Journal of Academia and Industrial Research (JAIR)
Volume 3, Issue 1 June 2014

ISSN: 2278-5213

RESEARCH ARTICLE

Location Based E-commerce Pricing Tool

D.C. Mehetre, Vivek Jadhav*, Yogeshwari Kapare, Rupali Jadhav and Sonali Zaware
Dept. of Computer Engineering, K.J. College of Engineering and Management Research, K.J. Educational Institute, 408/10, Purandar Complex, Mukundnagar, Pune-411037, India
vivek.jadhav@kjsedu.com*; +91 9730779783

Abstract

Location based E-commerce pricing tool is an android application aiming to provide solutions for users who want to perform E-commerce for real estate online in easy and fast fashion on their android devices. Application gives a complete end-to-end solution to its users by displaying approximate user requested data i.e. the location and price range. Application uses Google maps API for displaying maps which provides all the real estate locations available for E-commerce in the location entered by the user. Zone based color pricing is the main utility added in the application for the users to directly distinguish between the prices of the neighborhoods of a location. After selecting the locations, the user can view entire details of the properties available and perform E-commerce or users can view the contact details and contact the owner of the property. The updated data about approximate prices will be displayed from the server dynamically with the help of the smart tools which are available open source on internet. In this, a service will help to fetch the data intelligently from the backend. The data retrieval is done using simple math and shown to the user in clear and readable fashion so that the user interface will be easy as well as the data retrieval will be faster.

Keywords: E-commerce, google maps, color pricing, server, open source, data retrieval.

Introduction

According to new emerging trends in market and the growth in technologies based on PDAs (Personal Digital Assistants) such as mobile phones and tablets, Android operating system is an open source where users can invent something new and extend the functionality of existing systems. Drastically changing its versions one after another, android comes with support for large amount of hardware and new inventions. Location based services is one of them which is supported by android to allow its users to take help of GPS (Global Positioning System) available as a piece of hardware in a mobile device. Location based services means using the GPS functionality, one can connect to the internet and view the current location of the device and details of location from where the device is operated. All this information is displayed using map. Such simple functionality can be extended using Google Maps API and users can create their own overlays on the existing maps and display the same on the device. Location based E-commerce pricing tool uses similar functionality and displays maps having custom overlays. When user accesses the application and enters a keyword he wants to search, user gets options such as enter the price range MIN and MAX and enter the location. The application is connected online with the database which contains boundary coordinates in the form of latitude and longitudes. All the neighborhood boundaries are displayed on the map which is fetched from the database. Complex geocoding is required to display the boundary locations with the color. A fixed color format is displayed according to the price entered by the users. The colors are modified according to the price range entered by the user. Such application uses various complex technologies which are difficult to handle such as geocoding of the neighborhood boundaries entered by the user, support the developed application framework in contrast with the Google maps API framework and Android framework. This application supports both the frameworks in the bounds of rules of the Google maps API and Android application framework. The main objective of study is to focus on front end tools that can help E-commerce and facilitate the users regarding online property selling and buying to get approximated rates and locations of the properties faster.

Materials and methods

Google maps API: Google provides online developer services for users to use google tools and develop desktop, web and android applications (Dodsworth and Nicholson, 2012). Google provides major developer API's for android and iOS and one of them is google maps android API. There are various versions of google maps android API distinguished as Version 1 (V1), Version 2 (V2) and Version 3 (V3). These versions come for different purposes. Google map API V1 is used for application using android operating below android 2.2-2.2.3 Froyo (API level 8). Version 1 of the google maps android API has been officially deprecated as of December 3rd, 2012.

However, application's already working on this version can continue to work but no updates will be provided. Google maps android API V2 is used for operating systems from version Android 2.3-2.3.2. Gingerbread (API level 9) is the new support for android market also known as Google Playstore. We use google maps android API V2 in our application as we are using android play store services, android internet services and google maps for android. Google maps android API V3 is for developing web application and web services and is focused majorly towards online applications rather than mobile applications but, it still supports mobile applications with registering the application to API V3 online.

Google maps API V2 key: We require a key to use google API in our application; therefore we need to register our application to "console.developers.google.com" under API and auth section. We select google maps android API V2 option and switch it ON. As soon as we use this option we need to pass credentials such as Public API Access and then we can generate a key. The Key generation process requires the java package name which we have given to our project and the SHA 1 Fingerprint which can be obtained by following process.

In windows command prompt put following:

```
C:\Windows\system32>"C:\Program Files (x86)\
ist -alias androiddebugkey -keystore "C:\Use
orepass android -keypass android
androiddebugkey, Apr 9, 2012, PrivateKeyEntr
```

The displayed result will be a SHA1 and MD5 Fingerprint. SHA-1 fingerprint is a unique text string generated by SHA-1 hash algorithm and as it is unique, google maps uses it to identify the application. Thus, entering the package name and obtained SHA fingerprint the "console.developers.google.com" will generate a unique key. This key is free for development purpose and must be renewed after every 90 d, but we can purchase this key for commercial purpose if we want to put the application to sell on Android market.

Geocoding: It is the process of finding the geographic coordinates in the form of latitude and longitude. Geocoding means converting any type of postal street address, zip code address to a geographic coordinate for digital media such as google maps can understand the data and the area for which the application is coded. Geocoding allows geotagging by allowing to enter, address or pin point a location with photo or any user requested data over a map. User can use geocoder software freely available on internet to do the process. Geocoded locations are useful in many GIS analysis. Geocoding, along with GPS provides location data for geo-tagging media, such as photographs or RSS items. Geocoding allows turning address into latitude and longitude. Geocoding does not convert the address to the exact longitude and latitudes but it gives closest approximation which is mostly determined to be exact. Android developer API provides a geocoder API class for the process, a class for handling geocoding and reverse geocoding:

public final geocoder extends object
{ logic statements };

Boundary locations neighborhood coloring: Coloring the boundary coordinates of a region is the most major task on maps. If we are giving the boundary colors dynamically it takes huge amount of time and tremendously complex calculations. There are no maps available till now which provide intelligent data retrieval services and color the maps according to the user input though, heat maps can be used to color the regions on maps dynamically but heat maps are too complex and difficult to understand and implement. To color boundary, we can use third party software's or API such as twitter API which facilitates color on maps but that is only available for UK and US countries. Neighborhood means all the local places and areas available in a country which are randomly named by the country people on the basis of locality. These places must be exactly colored according to their boundaries. One common problem which occurs while coloring the boundaries is overlapping the neighborhood colors with each other because distinguishing the exact boundaries of particular location is very difficult task. Neighborhood coloring requires a best form of closest boundaries of the locations which are to be colored.

Results and discussion

Processing of application: Mapping is a very important requirement for providing services to the B2B and B2C clients. The B2B clients require integration of our framework in their application as well as B2C clients want a fully functional application without third party application integration. The color based pricing requires many different parameters such as longitude, latitude, geocoding, google maps API integration, displaying data from backend to front end dynamically, pushing of dynamic coordinates over the maps, accuracy of boundary location of various places etc. All these parameters need to work simultaneously. Google maps integration models are said to be computationally intensive because there is a lot of stuff to calculate. The prediction of next coordinate is based on previous coordinate's location and time scale with precision of a minute. Hence, the calculation is too much. For example, a huge earth (almost a sphere) has to be divided into discrete units and thus, forming a grid of longitude and latitudes. The cells formed are of non-uniform size. These calculations require spherical co-ordinates (θ, ϕ) which are uniform. It requires spectral transforms. Thus, the computation of such application will require a huge time for non-parallel processing system.

The whatsoever algorithm we use will require a lot of computations and must have a very high accuracy and capability to calculate the data from large database.

Data retrieval algorithm: Information retrieval system is to retrieve the documents from various sets of data including multiple subjects which are stored in the database in an unstructured format. Thus, the information retrieval systems mostly used on web based systems require a usage of heterogeneous systems to be connected to each other with each system containing database of different subjects having different information. Such systems are called as distributed database systems and are most widely used on web based applications. There are various information retrieval algorithms available such as Adarank algorithm (Cao and Huang, 2006). Application requires a separate layer to communicate with such databases which is known as multi-database layer which can be seen as a logical connection of a multiple databases connected to the application. Multi-database layer makes the application to see the logical connection as single database connected to it. Multi-database layer has many features such as integrating various models such as network model, relational model into a single model, various schemas having different parameters and values to be seen as a single schema having unique values (Freund *et al.,* 2003). In our application we are using a part of information retrieval. We cannot say it as full information retrieval because we are not retrieving documents in the application; we are retrieving the data according to the range and colors from the database and pushing it on the frontend of the map directly.

Algorithm used is as follows:
Notations used: Set of numbers S={M_a^{n-1} – M_n^{n-1}}, arr is array of strings, C^r is set of coordinates to be mapped

Algorithm for data retrieval:
Input: S {(M_a^{n-1} – M_n^{n-1}) , arr}
Output: C^r with color

1. Take set of values (Ma – Mn) as input
   if(M_a^{n-1} – M_n^{n-1}→ Null)
   exit()
   else
   input (C^r)
   Jump to step 2.
2. Input string as arr[string];
3. Map arrto Coordinates C^r
   if no mapping exit
4. Calculate color according to inserted range (M_a^{n-1} –M_n^{n-1}) and color preference (map coloring algorithm shows the logic).
5. Display sorted C^r and range of values.

Map coloring algorithm: There are various map coloring algorithms available which can be readily implemented in the applications having complex problems for coloring maps.

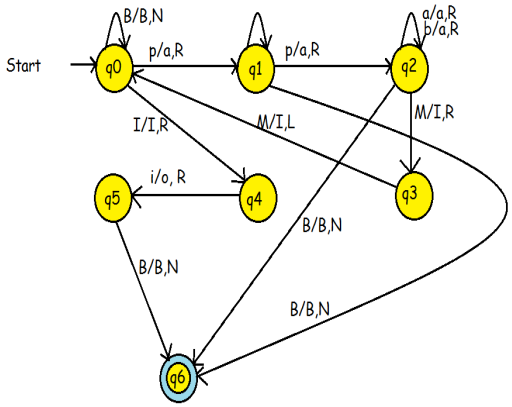
The algorithms for map coloring go from four color, five color graph coloring algorithms to graph canonization, Bellman-Ford, Nearest neighbor etc. (Gebremedhin and Manne, 2000). We use a different approach as an enhancement to the above mentioned category of algorithms. We are inspired by the four coloring algorithm and hence, continue to add efficiency to the algorithm by means of using partitions. Maps can be seen as huge graphs and different location in the maps can be seen as different nodes which are to be colored (Hussein and Khair, 2006). Therefore, map coloring is also known as map coloring. In conventional maps, it is required that each country must have a different color form its neighbor to differentiate each other therefore, the use of only four color can be done. As, planer graph are used for coloring countries on a map it gives only the coloring of entire country. States and cities and neighborhood of cites and areas are not colored in it. Therefore, we can conclude that coloring the countries is an example of planer graph and four colors are sufficient. In our case, we are coloring non-planar graphs; we do not know how many colors are required (Leighton, 2006). A planar graph can be embedded in a plane surface, i.e., it can be drawn on the plane surface in such a way that its edges intersect only at their endpoints.

Algorithm:

1. Partition the graph (given area) into k blocks according to location L preference.
2. Calculate C^r after plotting by mapping C^r to L in k blocks.
3. for i = 1 to k do in parallel
   for each v to vi do
   assign a color to v, paying attention to already colored vertices
   (both on and off processor).
4. Detect and resolve the conflicts and display colored non planer graphs.

Mathematical model: This model gives the entire working of the application in terms of math which will be carried out during the application. This model is solvable by putting values in place of alphabets to see whether the application will work or not. The application exactly follows the working shown in this mathematical model. Model consists of state transition diagram and state transition table with respect to the diagram (Fig. 1 and Table 1). Users start initially by opening the application. User passes username and password as input to the application that is transition from state q0 to q1 given as p/a, R. If username and password are not entered or are empty then user stays at state q0 given as B/B, n where B stands for Blank input. At state q1 if the username and password are not accepted then it will stay on state q1 or if accepted then transition from q1 to q2 i.e. p/a, R. The user will be displayed direct map of the application after success of state q2 and it will transit from q2 to q3 to display the map.

Fig. 1. State transition diagram.



Conclusion

Techniques and algorithms used in developing the efficient framework and application to help the field of E-commerce using location based services and thus, the tool named “Location Based E-commerce Pricing Tool” can be developed as an enhancement for performing E-commerce on properties using google maps. This tool can also be integrated into various websites and application as a whole for coloring using both the algorithms used for developing the technique.

Acknowledgements

Authors acknowledge the principal, Dr. Sanjeev J. Wagh, Head of Department, Mr. Soumitra S. Das and Mr. Manoj Sharma for the support and timely guidance.

Table 1. State transition table.

States	Input	Output	Transition
Customer (q0)	Username	Register user login success	q0 – q1 – q2
Login (q1)	Password	If invalid move left and register/again login. If valid move right	q1 – q1 q1 – q2
Map (q2)	Select area, Enter area, Measurement	Map displays information about area location, rates, owner, etc.	q2 – q2, q2 – q3
Service (q3)	Information of Owner, Area	Send to customer	q3 – q0
SMS/E-Mail (q4)	Contact No. and Email id of owner from information	SMS/E-Mail is sent to owner	q0 – q4 – q5
Owner (q5)	SMS/E-Mail	Received by owner	q5 – q6

State q2 stays as it is if the same map is to be displayed using same area and same measurement of the area according to the colors and algorithm. All the business logic will be applied here at state q2. The user sees all the information on his application at state q3 and then if he wishes to exit the transition is done form state q3 to q1. At q1 state user can also exit if he wishes to discontinue the work and continue it later. State q5 shows user communication with application and the owner of the property and it shows the decision weather user buys or rejects the property. State q6 is the final or exit state where user quits the application.

References

1. Cao, Y. and Huang, Y. 2006. Adapting ranking SVM to document retrieval. In SIGIR 29, pp.186-193.
2. Dodsworth, E. and Nicholson, A. 2012. Academic uses of google earth and google maps in a library setting. *Inform. Technol. Libraries*. 6(2): 102-117.
3. Freund, Y., Iyer, R.D., Schapire, R.E. and Singer, Y. 2003. An efficient boosting algorithm for combining preferences. *J. Mach. Learn. Res.* 4: 933-969.
4. Gebremedhin, A.H. and Manne, A. 2000. Scalable parallel graph coloring algorithms. *Concurrency: Pract. Exp.* 12: 1131-1146.
5. Hussein, A. and Khair, E.S. 2006. New graph coloring algorithms. *Amer. J. Math. Stat.* 2(4): 739-741
6. Leighton, F.T. 2006. A graph coloring algorithm for large scheduling problems. *J. Res. Notional Bureau Std.* 84(6): 489-503.