THE IMPACT OF PERTURBATION MECHANISMS ON THE OPERATION OF THE SWAP HEURISTIC

WOJCIECH MISZTAL [1]

Abstract

In this study, an attempt was made to assess the impact of the most popular perturbation movements (i.e. *Multiple-Swap*(2-2), *Multiple-Shift*(2-2) and *Multiple-K-Shift*(1)), as well as the number of their calls on the quality of solutions and the time in which *Swap*(2-1) heuristics returns them. For this purpose, the iterative local search algorithm (ILS) was triggered, in which *Swap*(2-1) heuristics has cooperated with a single perturbation mechanism. The number of perturbations was changed in the range from 1 to 30. Each time the time and the difference between the percentage improvement of the objective function value of the solution obtained utilizing the *Swap*(2-1) algorithm cooperating with the perturbation mechanism and this algorithm working alone was checked. Based on the results obtained, it was found that the overall level of improvement in the quality of the returned solution is similar when using all of the considered perturbation mechanisms (is in the range of 2.49% to 4.02%). It has been observed that increasing the number of initiated perturbations does not guarantee an improvement in the quality of the returned solution. Perturbation movements similar to the motion initiated by the local search algorithm do not significantly improve the solution (they only entail extending the duration of action).

The structure of the study has the following form. The Introduction chapter provides information on the Vehicle Routing Problem. The chapter Research methods contain a description of *ILS* and *Swap*(2-1) approaches and perturbation mechanisms considered. The last two chapters include the results of tests and conclusions.

Keywords: optimization; vehicle routing problem; local search; Swap (2-1); perturbation mechanisms

1. Introduction

Transport activities related to the obtaining of raw materials as well as the distribution of goods, as a rule, require servicing a large group of points often located far away from each other. This results in the need to travel long distances, which translates into high capital and time consumption of this type of projects, as well as their increased negative impact on the natural environment. This is highly unfavorable from the point of view of the profitability of the business, as well as taking into account the impact on the natural environment, and therefore requires certain rationalization measures. Limiting the capital intensity and the negative impact of transport on the natural environment can be achieved

[1] Department of Agricultural, Forestry and Transport Machines, University of Life Sciences in Lublin, 28 Głęboka Str., 20-612 Lublin, Polska, e-mail: wojciech.misztal@up.lublin.pl

in many ways, among which are actions aimed at reducing fuel consumption by reducing the construction weight of vehicles [13, 17], or the use of alternative drives [3, 9]. Significant benefits can also be achieved using mathematical optimization methods [25, 26, 30]. Additional extremely important advantages of this approach are: the fact that it is not associated with the need to incur practically any expenditure (except computational), and also that it has the ability to shorten the time of movement also in situations where this cannot be achieved by other methods, e.g. by using means of transport capable of traveling at a higher speed, which will have to adapt the speed to the applicable restrictions dictated by traffic regulations. It is important that different approaches are complementary to each other, and thus, in order to maximize the effect, they can and should be used together.

Among the issues of optimizing nature regarding aspects of transport functioning, problems from the *Vehicle Routing* group deserve attention. Representing a specific type of issue corresponding to the specificity of a large number of real-life problems [1, 6, 18]. The general form of the *Vehicle Routing Problem* (VRP) can be formulated as follows. In the base constituting the beginning and end of each route, K vehicles are available to be used to carry out the service (consisting in the delivery or collection of goods) of n differential in terms of the location customers. Travel routes should be planned so that each customer is visited (only once) and the goal function representing the total distance traveled (total cost or time associated with completing the task) has a minimum value [6, 18].

The mathematical model of this problem can be presented as follows [18]:

$$\min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij} \tag{1}$$

with restrictions

$$\sum_{i \in V} x_{ij} = 1 \qquad \forall j \in V \setminus \{0\} \tag{2}$$

$$\sum_{j \in V} x_{ij} = 1 \qquad \forall i \in V \setminus \{0\} \tag{3}$$

$$\sum_{i \in V} x_{i0} = K \tag{4}$$

$$\sum_{j \in V} x_{j0} = K \tag{5}$$

$$\sum_{i \notin S} \sum_{j \in S} x_{ij} \geq r(S) \qquad \forall S \subseteq V \setminus \{0\}, S \neq \varnothing \tag{6}$$

$$x_{ij} \in \{0,1\} \qquad \forall i, j \in V \tag{7}$$

where:

V – set of vertices of the considered network,

c_{ij} – cost related to moving from the vertex i to the vertex j,

x_{ij} – a binary variable that assumes a value of 1 in a situation where the connection (i, j) is in the solution and 0 in the opposite case,

K – the size of the vehicle fleet,

S – a subset of set V,

r – the minimum number of vehicles necessary to handle the set S.

This study assumes considering a basic variant of the *VRP*. However, the results obtained will also be true for other variants of this problem. Any algorithms considered here may also be used to solve other problems in this group (the issues related to their adaptation concern only instructions checking the feasibility of the current solution). Many variants of the *VRP* have been identified in the literature [4, 12, 18]. Numerous methods of solving them have also been developed, both exact [4, 5], heuristic [4, 15, 27] as well as metaheuristic [4, 16, 20]. Among these approaches, methods using the iterative local search strategy (*ILS*), which are used in solving a large group of problems from the *VRP*s family [2, 8, 19], deserve special attention.

ILS is an approach that, in order to increase the area of searches and thus improve the quality of the returned solution, implements a strategy that assumes repeated sequential calling on the considered solution the local search algorithm (algorithms) and the perturbation mechanism (mechanisms). As a result, several different local optima are achieved, which increases the chance of finding a global optimum, or at least a solution close to it [2, 14, 28].

Approaches from the *ILS* group can use a variety of local search algorithms [2, 14, 19]. However, it seems right to assume that the use of algorithms capable of returning better-quality solutions increases the chance of achieving more favorable final solutions. One of the widely used algorithms capable of returning good quality solutions in a relatively short time is *Swap* (2-1) heuristic [11, 24, 29]. This approach seeks to improve the considered solution by swapping the places of two adjacent clients i and i+1 from route r with client k belonging to route r' (where r and r' are routes of solution s). This algorithm assumes that customers moving from route r can be inserted into route r' in two ways, ie. i, i+1 and i+1, i [11, 29].

In fact, it is difficult to assess what effect the disturbing mechanisms have on the *Swap* (2-1) algorithm (and also on any other local search algorithm). It is not known which movements will ensure the improvement of the quality of returned solutions, and which will only cause an extension of the algorithm's operation time, and thus a decrease in the effectiveness of the entire procedure. Additionally, it is difficult to determine what the number of initiated perturbation movements. Therefore, it seems to be appropriate to carry out certain checking activities should be.

The purpose of the work is to assess the impact of specific perturbation movements and the number of their calls on the quality of solutions generated by the *Swap* (2-1) procedure, as well as the time in which this structure returns them.

2. Research methods

The following research strategy was adopted in the study. *Swap* (2-1) heuristic, whose pseudo-code is presented in Algorithm 1, is called in the local search phase by the *ILS* metaheuristics, whose pseudo-code is presented in Algorithm 2. Whereas in the perturbation phase a single perturbation mechanism is used (successive algorithms are tested, the specifics of which are presented in Table 1). The number of perturbations in each case increased in the range of 1 to 30. This strategy is used for individual test cases, which include initial solutions generated randomly following the guidelines contained in Table 2. Each time the amount of improvement in the value of the objective function was determined, as well as the time in which the final solution is returned.

Algorithm 1. *Swap*(2-1) *heuristic*

Input
Initial solution *s*
Output
Locally optimal solution *s*

```
1. Procedure Swap2_1(s):
2.  s'← s
3.   modification ← true
4.    while modification ≠ false:
5.     modification ← false
6.      for route in s:
7.       s" ← s – route
8.        for i ← 2 to n-2:
9.         for route' in s":
10.         for k ← 2 to n'-2:
11.          if condition of enforceability = true:
12.           f(new_s) ← the cost of the solution that would result from implementing the modification
13.            if f(new_s) < f(s'):
14.             modification ← true
15.              s'← news
16. return s
```

The notation contained in Algorithm 1 should be understood as follows. First, the initial solution s is considered as the best current solution s' (line 2). Then (line 3) the *modification* variable, which holds information about whether an improvement has been found for the best current solution, is set to *true*. The while loop (lines 4-15) evokes (cyclically) instructions from lines 5-15 continuously when the *modification* variable has a value other than *false*. The first of these instructions is responsible for setting the modification variable to false (line 5). The next (line 6) initiates the operation of the iterating for loop along the solution routes *s* and calls the instructions responsible for creating the solution *s″* constituting a copy of the solution *s*, minus the route currently selected by this loop (line 7) and initiates the operation of the next *for* loop (line 8). This loop is responsible for iterating over the elements of the route (in this way successive vertices i and i+1 are selected) and calling the next *for* loop (line 9), which in turn iterates along the routes of the solution s'' in which the next *for* loop (line 10) selects the next vertices k. This loop also contains instructions for checking the feasibility condition (line 11); calculating the value of the solution objective function that would result from exchanging the vertices i and i+1 with the vertex k (line 12); checking if the value of the objective function of this solution is be more favorable than the value of the objective function of the best current solution (line 13), as well as in the case where this condition turns out to be true setting the *modification* variable value as *true* (line 14), generating this solution and adopting it as the best current solution (line 15). The last instruction (line 16) is responsible for returning the solution that is the final effect of the algorithm.

Algorithm 2. Iterated Local Search metaheuristic [2]

Input
Initial solution s_0
Output
Final solution s^*
1. ***Procedure*** *ILS(s0):* 2. *s*←Swap2_1(s0)* 3. ***while*** *stopping criterion is not met:* 4. *s**←Perturb(s*, history)* 5. *s***←Swap2_1(s**)* 6. ***if*** *f(s***)<f(s*):* 7. *s*←s**** 8. ***return*** *s**

The notation contained in Algorithm 2 should be understood as follows. First (line 2) as a result of the *Swap* (2-1) algorithm, solution s^*, which becomes the best current solution, is generated. Then the operation of while loop is initiated (lines 3-7), which cyclically executes the instructions from lines 4-7 until the stop criterion is encountered (usually, this criterion concerns the inability to find improvement). The first of these instructions (line 4) assumes calling the perturbation mechanism on the solution s^*, as a result of which the solution s^{**} is generated. This solution is then modified again by the *Swap* (2-1) algorithm, which results in returning the solution s^{***} (line 5). The if conditional instruction covering lines 6-7 checks whether the value of the goal function of the solution s^{***} is lower than the value of the goal function of the current best solution s^* and in the case of the truth of this condition, assignment $s^* \leftarrow s^{***}$ is made, as a result of which the solution s^{***} becomes the new best solution. The study considered perturbation mechanisms developed based on the most common local search algorithms in the form of *Multiple-Swap* (2-2), *Multiple-Shift* (2-2) and *Multiple-K-Shift* (1).

Table 1. Considered perturbation mechanisms

Perturbation	Description
Multiple-Swap(2-2)	having random character move aimed at swapping places of two adjacent clients i and i+1 belonging to the route r with two clients k and k+1 from route r' (performed taking into account four insertion options, i.e. i, i+1 and k, k+1; i, i+1 and k+1, k; i+1, i and k, k+1; i+1, i and k+1, k) [7, 21, 23]
Multiple-Shift(2-2)	having random character move directed at moving two adjacent clients i and i+1 from route r to route r' and two adjacent clients k and k+1 from route r' to route r (performed with four insertion variants, as is the case of the Multiple-Swap (2-2) mechanism) [10, 19, 21]
Multiple-K-Shift(1)	having random character move targeted at the displacement of the customer i from route r to the end of route r' [18, 19, 20]

Table 2. Parameters of the used test cases

Problem size (number of vertices in the considered network)	Number of routes	Values of cost matrix elements
25 50	2-5	1-10
75 100	2-10	

10 variants of test cases of each type were generated. Striving to ensure the correct measurement of the algorithm's running time (limiting the impact of other processes implemented by the computer processor), each measurement was repeated five times and the obtained values were averaged.

All the algorithms used (including the test case generator) were programmed in Python 3.7.3. Calculations were made using a computer with Windows Professional 7 64-bit with the i5-4590 processor and 8 GB RAM.

3. Test results

The results obtained are presented in the graphs shown in Figures 1-6 and Table 3. The data plotted on the graphs shown in Figures 1, 3 and 5 show what the average percentage improvement in the value of the goal function of the solution returned by the cooperating *Swap* (2-1) algorithm with a perturbation mechanism that initiates a certain number of perturbs relative to the solution generated by the *Swap* (2-1) algorithm used alone is. The charts in Figures 2, 4 and 6 illustrate the impact of the number of initiated perturbation movements on the duration of the entire procedure. Table 3 presents the minimum and maximum amounts of the average percentage improvement in the value of the objective function achieved using individual perturbation mechanisms, as well as the shortest and longest times in which solutions were returned.

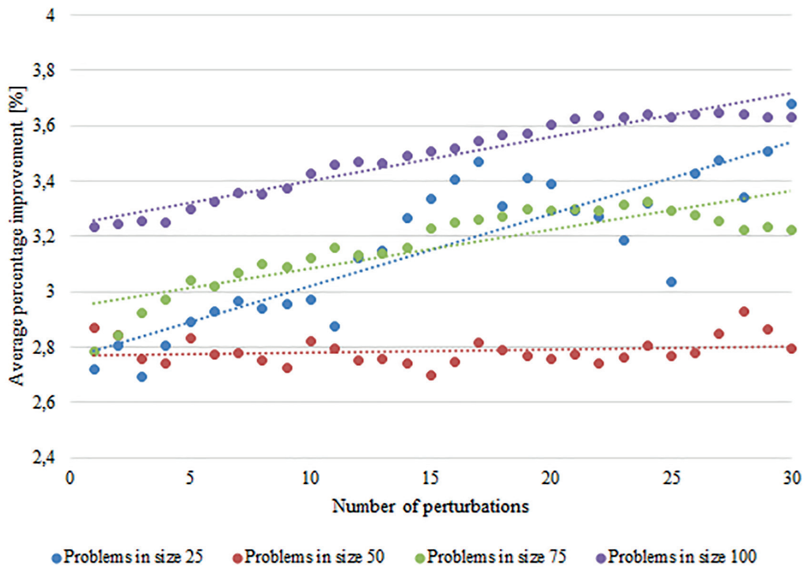

Fig. 1. The average percentage improvement in the value of the goal function during using *Multiple-K-Shift*(1) perturbation mechanism

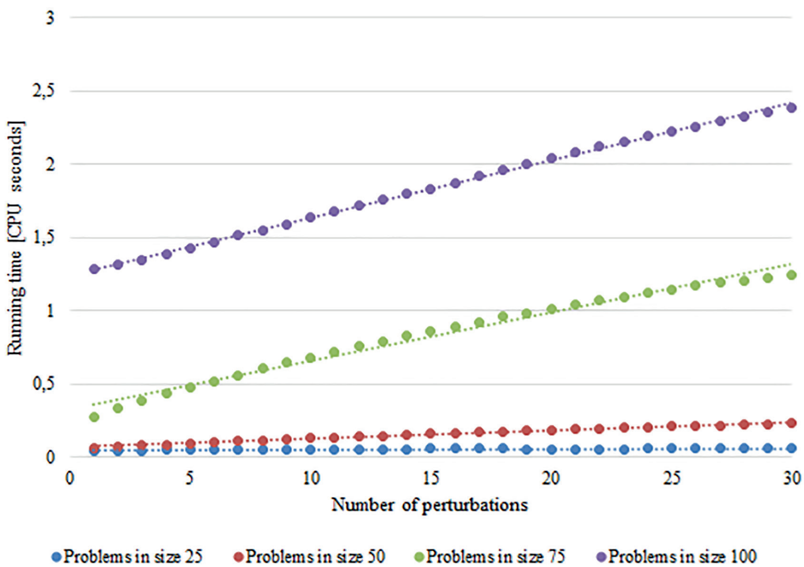

Fig. 2. The algorithm's running time during using *Multiple-K-Shift*(1) perturbation mechanism

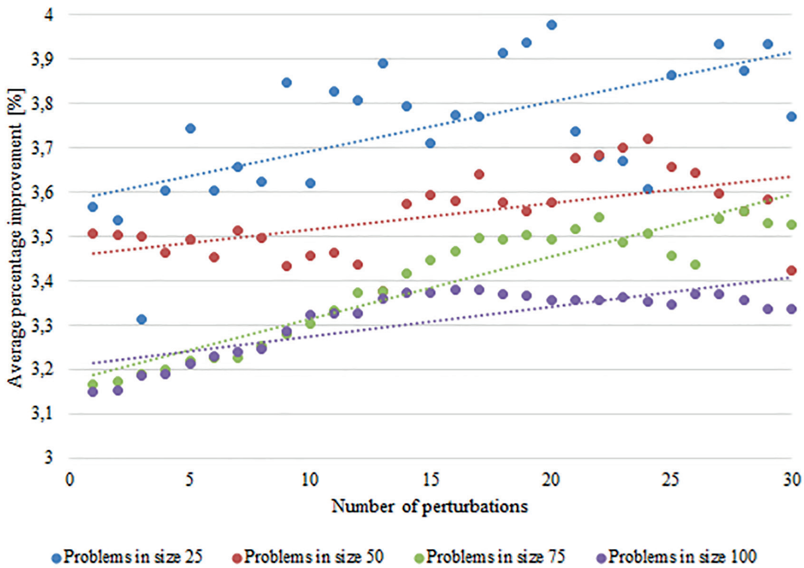

Fig. 3. The average percentage improvement in the value of the goal function during using *Multiple-Shift*(2-2) perturbation mechanism

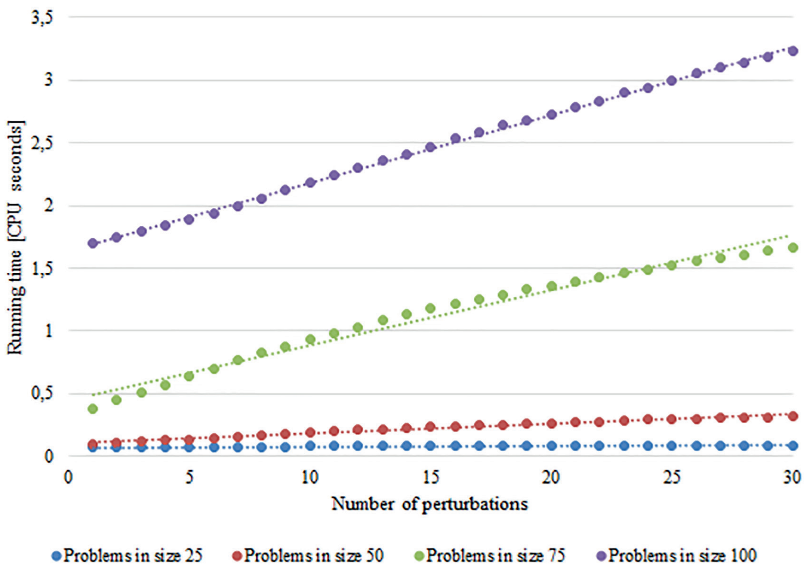

Fig. 4. The algorithm's running time during using *Multiple-Shift*(2-2) perturbation mechanism

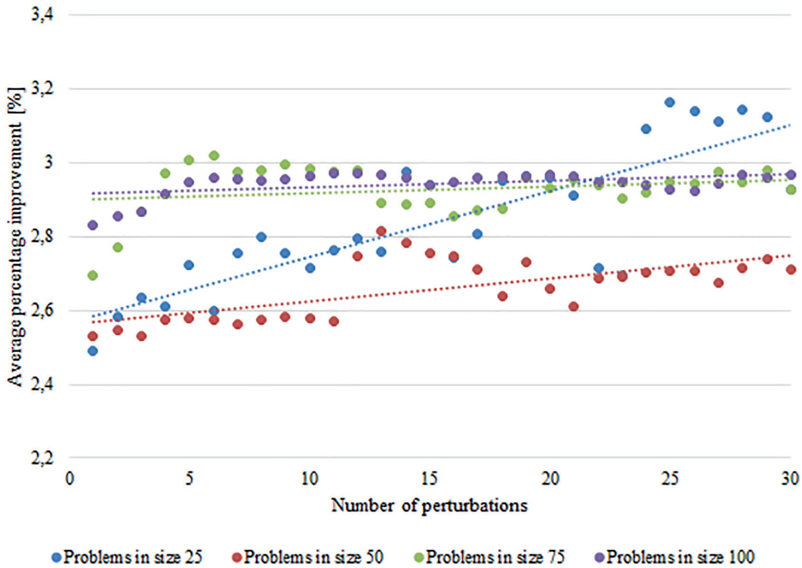

Fig. 5. The average percentage improvement in the value of the goal function during using *Multiple-Swap*(2-2) perturbation mechanism

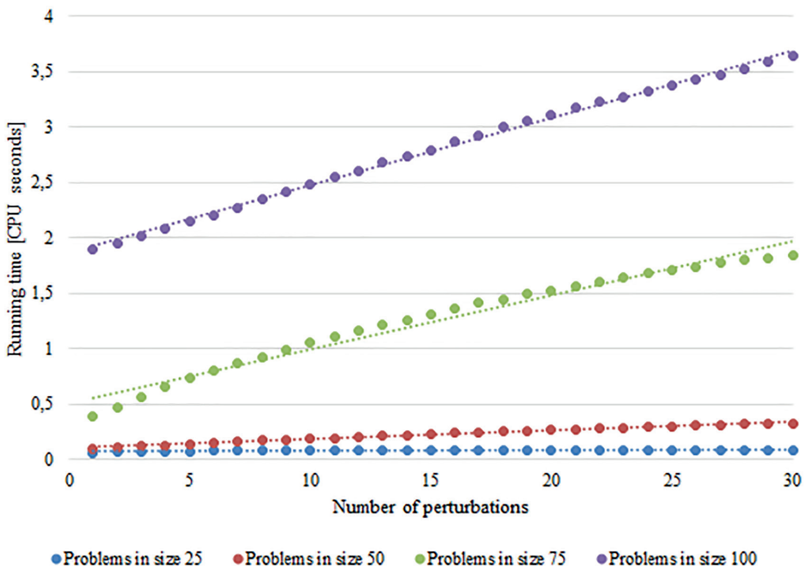

Fig. 6. The algorithm's running time during using *Multiple-Swap*(2-2) perturbation mechanism

Table 3. Minimum and maximum values for improving the solution quality and algorithm running times

Problem dimension	The average percentage improvement in the value of the goal function [%]					
	Multiple-K-Shift(1)		*Multiple-Shift*(2-2)		*Multiple-Swap*(2-2)	
	Min.	Max.	Min.	Max.	Min.	Max.
25	2.72	3.68	3.31	4.02	2.49	3.16
50	2.73	2.93	3.38	3.72	2.53	2.82
75	2.78	3.3	3.17	3.56	2.69	3.02
100	3.24	3.65	3.15	3.38	2.83	2.97
Problem dimension	**Average algorithm running time [CPU seconds]**					
	Multiple-K-Shift(1)		*Multiple-Shift*(2-2)		*Multiple-Swap*(2-2)	
	Min.	Max.	Min.	Max.	Min.	Max.
25	0.044	0.06	0.07	0.09	0.06	0.09
50	0.23	0.67	0.1	0.32	0.11	0.33
75	0.28	1.25	0.38	1.67	0.4	1.85
100	1.28	2.38	1.7	3.23	1.9	3.64

The data contained in the graphs presented in Figures 1-6, as well as in Table 3 indicate that the use of perturbation mechanisms *Multiple-K-Shift* (1), *Multiple-Shift* (2-2) and *Multiple-Swap* (2-2) gives similar results (the average percentage improvement is in the range of 2.49% to 4.02%). In the case of using the Multiple-K-Shift (1) perturbation mechanism, a clear increase in the average percentage improvement in the value of the objective function related to the increase in the number of initiated perturbations is observed when considering test cases of dimensions 25, 75 and 100. Differences between the largest and smallest average percentage improvement are respectively 0.96; 0.52 and 0.41. For problems of size 50, this increase is less intense (the difference is 0.2). The situation is slightly different in the case of the *Multiple-Shift* (2-2) perturbation mechanism, where the differences for problems of sizes 25, 50, 75 and 100 are respectively 0.71; 0.34; 0.39 and 0.23, and thus the smallest change is observed in the case of problems of the largest size. This situation is similar to the results obtained when using the *Multiple-Swap* (2-2) perturbation mechanism, where the differences were respectively 0.67; 0.29; 0.33 and 0.14.

In a slightly different way, depending on the number of initiated perturbations, the algorithm's operating time changed (the changes were more pronounced and unidirectional). For each of the perturbation mechanisms, the increase in the number of initiated perturbations caused an extension of the time in which the solution was returned. Besides, the extension of the algorithm's running time grew more clearly in the case of larger problems. The differences between the longest and the shortest averaged of the algorithm running time for problems with sizes of 25, 50, 75 and 100 in the case of the *Multiple-K-Shift* (1) perturbation mechanism were respectively 0.016 CPU seconds; 0.44 CPU seconds; 0.97 CPU seconds and 1.1 CPU seconds, *Multiple-Shift* (2-2) 0.02 CPU seconds; 0.22 CPU seconds; 1.29 CPU seconds and 1.53 CPU seconds, and *Multiple-Swap* (2-2) 0.03 CPU seconds; 0.22 CPU seconds; 1.45 CPU seconds and 1.74 CPU seconds.

4. Conclusions

Based on the obtained results, it can be concluded that the overall level of achieved improvement in the quality of the returned solution is similar in the case of all of the considered perturbation mechanisms. The level of obtained improvement is relatively low (2.49% - 4.02%), however, it should be remembered that the *Swap* (2-1) algorithm belongs to the group of local search algorithms characterized by the good quality of returned solutions. Besides, it is worth emphasizing that the use of perturbation mechanisms each time led to an improvement in the value of the objective function, and thus an approach of this type effectively widens the area of search, even in the case of a small number of initiated perturbations. It is important that in some cases the increase in the number of perturbations did not significantly improve the value of the objective function (in these cases only the calculation time was longer). Therefore, the use of a larger number of disorders will not always be justified (especially if the algorithm duration is considered to be the most important). It is important that the increase in the number of perturbations does not cause a significant extension of the algorithm's operating time, therefore it is possible to use this strategy in the case in which one is seeking to obtain better quality solutions at the expense of extending the calculation time. It is worth emphasizing that in the case of the *Multiple-Swap* (2-2) mechanism, extending the duration of action when considering problems of the largest sizes is completely unprofitable, as it leads to a slight improvement in the value of the objective function (0.14% when increasing the number of disorders in the range of 1 up to 30).

The worst results were obtained when using the perturbation mechanism *Multiple-Swap* (2-2). This is due to the fact that the move initiated by this mechanism is highly similar to the *Swap* (2-1) local search algorithm, and thus the search areas differ only slightly. In the case of this mechanism, it was also observed to be the most prolonged action, which is due to the complexity of the movement. The shortest operating times were recorded when the *Multiple-K-Shift* (1) disturbing mechanism was used, which is due to the specifics of this movement (low complexity, and thus ease and speed of implementation). The advantage of improving the value of the goal function of solutions obtained using this mechanism over the improvement associated with the use of the *Multiple-Swap* (2-2) approach is related to the fact that the specificity of this move is different from the specificity of move implemented by the *Swap* (2-1) local search algorithm. Less favorable in terms of the operation time, but more favorable in terms of the quality of returned solutions is the *Multiple-Shift* (2-2) perturbation mechanism. Quite unfavorable duration of action associated with the application of this approach is due to its complexity. On the other hand, effectiveness is a consequence of a significant distinction between the specifics of this move and the specifics of move implemented by the considered local search algorithm.

5. References

[1] Aghezzaf B., Fahim H.E.: Iterated local search algorithm for solving the orienteering problem with soft time windows. Springerplus. 2016, 5(1), 1781. DOI:10.1186/s40064-016-3440-6.

[2] Alvarez A., Munari P.: An exact hybrid method for the vehicle routing problem with time windows and multiple deliverymen. Computers & Operations Research. 2017, 83, 1-12. DOI:10.1016/j.cor.2017.02.001.

[3] Arnold F., Sörensen K.: Knowledge-guided local search for the vehicle routing problem. Computers & Operations Research. 2019, 105, 32-46. DOI:10.1016/j.cor.2019.01.002.

[4] Bräysy O., Gendreau M.: Vehicle Routing Problem with Time Windows, Part II: Metaheuristics. Transportation Science. 2005, 39(1), 119-139. DOI:10.1287/trsc.1030.0057.

[5] Cordeau J.F., Gendreau M., Hertz A., Laporte G., Sormany J.S.: New heuristics for the vehicle routing problem. In: Langevin A., Riopel D., eds.: Logistics Systems: Design and Optimization. New York: Springer-Verlag. 2005, 279-297. DOI:10.1007/0-387-24977-X_9.

[6] Den B.M., Stützle T., Dorigo M.: Design of iterated local search algorithms. In: Boers EJW, ed. Applications of Evolutionary Computing. Vol 2037. Lecture notes in computer science. Berlin, Heidelberg: Springer Berlin Heidelberg. 2001, 441-451. DOI:10.1007/3-540-45365-2_46.

[7] Eberle D.U., von Helmolt D.R.: Sustainable transportation based on electric vehicle concepts: a brief overview. Energy & Environmental Science. 2010, 3(6), 689. DOI:10.1039/c001674h.

[8] El-Sherbeny N.A.: Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods. Journal of King Saud University - Science. 2010, 22(3), 123-131. DOI:10.1016/j.jksus.2010.03.002.

[9] Feng Y.J., Zhu H.P., He F.: VRP Problem Research with Workshop Road Constraints Based on Tabu Search. Advanced Materials Research. 2014, 945-949, 3438-3443. DOI:10.4028/www.scientific.net/AMR.945-949.3438.

[10] Galos J., Sutcliffe M., Cebon D., Piecyk M., Greening P.: Reducing the energy consumption of heavy goods vehicles through the application of lightweight trailers: Fleet case studies. Transportation Research Part D: Transport and Environment. 2015, 41, 40-49. DOI:10.1016/j.trd.2015.09.010.

[11] Geroliminis N., Haddad J.: Quantitative methods in transportation systems. EURO Journal on Transportation and Logistics. 2014, 3(3-4), 177-178. DOI:10.1007/s13676-014-0044-6.

[12] Golden B., Raghavan S., Wasil E., eds.: The Vehicle Routing Problem: Latest Advances and New Challenges. Boston, MA: Springer US. 2008. DOI:10.1007/978-0-387-77778-8.

[13] Imran A., Salhi S., Wassan N.A.: A variable neighborhood-based heuristic for the heterogeneous fleet vehicle routing problem. European Journal of Operational Research. 2009, 197(2), 509-518, DOI:10.1016/j.ejor.2008.07.022.

[14] Irnich S., Toth P., Vigo D.: Chapter 1: the family of vehicle routing problems. In: Toth P., Vigo D., eds.: Vehicle Routing: Problems, Methods, and Applications, Second Edition. Philadelphia, PA: Society for Industrial and Applied Mathematics. 2014, 1-33. DOI:10.1137/1.9781611973594.ch1.

[15] Liong C.Y., Wan R.I., Omar K., Mourad Z.: Vehicle routing problem: Models and solutions. Journal of Quality Measurement and Analysis. 2008, 4(1), 205-218.

[16] Lourenço H.R., Martin O.C., Stützle T.: Iterated Local Search. In: Glover F., Kochenberger G.A., eds.: Handbook of Metaheuristics. Boston: Kluwer Academic Publishers. 2003, 320-353. DOI:10.1007/0-306-48056-5_11.

[17] McNabb M.E., Weir J.D., Hill R.R., Hall S.N.: Testing local search move operators on the vehicle routing problem with split deliveries and time windows. Computers & Operations Research. 2015, 56, 93-109, DOI:10.1016/j.cor.2014.11.007.

[18] Palhazi C.D., Goos P., Sörensen K., Arráiz E.: An iterated local search algorithm for the vehicle routing problem with backhauls. European Journal of Operational Research. 2014, 237(2), 454-464. DOI:10.1016/j.ejor.2014.02.011.

[19] Penna P.H.V., Subramanian A., Ochi L.S.: An Iterated Local Search heuristic for the Heterogeneous Fleet Vehicle Routing Problem. Journal of Heuristics. 2013, 19(2), 201-232. DOI:10.1007/s10732-011-9186-y.

[20] Pop P.C., Fuksz L., Marc A.H.: A variable neighborhood search approach for solving the generalized vehicle routing problem. In: Polycarpou M., de Carvalho A.C.P.L.F., Pan J.S., Woźniak M., Quintian H., Corchado E., eds.: Hybrid Artificial Intelligence Systems. Vol 8480. Lecture notes in computer science. Cham: Springer International Publishing. 2014, 13-24. DOI:10.1007/978-3-319-07617-1_2.

[21] Robinson J., Brase G., Griswold W., Jackson C., Erickson L.: Business models for solar-powered charging stations to develop infrastructure for electric vehicles. Sustainability. 2014, 6(10), 7358-7387. DOI:10.3390/su6107358.

[22] Serrenho A.C., Norman J.B., Allwood J.M.: The impact of reducing car weight on global emissions: the future fleet in Great Britain. Mathematical, Physical and Engineering Sciences. 2017, 375(2095). DOI:10.1098/rsta.2016.0364.

[23] Stopka O., Stopkova M., Kampf R.: Application of the operational research method to determine the optimum transport collection cycle of municipal waste in a predesignated urban area. Sustainability. 2019, 11(8), 2275. DOI:10.3390/su11082275.

[24] Stopka O., Zitricky V., Abramovic B., Marinov M., Ricci S.: Innovative technologies for sustainable passenger transport. Journal of Advanced Transportation. 2019, 1-2. DOI:10.1155/2019/4197246.

[25] Subramanian A., Penna P.H.V., Uchoa E., Ochi L.S.: A hybrid algorithm for the Heterogeneous Fleet Vehicle Routing Problem. European Journal of Operational Research. 2012, 221(2), 285-295. DOI:10.1016/j.ejor.2012.03.016.

[26] Subramanian A., Uchoa E., Ochi L.S.: A hybrid algorithm for a class of vehicle routing problems. Computers & Operations Research. 2013, 40(10), 2519-2531. DOI:10.1016/j.cor.2013.01.013.

[27] Tan K.C., Lee L.H., Zhu Q.L., Ou K.: Heuristic methods for vehicle routing problem with time windows. Artificial Intelligence in Engineering. 2001, 15(3), 281-295. DOI:10.1016/S0954-1810(01)00005-X.

[28] Toth P., Vigo D., eds.: The Vehicle Routing Problem. Society for Industrial and Applied Mathematics. 2002. DOI:10.1137/1.9780898718515.

[29] Wang S., Tao F., Shi Y., Wen H.: Optimization of Vehicle Routing Problem with Time Windows for Cold Chain Logistics Based on Carbon Tax. Sustainability. 2017, 9(5), 694. DOI:10.3390/su9050694.

[30] Zhang L., Niu H.: An Algorithm for Vehicle Routing Problem with Soft Time Windows Using Tabu Search. In: Wang Y., Yi P., An S., Wang H., eds. ICCTP 2009. Reston, VA: American Society of Civil Engineers. 2009, 1-6. DOI:10.1061/41064(358)458.