A Simple Heuristic for Basic Vehicle Routing Problem

Nodari Vakhania[1,*], Jose Alberto Hernandez[2], Federico Alonso-Pecina[2] and Crispin Zavala[2]

[1]*Centro de Investigación en Ciencias, UAEM or, Mexico*

[2]*Facultad de Contaduría, Administración e Informática UAEM or, Mexico*[#]

Abstract: The vehicle routing problem is an important real-life transportation problem. We propose a two-phase construction heuristic for the solution of the classical Euclidean (uncapacitated) vehicle routing problem in which the minimum cost k distinct vehicle tours are to be formed for the given n customer locations. At the first phase we construct a polygon in the 2-dimensional Euclidean space that girds all the given points (customer locations and the depot). The second phase consists of two stages. At the first stage the interior polygon area is partitioned into k triangle areas, and at the second stage the k tours for each of these areas are constructed.

Keywords: Vehicle routing problem, Heuristic algorithm, Euclidean distance, Weighted graph.

1. INTRODUCTION

The Vehicle Routing Problems (VRP) arise in vast amount of practical circumstances when the goods are to be distributed to the customers using a limited number of vehicles. In general, the transportation problems form a significant part of practical real-life problems (see, for example, Rodrigue *et al.* [11]) formalized as mathematical combinatorial optimization problems. They are typically intractable, hence one often looks for some heuristic methods for their solution.

One of the most practical and also complex combinatorial optimization problems is the Vehicle Routing Problem (proposed first by Dantzig and Ramser in early 1959). The basic (uncapacitated) version of this problem can be stated as follows. We are given an undirected weighted (complete) graph $G=(V,E)$ with edge weights w_e, for all $e \in E$, a distinguished node v_d from set V (called *depot*) and a positive integer number k. $(i,j) \in E$ is the edge connecting node i with node j. For any $Y \subseteq V$ containing node v_d, a *tour* T_Y defined by set Y is a directed cycle that starts at that node, visits every node in Y exactly once and returns to the same node v_d; in other words, $T_Y=(i_1,i_2,\ldots,i_l,1)$, where $(i_2,\ldots,_l)$ is an enumeration of the nodes in set Y not including node v_d and $i_1=v_d$. The *cost* of tour T_Y, $c(T_Y)$ is the sum of the weights of the edges on this cycle, *i.e.*, $c(T_Y)=w(i_1,i_2)+w(i_2,i_3)+\ldots+w(i_l,1)$. VRP aims to find the partition of nodes from set $V\setminus\{v_d\}$ into k subsets $V_1,\ldots,V_k$ with the minimal possible total cost; that is, with the minimal $\sum_{i=1,2,\ldots,k} c(V_i)$.

VRP is a generalization of a well-known Traveling Salesman's Problem (TSP): VRP with $k=1$ becomes TSP. Multiple TSP, a generalization of TSP with k-tours, k-TSP, is a VRP with k vehicles.

We deal with geometric two-dimensional version of the problem when edge weights represent Euclidean distances between the nodes, considering the nodes themselves as points (cities or customers) in the two-dimensional Euclidean space. The corresponding problem with already 1 vehicle, *i.e.*, the corresponding 1-TSP is already NP-hard Papadimitriou [10]. Therefore, we do not pretend to solve our VRP optimally but rather suggest an efficient heuristic for its solution.

Giving a practical interpretation to VRP, assume we have k identical distinct resources or vehicles (one for each of the subsets V_i) that can travel in between the cities. The weight $w(i,j)$ is the distance between nodes i and j. We aim to minimize the total travel distance (time) of all the vehicles. There are a number of extensions of VRP, the most common of which is the capacitated version in which every vehicle has a given capacity that cannot be exceeded during its tour.

The vehicle routing problems have been extensively studied, the most of the solutions methods in the literature being heuristic (see, for example, Laporte and Semet [7], Gendreau *et al.* [4] and Mester and Braysy [9]). There are a few enumerative algorithms as well that work on small instances relatively good (see, for

*Address correspondence to this author at the Centro de Investigación en Ciencias, UAEMor, Mexico; Tel +52 777 329 70 40;
Fax: +52 777 329 70 40; E-mail: nodari@uaem.mx

[#]A preliminary version of this work was presented at the Int. Conference on Abstract and Applied Analysis (ABAPAN 2016)

E-ISSN: 2410-2938/16

example Lysgaard *et al*. [8] and Fukasawa *et al*. [3]). Here we do not pretend to cover all the enormous related work, we rather refer the interested reader to the book edited by Toth and Vigo [12], a newer overview book edited by Golden *et al*. [5], review papers by Christofides *et al*. [1], Laporte [6] and Cordeau *et al*. [2].

In this paper, we propose a two-phase construction heuristic for the the classical Euclidean (uncapacitated) version of the problem. At the first phase we construct a polygon in the 2-dimensional Euclidean space that girds all the given points (customer locations and the depot). The second phase consists of two stages. At the first stage the interior polygon area is partitioned into k triangle areas, and at the second stage the k tours for each of these areas are constructed.

To form the above partition of the interior polygon area, we determine auxiliary k distinct border points on the polygon associating with them vectors directed from depot to each of these points in 2-dimensional Euclidean space. These k vectors define the k triangle areas of the interior of the polygon. The tour corresponding to a given triangle area is formed somewhat "across" the vectors defining this area: the close-by region sorrowing these vectors is a "most dance" zone within that triangle area (in the sense that it contains "the most" of the nodes of the triangle area). The density is measured by a specially introduces density parameter. If all the formed in this way k tours cover all the nodes, then our heuristic halts with the created solution. Otherwise, it updates the density parameter and adds more nodes to the current tours according to the new value of the density parameter. This procedure is repeated until all the nodes are included into the k created tours.

In the next section we describe our procedure for the construction of the polygon girding whole tour area. In Section 2 we describe the partitioning and routing phase, and in Section 3 we give a few final remarks.

2. PHASE 1: THE CONSTRUCTION OF THE GIRDING POLYGON

Without loss of generality and for the commodity, we assume that the given n points from set V have non-negative coordinates (otherwise, we can shift them uniformly without changing the distances between them).

Our task at phase 1 is to determine a special kind of a (closed) convex polygon that contains all the nodes from set V girding in this way the whole tour area. No node from set V may be left outside the area of such a polygon, and, of course, there are many such polygons. Though, we are interested in the minimal such convex polygon with its edges containing the maximal possible number of nodes from set V. We shall refer to this particular polygon as the *girding polygon* for set V and denote it by $P = P(V)$.

It follows that all vertices of polygon $P(V)$ are nodes from set V. Besides these vertices, polygon P may contain nodes from set V as the interior points of its edges, whereas the rest of the nodes from set V are within the internal area of the polygon. In Figure **1** we illustrate polygon P for a problem instance of VRP with 30 customers and one depot.

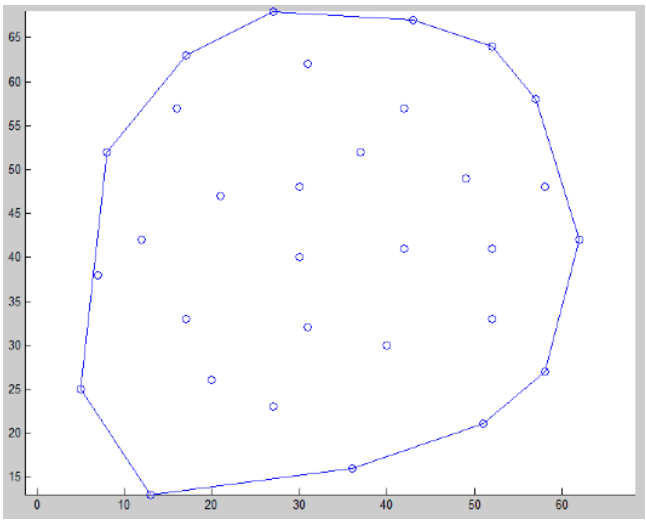

Figure 1: Polygon with 12 vertices and 19 nodes from its internal area.

Before we describe our procedure for the construction of polygon $P(V)$, we define special types of nodes from set V that pertain to polygon $P(V)$. These are the uppermost, lowermost, leftmost and rightmost points from V. Formally, we call a point in set V with the maximum (minimum, respectively) y-coordinate an *uppermost* (a *lowermost*, respectively) node. Likewise, we call a point in set V with the maximum (minimum, respectively) x-coordinate a *rightmost* (a *leftmost*, respectively) node.

We shall refer to these four types of nodes as *extreme* points in set V. From all the extreme points of the same type (if there are two or more such nodes), we call *exterior* nodes the two nodes with the maximum

and minimum co-coordinate, and the rest of the nodes the *interior* nodes of that type. The two exterior nodes are the endpoints of the corresponding edge on polygon P. In general, we may have more than two nodes from set V lying on the same edge of polygon P, two exterior ones of which are endpoints of that edge.

The next observation is straightforward.

Observation 1

All extreme points belong to polygon P, whereas all extreme points of the same type belong to the same edge of that polygon. In particular, the two edges containing all the uppermost and all the lowermost nodes are parallel to the x-axes, and the two edges containing all the rightmost and all the leftmost nodes are parallel to the y-axes.

We use the following notation for the four distinguished extreme points. (1) v_1 is the right exterior uppermost node (*i.e.*, among all the uppermost nodes v_1 has the maximum x-coordinate); (2) v_l is the lowest exterior leftmost node (*i.e.*, among all the leftmost nodes v_l has the minimum y-coordinate); (3) v_o the right exterior lowermost node (*i.e.*, among all the lowermost nodes, v_o has the minimum x-coordinate); (4) v_r is the lowest exterior rightmost node (*i.e.*, among all the rightmost nodes v_r has the minimum y-coordinate).

The procedure POLYGON that forms the polygon P, first determines all the extreme points verifying the corresponding coordinates in the straightforward way. The construction of polygon P goes on the following four stages (which are independent and can be carried out in parallel).

At stage 1 the construction proceeds in the "right-to-left" downward fashion that moves from vertex v_1 towards vertex v_l completing the upper left border of polygon P. At stage 2 the construction proceeds also in the "right-to-left" but upward fashion that moves from vertex v_o towards vertex v_l completing the lower left border of polygon P. At stage 3 the construction proceeds in the "left-to-right" upward fashion that moves from vertex v_o towards vertex v_r completing the lower right border of polygon P. At stage 4 the construction moves also in the "left-to-right" but downward fashion that moves from vertex v_1 towards vertex v_r completing the upper right border of polygon P. Below we describe these stages in more details.

At stage 1 ("right-to-left" downward) we add points to the left of the latest added so far point to polygon P, initially, to the left of vertex v_1. We first determine the next to v_1 vertex v_2 to the left of vertex v_1 (*i.e.*, $x_2 < x_1$) on the projected border of polygon P. v_2 is the uppermost closest to v_1 vertex on its left hand side. In other words, among all nodes in set V with no-larger than y_1 y-coordinate and no-larger than x_1 x-coordinate, v_2 has the maximum y-coordinate (note that by the definition of the initial node v_1, no other node in set V may have a larger than y_1 y-coordinate). If it turns out that $y_2 = y_1$, the corresponding edge of polygon P (one containing nodes v_1 and v_2) is parallel to x-axes. The next point v_3 on the border of polygon is similarly defined, where we restart now from node v_2 (replacing of node v_1), which is now the next node in set V with the maximum y-coordinate. If $y_3 = y_2$ then all three nodes v_1, v_2, v_3 lie on the same edge of polygon P and node v_2 turns out to be an intermediate point on that edge, *i.e.*, it is not a vertex of polygon P, but it belongs to its border. Observe that v_1 is a vertex of polygon P, whereas whether v_3 is a vertex of polygon P or not, depends on whether for the next point v_4, y_4 is equal to or is less than y_3 (it cannot be more).

The "right-to-left" downwards pass of stage 1 ends by adding the leftmost vertex (verices) (ones with the minimum x-coordinate) to polygon P; if there are several such nodes in set V, polygon P possesses an edge parallel to the y-axes containing all these nodes (Observation 1), which is the leftmost edge of the polygon. All of these nodes are successively added and stage 1 ends by adding the lowermost such node (one with the smallest y-coordinate), which we denoted by v_l.

Stage 4 works as stage 1 with the only difference that the construction here moves in the "left-to-right" (instead of "right-to-left") downward fashion from vertex v_1 now to vertex v_r. In the description of stage 1 above, we merely replace "left" with "right" and vertex v_l with vertex v_r.

At stage 2 ("right-to-left" upward) we add points to the left of the latest added so far point to polygon P, starting from vertex v_o. We determine the next to v_o vertex v_ω to the left of vertex v_o on the projected border of polygon P. v_ω is now the lowermost closest to v_1 vertex on its left hand side. In other words,

among all nodes in set V with no-smaller than y_o y-coordinate and no-larger than x_o x-coordinate, v_ω has the minimum y-coordinate. As at stage 1, it may again turn out that $y_\omega = y_o$, in which case we have an edge in polygon P parallel to x-axes. The next point on the projected border of polygon P is similarly defined, where we restart now from node v_ω (replacing of node v_o), which is now the next node in set V with the minimum y-coordinate. Proceeding in this way, the construction at stage 2 ends by matching the latest added vertex with vertex v_l (by the construction, this event will clearly take the place).

Stage 3 works as stage 2 with the only difference that the construction here moves in the "left-to-right" (instead of "right-to-left") upward fashion from vertex v_o now to vertex v_r. In the description of stage 2 above, we merely replace "left" with "right" and vertex v_l with vertex v_r.

This completes the description of procedure POLYGON. It straightforwardly follows from the construction that the obtained polygon is convex and contains all the nodes from set V either on its border or within it, and among all such polygons it is minimal. Hence, it is the girding polygon $P(V)$.

The brutal sequential time complexity of procedure POLYGON is $O(n^2)$. Indeed, initially, the selection of each of the extreme nodes v_1, v_l, v_o, v_r takes time $O(n)$. At all stages, the determination of every next added vertex takes also time $O(n)$, and since there are no more than n added points to the constructed polygon, we get a brutal sequential overall time of $O(n^2)$.

Thus we have proved the following result.

Theorem 1

Procedure POLYGON creates the girding polygon $P(V)$ in brutal sequential time $O(n^2)$.

3. PHASE 2: THE PARTITION AND ROUTING

In this section we describe how we form the k vehicle tours using the girding polygon $P(V)$ constructed at phase 1. For the convenience, assume for now that the depot v_d is within the internal area of polygon P (it normally shares the central area in between the rest of the nodes).

3.1. Partition stage

Let x be any node on the borderline of polygon P. We define an *auxiliary* edge (v_d, x) and associate with it the corresponding vector in the 2-dimensional Euclidean space (will shall refer to both, the edge and the corresponding vector interchangeably).

Denote by m be the number of the nodes from set V which are on the borderline of polygon P.

An auxiliary edge (v_d, x) is called a *separator* edge if $x \in V$. The m separator edges partition the interior area of polygon P into the m corresponding triangle areas (see Figure **2**). Every such triangle area is uniquely defined by two neighboring separator edges. In general, a pair of auxiliary edges define a triangle area. We shall specify a bit later how we determine the k auxiliary edges that delineate the destiny k triangle areas.

Let the weight of an auxiliary edge (v_d, x) be the length of the corresponding vector in the 2-dimensional plane.

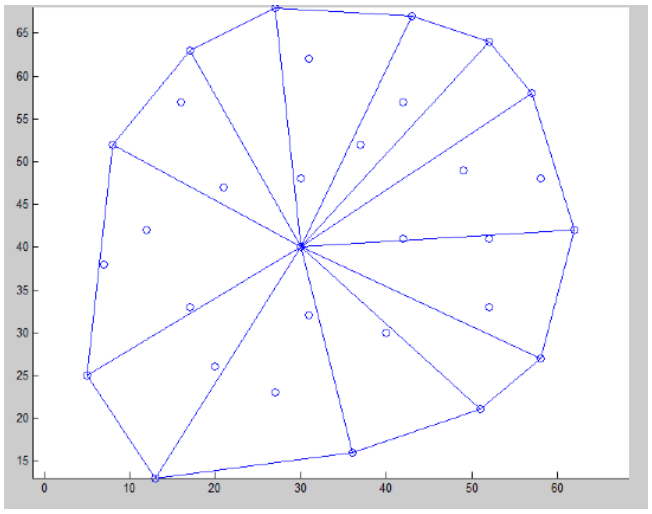

Figure 2: Polygon of Figure **1** with separator edges

Lemma 1

Twice the minimal separator edge weight plus the length of the border of polygon $P(V)$ is a lower bound $L(V)$ on the optimal solution.

Proof. Immediately follows from an easily seen observation that the above magnitude is the optimal tour length for the case $k = 1$ and for the subset of V containing only the nodes of polygon P and the depot.

As it is not difficult to see, the total length of a tour including a border node $v \in V$ is at least $2w(v_d, v)$. Intuitively, it is reasonable to include in a tour that visits node v also the nodes from the interior of polygon P lying "close-by" edge (v_d, v) and also "close-by" border node(s) of polygon P. If, for instance, we include in that tour $i \geq 1$ neighboring border nodes of the polygon then the nodes from the corresponding i triangle areas will be joined together in one unified triangle areas (one of the destiny k subsets of set V).

For every triangle area, we define one or more auxiliary edges across which the tour(s) within that area will be generated. Some of the nodes from that area will be visited on the way from depot to a border node x, whereas the rest of them on the way back to the depot. We describe the construction in the following subsections. Before, we need some definitions.

The number k, the length of edge (v_d, x), $w(v_d, x)$, its "relative length" which we let to be the ratio of the average separator edge length (*i.e.*, the sum of weights of all the m separator edges divided by m) and $w(v_d, v)$, and the lower bound $L(V)$ from Lemma 1 (a magnitude, greater than the perimeter of polygon P) are the parameters used in our definition of the closeness of two nodes. The magnitude b, our closeness measure, in one way or another takes into account these parameters (in practice, we try different kinds of combinations of these parameters for calculating the closeness measure b).

For a given auxiliary edge e, the b*-close* set of nodes is formed by all the nodes from set V which are within the distance b from vector e. We denote the b-close set for an edge e by $B(e)$.

The *density factor* of an auxiliary edge e, $\delta(e)$ as the number of nodes in set $B(e)$ divided by the length of the corresponding vector. We tend to direct our tours across the auxiliary edges with higher density factors. The total number of the auxiliary edges that we will create depends on the number k (the total number of tours that we need to construct).

If $k > m$ then triangle areas determined by neighboring separator edges might be joined together; if $k < m$ then an extra amount of auxiliary edges might be created within a single triangle area, as we describe in the next two subsections.

3.1.1. Creating extra triangle areas for $k > m$

When $k > m$ (the number of edges of polygon P is less than k), to organize k tours, we create $k - m$ extra triangle areas by introducing extra $k - m$ border points of polygon P. These border points give rise to new auxiliary edges, which, in turn, define extra triangle areas.

The $k - m$ extra auxiliary edges are selected within the most dance regions in the current triangle areas so that the formed edges will have the highest density factors.

3.1.2. Unifying triangle areas for $k < m$

The unification of different triangle areas for the case $k < m$ is accomplished by eliminating the $m - k$ separator edges with smallest density factors. As a result, we are left with k triangle areas.

3.2 Routing stage

We describe now how we form the tour corresponding to a given a triangle area from one of the formed k areas (as described in the previous section). Let v and u be the border points corresponding to that area, with auxiliary edges $e = (v_d, v)$ and $e' = (v_d, u)$.

The tour, roughly, is formed by two basic sub-tours determined by vectors e and e', correspondingly: the first one is directed from node v_d towards node v, and the second one from node u towards node v_d. Each of these sub-tours includes the (close-by) nodes from sets $B(e)$ and $B(e')$, correspondingly.

We first describe how we form the sub-tour across vector e. For the commodity in this presentation, we assume that vector e lies on the y-axes of the coordinate system and node v_d coincides with the origin $(0,0)$ (we rotate the whole polygon area by the necessary angle and shift, without altering any problem data).

We need to create a sub-tour that includes nodes v_d and v and all the rest of the (intermediate) nodes from set $B(e)$. This tour has a zigzag type trajectory and consists of a number of "slices".

Every slice defines a local tour on the left or right side of vector e and is restricted by a fixed magnitude β called the *thickness factor*, we let $\beta = \alpha b$, for some real $\alpha > 0$ (we normally let $\alpha < 1$). Roughly, the thickness factor determines the amplitude within which the nodes will be included in the generated local tour. We define below such local tours.

Let y_1 be the furthest point from node v_d in set $B(e)$ whose y-coordinate is at most β more than that of node v_d. If there is no such a point, *i.e.*, the minimum y-coordinate of a job from set $B(s)\setminus\{v_d\}$ is more than β more than that of node v_d, we (repeatedly) replace point v_d with the point $(0,i\beta)$ on the y-axes, for the minimum integer i, as long as that point remains within the area of polygon P (*i.e.*, it is on vector e) until point y_1 is determined (or point $(0,i\beta)$ turns out to be outside of the the area of polygon P). Denote the determined in this way point by v'. Without loss of generality and for the purpose of this description, assume node y_1 has a positive x-coordinate, *i.e.*, it is on the right-hand side of vector e.

The vector (v',y_1) forms the skeleton of the local tour from point v' to node y_1, *i.e.*, it is directed across that vector. Let $\beta(v',y_1)$ denote the set of nodes from $B(e)$ which are within the distance β from vector (v',y_1) on the same side of vector e. In a local partial tour defined by the former vector, all the nodes from set $\beta(v',y_1)$ are visited in the order of the closeness from node v'. In other words, if we let $i_1,i_2,\ldots,i_l$, $i_l = y_1$, be an enumeration of nodes in set $\beta(v',y_1)$ in the non-decreasing order of their distances from node v', then the node i_1 is visited from node v_d, then node i_2 is visited from node i_1, and so on, node $y_1 = i_l$ is visited from node i_{l-1}. Note that the latest visited node in this local tour is y_1 and that it included all the nodes in set $B(e)$ located below vector (v',y_1).

Now we replace point v' with the next point v'' of the same form $(0,i\beta)$ and defined similarly as point v', and carry out a similar construction. *I.e.*, we determine the next furthest point y_2 now from point v'' with the y-coordinate no more than $i\beta$ (for the newly determined value of i), form vector $((0,i\beta),y_2)$ and the next local tour (on the right or the left hand side of vector e, depending on the potion of point y_2).

At the stage in the above tour when the border node v is reached, another local tour, determined by vector (v,u) is formed, quite similarly as for vector e. Once node u is included to that local tour, the sub-tour determined by vector e' is similarly formed. The whole tour completes once the depot is again reached.

Once all the k triangle areas is processed as above, if all the nodes from set V are included in one of the k generated tours then our heuristic completes with the created solution. Otherwise, some of the formed tours are completed with the remained nodes at the successive iterations with modified values for parameters b and β.

CONCLUDING REMARKS

We have described a simple heuristic algorithm for the basic (uncapacitated) vehicle routing problem. Based on the proposed framework, more sophisticated heuristic algorithms might be derived due to its flexibility: the parameters of the heuristic can be calculated and defined in different ways (adapting, in particular, to the nature of the input problem instances).

REFERENCES

[1] Christofides N, Mingozzi A and Toth P. The vehicle routing problem. In N. Christofides, A. Mingozzi, P. Toth, and C. Sandi, editors, Combinatorial Optimization, chapter John Wiley & Sons 1979; 11: 315-338.

[2] Cordeau JF, Gendreau M, Hertz A, Laporte G and Sormany JS. New heuristics for the vehicle routing problem. In Logistics Systems: Design and Optimization, GERAD (Chapter) 2005; 279-297. https://doi.org/10.1007/0-387-24977-x_9

[3] Fukasawa R, Lysgaard J, de Aragão MP, Reis M, Uchoa E and Werneck RF. Robust branchand-cut-and-price for the capacitated vehicle routing problem. In Proceedings of IPCO X. Columbia University 2004.

[4] Gendreau M, Laporte G and Potvin JY. Metaheuristics for the capacitated vrp. In P. Toth and D. Vigo, editors, The Vehicle Routing Problem, volume 9 of SIAM Monographs on Discrete Mathematics and Applications, chapter 6, pages 129â€“154, SIAM, Philidelphia 2002. https://doi.org/10.1137/1.9780898718515.ch6

[5] Golden Bruce L, Raghavan S, Wasil Edward A. (Eds.) The Vehicle Routing Problem: Latest Advances and New Challenges. Springer (www.springer.com/us/book/9780387777771) (2008)

[6] Laporte G. The vehicle routing problem: An overview of exact and approximate algorithms. European Journal of Operational Research 1992; 59(3): 345-358. https://doi.org/10.1016/0377-2217(92)90192-C

[7] Laporte G and Semet F. Classical heuristics for the capacitated vrp. In P. Toth and D. Vigo, editors, The Vehicle Routing Problem, volume 9 of SIAM Monographs on Discrete Mathematics and Applications, SIAM, Philidelphia 2002; 5: pages 109-128. https://doi.org/10.1137/1.9780898718515.ch5

[8] Lysgaard AN. Letchford, and RW. Eglese. A new branch-and-cut algorithm for the capacitated vehicle routing problem. Mathematical Programming, Ser. A 2004; 100: 423-445.

[9] Mester D and Braysy O. Active guided evolution strategies for large-scale vehicle routing problems with time windows. Computers and Operations Research 2005; 32: 1593-1614. https://doi.org/10.1016/j.cor.2003.11.017

[10] Papadimitriou CH. The Euclidean traveling salesman problem is NP-complete. Theoretical Computer Science 1977; 4: 237-244. https://doi.org/10.1016/0304-3975(77)90012-3

[11] Rodrigue JP, Comtois C and Slack B. The geography of transport systems. Routledge, Taylor & Francis 2009.

[12] Toth P, Vigo D. (editors) Vehicle routing problem SIAM monographs on discrete mathematics and applications 386 SIAM, Philidelphia 2002.

Received on 29-11-2016 Accepted on 22-12-2016 Published on 29-12-2016

http://dx.doi.org/10.15379/2410-2938.2016.03.02.04