International Journal of Modeling and Optimization, Vol. 8, No. 3, June 2018

Solving a Modified TSP Problem by a Greedy Heuristic for Cost Minimization

Murat Çal and Ali Ekici

***Abstract*— Photographing a large area in an instant becomes hard if the area is very large like a rain forest or industrial territory. Therefore, more than one photograph is necessary to visualize the whole area. Agents such as drones are used for taking photographs. They take photographs at several points (nodes) to cover the area. In that sense, the problem may be defined as a variant of TSP. Photographs are able to cover multiple nodes if taken, so it is not practical to go to every node to take photos. Instead, several photos are taken in nodes at the minimum cost, and all nodes are covered. We make use of *accessibility* concept to incorporate this situation into our model. If several nodes are within the field of a particular node, drones can go to that node, take a photo at that node, and are able to cover all accessible neighbor nodes.**

In this study, we provide a unique mathematical model for this modified TSP, show that the problem is NP-Hard and provide a greedy heuristic to find solutions within 1-10% of the solutions and lower bounds on hand. We observe that if photographing costs are kept lower than distance costs, the algorithm yields 1-5% gap, and as the photographing costs increase, performance of the algorithm falls to around 5-10% gap. In all cases, we are able to solve problem instances within seconds to several minutes. Our model based on photographing and accessibility is unique as it involves accessibility based on distance and heights; and the heuristic we provide differs from previous ones in that it incorporates two different sub-heuristics in addition to SCP optimization algorithm.

***Index Terms*—Modified traveling salesman problem (TSP), heuristics, network optimization, OR applications.**

I. Introduction

For many years, Traveling Salesman Problem is approached in different ways by different researchers, and lots of studies are conducted to effectively and efficiently solve the problem. Given the number of nodes, one should find an optimal way in terms of time or cost minimization or some other purpose together with visiting each node.

In our problem, an agent does not have to go to each node physically, rather, the agent is able to reach some nodes from the current node visited by taking photographs, and this saves time and money. In that case, TSP behaves like the so called Set Covering Problem (SCP), where given an entity set S, one should select a minimum number of entities to cover all entities in terms of an objective. For example, municipalities aim to select a minimum number of sub-districts to open fire stations to reach all sub-districts within a predefined amount of time. In that sense our problem is a mix of TSP and SCP, where we should physically or photographically reach all nodes by either visiting them or taking photos with them. In doing this, we should minimize time and/or cost incurred. Accordingly, we define our problem in the next section.

Manuscript received December 1, 2017; revised May 1, 2018.

Murat Çal is with Tubitak Tusside, Turkey (e-mail: muratcal89@yahoo.com)

Ali Ekici is with Ozyegin University, Turkey (e-mail: ali.ekici@ozyegin.edu.tr)

II. Problem Definition and Previous Work

Our model is verbally defined as follows: Given node set N, edge set E that connects each and every node in N, a starting node where our agent is located, we should cover each node in N containing also the starting node, by either directly visiting that node or reaching that node by photographing it from a distance and without physically going there. It is possible for the agent to visit a node just to cover that particular node and not to take a photograph. As such, the agent is not allowed to take photograph at any node to which the agent does not physically go. This operation should be performed at the least cost, considering the travel costs between all the nodes making up a complete route and also the photographing costs that enable connecting to a node within a distance but incurring an extra technology cost. Another parameter to be taken into account is the reachability issue related to distances between the nodes. The reachability of a node from another one, at which a photo is taken, is defined by a function that assigns a probability of being "reachable" depending on the distance.

The reason why we refer to Set Covering Problem and also Traveling Salesman Problem is because the problem is a different version of TSP where all nodes should be covered without the necessity of visiting all of them one by one. We have the opportunity to take a "photo" at a node to cover other nodes that are reachable from that particular node. SCP plays an important role in the heuristic algorithm we designed. In the first step, we divide the problem into two parts, and in the first part we define the photo nodes by which we are able to cover all nodes in the node set N.

A. Mathematical Model

Parameters:

$\mathbf{d_{ij}}$	Distance between node *i* and node *j*
$\mathbf{c_i}$	Photographing cost in node *i*
$\mathbf{r_{ij}}$	Binary number showing whether a photo taken at node *i covers node j*
N	Set of nodes including starting node (0) such that N={0, 1, 2….n}

Decision Variables:

$$\mathbf{x_{ij}} \quad \begin{cases} 1 \text{ if there is a direct path from node i to node j} \\ 0 \text{ otherwise} \end{cases} \qquad \forall i, j \in N$$

$$\mathbf{y_i} \quad \begin{cases} 1 \text{ if a photograph is taken at node i} \\ 0 \text{ otherwise} \end{cases} \qquad \forall i \in N$$

u_i Order of node i in any complete route $\forall i \in N$

Objective Function: We seek to minimize the total cost incurred by traveling between the nodes and also photographing at certain nodes to consequently cover all nodes in N, as shown in (1):

$$\min z = \sum_{i=0}^{n}\sum_{j=0}^{n} d_{ij}x_{ij} + \sum_{i=0}^{n} c_i y_i \quad (1)$$

Constraints: We have 6 types of constraints.

Node Coverage: All nodes should be covered. This can be done either by directly going to that node or taking a photo from a node that "covers" that particular node. This is mathematically expressed in (2):

$$\sum_{i=0, i\neq j}^{n} r_{ij}y_i + \sum_{i=0, i\neq j}^{n} x_{ij} \geq 1 \quad \forall j \quad (1)$$

Visited Nodes Are Left: If there is a node for taking photo, that node should be left to go to another node to eventually build up the route, as given in (3):

$$y_i \leq \sum_{j=0, j\neq i}^{n} x_{ij} \quad \forall i \quad (2)$$

Note in advance that there is an additional constraint assuring that visited nodes are arrived at, however, the next constraint we write includes this arrival constraint by incorporating it into a balance equality.

Node Balance: Node balance, together with (3) makes sure that any node will be left after physical or photographical visit to that node. The balance is given in (4):

$$\sum_{j=0, j\neq i}^{n} x_{ij} - \sum_{j=0, j\neq i}^{n} x_{ji} = 0 \quad \forall i \quad (3)$$

Leaving the Source Node: Our agent is located at a starting node, so it should be leaving the starting node, shown in (5):

$$\sum_{j=0}^{n} x_{0j} = 1 \quad (4)$$

Arriving at the Source Node: Similarly, our vehicle should return to starting node after all nodes are covered. This is made sure by (6):

$$\sum_{j=0}^{n} x_{i0} = 1 \quad (5)$$

Sub-tour Elimination: The most important issue in solving our problem as well as solving a TSP is sub-tour elimination. This constraint correspondingly eliminates sub-tours and makes sure that additional decision variables are only allowed to reflect the order of the nodes within a feasible route. This is given in (7):

$$u_i - u_j + n \cdot x_{ij} \leq n-1 \quad \forall i > 0,\ j > 0,\ i \neq j \quad (6)$$

B. Previous Work

Much progress has been made since the introduction of TSP into literature. Soon after the problem was defined, solution procedures together with the required effort were suggested. The difficulty in defining and handling these constraints is handled in an elegant manner in [1], where they employ an estimation procedure to estimate upper bounds for a 49-city problem and where they illustrate the sub-tour concept by using cutting plane methods and optimality concept by orthogonality and feasibility. The study is a pioneer in that it is denoted as the first time such a large problem instance is solved to optimality. Cutting plane concept are further investigated in [2] and [3] where authors deeply discuss the concept of a cut procedure that is used throughout the solution process. It is the content of the authors that given a solution not satisfying all the constraints which impose another cut to be added, a cut based on tangled tours and combined with old fashion cutting plane process is employed to effectively define a plane. Same authors enumerate other methods derived from numerous studies in [4]. Another cutting plane algorithm is proposed in [5], that is based on a polyhedra strengthening the LP relaxation. In exchange, the linear program created has about twice the size of the usual LP relaxation. Therefore, they propose a lifting step to reduce the size for solvable complexity.

Laporte's work sheds light on exact and approximate algorithms and provides an extensive guide to make use of in finding lower bounds in [6]. In addition to integer programming formulations, the assignment lower bound and related branch and bound algorithms allow finding a lower bound by relaxing integer constraints, leading to an assignment problem that can be solved in O(n3) time [7]. Other exact algorithms that may be of interest are the shortest spanning arborescence bound and related algorithms as in [8], the shortest spanning tree bound and related algorithms and 2-matching lower bound and related algorithms as in [9] and [10]. As for approximate algorithms, of which we have talked and will be talking below, heuristics with guaranteed worst case performance that base on spanning trees and are improved by Christofides heuristics as in [11], heuristics with good empirical performance such as nearest neighbor which can further be divided into tour construction, tour improvement and composite heuristics studied in [12]. Another study compares the lower bound obtained by their model by the Held-Karp bound and the lower bounds obtained in [13] and [14]. It is shown that the Held-Karp bound is at least as tight as van der Veen, which is as least as tight as minimum weight tree. Both Held-Karp and can der Veen yielded best bounds, among which van der Veen prevailed since it provides values in linear time [15]. The study of [16] handles the issue of finding lower bounds by a 1-tree learning based Lagrangian relaxation technique. Here Lagrange multipliers are updated by weighted cost function of neighboring nodes, triggering a learning process to escape local optimality.

Another contribution that differs from traditional literature to the solution of TSP using branch and bound and cut methods is made by Angeniol *et al.* (1988), where, instead of constructing complete routes in each iteration of a pre-defined heuristic, a route that evolves in each iteration that leads to a complete route is defined [17]. This heuristic seeks for maximized gain by scanning all M cities defined by the network. Since there are always M cities in the network as the algorithm proceeds to compute the costs of them, complexity of the problem is reduced in exchange for possible deviations from optimal solution. As for other neural networks studies, [18] and [19] investigate different methods used in solving TSP. [18] compares three different approaches: integer linear programming to obtain optimal solutions without time consideration, Hopfield Neural Network as explained in [20], and Kohonen Self Organizing Feature Map similar to mapping structure in [17]. Studies [21]-[24] also consider Hopfield neural network and provide extensions in terms of computational structures of the same algorithm. For more information on neural networks, [25] can be consulted which classifies all neural network studies based on solving TSP and TSP with backhauls as in [26] and divides the work into three main groups, namely the Hopfield-Tank Network, elastic net algorithms and self-organizing maps. Another metaheuristics for solving TSP are swarm based optimization techniques, such as ant and bee colony algorithms [27], swarm particles and African Buffalos. Particle swarm is developed by [28] and parametric extensions are created by the same logic in [29], [30]. Parameters like inertia weight or discrete structures are also combined with the main concept to accelerate the method and obtain qualified solutions [31].

A different version of TSP is maximum reward collection problem, studied in [32] and its structural properties in terms of its polytope is explained in [33]. A variant of the same problem with multiple agents is studied in [34]. Salesmen try to maximize their reward minus any costs by visiting the nodes which worsen as time passes.

A penalty based heuristic is suggested in [32] that finds the optimal solution up to 20 nodes, after which its performance reduces rapidly. In [34], a Cluster and Route Algorithm (CRA) is suggested to find good enough solutions, where literarily famous k-Means algorithm is first employed to classify rewards in the nodes, after which these classes are planned by single agents to visit. It is observed that the algorithms provide good enough solutions compared to mathematical programming based ones and solution times as well as solution qualities improve as well.

The maximum collection problem is studied also in [35], where this time a single constraint is imposed such that each nodes does not have to be visited exactly once, each node has a given reward and starting node should be returned to, maximizing the total reward within the given time. Similar to what we formulated in our heuristics, this study adds assignment problem's constraints to easily solve the problem by Lagrange relaxation.

Obtaining lower bounds for large instances of a problem that cannot be solved within reasonable times is also important especially when it comes to compare a heuristic algorithm with other exact or approximate algorithms in terms of solution time and solution quality. Mathematical programming formulations such as single or multi-commodity flow can be used to formulate a minimum spanning tree (MST) and then applying branch and cut and price algorithms [36]. The MST formulation we use in our procedure of finding the lower bound originally included sub-tours, but we obtained our lower bound for larger instances by defining minimum cost based sub-tours and then defining the MST accordingly.

III. Solving the Problem by Mathematical Model and By our Heuristic

In this section, we give information about how we generated our own problem instances and also our comments on computational results. To solve the problem, IBM ILOG CPLEX Optimization Studio 12.6.1 is used. The software is installed on Windows 7 Professional Operating System 64 bit, with Intel® Core™ i7-3630QM CPU @ 2.40 GHz processor and 8 GB RAM.

A. Generation of Problem Instances

Before explaining the methodology we used to generate different problem instances, it is necessary to explain any assumptions we made for simplicity. In the construction of our distance matrix, distance is measured in Euclidean form. Since we want to maintain simplicity as much as possible, we directly took distance matrix as the traveling cost matrix, defining the traveling cost 1 monetary unit for 1 unit distance. We assumed a coordinate range between 0 and 1000, making the largest distance between any points at most $1000\sqrt{2}$. We defined different instances based on two different factors, neighborhoods and photographing costs.

TABLE I: Neighborhood Information

Instance Number	Neighborhood Structure	Cost Vector Structure
1A	No Neighborhood	[50,500]
1B	No Neighborhood	[250,1000]
1C	No Neighborhood	[1000,4000]
4A	NH1 ϵ [0,100] NH2 ϵ [900,1000]	[10,250]
4B	NH1 ϵ [0,100] NH2 ϵ [900,1000]	[250,750]
5A	NH1 ϵ [0,250] NH2 ϵ [400,600] NH3 ϵ [750,1000]	[50,500]
5B	NH1 ϵ [0,250] NH2 ϵ [400,600] NH3 ϵ [750,1000]	[50,2000]
6A	NH1 ϵ [200,400] NH2 ϵ [500,550] NH3 ϵ [800,1000]	[10,400]
6B	NH1 ϵ [200,400] NH2 ϵ [500,550] NH3 ϵ [800,1000]	[300, 850]
7A	NH1 ϵ [0,200] NH2 ϵ [200,400] NH3 ϵ [400,600] NH4 ϵ [600,800] NH5 ϵ [800,1000]	[50,500]
7B	NH1 ϵ [0,200] NH2 ϵ [200,400] NH3 ϵ [400,600] NH4 ϵ [600,800] NH5 ϵ [800,1000]	[500,1000]
8A	NH1 ϵ [0,250] NH2 ϵ [200,450] NH3 ϵ [300,700] NH4 ϵ [500,850] NH5 ϵ [925,1000]	[50,500]
8B	NH1 ϵ [0,250] NH2 ϵ [200,450] NH3 ϵ [300,700] NH4 ϵ [500,850] NH5 ϵ [925,1000]	[400, 800]

We created these instances to test our algorithm in different problem structures. Our instances are defined in Table 1. For instance, Instance 4A is defined as follows: x and y coordinates of N points are generated, N/2 of them being generated between 0 and 100, while the rest between 900 and

1000. Photographing costs are generated between 10 and 250.

B. Finding Lower Bounds for Large Instances

Up to 20 nodes, the resulting problem can be solved optimally, after which the problem increasingly becomes difficult to solve. Therefore, for large numbers of nodes, namely large instances with 30, 50, 100 and 400 nodes, we develop a model to find lower bounds by sub-tours. We introduce another additional variable:

$$\mathbf{z_j} \quad \begin{cases} 1 \text{ if the node j is physically visited} \\ 0 \text{ otherwise} \end{cases} \qquad \forall j \in N$$

Our objective function then becomes:

$$\min z = \sum_{i=0}^{n}\sum_{j=0}^{n} d_{ij}x_{ij} + \sum_{i=0}^{n} c_i y_i \tag{7}$$

Our constraints are given as follows:

Node Coverage: As previously stated, all nodes should be covered either by taking a photo or directly visiting them, assured in (9):

$$\sum_{i=0, i\neq j}^{n} r_{ij} y_i + z_j \geq 1 \qquad \forall j \tag{8}$$

Photographing Requires Physical Visit: If a node is used for taking a photo, then that node should be visited by the agent itself. This is satisfied by (10):

$$z_i \geq y_i \qquad \forall i \tag{9}$$

Physical Visited Nodes Should Be Arrived and Left: To formulate this constraint, (11) is used.

$$2z_i = \sum_{j=0, j\neq i}^{n} (x_{ij} + x_{ji}) \tag{10}$$

Visits Imply Edges: If a node is visited, necessary route to that node consisting of the edges should be formed, shown in (12):

$$\sum_{i}^{n}\sum_{j}^{n} x_{ij} = \sum_{i}^{n} z_i \tag{11}$$

The model is then solved in CPLEX and lower bounds for large instances are easily obtained.

C. The Proposed Algorithm

Our algorithm is a combination of a route construction algorithm for TSP and a modified SCP, which are solved in a loop to first select a group of solutions and then try to find the best solution among them. The pseudocode of our algorithm is given as follows:

```
Step 0   **Initialization**
         Read coordinates, cost vectors and
         reachability matrix from Excel
         Take the reachability matrix and
         give it as input to modified SCP
Step 1   **Determination of Base Node Set**
         Solve modified SCP to determine the
         nodes to cover all nodes in the
         network in the least cost
         Create the base list $BL_{SCP}$ such that
         all photo nodes in SCP are ordered in
         increasing order
             If the starting node 0 is already
             in $L_{SCP}$, go to Step 2
             Else add the starting node 0 to
             $L_{SCP}$ and go to Step 2
Step 2   **For each $L_{SCP}$ do**
             Approximate the associated TSP by
             Christofides Algorithm to obtain
             total route travel cost
         Add $z_{SCP}$ and $z_{TSP}$ to obtain the
         cumulative cost $z_{CUM}$ of
         photo-modified TSP
         If no more exclusion is possible in
         $BL_{SCP}$, go to Step 4
         Else go to Step 3
Step 3   **For *each photo node* in $BL_{SCP}$ do**
             Exclude the node from $BL_{SCP}$, solve
             SCP with remaining nodes to
             obtain new $L_{SCP}$, go to Step 2
Step 4   **Obtain the best $z_{CUM}$**
         Record the route, cost and solution
         time of all other itegapns
         Stop
```

Here, BL_{SCP} denotes the list obtained by solving the modified SCP for the first time, and the main loop follows this list. L_{SCP} is the list obtained by solving SCP in each re-start of the loop. The values z_{SCP} and z_{TSP} refer to objective function values of SCP and TSP, respectively. The value z_{CUM} is the total cost associated with the corresponding step of the algorithm, best of which is reported after the algorithm is complete. To illustrate our algorithm, we will solve instance 12-4A and compare the results.

Step 0: All input data is read by our algorithm, taken into memory by Excel files. The modified SCP differs from the original SCP in that it also incorporates physical visits to nodes in order to cover them, shown below:

$$\mathbf{min} \sum_{i=0}^{n} (c_{0i} \times z_i) + \sum_{i=0}^{n} f_i \times y_i \tag{13}$$

subject to

$$\sum_{i=0}^{n} (r_{ij} \times y_i) + z_j \geq 1 \qquad \forall j \tag{14}$$

$$z_0 = 0 \tag{15}$$

The second part in **(13)** facilitates physical visits with the cost of arriving at that node from the center node. Even though an agent does not necessarily arrive at a node from the center node in practice, it is assumed to do so for computational purposes. **(14)** ensures that all nodes are covered either by photo taking or physical visits while **(15)** ensure that the center node is not physically visited during the tour.

Step 1: Accordingly, SCP uses reachability matrix and solves the resulting problem. The problem is straightforward and our heuristic yields the base node set as follows:

Base Node Set: 10*, 7*, 3* where * denotes a photo taking node

Accordingly BL_{SCP} becomes: 3*, 7*, 10*

Since the node 0 is initially not in the list, we add it to BL_{SCP}.

Step 2 and 3: We apply Christofides Algorithm in the following sub-steps to approximate TSP tour for L_{SCP}:

1. Insert Basic Information
2. Find Minimum Spanning Tree
3. Find Odd Degree Vertices
4. Minimum Weight Matching
5. Find Euler Cycle Path
6. Find TSP Cycle Path

We obtain the solution route as 0-10*-7*-3*-0 with z_{CUM} = 2841.14. Next the algorithm searches for any exclusion possibilities. Exclusion means elimination of a photo taking node from the base solution set BL_{SCP}, which is followed by re-solving the problem under the new constraints. In our case, we have three photo-taking nodes, meaning the loop will be called three times to solve the problem in absence of these photo taking nodes. First, the algorithm removes node 3, as given in BL_{SCP} and resolves the modified SCP. We obtain new L_{SCP} as: 5*, 7*, 11* with z_{CUM} = 2901.17, which is greater than our Base Node Set solution. Secondly, the algorithm removes node 7, allows node 3 again and re-solves the problem. We obtain L_{SCP} as: 5*, 10*, 12*, with also node 0 and with z_{CUM} = 2914.28, again greater than initial z value. Lastly, the algorithm removes node 10 and solves the problem again, obtaining L_{SCP} as 5*, 7*, 12* with z_{CUM} = 2987, 032. Since there is no exclusion alternative left, the algorithm proceeds with Step 4.

Step 4: The best solution obtained is reported with z_{CUM} and the associated tour:

z_{CUM} = 2841.14 with tour 0-10*-7*-3*-0

D. Results and Comparison

We compute solutions from instances with no neighborhood and compare z values in terms of their gaps. Gaps are provided in Table II.

TABLE II: AVERAGE Z-VALUE GAPS

Node	Instance A	Instance B	Instance C
6	0.16	0.34	0.51
8	0.19	0.60	0.56
12	0.20	0.57	0.97
17	0.24	0.64	1.11
20	0.62	0.98	1.05
30	0.67	0.91	0.55
50	0.51	0.55	0.97
100	0.47	0.60	1.67

It is seen that as the numbers of nodes increase while increasing photographing costs, z value gaps also increase. Even if the algorithm provides good quality solutions for some instances, it performs increasingly worse as increasing photographing costs makes it preferable to physically visit the nodes rather than taking photos of them. To understand the effects of photographing costs and also the reachability matrix on z value gaps, we re-computed the instances as to solve them to optimality and also to obtain heuristic solutions. We derive three conclusions about the solution quality being affected by photographing costs and reachability structure:

Conclusion 1: Under the condition that all input data remain the same, as photographing costs are decreased, the algorithm performs better because each candidate node for taking photos will then incur much less values to optimal objective function value. This is best illustrated in instances with multiple photo taking nodes and also high z value gaps. We re-solve instances by stepwise reduction of photographing costs and obtain the results shown In Table III.

TABLE III: AVERAGE GAPS UNDER DECREASING PHOTO COSTS

%Reduction	Gap	%Reduction	Gap	%Reduction	Gap
0	1,12	20	0,83	60	0,60
5	1,08	40	0,72	80	0,58
				90	0,59

Conclusion 2: Increase in reachability (IncR) promotes the algorithm's solutions and enhances solution quality in that the feasibility regions of instances increasingly begin to contain the extreme points where photo nodes that cover more nodes exist. We illustrate this concept best by resolving instance 30-1B, where the optimal solution simply visits all nodes without taking any photos, and our heuristic algorithm takes photos at multiple nodes, leading to a z value gap of 0.28. We select instance 30-1B because it contains no neighborhoods and impacts of increasing 1s in reachability matrix can be measured easily without having to consider neighborhood complexity. Then we apply the same procedure to 5 more instances and take the average values. We increase number of visual connections of each node randomly by certain numbers, then investigate the change in z value gaps by resolving the problem, whose results are shown in Table IV.

TABLE IV: AVERAGE REACHABILITY-GAP CHANGES

%IncR	Gap	%IncR	Gap	%IncR	Gap
0	0,26	2	0,12	6	0,10
1	0,15	4	0,07	8	0,11

As is the case in photo costs, z gaps decrease significantly at first, and they increase after a certain point because of the change in feasibility region and also the stability of our heuristic to take into account these reachability changes. This graph also shows that our heuristic is able to produce solution within 5% of the optimal solution even in cases where photo costs are higher.

So far we looked at reachability and photographing aspects separately. We now illustrate these concepts together, where we decrease photo costs by percentages while increasing reachability. We use the same increasing and decreasing structure and obtain the results summarized in Fig. 1. This time we only use one instance, but the reader should note that at least 3-5 instances should be investigated to observe the difference.

Conclusion 3: The results show that it is not necessarily true that we obtain better results as we provide more reachability and lower photo costs, rather, it causes more time to evaluate each alternative and thus increases overall solution time. In addition, although CPLEX solves each problem to optimality and obtains better solutions as we give better input, our algorithm is not able to find better solutions in some cases, as seen in 60% and 80% decreased photo costs, in particular the cases in which reachability is increased by 4 and 6 nodes. We already stated that these increases in z gaps are due to preliminary SCP that our algorithm solves. Even if the z gaps increase by some percent, however, the overall performance of the algorithm reaches 6%, 14%, 11%, 10% and 6% in respective photo cost reductions.

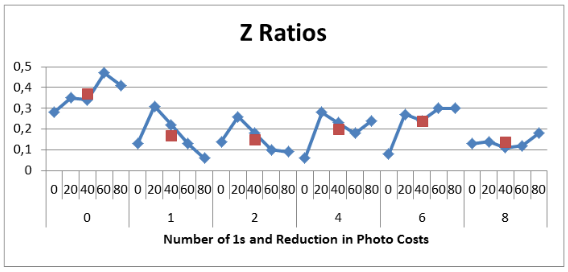

Fig. 1. 30-1B Z gaps under reachability and PHOTO Cost changes.

IV. Conclusion and Further Discussion

Monitoring and photographing large areas for security, safety or other specific reasons requires careful and detailed planning since decision makers always have limited time and other resources. In a given amount of time, an area within a building, a zone within a region or even a region itself should be monitored to take precautions against different emergencies. To prevent fires in a bank or school, to set up cameras or alarms for burglary, to conduct periodical controls in forests for fire prevention or to monitor borders for intruders and terror attacks are among different motivations of field scanning.

For many years, Traveling Salesman Problem is approached in different ways by different researchers, and tremendous numbers of studies are created to effectively and efficiently solve the problem. In this study we aimed to provide optimal as well as heuristic solutions for a modified TSP problem, where agents do not have to go to each node physically, rather, they are able to reach some nodes from the current node visited, and this saves time and money. We provided a mathematical model for this problem, showed that it is a combination of TSP and SCP and is NP-Hard, provided a greedy heuristic algorithm to solve the problem in acceptable times and compared the results. We have seen that the problem structure is dependent on reachability and photo costs, and we tried to investigate these structures in various numbers of nodes and input schemes. Our mathematical model is a uniquely modified version of traditional TSP, and although there are many different models regarding TSP in literature, no model has been found to automatically visit unvisited nodes through photographing and monitoring. Our heuristic is also unique by two aspects: 1-It is specifically designed for solving our mathematical model and 2-It successfully combines Christofides heuristic, SCM optimization algorithm and a looping scheme to minimize the time required to obtain solutions while trying to get the best result.

All in all, our algorithm is able to provide good solutions even within 1% of the optimal solution. We have observed that solution quality may change depending on the reachability probability function, and since our heuristic algorithm initializes by solving a preliminary SCP, it approaches to solutions with multiple photo taking nodes, thus failing to provide good enough solutions for the associated problem instance. Nevertheless, the algorithm is capable of providing solutions for each instance ranging from 6 to 400 nodes, and it take only nanoseconds to obtain a good solution. The algorithm provides solutions within %5-10 of the optimal solution on the average and performs flawlessly on digital framework.

There are several pitfalls of the algorithm, among which preliminary SCP's photo costs worsen the solution. Since a physically visited node should incur a cost based on distance, and since we cannot possibly know the physically visited node's distance cost before it is visited, we simply made the assumption that any physically visited node incurs the cost of its distance from the center node. This assumption leaves the algorithm's performance for probabilistically changing levels due to random number generation in distances. If the center node is close to physical visit node, then there is less cost incurred to objective function value, and if there is a large distance between the visit node and the center node, the algorithm is not able to measure it correctly to select it. Suppose that in a good enough solution that far distanced node incurs a little cost because its neighborhood nodes are very close. However, since the SCP of the algorithm cannot measure it directly, it may automatically eliminate that node and may start with a worse solution to start searching. So in future studies it should be the objective of us to handle the SCP part of the algorithm so that photographing as well as physical visits are given the correct costs and the algorithm does not miss any solution in the beginning.

References

[1] S. J. G. Dantzig and R. Fulkerson, "Solution of a large-scale traveling-salesman problem," *J. Oper. Res. Soc. Am.*, vol. 2, no. 4, pp. 393–410, 1954.

[2] D. Applegate, R. Bixby, V. Sek, and C. Atal, "Finding cuts in the TSP (A preliminary report)," 1995.

[3] D. Applegate, R. Bixby, V. Chvátal, and W. Cook, "TSP cuts which Do not conform to the template paradigm."

[4] D. Applegate, R. Bixby, C. Vasek, and W. Cook, "On the solution of traveling salesman problems," *Doc. Math.*, pp. 645–656, 1998.

[5] E. Balas, S. Ceria, and G. Cornuéjols, "A lift-and-project cutting plane algorithm for mixed 0–1 programs," *Math. Program.*, vol. 58, no. 1–3, pp. 295–324, Jan. 1993.

[6] G. Laporte, "The Traveling Salesman Problem: An overview of exact and approximate algorithms," *Eur. J. Oper. Res.*, vol. 59, pp. 231–247, 1992.

[7] E. Balas and N. Christofides, "A restricted Lagrangean approach to the traveling salesman problem," *Math. Program.*, vol. 21, no. 1, pp. 19–46, Dec. 1981.

[8] R. E. Tarjan, "Finding optimum branchings," *Networks*, vol. 7, no. 1, pp. 25–35, 1977.

[9] K. Helbig Hansen and J. Krarup, "Improvements of the Held—Karp algorithm for the symmetric traveling-salesman problem," *Math. Program.*, vol. 7, no. 1, pp. 87–96, Dec. 1974.

[10] T. H. C. Smith and G. L. Thompson, "A lifo implicit enumeration search algorithm for the symmetric traveling salesman problem using held and karp's 1-Tree relaxation," *Ann. Discret. Math.*, vol. 1, pp. 479–493, 1977.

[11] N. Christofides, "Worst-case analysis of a new heuristic for the travelling salesman problem," 1976.

[12] M. Gendreau, A. Hertz, and G. Laporte, "New insertion and postoptimization procedures for the traveling salesman problem," *Oper. Res.*, vol. 40, no. 6, pp. 1086–1094, Dec. 1992.

[13] E. de Klerk, D. V. Pasechnik, and R. Sotirov, "On semidefinite programming relaxations of the traveling salesman problem," *SIAM J. Optim.*, vol. 19, no. 4, pp. 1559–1573, Jan. 2009.

[14] J. V. D. Veen, "Solvable cases of the traveling salesman problem with various objective functions," 1992.

[15] E. Klerk and C. Dobre, "A comparison of lower bounds for the symmetric circulant traveling salesman problem," *Discreete Appl. Math.*, vol. 159, pp. 1815–1826, 2011.

[16] R. Zamani and S. K. Lau, "Embedding learning capability in Lagrangean relaxation: An application to the travelling salesman problem," *Eur. J. Oper. Res.*, vol. 201, no. 1, pp. 82–88, 2010.

[17] B. Angeniol, G. Croix Vaubois, and J. Y. Le Texier, "Self-organizing feature maps and the travelling salesman problem," *Neural Networks*, vol. 1, pp. 289–293, 1988.

[18] B. F. J. La Maire and V. M. Mladenov, "Comparison of neural networks for solving the travelling salesman problem," in *Proc. 11th Symposium on Neural Network Applications in Electrical Engineering*, 2012, pp. 21–24.

[19] C.-K. Looi, "Neural network methods in combinatorial optimization," *Comput. Oper. Res.*, vol. 19, no. 3, pp. 191–208, 1992.

[20] J. J. Hopfield and D. W. Tank, "Neural computation of decisions in optimization problems," *Biol. Cybern.*, vol. 52, pp. 141–152, 1985.

[21] D. E. V. D. Bout and T. K. Miller, "Graph partitioning using annealed neural networks," *IEEE Transp. Neural Networks*, no. 1, pp. 192–203, 1990.

[22] S. U. Hegde, J. L. Sweet, and W. B. Levy, "Determination of parameters in a hopfield/tank computational network," *IEEE Transp. Neural Networks*, vol. 2, pp. 291–298, 1988.

[23] J. Balicki, Z. Kitowski, and A. Stateczny, "Extended hopfield models of neural networks for combinatorial multiobjective optimization problems," in *Porc. 1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98CH36227)*, vol. 2, pp. 1646–1651.

[24] B. W. Lee and B. J. Sheu, "Combinatorial optimization using competitive Hopfield neural network," *Biol. Cybernet*, no. 62, pp. 415–423, 1990.

[25] J.-Y. Potvin, "The traveling salesman problem: A neural network perspective."

[26] H. Ghaziri and I. H. Osman, "A neural network algorithm for the traveling salesman problem with backhauls," *Comput. Ind. Eng.*, vol. 44, no. 2, pp. 267–281, 2003.

[27] D. Karaboga and B. Gorkemli, "A combinatorial Artificial Bee Colony algorithm for traveling salesman problem," in *Proc. 2011 International Symposium on Innovations in Intelligent Systems and Applications*, 2011, pp. 50–53.

[28] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. the Sixth International Symposium on Micro Machine and Human Science*, pp. 39–43.

[29] J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm algorithm," in *Proc. 1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, vol. 5, pp. 4104–4108.

[30] T. Zeugmann *et al.*, "Particle swarm optimization," in *Proc. Encyclopedia of Machine Learning*, Boston, MA: Springer US, 2011, pp. 760–766.

[31] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proc. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*, pp. 69–73.

[32] E. Erkut and J. Zhang, "The maximum collection problem with time-dependent rewards," *Nav. Res. Logist.*, vol. 43, no. 5, pp. 749–763, Aug. 1996.

[33] E. Balas, "The prize collecting traveling salesman problem," *Networks*, vol. 19, no. 6, pp. 621–636, 1989.

[34] A. Ekici and A. Retharekar, "Multiple agents maximum collection problem with time dependent rewards," *Comput. Ind. Eng.*, vol. 64, no. 4, pp. 1009–1018, 2013.

[35] S. Kataoka and S. Morito, "An algorithm for single constraint maximum collection problem," *Oper. Res. Soc. Japan*, vol. 31, no. 4, pp. 515–530, 1988.

[36] A. M. Chwatal, G. R. Raidl, and nther R., "Solving the minimum label spanning tree problem by mathematical programming techniques," *Adv. Oper. Res.*, vol. 2011, pp. 1–38, 2011.

Ali Ekici was born in Istanbul, Turkey. He obtained his BSc from Middle East Technical University, Ankara in 2003, obtained his MSc and Ph.D from Georgia Institute of Technology, the USA in 2006 and 2009, respectively. His major field of study is healthcare applications, disease spread modeling, and humanitarian/healthcare logistics.

After completing his Ph.D., he has worked in the Department of Industrial Engineering at the University of Houston for four years. He is currently employed as Assoc. Prof. in Özyeğin University, Industrial Engineering Department.

Murat ÇAL was born in Istanbul, Turkey, on August 31, 1989. He was graduated from Istanbul Lisesi and obtained German Abitur Diploma in 2008. He made his BSc in Middle East Technical University, Ankara (2008-2012) and his MSc in Özyeğin University, Istanbul (2014-2017). His major field of study is transportation economics and network optimization under the major Industrial Engineering.

Following his graduation from Middle East Technical University, he began to work as Researcher and Instructor in Tubitak Tusside (Turkish Management Sciences Institute), Kocaeli, Turkey.