Optimal Placement of Refugee Hot-Spots on Greek Islands

DIMITRA ALEXIOU[1], EVLAMPIA ATHANAILIDOU[2]

[1]Department of Spatial Planning and Development, School of Engineering,
Aristotle University of Thessaloniki, Thessaloniki, GREECE
Email: dimitraalexiou@plandevel.auth.gr

[2]Department of Mathematics, Aristotle University of Thessaloniki,
Thessaloniki, GREECE
Email: athanailidoueva@yahoo.gr

Abstract: In this paper, we propose an algorithm to recommend Greek Islands that can host refugees. The algorithm considers two conditions. The islands must be close to each other so that the transportation can easily serve the refugees without extra transportation costs and the cost of the new infrastructures is the minimum. The results showed that the Greek islands could host more than 100k refugees without suffering from overcrowding since the number of refugees will be significantly smaller than the number of locals. There is a mass influx of refugees in Greece for several reasons, such as the war taking place in developing countries or poverty. Refugees seek safe places for a better life with health and education systems, job opportunities, etc. However, this creates problems in small countries, which have not dealt with such issues before.
One such problem is the accommodation for these people.

Key-Words: - Graph Theory, transportation cost, Greek region

1 Introduction

In recent years, the European countries have been suffering from the immigration wave since every single day people pass their borders. Although the refugees seek safe places with better opportunities for themselves and their families, countries in the European Union have difficulty serving them and providing a better future for them. Greece is one country that has been influenced much more than the other countries in European Union since it is close to the borders from where most refugees go through to get into Europe. The main problem is the accommodation for these people in Greece, since the country is too small to select appropriate places to transfer the refugees without affecting the local regions.

To solve this problem, we propose an algorithm that recommends Greek islands where we could transfer a number of refugees. The main characteristic of the algorithm is that it locates refugees searching for a subset of islands close to each other so that the infrastructure costs can be the minimum possible. In this way, the transportation costs between the islands will be the minimum,and the refugees' service will be faster. Another characteristic is that the algorithm locates refugees based on a percentage of the local population which means that the refugees will not be more than this percentage on an island. In this way, we try to avoid overcrowding in the islands, which would affect the life of the locals. The background for this study is correlated with the traffic signal control and design [10],[11],[12],[13],[14].

The rest of the paper is structured as follows. In section 2, we describe the method used to solve the problem of transferring of refugees to the Greek Islands. Next, in section 3, we mention the experimental setup. Section 4 presents the results of this study. Section 5 presents a case study after executing the algorithm. Finally, section 6 concludes this work and proposes future directions.

2 Algorithm

The problem of transferring refugees to the Greek islands is an optimization problem [15],[16] since our aim is to minimize the total cost of the needed infrastructure to host the refugees. In parallel, we have to select a subset of islands where the islands have the minimum distance between them [1],[2]. Each island has more than one ports and when we calculate the distance between the two islands, we consider the distance between the two ports. If we select a port for the infrastructure, then the other ports of the island cannot be chosen. The assumption here is that we can establish only one infrastructure at each island.

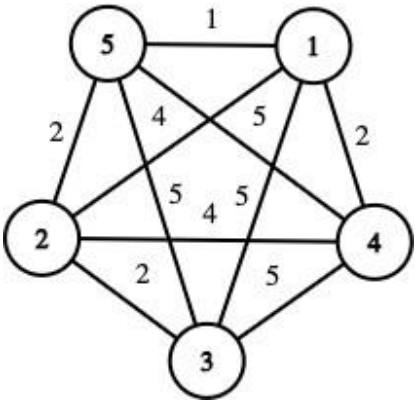

Figure 1: A graph example for describing the algorithm

The costs are calculated considering the individual costs as seen below:

1. Building Structure Cost: For each new building, the cost is $d + b * M$ where d is the cost of
   the building b is the cost per refugee and M is the total number of refugees.
2. Installation Costs: It depends on the density of an island p and a cost that the government gives as compensation to the island e. It is estimated as $p * e$. The density of an island can be calculated by considering the division between the total population of the island with the total area of the island in km^2.

As a graph-based problem following the graph theory [17],[18], we consider an undirected fully connected graph $G = (V, E)$ where V is the set of vertices of the graph representing the ports of the islands and E the set of edges that connects the nodes. Each edge has a weight that corresponds to the distance between two ports. Iteratively, we select a source vertex in the graph and search the minimum possible path between the vertices visiting the vertex which is closest to the source vertex. The decision of selecting the next vertex is crucial since the problem can be considered as NP-Complete [5]. Finally, we select the set of vertices that belong to the path with the minimum total cost. Figure 1 illustrates an example of a graph of 5 vertices and weights to their edges. In each iteration, we select a source vertex starting from the vertex with label 1. The vertex with label 1 is added to the set of nodes that we have visited, the cost of an infrastructure is added to the total cost and the closest node (label 5) is added to a stack. The other ports belonging to the same island (e.g. node with label 4) are also added to the set of nodes that we have visited. Next, we pop the node from the stack, and we add it to the path the node is closest to (label 2). We follow this procedure until we transfer all the refugees to the islands. If there are a few refugees available, it is important not to create a new infrastructure since the cost is high for just a few refugees. Instead of creating a new infrastructure, we transfer the refugees in an island with the lowest density which belongs to the current path. We choose such an island since we have already transferred refugees there but there is still plenty of room to transfer many more. Stack is necessary in the case where there are more than one paths to follow. For example, if we select the node with label 3 as the source node, there are two nodes close to it (labels 1,5) and it is possible for two paths to be created, but one of them will have the minimum total cost. As a result, we have to

follow the one path considering one of these nodes and then backtrack to follow the second path to find that path with the minimum total cost [6],[7]. Stack implementation can be accomplished using recursion. In detail, we can use a function considering the base cases and the recursive step.

The algorithm ends when we have selected all nodes as source nodes, and we have found the one path that has the minimum total cost. Below, we mention the algorithm in pseudocode.

Algorithm 1: Returns the path of a Graph with the minimum cost.

Input	:	A graph $G = (V, E)$, M number of refugees
Output	:	The path = $(a_1, a_2, \dots, a_k)$ where a_i belongs to the path where the total cost is minimum.
Step 1	:	$cost \leftarrow \infty$
Step 2	:	$path \leftarrow \emptyset$
Step 3	:	for sourceN$ode \in G$:
Step 4	:	$currentPath, currentCost, \_, \_ \leftarrow findPath\ Cost(sourceNode, G, cost, M, 0, \emptyset, \emptyset)$
Step 5	:	$if\ currentCost < cost$:
Step 6	:	$cost \leftarrow currentCost$
Step 7	:	$path \leftarrow \emptyset$
Step 8	:	$path \leftarrow currentPath$
Step 9	:	$endif$
Step 10	:	$endfor$

The *Algorithm 1* gets as input the graph with the ports with the weighted edges and the total number of refugees. The output of the algorithm is the path with the nodes of the graph that are close to each other and the total cost is minimum.

The recursive findPathCost function is responsible to find the path given the source node and the current total cost. Cost is used further to cut some paths in cases where the total cost of the new path tends to be higher than the cost of a previous path. The other parameters are discussed later in this section. Function returns more than 2 values, however in Algorithm 1, only two are necessary, while the other two are important during recursion. In step 5, the algorithm checks the cost of the new path returned by the function with the previous cost of the previous path. If the current cost is lower than the previous one, we update the path and the cost. Otherwise, the variables do not change meaning that the previous cost is lower than the current one. The algorithm ends when there is no other sourceNode in the graph implying that all nodes in the graph[6],[7] have become source nodes for the findPathCost function. Next, we present the algorithm for the function.

Algorithm 2: Returns the path with nodes with the minimum distance between them and the total cost for this path given a source node.

Input	:	A source node S, G graph, $cost$, M number of refugees, $currentCost$, $currentPath$, $visited$
Output	:	The path = ($S, a_2, \dots, a_k$) starting from the S and a_i belongs to the path where the total cost is minimum. The path cost
Step 1	:	$if\ M = 0\ or\ currentCost > cost$:
Step 2	:	$return\ currentPath, currentCost, M, visited$
Step 3	:	$else\ if\ M < Mth\ resh\ old$:
Step 4	:	minDensity, minNode ← min (currentPath)
Step 5	:	$minNode.add(M)$
Step 6	:	$currentCost \leftarrow currentCost + b * M$
Step 7	:	$return\ currentPath, currentCost, M, visited$
Step 8	:	$else$:
Step 9	:	$currentPath \leftarrow currentPath \cup \{S\}$
Step 10	:	$visited \leftarrow visited \cup \{S\}$
Step 11	:	$for\ port \in Island(S).getPorts()$:
Step 12	:	$visited \leftarrow visited \cup \{port\}$
Step 13	:	$endfor$
Step 14		$tmpM \leftarrow 0;\ x \leftarrow population(Island(S)) * a$
Step 15	:	$if\ x > c$:
Step 16	:	$tmpM \leftarrow c$
Step 17	:	$else$:
Step 18	:	$tmpM \leftarrow round(x)$
Step 19	:	$endif$
Step 20	:	$if\ tempM > M: tempM \leftarrow M;$
Step 21	:	$M \leftarrow M - tmpM$; currentCost ← tmpM * b + d + density(Island(S)) * e
Step 22	:	$if\ M > 0$:
Step 23	:	$closestNodes \leftarrow findClosestNodes(S, G, visited)$
Step 24	:	$if\ closestNodes \neq \emptyset$:
Step 25	:	$tCost \leftarrow 0; tPath \leftarrow 0; tM \leftarrow 0; tVisited \leftarrow 0;$
Step 26	:	$for\ closestNode \in closestNodes$:
Step 27	:	$cPath, cCost, cM, cVisited \leftarrow findPath\ Cost(closestNode, G, cost, currentCost, currentPath, visited, Mth\ r.)$

Step 28 : $if\ cCost < tCost$:

Step 29 : $tCost \leftarrow cCost; tPath \leftarrow cPath; tM \leftarrow cM; tVisited \leftarrow cVisited$

Step 30 : $endif$

Step 31 : $endfor$

Step 32 : $currentCost \leftarrow tCost; currentPath \leftarrow tPath; visited \leftarrow tVisited; M \leftarrow tM;$

Step 33 : $endif$

Step 34 : $endif$

Step 35 : $return\ currentPath, currentCost, M, visited$

Step 35 : $endif$

Algorithm 2 describes the steps for the function which finds the best path based on the criteria,we have selected. The input of the algorithm is the source node, where the algorithm starts to search to establish the infrastructure, the number of refugees which is important since the algorithm will end when all refugees have been transferred to the islands, the graph in which the algorithm will search for another node to be visited and the cost which will be used as a threshold for the total cost of the current path. If the current cost is greater than the cost, then the path is not better than a previous one[3].The other three parameters ($currentCost$, $currentPath$, $visited$) are used for the recursive nature of the algorithm. The first one tracks the changes onthe total cost when a new node is added to the path, the second one stores the current path and the third one stores the nodes that we have already visited and we do not have to visit them again. In the algorithm, there are free parameters that can be set during parameters tuning process named a, b, c, d, e, $Mthreshold$ which corresponds to the percentage of the island people under which we can transfer refugees, b is the cost for each refugee, c is the maximum number of refugees that can be transferred to an island, d is the cost of the building that will host the refugees, e is an extra cost for each amount of islands density, and Mthreshold is the number of refugees under which we don't create a new infrastructure but instead we transfer the rest of the refugees to the island with the lowest density. The algorithm has two base cases. In the first case, the function returns either when all the refugees have been transferred to the islands or if the current cost is greater than the cost of a previous path. The second case checks whether there are few refugees to transfer. In this case, we search for the island with the minimum density, and we transfer the refugees there, further we update the current cost calculating the costs per refugee. The third case is the recursive step. In this case, the current visited port along with the ports ofthe same island are added to the set of visited nodes since the algorithm does not have to visit them again. Then, the number of refugees that the island can host is subtracted by the number of refugees and the cost is updated properly. The cost is calculated estimating the cost per refugee, the cost of the building which will host the refugees and the cost based on the island density.

Next, if there are refugees that have to be transferred, we search for the closest node(s). Each closest node pass in the recursive function. The function returns the path, cost, visited set and M of the path that is created by the closest node of the current node. If there are multiple closest nodes, then the condition in step 28 guarantees that we will choose that path with the minimum total cost. Finally, in step 35, the current path, cost, visited set and M are returned.

3 Experimental Setup

We consider 100k refugees that we want to transfer to Greek Islands, the costs are the same as the ones described in the experimental setup section and the parameter of the percentage of the island people under which we can transfer refugees is defined as 15%.

Figure 6 illustrates the path that is extracted from our method. The source node is from a port in the island of Crete while the last destination is the port in the island of Evia. Table 3 shows the path starting from the source Island and moving to the destination island. For each source island, we also estimate the number of refugees that will be transferred[8],[9]. Crete is a large island with large number of locals, thus we can transfer the maximum accepted number of refugees there. We observe that the algorithm also chooses small islands to transfer refugees. For example, it is possible to transfer just 457 refugees to Patmos and 744 refugees to Skopelos. The decision of the algorithm depends on the condition that we defined to choose the next closest island. Thenumber of islands depends significantly on parameter a since if we choose a to be equal to 0.20 then the total number of islands would 15 instead of 16 and the path will be different if starting from the Evia island.

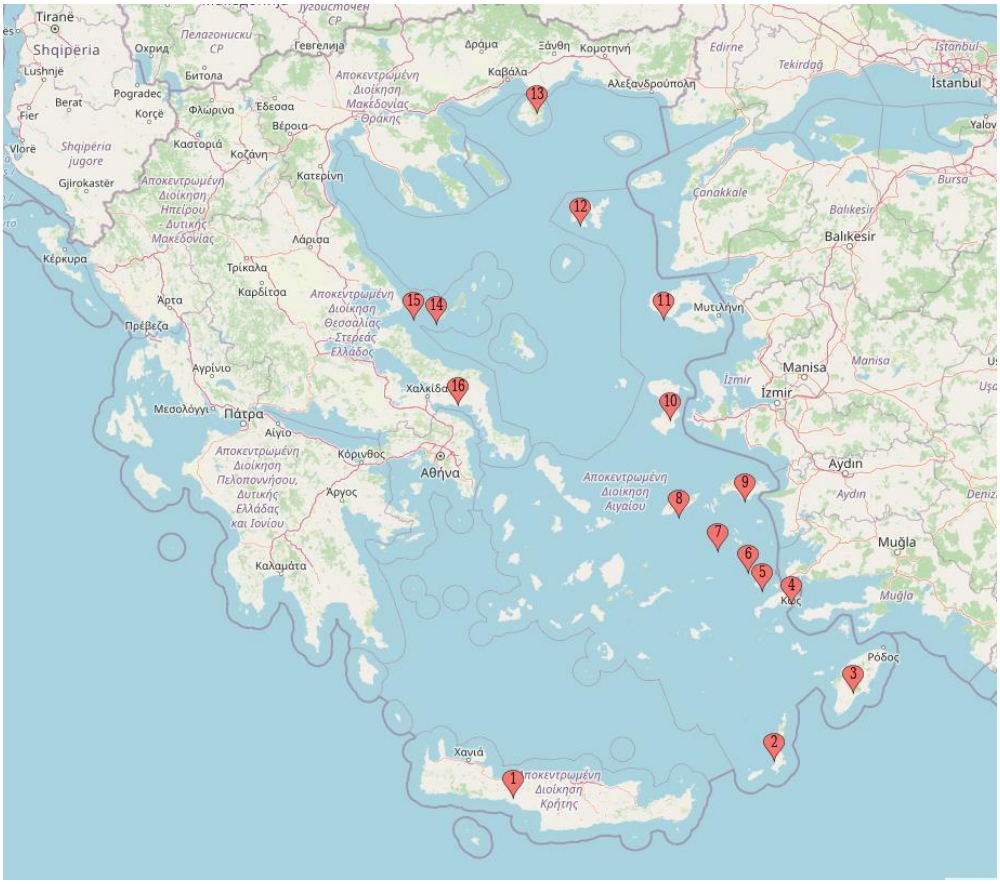

Figure 7: The path from the Crete to Evia for transferring 100k refugees.

Source Island	Destination Island	#refugees in Source Island
Crete	Karpathos	20000
Karpathos	Rodos	933
Rodos	Kos	17323
Kos	Kalimnos	5008
Kalimnos	Leros	2426
Leros	Patmos	1187
Patmos	Ikaria	457
Ikaria	Samos	1263
Samos	Chios	4946
Chios	Lesvos	7698
Lesvos	Limnos	12799
Limnos	Thasos	2550
Thasos	Skopelos	2065
Skopelos	Skiathos	744
Skiathos	Evia	991
Evia	-	19610

Table 3: The path from the source island to destination island. For each source island, we mention the total number of refugees that can be transferred.

4 Conclusion

In this paper, we present an algorithm so that we can recommend Greek islands to transfer refugees. The algorithm considers two conditions [16] the islands have to be close to each other so that the transportation will be easy to service the refugees better without any extratransportation costs and [4] the total cost of the new infrastructures will be the minimum. The results showed that the number of refugees selected for each island is important to decide the transferring of all the refugees to the islands. The significant costs for this movement are related to the maximum number of refugees that will be transferred to the islands and the costs of the infrastructure including the buildings.

References:

[1] Aho, A., Hopcroft, J., Ullmanm, J., *Data Structures and Algorithms*, Addison-Wesley, Boston, 1987.

[2] Alexiou, D., Katsavounis, S., In: Springer Proceedings in Mathematics & Statistics, *Determining the minimum number of ware-houses and their space- size for storing compatible items, optimization theory, decision making, and operations research applications*, 2013, pp. 189–198.

[3] Christofides, N., Academic Press, *Graph Theory. An Algorithmic Approach,* New York, 1975

[4] Diakaki, C., Papageorgiou, M., Aboudolas, K., Control Eng. Pract. 10(2), *A multivariable regulator approach to traffic-responsive network-wide signal control,* 2002, pp. 183–195.

[5] Garey, M.R., Johnson, D.S., *Computer Intractability: A Guide to the theory of NP-Completeness*, Freeman, San Francisco, 1979.

[6] Harary, F., *Graph Theory,* Addison Wesley, Boston, 1969.

[7] Jinwoo, L., Baher, A., Amer, S., Eui-Hwan, C., J. Intell. Transp. Syst. 9(3), *Real-time*

optimization for adaptive traffic signal control using genetic algorithms, 2005.

[8] Mathew, T., Krishna R., NPTEL, Chap. 41, *Traffic signal design-I*, 2006. www.cdeep.iitb.ac.in/nptel/Civil%20Engineering/Transportation%20Engg%201/41-Ltexhtml/nptel_ceTEI_L41.pdf

[9] Michaels, J.G., Kenneth, H.R., McGraw-Hill Inc, *Application of Discrete Mathematics*, New York, 1991.

[10] Opsut, R.J., Roberts, F.S., Networks 13, *I-Colorings. I-Phasings and I-intersection assignments for graphs, and their applications*, 1983b, pp. 327–345.

[11] Raychaudhuri, A., Discrete Appl. Math 40(3), *Optimal multiple interval assignments in frequency assignment and traffic phasing*, 1992, pp. 319–332.

[12] Roberts, F.S., *Some applications of graph theory. Department of Mathematics and DIMACS, Rutgers University*, draft October 1 2000. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.76.5815&rep=rep1&type=pdf

[13] Stoffers, K.E., *Scheduling of traffic lights–a new approach*, Transport. Res. 2, 1968, pp. 199–234.

[14] *Traffic Signal Design*, Section 7, Phasing and Signal Groups, Department of Transportation, Minnesota, 2013. http://www.dot.state.mn.us/trafficeng/publ/signaloperations/2013_Signal_Opt_and_Timing_Manual.pdf

[15] *Traffic Signal Design*, Section7, Phasing and Signal Group display sequence, Roads and Traffic 2011.

http://www.rms.nsw.gov.au/documents/business-industry/partners-and-suppliers/guidelines/complementary-traffic-material/tsdsect7v12-i.pdf

[16] *Traffic Signal Guidelines*, Boston Transportation Department, Massachussets, 2004. https://www.cityofboston.gov/transportation/pdfs/traf_signal_oper_design_guide.pdf

[17] *Traffic Signal Operation Handbook*, Texas Transportation Institute, The Texas A&M University System, Research and Technology Implementation Office, Austin, Texas, 2009. http://d2dtl5nnlpfr0r.cloudfront.net/tti.tamu.edu/documents/0-5629-P1.pdf

[18] Van Rossum, G., & Drake Jr, F. L. Python tutorial, *Centrum voor Wiskunde en Informatica*, Amsterdam, Vol. 620, 1995.