Algorithm Performance and Time Complexity

Leah A. Chrestien

August 21, 2017

1 Filtered Output

Of all the coordinates returned by the travellingSalesman algorithm(see Network.java, line 228) at some distance from the chosen centre, the coordinates that are separated by a distance of less than 1 mile are removed. This avoids an overcrowding of nodes and reduces the risk of running into a non-convex polygon boundary.

Determining the shortest route between a given set of points without repeating any point is a well-known problem in Mathematics and is no easy task. While there exists many approaches to sort coordinates in a circular order, all such approaches have their own drawbacks and the accuracy decreases as the number of points keep on increasing[1]. The range-anxiety project uses the nearest - neighbour approach[2] to sort the coordinates in a circular order. When the number of coordinates exceed 150, the algorithm fails to exhibit the best of results and there arises a need to filter the coordinates.

A brute-force filtering method(see Network.java, lines 272-293) is performed on the resultant array returned by the travellingSalesman algorithm. It takes the first coordinate (having a unique key) and selects(using the Haversine function) the very next coordinate from the resultant array which is at a distance of more than 1 mile. This coordinate then becomes our new coordinate of reference and the algorithm performs this selection process to determine the final set of coordinates. Section 2 illustrates 5 results along with the respective API's.

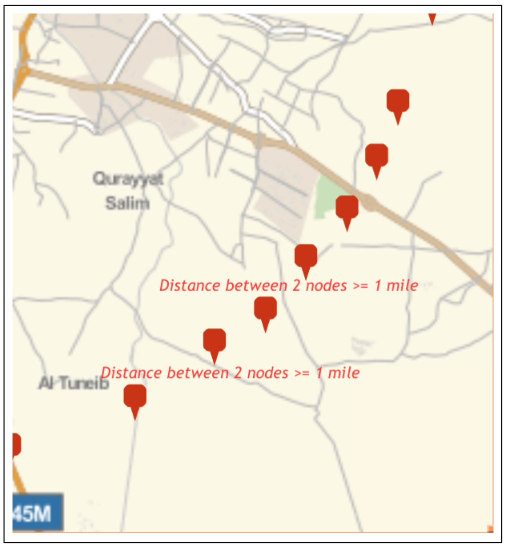
Qurayyat Salim
Distance between 2 nodes >= 1 mile
Al-Tuneib
Distance between 2 nodes >= 1 mile
45M

Figure 1: Node seperation.

2 Results

In Figures 2-6, the figure on the left illustrates the case when the distance between any two coordinates exceeds 1 mile. The figure on the right illustrates the case when the distance between any two coordinates exceeds 2 miles. One can notice that in each case, the figure on the left has more markers than the figure on the right.

Case 1:
http://localhost:8111/greennav/polygon?startlat=31.7239898&startlng=35.6429683&range=11.0

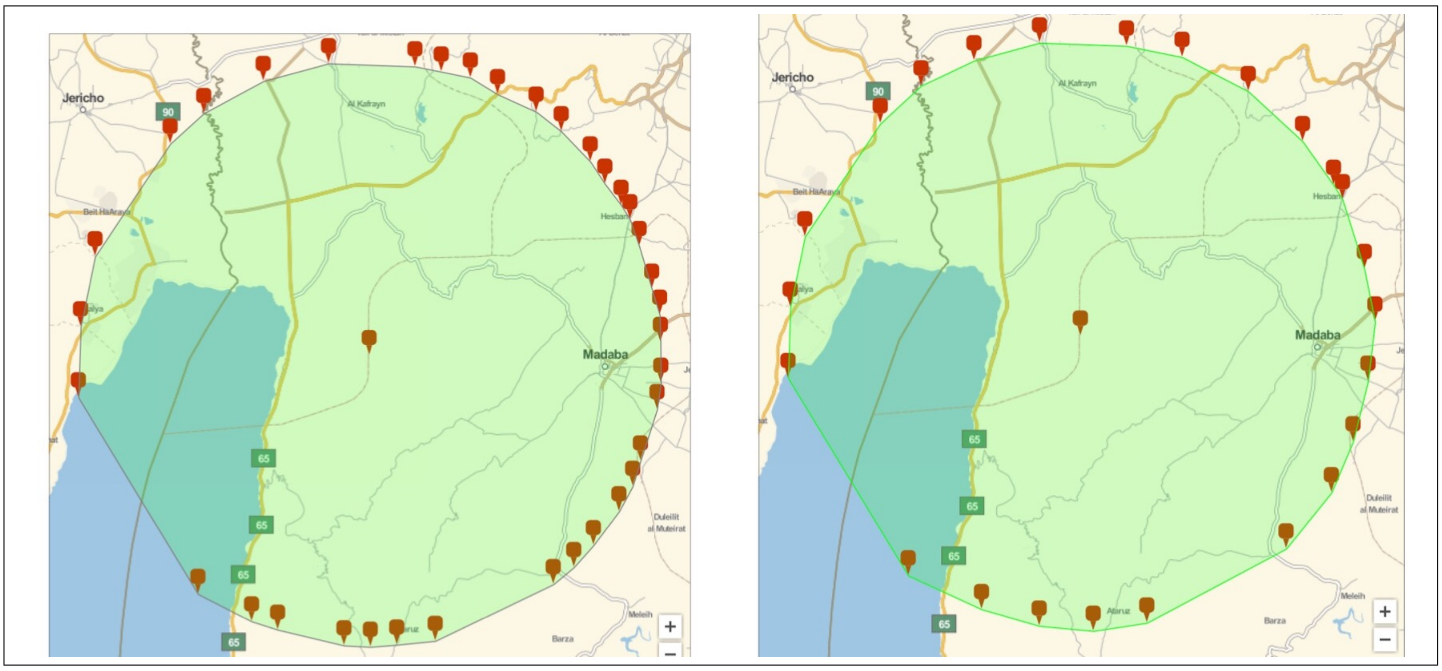

Figure 2: Centre coordinate: [31.7239898,35.6429683]

Case 2:
http://localhost:8111/greennav/polygon?startlat=32.550243&startlng=35.7073606&range=20.0

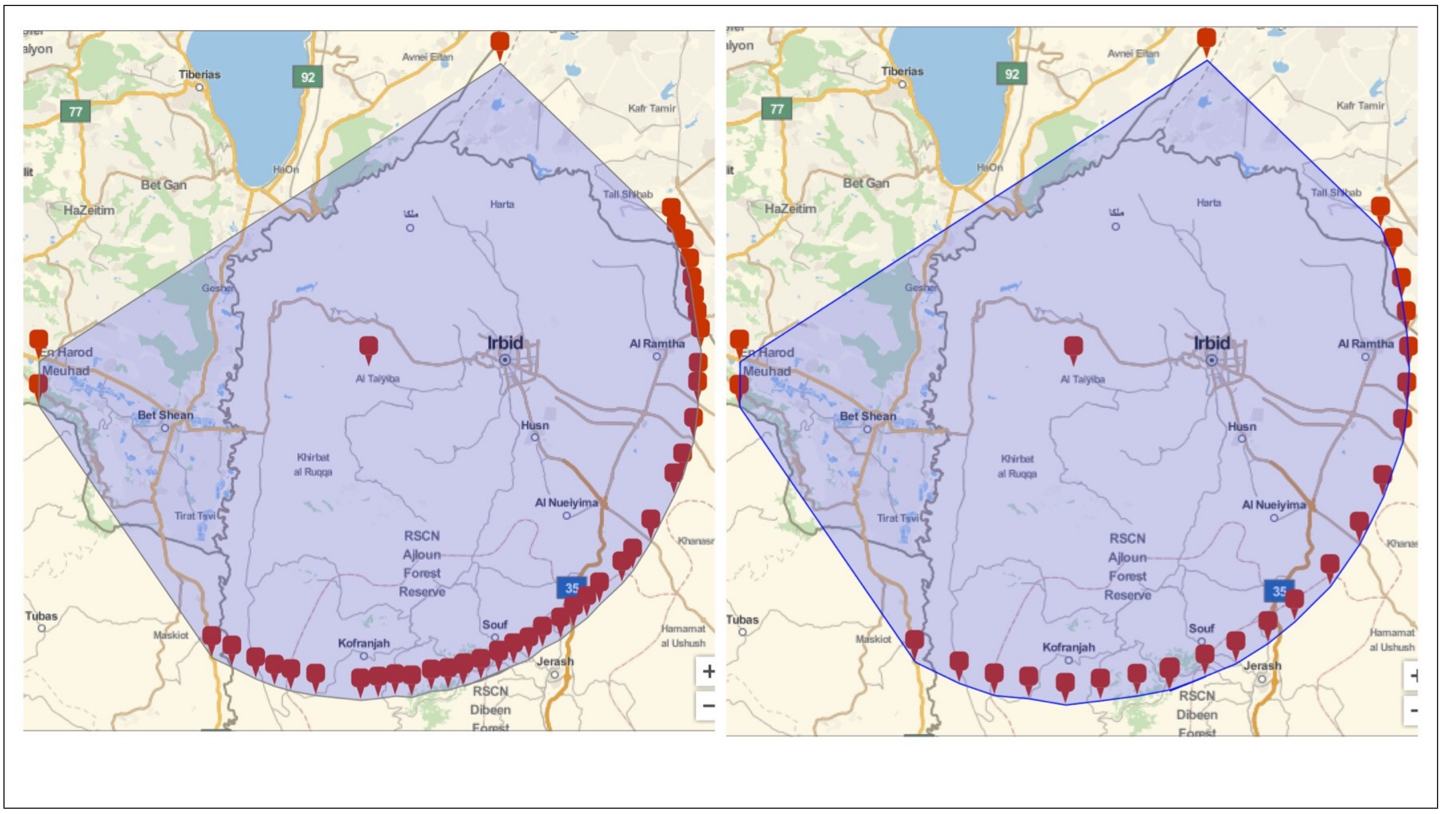

Figure 3: Centre coordinate: [32.550243,35.7073606]

Case 3:
http://localhost:8111/greennav/polygon?startlat=31.8690864&startlng=35.7073606&range=25.0

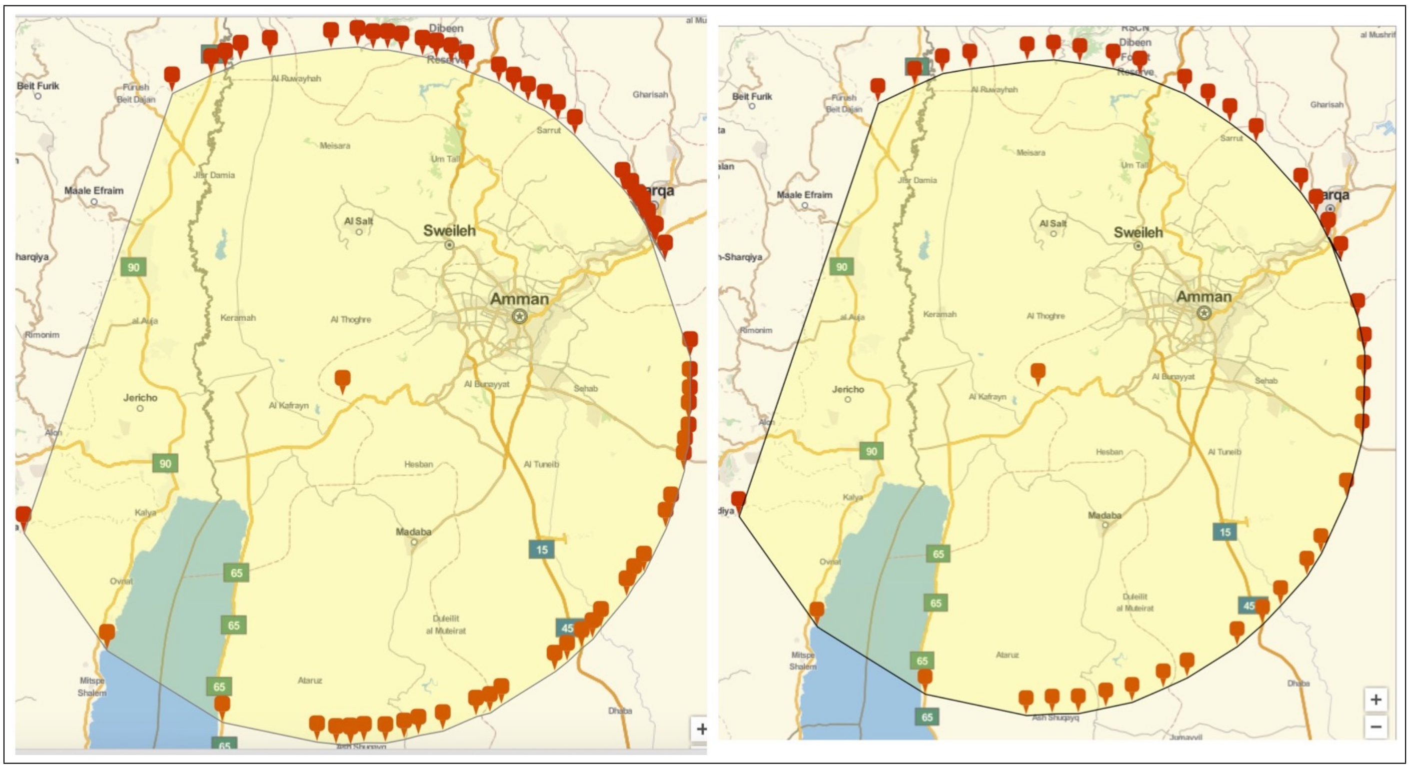

Figure 4: Centre coordinate: [31.8690864,35.7073606]

Case 4:
http://localhost:8111/greennav/polygon?startlat=31.917632100000002&startlng=35.891095&range=30.0

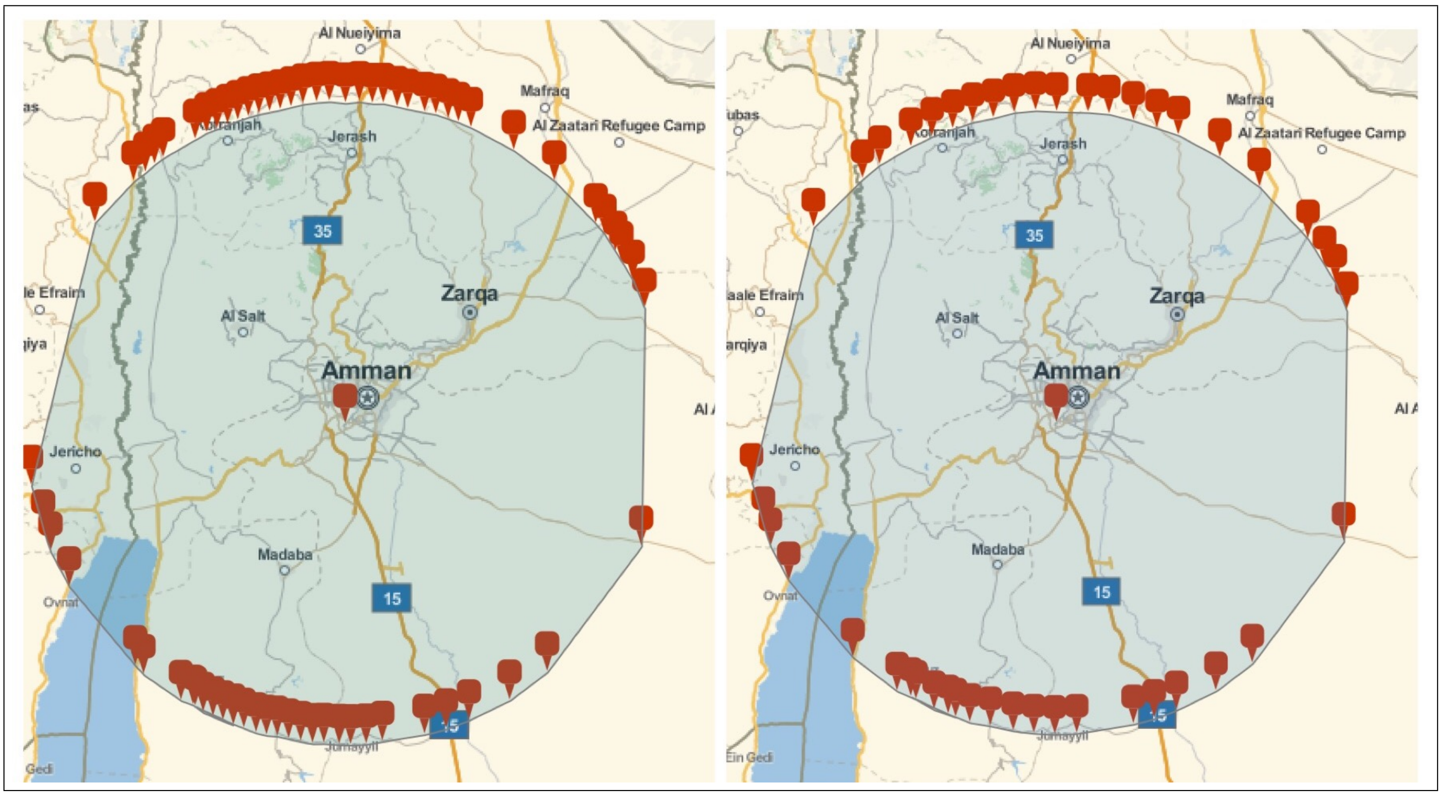

Figure 5: Centre coordinate: [31.917632100000002,35.891095]

Case 5:
http://localhost:8111/greennav/polygon?startlat=31.3842340003&startlng=35.66180160004&range=30.0

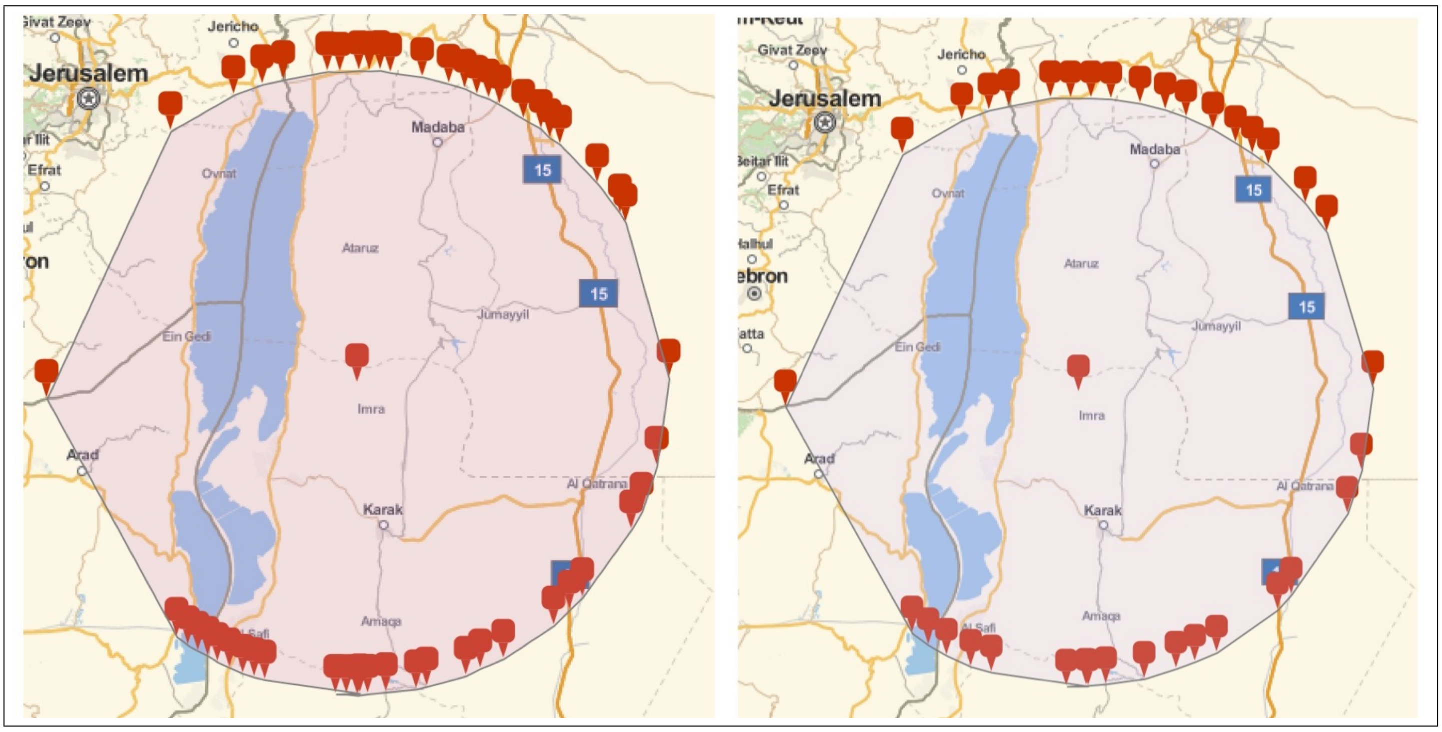

Figure 6: Centre coordinate: [31.384234000000003,35.661801600000004]

While Figures 2-6 form neat boundaries, there is no guarantee that this will be achieved every time the user changes the value of the coordinates or the range. Sometimes, for the same coordinate, the polygon boundary may or may not intersect depending on the value of the range. This is illustrated in Section 3.

3 Uncertain Polygon boundaries

When seen in a clockwise-order, Figure 7 illustrates **Case 4** from Section 2 with 4 four different values of range. It is seen that as the range in increased, the intersection gradually disappears. In a clockwise order, the range associated with the four figures are 15, 20, 25 and 30 miles.

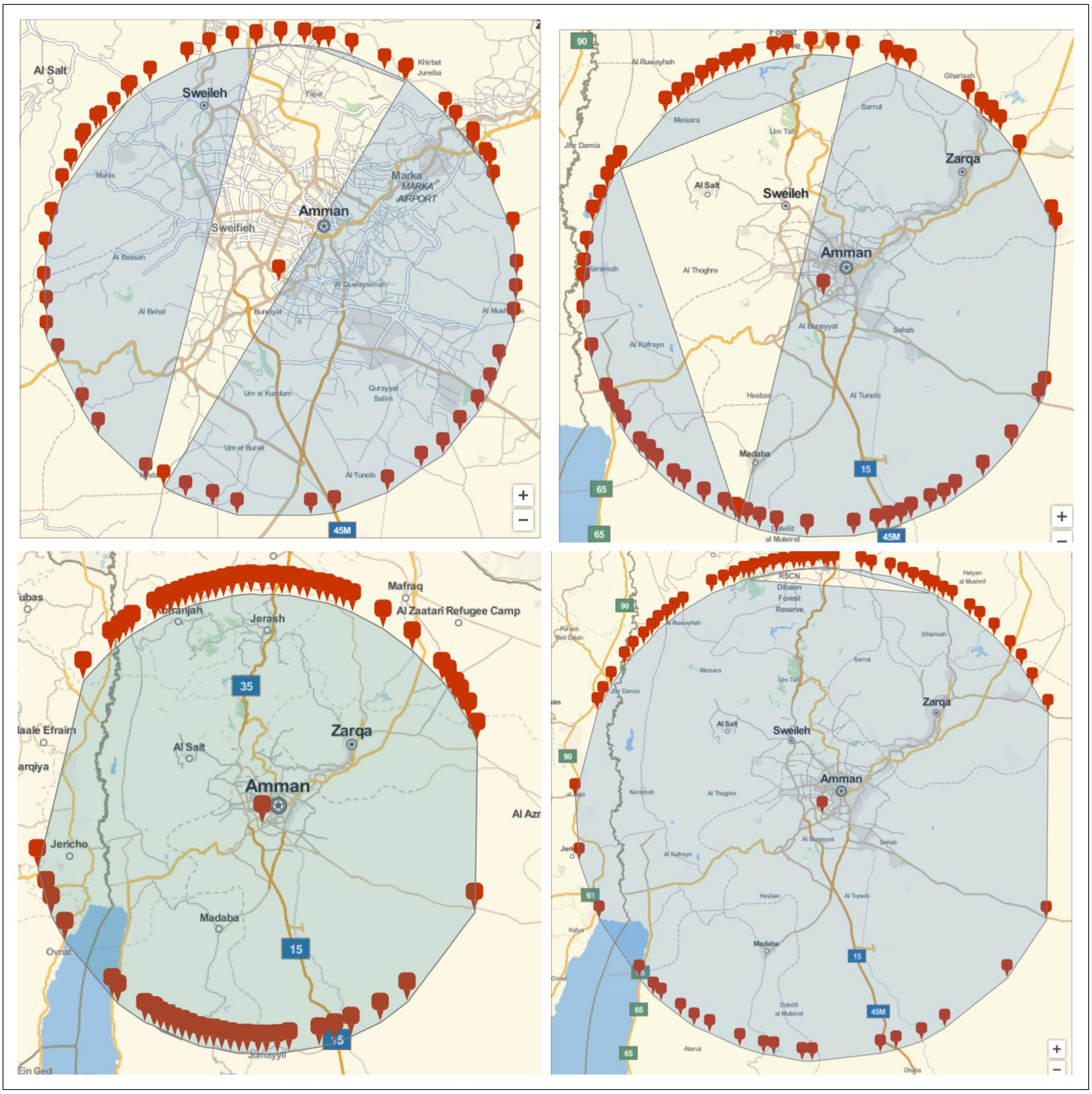

Figure 7: Coordinate: [31.917632100000002,35.891095]

4 Time Complexity

This section records the worst cast time complexity associated with every class file.

Class	**Complexity**
Marker.java	$\mathbf{O}$(n)
Polygon.java	$\mathbf{O}$(n)
Haversine.java	$\mathbf{O}$(1)
Network.java	$\mathbf{O}(n^2)$

Table 1: Time Complexity of the .java files.

5 Suggestions

It would be worthwhile to look into other methods that can be used as a substitute for the nearest-neighbour approach. This substitute should have a better time complexity. Furthermore, it should sort the coordinates in a clockwise order as accurately as possible.

6 References

[1]https://math.stackexchange.com/questions/1018164/sorting-a-list-of-points-in-2-d-clockwise
[2]https://www.math.ku.edu/ jmartin/courses/math105-F11/Lectures/chapter6-part5.pdf