ÇUKUROVA UNIVERSITY
INSTITUTE OF NATURAL AND APPLIED SCIENCES

MSc THESIS

Fatih KILIÇ

A PUBLIC TRANSPORTATION ROUTE PLANNING TOOL: ToTRoP

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

ADANA, 2010

ÇUKUROVA UNIVERSITY
INSTITUTE OF NATURAL AND APPLIED SCIENCES

A PUBLIC TRANSPORTATION ROUTE PLANNING TOOL: ToTRoP

Fatih KILIÇ

MSc THESIS

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

We certify that the thesis titled above was reviewed and approved for the award of degree of the Master of Science by the board of jury on ../10/2010.

…….....	……..	…….....
Assoc. Prof. Dr. Mustafa GÖK	Assoc. Prof. Dr. Ulus ÇEVİK	Asst. Prof. Dr. Mutlu AVCI
SUPERVISOR	MEMBER	MEMBER

This MSc Thesis is written at the Department of Institute of Natural And Applied Sciences of Çukurova University.
Registration Number:

Prof. Dr. İlhami YEĞİNGİL
Director
Institute of Natural and Applied Sciences

Not: The usage of the presented specific declerations, tables, figures, and photographs either in this thesis or in any other reference without citiation is subject to "The law of Arts and Intellectual Products" number of 5846 of Turkish Republic

ABSTRACT

MSc THESIS

A PUBLIC TRANSPORTATION ROUTE PLANNING TOOL: ToTRoP

Fatih KILIÇ

ÇUKUROVA UNIVERSITY
INSTITUTE OF NATURAL AND APPLIED SCIENCES
DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

Supervisor :Assoc. Prof. Dr. Mustafa GÖK
Year: 2010, Pages: 59
Jury :Assoc. Prof. Dr. Mustafa GÖK
:Assoc. Prof. Dr. Ulus ÇEVİK
:Asst. Prof. Dr. Mutlu AVCI

In this thesis, a web based public transportation route planning tool (ToTRoP) is developed. The routes are generated to transport people to a specific location (a hospital, a stadium, a university campus etc.) while keeping minimum cost and maximum passenger number. ToTRoP uses Google APIs to convert text based address data into geographical location data and then generates the most beneficial routes to a target point. ToTRoP is used to derive possible transportation routes from the city of Adana to Çukurova University's campus based on the anonymous address data of the freshmen students. According to the test results, transportation of approximately 76 % of the students is enabled through seven routes. The tool can work integratedly with other GIS software thanks to WFS support. This feature enables it to be a SOA based GIS application.

Key Words: GIS, Public Transportation, Routing, Google Map API, WFS

ÖZ

MASTER TEZİ

TOPLU TAŞIMA ROTA PLANLAMA YAZILIMI: ToTRoP

Fatih KILIÇ

ÇUKUROVA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

Danışman :Doç. Dr. Mustafa GÖK
Yıl: 2010, Sayfa: 59
Jüri :Doç. Dr. Mustafa GÖK
:Doç. Dr. Ulus ÇEVİK
:Yrd. Doç. Dr. Mutlu AVCI

Bu tezde, halk otobüsü taşımacılığı için coğrafi bilgi sistemi tabanlı rotalar üreten bir web aracı geliştirilmiştir. Farklı adreslerde yaşayan kişileri hedef bir noktaya (hastane, stadyum, üniversite kampüsü v.b.) taşımak amacıyla üretilen rotaların hedefe minimum maliyetle en fazla yolcunun erişmeleri amaçlanmıştır. Sunulan araç verileri coğrafik verilere dönüştürmek için Google Map API'leri kullanır ve hedef noktaya ulaşmak için en faydalı rotaları üretir. Bu araç Çukurova Üniversitesi kampüsüne ulaşım için yeni öğrencilerin adres bilgilerini baz alarak uygulanabilir taşıma rotaları üretti. Bu test verilerine göre yedi tane rota ile öğrencilerin yaklaşık %76 sının taşınması sağlandı. Ek olarak OGC tarafından tanımlanan WFS servisi ile diğer CBS uygulamaları ile entegre olabilir. Bu saye de yapılan uygulama SOA tabanlı olmuştur.

Anahtar Kelimeler: GIS, Toplu Taşımacılık, Rotalama, Google Map API, WFS

ACKNOWLEDGEMENTS

I am heartily thankful to my supervisor, Dr. Mustafa GÖK, whose guidance and support from the initial to the final level enabled me to develop an understanding of the subject and instructed me through my thesis.

I offer my regards to Mr. Ramazan YAZGAN, The head of IT Department at Çukurova University, who supported me in any respect during the completion of my thesis.

I owe my deepest thanks to Çukurova University Engineering-Architecture Faculty, Electrical and Electronic Engineering Department that enabled me to benefit form all facilities of the department during my thesis study.

CONTENTS **PAGE**

ABSTRACT I
ÖZ II
ACKNOWLEDGEMENTS III
CONTENTS IV
LIST OF TABLES VI
LIST OF FIGURES VIII
LIST OF ABBREVIATIONS X
l. INTRODUCTION 1
2. PREVIOUS WORK 3
3. METHODS 9
3.1. J2EE Technologies and Web Server 10
3.2. Web Technologies 12
3.3. Data Base Management System and Support Components 15
3.4. Utility Tools 17
3.5. Test Data 18
4. THE ToTRoP TOOL 21
4.1. The Software Architecture of ToTRoP 21
4.2. Used Algorithms in ToTRoP 25
4.3. Tool Functions 27
5. USER'S MANUAL of ToTRoP 37
6. RESULTS AND DISCUSSION 45
7. CONCLUSION 53
REFERENCES 55
CURRICULUM VITAE 59

LIST OF TABLES **PAGE**

Table 3.1. The result set obtained by executing the sql code. 17
Table 3.2. Raw address data 19
Table 3.3. Address data in DB 19
Table 6.1. Routing that minimizes the distance according to first test set 50
Table 6.2. Routing that minimizes distance / student according to first test set 51
Table 6.3. Routing that minimizes the distance according to second test set 51
Table 6.4. Routing that minimizes distance / student according to second test set 51

LIST OF FIGURES **PAGE**

Figure 2.1. A candidate routes set generation algorithm (CRGA) 5
Figure 3.1. The deployment diagram of ToTRoP 9
Figure 3.2. An HTTP servlet handling GET and POST requests 11
Figure 3.3. Google map sample code 1. 13
Figure 3.4. Google map sample code 2 14
Figure 3.5. The graphical output of the sample code 2 15
Figure 3.6. How pgRouting Works (Patrushev, 2007) 16
Figure 3.7. Sample code using FWTools 18
Figure 3.8. Importing from SHP to DB 18
Figure 4.1. The result of getCapabilities method in the ToTRoP 23
Figure 4.2. A part of the schema description of address information in the ToTRoP 24
Figure 4.3. A request and a record of address information in the ToTRoP 24
Figure 4.4. DBSCAN algorithm (LIU et al., 2007) 25
Figure 4.5. The K-Means Algorithm (Deelers and Auwatanamongkol, 2007) 26
Figure 4.6.A sample graph 26
Figure 4.7. The UML diagram of the ToTRoP 27
Figure 4.8. Student addresses on the main road map of Adana 29
Figure 4.9. Student addresses after the elimination of low density points 30
Figure 4.10. The computation of the angle between two paths 30
Figure 4.11. The distribution of the addresses after K-Means clustering is performed 31
Figure 4.12. A sample network 32
Figure 4.13. The pseudo code for the modified Dijkstra's algorithm adapted from Wkipedia (Wkipedia - Dijkstra, 2010) 34
Figure 4.14. The class diagram of ToTRoP's WFS 35
Figure 5.1. The DB Configuration page 37
Figure 5.2. The Address Information page 38
Figure 5.3. The Data Preparation page 39

Figure 5.4. The Start Point Selection page 40
Figure 5.5. The Route Generation screen 42
Figure 5.6. A route on the Route Generation Screen..... 43
Figure 5.7. uDig desktop software 43
Figure 6.1. The routes generated by ToTRoP for minimum route distance according to first test set..... 46
Figure 6.2. The routes generated by ToTRoP for minimum distance / student rate according to first test set 47
Figure 6.3. The routes generated by ToTRoP for minimum route distance according to second test set 48
Figure 6.4. The routes generated by ToTRoP for minimum distance / student rate according to second test set..... 49

LIST OF ABBREVIATIONS

GIS	:	Geographic Information System
GML	:	Geography Markup Language
XML	:	Extensible Markup Language
HTML	:	HyperText Markup Language
OGC	:	Open Geospatial Consortium
J2EE	:	Java 2 Enterprise Edition
SQL	:	Structured Query Language
SHP	:	Shapefile
MVC	:	Model View Controller
DBMS	:	Database Managment System

1. INTRODUCTION

Geographic Information Systems (GIS) use spatial data to define, analyze and solve complicated problems in numerous areas including health, social, environment, industry, transportation and economics (Maquire Et Al., 1991; Thill, 2000) GIS encompasses anything related to the spatial data such as collection of the spatial data, presentation of the data, analysis of the data, and developing hardware and software tools that use geographic data. Today, GIS based applications are not used only by government decision makers but used by public for a variety of activities spanning from trip planning to real estate hunting.

One of the important problems of the transportation is route planning. Routine planning is widely used in the distribution networks and public transportation and can be viewed as a complicated optimization problem where many special constrictions exist. For example, a school bus routing method can be very different than a mail distribution routing problem since their constrictions and the optimization criteria are different. For instance, the mail needs to be delivered to every location while it is impossible to transport everyone to their specific addresses. In general, there is no generic solution applicable for all routing problems.

Public transportation routing is also unique and may be divided into several sub-problems. One of the most important purposes of the public transportation route planning is increasing the number of people who benefits from the routes; of course while achieving this purpose the route length should be as short as possible. Though transportation many points to a single point is not a full scope problem such as transportation from many points to many points, it is a crucial problem which is frequently faced in practice. Commuting to city center, accessing major health facilities from suburban, transportation of students to campus from off-campus residences are typical examples of this problem.

Since there are many users with distinct computing platforms and preferences, urban services data must be accessible using interoperable solutions. Numerous solutions for providing interoperability are proposed by the Geospatial

Information (GI) community, as well as the Information Technology (IT) community. The most widely used technologies for providing interoperability in the GI community are OGC services framework (Amirian et al., 2010). Open Geospatial Consortium-OGC was founded 1994 to define interoperability standards. The Open Geospatial Consortium, Inc (OGC) is an international industry consortium of 398 companies, government agencies and universities participating in a consensus process to develop publicly available interface standards (OGC, 2010).

Vast amounts of geographical data require large storage facilities, high-performance hardware and efficient software to analyze them. On the other hand, there is a significant gap between theoretical solutions and practical applications. GIS applications continuously generate and store spatial data while software tool development cannot keep up a similar pace. Many routing problems are still waiting their efficient software solutions.

This thesis is a humble effort for providing such a tool for public transportation route planning. The presented tool is named as ToTRoP [Toplu Tasıma Rota Planlama Yazılımı (Turkish)] and has two main objectives: The first objective is providing an interface to visualize the public transportation needs. The second one is generating effective routes for public transportation which is impossible to do without the help of computers. ToTRoP is tested for the transportation route planning of Çukurova University students from off campus addresses to the campus. ToTRoP was developed as Service Oriented Architecture (SOA) based (Fang et al., 2009). ToTRoP presents collecting and displaying of information as a service with the support of WFS service.

2. PREVIOUS WORK

Public transportation route planning is a difficult problem since finding an efficient route requires analyzing a complex network and considering many constrictions. General solutions are based on a graph. Dijkstra and Floyd algorithms are well-known shortest-path computing algorithms on a graph (Takaoka, 1998). There are different problems in real life to generate routes for public transportation and shortest path is not always the best solution. Solutions are generated to find the best routes, selecting origin and destination in working area considering population density (the least overlap routes generation, capacity of vehicle and frequency of bus are considered) **(**Desaulniers and Hickman, 2007**;** Gebeyehu and Takano, 2008).

Current et al. (1985) described the maximum covering shortest path problem (MCSP) to cover maximum demand with minimum cost (Current et al., 1985). They presented a scenario. Here a demand value is added to each node. A demand value may be a population density. A demand is covered when the route includes the node or passes through another node that within some specified distance range of that node. Commonly, the solution increases the total path length but covering demand area is increased too (Current et al., 1985; Bapna et al., 2002; Matisziw et al., 2006).

The maximal tour covering problem (MTCP) is referred as a new version of MCTS by Current and Schilling (1994). MTCP allows for complete tours. Although they are similar, some differences exist. One of the differences is that the MCSP allows for the development of a single path while the MTCP allows for the development of a single tour (Current et al., 1994; Matisziw et al., 2006). These methods are heuristic approaches. Nowadays multi routes generation must be considered while developing a heuristic solution to solve the problem as there are multi routes to access to a location in real life. Matisziw et al. (2006) presented a constraint structure that handles extending an existing route system by adding a specified number of new stops such that potential demand coverage is maximized and added route length is minimized. To do this, they expand to formulation of MTCS and MCTS. The method is called as the maximum covering route extension

problem (MCREP). While developing their method, they consider multiple start points (Matisziw et al., 2006).

Another problem is called as the transit network design problem (TNDP) to determine a set of routes for a public transportation problem. It is more complex than others. Heuristics and metaheuristics are used to solve the problem. Heuristics is a classical approximative method. This approach is based either on constructive or local improvement procedures. Other variants of this approach select the best routes in a pool which has previously generated set of routes. Metaheuristics is a modern approximative method. Mostly, the applications of this method use genetic algorithm (Mauttone and Urguhart, 2009; Desaulniers and Hickman, 2007**)**.

Baaj and Mahmassani (1995) decompose the network design problem into three parts. One part is the route generation. They proposed a heuristics algorithm for route generation. RGA (Route Generation Algorithm) continues generating routes until demand covering constrains are fulfilled. RGA use predetermined routes as set of initial routes. In this stage, routes are generated by using shortest path according to high demand. The nodes which are suitable to a prespecified criterion are inserted in these routes. And this process continues until generated routes satisfy demand constrains (Baaj and Mahmassani, 1995; Mauttone and Urguhart, 2009).

The route generate algorithm (RGA) of Baaj and Mahmassani (1995) is used By A. Mauttone and M.E.Urquhart (2009) in their study. And they proposed Pair Insertion Algorithm (PIA). PIA is that its original expansion of routes by inserting individual vertices is replaced by a strategy of insertion of pairs of vertices (Mauttone and Urguhart, 2009).

Another solution is GA (Genetic algorithm) used for optimum solution for route network design problem in a study by Pattnaik et al.(1998). GA is a search algorithm based on natural selection. Their solution has two stages. In first stage, set of candidate routes are generated to achieve the optimum solution. In the second stage the optimum routes are selected by GA (Pattnaik et al., 1998).

In Pattnaik et al.'s study, the first stage is a candidate routes set generation algorithm (CRGA). The CRGA is on heuristic algorithm and consists of the following steps:

The CRGA

Step 1: Generate routes for every terminal in (O,D) matrix.
Step 2: Generate a route by using shortest path algorithm in Step 1.
Step 3: The route is checked for minimum and maximum route length. If the route is enough for the constraints then mark the route as a candidate route.
Step 4: Generate alternate routes by clamping every link of the shortest route generated in Step 2 successively and by finding shortest path between the same origin and destination and then, release the link.
Step 5: Be sure that following constrains are satisfied:

- the existence of the routes
- duplication of routes
- significant overlap with the shortest route
- maximum route length
- maximum detour if the route satisfies

Step 6: If these are satisfied alternate route is accepted as a candidate route.
Step 7: Classify all routes according to their performance and store them as the candidate routes set.

Figure 2.1. A candidate routes set generation algorithm (CRGA) (Pattnaik et al., 1998).

In the second stage, optimum routes are found in the candidate routes. To do this, GA is used. Two methods are developed for using a genetic algorithm. These are FSLC (Fixed String Length Coding) and VSLC (Variable String Length Coding). These methods are tested obtained by a sample network which includes 25 nodes and 39 links. Optimum solution is reached with 16 routes obtained by using FSLC. The routes cover all demands which are 16213 numbers. On the other hand, a solution with 13 routes is generated by using VSLC. Only 78 demands are unsatisfied from all demands. Generated solution by FSLC is better than VSLC's solution. Otherwise VSLC generates a smaller number of routes than FSLC (Pattnaik et al., 1998). 25 nodes used in the above network are too small for an efficient network. In this thesis 1364 real nodes are used for the simulation of Adana city. As a consequence, the feasibility of the generated results is shown on the real map. In addition generated solutions obtained by these methods include too many routes for 25 node and 39 links. That a solution has less number of routes is considered to generate routes in this thesis.

Another method for route generation is proposed by Mintesnot and Shin-ei (Gebeyehu and Takano, 2008). This solution uses Dijkstra algorithm to cover maximize population with minimum travel time according to route impedance factors. They proposed a The Traffic Analysis Zone (TAZ)-based approach. They divide the study area into a TAZ. Population information for each area is collected. The bus and road network and bus stops are included in the system. The basic idea of the method is that each edge is assigned to score according to passenger density, capacity, coverage and frequency. They developed the formula to compute a value of an edge obtained by $C_i = (S_{max}/S_i)*l_i$. S_{max} is the maximum score of S_i for all edges. S_i is the optimization score of edge i. L_i is the length of edge i. C_i is calculated for each edge. The route is generated according to C_i by using Dijkstra algorithm (Gebeyehu and Takano, 2008).

Calculating cost and demand of edges in the graph is very important to generate routes but this problem must be seen as NP-Hard problem. The problem is not solved using shortest path algorithm and calculating cost and demand. Optimum solution can be reached with heuristics or metaheuristics methods. A heuristic is developed to solve this problem in this thesis.

Another important issue is that interoperability. This property has become a must with the increase of new technologies in GIS applications. A WFS service was written within ToTRoP after studying published documents on interoperability by OGC and GeoServer Web Map Server (Davis, 2007).

ToTRoP, the product of this thesis, basically generates solutions to public transportation problem considering the least overlapping routes, maximum coverage student with minimum distance.

While ToTRoP is developed, for the solutions of these problems, a workflow enabling to generate multi routes in order for maximum student to benefit from the routes with the least overlap is developed. To do this, the problem is divided into two stages. First phase uses modified Dijkstra algorithm to generate a route from an origin to a destination to cover maximum student with minimum distance. The basic idea is that the route is generated according to minimum current total distance of the

route / current total demand of the route for current node. Another phase is developed for the least overlapping routes.

In conclusion, ToTRoP has become a useful software tool in Route Planning of Public Transportation with Web services prepared according to standards determined by OGC, and in collecting and presenting of data as Web based.

3. METHODS

Figure 3.1 shows the deployment diagram of ToTRoP. In this section, software and hardware tools and used theory which are needed for ToTRoP to run. Free and independent OS software is preferred in the development of the tool. Thus, cost of the tool is reduced.

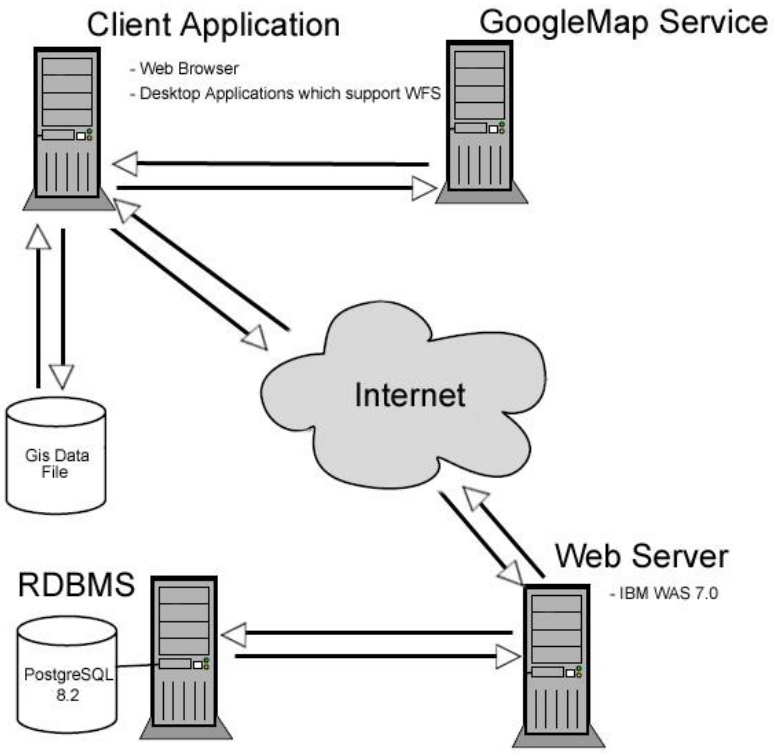


Figure 3.1. The deployment diagram of ToTRoP.

Programming Language and Software Developer Tools: ToTRoP is developed using Java 2 Enterprise Edition (J2EE). So that it can run on almost every OS and web servers which support J2EE. ToTRoP is a dynamic web application thanks to J2EE technologies.

3.1. J2EE Technologies and Web Server

J2EE architecture is a multitier application. Web application server is set up based on this architecture too. J2EE architecture has a client tier, a middle tier, and a back-end tier. A Client tier includes internet explorer, Java FX application and Java applet and client application using services on web application. The middle tier is the section in web tier and has EJB, web container (JSP, Servlets, JSF..), web component. The back-end tier consists support of systems which are enterprise information systems for example rational database management system (RDBMS), enterprise resource planning (ERP) application (Green, 2002).

JavaServer Faces (JFS) technology is employed for standard web interfaces. JSF technology is preferred as it supports Model-View-Controller (MVC) structure for ToTRoP. Interactive pages can be easily prepared with its components supporting new web technologies.

JSF has a lot of user interface (UI) component. Some of these are command button, combo box, text box, panel group. These components are used while developing ToTRoP. Using java beans is vey simple as JSF supports drag and drop. At the same time UI component is supported with drag and drop. In addition events of UI can be handled (Sun Microsystems-JSF, 2010).

Servlet run on web server and is a java technology. It is developed against cgi programming. It is used via a request-response with http speciation (Sun Microsystems-J2EE, 2002). It can be used in developing web page or data transfer. A Servlet must implement the javax.servlet.Servlet interface. A HTTP Servlet has HTTP specific functions which are service, get and post. (Hunter, 2001). Figure 3.2 shows how an HTTP servlet handles GET and POST requests. This sample code sends GET message to GET request as a response; in POST process, it sends the message “POST is performed”.

Servlet technology is used within ToTRoP for information transfer. WFS process is also performed by using servlet. In addition, Servlet technology is used in address information page and route generation page. Thus, JavaScript is used in JSF pages, and information is asked from related Servlets, then results are imaged on pages.

Servlet Sample Code

```
package tr.edu.cu.gis.service;
import javax.servlet.ServletContext;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
public class MapAddressService  extends HttpServlet{

	private static final long serialVersionUID = 1L;

	protected void doGet(HttpServletRequest request,
			HttpServletResponse response)
			throws ServletException, IOException {
	response.setContentType("text/html; charset=ISO-8859-9");
	PrintWriter out = response.getWriter();
		out.append("Get\n");
	}

	protected void doPost(HttpServletRequest request,
	HttpServletResponse response)
	throws ServletException, IOException {

		out.append("POST is performed \n");
	}
}
```

Figure 3.2. An HTTP servlet handling GET and POST requests.

Web Server: Tool uses J2EE architecture because of that the web server must support J2EE. Popular J2EE web servers are IBM WebSphere Application Server, JBoss, Sun Java System Web Server, GlassFish Server, Apache Geronimo, et al.. Some of them are free, and others are not. In this application, WebSphere is preferred as it is compatible with Rational Software Architecture (RSA) whose product developer is IBM.

3.2. Web Technologies

Ajax: Some parts of the ToTRoP processes substantial data. In these parts, page is update in every process is unnecessary. For this reason, Asynchronous JavaScript and XML (AJAX) technology is used in order to refresh only certain sections in a page. In address information page and start point selection page, this technology is used in the refreshment of the lists and displaying of map information.

Most of JSF's UI component supports AJAX. One of them is panelGroup. Objects are refreshed in this UI component with AJAX. Refreshing process can be done with AJAX UI component. These components are ajaxRefreshRequest, ajaxRefreshSubmit and ajaxExternalRequest. These are JSF component of IBM. Firstly, hx:ajaxRefreshRequest refresh content using HTTP get request. Secondly, ajaxRefreshSubmit refresh content using HTTP submit request. Finally, ajaxExternalRequest refresh content using HTTP get request but the content is taken from a different page (IBM – AJAX Component, 2010).

Selecting start the point of routes process provides AJAX support by using ajaxRefreshSubmit to show address location on map and to make record operation on manual start point of routes and target point.

Presentation Tools and Geographic Support Tools on Web Page: In ToTRoP, Google Map APIs are used in digitizing and presenting of information on web page. GoogleMap is an internet based application including many geographic functions and services. Among these functions, GClientGeocoder is used in digitizing information, thus coordination of an address can be found, and in the same way, address of a point is found when its location is given. In addition, roads, markers, graphical elements can be drawn on map (GoogleMap, 2010).

Finding A Coordinate: For this process, GClientGeocoder is used. We can both get address information of a point and location information of a given address information. All information and services are provided by Google servers. By using this service, we can change the whole information in hand, and develop this by using javascript and server based codes.

Figure 3.3 shows a simple usage of GClientGeocoder. Sample code converts entered address information to location information and adds it to table on page. Showlocation() method is triggered by clicking to a button. After finding address information, showaddress function runs.

Google Map Sample Code 1

```
function showLocation() {
  var address = document.getElementById("address").value;
  geocoder.getLocations(address, addAddressToMap);
}

function showAddress(address) {
     geocoder.getLatLng(    address,    function(point) {
          if (!point) {
                    addRow("not found"," not found "," not found ");
       } else {

          document.getElementById("form1:txtLat").value= point.lat();
          document.getElementById("form1:txtLng").value= point.lng();

          map.setCenter(point, 13);
          var marker = new GMarker(point);
          map.addOverlay(marker);
          marker.openInfoWindowHtml(address);
          addRow(address,point.lat() ,point.lng());

       }
     } );
 }
```

Figure 3.3. Google map sample code 1.

Drawing Graph Elements: With Google Map API, graphical elements according to geographic coordination system can be easily drawn as web based. The explanation of main graphic elements of Google is given as fallowing:

Map: The main element is Map Object in Google Maps API. Most operations are made on map object.

GLatLng: Used for referencing a geographical location.

Marker : Used to indicate a point on a map or to give information about that point.

PolyLine: Multi line is drawn on map by GPolyline.

Polygons: GPolygon objects are similar to GPolyline objects because they both include multi line. But Polygons defined regions within a closed loop not in open ended ones (GoogleMAP., 2010).

In Figure 3.4 Google Map Sample Code 2 is shown, and in Figure 3.5, output of Google Map Sample Code 2 is seen.

Google Map Sample Code 2

```
var map;
function initialize() {
        if (GBrowserIsCompatible()) {
                map = new GMap2(document.getElementById("map"));
                map.setCenter(new GLatLng(37.0537674,35.3607260), 12);
                map.addControl(new GSmallMapControl());
                createGraphic();
        }
}

function createGraphic(){
        var p1 = new GLatLng(37.05376743,35.36072604)
        var marker = new GMarker(p1);
        map.addOverlay(marker);
        var p2 = new GLatLng(37.00376743,35.30072604)

        var polyline = new GPolyline([ p1,p2], "#ff0000", 10);
        map.addOverlay(polyline);

        var lat = p1.lat();
        var lon = p1.lng();
        var latOffset = 0.01;
        var lonOffset = 0.01;

        var polygon = new GPolygon([
        new GLatLng(lat, lon - lonOffset),
        new GLatLng(lat + latOffset, lon),
        new GLatLng(lat, lon + lonOffset),
        new GLatLng(lat - latOffset, lon),
        new GLatLng(lat, lon - lonOffset)
        ], "#f33f00", 5, 1, "#ff0000", 0.2);

        map.addOverlay(polygon);
}
```

Figure 3.4. Google map sample code 2.

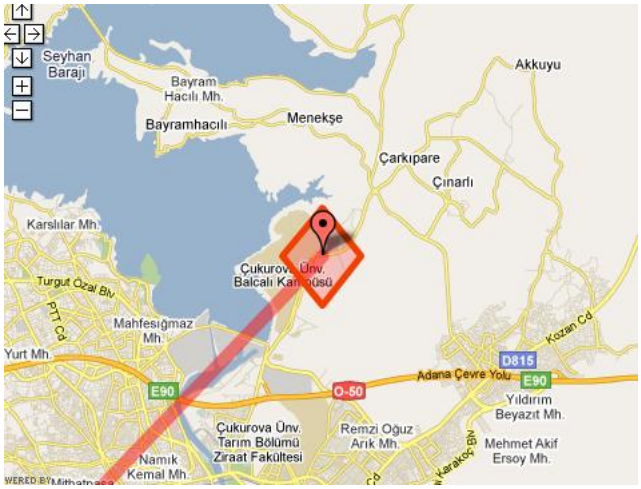


Figure 3.5. The graphical output of the sample code 2.

3.3. Data Base Management System and Support Components

Data Base Management System (DBMS): DBMS is the system responsible for storage, query and management of data. It is important to choose suitable database for the project's performance. For this project, DBMS must support geographic data, run some type OS platform (unix, FreeBSD, windows, Mac) and must be compatible with Java.

When these conditions are considered, PostgreSQL is selected as it supports above mentioned requisites. Also, PostgreSQL is user-friendly and there are a myriad of sources about it, and these sources are accessible on the internet (Davis, 2007; PostgreSQL, 2010).

Geographic Support Plug-in for DBMS: PostgreSQL cannot support geographic functions alone, so prepared definitions and functions must be added on DBMS. This problem is solved by PostGIS. PostGIS is the tool including query, storage and management of geographic data as well as many useful geographic functions (Davis, 2007).

PostGis Functions Used in the Tool: In ToTRoP, many of PostGis functions are used. Several of them are AsText, GeomFromText, Distance, X, Y, MakePoint, LineMerge. Brief information on the usage of these functions will be given:

AsText: Used to convert geometric field to text information.

GeomFromText: Used to convert text information to geometric field.

Distance: Computes the distance between two geometric point.

X,Y : Used to get x and y coordinate of a geometric point.

MakePoint: Forms a geometric data in point type by entering x and y points.

LineMerge: Forms a new geometric data in MultiLineString type by merging lines more than one (Davis, 2007).

Routing Solution Package: pgRouting is developed to provide routing functionality to PostGIS and PostgreSQL by Orkney and the people of Georepublic as Open Source Software (OSS) (pgRouting, 2010).

pgRouting is used for route finding problems in ToTRoP. Also, SQL based route finding function is developed to cover maximum demand with minimum cost. In Figure 3.6 working schema of pgRouting is seen. As it is coded with C, it finds solutions very fast, and works on PostgreSQL directly.

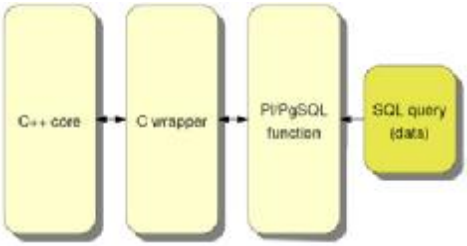


Figure 3.6. How pgRouting Works (Patrushev, 2007).

Using PgRouting: In this section, usage of pgRouting tool on a sample roadmap table will be explained (Philipone and Kastl, 2007). Let us assume that we have adanaroad table containing gid(integer) and the_geom (geometry) fields. The_geom field includes MultiLineString records. According to this, usage of the shortest path function is as follows:

Add source, target and length column to adanaroad: Target and source columns are added in order to show the connection between two nodes and the length field is cost field between two nodes.

ALTER TABLE adanaroad ADD COLUMN source integer;

ALTER TABLE adanaroad ADD COLUMN target integer;

ALTER TABLE adanaroad ADD COLUMN length double precision;

Create network topology: The code shows usage of assign_vertex_id. The first parameter determines the road map table on which processes are performed, the second one determines the tolerance gap of nodes to be generated, the third one determines the column to be analyzed, and the last one is the id column of the relevant column.

SELECT assign_vertex_id('adanaroad ', 0.001, 'the_geom', 'gid');

UPDATE adanaroad SET length = length(the_geom);

Generating Path by Dijksta Algorithm: The first parameter of the code determines the road map table. Second parameter determines gid of source node and final parameter determines gid of destination (gid is identification no of row in set of vertices).

SELECT gid, AsText(the_geom) AS the_geomtext FROM dijkstra_sp('adanaroad', 1, 789);

Results : Table 3.1 shows the result set obtained by executing the sql code.

Gid	The_geomtext
335	MULTILINESTRING((35.2987497446315 37.0479694178047,3….))
336	MULTILINESTRING((35.2987816939285 37.0479892920752,35….))
1226	MULTILINESTRING((35.2960285563932 37.0464270647….))
.	.

Table 3.1. The result set obtained by executing the sql code.

Generating Path by A Star Algorithm : SELECT gid, AsText(the_geom) AS the_geomtext FROM astar_sp ('adanaroad, 1, 789)

3.4. Utility Tools

Exporting From Database to Graph File Tools: FWTools is used in order for graphic based information in Database to be exported to graphic formats (FWTools, 2010; Davis, 2007). Figure 5.3, figure 5.4. and figure 5.6 are generated

by using FWTools in this thesis. Figure 3.7 shows the usage of sample FWTools in order for data to be transferred from geodatasum table to SHP format.

Sample Code Using FWTools

```
ogr2ogr -f "ESRI Shapefile" out\Road_1.shp PG:"host=localhost user=postgres dbname=wayroad password=17081981 port=5431" -sql "select id as gid, studentnumber as name,the_geom from geodatasum where roadgid=1"
```

Figure 3.7. Sample code for using FWTools.

Importing From Shapefile (SHP) to Database (DB): In ToTRoP, any process for the import of SHP file is not performed, for this aim, "shp2pgsql" function coming in PostGIS is used. This function automatically transfers the file in designated SHP type to DB (Davis, 2007). Figure 3.8 shows the code transferring data from SHP file in SQL format to txt file.

Sample Code Importing from shp to db

```
shp2pgsql -d adanaYollar\adana.shp way wayroad > deneme.sql
```

Figure 3.8. Importing from SHP to DB.

WFS Desktop Client Tool: To test WFS services of ToTRoP, uDig tool is used. uDig (user-friendly Desktop Internet GIS) is a desktop application to access, edit, and view GIS data. In addition, it supports WMS, WFS, WCS defined by OGC. On the other hand, uDig applications can be developed with SDK (Software Development Kit) since it is framework and is built on Eclipse (uDig, 2010).

3.5. Test Data

For ToTRoP test, real data is collected. Registration information of students who registered at Çukurova University in 2009 is taken from Registrar's Office Student Information System. This information is digitized for ToTRoP, and some results are given via ToTRoP.

This information is gathered collectively in text format. They are standardized and street, hometown and town names are separated. Table 3.2 shows the result of standardized process. This information is collectively transferred to database with the module prepared in ToTRoP. In consequence of transferred information, in the structure seen in Table 3.3, point based data list is transferred to database. In addition to this, 5000 pieces of data are randomly generated within Adana provincial borders, and graphical and digital data for two different data sets are composed, then, situation assessment is made in result and discussion section.

Address	Hometown	Street	Town
Kışla mah.4420 .Sk. xxx Yüreğir	Kışla	4420	Yüreğir
Yurt Mah. 71225 Sok. xxx Çukurova	Yurt	71225	Çukurova
Sümer Mh. 69041 Sokak xxx Seyhan	Sümer	69041	Seyhan
…	…	…	…

Table 3.2. Raw address data.

Look For Address	Geo Data
Kışla mah. 4420 sk. Yüreğir Adana	POINT(35.334233 37.019917)
Yurt mah. 71225 sk. Çukurova Adana	POINT(35.299669 37.02692)
Sümer mah. 69041 sk. Seyhan Adana	POINT(35.315014 37.020401)
…	…

Table 3.3. Address data in DB.

4. THE ToTRoP TOOL

This section gives a general description of the common methods and algorithms used in ToTRoP. The system supports four main functions on data: collection, management, processing and presentation. The brief explanation of these functions is given in the following. In most cases the collected data is not in the form of geographic data where textual address information has to be converted into geographic coordinates. After the format conversion, the data can be used to generate the actual geographic route. ToTRoP has a module which uses GoogleMap objects to convert textual address information to geographic coordinates. Address information is converted to geographic information, and a module to transfer this data to the database is developed. To maintain relatively large geographic data a strong database management system is required. Fortunately, PostgreSQL and its open source extension postGIS supports all the special functions needed to manage geographic data. ToTRoP works in conjunction with this database management system. ToTRoP generates routes to a specific point from multiple points. To achieve this task, the initial step that should be taken is the determination of these multiple points. This can be made by conducting a spatial density analysis. In this thesis, DBSCAN (Density-Based Spatial Clustering of Application with Noise) and K-Means algorithm is used to identify starting points. The second step is the routing process. This has been achieved by using the pgRouting library and modified Dijkstra algorithm. PgRouting includes Dijkstra, A Star and shorting star functions. A modified version of Dijkstra's algorithm is used in the tool based on SQL. With dynamic web pages in ToTRoP, a rich content is enabled. In addition, due to web service, it can work integratedly with other GIS software.

4.1. The Software Architecture of ToTRoP

ToTRoP is designed according to MVC architecture. In addition to this, except this architecture, ToTRoP supports SOA architecture with WebService. Basic principle of SOA is to enable interoperability of different software

applications. GIS applications supported by SOA are useful on the internet because there a lot of GIS applications and these have different advantages. Thus, if an application is based on SOA, it can make use of them.

SOA is the architecture explaining the presentation of works within different systems as service, and the integration between these services (FANG et al., 2009). Here, each service solves a problem. In SOA, as long as input/output parameters of services are stable, updating in these services is very easy, as updating only in one service is enough (Microsoft, 2010).

That the application supports web service is not enough. At the same time it defines international standards. So OGC was founded in 1994 to define interoperability standards. Members of OGC are industrial companies, governments and academies. Some standards of it are GML, WMS, WFS and WCS on the internet (OGC, 2010).

GML is based on XML and is defined to express geographical features. In a GML file, both verbal and digital records can be kept with vectorial graphical data (OGC, 2010). Mostly it is used to transfer for geographic data from a system to another system.

A WMS is used on internet to generate image for requesting geo-registered using graphical file or RDBMS (OGC, 2010). WFS is a web service on which updating, deleting and adding processes can be performed, and it generates results in GML format (OGC-WFS, 2010). Within ToTRoP, WFS is preferred, as information collection process is prepared as a service. Thus, address data can be entered with WFS services. In addition, generated routes will be presented in map format as WFS service. A WFS has eight methods to make operations which are querying, inserting, updating, deleting on registers. These are getCapabilities, DescribeFeatureType, DescribeFilterModel, getFeature, FeatureWithLock, Property, LockFeature, Transaction. getCapabilities, DescribeFeatureType, getFeature of them are obligatory. Briefly defining, some of WFS functions are as follows:

GetCapabilities: This method is used to get WFS capabilities. It generates XML result. Figure 4.1 shows a request and a part of XML result.

Request

http://localhost:9081/ToTRoPWeb/RoadWfs?request=getCapabilities&version=1.0.0

Response

```
…
<FeatureType>
        <Name>totrop:roads</Name>
        <Title>ToTRoP Roads</Title>
        <Abstract>ToTRoP Roads..</Abstract>
        <Keywords>totrop_roads, ToTRoP, roads</Keywords>
        <SRS>EPSG:4326</SRS>
        <LatLongBoundingBox minx="35.133762" miny="36.84468"
        maxx="35.45612" maxy="37.105682" />
</FeatureType>
<FeatureType>
        <Name>totrop:address</Name>
        <Title>ToTRoP Address</Title>
        <Abstract>ToTRoP Address...</Abstract>
        <Keywords>poi,ToTRoP, Address of Students</Keywords>
        <SRS>EPSG:4326</SRS>
        <LatLongBoundingBox minx="34.904694" miny="36.7978"
        maxx="35.5722" maxy="37.061527" />
 </FeatureType>….
```

Figure 4.1. The result of getCapabilities method in the ToTRoP.

DescribeFeatureType: It generates a schema description of requested feature type which is structure of served feature by WFS (OGC-WFS, 2010). Figure 4.2 shows a request and a part of the schema description of address information in ToTRoP.

Request

http://localhost:9081/ToTRoPWeb/RoadWfs?REQUEST=DescribeFeatureType&service=wfs&version=1.0.0&TYPENAME=totrop:address

Response

```
…
  <xsd:complexType name="addressType">
  <xsd:complexContent>
   <xsd:extension base="gml:AbstractFeatureType">
    <xsd:sequence>
      <xsd:element maxOccurs="1" minOccurs="0" name="the_geom"
      nillable="true" type="gml:PointPropertyType"/>
      <xsd:element maxOccurs="1" minOccurs="0" name="name"
      nillable="true" type="xsd:string"/>
```

```
        <xsd:element maxOccurs="1" minOccurs="0" name="boundedBy"
        nillable="true" type="xsd:string"/>
        <xsd:element maxOccurs="1" minOccurs="0" name="address"
        nillable="true" type="xsd:string"/>
        <xsd:element maxOccurs="1" minOccurs="0" name="gid"
        nillable="true" type="xsd:long"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
 </xsd:complexType>
…
```

Figure 4.2. A part of the schema description of address information in the ToTRoP.

GetFeature: it generates GML content which includes geometric and feature data for a request. A request and the result of it are shown in Figure 4.3. In addition, this usage of it is simple. Moreover, it has filter parameter. Spatial query is run with this parameter.

Request

```
http://localhost:9081/ToTRoPWeb/RoadWfs?REQUEST=getFeature&serv
ice=wfs&version=1.0.0&TYPENAME=totrop:address
```

Response

```
…
<gml:featureMember>
<totrop:address fid="poi.2254">
        <totrop:the_geom>
        <gml:Point>
        <gml:coordinates>35.30345,37.028453</gml:coordinates>
        </gml:Point>
        </totrop:the_geom>
        <totrop:name>55237648426</totrop:name>
        <totrop:boundedBy>null</totrop:boundedBy>
        <totrop:address>Yurt Mahallesi Mavi Bulvar 1 cuk ve Dinlenme
        Park Seyhan, 01160 Yurt, Tkiye</totrop:address>
        <totrop:gid>2254</totrop:gid>
</totrop:address>
</gml:featureMember>
…..
```

Figure 4.3. A request and a record of address information in the ToTRoP.

Transaction: Record operations which are inserting, updating, deleting are done with transaction method (OGC-WFS, 2010). XML result is generated after the operation. It includes state information of the operation which is success or failed.

4.2. Used Algorithms in ToTRoP

DB-Scan: Density-based clustering algorithms are commonly used in GIS analyses. Within ToTRoP, in the first analysis of address data, DBSCAN clustering method is used, thus points having less density are eliminated.

DBSCAN, in general, considers the density of point. It has two parameters to cluster. These are MinPts and radius (ε) (DUAN et al., 2007). Implementation of the algorithm is developed based on SQL in PostgreSQL for ToTRoP. DBSCAN algorithm is as follows:

DBSCAN Algorithm

1. Repeat steps numbered two and nine for all registers unvisited in data set.
2. Mark P (P is selected point in Dataset) as visited.
3. N=Select neighbors of P in radius.
4. If the number of neighbors is less than determined MinPts, mark it as noise, skip to the next iteration.
5. Mark P as core.
6. Check for any cluster in determined eps, if there is no cluster, create a cluster register.
7. Include the related point to the cluster.
8. Include not clustered ones in N to this cluster and mark them as visited.
9. Mark noise point in N as border

Figure 4.4. DBSCAN algorithm (LIU et al., 2007).

K-Means: ToTRoP benefits from K-Means for selection of start points. K-Means algorithm has a widely-known and simple logic. Basically, it makes clustering by considering centroids of clusters. As parameter, it takes only cluster number (Deelers and Auwatanamongkol, 2007). In ToTRoP, route number is selected as cluster number. Implementation of the algorithm is developed as SQL-based in PostgreSQL. The K-Means algorithm is as follows:

The K-means algorithm

1. Select randomly initial centers in the dataset as many as cluster number.
2. Each point in the dataset is clustered according to closest centeroids.
3. Recalculate centroid of every cluster.
4. Repeat steps 2 and 3 until new centroids are stable.

Figure 4.5. The K-Means Algorithm (Deelers and Auwatanamongkol, 2007).

Graph Theory: Processing on geometric data is rather difficult and slow. For algorithmic processes to be made on these types of information, usage of functional algorithms is enabled by converting data to known structures.

In addition to this, an understandable methodology can be developed. Road information previously transferred is defined in graphic format. Within information in this format, finding route from one point to another is rather difficult, but this problem can be easily resolved with graph structures.

A graph is shown as G=(V,E). Vertex list is represented with V. Connection between two vertex is represented by Edge, and E shows its list. (Desaulniers and Hickman, 2007).

Figure 4.6 shows a sample graph. Figures shown with A, B, C and D represent vertex, and connections represent edges.

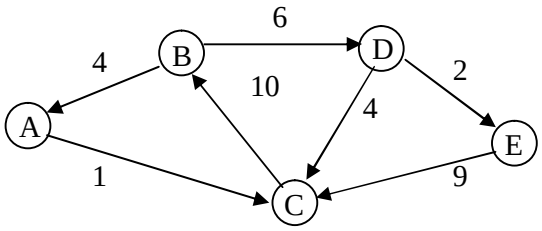


Figure 4.6. A sample graph.

Road map transferred with assign_vertex_id function coming within PgRouting module is transferred to graph structure.

Dijkstra: In order to solve the shortest path problem, an algorithm for the single source problem is developed by Dijkstra, and Floyd developed an algorithm for all pairs problem (Takaoka, 1998).

Dijkstra algorithm is used to find the shortest route from one node to another within ToTRoP. Dijkstra algorithm computes the path with the lowest cost by visiting all nodes from a given source. Here, the cost should not be considered as distance. It can be the travel time or traffic light numbers. These values are defined as attribute on the edges.

4.3. Tool Functions

In this section, tool functions and algorithms used to implement these functions are explained. Figure 4.7 shows the UML diagram of the ToTRoP.

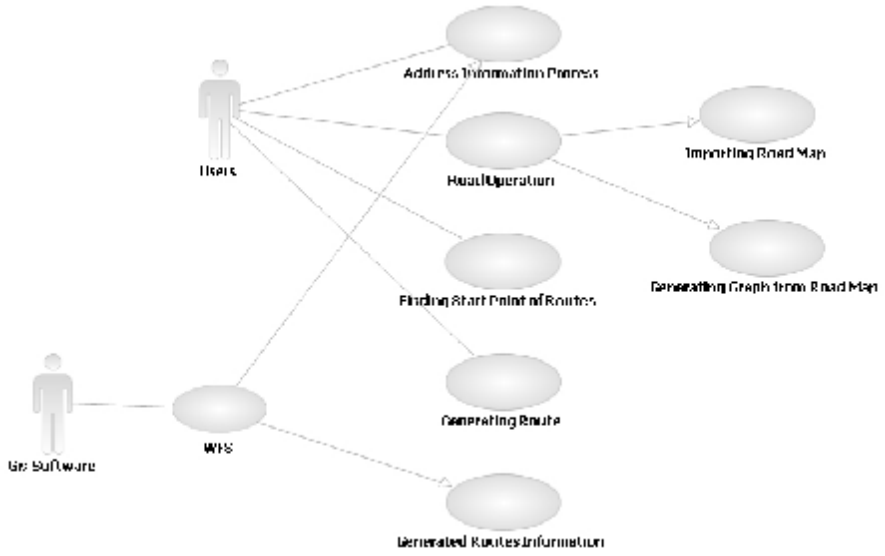


Figure 4.7. The UML diagram of the ToTRoP.

Address Information Process: As it is noted before, address data is usually available in textual format, converting it from this data format to geographic data format is an important part of the processing. This is achieved by using GoogleMap API developed by Google. This API is accessible through the internet and it is a free service provided by Google.

Importing Road Map: Route maps are available on the internet as paid or free. One of the most popular vectorial graphic files is SHP (ShapeFile). It is a vector based graphic file supported by Esri. PostgreSQL has a service which automatically transfers SHP file to DB.

SHP file is stored into the "Ways" table. There are two fields in this table. These are gid and the_geom. Gid is the primary Key, the_geom keeps graphic information which is vectorial.

Generating Graph from Road Map: The transferred route map is to be converted to a graph structure. The imported road map includes line records. However, the graph structure is necessary so road map is converted to the graph structure. To do this, start and end points of a line are analyzed and node and edge are generated. To perform this process "assign_vertex_id" function in pgRouting is used.

As a result of these processes vertex table which contains nodes is generated. Id number of the node is assigned to edge and target columns which are in ways table. Length of an edge constitutes the cost. Thus, a graph is completed.

Finding The Start Point of The Routes: The first process of data processing is the selection of the starting points for the routes. Selecting the start points is as important as generating the routes. Opting for appropriate start points increase the performance of roads to be generated.

These steps that must be taken are as follows:

1. By using DB-Scan clustering method, the points in low density regions are eliminated from the graph.
2. Angle between each point and the target point is calculated.
3. By using K-means clustering method, the data is divided into the groups as many as the route number.
4. Among the points in each the cluster, the remotest point to the target is selected and added to the route start list.

DB-Scan Cluster Method and Implementation: Figure 4.8 shows the distribution of the students who register at Çukurova University in 2009 and the road map of Adana after data format conversion is done. In this figure, 2340 student addresses

are represented by dots, and main roads of Adana are represented by red lines. The picture clearly shows the density distribution of the students with respect to the campus location.

Selecting low density and far points as starting points may generate inefficient routes. To solve this problem, the DB-SCAN clustering method is used. This method first clusters the address points, and then eliminates low density address points.



Figure 4.8. Student addresses on the main road map of Adana.

Figure 4.9 shows the student addresses on the main road map of Adana after the elimination of low density points. The elimination step discarded only 80 points which corresponds to 3.3.% of the data. DB-SCAN clustering prepares the data for more effective routing operation.



Figure 4.9. Student addresses after the elimination of low density points.

Computing Angle Between Two Paths: To compute the angle between two paths, the tangent formula is used. Basically, it is assumed that when moving from one point to another, the shortest distances are linear routes. A linear clustering is done by clustering points according to their angular information. In Figure 4.10 angular information of two points are shown.

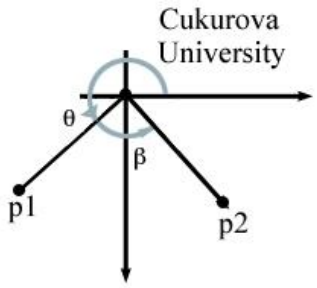


Figure 4.10. The computation of the angle between two paths.

K-Means Cluster Method and Implementation: K-Means algorithm is used to enable intra-cluster similarity at minimum level, and inter-cluster differences at maximum level. The points are clustered as many as the intended line count, according to their angular information. Figure 4.11 shows the addresses after K-Means clustering is performed. Each cluster is represented by a different color.

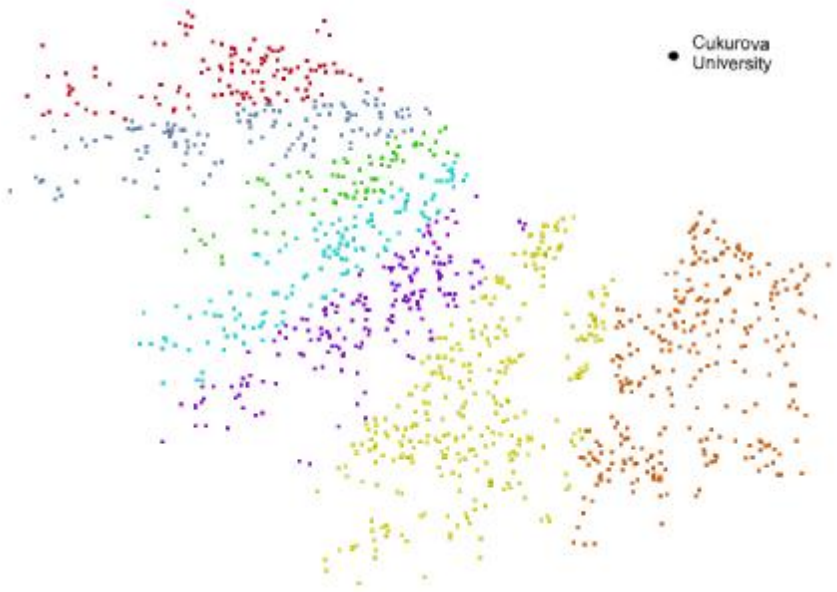


Figure 4.11. The distribution of the addresses after K-Means clustering is performed.

Selecting Start Point in the Clusters: The furthest point to the target point is chosen as a starting point and the route is generated from this starting point. Thus more students benefits from the routes.

Generating A Route: After choosing the starting points, the second step is determining the routes between these points and the target point. To do this, two processes are executed in ToTRoP: The first one generates routes according to the shortest distances, and the second one builds a structure enabling maximum number of students to benefit from the roads with minimum cost. In the first one, by using Dijkstra algorithm, roads are generated according to minimum cost. In the second one, a complex work flow is generated to find multi-routes. A complex work flow is built and called as CWF. An important issue to consider while generating multiple routes is the prevention of overlap between routes. Another issue is the prevention of zigzag on the routes. Thus, it can be applied to real life routes. Two work flows are generated for this aim. These are as follows:

1. Calculating values to generate routes.
2. Generating routes.

Calculating Values to Generate Routes: To generate routes, selecting appropriate data and analyzing them within a system increase performance of routes. There are two kinds of information. The first one is the student data, and the second one is the road map.

In the previous work flow, the student data and road map is made available. In this process, student data is going to be transferred to a road map, thus necessary platform is made ready. For this process, work flow is as follows:

1. Merge the close points based on a distance value, the new points are called as Large Points (LP) and the count of the number of merged points becomes the weight of the LPs.
2. The LPs are associated with the road through the nodes. The closest node to a LP gets the weight of the LP.

With the first step, distinct points are grouped in the center points, and while generating routes, route distances are shortened. In the second step, student data is transferred to a road map and a source table which has nodes, edges, edge costs, and student data is built. In Figure 4.12, a sample network is seen.

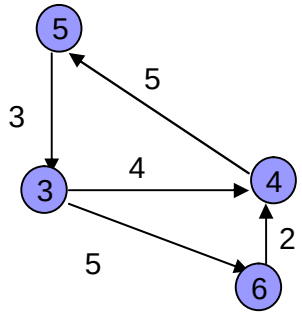


Figure 4.12. A sample network.

Generating Routes: Graph structure is necessary to generate routes. In the graph, the nodes are shown with their assignment cost values. Now, routes can be generated. Three types of route finding algorithms are used. These are Dijkstra and A* algorithms supported by pgRouting and modified Dijkstra algorithm written using SQL commands. In this section, mostly, steps taken for the third one is described. The other two methods are explained in pgRouting section in detail.

The workflow composed for the modified Dijkstra is as follows:

1. Routes are generated with Dijkstra algorithm from start points to the target point according to the shortest distance.
2. Then each node in the graph is associated to the closest generated routes by shortest path algorithm
3. Each route is regenerated in such a way that it meets maximum demand with minimum distance by considering demands of node in graph associated to itself by modified Dijkstra.

Step 1 and 2: In these processes, the shortest routes are generated from the start points to target point and each node in graph is associated to the closest the shortest routes. Here, the aim is to associate the values of nodes with the generated routes. Advantages of this are as follows:

1. To enable equal student distribution on the routes.
2. To provide more proper routes.

While performing these, pgRouting module is used, as it is faster then based SQL Dijkstra.

Step 3: The shortest way is not always the best one. Here, the goal is to enable maximum number of students to benefit from the routes. To achieve this goal, some modifications in Dijkstra algorithm are made. When each route is generated, in each step current total distance of the route / current total demand of the route ratio is considered. The pseudo code of Disjkstra's algorithm adapted from Wikipedia is given below. The modifications are designated by red and italic font in the pseudo code.

Modified Dijkstra Code

```
1  function Dijkstra(Graph, source):
2      for each vertex v in Graph:                 // Initializations
3          dist[v] := infinity                     // Unknown distance function from
                                                   // source to v
4          previous[v] := undefined                // Previous node in optimal path
                                                   // from source
5          student[v] := infinity                  // Unknown SN function from
                                                   // source to v
6          rate[v] := infinity                     // Unknown Rate function from
                                                   // source to v
7      dist[source] := 0                           // Distance from source to source
8      student[source] := exValue[source]          // SN from source to source
9      rate[source] := 0                           // Rate from source to source
10     Q := the set of all nodes in Graph          // All nodes in the graph are
                                                   // unoptimized - thus are in Q
11     while Q is not empty:                       // The main loop
12         u := vertex in Q with smallest dist[]
13         if dist[u] = infinity:
14             break                               // all remaining vertices are
                                                   // inaccessible from source
15         remove u from Q
16         for each neighbor v of u:               // where v has not yet been removed
                                                   // from Q.
17             alt1 := dist[u] + dist_between(u, v)
18             alt2 := student[u] + exValue[v]
19             rate:= alt1 / (alt2+1)
20             if rate < rate[v]:                  // Relax (u,v,a)
21                 rate[v] := rate
22                 dist[v] := alt1
23                 student[v] := alt2
24                 previous[v] := u
25     return dist[]
```

Figure 4.13. The pseudo code for the modified Dijkstra's algorithm adapted from Wkipedia (Wkipedia - Dijkstra, 2010).

Presentation: The presentation is implemented in two ways. The first one is presentation on JSF page on user's demands and the maps to be generated is displayed by using GoogleMap. The second one is Web Service.

Java Server Faces (JSF) Pages in ToTRoP: ToTRoP interface has five pieces of pages. These pages are called as DB Configuration, Address Information, Data Preparation, Start Point Selection, and Route Generation. The detailed

information on the functions and usages of these pages is given in User's Manuel chapter.

Web Feature Service (WFS) in ToTRoP: WFS within ToTRoP uses two types of data. The first one is address data, and on these data, add, update and delete, and query processes are performed. The second one is the generated route data. Thus, within data collecting period, usage of WFS services, and presentation of generated routes in result section are enabled.

Among WFS services, GetCapabilities, DescribeFeatureType, GetFeature, Transaction and GetFeatureWithLock functions are supported. Figure 4.14 shows class diagram for WFS module of ToTRoP. The detailed information on the usage of WFS service will be given in User's Manuel Section.



Figure 4.14. The class diagram of ToTRoP's WFS.

5. USER'S MANUAL of ToTRoP

In this section, information on usage and details of ToTRoP application is given. No installation for clients is necessary, the software can run on any web browser. Within ToTRoP, there are 5 pieces of screen and one WFS service.

The DB Configuration Page: The first page is the DB the Configuration page. In this screen, connection information for database used by the system is entered. Figure 5.1 shows the DB Configuration page. Hostname, port number, database name, user name and password areas must be filled by the user.

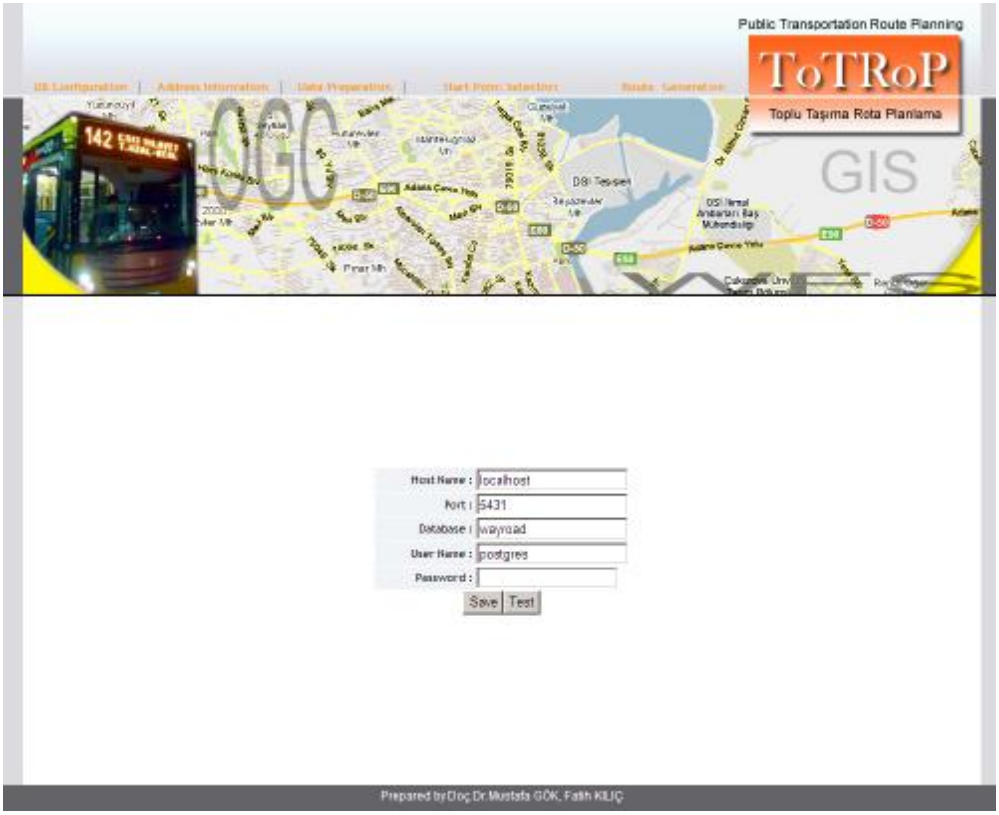


Figure 5.1. The DB Configuration page.

The Address Information Page: The Address Information page is the one on which entry, updating, deleting and query of address information is performed. Figure 5.2 shows the address information page. In this page, the text based address data is converted to latitude and longitude information with the help of

GoogleMap functions. Registration of this converted information is made on this screen. Gid field is the definition of the row data and is generated automatically. The key field is the helper information in accessing to the registration specified by user. For example, for persons identity number can be entered or this can be personal number for personal; and address information is real address information of the found location. By using address or key information, updating and deleting processes can be performed by making query in search section.

In this page, multiple address entry can be made. Address information can be entered in the text area, and up to 25 address entries can be made at once.



Figure 5.2. The Address Information page.

The Data Preparation Page: On this page, necessary parameters are set by using entered address information needed to generate routes, and externally transferred road map. Figure 5.3 shows the data preparation screen. The first process in this screen is determining the radius distance used to set large points, LPs. The address data entered with this process classifies addresses close to each other as the value

entered in radius distance field, and a new point list is formed. The number of classified addresses is designated as value to this point.

The second process is converting externally entered road map to graph structure. Tolerance value is the minimum distance between nodes when they are generated.

The final process is the combining the generated graph and LP information. With this action, process in data preparation page is completed.



Figure 5.3. The Data Preparation page.

The Start Point Selection Page: On this screen, the selection of the start points of routes is made. There are two types of selection processes. The first one is selecting start point through analyzing of data by the ToTRoP, and the second one is selecting this point manually. Figure 5.4 shows the start point selection page.

When "Show Address Location" button is pressed, the geographic data entered in address information page is shown on map. Thus, manual choice is facilitated. Also, distribution of data is seen.

In addition, in this screen target point information is entered. In this way, target point is determined. List of all start points and target point information can be seen in Start point list. Point information is seen on the map by pressing on sequence numbers on this list. Only, target point and manual start point information can be updated, also, deletion process can be performed in manual start points. Updating or deleting cannot be performed in start points determined by ToTRoP.



Figure 5.4. The Start Point Selection page.

The Route Generation Screen: Figure 5.5 shows the route generation screen. In this screen, data is classified as two groups. The first one is entry parameters, and the second one is generated result data. Routes are generated according to entered parameters and results are shown on the screen.

Required parameters:

- Algorithm: Three types of algorithm can be selected. These are Dijkstra, Modified Dijkstra and A* functions. The selected algorithm is used in route generation.
- Start Point Type: In this section there are two options. These are manual and automatic. According to this choice, related start points are used and routes are generated.
- The Walking Distance: ToTRoP computes the number of students benefiting from routes generated according to walking distance. Information of students benefiting from sum routes for both in sum and for each route.
- Bus Stop Distance: ToTRoP also shows bus stop points on routes. Bus Stop Distance determines the minimum distance between bus stop points.
- Bus Stop Number: The most dense points as many as intended number on the line are determined and marked on the map.

After this information is entered, the results are generated when submit button is clicked. Generated result information:

- Address Number: Total address number.
- Sum Benefit Student Number: Total number of students benefiting from all routes.
- Sum Cost: Total cost for generating routes.
- Route Distance: Length of the shown route.
- Route Benefit Student Number: Total of students benefiting from the shown route.
- Target Point: Geographical coordinate of the target point is shown.

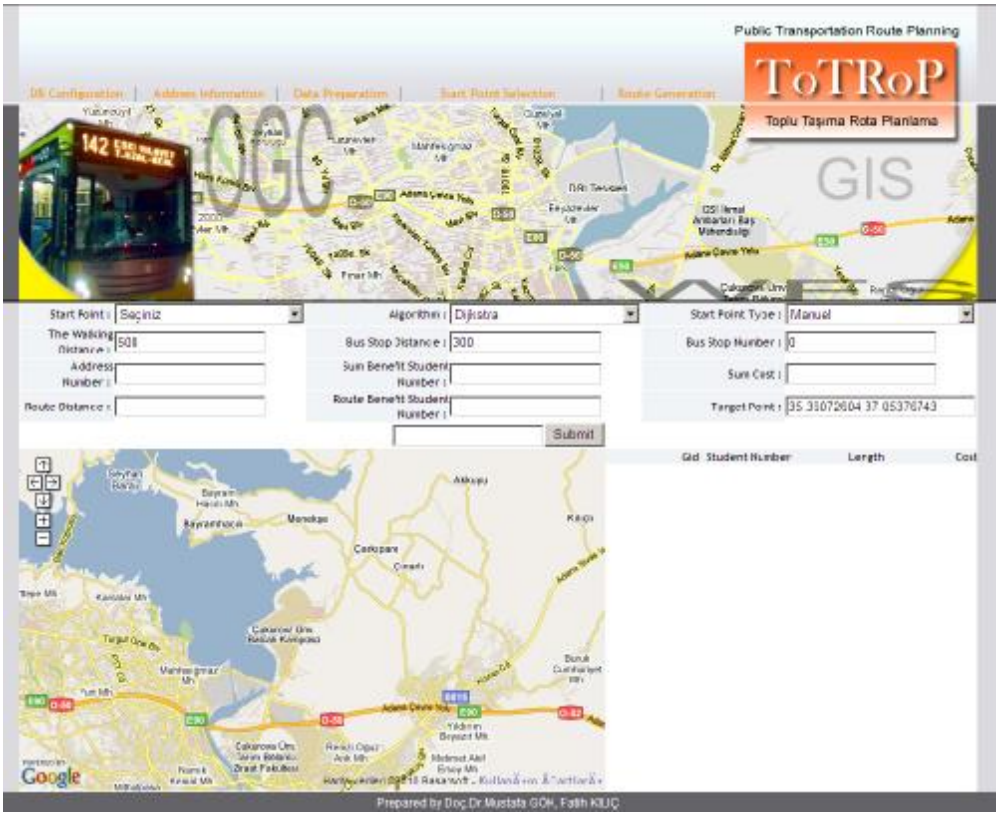


Figure 5.5. The Route Generation screen.

Related start points are selected from start point list, and information and route of that start point are shown on the map in the screen. In the table on the screen, length of the routes, student number benefiting from the routes are given as a list. In Figure 5.6, a route and its information are seen.

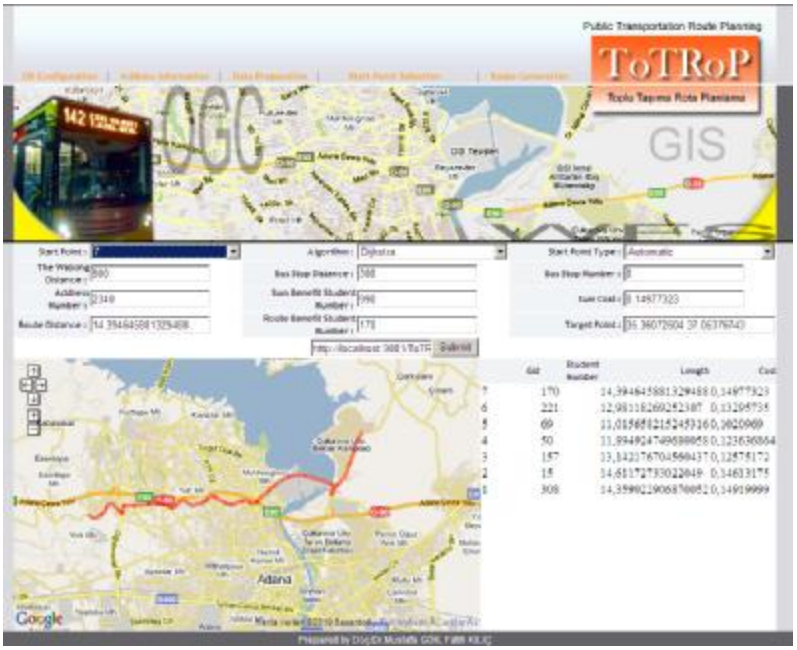


Figure 5.6. A route on the Route Generation Screen.

Using Web Feature Service (WFS) in ToTRoP: uDig desktop software can be use for WFS services of ToTRoP. it supports WMS, WFS, WCS defined by OGC. In Figure 5.7, uDig Screen is shown.

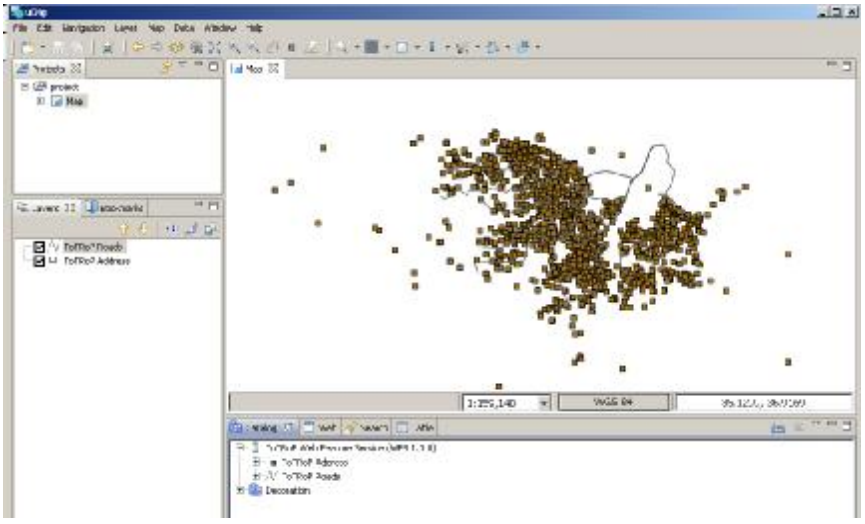


Figure 5.7. uDig desktop software.

6. RESULTS AND DISCUSSION

The results in this section are presented in two formats. The first format is the graphical format, the second format is the numerical format. Figure 6.1 and Figure 6.2, and Table 6.1 and Table 6.2 are generated using the real address of freshmen students enrolled at Çukurova University in Adana. Figure 6.3 and Figure 6.4, and Table 6.3 and Table 6.4 are generated using randomly generated pseudo addresses. The routes in Figure 6.1 and 6.3 are generated to optimize distance. Figure 6.2 and Figure 6.4 are generated to distance/student rate.

The Graphical Results: Figure 6.1 shows the seven routes generated with the goal of obtaining minimum route length according to the first test set. In this figure colored dots represent the student location data and the colored lines represent the routes. The color of each route is the same as the color of the students that can use the route. The black lines represent the roads shared by two or more routes. The black dots represent the addresses that are not assigned to any route.

Figure 6.2 shows the seven routes generated with the goal of obtaining minimum distance / student rate according to first test set. This route is generated based on the first routing process, however, this time student / distance rate is increased. The length of the overlapping roads in the second processing decreased. Route 2 after the second processing becomes more efficient to transport students in the far vicinity of the campus.

Figure 6.3 shows the seven routes generated to minimize the route length according to the second test set. Figure 6.4 shows the seven routes generated to minimize distance / student rate according to second test set.

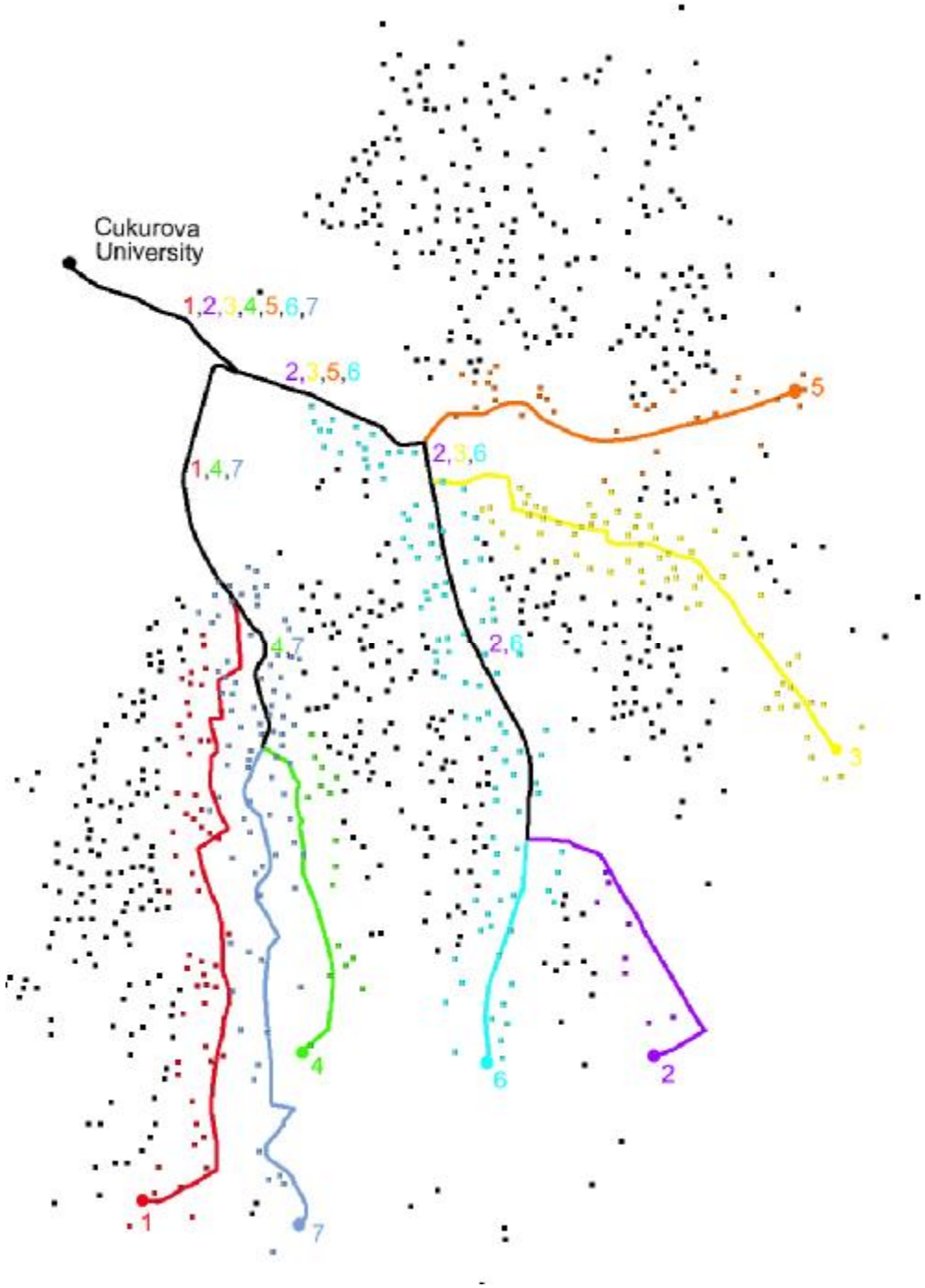


Figure 6.1. The routes generated by ToTRoP for minimum route distance according to first test set.

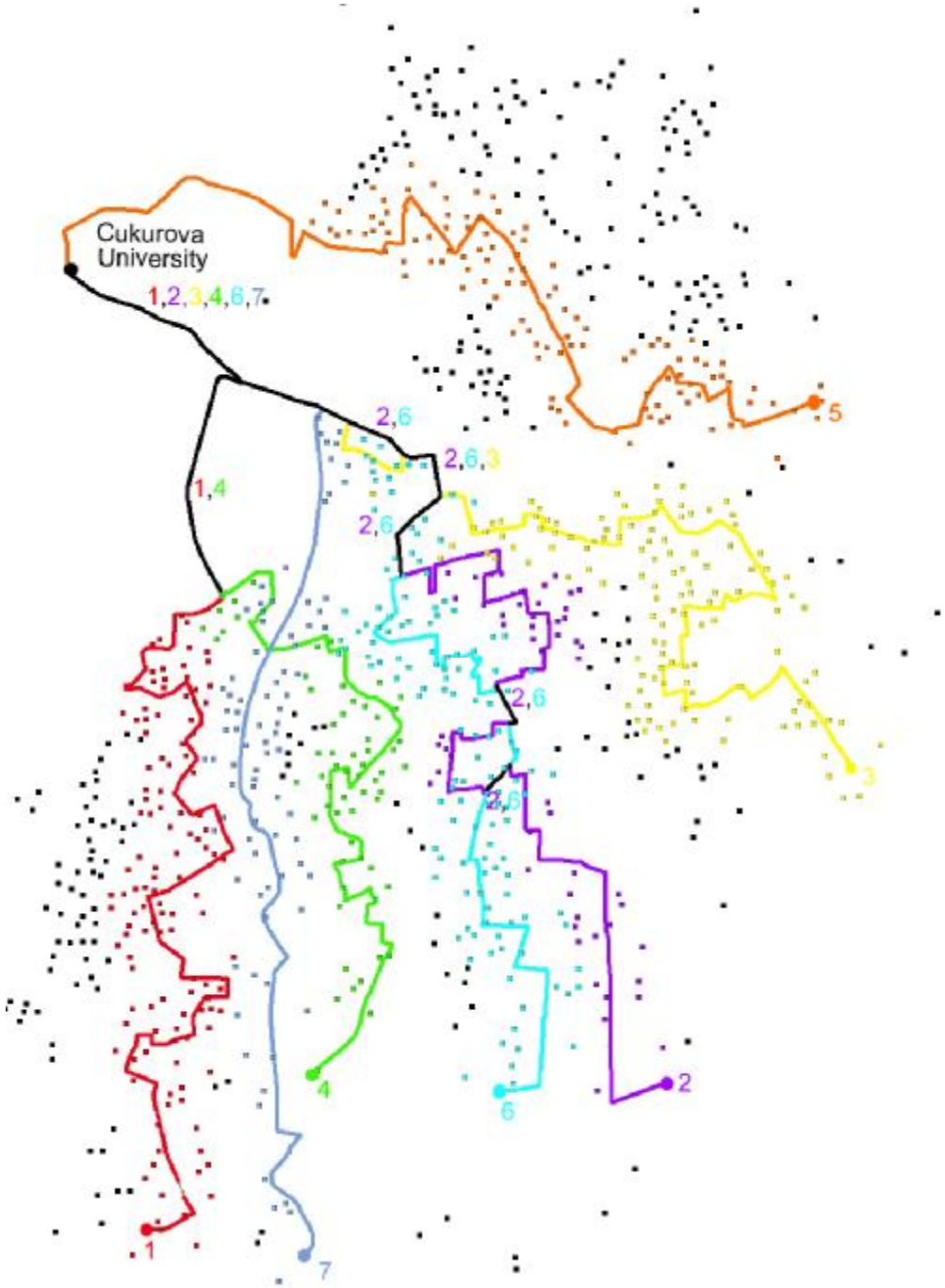


Figure 6.2. The routes generated by ToTRoP for minimum distance / student rate according to first test set.

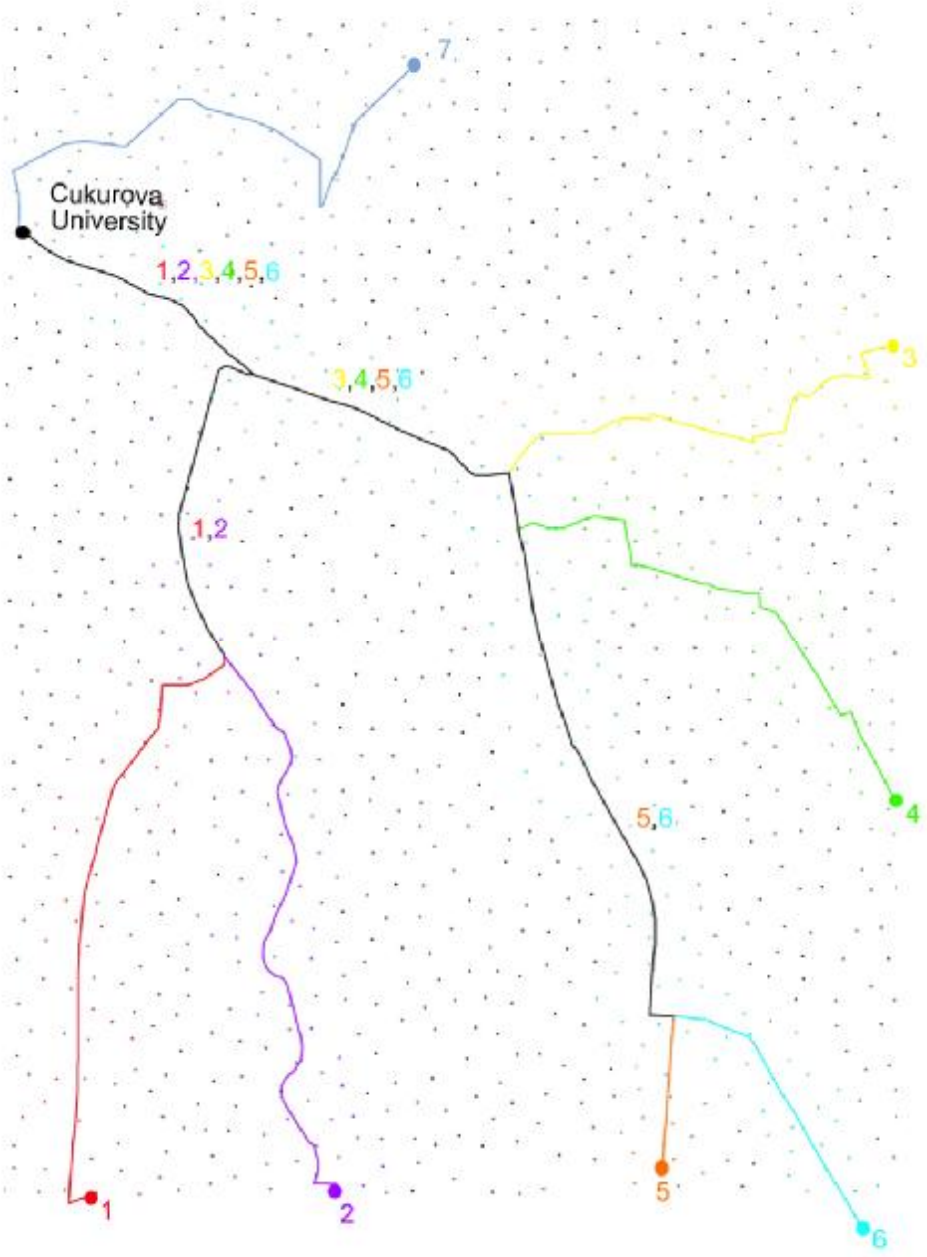


Figure 6.3. The routes generated by ToTRoP for minimum route distance according to second test set.

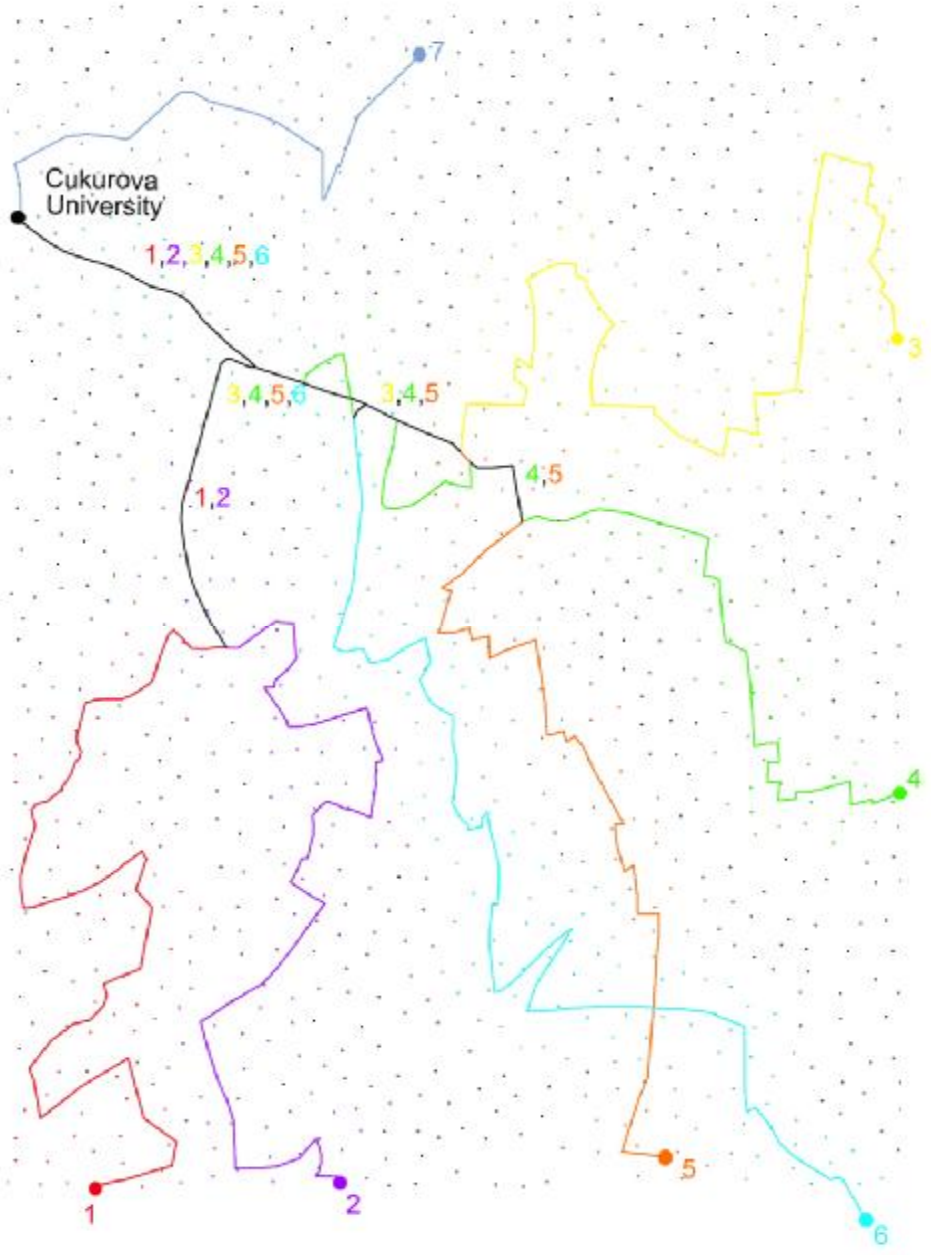


Figure 6.4. The routes generated by ToTRoP for minimum distance / student rate according to second test set.

The Numerical Results: Table 6.1 shows the results obtained by minimizing the distance using to first test set. Table 6.2 shows the results obtained by minimizes distance / student rate according to first test set. In results generated according to the first test set considering the minimum distance, a utilization rate of 42% is materialized. According to distance / student rate, a utilization rate of 76% is enabled. However, total route length is 92.396 Km for the first optimization, 124.208 km in for the second optimization. For routes generated according to minimum distance, 11 students fall into every km, while for routes generated according to minimum distance / student, 14 students fall into every km, approximately

Table 6.3 shows the results obtained by minimizing the distance using second test set. Table 6.4 shows the results obtained by minimizing rate using the second test set. By considering minimum distance, a utilization rate of 45% is materialized in the second test set. The utilization rate of 64% is archived for distance / student rate optimization. But, as total route is 74.525 Km for first optimization, it is 99.911 km for second optimization. For routes generated according to minimum distance, 31 students fall into every km, while for routes generated according to minimum distance / student, 32 students fall into every km, approximately.

Route	S. N.	Distance (km)	S. N. / Dist.
1	308	14,359	21,44996
2	15	14,611	1,026624
3	157	13,142	11,94643
4	50	11,894	4,2038
5	69	11,015	6,264185
6	221	12,981	17,02488
7	170	14,394	11,81048
Total	**990**	**92,396**	**10,7147**

Table 6.1. Routing that minimizes the distance according to first test set.

Route	S. N.	Distance (km.)	S. N. / Dist.
1	478	19,495	24,51911
2	127	21,402	5,934025
3	293	19,000	15,42105
4	194	16,859	11,50721
5	204	15,198	13,42282
6	303	17,560	17,25513
7	186	14,694	12,65823
Total	**1785**	**124,208**	**14,3711**

Table 6.2. Routing that minimizes distance / student according to first test set.

Route	S. N.	Distance (km)	S. N. / Dist.
1	293	11,095	26,40829
2	453	11,006	41,15937
3	208	10,319	20,15699
4	249	11,116	22,40014
5	61	11,783	5,17695
6	704	13,321	52,84889
7	324	5,885	55,05523
Total	**2292**	**74,525**	**30,7548**

Table 6.3. Routing that minimizes the distance according to second test set.

Route	S. N.	Distance (km.)	S. N. / Dist.
1	453	16,677	27,16316
2	555	15,39	36,06238
3	474	16,272	29,12979
4	305	14,17	21,52435
5	411	14,431	28,48035
6	725	17,086	42,4324
7	324	5,885	55,05523
Total	**3247**	**99,911**	**32,4989**

Table 6.4. Routing that minimizes distance / student according to second test set.

7. CONCLUSION

Due to the scale of the data and complexity of the road grid, route planning problem cannot be solved without the aid of computers and efficient software. This thesis presents a web based public transportation route planning tool, ToTRoP, which can be used by city decision makers to plan public transportation routes.

ToTRoP is used to generate routes in two steps. The first step minimizes the route length, the second step maximizes the number of individuals who benefit from the route. ToTRoP can be tuned to optimize for other constraints and criteria as well. The tool is tested for the transportation of students to a university campus but it can be also used to plan transportation to any other central facility such as an airport or a hospital or a stadium. Another use of ToTRoP might be that it can aid people to see their transportation options while making their housing decisions. The tool is expected to serve as a good basis for more complex software since it is designed using standard web technologies. It can be easily used as a standalone web service and also it can be integrated to other web based applications.

Currently, the tool can not be used to find routes to transport the total target population. However this is hardly the goal of a public transportation route planning. On the other hand, the benefits of the tool can be increased by adding other features. Such as, the roads can be classified (based on their size and capacity) so that more efficient routes can be generated. For example, the public buses are more likely to travel on main roads. This constraint may provide more realistic route generation options. Another improvement can be made by allowing to set obligatory transit point where these points can be included in the route manually.

REFERENCES

MAQUIRE D., GOODCHILD M., RHIND D.W., 1991. An Overview and Definition of GIS. Longman

THILL J., 2000. Geographic information systems for transportation in perspective. Transportation Research Part C 8, USA. 3-12 p.

AMIRIAN P., ALESHEIKH A.A., BASSIRI A., 2010. Standards-based, interoperable services for accessing urban services data for the city of Tehran. Computers, Environment and Urban Systems. 34, Iran. 309–321 p.

OGC, 2010, "OGC" , "Http://www.opengeospatial.org/", (Access Date: 3 Aug 2010)

FANG Y., LIN L., HUANG C., CHOU T., 2009. An integrated information system for real estate agency-based on service-oriented architecture. Expert Systems with Applications. 36, Taiwan. 11039–11044 p.

TAKAOKA T., 1998. Shortest path algorithms for nearly acyclic directed graphs. Theoretical Computer Science. 203, New Zealand. 143-150 p.

DESAULNIERS G., HICKMAN M.D., 2007. Public Transit Chapter 2. Handbook in OR & MS. 14, CANADA. 69-127 p.

GEBEYEHU M., TAKANO S., 2008. Demand Responsive Route Design: GIS Application to Link Downtowns with Expansion Areas, Journal of Public Transportation. 11, JAPAN. 43–62 p.

CURRENT J.R., REVELLE C.S., COHON J.L., 1985. The maximum covering/shortest path problem: A multiobjective network design and routing formulation, European Journal of Operational Research. 21, USA. 189–199 p.

CURRENT J.R., SCHILLING D.A, 1994. The median tour and maximal covering tour problems: Formulations and heuristics, European Journal of Operational Research. 73, USA. 114–126 p.

MATISZIW T. C., MURRAY A. T., KIM C., 2006. Strategic route extension in transit networks, European Journal of Operational Research. 171, USA. 661–673 p.

BAPNA R., THKUR L. S., NAIR S. K., 2002. Infrastructure development for conversion to environmentally friendly fuel, European Journal of Operational Research. 142, USA. 480–496 p.

BAAJ M.H., MAHMASSANI H., 1995. Hybrid route generation heuristic algorithm for the design of transit networks, Transportation Research C. 3(1), USA. 31–50 p.

MAUTTONE A., URGUHART M.E., 2009. A routeset construction algorithm for the transit network design problem, Computers & OperationsResearch. 36, URUGUAY. 2340-2449 p.

PATTNAIK S.B., MOHAN S., TOM V.M., 1998. Urban Bus Transit Route Network Design Using Genetic Algorithm, Journal of Transportation Engineering. 1998, USA. 368–375 p.

DAVIS S., 2007, GIS for Web Developers, The Pragmatic Programmers LLC, ISBN-10: 0-9745140-9-8, 108-133 p.

GREEN R., 2002, **"**Designing Enterprise Applications with the J2EETM Platform", http://java.sun.com/blueprints/guidelines/designing_enterprise_applications_2e/introduction/introduction3.html (Access Date: 3 Aug 2010)

SUN MICROSYSTEMS-JSF, 2010, **"**JSF Technology Documentation", "http://java.sun.com/javaee/javaserverfaces/reference/docs/index.html" (Access Date: 3 Aug 2010)

SUN MICROSYSTEMS-J2EE, 2002, **"**J2EE 1.3 Tutorial", "http://java.sun.com/j2ee/tutorial/1_3-fcs/index.html" (Access Date: 3 Aug 2010)

HUNTER J., 2001, Java Servlet Programming, O'Reilly & Associates, ISBN 1-56592-391-XE

IBM – AJAX COMPONENT, 2010, **"**AJAX components", "http://publib.boulder.ibm.com/infocenter/radhelp/v7r0m0/index.jsp?topic=/com.ibm.etools.jsf.doc/topics/rjsfajaxRefreshRequest.html" (Access Date: 3 Aug 2010)

GOOGLEMAP., 2010, **"**GoogleMap", "http://en.wikipedia.org/wiki/Google_Maps" (Access Date: 3 Aug 2010)

POSTGRESQL, 2010, **"**Documentation", "http://www.postgresql.org/docs/" (Access Date: 3 Aug 2010)

PGROUTING, 2010, **“**pgRouting Project”, “http://pgrouting.postlbs.org/” (Access Date: 3 Aug 2010)

PATRUSHEV A., 2007. Shortest path search in real road networks with pgRouting. Free and Open Soruce Software For Geospatial. 2007

PHILIPONE C., KASTL D., 2007. Web-based Routing: An Introduction to pgRouting with OpenLayers. Free and Open Soruce Software For Geospatial. 2007

FWTOOLS., 2010, **“**FWTools: Open Source GIS Binary Kit for Windows and Linux”, “http://fwtools.maptools.org/” (Access Date: 3 Aug 2010)

UDIG., 2010, **“**uDig”, “http://udig.refractions.net” (Access Date: 3 Aug 2010)

MICROSOFT., 2010, **“**Service Oriented Architecture (SOA)”, “http://msdn. microsoft.com/en-us/library/bb833022.aspx” (Access Date: 3 Aug 2010)

OGC-WFS, 2010, “Web Feature Service”, “http://www.opengeospatial.org/ standards/wfs”, (Access Date: 3 Aug 2010)

DUAN L., XU L., FENG G., LEE J.,YAN B., 2007. A local-density based spatial clustering algorithm with noise. Information Systems. 32, China. 978–986 p.

LIU P., ZHOU D., WU N., 2007. Varied Density Based Spatial Clustering of Applications with Noise. IEEE, China

DEELERS S., AUWATANAMONGKOL S., 2007. Enhancing K-Means Algorithm with Initial Cluster Centers Derived from Data Partitioning along the Data Axis with the Highest Variance. International Journal of Computer Science. 2, China

WKIPEDIA - DIJKSTRA, 2010, “Dijkstra's algorithm” , “http://en.wikipedia.org /wiki/Dijkstra's_algorithm”, (Access Date: 3 Aug 2010)

CURRICULUM VITAE

I was born in Adana, Turkey in 1981. I graduated from Mersin University, The Department of Computer Engineering in 2002. I worked as a computing specialist in the IT Department of Çukurova University between 2002 and 2006. After that I worked as a software engineer in the IT Department of Isdemir between 2006 and 2009 and now I am working as a engineer in the IT Department of Çukurova University.