An Android Application for Google Map Navigation System Implementing Travelling Salesman Problem

Anupriya, Mansi Saxena
School of Computing Science and Engineering, VIT University
Vellore-632014, Tamilnadu, India

***Abstract*—** Genetic Algorithm is used to determine the optimum route on Google map and solves the Travelling Salesman problem. This is implemented using Google API and Android OS. Different Geo Locations (Latitude and Longitude) are obtained in string format using Google GeoCoder API.

***Keywords*—** Genetic Algorithm, Travelling Salesman Problem, Google maps, Google API, GeoCoder API

I. INTRODUCTION

A. Google maps

Google maps provide an intuitive and highly responsive mapping interface with aerial imagery and detailed street data. In addition, map controls and overlays can be added to the map so that users can have full control over map navigation. Map panning can also be performed by dragging the map via the mouse or by using "arrow" keys on a keyboard. Google maps can be customized according to application specific needs. Various web-based application and the results can be displayed on Google maps. At this time, the Google maps are a beta version. Code changes may also be required since the product is in beta and there can be changes in the API. For instance, the parsing of Google data was earlier done using KML/XML but now it has been changed to JSON/XML.

B. Creating an android application that uses Google maps

- Install Google play services. Open the Android SDK Manager. Go to Extras and install Google Play services

- Obtain an API key. You have to add Map API key to your application to access the Google maps server with the maps API.
  Open a command prompt and navigate to the .android directory. The keytool command is used to get the SHA-1 fingerprint.

```
Keytool –list –alias androiddebugkey –keystore debug.keystore –storepass android –keypass android
```

Register in the Google APIs Console to use Google Maps for android. Enter the SHA-1 fingerprint to obtain a unique API key. In the androidmanifest.xml, add the following code:

```
<meta-data
android:name="com.google.android.maps.v2.API_KEY"
   android:value="your_api_key"/>
```

- Add a map to an android project.

C. Understanding the GeoCoder API

GeoCoder is a class for handling geocoding and reverse geocoding. Geocoding is a process of converting addresses into geographical coordinates (latitudes and longitudes). Reverse geocoding is the process of transforming (latitude, longitude) into addresses. Public Constuctors of GeoCoder API are as follows:

1) *Public Geocoder(Context context, Locale locale)* :

The response of the above constructor will be localized for the given Locale.
Where,
Context is the Context of the calling activity and
Locale is the desired Locale for the query results.

2) *Public Geocoder (Context context)* :

The response here is localized for the default system Locale.
Where,
Context is the Context of the calling activity.

3) *Public Methods*
Public List<Address> getFromLocation (double latitude, double longitude, int maxResults) :

It returns an array of addresses that describe the area that are surrounded by the given latitude and longitude.
The parameters are:
Latitude: the latitude a point for the search
Longitude: the longitude a point for the search
maxResults: max number of addresses to return.

4) *Public List<Address> getFromLocationName (String Location_Name, int max_Results, double Lower_LeftLatitude, double Lower_LeftLongitude, double Upper_RightLatitude, double Upper_RightLongitude)* :

It returns a list of address objects.

The parameters are:

Location_Name : description of a location

Max_Results: max number of addresses to return.

Lower_LeftLatitude: lower left corner latitude of the bounding box

Lower_LeftLongitude: lower left corner longitude of the bounding box

Upper_RightLatitude : upper right corner latitude of the bounding box

Upper_RightLongitude: upper right corner longitude of the bounding box.

5)*Public static Boolean isPresent( ):*

The above method returns true if the methods getFromLocation and getFromLocationName are implemented. Due to lack of network connectivity, the method may return null or empty lists.

D. Overlays

Overlays are objects that can be added on the Google maps. They help to reflect locations that are added on the map to indicate lines, points or areas. They are tied to coordinates so dragging or zooming does not affect their position.

Overlay is a base class representing an overlay which may be displayed on top of a map.

1) *Adding Overlays:*

To add an overlay, subclass this class, create an instance, and add it to the list obtained from MapView. getOverlays().

2) Removing Overlays:

The overlay's setMap() method is called and null is passed to remove an overlay from the map. The above method removes the overlay from the map but does not delete it. To delete the overlay, its value is set to null.

E. Markers

They are used to identify locations on the map. The following fields are commonly set when constructing a marker.

- Position- specifies a LatLng which identifies the initial location of the marker. This field is a must.
- Map- specifies the map object where the marker is to be placed. This field is optional.

Fig 1: Position of markers in Google map in an android application

F. TSP

A Travelling Salesman Problem (TSP) is a problem which determines the shortest possible route that visits each place exactly once and returns to the original place, given a list of places and the distances between each pair of places. The problem consists of a salesman and a set of cities. TSP finds its use in various applications such as DNA fragment assembly and VLSI design. The TSP belongs to the class of NP-hard problems, and consequently, it is unlikely that there is a polynomial time algorithm that solves each instance of the problem at optimality. [1] Many methods have been developed to solve the TSP such as Branch and Bound technique, Dynamic Programming technique, Heuristic method and Genetic Algorithm technique. [2] Analyzing the comparative study between three algorithms namely, Exhaustive algorithm, Heuristic algorithm and Genetic Algorithm (GA), following conclusion was obtained: [3] Exhaustive may be conventional but the result was impressive on yielding optimal total path and traversal. We have found that GA is a close rival to exhaustive, as their total path was similar, but at the end GA won. While, on the other hand, heuristic has outcome a close result as well as maintaining mild consume of time. Exhaustive and GA suffers from time deterioration, which was also happen to heuristic but less. To produce a consistent result, GA may need bigger iteration with respect to number of cities, thus a lot of time taken. However, GA excelled compared to heuristic as we increased the number of cities to the highest. We may assume that with accurate iteration and population size, GA approach may be able to cater larger problems. Thus, GA was proved to be best among the three.

G. Genetic Algorithm

Genetic Algorithms are relatively new paradigms in artificial intelligence which are based on the principles of natural selection. The formal theory was initially developed by John Holland and his students in the 1970's [4, 5]. The Genetic Algorithms mimic the evolution and improvement of life through reproduction, where each individual contributes with its own genetic information to build up new ones adapted to the environment with higher chances of survival. [6] An initial population called genome (or chromosome) is randomly generated then successive populations, or generations, are derived by applying genetic operators such as selection, crossover and mutation to obtain the best results. The selection operator chooses two members of the present generation in order to participate in the next operations: crossover and mutation. The crossover operator inter-mixes the alleles of the two parents to obtain an offspring. The mutation operator occurs a short period of time after crossover and as in nature it exchanges alleles randomly. The three most important aspects of using genetic algorithms are: (1) definition of the objective function, (2) definition and implementation of the genetic representation, and (3) definition and implementation of the genetic operators. [7]

II. PROPOSED SYSTEM

The system so implemented will solve the Travelling Salesman problem to determine the optimum route on Google map using Google API and Genetic Algorithm in Android OS. The system starts from getting different Geo Locations (Latitude and Longitude) on the map triggered by the user in String Format through Geo Coding. The Google Geo Coding API provides a direct way to access a Geo Coder via an HTTP request. Then, a distance matrix of these locations to be visited by the user is determined by parsing the JSON file based on pair of source and destination Geo Points.

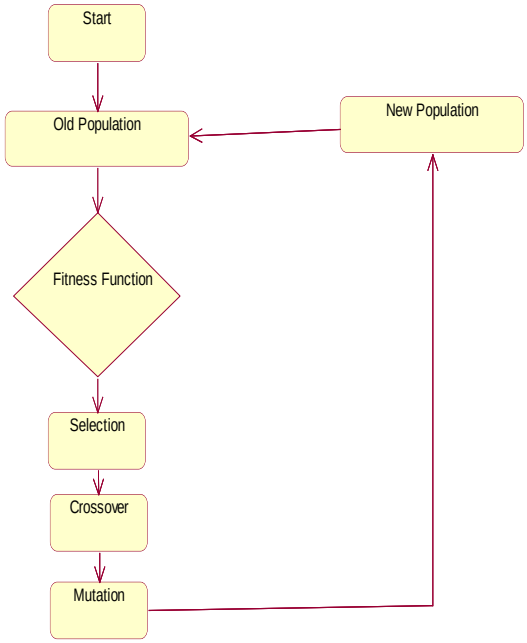

Fig 2: Genetic Algorithm Flowchart

III. RESULTS AND ANALYSIS

Fig 3: A TSP problem showing a list of places using markers

Fig 4: Optimal distance obtained by implementing Genetic Algorithm

Fig 5: Optimal path shown by lines

IV. CONCLUSION

The steps that should be followed for working with Google maps for an android application have been discussed. The various techniques for implementing Travelling Salesman Problem have been analyzed and Genetic Algorithm has been implemented successfully.

ACKNOWLEDGEMENT

This paper is made possible through the help and support from everyone, including parents, teachers, friends and families. This could not have been possible without their advice and financial support.

REFERENCES

[1] J. H. Holland, *"Adaptation in Natural and Artificial Systems. Ann Arbor"*, Michigan: University of Michigan Press, 1975.

[2] Mrs. Geetha Ramani.R, Nishaa Bouvanasilan, Vasumathy Seenuvasan, *"A perspective view on Travelling Salesman Problem using Genetic Algorithm"*, *World Congress on Nature & Biologically Inspired Computing , 2009*

[3] Nur Ariffin Mohd Zin , Siti Norul Huda Sheikh Abdullah, Noor Faridatul Ainun Zainal Esmayuzi Ismail, *"A Comparison of Exhaustive, Heuristic and Genetic Algorithm for Travelling Salesman Problem in PROLOG"*, International Journal on Advanced Science Engineering Information Technology vol2 2012

[4] J. H. Holland et. Al., "*Induction: Processes of Inference, Learning, and Discovery*", MIT Press, 1989.

[5] M. Melanie, "*An Introduction to Genetic Algorithms*", MIT Press, 1996.

[6] Vijendra Singh, Simran Choudhary, *"Genetic Algorithm for Traveling Salesman Problem: Using Modified Partially-Mapped Crossover Operator"*, IMPACT-2009

[7] Buthainah Fahran Al-Dulaimi, Hamza A.Ali, "Enhanced Traveling Salesman Problem Solving by Genetic Algorithm Technique(TSPGA)", World Academy of Science, Engineering and Technology 38, 2008