# A Proposed Algorithm for Finding the Minimum Cost of Traveling in the Philippines,

Arexis Emmanulle Cuervas, Zindi Nicole Otero, and Isagani B. Jos*
*Mathematics Department, De La Salle University*
**Corresponding Author: isagani.jos@dlsu.edu.ph*

**Abstract:** We propose an algorithm combining the farthest insertion algorithm and Dijkstra's algorithm. This algorithm is an application of the Traveling Salesman Problem in a graph. The vertices of the graphs are the regions of the Philippines. In each vertex, there are sub-vertices representing the various cities/provinces of that particular region. We call such graph as the regional graph of the Philippines. The main objective of the study is to generate a tour visiting all or a selected number of regions in the Philippines exactly once, with the tour having the least cost or its approximation. A computer program applying the proposed algorithm to the regional graph of the Philippines was developed. This program written in Microsoft Visual Basic 2010 (VB.net) is called Phil Trip. All eighteen regions (vertices) and cities/province containing the airports (sub-vertices) are in the database of Phil Trip.

**Key Words:** traveling salesman problem; Hamiltonian cycle; fartest insertion algorithm; Dijkstra's algorithm.

## 1. INTRODUCTION

In today's world, optimization is very much needed in various aspects; from finding the right mixture of coffee to getting the maximum profit of a business. Graph theory has various concepts that can locate the optimal solution of a particular problem using graphs.

In this study, the concept of Hamiltonian graphs will be discussed, a class of graphs studied by Sir William Hamilton. A Hamiltonian cycle is a path that visits every vertex once and whose initial and terminal vertex are the same. A graph that contains a Hamiltonian cycle is then called Hamiltonian.

One problem that requires the use of Hamiltonian cycles is the Traveling Salesman Problem (TSP). A salesman needs to go to $n$ cities to retail goods, but he wants to minimize the cost of visiting each city once. In solving the TSP , a weighted graph must be constructed. The vertices represent the cities and the edges will contain weights that give the travel cost of the salesman.

Suppose a person wants to travel around the Philippines and minimize its cost. Then, the tour that the person will take should be the cheapest, or the optimal tour. Since there is no known efficient algorithm for finding the optimal tour of a TSP [23], we can only look for the sub-optimal tour, a tour whose value is an approximation of the value of the optimal tour. This paper will find a sub-optimal tour for the tourist,

- if he/she wants to visit all the regions of the Philippines, or
- if he/she wants to visit only a certain number of regions in the Philippines.

The researchers need to find the Hamiltonian cycle with the least total weight given all the regions of the Philippines, if the traveler wants to visit all eighteen regions. Similarly, the researchers need to find the Hamiltonian cycle with the least total weight given only a selected number of regions chosen by the traveler.

A program is to be developed and it will ask the traveler for the regions he wants to visit. We will assume that the travel cost from a place, say A, to another place, say B is the same as going from B to A. Since we are to use Hamiltonian cycles, the program will present the sub-optimal tour in which the traveler will visit the regions exactly once and then return to their initial place.

Each of the eighteen regions will be represented by a vertex . Cities or provinces that contain the airports will be called the sub-vertices in their respective regions. While region IV-A does not have an airport, the researchers will make the entire region as a single vertex with respect to where its regional center is. The regional center of region IV-A is Calamba, Laguna. Only domestic and international airports are to be considered.

We propose a graph, called the regional graph, in which each region in the Philippines is represented by a vertex, and within each vertex there are sub-vertices representing cities or provinces that contain the airports in that particular region. We can see that the regional graph is a multi-graph, a graph where any two vertices may be connected by more than one edge, with respect to the vertices. However, if we consider only the sub-vertices and not the vertices, the regional graph is a simple graph.

The main data of the transportation prices will be taken from www.Rome2rio.com that was launched in April 2011. This website was voted website of the year in 2013 given by Traveltech GlobalCollect and other rewards recognized by several global industry leaders [22]. It shows multiple ways of going to and from different places and it also links the user to websites of transportation companies like Philippine Airlines and Cebu Pacific. The website gives several options in traveling from one city to another city.

For each mode of transportation, a range of fees is indicated. The researchers will determine the median in this range of values and then select the least among these. The selected value is the weight of the edge representing the travel cost between the two cities. The prices also differ from time to time because of seasonality. The data in our program were taken from October 23 to October 29, 2016.

In the TSP , it is intuitive that it will take a longer time of calculation when the input, which are the number of cities, increases. For example, if we are to generate all the distinct Hamiltonian cycles in a complete directed graph, then fixing any of the $n$ vertices of the graph we would have $(n-1)!$ ways of arranging the remaining $(n-1)$ vertices. In our case, we have a complete undirected graph. Half of the permutations will just be the reverse of another, and so there will be $(n-1)!/2$ distinct Hamiltonian cycles in the given graph.

An algorithm is considered to be efficient if its computation time grows polynomially with the size of the input [23]. So, if $n$ represents the input, the running time of an algorithm is polynomial if $n$ increases polynomially. From the example mentioned above, we see that algorithms for the TSP would have a faster growth rate.

We created a program applying the proposed algorithm. Determining the time complexity, or the amount of time for the algorithm to run as a function of the size of the input, would make it easier for us to know the running time of the program. But because an accurate calculation for a program's running time requires a lot of work and is a long process[24], we will not include it in the study.

## 2. THE PROPOSED ALGORITHM

The regional graph's optimal tour cannot be solved by using the farthest insertion algorithm (FIA) or Dijkstra's Algorithm (DA) alone. Getting a single edge that connects two regions will be easier in finding the sub-optimal tour with the FIA. The researchers want to know if a much cheaper tour can be formed, since a region is made up of numerous cities or provinces. Thus, the regional graph will be used to obtain a better and cheaper tour. The researchers will propose an algorithm combining the aforementioned algorithms to solve an approximation of the optimal tour of the regional graph.

In our proposed algorithm, the traveler will pick a sub-vertex as a starting point, which is also the end point of the tour. Furthermore, the traveler can visit at least one sub-vertex in every region chosen. Since the proposed algorithm uses the DA for the sub-vertices, then the sub-vertices as well will not be repeated. Except possibly for the sub-vertices of the initial region, because it is also the final region. Also there is a possibility for a specific region to repeat some of its sub -vertices. This region is the second one to be included in the tour after the starting region. But along the process, if a region is to be inserted between that region and the starting region, then all regions except the starting region will not have repeated sub-vertices. We will discuss later the possibility of repeated sub-vertices in those said regions after discussing the steps of the proposed algorithm. Other regions or vertices cannot be repeated since a Hamiltonian cycle is to be formed with it.

The proposed algorithm takes the form of the FIA since it is used to solve the minimum-weight Hamiltonian cycle of the regional graph. The flow of the FIA mostly depends on the selection step and insertion step. Thus, the proposed algorithm has these steps as well. In the proposed algorithm, FIA acts on the vertices. The selection step is closely the same as the one in FIA . We check all the possible edges from the regions in our sub-tour to the regions not in our sub-tour. Even if some sub-vertices are not included in the cycle, we will still check those sub-vertices because they still belong to the regions already included in the tour.

The insertion step behaves differently though quite similar. The expression $C_{ir}+C_{rj}-C_{ij}$ is an important part of this step. We have to find the two vertices $i$ and $j$ that minimizes this expression in order to insert the vertex $r$ in between them. In a simple graph, it is easy to determine these vertices. But in the regional graph, it can have two or more different results in just one set of vertices $(i,j)$. In this case, minimizing the expression $C_{ir}+C_{rj}-C_{ij}$ alone becomes hard to determine. This is where the concept of shortest path problems come in.

Similar to the FIA , we denote $C_{XY}$ as the weight of the edge or path connecting vertices $X$ and $Y$ . Let $X-Y$ be the edge or path from $X$ to $Y$ , where $X$ and $Y$ each contain one or more sub-vertices. Moreover, we represent a tour starting from $X$ going to $Y$ as $X-Y-X$ and the current unfinished tour as a sub-tour. Let $X$ represent a vertex and $X_i$ a sub-vertex in $X$. The steps of the proposed algorithm is as follows:

*Step 1*: Start with a sub-tour consisting of the sub-vertex $S$ in region $S$ only.
*Step 2*: Find the sub-vertex $F_b$ in region $F$ such that $C_{SkFb}$ is maximal, for all sub-vertices $S_k$ in *S*. Then, form the sub-tour $S-F-S$ .

*Step 3*: Given the sub-tour $S-F-S$ , and the sub-vertex $R_c$ in region *R* that is farthest from any sub-vertices in *S* and any sub-vertices in *F*.
*Step 4:* Find the path from $S_i$ to *F* that passes through *R* using the DA. This step will get the corresponding labels of the sub-vertices of *F* in the matrix as discussed in Section 2.4. In the DA, if a sub-vertex has already been visited, it cannot go back to any sub-vertex in the vertex prior to that.
*Step 5*: Use the DA to find the shortest path starting from $S_i$ directly to vertex *F*. Add the distinct labels of the sub-vertices of *F* to their corresponding labels obtained from step 4. Get the minimum sum sub-vertex, $F_s \in F$ . Using DA and the values obtained from step 4, we add to each path starting from $S_i$ to each sub-vertex in *F* passing through *R* the value of the path that ends in $S_i$ .

*Step 6*: Connect the path obtained in step 4 to the path obtained in step 5 with respect to $F_s$ to form the sub-tour $S-R-F-S$ . The paths from steps 4 and 5 both start at $S_i$ and end at the minimum sum sub-vertex $F_s$ . Combining both paths will construct a sub-tour. It is possible to repeat certain sub-vertices of *S* and *F* because of the connection of the two paths obtained from the DA.

*Note:* One or more sub-vertices from *S*, *F* and *R* can be included in the sub-tourconnected by the two paths that was obtained from the DA in steps 4 and 5. $S_i$ is both the starting and final vertex in this sub-tour.
*Step 7:* Selection Step: Given a sub-tour, find sub-vertex $Z_k \in Z$ not in the sub-tour that is farthest from any sub-vertex of the vertices in the sub-tour.
*Step 8:* **(Insertion Step)** Find the edge or path $X_i - Y_j$ , where $X_i$ is the first sub-vertex in *X* and $Y_j$ is the last sub-vertex in *Y*, in the sub-tour which minimizes $C_{X_i Z} + C_{Z Y_j} - C_{X_i Y_j}$ . Let $C_{X_i Y_j}$ be the value of the edge or path from $X_i$ to $Y_j$ . Use DA to determine the value of $C_{X_i Z} + C_{Z Y_j}$ by getting the shortest path from $X_i$ to $Y_j$ that passes through *Z*. In the DA if a sub-vertex has already been visited, it cannot go back to any sub-vertex in the vertex prior to that. After determining the minimal edge, replace it with the path $X_i - Z - Y_j$ that was obtained from the DA.

*Note:* One or more sub-vertices from *X, Y* and *Z* can be included in the path obtained from the DA.
*Step 9*: If all major vertices are added to the tour, stop. Else, go to step 7. However, step 7 will be skipped if it is only the last vertex left to be inserted.

The DA is used to find the sub-vertices in vertices *S* and *F* to be included in the tour twice, once in step 4 and once in step 5. In step 4, the shortest path from $S_i$ to $F_s$ passing through *R* is located. In that path, it is possible to visit other sub-vertices in *S* , *R* and *F* depending on the result from the DA . This is the same case in step 5 where we will locate the shortest path from $F_s$ to $S_i$ . The path from step 5 will possibly also have the same sub-vertices included in the path from step 4. Afterwards, in step 6, we will combine those aforementioned paths to create our sub-tour $S-R-F-S$ . Thus, there will be possible repeated sub-vertices in vertices *S* and *F* .

If a vertex is to be inserted between *F* and *S* then *F* will not have repeated sub-vertices anymore. This is because the insertion of a vertex between *F* and *S* will make a new path from *F* to *S* passing through the newly added vertex. This will then replace the old path from F to S obtained from steps 4 and 5 resulting to no repeated sub-vertices in F . Since S is used in the tour twice, then it is always possible to have repeated sub-vertices in S.

The proposed algorithm's steps are similar to the FIA with some modifications. We added three more steps prior to the selection and insertion steps. Also, we explained earlier the differences of the selection and insertion steps between the FIA and the proposed algorithm. The DA is used in steps 4 and 8. If there are only two regions, then the DA alone can be used. If the traveler chooses only three regions, then steps 1 to 6 only are performed to find the cheapest tour of those regions. For four or more vertices, steps 7 and 8 are looped until all vertices are included in the tour.

## 3. PHIL TRIP

The researchers will introduce the computer program called Phil Trip that applies the proposed algorithm to the regional graph of the Philippines. Phil Trip was developed using Microsoft Visual Basic 2010 (VB.Net). The data of Phil Trip, taken from www.Rome2rio.com, are stored in MS Access 2016. It acts as the storage or also known as the database. It stores the raw information that the VB.Net uses. Phil Trip will have the following:

*Inputs*

- Regions the traveler wants to travel to.
- Starting region.
- Starting city/province in the starting region.

*Outputs*

- List view of the tour generated.
- Cost from one city/province to another.
- Modes of transportation from one city/province to another.
- Total cost of the generated tour.
- Reset button going back to the start up page.

The flow of the program is as follows:

1. Check the regions that you want to travel to or check 'visit all' to select all regions.

2. At the drop down box under Starting Region, it will show the regions that you selected in (1). Select the region where you want to start/end your tour.

3. At the drop down box under Starting City or Province, it will show the cities/provinces in your starting region in (2). Select the city/province you want to start/end your tour.

4. Once you are satisfied with your selections, click the button 'Generate Itinerary' then click yes.

5. The final tour is shown in list form along with its cost and mode of transportation from one city/province to another, and the total cost of the tour.

6. If you want to use the program again, click the

Reset button to go back to the start up page.

Results should be discussed thoroughly but concisely in this section with the aid of figures and tables whenever necessary.

The main form or the start up page is shown in Figure 3.1

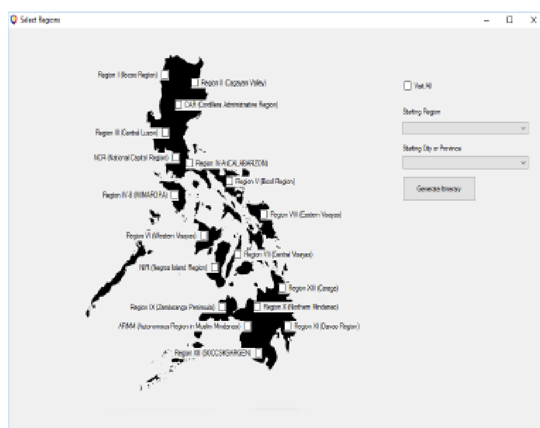
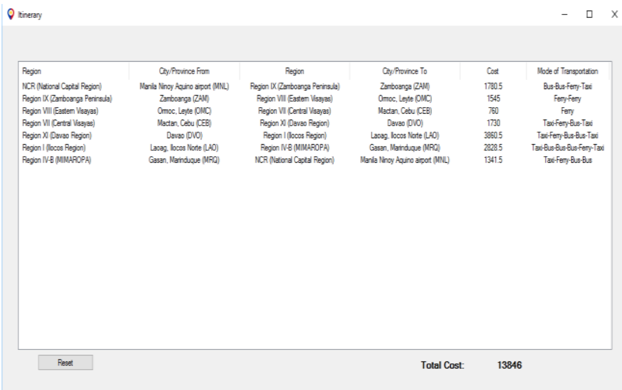


**Figure 3.1**. The start up page

An example of the output page is shown in Figure 3.2. The regions selected were Regions I, IV-B, VII, VIII, IX, XI, and NCR. The starting sub-vertex is Manila , which is in NCR.




| Region | City/Province From | Region | City/Province To | Cost | Mode of Transportation |
|---|---|---|---|---|---|
| NCR (National Capital Region) | Manila Ninoy Aquino airport (MNL) | Region IX (Zamboanga Peninsula) | Zamboanga (ZAM) | 1780.5 | Bus-Bus-Ferry-Taxi |
| Region IX (Zamboanga Peninsula) | Zamboanga (ZAM) | Region VIII (Eastern Visayas) | Ormoc, Leyte (OMC) | 1545 | Ferry-Ferry |
| Region VIII (Eastern Visayas) | Ormoc, Leyte (OMC) | Region VII (Central Visayas) | Mactan, Cebu (CEB) | 760 | Ferry |
| Region VII (Central Visayas) | Mactan, Cebu (CEB) | Region XI (Davao Region) | Davao (DVO) | 1730 | Taxi-Ferry-Bus-Taxi |
| Region XI (Davao Region) | Davao (DVO) | Region I (Ilocos Region) | Laoag, Ilocos Norte (LAO) | 3860.5 | Taxi-Ferry-Bus-Bus-Taxi |
| Region I (Ilocos Region) | Laoag, Ilocos Norte (LAO) | Region IV-B (MIMAROPA) | Gasan, Marinduque (MRQ) | 2828.5 | Taxi-Bus-Bus-Bus-Ferry-Taxi |
| Region IV-B (MIMAROPA) | Gasan, Marinduque (MRQ) | NCR (National Capital Region) | Manila Ninoy Aquino airport (MNL) | 1341.5 | Taxi-Ferry-Bus-Bus |

**Figure 3.2.** A sample run of the program

## 4 SUMMARY, CONCLUSION, AND RECOMMENDATIONS

Two algorithms, the FIA and DA, were combined in order to find a Hamiltonian cycle with the least total weight in a graph we introduced. Our graph, the regional graph, has the regions of the Philippines as its vertices and the airports in the cities/provinces within those regions as the sub-vertices. The FIA determines the sequence of the vertices/regions in a minimal tour. We use it to find the best possible ordering of the regions that yields an approximation of the optimal tour. Thus, the flow of the proposed algorithm is mapped with the FIA with some improvements. Also, since there are sub-vertices within the vertices of the graph, then the cost of going to and from a certain region is not unique. Hence, the DA acts as the solution for us to know what sub-vertex/sub-vertices should be included in the tour that gives the least cost or close to it. For a faster implementation of the proposed algorithm, we developed a program called Phil Trip.

The proposed algorithm solves for the regional graph's minimum-weight tour or an approximation of it. The FIA and the DA fit with each other in solving the TSP of this graph. Both algorithms finding the least cost in their respective fields are needed in the regional graph, the FIA and DA are for the vertices and sub-vertices, respectively. Combining them yielded a feasible algorithm in solving the TSP in the regional graph.

The proposed algorithm's efficiency depends on the two algorithms but mostly on the FIA , since the proposed algorithm has the same flow as the FIA . The proposed algorithm gives the traveler an answer, but the accuracy of this result is unsure. For future studies, a comparison of the proposed algorithm's results to other methods of solving the TSP in graphs like the regional graph can be done, if there would be any.

One way to further improve the proposed algorithm, including the program, is to lessen its limitations. Phil trip is a program wherein a traveler inputs the regions he wants to include in his tour. The program itself is the one that determines what sub-vertices are to be included. In order for us to not limit what the traveler can visit in a region, we recommend for future studies to give the traveler the freedom to choose the cities or provinces he will visit, along with their respective regions. In our data, we based the sub-vertices on the airports within a region. Thus, not all cities or provinces are considered. We recommend including other cities or provinces in the data, although it would be hard to collect a specific cost for a city/province since the place where the traveler will be going is not specific within that city or province. This is the reason why we chose to base on airports, since considering a whole city/province would appear to be a new vertex and again have sub-vertices within them.

The goal of the algorithm is to find the cheapest Hamiltonian cycle, hence the cheapest mode of transportation were collected for data. So, the traveler's mode of transportation is also assigned by the program and oftentimes the choice of riding the plane, which is more convenient, is disregarded. Time is also ignored by the program since the data gathered are only for transportation costs. Hence, the traveler does not have a choice but to travel the cheapest way, even though it might result to a longer travel time. Another recommendation would be allowing the traveler to choose what mode of transportation he wants along with his travel time limit and still attain the cheapest Hamiltonian cycle given the modes of transportation he wanted. In addition, for the traveler to clearly see and picture the tour created, a sketch of the tour on the map of the Philippines can be included in the output page.

## 5. REFERENCES

[1] Harary, F. (1969). Graph Theory. Reading, Massachusetts: Addison Wesley.

[2] J. Gross and J. Yellen,Graph theory and its applications. Boca Raton: CRC Press, 1999.

[3] A. Dharwadker, A new Algorithm for finding Hamiltonian circuits. Institute of Mathematics H-501 Palam Vihar District Gurgaon Haryana 1 2 2 0 17 India, 20014.

[4] J. Ong and M. Si, Maffersons algorithmic application to Philippine voyager 1998, 1998.

[5] N. Ebanos and S. Florendo, On finding the cheapest and fastest route, De La Salle University–Manila, 2008

[6] S.-A. Kim and J. Tan, Finding the minimum cost of an airline route using the traveling salesman problem, 2014.

[7] Hamiltonian graphs, Cybernetica. [Online]. Available: http://research.cyber.ee/ peeter /teaching/graad08s/previous/loeng3eng.pdf.

[8] D.West, Introduction to graph theory. Prentice Hall:Upper Saddle River, NJ, 1996.

[9] W. D. Wallis, The traveling salesman problem, in a beginners guide to discrete mathematics, Springer Science + Business Media, LLC, 2003, p. 245

[10] Traveling salesman problem. (n.d.). Retrieved from http://www2.isye.gatech.edu/ mgoetsch/cali/VEHICLE/TSP/TSP015:HTM

[11] Syso, M. M., Deo, N., Kowalik, J. S. (1983). Discrete optimization algorithms: With pascal programs. Englewood Clis, NJ: Prentice-Hall.

[12] Lambert, J., Lee, D., Laycock, L., Morgan, E. (n.d.). TSP comparison project. Retrieved from http://web.stanford.edu/ johnwl/ FarthestInsertionP resentation:pdf

[13] Goddard, W. (2004). Introduction to algorithms. Retrievedfromhttps://people.cs.clemson.edu/ goddard/texts/cpsc8400/part3.pdf

[14] Errickson, J. (2014). Algorithms. Retrieved from http://jee.cs.illinois.edu/teaching/algorithms/ notes/30-nphard.pdf