Proceedings of FIKUSZ '15 Symposium for Young Researchers, 2015, 89-102 pp © The Author(s). Conference Proceedings compilation © Obuda University Keleti Faculty of Business and Management 2015. Published by Óbuda University http://kgk.uni-obuda.hu/fikusz

An Android Application for Google Map Navigation System, Solving the Travelling Salesman Problem, Optimization throught Genetic Algorithm

Laurik Helshani

European University of Tirana, Albania
helshani@gmail.com

Abstract: TSP - A salesman plans a trip through which he wants to visit his clients and come back to the starting point. During the trip, same client should not be visited more than once and the route should be shorter and less costly. The mathematical modeling of this problem has to do with the theory of graphs and combinatorics. Graph vertexes symbolize places to visit, and edges that connect the nodes are the paths (the distance of which is known from the beginning) that lead to those places. The complexity of the exact algorithm for solving this problem increases with the number of places to visit. For this reason genetic algorithm is offered as the optimal solution of TSP problem. The whole system is implemented as client-server system using RESTFul web services, Google-services and Android OS. Genetic Algorithm is used to determine the optimum route on Google map and solves the Travelling Salesman problem.

Keywords: Travelling Salesman Problem (TSP), Genetic Algorithm (GA), Optimization, Android, Google maps, RESTFul web services in Java

1 Introduction

Travelling salesman problem has inspired studies by mathematicians, computer scientists, physicists and a host of nonprofessional researchers. The TSP has seen applications in the areas of logistics, genetics, manufacturing, telecommunications, and neuroscience, to name just a few.

The traveling salesman problem consists of a salesman and a set of cities. The salesman has to visit each one of the cities starting from a certain one (e.g. the hometown) and returning to the same city. The challenge of the problem is that the traveling salesman wants to minimize the total length of the trip. [1]

The most direct solution for a TSP problem would be to calculate the number of different tours through places. Given a starting place, it has n-1 choices for the second place, (n-2) choices for the third place, etc. Multiplying these together one gets (n-1)! for one place and n! for n the places. [8]

If we would count the cost of each route, we could end up waiting for the computer to give us a result for ages. In real life, we need to have such routes solved in reasonable time – we are even able to relax on quality of found route, just to have a route, which converges to most optimal one. [9]

The exact algorithms search for an optimal solution. One of them is branch-and-bound algorithm and its execution time is $O(n^3)$.

Heuristic solutions are approximation algorithms that reach an approximate solution (close to the optimal) in a time fraction of the exact algorithm. [8]

2 The mathematical formulation throught graphs theory

Let G = (V, A) be a graph where V is a set of n vertices. A is a set of arcs or edges, and let C : (C_{ij}) be a distance (or cost) matrix associated with A. The TSP consists of determining a minimum distance circuit passing through each vertex once and only once. Such a circuit is known as a tour or Hamiltonian circuit (or cycle). In several applications, C can also be interpreted as a cost or travel time matrix. It will be useful to distinguish between the cases where C (or the problem) is symmetrical, i.e. when $c_{ij} = c_{ji}$ for all $i,j \in V$, and the case where it is asymmetrical. Also, C is said to satisfy the triangle inequality if and only if $C_{ij} + C_{jk} \geq C_{ik}$ for all $i,j,k \in V$. This occurs in Euclidean problems, i.e. when V is a set of points in R^2 and C_{ij} is the straight-line distance between i and j. [2]

Definition: A complete graph KN is a graph with N vertices and an edge between every two vertices.

Definition: A weighted graph is a graph in which each edge is assigned a weight (representing the time, distance, or cost of traversing that edge).

Definition: A Hamilton circuit is a circuit that uses every vertex of a graph once.

Definition: The Traveling Salesman Problem (TSP) is the problem of finding a minimum-weight Hamilton circuit in KN.

The traveling salesman problem can be described as follows:
TSP = {(G, f, t): G = (V, E) a complete graph, f is a function $V \times V \rightarrow Z$, $t \in Z$,

G is a graph that contains a traveling salesman tour with cost that does not exceed t}. [1]

3 Google services

3.1 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. For instance, the parsing of Google data using JSON/XML. [3]

3.2 GeoCoding

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. [3]

3.3 Distance Matrix Service

Google's Distance Matrix service computes travel distance and journey duration between multiple origins and destinations using a given mode of travel.

To calculate the distance between two places for visiting, which is vital for the genetic algorithm to perform his work, using google service: Google Maps Distance Matrix. This service is not available to countries that are not members of the UN. In order for the application to be general, I used the Haversine formula to calculate this distance.

3.3.1 Haversine formula

R = earth's radius (mean radius = 6,371 km)

Δlat = lat2− lat1

Δlong = long2− long1

a = $\sin^2$(Δlat/2) + cos(lat1)*cos(lat2)*$\sin^2$(Δlong/2)

c = 2*atan2(√a, √(1−a))

d = R*c [4]

Note that angles need to be in radians to pass to trigonometric functions.

Java-implementation of this formula:

```
public static Double toRad(Double degree) {
    return degree * Math.PI / 180;
}

public double haversineDistance(double lat1, double lon1, double lat2, double
lon2) {
        double earth_radius = 6371.00;
        double latDistance = toRad(lat1-lat2);
        double lonDistance = toRad(lon1-lon2);
        double lat1_r = toRad(lat1);
        double lat2_r = toRad(lat2);
        double a = Math.pow(Math.sin(latDistance/2), 2) +
                Math.pow(Math.sin(latDistance/2), 2) *
       Math.cos(lat1_r) * Math.cos(lat2_r);
    double c = 2.0*Math.atan2(Math.sqrt(a), Math.sqrt(1.0-a));
    return earth_radius * c;
}
```

4 Genetic Algorithm (GA) in general

Genetic algorithm (GA) is a kind of evolutionary technique that emulates biological theories that are useful in solving optimization problems. According to Darwin's survival of the fittest evolutionary theory, only the most potential elements in a population are likely to survive generate offspring. The operation of GA begins with a population of random strings the design variable. Each string is evaluated to find the fitness function.

The three main GA operators - reproduction, crossover and mutation are applied on the random population to create new population. The population is evaluated and tested until the termination criterion is met, iteratively altered by the GA operators. Generation in GA represents the cycle of operation by the genetic operators and the evaluation of the fitness function. [5]

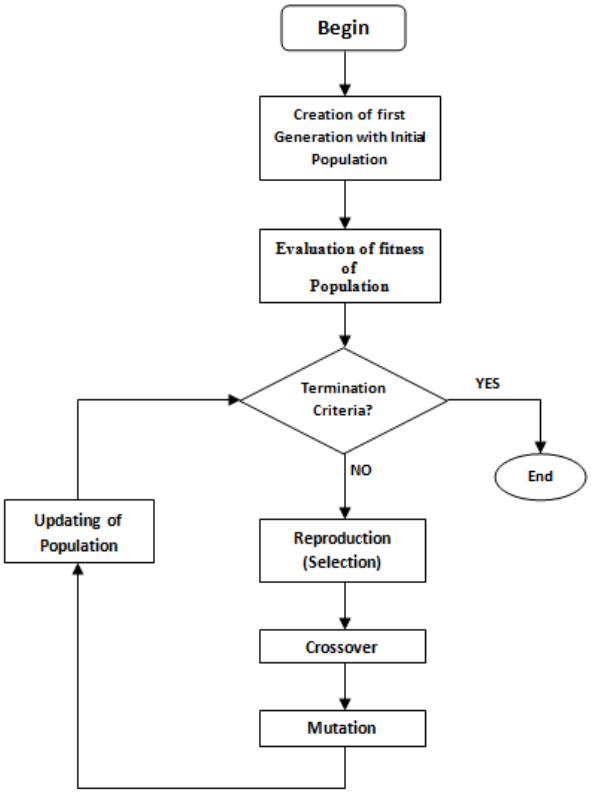

Figure 1
Flowchart of Genetic algorithm [5]

4.1 Applying of GA to the traveling salesman problem

First, create a group of many random tours in what is called a population. This algorithm uses a greedy initial population that gives preference to linking cities that are close to each other.

Second, pick 2 of the better (shorter) tours parents in the population and combine them to make 2 new child tours. Hopefully, these children tour will be better than either parent.

A small percentage of the time, the child tours are mutated. This is done to prevent all tours in the population from looking identical.

The new child tours are inserted into the population replacing two of the longer tours. The size of the population remains the same.

New children tours are repeatedly created until the desired goal is reached.

As the name implies, Genetic Algorithms mimic nature and evolution using the principles of Survival of the Fittest. [6]

4.1.1 Termination conditions of genetic algorithm

The following three kinds of termination conditions have been traditionally employed for genetic algorithm:

- An upper limit on the number of generations is reached

- An upper limit on the number of evaluations of the fitness function is reached, or

- The chance of achieving significant changes in the next generations excessively low [7]

4.2 Testing of GA and result interpretation

Test- tool programming is done in JAVA programming language and his work is displayed graphically using the 2D library.

The tool simulates graphically, what the algorithm is doing, generating random points in the XY-plane that symbolize places to visit, and draws straight lines between points that symbolize the roads that lead to these countries.

#places	population size	#generations	The initial length of the route	The length of the route after optimization	Optimization in %
30	1000	1000	6245.4	1724.4	362.178
50	1000	1000	10137.5	2277.5	445.115
70	1000	1000	15483.8	2822.9	548.506
100	1000	1000	20275.4	3609.6	561.707
120	1000	1000	22316.5	3996.6	558.387
150	1000	1000	28748.9	5243.7	548.256

#places	population size	#generations	The initial length of the route	The length of the route after optimization	Optimization in %
100	1000	5000	20359.4	3024.9	673.060
120	1000	6000	25794.4	3568.4	722.856
150	1000	10000	29885.9	3777.1	791.239

#places	population size	#generations	The initial length of the route	The length of the route after optimization	Optimization in %
150	20	100000	31156.0	3984.4	781.949
150	100	100000	31190.3	3786.6	823.702

As seen from the above tables, there are three key variables, wich effects directly to the performance of the algorithm work, to its termination and to the final result (optimization of the route of travelling salesman). These variables are: Number of places to visit (#places), population size and number of generations (#generations). The other three colums: The initial length of the route, The length of the route after optimization, Optimization expressed as a percentage are results of the tool.

With simulated data for testing, I have tested the impact of population size and the number of generations in the genetic algorithm convergence.

The population size determines the number of potential solutions of the optimization problem.

Test cases and results of genetic algorithm are presented in above tables.

With increasing population size also increases the computational effort (complexity).

Optimization reaches the culmination (~823.7%), if the number of generations is minimum ten times larger than the population number.

5 The implemented system

The whole system consists from android application as a client and a system as server.

5.1 Client side

The system is implemented to 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 two ways:

a) The user clicks on the google map and marks the place he wants to visit

Java implemention of "MapLongClick":

```
@Override
public void onMapLongClick(LatLng point) {
      try {
           GoogleMap myMap;
           String key = "MyApiKey";
           String url_geocoding =
"https://maps.googleapis.com/maps/api/geocode/json";
           StringBuilder builder = new StringBuilder();
           builder.append(url_geocoding).append("?latlng=")
                      .append(point.latitude)
                      .append(",").append(point.longitude);
                builder.append("&sensor=true");
                builder.append("&key=").append(key);
           JSONObject objekti = read(builder.toString());
           JSONArray arr = objekti.getJSONArray("results");
```

```
            String address = arr.getJSONObject(0).getString("formatted_address");
            JSONObject components = arr.getJSONObject(1);
            JSONArray
            address_components=components.getJSONArray("address_components"
            );
            String place_name =
address_components.getJSONObject(1).getString("long_name");
             myMap.addMarker(new
MarkerOptions().position(point).title(place_name).snippet("Address: "+address));
        } catch(Exception e) {
           Log.e("Error", "Response string buffer error. " + e.getMessage());
        }
}
```

b) The user knows the name or address of the place he wants to visit and search through autosuggestion. Then the list will be proposed to user, then user selects the desired place, and after that, the place will be marked on the map. This is possible by using a google web service called Google places.

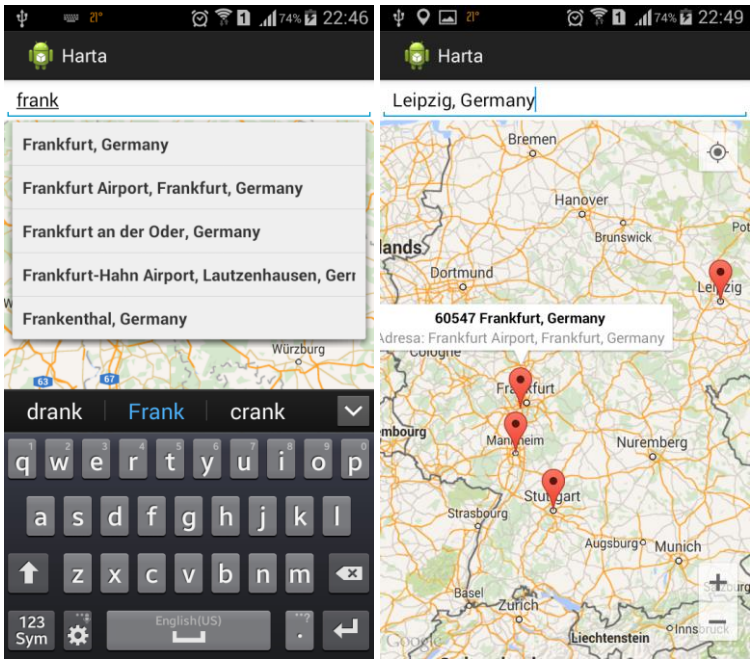

Figure 2
Autosuggestion and marked places on implemented Android application

Once the user has chosen places to visit and has select function "Optimize and map the road", the system derives all the necessary information from Google map and packet them into JSON. JSON data are sent by application to the server via HTTP-Post request.

Figure 3
Optimized route and drawn on google map throught polylines

5.2 Server side

Travelling Salesman Problem has been analyzed and Genetic Algorithm has been implemented, in Java, as RESTful web service, successfully. REST involves the transfer of *resources* between clients and servers. This algorithm consists of the following classes:

a) Place→ models a google place. It contains all the characteristics of a google place: id, name, address, latitude and longitude
b) Tour → stores a candidate tour and is a list of places (ListArray<Place>)
c) TourManager → holds the places of a tour and is a list of places (ListArray<Place>)
d) Population → Manages a population of candidate tours
e) Genetic Algorithm→ manages algorithms for evolving population

This server side system consumes and produces JSON formatted data and is running on tomcat server.

Java implemention of this RESTful web service:

```
public JSONArray arrayListToJSON(ArrayList object) throws JSONException {
        JSONArray json = new JSONArray();
        Place place= new Place();
        for(int i=0; i<object.size(); i++) {
                json.put(place.konvertoNeJSON((Vendi) object.get(i)));
        }
        return json;
 }


@POST
@Consumes("application/json")
@Produces("application/json")
public String postJson(String content) {
    JSONObject result = new JSONObject();
     try {
       JSONObject json = new JSONObject(content);
       JSONArray jsonArray = json.getJSONArray("places");
       for(int i=0; i < jsonArray.length(); i++) {
                TourManager.addPlace(new
         Place(jsonArray.getJSONObject(i).getString("id"),
                jsonArray.getJSONObject(i).getString("name"),
          jsonArray.getJSONObject(i).getString("address"),
          jsonArray.getJSONObject(i).getDouble("latidude"),
          jsonArray.getJSONObject(i).getDouble("longitude")));
       }
       Population pop = new Population (30, true);
       pop = GeneticAlgorithm.evolvePopulation(pop);
       for (int i = 0; i < 100; i++)  {
         pop = GeneticAlgorithm.evolvePopulation(pop);
       }
       ArrayList< Place> place = pop.getFittest().placeslist();
       result.put("places",  arrayListToJSON(vendet));
     }  catch(Exception $ex)  {

     }
     return result.toString();
}
```

6 Conclusions

a) If the number of places to visit is less than 10, it recommended to use the exact algorithm, otherwise gjenetic algorithm.

b) Genetic Algorithm is one of the best methods which is used to solve various NP-hard problem such as TSP.

c) Genetic Algorithm has a great influence on the optimization of round trip of TSP, based on results in above tables.

d) The purpose of this paper is to show how to use Google 's services in combination with genetic algorithm to solve TSP problem . This purpose makes working to discern from the papers of the earlier or existing

e) I presented an analysis of various aspects associated with the specification of termination conditions for genetic algorithm

f) Genetic algorithm reaches an approximate solution (close to the optimal),

g) Web - Google services cannot be used more than 2,500 times a day, as an ordinary user. This limitation does not apply to business customers, since they pay for services.

Acknowledgement

This paper is made possible through the help and support from my family, my PHD-Advisor and my workmates.

References

[1] Ioannis G. Tollis: The Traveling Salesman Problem, Computer Science Department, University of Crete, 2014

[2] Gilbert Laporte: The Traveling Salesman Problem: An overview of exact and approximate algorithms, in Proceedings of European Journal of Operational Research 59, North-Holland,1992, pp. 231-247

[3] Anupriya, Mansi Saxena: An Android Application for Google Map Navigation System Implementing Travelling Salesman Problem, in Proceedings of International Journal of Computer & Organization Trends – Volume3 Issue4, 2013

[4] Alexander, R. (2006), Geo (proximity) Search with MySQL

[5] S. Vishnupriyan, L. Govindarajan, G. Prabhakaran, K.P. Ramachandran: Quality Improvement in Higher Education through Normalization of Student Feedback data Using Evolutionary Algorithm, in Proceedings of International Journal of Applied Management and Technology, Vol 6, Num 3

[6] R. Imbach: Genetischer Algorithmus, Fachhochschule, Nordwestschweiz, p. 4

[7] M. Safe, J. Carballido, I. Ponzoni, N. Brignole: On Stopping criteria for genetic algorithms, Universidad Nation del Sur, Argentina, 2004

[8] H. Demez: Combinatorial Optimization: Solution Methods of Traveling Salesman Problem, Submitted to the Institute of Graduate Studies and Research, Eastern Mediterranean University, North Cyprus, 2013

[9] R. Stanec: Solving of Travelling Salesman roblem for large number of cities in environment with constraints, Diploma thesis, Mendel University, Brno, 2011