ, с 2019 года – аспирант НИУ ВШЭ. Профессиональные интересы – дискретная математика, задача маршрутизации транспорта, задача коммивояжера.

Ekaterina Nikolaevna BERESNEVA – lecturer at the School of Software Engineering, Faculty of Computer Science, National Research University Higher School of Economics since 2017. Her research interests include discrete mathematics, the vehicle routing problem and the travelling salesman problem.

Сергей Михайлович АВДОШИН – профессор, руководитель департамента программной инженерии факультета компьютерных наук НИУ ВШЭ с 2005 года. Сфера научных интересов: разработка и анализ компьютерных алгоритмов, имитация и моделирование, параллельные и распределенные процессы, теневой интернет, технология блокчейн.

Sergey Mikchailovitch AVDOSHIN – Professor, Head of the School of Software Engineering at National Research University Higher School of Economics since 2005. Research interests are design and analysis of computer algorithms, simulation and modeling, parallel and distributed processing, deep Web, blockchain technology.