# A Note on Route Planning in Transportation with Open Sources

Jiangshan MA[1] and Daisuke FUKUDA[2]

[1] Transport Study Unit, Department of Civil Engineering, Tokyo Institute of Technology, O-okayama, Meguro-ku, Tokyo 152-8552, Japan; PH +81 (3) 57342577; E-Mail: ma.js@plan.cv.titech.ac.jp

[2] Transport Study Unit, Department of Civil Engineering, Tokyo Institute of Technology, O-okayama, Meguro-ku, Tokyo 152-8552, Japan; PH +81 (3) 57342577; E-Mail: fukuda@ plan.cv.titech.ac.jp



## ABSTRACT

Route planning is one of the core issues in transportation fields such as logistics, route guidance and traffic assignments. Many useful algorithms dealing with various problems have been proposed in the past and some of them have been implemented in commercial software, while some others are still far from practice. In recent decades, the open source GIS has been developing very fast and many excellent projects have been built. In this paper, we make a note on route planning in transportation by utilizing such open sources. A data storage model as well as the architecture of open sources is introduced. As an example, the process of developing the hyperpath algorithm is illustrated. This paper may be useful for those who would like to customize GIS-based transportation analyzing tools by themselves for various purposes.

## INTRODUCTION

Transportation is one of the most important and increasingly popular applications of GIS (Wiggins et al. 2000). Geographic information systems for transportation (GIS-T) are interconnected hardware, software, data, people, organizations and institutional arrangements for collecting, storing, analyzing and communicating particular types of information about the Earth (Miller and Shaw, 2001). Many fields in transportation such as travel behavior analysis, urban transportation planning, network analysis, and logistics have taken advantage of GIS tools to achieve objectives more efficiently, conveniently and intuitively.

Commercial GIS software such as MapInfo and ArcGIS are costly and not specially designed for transportation studies. Transportation software such as TransCAD and CUBE also provide GIS modules for visualizing planning. Although they are extensible by providing API to developers, many researchers tend to get lost in the large complicated libraries of commercial GIS software. Besides, the implementations of provided API are invisible, which results in limited flexibility.

For these reasons, open sources are usually considered as alternatives which provide a cheap and efficient way to accomplish a specific study. Open sources in transportation studies have been increasing over the past years and several open source projects have been developed. These projects are of different purposes such as microscopic traffic simulation, traffic assignment, transportation planning, and traffic management. In the following, several projects and their integration of open GIS are reviewed. As the development of open source transportation projects lags behind open GIS projects, earlier open projects tend to integrate less open GIS features.

- MITSIMLab (written in C++ under MIT License)

MITSIMLab was developed at the MIT Intelligent Transportation Systems (ITS) Program (Yang 1997) for evaluating advanced traffic management systems (ATMS) and route guidance systems (RGS). As an early project, MITSIMLab is poor in GIS support and is independent of any open GIS projects.

- TRANSIMS (written in C++ under NOSA-1.3)

TRANSIMS is an integrated set of tools developed to conduct regional transportation system studies which was originally developed by Los Alamos National Laboratory (Nagel et al. 1999). It was first closed-source and then became open-source in 2006 with support from the Federal Highway Administration (FHWA). TRANSIMS supports some GIS data formats such as shapefile regarding network editing and visualization in itself and is independent of open GIS projects.

- MATSim (written in Java under GPLv2)

MATSim is a framework which implements large-scale agent-based transport simulations (Raney et al. 2003). It is currently developed by two groups: Transport Planning at the Institute for Transport Planning and Systems (IVT), Swiss Federal Institute of Technology Zurich, and the Transport Systems Planning and Transport Telematics at the Institute for Land and Sea Transport Systems, Technische Universität Berlin. It provides GIS support using open GIS projects of GeoAPI, JTS and GeoTools.

- UrbanSim (written in Python under GPL)

UrbanSim is simulation software for supporting planning and analysis of urban developments, incorporating the interactions between land use, transportation, the economy, and the environment; it is maintained mainly by the University of California Berkeley team. The initial implementation was in Java and has been distributed since 1998 (Waddell 2002). The software architecture was modularized

and re-implemented in Python beginning in 2005, making extensive use of the Numpy numerical library. Now the software has been generalized and abstracted from the UrbanSim model system, and is referred to as the Open Platform for Urban Simulation (OPUS) (Waddell 2005), in order to facilitate a plug-in architecture for models such as activity-based travel, dynamic traffic assignment, emissions, and land cover changes. Mapnik is utilized to create maps.

- SUMO (written in C++ under GPL)

SUMO is mainly developed by employees of the Institute of Transportation Systems at the German Aerospace Center (Behrisch et al. 2011) as a microscopic and continuous road traffic simulation package designed to handle large road networks. It is dependent on Proj and GDAL open GIS projects, which deal with coordinate conversion and raster map processing in respective.

- IRIS (written in Java under GPL)

IRIS is an open-source Advanced Traffic Management System (ATMS) software project developed by the Minnesota Department of Transportation. It is used by transportation agencies to monitor and manage interstate and highway traffic. It takes a PostGIS-enabled PostgreSQL database to save spatial data and utilizes the open GIS project Mapnik for rendering a background layer.

Among the software, the use of open GIS varies according to their core functions. In general, microscopic simulation software requires less GIS support than macroscopic simulation software. Route planning plays a fundamental role in simulation software, especially for conducting traffic assignment required in macroscopic simulation tools. Depending on research objectives, algorithms including shortest path algorithms, time-dependent shortest path algorithms and/or other route planning are implemented. However, it is generally not easy for users to develop new algorithms for existing platforms since they could be still too large for most researchers. In this case, one may be interested in developing customized light platforms or developing customized plugins for existing platforms. Transportation studies using customized platforms/plugins are still few compared with those using commercial software directly. For example, Papinski and Scott (2011) developed a VBA plugin for ArcGIS as a GIS toolkit for route choice analysis.

In this paper, we focus on route planning issues which are quite common and fundamental in many fields in transportation planning. The objective of this paper is to make a note for developing route planning algorithms with open sources from research to practice, which could be useful for those who would like to develop a customized research tool. The remaining sections are organized as follows. The

second section proposes a spatial data model for route planning. In the third section, we introduce the open source architecture that we actually used in our developments. In the fourth section, we illustrate the development of GeoRouting.Net which implements the hyperpath route planning algorithm with the proposed open source architecture. The fifth and final section concludes this paper and discusses some related issues.

## SPATIAL DATA MODEL FOR ROUTE PLANNING

A comprehensive introduction about data models in GIS-T including the fundamental node-arc model, Linear Referencing System (LRS) model, dynamic segmentation and some other ITS-related data models can be found in Miller and Shaw (2001). For networks with GIS data stored bi-directionally, the node-arc data model can be represented as the example shown in Figure 1. The simple example GIS map is related to directional road links and stored bi-directionally in the database, which means that each entity corresponds to a row of data saved in the database. Link 3 and link 6 are one-way streets because only the directions of 2-3 and 4-3 are stored. Each link has two pointers which point to its start and end node, respectively. This bidirectional level GIS map is preferable because it can represent more directional details on networks. For example, for bus route planning, stops are usually represented at both sides of streets (Xia 2009). Furthermore, sometimes turn penalties at intersections are taken into consideration while calculating the optimal route, in which turn costs can be represented by a turn table, as illustrated in Figure 2 without changing the topology of the original network.

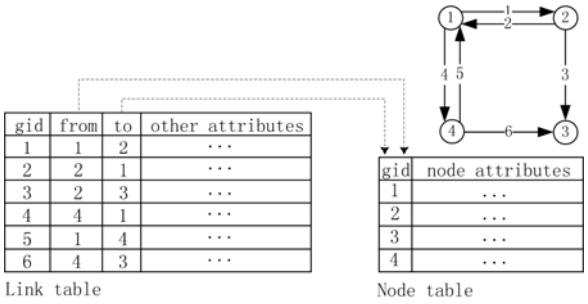



**Figure 1. Node-arc data model in relational databases with bi-directional level storage.**

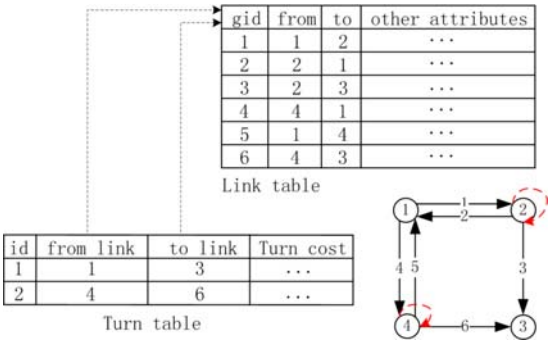



**Figure 2. Turn cost representation in node-arc data model.**

However, due to data limitations, all of the links in GIS maps may not be represented bi-directionally. A different data model which is able to deal with different GIS store types is needed. Besides, it might be inconvenient for this representation to reflect temporary changes in direction of traffic management.

We consider a data model shown in Figure 3. Network storage in relational databases: partially bidirectional level with both bidirectional level and unidirectional representations (e.g. ArcGIS use "FT" and "TF" fields to represent one-way streets of different directions). Note that the network is the same one as in Table 1 but with different storage representations in the database. Links between nodes 1 and 4 remain a bidirectional representation (especially important for transit design and lane-based studies). Links between nodes 1 and 2 are represented uni-directionally, which means the bidirectional street is represented as only one geometric entity in the GIS map. We take advantage of digital directions of these links and add an additional column "direction" which represents the relation between the stored objects and logical objects. As shown in Table 1, the real direction is determined by the direction code while the "from" and "to" columns only represent digital direction. That is, only geometry topologies are saved in databases while the transportation network topology will be built when loading the network into memory.

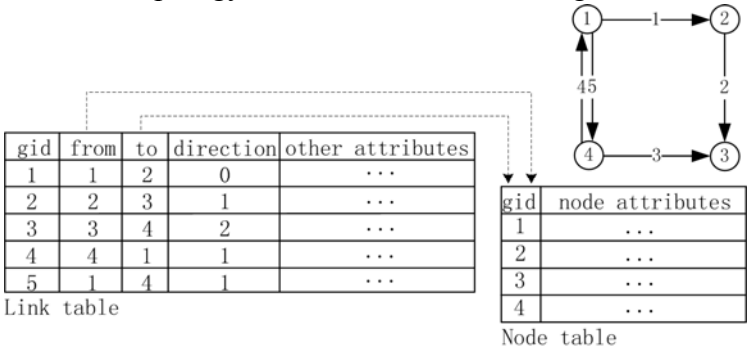



**Figure 3. Network storage in relational databases: partially bidirectional level.**

**Table 1. Direction codes for traffic network storage**

| Direction codes | Descriptions |
|---|---|
| 0 | Bidirectional link |
| 1 | Same as digital direction |
| 2 | Opposite to digital direction |

For example, in Figure 3, link 1 is digitalized from node 1 to node 2 according to its "from" and "to" fields and thus two links actually will be created when building the topology of the transportation network because the link allows driving in both directions. A network consisting of a node collection and a link collection can be built by the following two steps: (1) Load the node geometries into a node collection, with each node item corresponding with its geometry; (2) Load the link geometries into a link collection and keep creating new link items according to direction codes and set reference pointers to head and tail nodes until all link geometries are processed. Consequently, the geometric link with direction code 0 actually implies two links in transportation network topology.

After building a network topology, the link table would be like Table 2, which has the same network topology represented in Figure 1. To distinguish a link item in a transportation network in geometric network topologies, a new field of "id" (compared with "gid" which identifies geometric id) will be used as the unique identifier for each link in a network. The topology can be either pre-stored or it can be constructed on the fly.

**Table 2. Link table after building transportation network topology**

| id | gid | from | to |
|---|---|---|---|
| 1 | 1 | 1 | 2 |
| 2 | 1 | 2 | 1 |
| 3 | 2 | 2 | 3 |
| 4 | 3 | 4 | 3 |
| 5 | 4 | 4 | 1 |
| 6 | 5 | 1 | 4 |

## USING OPEN SOURCES FOR ROUTE PLANNING

In general, commercial software is huge and costly for most specific studies. Besides, commercial software is not as good as open sources in flexibility. There have been many possible open source solutions according to system environments and programming languages. One may refer to Ramsey (2007) for detailed introductions about open source GIS. The architecture of open source that we adopt is illustrated in Figure 4. In this solution, two tools named "GeoRouting.Net" and "PyGeoRouting" are developed. Both B/S (Browser/Server) and C/S (Client/Server)

modes are involved in this proposed architecture. Details about the two tools will be introduced in the fourth section.

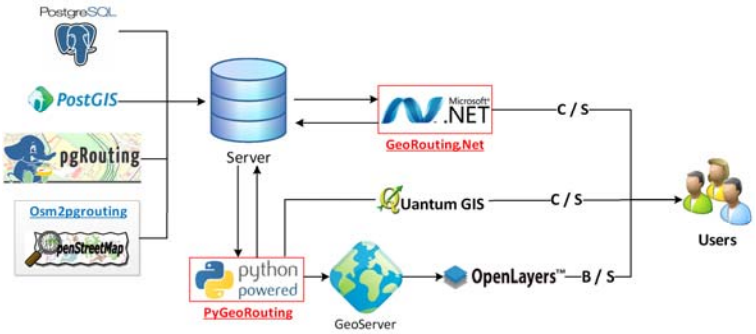



**Figure 4. An open source solution for route planning.**

In Figure 4, PostgreSQL is adopted as the database because it has an excellent GIS support with the extensions of PostGIS and PgRouting. Similar to the functionality of commercial ArcSDE, PostGIS enables the PostgreSQL to serve as a backend GIS database saving and managing spatial data. One of the practical problems before conducting route planning is the fact that there are usually only line layers in most available GIS maps of road networks, but the geometric topological relations between vertices and edges are necessary for route calculations. As a complement of PostGIS, PgRouting focuses on building geometric network topology as well as some classical routing issues. The functions that are interesting to route planning lie in its ability for geometric topology generation. Besides, classical algorithms such as Dijkstra, A*, and TSP (Traveling Salesman Problem) are provided in the form of extended functions in PostgreSQL by PgRouting. Another useful tool named Osm2PgRouting is used for directly getting free data which are ready for routing provided by OpenStreetMap (OSM).

In the illustrated architecture in Figure 4, two tools for route planning are involved. GeoRouting.Net is developed as a C/S application while PyGeoRouting is a library which aims to be used in either B/S web applications or C/S applications. GeoRouting.Net and PyGeoRouting are developed for different objectives and they are created in different stages of our study. GeoRouting.Net is developed in C# for higher developing efficiency, which is meaningful at algorithm design stage for saving the time of development. After an algorithm is designed, it is usually better to re-implement the algorithm for practical reasons considering running efficiency. PyGeoRouting is a python library which aims to attain higher running efficiency and cross-platform deployment as well. The core route calculation is implemented in C++ and then wrapped to python. Unlike GeoRouting.Net which also has GIS visualization support using the open source SharpMap library, PyGeoRouting plays

the role of a routing plugin and is expected to be used together with other open source projects. The visualization can be achieved by using QGIS, GeoServer and OpenLayers for C/S and B/S deployment in respective. QGIS is a standard desktop GIS application and PyGeoRouting can be simply called from its built-in python console. GeoServer is an open source map server by which the calculated routes can be distributed as web services. OpenLayers is an open source javascript library (Similar to Google Map API) which is used for client-side map display. Section 4 gives some details about these two tools, taking the development of hyperpath route planning algorithm as an example.



## AN EXAMPLE: HYPERPATH ROUTE PLANNING

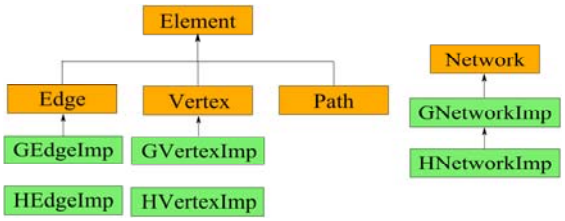
Under the architecture mentioned earlier, this section will introduce more details about GeoRouting.Net and PyGeoRouting. These tools are designed in general and are used for testing and implementing the hyperpath generating algorithm in our case. Different from the shortest path algorithms, the hyperpath algorithm finds a set of paths/links which are probabilistically used in networks with uncertain link travel times. The hyperpath algorithm (namely SF algorithm) is proposed by Spiess and Florian (1989) for frequency-based transit assignments and then transplanted to route guidance problems by Bell (2009). Different from shortest path algorithms, hyperpath finds a set of potential optimal links by considering the optimistic(minimum) and pessimistic(maximum) anticipation of travel times. Each hyperpath link will be associated with link choice possibilities, which enables a probabilistic route guidance that may help alleviate risks over travel time uncertainty. In this paper, hyperpath is taken as an example to show the development of two tools and one may refer to Spiess and Florian (1989) and Bell (2009) for methodological details.

- GeoRouting.Net for algorithm design

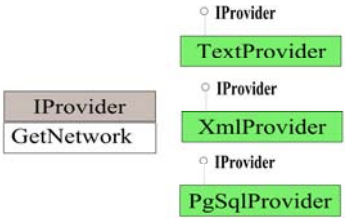
GeoRouting.Net is written in C#. Figure 5(a) shows the network model in which "Network" is an abstract class consisting of "Edge" and "Vertex" abstract classes. Both "Edge" and "Vertex" inherit from the base abstract class "Element" and are designed as generic abstract classes that allow making variant implementations. We implement a general network "GNetworkImp" which contains "GEdgeImp" and "GVertexImp" implementations. Hyperpath route planning requires some additional network information, so we implement a "HNetworkImp" class as a derived class of "GNetworkImp".

In Figure 5(b), the spatial data model for building transportation network topology introduced in section 2 is implemented in classes which implement the interface of "IProvider." It makes the rule that a "GNetworkImp" type network

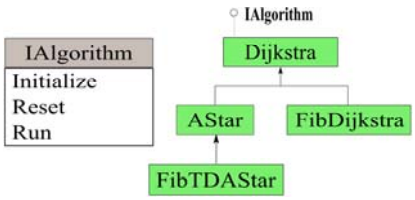
available simply by using the "GetNetwork" function of any data provider. So far we have implemented 3 data providers: "TextProvider" which deals with delimited text, "PgSqlProvider" which deals with PostgreSQL database and "XmlProvider" which deals with XML files. Figure 5(c) and Figure 5(d) show the class diagrams of implemented algorithms. Although our purpose is to implement hyperpath algorithms, classical shortest path algorithms are also implemented. The "IAlgorithm" interface abstracts the methods which are required for route planning algorithm design. "Initialize" prepares the necessary data before start route calculation and "Reset" recovers the algorithm calculation to the initialized status preparing for multiple runs. "Run" starts a route calculation and returns the running time consumed. New algorithms can be developed simply by implementing the "IAlgorithm" interface.



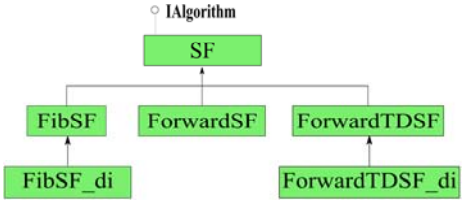

(a) Network topology




(b) Data providers




(c) *Shortest path algorithms




(d) *Hyperpath algorithms

**Figure 5. Fundamental diagrams of GeoRouting.Net.**

(*"Fib" refers to Fibonacci heap data structure, "TD" refers to time-dependent, "di" refers to the accelerated search in Ma et al., 2013)

Figure 6 illustrates the user interfaces of GeoRouting.Net including the setting panel and the GIS visualization form. The GIS visualization is supported by the open source SharpMap project. SharpMap is a light and easy-to-use mapping library for use in web and desktop applications. It provides access to many types of GIS data, enables spatial querying of that data, and renders beautiful maps. Because the route planning results are saved in the database, the visualization is independent of route calculation. In the case of developing hyperpath algorithm, we animate the node/link visiting process so that the differences among algorithms of different searching strategies but result in the same path can be clearly found. Also, the importance of links can be easily identified by imposing a theme over link choice

possibilities.

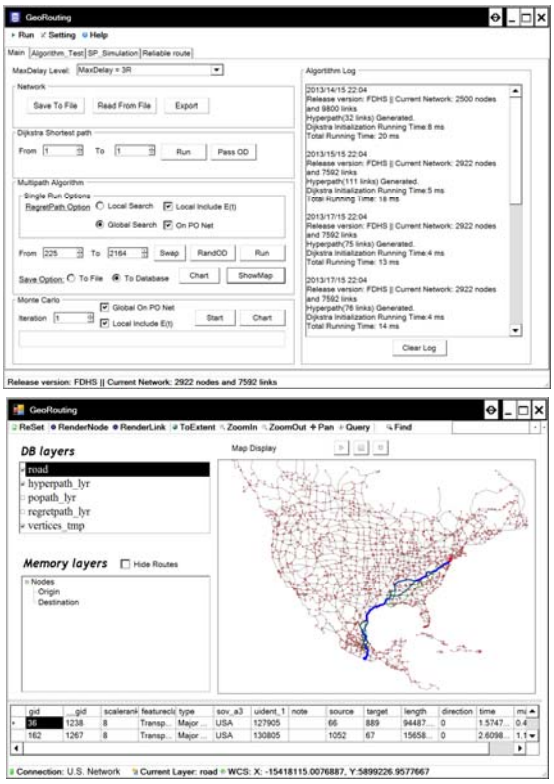



**Figure 6. User Interfaces of GeoRouting.Net.**

- PyGeoRouting for better performances

After the route planning algorithms are designed and tested in GeoRouting.Net, we re-implement the algorithm considering better efficiency and cross-platform compatibility. We implemented $SF^{di}$ algorithm (Ma et al., 2013), which is a variant of SF algorithm by implementing optimistic node potentials and node-directed search for better performances. At the stage of designing and testing algorithms, we focused on the correctness and stability of algorithms so that we mainly used the data structure of linked list or Fibonacci heap to maintain the updated nodes/links. More advanced data structures can be used to accelerate the running time. In the case of implementing the $SF^{di}$ algorithm, we use bucket data structure to generate optimistic node potentials together with Fibonacci heap data structure to maintain the node/link updating process. The C++ route planning algorithm implementations are wrapped as python module by using SWIG.

## CONCLUSIONS

In this paper we noted route planning with open sources. Taking the development of the hyperpath algorithm as an example, the process from research to practice is illustrated. We show that to build a customized platform/plugin is not difficult with the help of open sources. This paper can be viewed as a tutorial for

transportation researchers in developing or implementing their route planning based systems (for transportation planning, Dynamic Traffic Assignment, Route choice behavior study etc.) To the best of our knowledge, there is still no specific open source transportation platform repository to deploy plugins. An alternative solution is to deploy plugins on existing open GIS platforms. For example, QGIS has a repository where people can contribute the plugins they created. As for the programming language, one may start from Python since it has many open source supports and is very popular in GIS platforms (e.g. ArcGIS and QGIS). Besides, considering the fact that UrbanSim has been abstract to a framework, one may use many transportation related functions that are already there instead of developing new ones. We expect there will be a prosperous open source community in the field of transportation. Now we are revising PyGeoRouting and building Dynamic Traffic Assignment (DTA) modules on top of it and hopefully we will be able to release them soon.

## REFERENCES

Behrisch, M., Bieker, L., Erdmann, J. and Krajzewicz, D. (2011). "SUMO - Simulation of Urban Mobility: An Overview." *Proceedings of the Third International Conference on Advances in System Simulation*, 63–68.

Bell, M. G. H. (2009). "Hyperstar: A multi-path Astar algorithm for risk averse vehicle navigation." *Transportation Research Part B: Methodological*, 43, 97-107.

Ma, J. S., Fukuda, D., and Schmöcker, J. D. (2013.) "Faster hyperpath generating algorithms for vehicle navigation." *Transportmetrica A*: *Transport Science*, 9, 925-948.

Miller H. and Shaw S. (2001). "Geographic Information Systems for Transportation: Principals and Applications." Oxford University Press.

Nagel, K., Beckman, R. J., and Baret, C. L. (1999). "TRANSIMS for urban planning." *Proceedings of the 6th International Conference on Computers in Urban Planning and Urban Management*, Venice, Italy.

Papinski, D., and Scott, D. M. (2011). "A GIS-based toolkit for route choice analysis." *Journal of Transport Geography*, 19, 434-442.

Ramsey, P. (2007). "The state of open source GIS." ONLINE, http://www.refractions.net/expertise/whitepapers/opensourcesurvey/survey-open-source-2007-12.pdf.

Raney, B., Çetin, N., Völlmy, A., Vrtic, M., Axhausen, K. W., and Nagel, K. (2003). "An agent-based microsimulation model of Swiss travel: First results." *Networks and Spatial Economics*, 3, 23–42.

Spiess, H., and Florian, M. (1989). "Optimal strategies: A new assignment model for

transit networks." *Transportation Research Part B: Methodological*, 23, 83-102.

Waddell, P. (2002). "UrbanSim: Modeling urban development for land use, Transportation and Environmental Planning." *Journal of the American Planning Association*, 68, 297–314.

Waddell, P., Ševcíková, H., Socha, D., Miller, E., and Nagel, K. (2005). "OPUS: An Open Platform for Urban Simulation." *Presented at the Computers in Urban Planning and Urban Management Conference*, London, U.K.

Wiggins, L., Dueker, K., Ferreira, J., Merry, C., Peng, Z. and Spear, B. (2000). "Application challenges for geographic information science: Implications for research." *Education and policy for transportation planning and management, URISA Journal*, 12, 52–59.

Xia, X. (2009). "Bus Trip Optimization at Directional Level in GIS." Master Thesis, International Institute for Geo-Information Science and Earth Observation Enschede, The Netherlands.

Yang Q. (1997). "A simulation Laboratory for Evaluation of Dynamic Traffic Management Systems." Massachusetts Institute of Technology, PhD. Dissertation.