Dynamic and Time-Dependent Routing for Dynamic Dispatching Problem

by

Nazila Yaghini

Thesis submitted to the International Institute for Geo-information Science and Earth Observation in partial fulfilment of the requirements for the degree of Master of Science in Geo-information Science and Earth Observation, Specialisation: Geoinformatics

Thesis Assessment Board:
Chair: Dr. R.A. de By
External examiner: Dr. Jing Bie
Supervisor: Dr. O. Huisman
Second supervisor: Dr. R.A. de By

INTERNATIONAL INSTITUTE FOR GEO-INFORMATION SCIENCE AND EARTH OBSERVATION
ENSCHEDE, THE NETHERLANDS

Disclaimer

This document describes work undertaken as part of a programme of study at the International Institute for Geo-information Science and Earth Observation. All views and opinions expressed therein remain the sole responsibility of the author, and do not necessarily represent those of the institute.

Abstract

As cities continue to grow, urban traffic grinds to a halt during rush hours, and the need to optimize urban travel increases, for all traveller types. Particularly affected in the current situation are transportation logistics companies, responsible for delivering goods to customers distributed throughout different parts of a city. Today, optimisation of vehicles to serve customer orders on time, and routing the assigned vehicles through their destinations at the lowest cost requires computational solutions.

To help in dispatching of vehicles and to lead drivers via the most feasible roads, the routing problem is an important one to be considered and solved. The *time-dependent (dynamic)* vehicle routing problem is much more valuable than *static* vehicle routing because *it provides a more accurate way to model 'real world' problems than static ones*.

Understanding the relationship between dynamic dispatching and dynamic routing is essential to solve the problem of dynamic dispatching. The dynamic Pick-up and Delivery problem is formulated, and an objective function introduced which should be minimized to give an optimum schedule to serve customers. Three modules were presented each with defined responsibilities to provide a solution for the formulated problem.

Dynamic routing is defined as the core problem of dynamic dispatching problem and two different approaches are provided to solve it. The first approach presents a time-dependent A* algorithm which is adapted to be used with Tele Atlas speed profiles and should be replaced with the existing (static) A* algorithm in the source code of the pgRouting. The second approach presents a step-by-step algorithm to find a dynamic and time-dependent path using the static A* algorithm iteratively. An implementation of this algorithm is given as a function of postgreSQL/postGIS using a static least-cost path function of pgRouting. The function is tested using the Tele Atlas spatial data set and a generated table as a real-time updates table. Comparing routes showed the presented function in the postgreSQL/postGIS provides different travel-times for different times of a day and different days of a week. This result is the one which could be expected, having different travel-times/routes based on the different start times to travel. Although the provided function might not generate an optimum path to be used in the dynamic dispatching problem, but provides improved solutions over purely static ones. This function includes the real world last updated travel-times into the provided solution.

Acknowledgements

There are many people whom I would like to thank for their support and encouragement during the process of writing this thesis.

My sincere thanks to my first supervisor Dr. Otto Huisman for his insightful comments, motivation and kind support. I am thankful to my second supervisor Dr. Rolf de By for his comments and his help in postgreSQL/postGIS. I am grateful to Dr. Ivana Ivanova for her friendship and advice.

Big thanks to Tele Atlas, which provided data set. I am thankful to Dr. Mark Zuidgeest for the given papers in the transportation field. Special thanks to Mr. Gerrit Huurneman for his support and helpful comments. Thanks to Kiavash Bahreini for his help in C programming.

My heartfelt thanks to my family especially my father for his support and love, without him all these would not have been obtainable.

Table of contents

List of figures

List of tables

1. Introduction

1.1. Motivation and Problem Statement

1.1.1. Background

As cities continue to grow, urban traffic grinds to a halt during rush hours, and the need to optimize urban travel increases, for all traveller types. Along with increasing population, the general trend in most European cities has also been an increase in car ownership. Due to the changing relative location of residential, industrial and business districts in and around cities, journey-to-work travel-times in Europe continue to increase as a result of crowded road networks and longer travel distances. The dimension of this problem increases proportional to the sizes of the cities themselves: as the city becomes larger, transportation problems become larger too.

Particularly affected in the current situation are transportation logistics companies, responsible for delivering goods to customers distributed throughout different parts of a city. These companies are often faced with (heavy) traffic jams and unpredictable problems like accidents or road construction on their way to the intended destinations. As a result they may not be on time to serve their customers. These companies need to decrease the costs of their services to get more benefit of their businesses. The 'cost' can be defined as travel *length* (i.e. distance), *travel-time*, the *number of unsatisfied customers*, the amount of unused *capacities* of vehicles, and so forth.

In the past, cities were smaller and the numbers of orders were less, and human dispatchers were able to manage the company's vehicles in an efficient way, although it might not have been an optimal way. Today, optimisation of vehicles to serve customer orders on time, and routing the assigned vehicles through their destinations at the lowest cost requires computational solutions. This is because of the complexity of the problem: larger cities and more complex transportation networks, large numbers of orders, and large numbers of company vehicles, traffic, and etcetera.

Fleet management (or more precisely, 'the dispatching problem') represents a key field in transportation science. To help in dispatching of vehicles and to lead drivers via the most feasible roads, the routing problem is an important one to be considered and solved. The *time-dependent (dynamic)* vehicle routing problem is much more valuable than *static* vehicle routing because *" it provides a more accurate way to model 'real world' problems "* (Ichoua, Gendreau and Potvin, 2003; p.394) than static ones. In the static vehicle routing the static travel lengths or travel-times are the basis for computing the shortest or least-cost paths. A significant number of authors have tried to provide theory for the routing and dispatching problem, each with a different approach, (Cooke and Halsey 1966; Halpern, 1997; Goldberg and Harrelson, 2005; Du, Wang and Lu, 2007; Ichoua, Gendreau and Potvin, 2003; Li, Mirchandani and Borenstein, 2009). These are reviewed in detail in the following chapter.

1.1.2. The Dispatching Problem

The main components of the dispatching problem are a set of orders, vehicles and roads each with different characteristics and constraints and all related to each other. In the 'standard' problem, there is a company with a known number of vehicles and this company aims to serve its customers in a

way that *minimises* costs. Under 'real world' conditions, the orders arriving during a working day are completely dynamic and unknown before being placed. Vehicles availability time and location is also dynamic because they can be subject to breakdowns or accidents. In addition, the travel-times of road networks, which affect the costs of the company and the satisfaction level of customers, are also dynamic because (among other things) they depend on traffic conditions on the road networks and 'road feasibility', which refers to them being closed or under construction. These different components of the problem mean that static solutions are not satisfactory enough to account for all the dynamic phenomena of real world. The everyday problem, which transport/logistics and courier companies face has a dynamic property and requires real-time/dynamic solutions.

"*From a modelling standpoint, fleet management problems correspond to combinatorial optimization problems (e.g., vehicle routing, covering, or design problems) that are notoriously difficult to solve for human dispatchers, even in a static context*" (Crainic and Gendreau, 2004; p.10). Hence in many applications, the real world is not considered to be dynamic, for example, static known travel-times for road links. The main problem in all dispatching applications is the need to handle dynamic data and this makes the whole process dynamic. On the one hand, most of the orders can arrive at any time and therefore are not easy to be predicted precisely and need very fast assignments (the customer could place a phone or Internet order). On the other hand, immediately after any order completion by a vehicle, the vehicle should be assigned to serve a new order to optimize the solution.

Furthermore, today the applied policies result to have transitional storage, and goods should receive to the intended destinations without being stored (TNT Logistics, 2005). As a result, transportation logistics companies aim to optimize their costs and revenues by replacing their traditional methods (Gonzalez, 2006). It is important for a transportation company to be confident about the optimum use of vehicle capacities and efficiency of the planned routes between different order locations (Greenwood, Dannegger and Dorer, accessed in May 2009). Of course, the two aspects are not separate of each other in a complete dispatching problem, but they can be treated like this for the sake of simplicity. This research focuses on the second aspect of the problem using spatial databases.

According to Greenwood, Dannegger and Dorer (accessed in May 2009), available dispatching systems like Transportation Planner from i2 Logistics, AxsFreight from Transaxiom, Cargobase, Elit and Transflow, provide basic strategies and planned routes to solve the problem of transportation companies. Most of the mentioned transport management systems cannot handle the dynamic and unusual cases and are not capable of providing alternative solutions when events like vehicle accidents, vehicle breakdowns, order cancelation, etc. occur after assigning a schedule to a vehicle. The main reason for this inefficiency can be detected in the static solutions offered by these systems.

1.1.2.1. The Pick-up and Delivery Problem

According to Ghiani, Manni, Quaranta and Triki (2009; p.1), *"The Vehicle Dispatching Problem with Pick-ups and Deliveries is a problem faced by local area courier companies serving same-day pick-up and delivery requests for the transport of letters and small parcels"*. In a common

transportation logistics company, the pick-up and delivery problem is completly dynamic since the company receives new order placements frequently (Gendreau, Guertin, Potvin and Seguin, 2006). As a result, in each new customer call, a quick decision about the assignment of the most appropriate vehicle and its routing along the road network should be taken.

Figure1.1. explains a general assigning and routing problem for one vehicle. Let $+i$ and $-i$ are the pick-up and delivery location of order i, accordingly (Gendreau, Guertin, Potvin and Seguin, 2006). The current location of vehicle is point X and it is travelling to the associated pick-up location for order number 1. The last assigned path can be visualized as follow: first pick up order number 1, then pick up order number 2, then deliver order number 2, then pick up order number 3, then deliver order number 3 and finally deliver order number 1. Regarding to the last planed path and receiving a new order from customer number 4, the schedule should be changed to include the order number 4 pick-up and delivery locations in the vehicles path in a way that minimizes the whole travelling costs of the vehicle. This might result to have a completely different path comparing to the last one. This problem is a kind of general pick-up and delivery problem, which a vehicle can serve more than one order in the same time.

A *dynamic* dispatching problem is defined here as a problem of assigning vehicles to serve customer requests in an optimal way to reduce costs and/or increase revenues. The problem focuses on this fact that assigning of vehicles to different customers is based on spatial location of vehicles at the time of assignment and the *time-dependent* cost of travelling to the intended customer location for each vehicle.

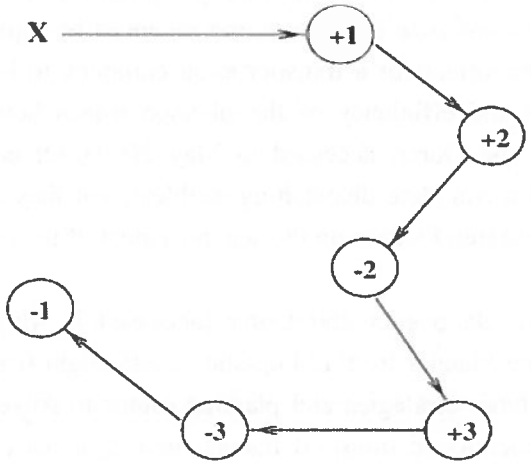

Figure.1.1. A planned route (Source: Gendreau, Guertin, Potvin and Seguin, 2006)

Compared to the static version, in dynamic dispatching the dispatcher is not aware of future data (e.g., orders to be serviced and traffic situation in road network) (Gendreau, Guertin, Potvin and Seguin, 2006). The data to be used in the solution is not completely known prior to get the final solution and change dynamically during the procedure and results to have uncertainties in the solution. The new changes should be applied to the obtained solutions frequently. In this case finding the final solution at the first step seems to be impossible. These problems can be found in various application types, like the *same-day local* pick-up and delivery problems, which represent the focus

of this research. In this problem type, there is a difficulty in assigning a dynamic route to vehicles since the pick-up and delivery locations and associated time windows for each order are being known dynamically and frequently during the day. The response time also is another important constraint, which should be taken into account in the provided solutions: "*In this context, many different factors must be considered when a decision about the allocation and scheduling of a new request is taken. The current location of each vehicle, their current planned route and schedule, characteristics of the new request, travel-times between the service points, characteristics of the underlying road network, service policy of the company and other related constraints should be considered*" (Gendreau, Guertin, Potvin and Seguin, 2006; p.158).

The procedure of formulating dynamic vehicle routing problems can be explained as basic problems in transportation field, which are not easy to be solved (Powell, Jaillet, and Odoni, 1995). A critical issue in real-time settings is the response time, and as a result, implementing just-in-time pick-up/delivery is very challenging (Chang, Wan and Tsang Ooi, 2008). Since the traffic situation is very unstable during a day, travel time of all road network links is dependent on time. Also the required time to serve orders in pick-up and delivery locations cannot be predicted properly. Providing unsuitable solutions for dynamic routing as the main part of a dynamic dispatching problem result to service lateness or early arrival of vehicle to the customer locations.

In a common pick-up and delivery problem a *node* can be defined as a junction, an *arc* can be defined as a road link, and the travel cost (static, time-dependent or dynamic) can be defined as road link travel time. This definition can be adjusted to consider the real world situations more precisely. Since the dimension of the time-dependent or dynamic dispatching problem are so big and have so many constraints to be considered, simplifying assumptions must likely be made to solve the problem. That is, we believe that at present only heuristic solutions are feasible. Table.1.1. illustrates four possible components of a dispatching model with three different kinds of data sets for each component.

	Real World data	Modelling/Prediction	Assumed State
Traffic			
Link	Flow Sensor, FCD, GPS, Traffic management center	Logical/Historical	Static traffic condition
Customer			
Location	Digital Map	Forecasted	Random generation of points
Time Windows	Getting from Customer		Random generation of time windows
Request	Storing on Demand	Modelling/Prediction	Random generation of requests
Vehicle			
Availability Time	Communication with drivers	Could be predicted based on real world data (current location and distance to the new destination).	Random generation of availability times
Availability Location	Communication with drivers	Could be predicted based on the planned route	Random generation of availability location
Current Location	GPS	Predicted based on the planned route	
Road			
Road Feasibility	FCD	Accidents can be predicted or modelled somehow but it will be very difficult.	Random construction or accident points, deterministic model of accidents

Table.1.1. Possible components of a dispatching model

1.2. Transportation and Other Sciences

1.2.1. Applications of GIS and Spatial Databases

Different aspects of transportation problems like transportation planning and management have been solved effectually in the past years, using GIS tools (Huang and Pan, 2007). Databases have also

expanded to handle the complex aspects of spatial problems like data management, modelling and analysis.

Nowadays, spatial databases and GIS methods are being used by a wide range of people, for both public and private transportation applications (ESRI webpage, accessed in May 2009). Using GIS technologies, fleet managers can manage different applications on the field of transportation in more efficient ways. As summarised in Venigalla (2005) and ESRI webpage (accessed in May 2009) specific applications include:

- *Mass transit/Public transit,*
- *Route planning and analysis,*
- *Bus dispatch*
- *Emergency response*
- *Automatic vehicle location and tracking,*
- *Transportation planning and modelling,*
- *Accident reporting and analysis,*
- *Paratransit scheduling and routing*

"Spatial databases support data management and spatial analysis in order to extract information and insight" (Goodchild, accessed in May 2009; p.4). For example, traffic modelling requires traffic pattern extraction. Accurate traffic pattern extraction needs the acquisition, management, analysis and visualization of very large and multivariate spatial time-dependent datasets of traffic information to find and derive meaningful patterns and relationships in space and time. The main reasons to suggest spatial databases in the solution for dispatching problem is because the management and integration of spatial data and data consistency are important issues and necessary in dynamic routing, which road travel-times are subject to many updates.

Since the ability of new computers in storing data via their powerful memories improved comparing to the past decade, provided solutions for routing are improved accordingly (Gonzalez and Machin, 2003). Most commercial software can provide reasonable solutions for transport related problems. Various types of commercial services like vehicle navigation systems are currently available in the market that can efficiently manage routing, scheduling and dispatching problems, including Arc Logistics Route software by ESRI, TomTom, and Caliper, to name a few. These systems typically use GPS or inertial navigation systems or a combination of both to determine the current locations of vehicles (Sivaram and Kulkarni, accessed in April 2009). These systems can be used in combination with dispatching systems to assign vehicle drivers to different customers and route them to the intended destinations. The planned path is optimized for distance or the most or the least used path, commonly (Pedersen, 1998). Although some commercial services to solve dynamic routing problems exist, most of the available scientific ones solve the non-dynamic problems (Network Analyst of ArcGIS and pgRouting).

The matrix in Table 1.2. illustrates a group of basic questions, which a dispatcher should be able to answer them during the assignment procedure or when he/she prepares a report (ESRI webpage, accessed in May 2009). The applications and abilities of a GIS system to help to the dispatcher to

answer these questions are provided in the same row as the question. This research mainly considers the third question, and attempts to find a dynamic spatial solution for it.

Question	Function	Application
Where are vehicles?	Vehicle Tracking	Dispatch
Where is the new customer?	Geocoding	Customer Address Matching
How do I route vehicle to the customer location?	Network Topology	Routing and Scheduling
Where have the vehicles been?	Route Logging	Post Trip Analysis
What is the potential demand?	Accessibility to Customer	Site Selection
Service from a presented warehouse?	Order Fulfilment	Customer Service

Table 1.2. Main questions in fleet management (Source: ESRI webpage, accessed in May 2009).

Although using spatial databases and GIS to solve the transportation-related problems are beneficial, but they cannot solve the complex optimization problems by themselves (Huang, Cheu, and Liew, 2004). Including models from other science fields to solve the problem is essential in this case.

1.2.2. Transportation and Operational Research

"Operational research principles, models, and methods are increasingly found at the core of advanced fleet management systems" (Crainic and Gendreau, 2004; p.9). While the ability and usefulness of GIS and spatial databases in data collaboration and evaluation have been established for wide range of transportation problems, one aspect can be determined as the minimization of total travel-time of vehicles and other costs (Huang and Pan, 2007). To solve this aspect of the problem optimization techniques are necessary and GIS cannot solve the problem solely. GIS should be supplemented with optimization techniques to solve the linear, nonlinear, and quadratic programming problems within a transportation problem (Huang, Cheu, and Liew, 2004). It is proven that combination of a GIS and different optimization methods provides more practical and efficient solutions for a dynamic pick-up and delivery problem (Abler 1987, Mark and Frank, 1989).

Using different optimization methods, solution policies and problem identification methodologies can be introduced by operational research (Crainic and Gendreau, 2004). Accordingly, to solve the problem of dynamic dispatching, GIS should be used together with travel-time simulation and optimization systems (Huang and Pan, 2007).

1.3. Aim, Objectives and Research Questions

Having precise data does not necessarily ensure a precise solution, but the lack of precise data is a big obstacle for finding a good solution. As discussed above, there is a lack of dynamic and time-dependent dispatching solutions. Specifically, a lack of dynamic solutions in the field of dynamic dispatching problem in existing literature, and reflected in existing software. For example, the pgRrouting module of postgreSQL/postGIS and Network Analyst of ArcGIS are not well-equipped to handle the dynamicity of the real world situations, and consequently in their proposed routing solutions. These issues represent an important part of a dynamic dispatching problem.

This research tries to provide a solution for formulating the time-dependent (dynamic) dispatching problem and solving the time-dependent (dynamic) routing problem using time-dependent and real-time travel-time information stored in a spatio-temporal database. It also aims to improve the functionality of existing mathematical models within this solution. The solution expands the functionality of spatial databases to route vehicles dynamically through the least-cost paths and generates routes to be used in an assignment procedure to serve different orders of customers with different time windows. Together, these results will be the contribution of the present work to the existing research field.

The overall objective is to develop a solution that fulfils the following tasks:

A) Formulating the dynamic dispatching problem
B) Finding an available mathematical algorithm for static shortest path problem and extending it to be dynamic and time-dependent
C) Finding a solution to handle real world travel-times for road links in an efficient way and use them in the dynamic routing solution
D) Finding an appropriate software that can handle the static routing to extend its functionality to be used for dynamic routing
E) Finding a way to predict the future travel-times in road links
F) Extending the software for dynamic routing
G) Testing the dynamic routing solution
H) Connecting the dynamic dispatching model with the dynamic routing solution

During this research project and in the remaining chapters, the following questions will be answered:

1. How we can formulate the dynamic routing problem as the core part of a dynamic dispatching problem?

2. Is there a suitable existing mathematical algorithm to solve the shortest path problem? If so, how it can be extended for dynamic shortest path problem?

3. What are the dynamicity scenarios for the defined dynamic routing problem?

4. How to handle real-time traffic information? Which criteria should be considered in travel-times of road links (Day of the week, Time of the day, Weather conditions, etc)? How to use the information to predict the future travel-times for each road link? Is it possible to

construct a model based on the historical or logical patterns of traffic congestions or it is better to have the real world road travel-times in a spatial database? Is it possible to define travel-time functions for road links?

5. How we can apply the extended algorithm in a spatial database?

6. What is the appropriate way to minimize the objective function in a dispatching problem, operational research methods or GIS spatial functions?

7. How should the proposed model assign or reassign the available vehicles to serve customers with using real-time travel information? How should the assigned vehicles be routed?

2. Literature Review and Conceptual Framework

2.1. Chapter Structure

As noted in Chapter One, the dynamic dispatching problem and within it dynamic routing problem are complex problems, consisting of several main components. To realize the state of the art for each component and to find the weak points of each component to be used in our solution, this chapter presents a review of relevant literature. Considering each of the different components of the problem, the review follows four main directions:

1. Formulating a dynamic and time-dependent dispatching problem using mathematical concepts;
2. Finding shortest path and solving the time-dependent routing problem;
3. Traffic modelling and travel-time estimation;
4. Optimization and assignment problem;

Together, these form the structure of this chapter.

2.2. Time-Dependent Shortest Path and Time-Dependent Routing

The fundamental problem of any routing process is finding a shortest path in a road network. Therefore, "*the computation of shortest paths is an important task in many network and transportation related analyses*" (Zhan and Noon, 1998; p.65). However, in the dispatching problem the running time of algorithm is also an important issue when a customer waits on the phone or internet to get the order acceptance of the transportation company. In addition, the efficiency of the shortest path is very significant in routing to guarantee the optimal path for vehicle. A path is the most acceptable one only if its achievement in the most unfavourable condition can be preferable than the others (Gang and Jian, 1998).

The shortest path problem is a classical problem (Dijkstra, 1959; Floyd, 1962; Spira and Pan, 1975; Aho, Hopcroft and Ullman, 1974). As the name suggests it is the problem of finding a shortest route between an origin and a destination and is applicable in different fields. Transportation is a field that in the most of its applications, shortest path solutions are required. In recent years in the transportation field, the shortest path problem is considered mostly in static graphs (graphs with non-varying travel costs for their links or arcs), this consideration is because of its interesting practical applications and its algorithmic challenges (Delling and Nannicini, 2008).

Up until today, many research works have attempted to solve the problem of finding the shortest path between an origin and a destination. Initial algorithms that were presented to solve this problem, paid attention to static networks only, including the well-known A* (Wikipedia, accessed in May 2009) and Dijkstra's algorithm (Dijkstra, 1959) proposed many years ago. A* is a graph search algorithm that provides the least-cost path from a given origin node to a single destination node (out of one or more possible destinations). In the recent years researchers have tried to generalize the A* algorithm to solve the time-dependent shortest path problem (Zhao, Ohshima and Nagamochi, accessed in April 2009).

Dijkstra's algorithm is an algorithm that finds a shortest path from a single origin to all other nodes in the graph with none-negative arc weights and generates a shortest path tree as solution (Wikipedia, accessed in May 2009). As mentioned above, recent work has tried to adapt different methods to introduce time-dependent A* and Dijkstra algorithms (Alexander and Swanepoel, accessed in May 2009).

Since A* and Dijkstra, a wide range of solutions has been developed to find optimum paths in different kinds of transportation networks, with the aid of algorithms or other mathematical concepts like fuzzy logics or time series. Very fast queries have been developed to find the shortest path between different nodes of a road network (Bast, Funke and Matijevic, accessed in April 2009), and even accelerated algorithms for shortest path queries have been developed (Maue, Sanders and Domagoj, accessed in March 2009). All of these algorithms minimize the travel-time or other cost types.

In contrast to the static problem, still there is a great lack in time-dependent shortest path research, and few authors have addressed time-dependent or dynamic routing problems (Cooke and Halsey 1966). Most of the available literature has only *partially* solved the routing problem, up until now, because in the suggested solutions only the distance, time or cost reduction or combination of them has been considered, and the dynamicity of the system was not considered at all. Dynamicity is an important issue in this field because the real world is dynamic.

However, defining time-dependent cost functions for network links or arcs is very complicated. Since we are not able to determine time-dependent cost functions in advance, we have to simplify the reality. More recently, researchers have presented solutions in dynamic networks and with time-dependent arc weights, which all are simplifications of real world (Zhao, Ohshima and Nagamochi, accessed in April 2009; Mahmassani, Zhou and Lu, 2005; Nannicini and Liberti, 2008).

The remainder of this thesis will focus on time-dependent or dynamic shortest path solutions, since the real problem is a dynamic one.

A key work in time-dependent shortest path problem was published by Jayakrishnan, Tsai, Adler and Oh (1997). They developed an event-based framework to find optimal routing scheme to improve the functionality of Advanced Traveller Information System. Network optimization, heuristics and driver-behaviour-based simulation used in the proposed framework to find the concrete optimal routing scheme.

According to Chabini (1997, 1998) there are different ways to group dynamic shortest path problems:

- Considering continuous time against discrete time
- Considering least time path against least length path
- Using deterministic data type against stochastic time-dependent data type

- Considering FIFO (First-In-First-Out) property in networks against non-FIFO ones. In the first network type, the vehicle that enters the network first, will be exiting it first. In the latter networks, the vehicle that is entered later can exit before others that are entered before it
- Allowing to wait at nodes against waiting forbidden at nodes before starting to go to the location of a new order
- Shortest path type: one-to-all against all-to-one

Soft time windows and hard time windows can also be added to the above-mentioned list as criteria for a dispatching problem. Soft time windows are costs of earliness and lateness but hard time windows forbid earliness and lateness completely (Chang, Wan and Ooi, 2008). In the common sense, soft time windows allow to have delays in the assigned schedule and apply predefined costs for the occurred delays while hard time windows do not allow having such delays by setting the delay costs to infinity.

The dispatching problem is likely to be an all-to-one shortest path problem because in a dynamic dispatching problem the calculation of the arrival time to the customer location and the associated cost for each vehicle-route is important. The accelerated algorithms proposed by Chabini and Ganugapati (2002; p.279), could be suitable for solving the dynamic shortest path problems. These algorithms use parallelization strategies to accelerate the *DOT* algorithm, which is a "*sequential algorithm that computes shortest paths in discrete time dynamic networks from all nodes and all departure times to one destination node*". This algorithm could be related to a dispatching problem via the assumption that the current locations of vehicles are 'on-the-fly' nodes and the unique destination is the new arrived order location.

Dean (1999) offered a set of algorithms which provide shortest path solutions in dynamic networks using continuous travel-time functions. The explained travel-time functions are piece-wise linear functions, which are able to model dynamic situations. The given approach can possibly be used to solve the dynamic routing problem, but defining such functions for traffic modelling is not an easy task.

The remainder of this section can be divided into three groups of works. Firstly, those that consider the FIFO property in their solutions for shortest path problem. Secondly, those that focus on the traffic information in their routing procedure, and thirdly approaches which consider vehicles as moving objects for routing applications. We will look at each in turn.

2.2.1. FIFO Property and Shortest Path

Several researchers have considered the FIFO property in their solutions. Forward and backward shortest path problems are explained by Daganzo (1998). The forward shortest path problem can be defined as the problem of finding the *least-cost or least-time path for a set of a provided origin node, a destination node, and a departure time from origin node*. The backward shortest path problem can be defined as the problem of finding the *least-cost or least-time path for a set of a provided origin node, a destination node, and a required arrival time to the destination node*. In the dispatching problem the backward shortest path can be considered more useful than the forward one. The paper

showed that in the networks with FIFO property, if an algorithm can solve the forward shortest path problem, then it can solve the backward shortest path problem on the same network and vice versa.

Dean (accessed in June 2009) explained two different kinds of dynamic shortest path problems with concentration on the FIFO networks. In the first type, the shortest path should be recomputed again and again because of network data is subject to frequent changes in an unpredictable fashion. The presented solution in this way can be expressed as the reoptimized solution of the series of related static solutions. In the second type, there is a rule to predict the changes in network travel-times. In the formulated problem the travel-times are assumed as the known functions of departure times from link start nodes. Regarding this definition, the problem is defined as a time–dependent but not a dynamic problem.

In addition, Dean (accessed in June 2009; p.1) studied FIFO networks and stated that "*under the FIFO assumption, time-dependent shortest path problems exhibit many nice structural properties that enable the development of efficient polynomial-time* (the required time to solve the problem can be defined as a polynomial) *solution algorithms*". Also he presented a unified framework and algorithms, which cover different types of shortest path problem with discrete and continues link travel-times.

Dean (2004) also investigated about different solutions of least-cost paths in time-dependent networks. The paper provides a research about waiting policies at nodes and mentions having possibility to wait at nodes may generate least-cost routes. In addition different ways to accelerate the procedure of finding shortest paths using dynamic programming (a method of solving complex problems in mathematics and computer science by breaking them down into simpler steps (http://en.wikipedia.org/wiki/Dynamic_programming)) are provided. Speed-up techniques have been reported by other research groups in the literature also; Veit Batz, Delling, Sanders and Vetter (2008); Bauer and Delling (accessed in April 2009); and Delling (2008).

Balev, Guinand and Pigne (accessed in May 2009) divided the dynamic shortest path problem into three groups. The first group of problems explains *time-dependent* shortest paths. They explain solutions for directed graphs with completely known properties. It has been shown that finding the least-cost path in FIFO networks is polynomial (Kaufmann and Smith, 1993). However, without having FIFO properties the problem changes to be NP-Hard (Orda and Rom, 1990). In the second group the *set of dynamics are under control*. A suitable solution type for this group can be explained as re-optimization techniques. Generally these techniques use the last solution as the base for finding the next-step solution. The last group is made up problems for which the *dynamics is not controlled*. The dynamic problem that is defined in Chapter Three and four falls in the last group of this division since real world road traffic conditions are fully dynamic and uncontrolled. For example, an accident can happen just because of the mistake of a driver.

2.2.2. Traffic Information and Routing

Other researchers have considered the traffic information in their routing solutions. Wahle, Annen, Schuster, Neubert and Schreckenberg (2001) formulated the problem in two steps. The solution presents optimum planned routes for network users. In the first step, real world road travel-times are

simulated using real world traffic information online. In the next step, the calculated travel-times are used in a guidance system to suggest an optimum route to the user based on his/her preferences. The proposed problem in this paper is a multi-criteria optimization problem and fuzzy set theory is used to solve it. This is the simplest approach to solve the problem in the most of the routing literature.

In contrast to this approach, Fu (2001) explored the problem of routing in the road networks and defined link weights as random variables with known mean and standard deviation. These kinds of solutions can be assessed based on real-time traffic data like data from loop detectors, FCDs or video surveillance systems. The provided solution can guide vehicle drivers turn-by-turn from origin to the intended destination via optimum paths. Real world travel-time information is included in the presented solution. The idea of guiding drivers turn-by-turn according to the latest information of real world traffic seems to be a good idea but it makes the problem very complicated, because the trace of all the planned routes and the current location of all vehicles should be taken into account.

Kalinowski and Wenk (2006) implemented two different solutions for finding shortest path in networks with time-dependent travel-times, central computation and local computation. In the central computation a powerful computer system is responsible for calculating least-cost paths based on the available road links travel-time information. In the local computation an on-board computer in vehicle is responsible to estimate the least-cost paths. The link travel-times are considered to be time-dependent as they can change frequently.

2.2.3. Moving Objects and Routing

Some researchers have tackled the problem of time-dependent shortest path in a more interesting method. As an example of this research type Chon, Agrawal and Abbadi (2003) defined a model for moving object. A moving object in this model is described as a physical entity with a defined size which has a computer and GPS and also is able to communicate. Special paths are defined to be used by these moving objects and a distributed system is provided for data management issues. This system guides moving objects along time-dependent least-cost paths. In the performance study, it is shown that the provided system yields shorter average trip time when there is a most congested route in the network. In contrast to the static approaches, which use a digital map with static information for distances of the paths and the speed limits along the ways (like a GIS), they used a method to obtain the cost in real-time and then apply a time-dependent shortest path algorithm.

2.3. Dynamic and Time-Dependent Dispatching Problem

Bertsimas and Ryzin (1991) presented and examined a stochastic and dynamic model to solve dynamic routing problem. The problem is formulated as a dynamic salesman problem, which a vehicle should serve the orders inside time windows. The orders are defined as dynamic orders and their arrival times and locations are considered to be stochastic. An objective function is defined to minimize the whole travelling time (waiting time plus service time plus travelling time). A year later in 1992 they proposed a solution (using more than one server) for a problem with m vehicles with the same characteristics and proved in a congested area the running time of the algorithm is reduced comparing to the case of having only a single server. Furthermore, they considered the case in which each vehicle can serve at most q customers before returning to a depot. They mentioned that the

stability condition in this case depends strongly on the geometry of the region; an issue which can be handled efficiently with the aid of GIS and has not been adequately addressed yet.

Different approaches have been adopted to solve the dispatching problem from different points of view. Relevant literature can be classified based on different problem definitions. The pick-up and delivery problem, dial-a-ride problem and milk runs are the major ones, for which their related works are discussed below. The two latter types can be considered as special cases of the *general pick-up and delivery* problem.

2.3.1. Dynamic, Time-Dependent Pick-up and Delivery Problem

The precise definition and formulation of this kind of dispatching problem will be given in the next chapter but as an introduction, this is the problem of assigning one or more vehicles to different customers to pick up some thing and deliver it in another place. The most related works are provided to review here. The related works in this section can be divided into three different categories. The works that considered time dependency (application of real world road networks traffic situation) but not dynamicity of the problem, the works that considered the dynamicity but not time dependency, and finally the works that considered both. The majority of the researches in this field belong to the second group.

2.3.1.1. Time-Dependent Pick-up and Delivery

Ichoua, Gendreau and Potvin (2003) presented a time-dependent model for time-dependent but not dynamic pick-up or delivery problem. The model, which is one of the former works in this field, considers the FIFO property of the road links. They evaluated their model in a static and a dynamic environment and noted that a time-dependent model gives better and improved solutions compared to static models.

2.3.1.2. Dynamic Pick-up and Delivery

Li, Mirchandani and Borenstein (2009; p.711) mentioned that "*the dynamic vehicle routing problem has gained increasing attention among logisticians since the late eighties*". In dynamic problems the main concentration is on *uncertain travel-times* and *online requests,* which are being placed after having initial(s) solutions (Bertsimas and Simchi-Levi, 1996; Gendreau and Potvin, 1998; Ghiani et al., 2003; Psaraftis, 1995). They provided real-time vehicle rerouting problem for a dynamic pick-up and delivery problem with time windows for the case of breakdown vehicles. In this research, a cancellation cost for each order is defined based on its significance. A dynamic programming-based algorithm heuristically solves the shortest path problems with resource constraints but time-dependent travel-time for network arcs is not taken into account.

Yang, Jaillet and Mahmassani (2004) presented a broad framework for a real-time pick-up and delivery problems. Different cost types like refusing orders, travel of vehicles to the pick-up locations and service lateness are included in the defined problem. Evaluating several policies based on various heuristics, they found the best policy is the one that takes into account some *future order*

distribution. The proposed solution excludes the traffic situation of real world road networks (and resulting time dependency) at all.

Mitrovic'-Minic' and Laporte (2004) described and compared four main waiting policies in the scheduling procedure for a dynamic (but not time-dependent) pick-up and delivery problem. According to the explained waiting strategies, vehicles are allowed to wait during their daily routes to dispatch letters and small parcels. The defined four waiting strategies are *drive-first*, *wait-first*, *dynamic waiting* and *advanced waiting* strategies. They reached to a conclusion that the *wait-first* waiting strategy generates shorter paths as solution but uses many vehicles comparing to other strategies to serve the same orders. The *advanced dynamic waiting* strategy is describes as the most efficient one since it minimizes both waiting time and the number of used vehicles.

Ghiani, Manni, Quaranta and Triki (2009) introduced an anticipatory method to solve the dynamic pick-up and delivery problem, which aims to serve same-day orders without considering vehicle capacities. The presented algorithm selects different regions and estimates alternative solutions for temporary orders. Different aspects of the real-time dispatching problem like vehicle assignments to orders for serving them, leading the assigned vehicles to the order locations, planning a daily path for each vehicle and reassigning available vehicles have been recognized in this research work. The proposed solution is not a time-dependent one.

Gendreau, Guertin, Potvin and Seguin (2006) defined a dynamic but not a time-dependent problem. The solution is presented for same-day pick-up and delivery problem of courier services. The goods to be picked-up and delivered are assumed be of small size like letters and small parcels. In the defined problem, daily regular paths should be defined for each vehicle to serve the customer orders, which are placed. In this research neighbourhood search heuristics are used to cope with the defined dynamic dispatching problem in an effective and efficient way. These heuristics optimize the assigned routes in each new order placement. It is shown that such heuristics can provide better solutions comparing to insertion based algorithms.

Branchini, Armentano and Lokketangen (2009) defined an objective function as the difference between whole company incomes and all cost types including travel costs and delay costs. This function is to be optimized during the procedure of vehicle assignment and routing to serve new customer orders in a dynamic environment. As a result a set of daily routes are developed for each vehicle using constructive heuristics method. In the proposed solution the service area is divided to different zones and vehicles are scattered within it. Different policies are defined to distribute vehicles and assigning them. According to this approach a vehicle can wait to receive new orders in the regions that are possibly new orders will be placed in there. This paper also does not pay attention to apply time-dependent travel-times.

2.3.1.3. Dynamic and Time-Dependent Pick-up and Delivery

In the same year with Ichoua and his co-workers Fleischmann and Sandvoss (2003) and Fleischmann, Gnutzmann and Sandvoss (2004) worked on a dynamic dispatching system which assigns and routes vehicles to answer the random customer requests. In the formulated problem dynamic orders for pick-ups and deliveries should be satisfied within the required time windows. In

the presented solution the dispatching center receives real-time traffic information from traffic management center and is able to communicate with drivers. Three different routing strategies with different planning horizons (the available time to serve) are introduced. The generated model calculates the shortest path dynamically and has two main objectives: minimization of *delays* and minimization of *costs*. The insertion based algorithms, which 'insert' new arrived orders to the queue of previously assigned orders, are used to construct routes. Clearly, the research considers dynamicity and time dependency aspects, but the proposed solution does not work if online traffic information is not available. Storing time-dependent travel-times in spatial databases and updating them by real world traffic data will cope with this risk.

Haghani and Jung (2005) formulated the dynamic routing problem within a dynamic time-dependent pick-up and delivery problem. They considered soft time windows for orders and different capacities for different vehicles. The orders are defined as real-time requests, which should be served using road links with real-time traffic information. They developed a genetic algorithm as a solution for the formulated problem and demonstrated that dynamic routing can result in better solutions compared to static routing when there are uncertain situations in real-time road links travel-times. Travel-times are introduced using continuous travel-time functions. Also, the research proves that waiting policies should not be applied for networks with FIFO properties. This problem also works with the assumption of availability of real-time travel-times as continuous functions which are not stored in spatial databases, as these might not be accessible always.

Chang, Wan and Ooi (2008) formulated the well-known salesman problem as a pick-up / delivery problem with hard time windows. In the formulated problem collecting an order from a pick-up location and delivering it to another location is not the issue. In each order one pick-up or one delivery is considered only, i.e. the problem is formulated for collecting goods from different pick-up locations and delivering them to a single depot or vice versa. The travel-times of vehicles assumed to be random and are subject to changes during a planning period (generally one day for a same-day pick-up and delivery problems). In the proposed solution, heuristic algorithms are provided. This research considers the non-real and random travel-times to apply, which can provide a non-realistic solution.

2.3.2. Dial-a-Ride Problem

The second type of the general dispatching problem is a *dial-a-ride* problem. In a dial-a-ride problem a dispatcher should assign one or more vehicles to serve customers. This can implicitly be considered as a pick-up or delivery problem. Taxi or bus dispatching are well known examples. Related works to this problem can be summarized as follows:

Firantas and Jusevičius (2006) explored automation (no need to have human input in dispatching or routing procedure) potential in the real-time dynamic dispatching and routing solutions. They provided a conceptual design for automation of taxi dispatch and developed a routing process for real-time requests using location-based services and context-aware technologies.

Coslovich, Pesenti and Ukovich (2006) studied a dynamic dial-a-ride problem, which considers time windows without identifying them to be soft or hard. As a solution to serve new unexpected requests,

they developed an insertion algorithm, which works in two phases; an off-line phase (the objective of this phase is the generation of suitable buffer for the last planned path) and an on-line phase (the objective of this phase is the assignment of new orders to the former path in a way that can minimize the inconvenience of the previous orders using insertion algorithms).

Potvin, Xu and Benyahia (2006) investigated the dial-a-ride problem to collect mails from different pick-up locations and transfer them to a single depot to be dispatched from there to their final destinations. Their main contribution is the introduction of a 'tolerance' concept to consider the unforeseen delays.

2.3.3. Milk Runs

The last type of dispatching problems is those that are referred as *milk runs*. In this type of problem, vehicles can be assigned to serve both customers and suppliers in the same time to take the advantage of vehicle capacity as much as possible (Du, Wang and Lu, 2007). These researchers explored different policies to set the parameters of a dynamic and time-dependent dispatching problem for a special kind of transportation network (or milk runs). They developed seven modules to be applied to the real-time system; among them an initial vehicle-dispatching module and an inter-route improvement module can be mentioned.

2.4. Dynamic Assignment and Optimization Problem

The main goal of this research is to find the optimum dispatching strategy and optimum paths to route the assigned vehicles. To further investigate vehicle assignment and route optimization problems, researches that are related to these two topics have been reviewed. Most of these have addressed the dynamic problem, to which each of them applies a very different solution. These are as follows:

- The first research belongs to Spivey and Powell (2004), which introduced a new accurate model for dynamic assignment problem that extracts information about the vehicle situations and dynamic orders before any event or order placement. Their solution is based on the presentation of adoptive learning algorithms to solve a queue of scheduling problems.

- Ahner, Buss and Ruck (2006; p.1349) studied the problem of schedule assignment for unmanned aerial vehicles. They provided an implementation of dynamic programming techniques for estimating none-continues phenomenon and simulating it using optimization methods. The solution is presented in a practical way and can be implemented in the real world in large simulation projects. Regarding to this paper" *optimization is very effective at identifying the best decision for static problems, but is weaker in identifying the best decision in dynamic systems. Simulation is very effective in modelling and capturing dynamic effects, but is weak in optimizing from alternatives*".

- Finally, Xiao, Huang and Lou (2007) introduced a framework to cope with the complex dynamic optimization problems. They developed a mathematical solution which estimates the

probability distribution. This probability distribution is based on the population and is used to find the final solution for a dynamic nonlinear problem iteratively.

2.5. Traffic

The final part of this literature review considers related works in traffic modelling and travel-time estimation. Driving a vehicle is an essential part of a dispatching problem, but traffic congestion play an important part in trip time and cost. Consequently, this is one of the most important components that should be solved in a dispatching problem: traffic modelling or finding functions for traffic estimation. Previous research in this field is discussed below. To estimate the flow in the road networks, some researchers modelled the traffic congestions or provided a time-dependent function to calculate them, while others developed models to examine the travel-time. In this section, we review both of these approaches.

2.5.1. Traffic Modelling and Traffic Estimation

One of the primitive researches in this field has been done by Miller, Wu and Hung (1999). The research provides a *GIS-based* prototype as a decision support system for dynamic routing and dynamic traffic modelling. The model is a time-dependent model, which predicts roads traffic flow in the predefined time intervals based on dynamic network assignment methods. The predicted traffic flows are used to route the vehicles based on their departure times. The system leads vehicles to arrive to the intended destinations on-time. The role of *GIS* in the provided prototype is significant due to its capabilities in data management techniques and its powerful visualisation tools.

More recently Yoon, Noble and Liu (2007) proposed a simple and effective system, which determines the traffic situation of roads with the aid of GPS and low-bandwidth cellular updates that are receiving from en-route vehicles. A unique traffic pattern and any odd traffic condition can be recognized for each road link using the proposed system. They also demonstrated that having none-varying road conditions, traffic situation shows very stable patterns on the road links. The proposed method can be adapted and used in the intended solution for dynamic routing problem because of two main reasons. First, it finds patterns for each road link and second, it uses historical data, which we are likely to use.

2.5.2. Travel-Time Estimation

Road link travel-times are dependent on several factors like accidents, traffic situations and weather conditions (Haghani and Jung, 2005). Without considering time-varying travel-times in real-world road networks, the assigned paths for vehicles that are responsible for pick-ups and deliveries can lead them into possibly blocked road links. Because of having variations in real world road network travel-times, vehicles arrive to the order locations latter than the planned times and customers have to wait to be served. This situation results in having unsatisfied customers because of violated time windows and extra costs for the transportation company due to wasted times in congested road links and service lateness. Accordingly, "*real-time traffic information for all major roadways in an urban area is one of the most important pieces of information necessary to produce a dynamic route guidance system*" (Nual, Wlodyka and Phiu-Nual, accessed in May 2009; p.1). This kind of

information in the time-dependent setting for world road networks can be obtained using spatial data sets like Tele Atlas, which are provided to be used with TomTom navigation system.

Most of the related works in this section used the concept of FCD (Floating Car Data). As a good example, we can mention dynamic travel-time creation for real world road networks by Pfoser, Brakatsoulas, Brosch, Umlauft, Tryfona and Tsironis (2008). *FCD* refers to *"using data generated by one vehicle as a sample to assess to overall traffic conditions"* (Brakatsoulas, Pfoser and Tryfona, 2005; p.1). In addition, *"with vehicle tracking data becoming an important data resource for a range of applications related to traffic assessment and prediction, fast and accurate map-matching algorithms become a necessary means to ultimately utilize this data"* (Wenk, Salas and Pfoser, 2006; p.1). In-car navigation systems have to use these algorithms to match the assigned routes to the available map via spatial data set like Tele Atlas to visualize the planned route to the drivers.

Various research works proposed fast map-matching algorithms (Brakatsoulas, Pfoser, Salas and Wenk, 2005), which are beyond the scope of this research. A range of research reports on spatio-temporal databases, moving objects and their trajectories and also different methods to index and query the trajectories (Erwig, Guting, Schneider and Vazirgiannis, 1999; Pfoser, Jensen and Theodoridis, 2000; Vazirgiannis and Wolfson, 2001; Pfoser and Jensen, 2003; Brakatsoulas, Pfoser and Tryfona, 2005). Different practical projects have been done using FCDs like the one that has been done for Athens, Greece (Kelpin, Sohr, Umlauft, Brosch, Schimon and Pfoser, 2007 and Brakatsoulas, Pfoser and Wenk, 2007).

The related works of this section can be split up into two categories: those that seek to generate output maps, and those aiming to develop models.

2.5.2.1. Dynamic Travel-Time Maps and FCDs

Pfoser, Tryfona and Voisard (2006; p.1) created a" *dynamic weight database"* or a "*dynamic travel-time map"* (DTTM). The introduced spatio-temporal data portal is used to calculate the dynamic travel-times as a reflection of road link speeds in the given time. FCD is used to collect dynamic and time-dependent travel-times and different algorithms and data mining methods are provided to extract the temporal patterns of road links traffic situation.

Later on Brakatsoulas, Pfoser and Tryfona (2005) introduced a FCD and GPS-based strategy, which extracts precise and updated traffic information. In this method with using map-matching techniques, traffic information is used to capture the travel-times of road links. The DTTM is explained as an efficient data storage tool for travel-times.

2.5.2.2. Travel-Time Prediction Modelling

Lee, Tseng and Tsai (2009) explained the complexity of real-time travel-time forecasting in urban area road network and mentioned several reasons for it. The provided approach presents a model to forecast the real world road travel-times. The model is an interesting knowledge based model, which takes the advantage of real-time and historical time-dependent data to provide and predict traffic patterns. The real-time data is collected using location based services and data mining techniques are

used to extract the rules. Then GIS is used to convert the travel-time data to travel-time information. The proposed forecasting model is proven to be cost-effective for urban roads. The suggested model is likely to help in the current work to estimate the road links travel-time and then find the best route for the assigned vehicle.

2.6. Conclusion

Referring to the reviewed literature for dynamic dispatching problem the solution provided in Fleischmann, Sandvoss and Gnutzmann (2004) for the formulation of dynamic dispatching and the solution provided in Kalinowski and Wenk (2006) for dynamic routing are the most appropriate and the most promising approaches from mathematical point of view. While we will extract the problem components from the first research, we will extend the proposed shortest path algorithms in the second one to be used as a core function of dynamic spatial routing in the following chapter. In the remainder of this research *travel-times* will be considered as the travel costs for routing issues.

3. Problem formulation

3.1. Chapter Structure

The dynamic pick-up and delivery problem is described and formulated in this chapter as a specific type of the dynamic dispatching problem on a road network with updated traffic information and time-dependent travel-times for each road link. Here, the dispatching problem is split into three major components, each of which is addressed by a specific solution *module*. Each module executes a set of predefined functions in four cases: receiving a new order, completion of an order, order cancelation, and a major change in travel-times for routed vehicles. The functionality of each module is described in detail.

Generally, a variety of components for a dynamic dispatching problem can be considered. Here we identify four major dynamic variables. While other variables exist, they arguably have minor importance in the problem:

1. Travel-times in road networks;
2. Orders;
3. Vehicles availability location and time;
4. The road networks feasibility, which affects the route plan;

Among real world dynamic situations, the main focus will be on the *dynamic traffic situations which affect the network-link travel-times*. They are important in dynamic vehicle routing, and must be dealt with to have an optimal routing procedure in a dynamic dispatching problem. Furthermore, the close relationship between dynamic dispatching and dynamic routing and the dependency of an efficient dynamic dispatching plan on an efficient dynamic routing plan is explained in detail.

3.2. Problem Description

The dynamic pick-up and delivery problem is the problem of dispatching a known number of available vehicles of a transportation logistics company to pick up the dynamic orders (orders can arrive at any time) from specific locations in specific time windows and then deliver them to the delivery locations. The vehicles use real world road networks, with time-dependent and dynamic travel-time information. Finding solutions to this problem without no doubt will be interesting for transportation logistics companies, which are dealing with dynamic situations of orders and road links traffics in their daily dispatching and routing process.

The dispatching problem defined here consists of:

- A single depot with a known location
- Known number of vehicles each with a fixed capacity
- Dynamically arriving customer orders, which are requesting pick-ups from specific pick-up locations inside specific time windows and deliveries to assigned delivery locations

In the defined problem, we assume all vehicles are available in the beginning of a working day in the defined depot and they should start their daily routes from depot and return to it back in the end of the day. Orders that are arriving dynamically during the day should be served; in other words, there is no possibility to reject the orders. The objective is to assign vehicles to serve dynamic orders and route them along the road links with time-dependent travel-times in a way that minimizes the total costs of the transportation logistics company. The term cost is defined in the following section.

According to Fleischmann, Sandvoss and Gnutzmann (2004) two objectives can be defined for any dynamic dispatching problem: The first is the minimization of delays, which always happen due to the unpredictable events in real world road links and cause to delay costs for transportation company. The second is the minimization of moving and waiting costs, which are assumed to be relevant to the total time of all travels that are the results of the time-dependency. Obviously, an efficient dynamic routing plan has a major effect on both these objectives. Minimizing delays and moving costs seems to be impossible without leading the vehicles via the most feasible and optimal road links. For example, routing a vehicle through a road link in a time of day, which is the time of the daily congestion on that link, will cause the lateness of that vehicle to the intended destination and result to a delay cost. Alternatively, routing a vehicle through a set of links, which are sub-optimal, will increase the movement costs like fuel usage. Having a solution for dynamic optimal routing to cope with these problems in a dispatching system can help dispatchers to provide better solutions for the transportation company.

"Dynamic routing inside the dynamic dispatching takes place in a sequence of planning runs that update the current plan over a defined period starting from the current time" (Fleischmann, Sandvoss and Gnutzmann, 2004; p.421). In each iteration a solution should be provided for the closest assignment. It is possible to have a schedule for future actions but communicating them with the vehicle drivers is not permitted to support the highest adaptability. Therefore, the schedule or the route to serve next customers for each vehicle based on the current available information can be planned but cannot be communicated with the drivers up to the next availability time.

The following section introduces the notation used to formulate the dispatching problem dealt with in the reminder of this thesis.

3.3. Notation

The notation adopted here for formulation of the problem follows Fleischmann, Sandvoss and Gnutzmann (2004) for reasons of comparability.

I= Set of all available vehicles. All vehicles are considered to have the same characteristics, and the capacity of vehicles is not considered here (simplifying assumption);
J= Set of all un-served orders;
J^{NEW}= Set of all new orders when a new order is placed;
tc= Current time;
lci= Current location of vehicle i;

The vehicle situation $i \in I$ can be determined as follow:

TAi= The next time, which vehicle is ready to assign an order (*TAi= tc* for $i \in I$);
LAi= The next location, which vehicle is ready to assign an order (*LAi*= Depot location in the beginning of the working day);

An order $j \in J$ can be characterized with the following properties:

PLj= Pick-up location;
DLj= Delivery location;
EPTj= Earliest pick-up time, if a vehicle arrives before this time to a pick-up location, it should wait until this time;
LPTj= Latest pick-up time;

Travel-times presented below are needed during the assignment procedure:

TULij= Travel-time of the empty move from *LAi* to *PLj*, for start time *TAi* ($i \in I, j \in J$);
TLj (t) =Travel-time of the loaded move from *PLj* to *DLj*, for a start time *t*;

3.4. Constraints of the Dispatching Problem

We can define a set of constraints for this problem as follows:

- A daily path for all vehicles begins from the depot at *a* AM and finishes at the depot at *b* PM (we consider a single depot problem).
- The vehicles should be in the depot by *b* PM.
- An extra order in one dial and with different pick-up or delivery location is considered as another order.
- For simplicity we assume that the number of vehicles is fixed i.e. the company cannot rent other vehicles from other companies even when it is beneficial.
- The planned routes for vehicles cannot be changed when they are empty and headed to the pick-up locations or when they are loaded and are en-route to the delivery locations.
- As a result of previous constraint vehicles can receive new information just in two cases: when they have completed the service and when they are waiting.
- Each order should be serviced only by one vehicle.
- The transportation company is aware of vehicle locations any moment.
- Communication between the dispatching center and the drivers about new order locations takes place after any job completion. "*The drivers are not aware of the global 'picture' represented by their current planned route*" (Gendreau, Guertin, Potvin and Seguin, 2006; p.159) (which may be modified frequently by the optimization procedure).

A vehicle daily path can be defined as travels along the road links to the pick-up locations and then travel from pick-up locations to the relevant delivery locations. This is true when we do not want to assign waiting locations, which can be not the same as the last delivery locations. "*Assigning waiting locations in FIFO networks is never beneficial and it will never reduce the arrival time at ones eventual destination*" (Dean, accessed in June 2009; p.5). In this case, we can take into account only the pick-up time window, which is not precise in the case of having time-dependent travel-times for road links (Fleischmann, Sandvoss and Gnutzmann, 2004) and non-FIFO networks. We do this because the problem changes from dynamic dispatching problem to a dynamic routing problem after having a load in the pick-up location.

3.5. Cost Criteria

Main cost sources and the cost criteria defined here can be used to make decisions to reach to the main objectives of a dynamic dispatching problem, which are the minimization of whole costs of the transportation logistics company. The following cost sources are adopted from Fleischmann, Sandvoss and Gnutzmann (2004):

- Costs of empty move= *c_tul TULij*;
- Costs of loaded move= *c_tl TLj(ts)*;
- Costs for waiting= *c_Wait(max(EPTj-ta,0))2*;
- Costs for delay= *c_Late(max(ta-LPTj,0))2*;
- Costs of fixed daily demands= *c_FL;*

Where:

ta= TAi+TULij is the time of arriving to *PLj*;
ts= max (ta, EPTj)+ SPTj is the beginning time from *PLj*;

c_tul and *c_tl* should be considered as coefficients of travel-times, which are estimated in the travel-time calculation module (TTCM), we will come back to the functionality of this module later on this chapter . *c_Wait* and *c_Late* are the penalty costs associated with the vehicle-waiting costs and service lateness costs respectively. *c_Late* is included since we considered soft time windows. According to Haghani and Jung (2005; p.2962) "*soft time windows are more realistic and more flexible to be used in the formulation of the problem comparing to the hard time windows*". There is another cost type which is not dependent on the orders and vehicles daily route. These costs are daily predefined costs and are based on the vehicle maintenance and fixing costs, insurance, deriver's salary, and etc. and can be provided with *c_FL* (Branchini, Armentano and Lokketangen, 2009).

3.6. Objective Function

According to the previous section, the objective is the minimization of the following cost function in a way that the proposed solution meets all defined constraints:

$$F= \sum i \sum j\ c\_tul\ TULij + \sum j\ c\_tl\ TLj\ (ts) + \sum j\ c\_Wait\ (max\ (EPTj-ta,\ 0))\ 2 + \sum j\ c\_Late\ (max\ (ta-LPTj,\ 0))\ 2 + c\_FL$$

To solve the minimization problem of this objective function, Operational research (OR), a branch of applied mathematics, could be used.

The Dispatching module (DM) that is responsible to propose a solution for this optimization problem will be explained in the next section of this chapter. This module asks for *TULij* and *TLj* for each *j* and each vehicle *i* from TTCM and minimizes the objective function based on these two estimated parameters.

For none-negative travel-times, the main objective of the proposed function is the minimization of "*a weighted summation of travel-times for empty moves and loaded moves, sum of lateness at customer locations ,waiting times*" (Potvin, Xu and Benyahia, 2006; p.1130) and fixed daily costs. The travel-times of empty moves *TULij* and the travel-times of loaded moves *TLj* are in fact dynamic due to the varying traffic situations in road networks. They require the calculation of dynamic travel-times and dynamic least-cost paths as the main parts of the dynamic dispatching problem to be solved.

3.7. Method Adopted

The adapted method tackles the identified problem above by defining a series of modules that address specific aspects of it. It is possible to define three main components of this solution: a management module (MM), a dispatching module (DM) and a travel-time calculation module (TTCM). The relation between these three modules and all information and events, which come from real world or environment are illustrated in Fig.3.1.

This work is based on the method adopted in Fleischmann, Sandvoss and Gnutzmann (2004). The existing difference is related to the functionalities of the two modules DM and TTCM. Despite of the mentioned paper that have focused on the functionality of the DM to provide mathematical solutions for vehicle assignments in a dynamic dispatching problem, the focus here is on the TTCM to provide solutions for dynamic routing problem as a core problem of dynamic dispatching problem.

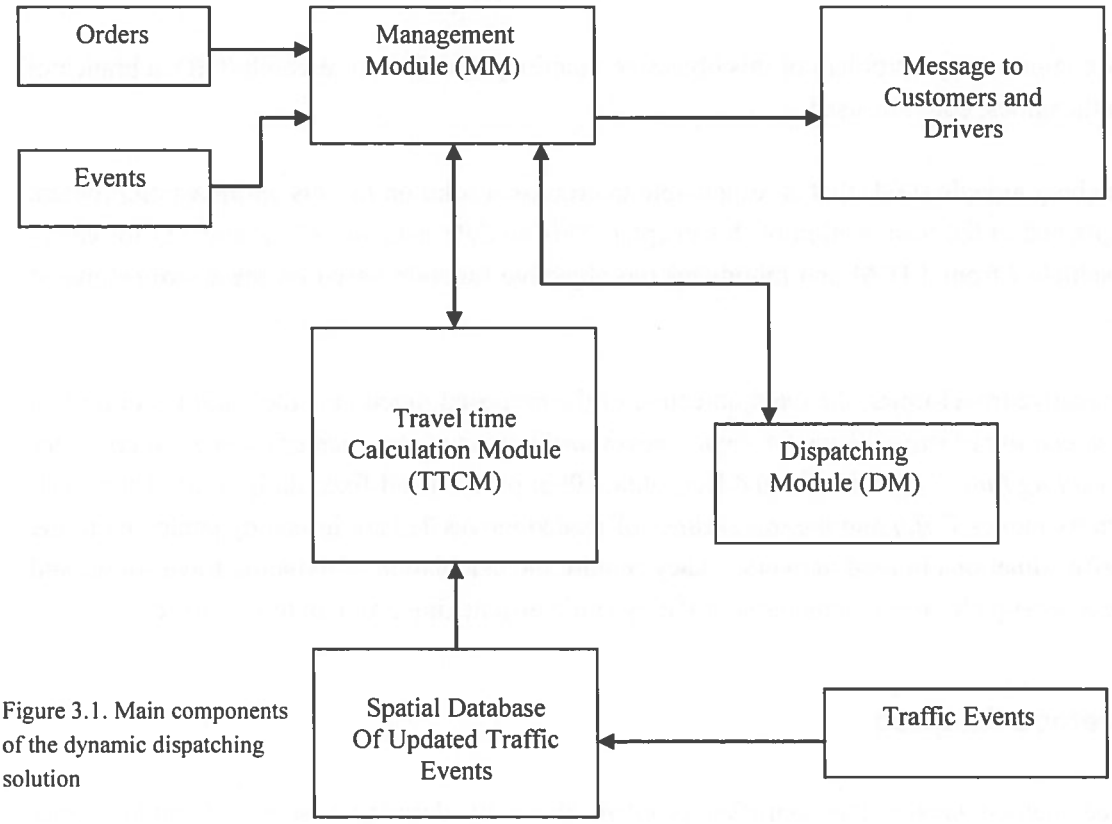

Figure 3.1. Main components of the dynamic dispatching solution

Briefly, the MM receives orders of customers and events like order completions, delays, arrival of vehicles to pick-up locations and order cancelation, weather conditions, road constructions and vehicle breakdowns, manages them, and informs the drivers and customers about the resulted schedule from the DM. This module also receives calculated travel-times and shortest paths from the TTCM and transfers them to the DM. In general, this module is responsible to communicate with other two modules to transfer information to them and receive results.

The TTCM receives the real-time information in the form of updates from spatial database, calculates travel-times and assigns paths based on the most up-to-date travel-time information. This module is also responsible for recalculating the travel-times and generating new shortest paths based on new traffic 'updates'.

DM plans schedules to serve orders based on the calculated travel-times and shortest paths by TTCM and the information transferred about orders and vehicles from MM.

In the next section the functionalities of MM and DM will be explained in detail. The functionality of TTCM and routing procedure will be explained precisely in the next chapter since the focus of this work is on the dynamic routing part of the dynamic dispatching problem.

3.8. Method Functionality

In the cases below the system has to make a new decision about the schedule; receiving a new order, completion of an order, order cancelation, and major change in travel-times for routed vehicles. The functionality of each module is described in each case in this section.

3.8.1. The Management Module (MM)

In the proposed method, new information about orders (*PLj*, *DLj*, *EPTj* and *LPTj*) and mentioned events like order-completion with the vehicle availability location and time come to the MM and stored there to be used in the two other modules. This module also keeps track of the last schedule to determine the current condition of whole system. When a new order comes, the MM transfers it to the DM. After having assignment in the DM the new schedule should be transferred to the MM to be stored there. To apply assignment algorithms and to minimize the defined objective function, the DM needs the travel-time information for all vehicles from the next availability location to the location of all open orders. MM asks these times from the TTCM and transfers them to the DM to be used in the assignment of vehicles to serve orders in the new schedule. In general, the tasks that this module is responsible to fulfil them in the arrival of new information can be described as follow.

3.8.1.1. Arrival of a New Order

- Receiving and storing new order $j \in J^{NEW}$ with all relevant information (*PLj*, *DLj*, *EPTj* and *LPTj*).
- Transferring the new order $j \in J^{NEW}$ with all relevant information to the DM and ask for a new schedule.
- Transferring *PLj*, *DLj*, *TAi* and *LAi* for all $i \in I$ and $j \in J$ to the TTCM and ask the travel-times from vehicle *LAi* to *PLj* and then to *DLj* using new *TAi*.
- Getting the results for travel-times from the TTCM.
- Transferring the result of travel-times to the DM.
- Getting the new schedule from the DM and storing it as a current schedule.
- Message to drivers based on the new schedule about their routes and their next destination.
- Keep the trace of all planned routes to be used if any rerouting procedure is needed before the vehicle arrives to the delivery location.

3.8.1.2. Completion of an Order

- Receiving and storing the availability location and time of vehicle $i \in I$, its current location *lci* and current time *tc*.
- Checking the current schedule for any order that is assigned previously and should be served next by vehicle *i*.
- If there is any assigned order, message to the driver of vehicle about new route.
- If there is no assigned order, message to the driver of vehicle to wait.

3.8.1.3. Arrival of New Travel-Time Information

- Transferring the *PLj*, *DLj*, *TAi* and *LAi* of all $i \in I$ and $j \in J$ to the TTCM.
- Asking for new travel-times and new optimal paths for en-route vehicles from the TTCM based on the updated traffic information for road links.
- Transferring the new travel-time information to the DM and asking the new schedule.
- Storing the new schedule as the current schedule.
- Sending messages to the drivers if any route is changed with compare to the previous schedule.

3.8.1.4. Order Cancelation

- Receiving and storing the new canceled order *j*.
- Monitoring the current schedule for any en-route vehicle to serve this order.
- If there is any en-route vehicle to serve this order, sending message to driver of that vehicle to stop in the first feasible location.
- If there is no en-route vehicle to serve this order, deleting the order *j* from *J*.
- Asking the DM to plan a new schedule based on the updated information.
- Storing the new schedule as the current schedule.

3.8.2. Functionality of the Dispatching Module (DM)

DM processes and then assigns the coming dynamic orders to the most feasible vehicles or precisely to the most feasible routes to be served. These assignments can be done in this module using optimization rules and assignment algorithms like LAPJV by Jonker and Volgenant (1987), which are proved to have an advantage upon other algorithms for dynamic orders (Fleischmann, Sandvoss and Gnutzmann, 2004) to determine routs and minimize the objective function. To receive the information of new orders, current schedule, and the travel-times calculated by the TTCM, this module communicates with the MM. Using time-dependent travel-times the DM plans the most optimal schedule and reports the new schedule to the MM to be stored there as the current schedule.

In general, the tasks that the DM is responsible for in the case of new information can be described as follows.

3.8.2.1. Arrival of a New Order

- Receiving the information about the new order *j* (*PLj*, *DLj*, *EPTj* and *LPTj*) from the MM.
- Receiving the updated travel-time information from the MM.
- Receiving the current schedule from the MM.
- Assigning a new schedule and a new route for each vehicle based on the arrived information from the MM.
- Transferring the resulted schedule to the MM to be stored there as the current schedule.

The procedure of assignment in this module is beyond the scope of this research.

3.8.2.2. Completion of an Order

There is nothing to be done in this module in an order completion.

3.8.2.3. Arrival of New Travel-Time Information

- Receiving new travel-time information from the MM.
- Receiving current schedule from the MM. The start point of the en-route vehicles to change their planned path is the nodes where the vehicles are currently located or the ones that they will arrive next.
- Assigning a new schedule and a new route for each vehicle based on the arrived information from the MM.
- Transferring the resulted schedule to the MM to be stored there as the current schedule.

3.8.2.4. Order Cancelation

- Receiving the information of the canceled order j and its current situation (checking if there is any en-route vehicle i, and if yes, retrieving lci).
- Receiving new travel-time information from the MM.
- Receiving current schedule from the MM.
- Assigning a new schedule and a new route for each vehicle based on the arrived information from the MM.
- Transferring the resulted schedule to the MM to be stored there as the current schedule.

3.8.3. Functionality of the Travel-Time Calculation Module (TTCM)

Based on the given information by the MM about the orders pick-up and delivery locations and current planned schedule, and updated travel-time information stored in this module for road links, the TTCM is responsible to calculate the required travel-times for vehicle routes and find the shortest paths between all pick-up locations and all delivery locations. The results of these calculations should be transferred to the MM to be communicated them with the DM.

The major part of the dynamicity of the dynamic dispatching problem belongs to this module since the real world road network traffic conditions are highly dynamic and even inside the very minor intervals significant changes can happen. Briefly, the responsibility of the TTCM in the arrival of new information can be described as follows.

3.8.3.1. Arrival of a New Order

- Receiving PLj, DLj, TAi and LAi for all $i \in I$ and all $j \in J$ from the MM.
- Computing the travel-times and plan paths with least travel-times for PLj, DLj and LAi for all $i \in I$ and all $j \in J$, Tow-by-Tow based on the last road link travel-times stored in the module and travel start-times.

- Transferring the results to the MM.

3.8.3.2. Completion of an Order

There is nothing to be done in this module in an order completion.

3.8.3.3. Arrival of New Travel-Time Information

- Storing the new travel-time information as the current travel-time information to be used in routing procedure after this moment.
- Checking a predefined threshold to find if it has a significant effect on routes.
- If the new information is effective, extracting the affected links of planned paths.
- Recalculating the related travel-times and paths starting from the source node of the affected link.
- Transferring them into the MM.

3.8.3.4. Order Cancelation

- Receiving the current schedule and the *PLk* and *DLk* of canceled order *k* or *lci* if there is any vehicle en-route to the *PLk*, from the MM.
- Recalculating the travel-times and least-time paths based on the current schedule and new information
- Transferring the new results to the MM.

TTCM has a major effect in the proposed method because its direct responsibility is to route vehicles dynamically in real world road networks to lead them to the intended destinations via the least-time paths. The reminder of this chapter and also the next chapter will focus in dynamic routing problem, its role in the more bigger problem: dispatching, and solutions to cope with this problem.

3.9. Dynamic Routing

As discussed in the previous chapters, routing has a fundamental role in all dispatching problems. A dynamic pick-up and delivery problem can be considered as a dynamic assignment of daily optimum routes for each vehicle to serve the orders in pick-up and delivery locations considering the placed time windows (Greenwood, Dannegger and Dorer, accessed in May 2009). The assigned paths can be changed during the travel time of the vehicle to its destination due to the variations of real world road link travel-times. These changes can be introduced differently for different days of a week and different times of a day. A key challenge in deriving a solution to this problem lies in assigning a known number of vehicles located in known locations in known times to serve the placed orders in a way that minimizes the whole costs of the transportation logistics company and maximizes the total number of pick-ups and deliveries and customer's satisfaction level.

In the proposed method, the TTCM is responsible to find the least-time paths and also is responsible to route the vehicles along these paths. Due to the real world dynamic conditions, using dynamic routing in dispatching models is more realistic than using the static ones. The related research works that have tackled the static routing problem are described in Chapter Two, where it is shown that there is still a significant lack of dynamic routing algorithms and software. As it is explained before, spatial software like ArcGIS and spatial tools like, pgRrouting have not had efficient solutions for dynamic routing problem, while a routing problem is a spatial problem.

Different scenarios can be defined to reflect the dynamicity in routing procedure, each of which considering different aspects of dynamic real world. Here we define a dynamicity scenario and then propose a solution in the next chapter.

In the context of road network, routing is a procedure highly dependent on dynamic road traffic situations, as there are effective factors in determining link travel-times. Therefore, as a first dynamic variable in the routing procedure, dynamic traffic conditions on real world road links must be defined. Dynamicity in traffic conditions might happen because of accidents, weather conditions, constructions or some other reasons like being in the end of a working day.

In a time-dependent routing procedure an initial path from the origin node to the destination node can be assigned based on the latest available travel-time information of the road links, but this path must be changed if significant changes happen in the traffic conditions of one or more assigned path links latter. It is clear that the changes in the road links already passed by the vehicle cannot affect its total travel-time. The planned routes are also dynamic since traffic conditions are dynamic in the real world.

The road network can be considered as a dynamic network because in the situation of having significant changes in the travel-times of planned routes, for rerouting plan there is no need to consider the whole road network once again to lead the vehicle to the destination. The road links that have been passed can be eliminated to reduce the size of search space in the network to find least-time paths. Thinking this way the road network can be considered as a dynamic variable in a routing procedure also.

The next chapter will explain the dynamic routing problem in detail and will propose solutions to solve the problem of dynamicity cases, using an open-source spatial database.

4. Dynamic Routing and Case Study

4.1. Chapter Structure

In order to solve the core problem of any dynamic dispatching problem, dynamic routing and its differences with static routing will be considered in this chapter. We start with static routing and try to fit the related solutions to solve the time-dependent (dynamic) routing problem to be used in the dispatching module introduce in the previous chapter.

This chapter implements extended dynamic functionality for existing static routing algorithms. The two well-known algorithms, Dijkstra and A* will be investigated based on their appropriateness to have a solution for the defined problem and the compatibility with the Tele Atlas data set. The network link weights are defined as time-dependent (dynamic) travel-times that are subject to possibly many updates. Furthermore, the dynamicity scenario that is defined for dynamic routing of vehicles in the previous chapter is explained further and solutions to solve it is proposed, using the pgRouting extension of postgreSQL/postGIS, and Tele Atlas network data and speed profiles for the road networks of the Netherlands as a case study respectively. The results for both static and dynamic routing will be provided.

4.1.1. Static Routing

We face with the routing problem to deal with dynamic situations in various fields of science to find the least-cost path between one or more origin(s) and one or more destination(s). For example in transportation, an efficient routing plan is needed to guide vehicles through least-cost paths or to lead the walking people to use the fastest routes to their destinations. In computer science, a good routing plan is needed to transfer information packs between different nodes of multimedia network quickly; these nodes could be different internet users, who have performed a search request.

As discussed in Chapter One and Two the static routing or shortest path problem has received attention in different branches of science since the late 20th Century. As a result, different algorithmic solutions (like Dijkstra, A*, B*, Shooting Star, Floyd-Warshall) and different software (like Arc Logistics by ESRI and pgRouting by Orkney) are now available to solve static routing problems in transportation applications.

Since the currently available software is unable to handle the time-dependent routing procedure for real world road networks because of its complex aspects, the main objective of this chapter is to provide an algorithm and use it in postgreSQL/postGIS, which handles the two previously defined dynamicity situations. The resulting solution could be used in the TTCM to estimate dynamic travel-times between nodes and are responsible to route vehicles within least- cost [1] paths.

[1] The term cost is considered as travel-times between nodes in routing part of this research. Other cost sources are not considered for simplicity in routing process

Figure 4.1. represents a general scheme of a routing problem in transportation:

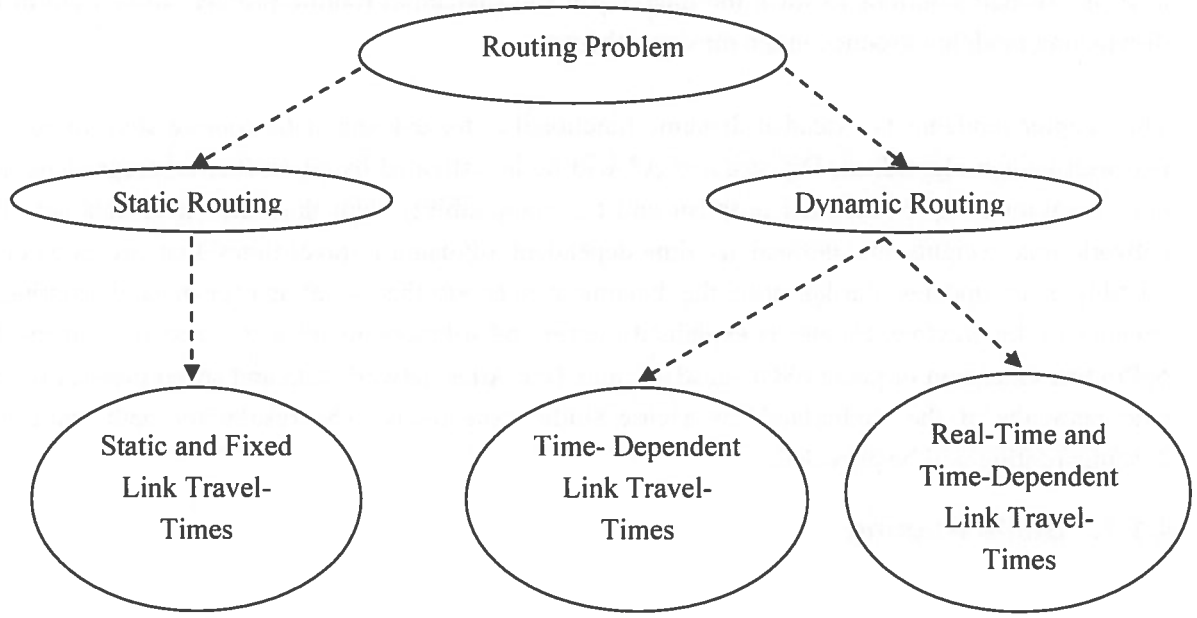

Figure 4.1. Static and Dynamic routing problems

To reach to this goal, pgRouting, the extension of postgreSQL/postGIS was tested for its functionality in static routing with two different shortest path algorithms. The used data set to perform this test was Tele Atlas network spatial data set for the whole Netherlands roads and their related speed profiles. The next section of this chapter will provide the required information about pgRouting and the Tele Atlas routing data set.

4.1.1.1. Tele Atlas Spatial Dataset and Data Preparation

All provided information about Tele Atlas MultiNet dataset is adopted from MultiNet® Shapefile 4.4 vs 4.3.2.2 Format Specifications (2008). The dataset contents are structured according to a layered data model and in a standard GIS format. The following themes are included in the dataset:

- Road and Street Network. Detailed road and street network geometry includes these Attributes:

  Main Attributes: functional road class, network classification, name, alternate name, side of line, route number, length (meters)

The processing status Traffic Attributes: form of way, road condition, slip road type, freeway, back road, construction status, toll, direction of traffic flow, blocked passage, stubble, special restrictions, Z-level information, vehicle type-specific restrictions, restricted time validity, opening period, plural junction, manoeuvres (bifurcations, permitted, priority, prohibited, restricted), signpost information , RDS/TMC (Radio Data System / Traffic Message Channel) locations and path information, intersections, and center point of freeway intersections

Geocode Attributes: official and alternative street names (full and parsing information), side of line, left and right administrative areas (all levels), left and right built-up areas, left and right postal codes (ZIP +4®), left and right address IDs, left and right house number ranges (first, last intermediate, structure, full and base house number information) and official street codes.

- Ferry Connections
- Address Area Boundaries
- Railways
- Points of Interest
- Settlement Centers
- Water Areas and Water Lines
- Land Use and Land Cover
- Built-up Areas
- Administrative Areas and Places
- Postal Districts
- Other Named Areas
- Structures (bridges and tunnels)

The MultiNet geographical dataset is currently designed as a layered data model and it is used by navigation systems like TomTom. The map data are stored, and are grouped conceptually into different thematic units. Each unit contains the following:

- One (or more) geometrical layer(s) each one including a geometry table containing the main attributes directly related to the geometrical features.
- Additional (relational) attribute tables
- Extended attribute tables: contain extra information, or added captured attributes that were not available during the creation of the product format.
- Relational tables: contain other feature IDs within the same layer as an attribute to define a relationship with this feature
- Index tables: provide an index of the elements that form part of a higher-level

The geographic coordinate system is base on the Datum WGS84 with the following parameters and the layers are not projected geographically:

- GEOGCS["GCS_WGS_1984",
- DATUM["D_WGS_1984",
- SPHEROID["WGS_1984",6378137,298.257223563]],
- PRIMEM["Greenwich",0],
- UNIT["Degree",0.0174532925199433295]]

4.1.1.2. Tele Atlas Network Dataset

According to Fayyad, Piatetsky-Shapiro and Smyth (1996) and MacEachren, Wachowiez, Edsall and Haug (1999) extracting information from datasets follows five basic steps:

1. Data selection;
2. Pre-processing;
3. Transformation;
4. Data mining;
5. Interpretation/ evaluation;

To fulfil the above-mentioned steps, out of the verity of available datasets presented in the Tele Atlas dataset for the Netherlands, *NW Network, Geometry with basic attributes* is chosen as the road network dataset, since it has the all required road network information to calculate static least-cost paths. As *NW Network, Geometry with basic attributes* had the related information of three different network elements with the names of *Road Element*, *Ferry Connection Element,* and *Address Area Boundary Element* together; using ArcGIS a dataset of Road Elements is created. To get an idea about the functionality of pgRouting for different datasets with different sizes or data volumes, three categories of roads were defined using the dataset information about road types. Table 4.1. shows the components of each defined category and the total number of relevant road links for each category respectively.

	Road Types in NW Network	Road IDs in NW Network	Number of Links
Major Roads	Motorway, Freeway or other Major Roads	0	52,385
	Major Roads less important than a Motorway	1	
	Other Major Roads	2	
Major and Secondary Roads	Secondary Road	0,1,2,3	109,835
All NL Roads	Local connecting Road	0,1,2,3,4	1,314,710
	Local Road of high importance	5	
	Local Road	6	
	Local Road of minor importance	7	
	Other Roads	8	

Table 4.1. Defined road categories

The created road networks for each category are shown in Figure 4.2. to Figure 4.4. The difference between the number of nodes and the number of links as a function of node numbers for each category are clear in the provided figures.

Fig 4.2. Major Roads Network

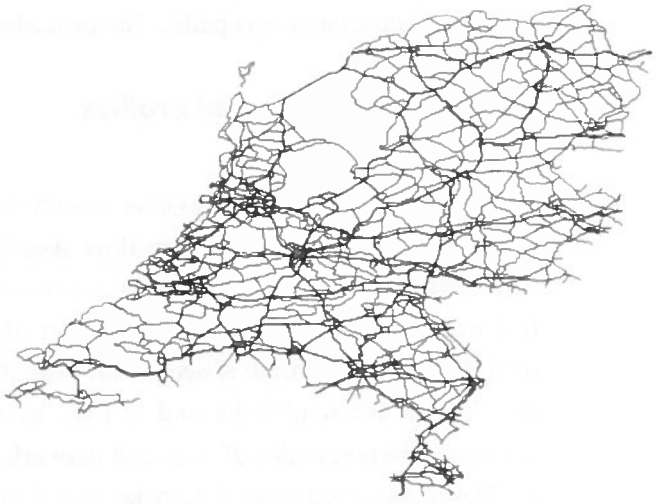

Fig 4.3. Major and Secondary Roads Network

Fig 4.4. All NL Roads Network

In the next step of procedure, the format of created road network categories (.shp) are converted to SQL format to be loaded to the spatial database postgreSQL and to be used in pgRouting for finding static least-cost (shortest) paths. The procedure of format changing is documented in Appendix A.

4.1.1.3. Tele Atlas Speed Profiles

The created dataset in the previous section has not the all required information to handle the dynamic routing procedure since the free flow speed is given in this database for road links and using free flow speed to calculate the real world travel-times cannot be realistic. To have a dynamic route in the real world road networks, time-dependent (dynamic) information of real world travel-times for each road link in the real-time is necessary. Speed profiles provided by Tele Atlas present time-dependent travel-times for road links and is used as the main databases to calculate the total travel-times between different nodes of the road network in this research. The following information about speed profiles are adopted from Tele Atlas Speed Profiles user guide (2008).

Speed profiles aggregate and enhance billions of anonymous GPS probe traces from millions of devices that reflect consumer-driving patterns. These consumer-driving patterns provide true average speeds on individual road segments. The growing number of these data points collected over the

years determines average roadway speeds on a temporal basis. Speed Profiles data can be linked to the transportation network of the related MultiNet release by using the Transportation Element ID.

The data set is feasible to be used with the created extension of pgRouting to have dynamic routing in TTCP. It is noticeable that this data set is time-dependent but not dynamic, to make it a dynamic data set; it is assumed this database could be updated according to the dynamic situations of real world road networks and therefore can be used as a real-time dynamic database.

Since using Speed Profiles does not require any connectivity to other sources (in fact, Speed Profiles can reside in the memory), the initial route calculation can start with Speed Profiles as the primary source for route planning and ETA calculation. Then real-time traffic data can be used as an exception to Speed Profiles.

The speed values are not meant for display and are globally provided in km/h (kilometres per/hour). Freeflow Speed that is the average speed measured during a period of least traffic congestion and most often reflects night-time speeds, average Weekday Speed, and average Weekend Speed are presented separately for all roads. These profiles contain average speed information for every five minutes of the day for each specific day of the week. On the average, there are over 1,000 speed measurements for any road or street included in the product. These measurements are extended significantly on a daily basis. Distinct speed profile is provided for each day of week for each road link.

Historical Speed information values provided are averaged out per Transportation Network Element within a given time domain. The different sets of Historical Speeds behaviour per Time Domain are called Speed Profiles. A Speed Profile provides the behaviour of changes in average speeds during a given time period for a specific Transportation Element (i.e., a Road Element) or a set of Transportation Elements sharing the same behaviour.

Speed Profiles are provided based on Time Bins of 5 Minutes. The Speed Profiles are provided as a set of relative speeds (speed values for a given time or profile are expressed in percentage values relative to the Historical Freeflow Speed) starting after a given period after midnight, valid for 5 minutes. Each Profile has a set of 288 Relative Speeds (i.e., Time Bins where 1 hour = 12 * 5 Min; 24 hours * 12 = 288).The start time is provided as a 'Time Slot'. The Time Slot is a relative start time indication for which a given relative speed is valid. The start time is midnight Time = 0 as start point + Value Time Slot* 5. This results in the number of seconds after midnight when the related Relative Speed is valid for the next 5 Minutes.

Average speed for a Transportation Element can be calculated at any time of the day by combining the freeflow speed of the Transportation Element with the appropriate profile entry for the current time period. If a Transportation Element does not have a Freeflow speed, it is assumed that the Transportation Element's average speed does not change as a function of time. For practical implementation purposes, this average speed is represented by only one value each for the weekday and weekend speeds. Figure 4.5. below shows a single speed profile. The value in the y-axis of the profile is a percentage value of the freeflow speed for a Transportation Element; the value in the x-axis represents the change in time over a 24-hour period.

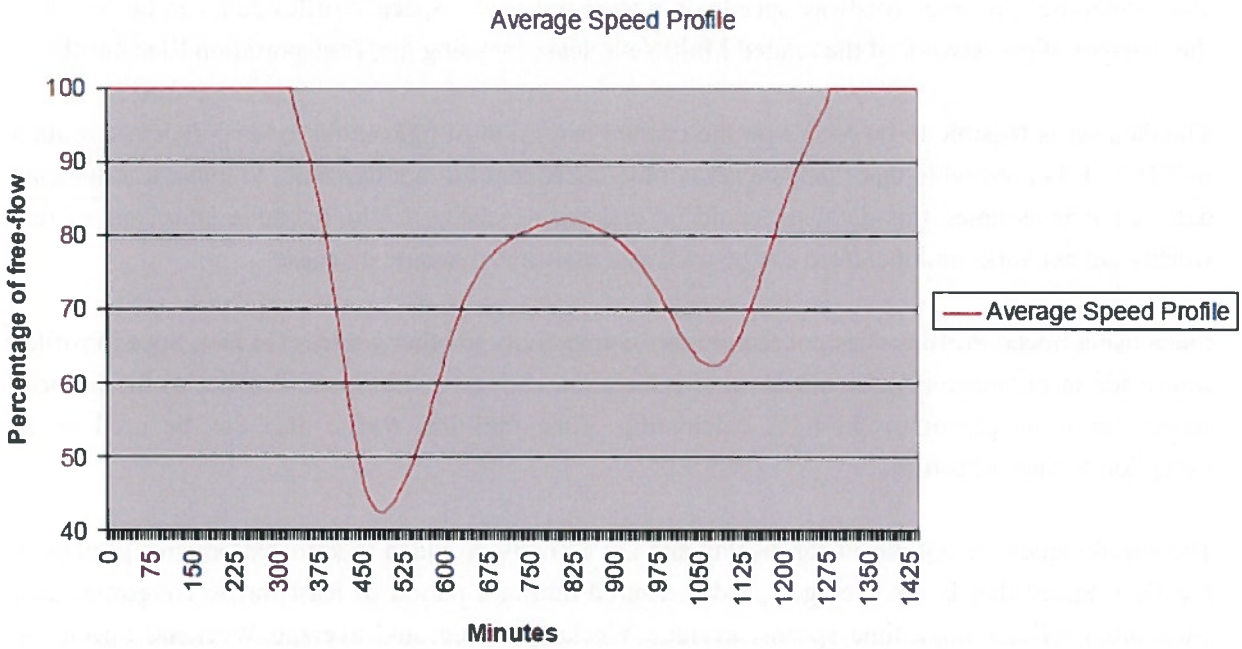

Figure 4.5. Example of a single speed profile (Source: Tele Atlas Speed Profiles user guide, 2008)

The speed values in a profile are relative (i.e., stored as percentages) and a single profile can be applied to multiple Transportation Elements sharing the same traffic congestion behaviour. A single product tile contains approximately 60 speed profiles. An example collection of profiles for one product tile is shown on Figure 4.6. and an example of real- world is provided in Figure 4.7. respectively.

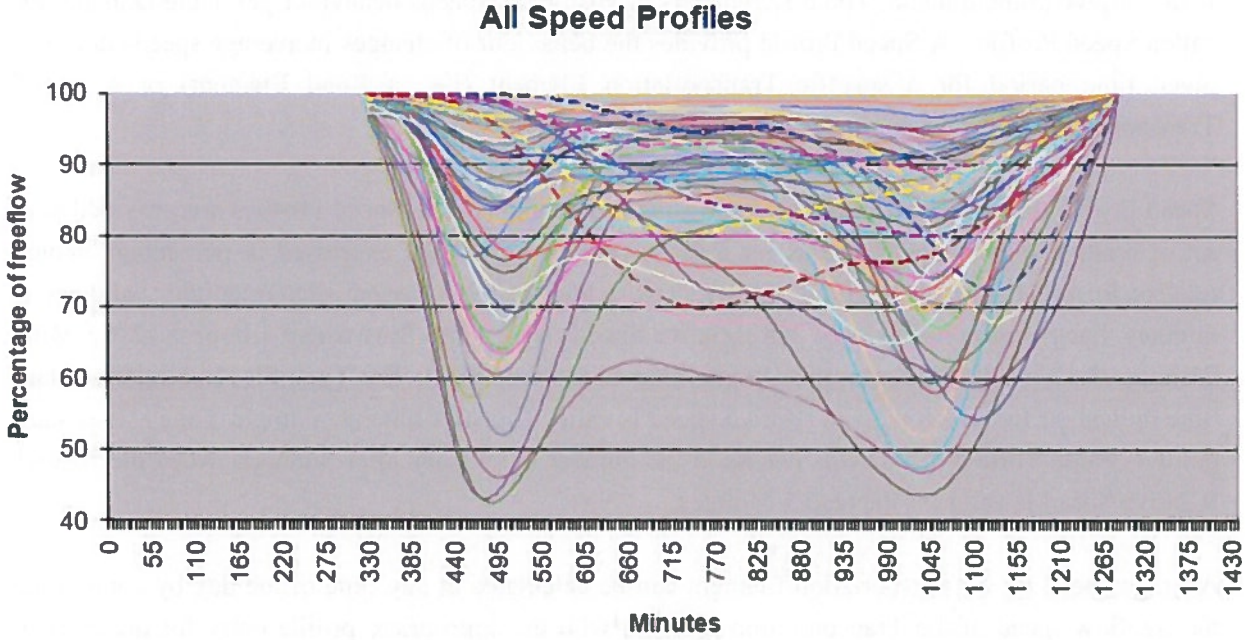

Figure 4.6. Collection of speed profiles for one product tile (Source: Tele Atlas Speed Profiles user guide, 2008)

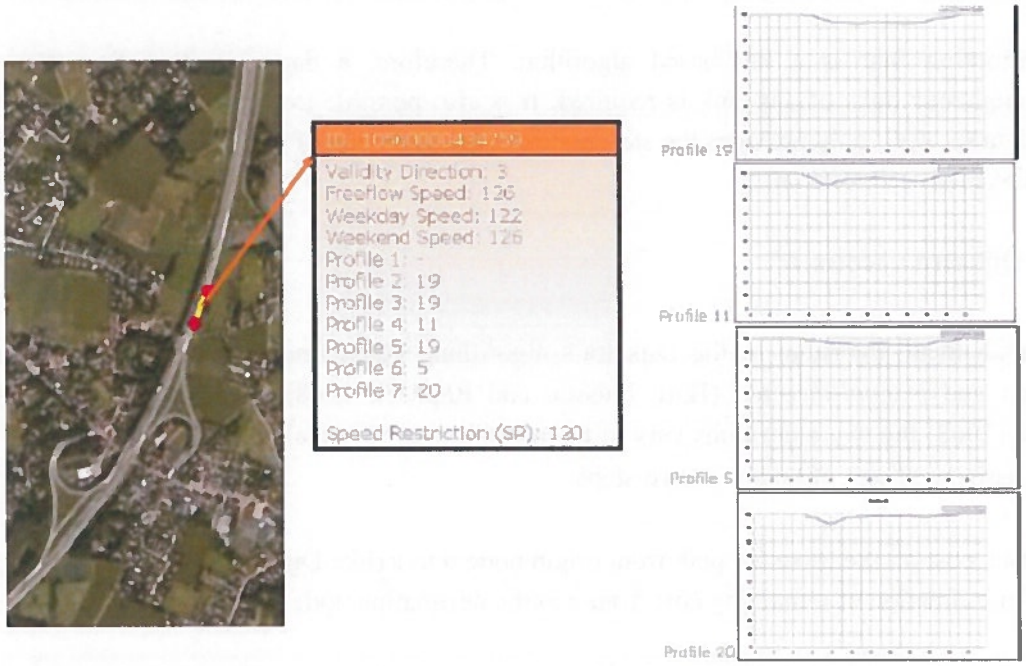

Figure 4.7. An example of real world to show how MultiNet® and Speed Profiles are linked (for a ring road in Gent, Belgium). (Source: Tele Atlas Speed Profiles user guide, 2008)

In Section 4.2. of this chapter, the ETAs (Estimated Time of Arrival based on the departure time) will be defined more precisely and their role in dynamic routing will be explained in greater detail.

4.1.2. pgRouting and Static Routing

As previously explained, pgRouting is the extension of postgreSQL/postGIS, which includes routing capability to postgreSQL/postGIS (http://pgrouting.postlbs.org/). It is an Open Source Software, the main reason to select it as a tool to solve the dynamic routing problem. It provides three different static routing solutions based on three different static shortest path algorithms: Dijkstra as an absolute solution, A* and Shooting star including heuristics. The two well-known algorithms, Dijkstra and A* were explained in Chapter One. Shooting Star is the last production of pgRouting shortest path algorithm, which is developed for real world road networks (http://pgrouting.postlbs.org/). Despite of Dijkstra and A* that finds the shortest path using the nodes of a given network; this algorithm finds the shortest path between an origin node and a destination node using the links of the road network.

The first two algorithms were chosen to be tested in this research since they are well-documented, widely recognized algorithms while shooting star is not. Furthermore Shooting star was deemed unsuitable because:

- There is a need to prepare a network table and add the 'reverse_cost' and 'to_cost' columns. Like A* this algorithm also provides heuristic- based solutions. The problem with these columns in the provided approach is the availability of this information. In the provided dataset having the reverse_cost for road links as a travel-time might be possible but is not straightforward.

- Shooting Star is a link-based algorithm. Therefore, a dataset with information about neighbour links of any link is required. It is also possible to create such a column but the data set itself only provides the start node and the end node of the all links, not the neighbour links for each road link.

4.1.2.1. Dijkstra versus A*

A* is an identical algorithm to the Dijkstra's algorithm, which finds a shortest path to a single destination and is *goal-directed* (Hart, Nilsson and Raphael, 1968). According to Delling and Nannicini (2008) the two algorithms vary in the method used to calculate the cost of each node. A*, finds the associated cost of node v in two steps:

i. The cost of the travelled path from origin node o to v (like Dijkstra algorithm)
ii. An estimation of remaining cost from v to the destination node d

Consequently, the cost of v provides as the all travelling costs from origin o to the destination d via v. A function examines the total cost between a given node and the destination node d *and* is called *potential function* π. In the case of having $\pi(v) \leq d(v, t)$ for any v of V, where $d(v, t)$ is the associated cost from node v to the destination node d, A* provides the least-cost routs (Hart, Nilsson and Raphael, 1968). If $\pi(v) > d(v, t)$ for a v of V, A* becomes a heuristic algorithm.

Patrushev, accessed in September (2009) proposed the following properties for Dijkstra and A* algorithms:

Dijkstra:

- *Well known and fair shortest path algorithm*
- *Always finds mathematically optimal shortest path if there is any*
- *Good for sparse networks*

A*:

- *Well known heuristic shortest path algorithm*
- *Needs node geometry information*
- *Searches through less number of nodes*
- *Good for dense networks*

Routing data structures for Dijkstra and A* in pgRouting are illustrated in Figure 4.8.

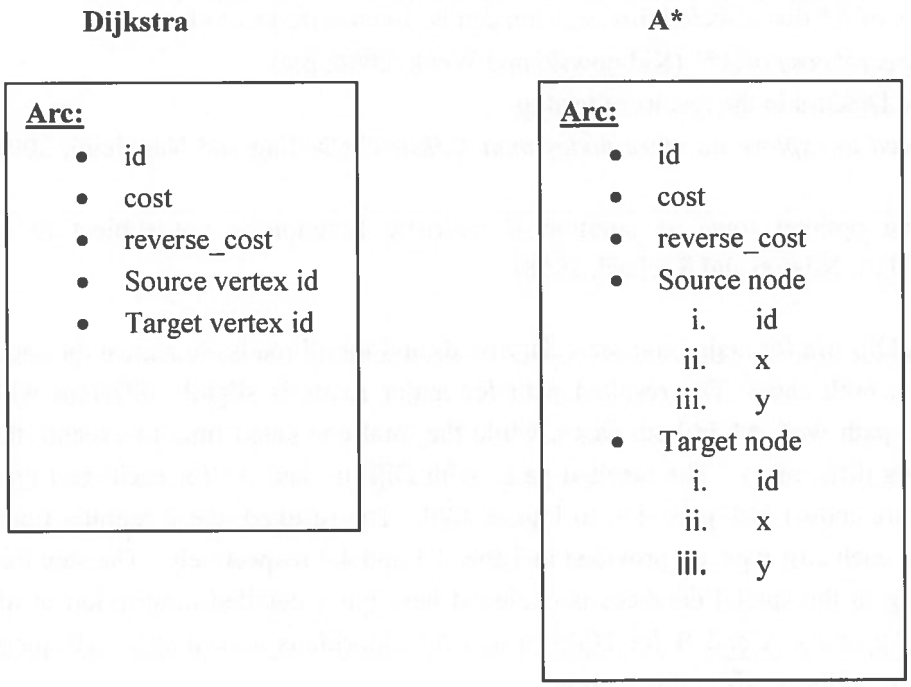

Figure 4.8. Routing data structure in pgRouting
(Source: http://www.comp.dit.ie/pbrowne/Spatial%20Databases%20SDEV4005/Spatialdblecture3.PPT)

In addition, the used functions in each of algorithms to route a vehicle from its origin to the intended destination can be compared as Table 4.2. The H (node1, node2) represents the heuristic function, which is defined as Euclidian distance for pgRouting's *shortest_path_astar ()* function. This function is used in the final presented solution as the function to find the static shortest path result.

Dijkstra	cost=cost(node1, node2)
	Each node can be visited only once
A*	cost=cost(node1, node2)+H(node1, node2)
	Each node can be visited only once

Table 4.2. Dijkstra function versus A* function
(Source: http://www.comp.dit.ie/pbrowne/Spatial%20Databases%20SDEV4005/Spatialdblecture3.PPT)

The two algorithms are tested using three kinds of categorized static road networks of the Netherlands provided by Tele Atlas spatial dataset namely major roads, major and secondary roads, and all Netherlands roads. Two types of static costs are considered separately; distance and average travel-time for each road link. Different characteristics of two algorithms and furthermore, query running times in pgRouting that is a very important criterion in a dispatching procedure made us to choose the A* as the algorithm which its functionality should be extended to handle the dynamic situations in pgRouting.

The important properties of A* that affected this decision can be summarized as follows:

- *"Dijkstra is a special case of A*"* (Kalinowski and Wenk, 2006; p.4)
- A* is faster than Dijkstra in the results of testing
- "*A* is guaranteed to explore no more nodes than Dijkstra*" (Delling and Nannicini, 2008; p.5)
- A* provides the optimal route as solution if heuristic function is not subject to be overestimated (Hart, Nilsson and Raphael, 1968)

The resulted paths from Dijkstra for major and secondary roads and for all roads are almost the same as the paths with A* in both cases. The resulted path for major roads is slightly different with compare to the resulted path with A* in both cases, while the total computed time to execute the required queries have big differences. The resulted paths with Dijkstra and A* for each road type and for each cost type are shown in Figure 4.9. to Figure 4.20. The resulted query running times with Dijkstra and A* for each cost type are provided in Table 4.3 and 4.4 respectively. The step that is related to data loading to the spatial database is excluded here but a detailed description of all steps can be found in Appendix A and B for Dijkstra and A* algorithms accordingly. All query running times are given in milliseconds in this section.

	Major Roads	Major and Secondary Roads	All NL Roads
Step One	43	30	1,051
Step Two	3,550,261	15,055,383	85,624,616
Step Three	2,091	4,717	543
Step Four	541	6,877	260,986
Total Time	3,552,936	15,055,413	85,887,196

Table 4.3. Dijkstra query-running times with pgRouting using distance as cost

	Major Roads	Major and Secondary Roads	All NL Roads
Step One	43	30	1,051
Step Two	3,550,261	15,055,383	85,642,616
Step Three	2,091	4,717	543
Step Four	681	1,444	325,285
Total Time	3,553,076	15,061,574	85,969,495

Table 4.4. Dijkstra query-running times with pgRouting using travel-times as cost

	Major Roads	Major and Secondary Roads	All NL Roads
Step One	31	31	21
Step Two	105,779	250,146	1,378,478
Step Three	82	43	244
Step Four	53	165	211
Step Five	219	539	139,493
Total Time	106,164	250,924	1,518,447

Table 4.5. A* query-running times with pgRouting using distance as cost

	Major Roads	Major and Secondary Roads	All NL Roads
Step One	31	31	21
Step Two	105,779	250,146	1,378,478
Step Three	31	21	244
Step Four	81	60	211
Step Five	584	750	116,705
Total Time	106,506	251,008	1,495,659

Table 4.6. A* query-running times with pgRouting using travel-times as cost

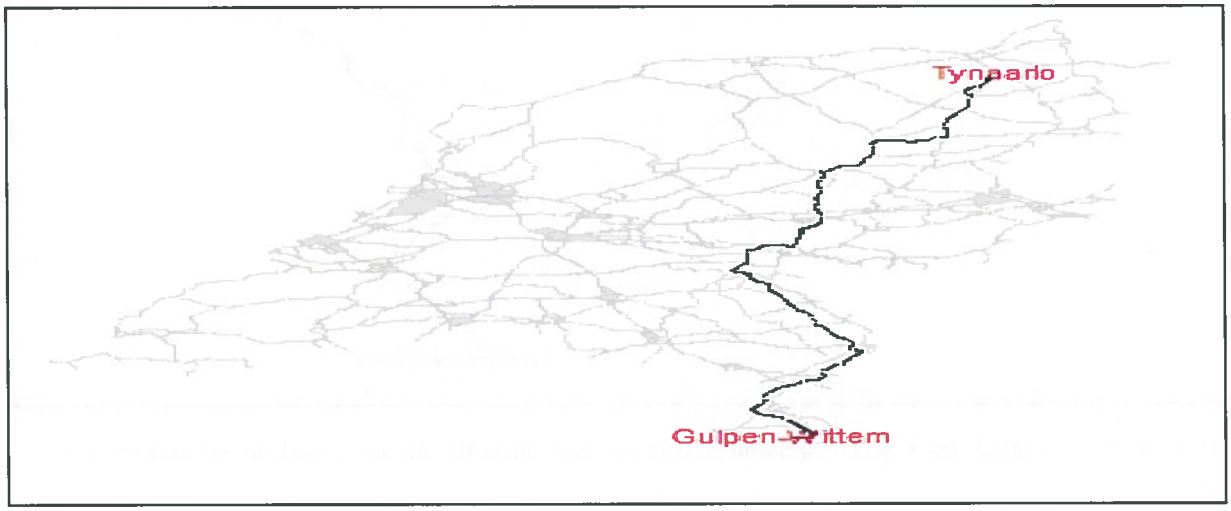

Figure 4.9. The resulted path in Major roads with Dijkstra between Gulpen-Wittem and Tynaarlo using distance as cost

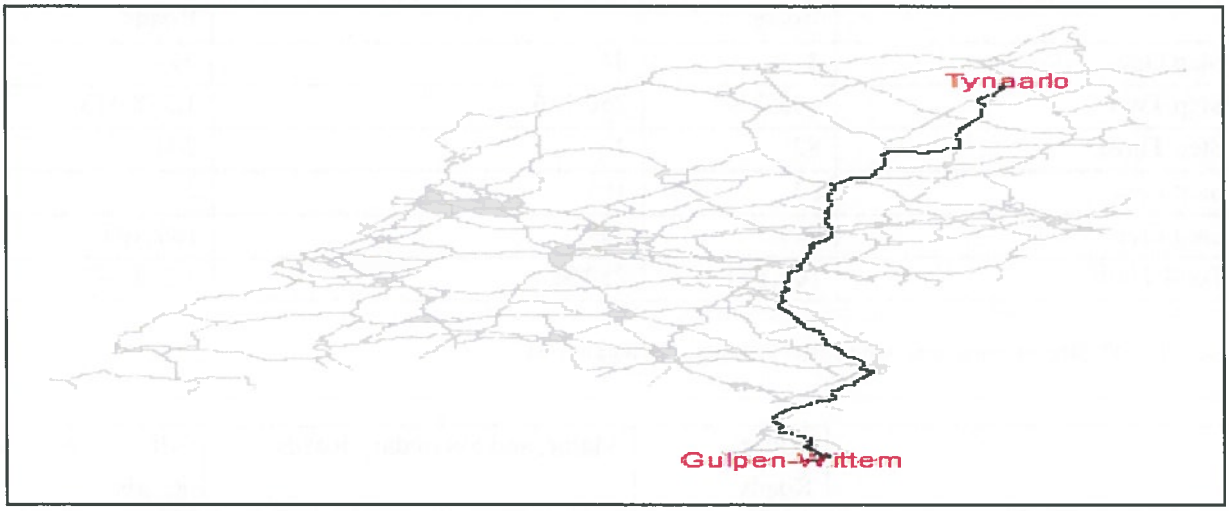

Figure 4.10. The resulted path in Major roads with A* between Gulpen-Wittem and Tynaarlo using distance as cost

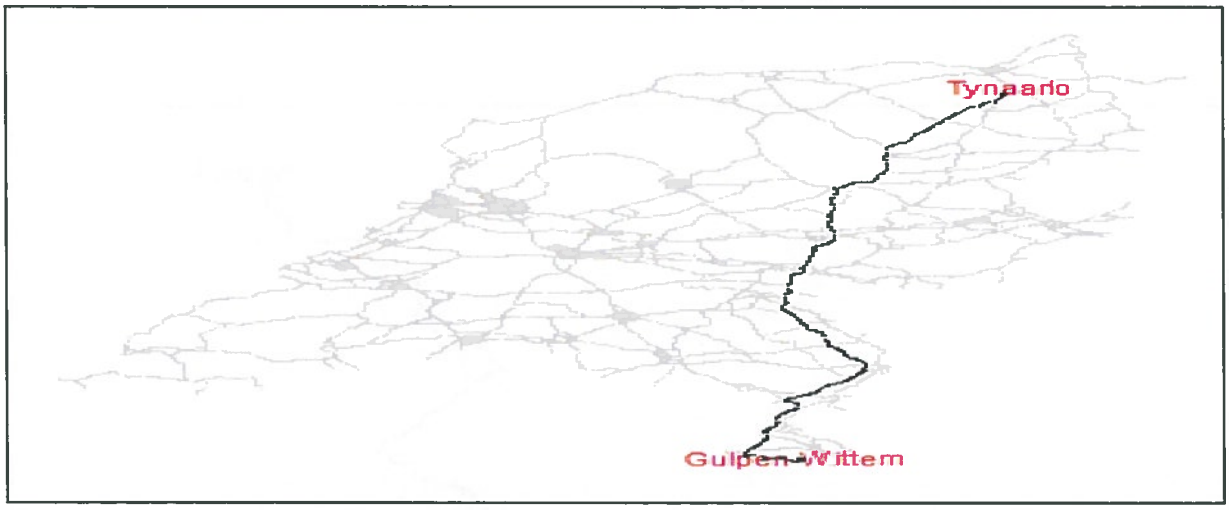

Figure 4.11. The resulted path in Major roads with Dijkstra between Gulpen-Wittem and Tynaarlo using travel-times as cost

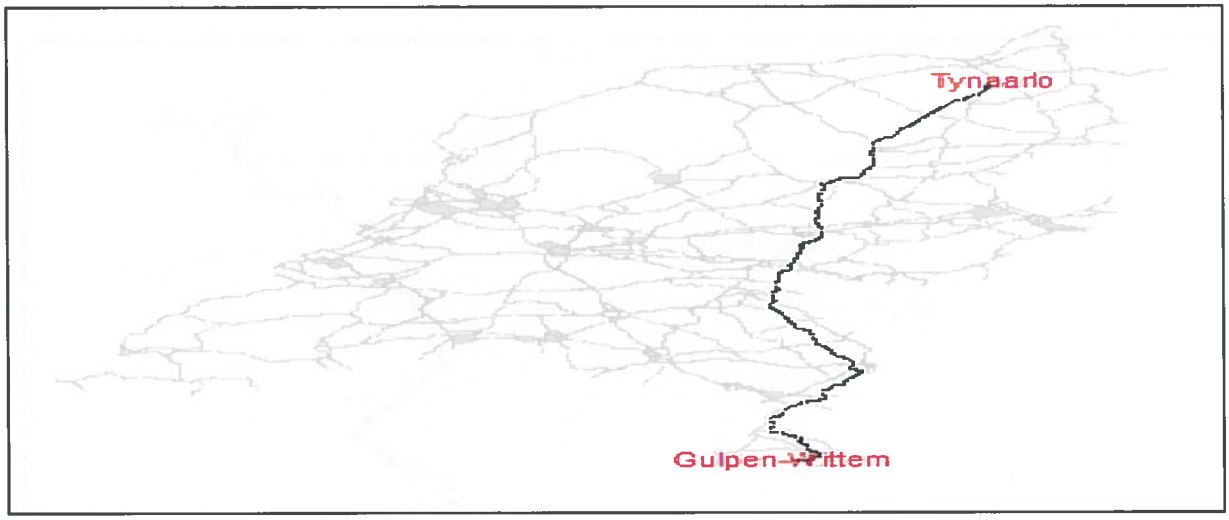

Figure 4.12. The resulted path in Major roads with A* between Gulpen-Wittem and Tynaarlo using travel-times as cost

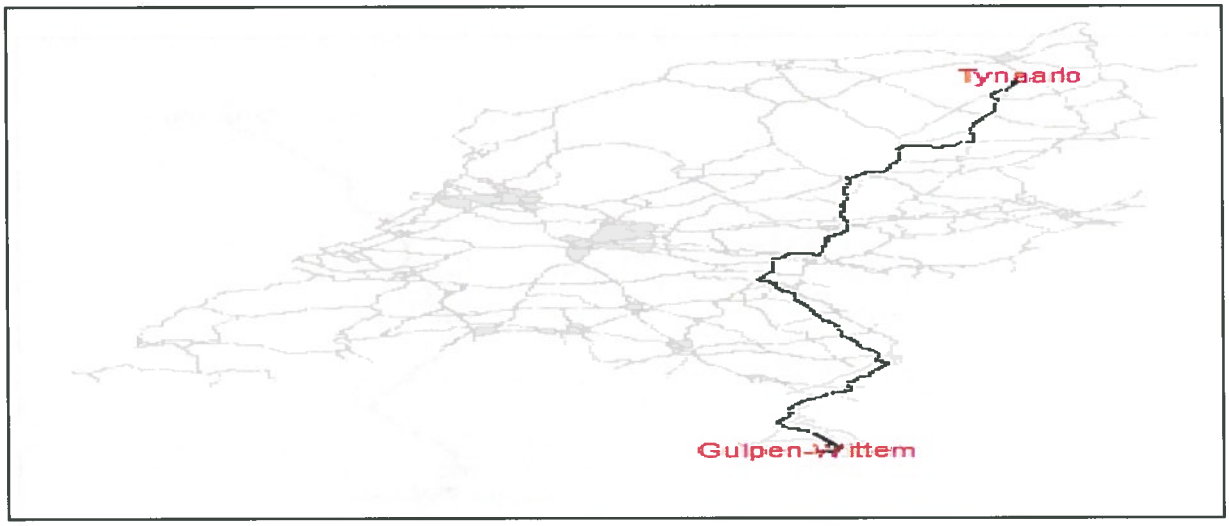

Figure 4.13. The resulted path in Major and secondary roads with Dijkstra between Gulpen-Wittem and Tynaarlo using distance as cost

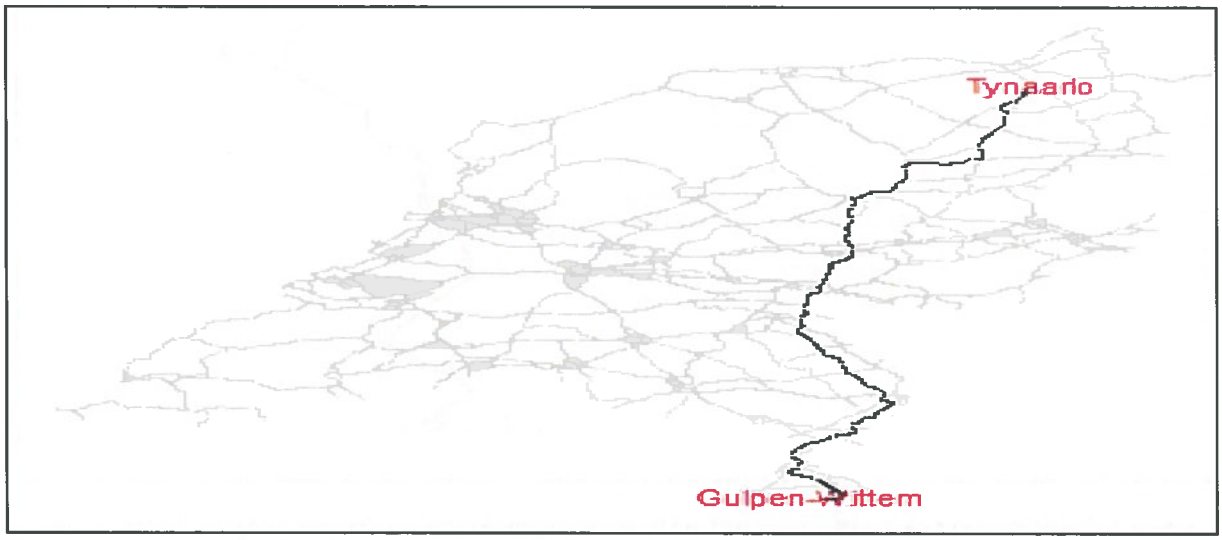

Figure 4.14. The resulted path in Major and secondary roads with A* between Gulpen-Wittem and Tynaarlo using distance as cost

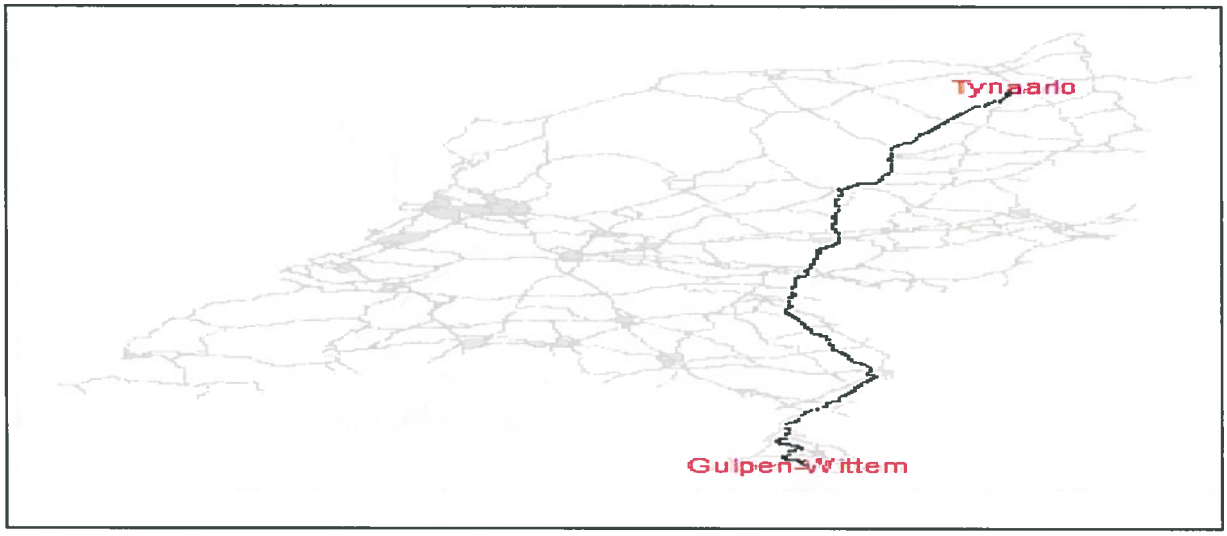

Figure 4.15. The resulted path in Major and secondary roads with Dijkstra between Gulpen-Wittem and Tynaarlo using travel-times as cost

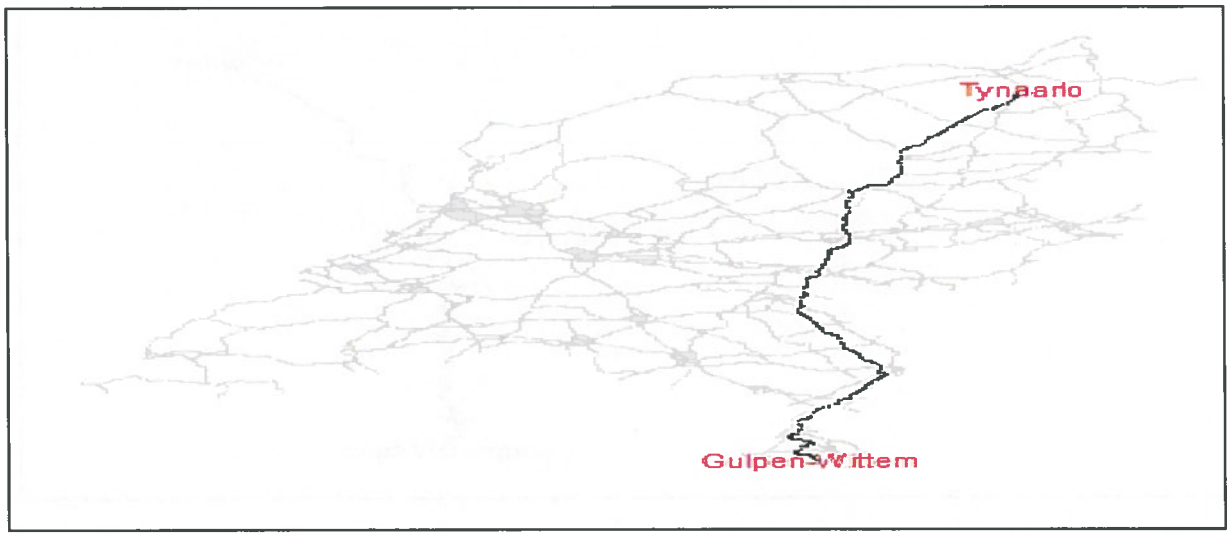

Figure 4.16. The resulted path in Major and secondary roads with A* between Gulpen-Wittem and Tynaarlo using travel times as cost

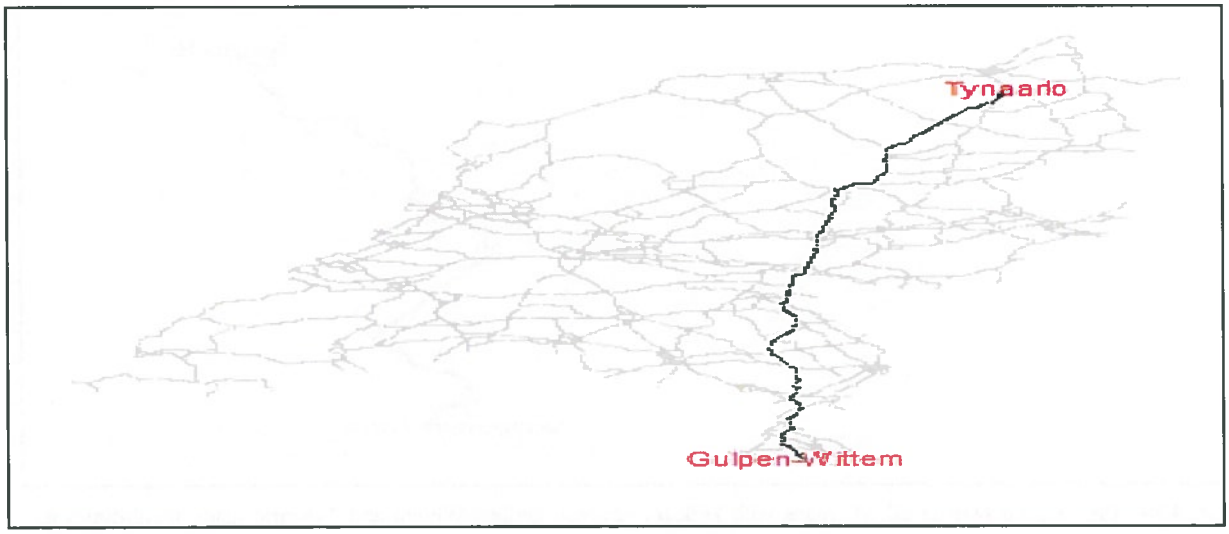

Figure 4.17. The resulted path in all NL roads with Dijkstra between Gulpen-Wittem and Tynaarlo using distance as cost

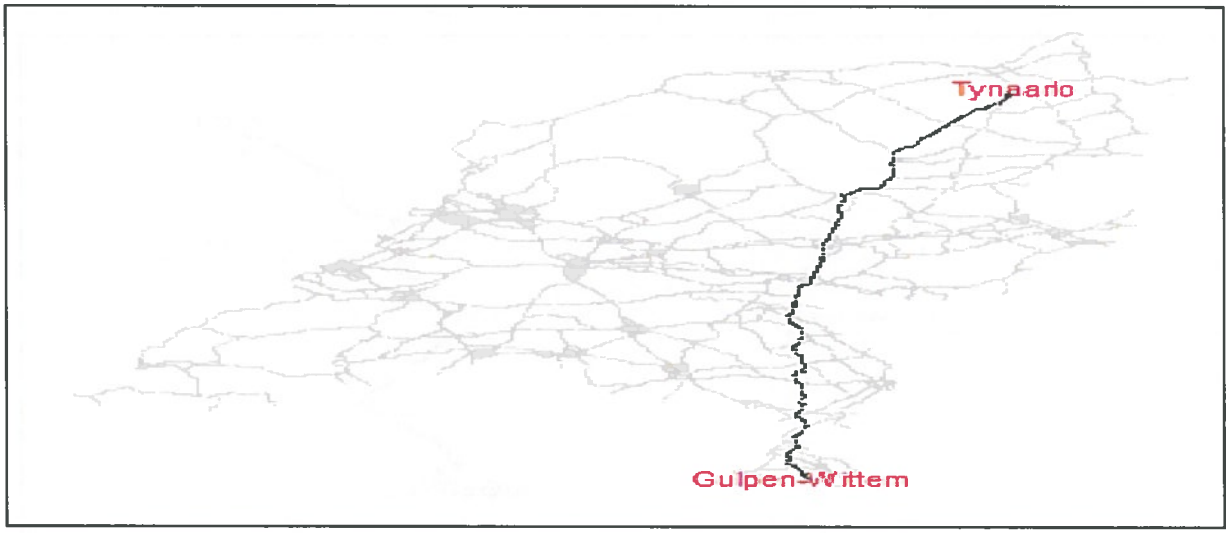

Figure 4.18. The resulted path in all NL roads with A* between Gulpen-Wittem and Tynaarlo using distance as cost

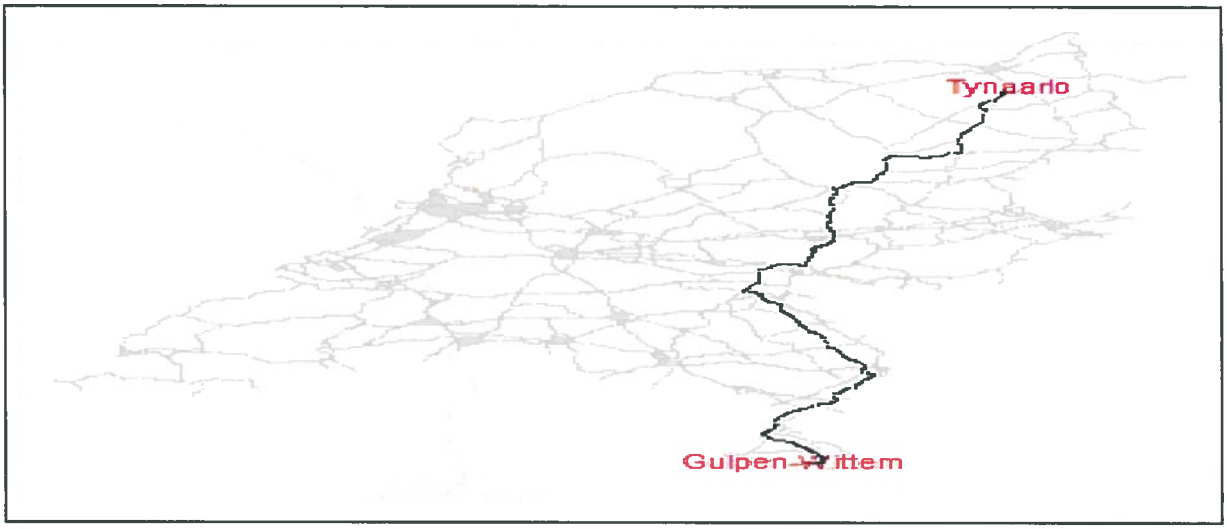

Figure.4.19. The resulted path in all NL roads with Dijkstra between Gulpen-Wittem and Tynaarlo using travel-times as cost

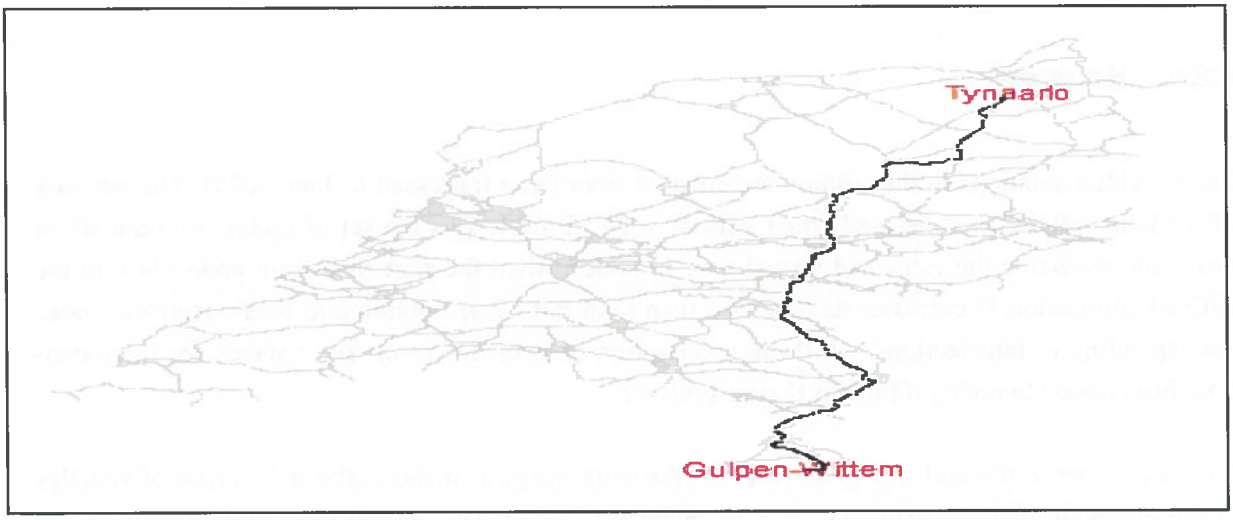

Figure. 4.20. The resulted path in all NL roads with A* between Gulpen-Wittem and Tynaarlo using travel-times as cost

4.2. Dynamic Routing

Because of the dynamic nature of travel-times in different times, dynamic routing is more realistic, but also more computationally intensive when there is a need to answer the customers on the phone or online. Static routing might be enough to provide a route very quickly but it will not be accurate compared to the reality.

One of the most important applications of dynamic routing is to guide vehicle drivers via optimal routes (Kalinowski and Wenk, 2006). In the current navigation systems current traffic conditions of real-world is not taken into account generally. In the case of having access to the real-time traffic information dynamic routing algorithms will be able to handle dynamic and updated travel-times for road links.

As such illustrated in Figure 4.1, *dynamic* is used here to determine the varying travel-times on road networks links. In addition having constructions on the roads, results to eliminate that road links from the original network, this causes to have a dynamic network. Other dynamic conditions like the behaviour of the drivers are not considered for the purpose of this research.

As it is mentioned in Chapter Two, Dean (accessed in June 2009) explained two different kinds of dynamic shortest path problems with concentration on the FIFO networks. In the first type, the shortest path should be constantly recomputed because of network data is subject to frequent changes in an unpredictable fashion. The presented solution in this way can be expressed as the reoptimized solution of the series of related static solutions. In the second type, there is a rule to predict the changes in network travel-times. In the formulated problem the travel-times are assumed as the known functions of departure times from link start nodes. Regarding this definition, the problem is defined as a time–dependent but not a *dynamic problem*. In the transportation field these problems occur most of the time. The proposed solution attempts to answers to both of these situations: first a

time-dependent shortest path is calculated and then it becomes updated in each new travel-time arriving.

4.2.1. Notation

The provided notations in this section are adapted from Dean (accessed in June 2009). The network of roads is defined as a directed graph with $G = \{V, E, eta\}$. V is the set of nodes, E is the set of links, and *etaij(t)* is the estimated arrival time to node j when the start time from node i is t. In the defined application G corresponds to a road map with links corresponding to road segments, nodes corresponding to intersections. With the assumption of $etaij\ (t) \geq t$, $aij\ (t) - t$ stands for the travel-time from node i to node j if the travel start-time is t.

ETA stands for '*estimated time of arrival*'. As the name suggests it shows the arrival time of vehicles to the end-node of each road link. It is assumed that the arrival time to each middle node in the shortest path result is the same as the departure time of that node. Any delay in nodes is not considered here.

Despite of static routing using speed profiles data as ETAs in time-dependent and dynamic routing adds new dimensions to the presented solution as time and day (Tele Atlas Speed Profiles user guide, 2008). Although adding the mentioned dimensions to the problem improves the accuracy, it makes the problem more complicated.

4.2.2. Implementation of Dynamic Routing Solution

To cope with the dynamicity scenarios, which were defined in the last section of previous chapter, two different solutions are provided in this section. These solutions calculate the path dynamically and are adapted to be used with pgRouting and any spatial dataset, which provides the required information, like Tele Atlas. The first solution provides a time-dependent algorithmic solution to be used instead of static A* algorithm in the source code of the pgRouting to compute a time-dependent shortest path. The provided time-dependent A* algorithm is not implemented here since the source code is very complicated to be changed. The second solution is a stepwise algorithm, which makes use of static A* algorithm to generate a time-dependent and dynamic solution.

The static A* algorithm's pseudo-code is modified to extend the functionality of pgRouting to handle time-dependent routing. This solution is explained theoretically. The second solution is explained theoretically and is implemented using the created function for postgreSQL/postGIS and Tele Atlas data set. The results of implementation are provided.

4.2.2.1. Using Speed Profiles for Dynamic Least-Cost Paths

Goodchild (accessed May 2009; p.5) defined a data model as "*the set of entities and relationships used to create a representation of some real phenomenon or phenomena*".

The initial idea about modelling dynamic travel-times for real world road networks was defining specific travel-time functions for road links in the form of step-wise functions for different road types like the one in Figure 4.21. It was found from the literature that step-wise functions cannot cope with FIFO constraint in the road networks in a proper way (Haghani and Jung, 2005). These functions break the time of planning horizon into very small distances. Applying step-wise functions to networks with FIFO property causes to have an unreasonable situation. As it is illustrated in the Figure 4.11, a vehicle that started to travel after another vehicle can receive earlier to the destination before it, and this result disturbs the FIFO property. According to these functions any start time between *a* and *b* results to reach to the destination later than start time at *b* and so it is wise to depart at time *b*. This means vehicles should wait until time b and this is not the case of real world road networks situations. Consequently, it is not possible to introduce road network link travel-times as step-wise functions.

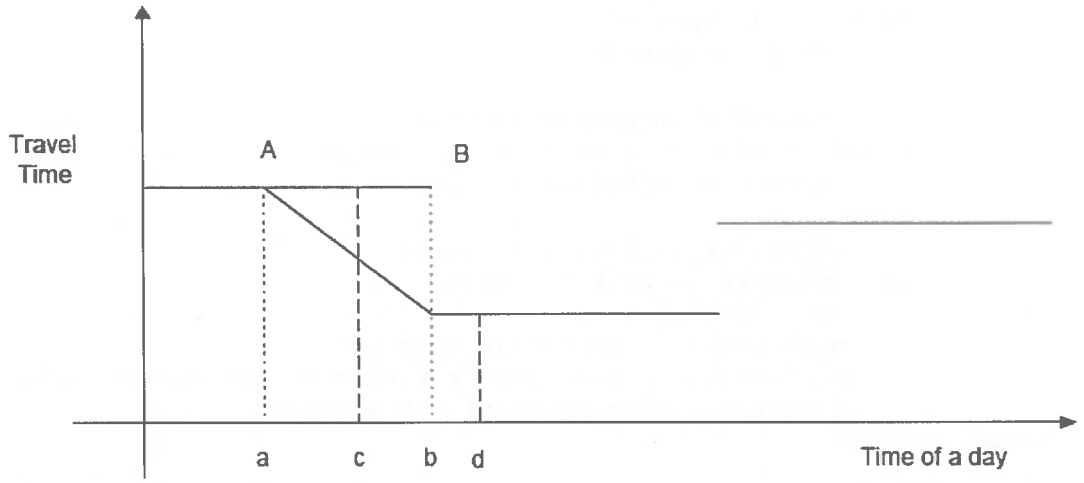

Figure. 4.21. A stepwise function for travel-times (Source: Haghani and Jung, 2005)

Another option to have more realistic travel-times information is having them in a spatial database and working with software or a tool, which can make use of this information. As a more logical choice, we come up with choosing the Tele Atlas spatial database and postgreSQL/postGIS extension to continue our study.

4.2.2.1.1.Time-Dependent A* Algorithm

The first approach to solve the dynamicity and time-dependency in routing problem is to modify the static A* algorithm. The modified A* search algorithm can be used to cover time-dependency and to

include ETAs. This results to reflect the speed profiles data in the final provided solution by the modified algorithm. The proposed time-dependent pseudo-code is based upon A* search algorithm that can be found in http://en.wikipedia.org/wiki/A*_search_algorithm#Pseudo_code , for which the pseudo-code is in below:

```
1 function A*(start,goal)
2     closedset := the empty set    //The set of nodes already evaluated
      openset := set containing the initial node // The set of tentative
                                                    nodes to be evaluated

3     g_score[start] := 0           // Distance from start along optimal path
4     h_score[start] := heuristic_estimate_of_distance(start, goal)
5     f_score[start] := h_score[start]   // Estimated total distance from
                                            start to goal through y
6     while openset is not empty
7         x := the node in openset having the lowest f_score[] value
8         if x = goal
9             return reconstruct_path(came_from,goal)
10        remove x from openset
11        add x to closedset
12        foreach y in neighbor_nodes(x)
13            if y in closedset
14                continue
15            tentative_g_score := g_score[x] + dist_between(x,y)

16            if y not in openset
17                add y to openset

18                tentative_is_better := true
19            elseif tentative_g_score < g_score[y]
20                tentative_is_better := true
21            else
22                tentative_is_better := false
23            if tentative_is_better = true
24                came_from[y] := x
25                g_score[y] := tentative_g_score
26                h_score[y] := heuristic_estimate_of_distance(y, goal)
27                f_score[y] := g_score[y] + h_score[y]
28    return failure

29 function reconstruct_path(came_from,current_node)
30    if came_from[current_node] is set
31        p = reconstruct_path(came_from,came_from[current_node])
32        return (p + current_node)
33    else
34        return the empty path
```

The main modification made to the pseudo-code of A* above is the search between source and target has been changed from a pure-distance search to a time-based search using ETAs. It is adapted to use Speed Profiles data during the search.

```
1 function A*(Graph, origin, destination):
2     for each node v in Graph: // Initializations
3        cost[v]:= g[v]:= infinity // Unknown cost functions from
                                    origin to v
4        previous[v]:= undefined // Previous node in optimal path
                                 from source
5        eta[v]:= undefined // Departure time on node v
6        cost[origin] :=g[origin]:= 0 // Cost from source to source
7        eta[source] := current time // Departure time at source
8        Q := the set of all nodes in Graph // All nodes of the
                                              graph
9     while Q is not empty: // The main loop
10       u := node in Q with smallest cost[]
11       remove u from Q
12          if u == target
13            return found // stop the search when target is found
14          for each neighbour v of u: // where v is still in Q.
15            alt := g[u] + cost_between(u, v, eta[v])+h[v]
16            if alt < cost[v] // Relax (u,v)
17               g[v]:= g[u]+ cost_between(u, v, eta[v])
18               cost[v]:= g[v]+h[v]
19               previous[v] := u
20               eta[v] := eta[u] + traveltime_between(u, v, eta[v])
21    return not found
```

The provided algorithm is a modification of general static A* algorithm and time-dependent A* algorithm presented in Kalinowski and Wenk (2006).The modified algorithm for a directed network with non-negative link travel-times is explained and *cost_between (u, v, eta[v])* is indicated as the travel-time of arc *(u, v)* when the estimated arrival time in v is *eta[v]* (Kalinowski and Wenk, 2006). The algorithm maintains a list of nodes .A node *v* should be stored with the value *cost[v]* as its cost from origin. Cost value for any node $v \in V$ can be defined as $cost[v] = g[v] + h[v]$, *g[v]* is the cost of a least-cost path from *o* to *v* and *h[v]* is a heuristic function that gives the cost of a least-cost path from *v* to *d*. In Dijkstra's algorithm heuristic function is defined 0.

The modified algorithm can be applied to road networks with time-dependent link travel-times using speed profiles as follows: Assume current time be the time of departure from node o and set *g[o]* = current time (Kalinowski and Wenk, 2006). Time-dependent travel-times *cost_between (u, v, g[u])* are used instead of constant travel-times for the arc *(u, v)* in this algorithm. In node *u* total travel-time from origin to this node is computed and it is equal to *g[u]*. *cost_between (u, v, g[u])* represents travel-times; *g[u]* is the departure time from node *u* and starting time to travel along arc *(u, v)* in the case of using least-cost path from *o* to *u*. Kaufman and Smith (1993) showed that this approach computes a correct least-cost path only whenever the network has FIFO property (the *cost_betweens* maintain the FIFO characteristics):

For all $e \in E$ and $t1, t2 \in T$ (*T* is a suitable representation for different times on different days, months, years, etc):

$$If\ t1 \leq t2\ then\quad t1 + cost\_between\ (e, t1) \leq t2 + cost\_between\ (e, t2)$$

This indicates that if the departure time of a car from the link start node is later than another car, it cannot receive to the end point of that link before the car that is departured before. The assumption here is that the time-dependent travel-time functions in the network fulfil FIFO condition. In another word it is assumed that the vehicle that is used to provide dataset was tested in the networks with FIFO property.

The main modification to apply dynamicity to the standard A* are on lines 5, 7, 15, 17 and 20. The travel start-time for each node of road network is required to calculate the travel-times. The eta values for the nodes are initialized on line 5. The start-time from the source node is set on line 7. In the course of least-time path calculation, start-time should be used to calculate the total travel-time for travelling from *u* to *v* (line 15), and also to calculate the arrival time to the node *v* (line 20).

Prior to the adaptation of the presented pseudo-code to be worked with pgRouting, ETAs should be calculated since the written pseudo-code needs them to calculate travel-times and find least-cost paths.

Two variant of this approach are provided below to extend its functionality:

i. Recursive Least-Cost Paths

To extend the functionality of the provided pseudo-code in practice and to cope with the dynamicity of travel-times in real world road networks one way is to route vehicle node by node. This gives a real-time solution if there is a database of real-time travel- time data. To do so the first variant of the above mentioned dynamic routing solution relies on the recalculation of the calculated least-cost paths recursively. This includes the new travel-time's information and the occurred changes, if there are any. For example updating the path every 30 seconds, or some other time interval, based on the expected occurrence of new updates to the link travel-times can result to a real-time solution. This solution does not take into account if new travel-time information has arrived after the last calculation. The paths are calculated and recalculated up to the time that vehicle arrives at the intended destination. It is clear that in the case of having no new information about travel-times or in the case of minor changes, the new least-cost paths will be the same as the previous ones. However, this may cause rerouting of vehicles along other road options when significant changes have happened along the road links of previous routes.

To adopt this method to be used in pgRouting, a while loop should be created before calling the function *dyn _boost_astar ()/ Boost_astar ()* in the source code. These functions calculate least-cost paths using A* algorithm in pgRouting.

ii. Updated Least-Cost Paths for New Travel-times

The second variant of the first solution is resulted from this fact that computing the calculated least-cost paths recursively is not wise if there is no new information. Instead, it is possible to extend the functionality of previous variant to recompute least-cost paths just whenever changes

are reported in travel-times in one (more) of the assigned route links. To do so the A* algorithm's source code in pgRouting library should be modified by definition of an extra *if* clause.

To have more efficient updates a data structure should be defined that provides facilities for computer to quickly determine which paths need to be recalculated when a single link travel-time is updated (Kalinowski and Wenk, 2006). A pointer should be defined for each road link used in the planned path and this pointer should be used in each update to find the affected links.

4.2.2.1.2. Dynamic Routing Function

The second approach provides a step-by-step algorithm which applies a method to find time-dependent and dynamic paths using a static shortest path function of pgRouting. This algorithm is implemented using '*plpgsql*' language in postgreSQL/postGIS. The used function is *shortest_path_astar (text, integer, integer, boolean, boolean)* as the static shortest path function. The road network is the road network of major roads in the Netherlands. This network is provided in the Tele Atlas data set as a geometry table. The algorithm consists of sixteen steps, which are provided below:

Step 1:

Get the origin node, destination node, start time, start day and set i=1;

Step 2:

Create a table to store results of least-cost path in it;

Step 3:

Create a sub-table of road's table for all neighbour nodes of origin node;

Step 4:

For all links in step 3, investigate if there is any real-time information in the table of updated real-time travel-times;

Step 5:

If there is, update the travel-time of the related link in the road's table to the real-time travel-time value of this table; (It is assumed there is a table, which stores the real-time travel-times and gets updates whenever a change happens in the real world road links. The associated value for each road link in this table is assumed null if there is no new information about its travel-time.)

Step 6:

If there is no real-time data, calculate the travel-time based on the speed profiles of the data set for the start time on the given day;

Step 7:

Update the travel-time of the link in the road's table to the calculated travel-time in step 6;

Step 8:

Investigate if there is any real-time information in the table of updated real-time travel-times for all links of roads;

Step 9:

If there is, update the travel-times of the links one by one in the road's table to the real-time travel-time values of this table;

Step 10:

Calculate the least-cost path from origin node to the destination node based on the new resulted travel-times;

Step 11:

Find the maximum coordinates(x, y) for all the nodes in the resulted shortest path in step 10;

Step 12:

Applying a threshold make a box using the resulted coordinates of step 11 and cut the rest of the road network (to limit the search space);

Step 13:

Insert the first row of the resulted path in the step 10, into the table created in the step 1 as its *ith* row;

Step 14:

Set the new origin as the end-point of the resulted first link in the step 10;

Step 15:

Check the new origin, if it is not the same as the destination node go to the step 16,

Step 16:

Set the new start time as the previous start time plus the travel-time of the previous link and go to the step 2;

In the first iteration, the algorithm finds a least-cost path between the origin and the destination with time-dependent travel-time only for the first link and with the static travel-times for the rest of links in the resulted least-cost path. In the second iteration, it proposes a least-cost path with time-dependent travel-times only for the first two links of the resulted path. It changes the travel-time of the links of the resulted path gradually in each iteration up to the time that new origin becomes the same as the destination node and finally gives a route with time-dependent travel-times for all links of the planned route. It is clear that the proposed algorithm includes the last updated travel-times of the next links in the solution only in the case of having new information about them. In the other words the algorithm takes into account all the reported congestions for all road network links in each iteration.

It is argued that the resulted path is time-dependent since it provides different travel-time values for different days and also for different times of day. It is dynamic since in each iteration a new path might be generated, which differs from the previous one and it is subject to changes in the next iterations. Also all the involved tables are created dropped or updated dynamically. It is real-time because it gets the updates from real world for road link travel-times, using both the main function during path finding procedure and the trigger function after path finding procedure (is explained below).

To test the functionality of the presented function in the postgreSQL/postGIS, a network of 8 nodes and 7 links is created. Table 4.7. to Table 4.9. are shown the examples of resulted paths for 9:00 on Monday morning, 00:00 on Monday night and 9:00 on Sunday morning. The differences between resulted travel-times are obvious for different start times in a day and for the same start time in different days of a week. Since the used network for these examples is a small network and the difference between link travel-times for different start-times is small also, resulted paths are the same for all cases but differs in the resulted total travel-times. It is clear for long distances; the planned path won't be a constant path like the provided examples. The running time of the function for the mentioned network is few milliseconds.

Link Source Node	Travel Time(s)
10218	12.7997448979
10217	14.7145784543
10266	1.12171997062
10268	23.5088351965
34826	0

Table. 4.7. The resulted path for 9:00 on Monday morning using the dynamic extension of pgRouting

Link Source Node	Travel Time(s)
10218	12.7997448979
10217	13.1842622950
10266	0.98823529411
10268	16.6442553191
34826	0

Table 4.8. The resulted path for 00:00 on Monday night using the dynamic extension of pgRouting

Link Source Node	Travel Time(s)
10218	11.4685714258
10217	14.8304412768
10266	1.11162575266
10268	18.7224469281
34826	0

Table 4.9. The resulted path for 9:00 on Sunday morning using the dynamic extension of pgRouting

In another test the travel-time of link with start-node id equal to 10217 and end-node id equal to 10266 is set to 800. The resulted path is changed. Table.4.10. illustrates the path.

Link Source Node	Travel Time(s)
10218	12.7997448979
10217	20.6906035141
10265	1.27419233376
10267	15.5636537427
34826	0

Table. 4.10. The resulted path with the changed travel-time using the dynamic extension of pgRouting

To be able to understand the reality and the concepts behind it, making some abstractions of it is required (Couclelis and Gale 1986).

"*Conceptual models provide the possibility to communicate spatial concepts in a formal and unambiguous way and Concepts are used to organize space and structure the perception of reality*" (Timpf, 1992; p.1).

A conceptual model can be defined to model and abstract the real world objects before including them into logical and physical design step (Brodie, Mylopoulos and Schmidt, 1984). As Timpf (1992) mentioned, in complicated applications conceptual models are' guaranteed' models. They can be used to make a connection between different project members and users. Furthermore, having an initial idea about real-world objects before starting any scientific study is essential.

A conceptual model which represents the workflow between different tables of the created function for postgreSQL/postGIS is provided in Figure 4.22. in the next page.

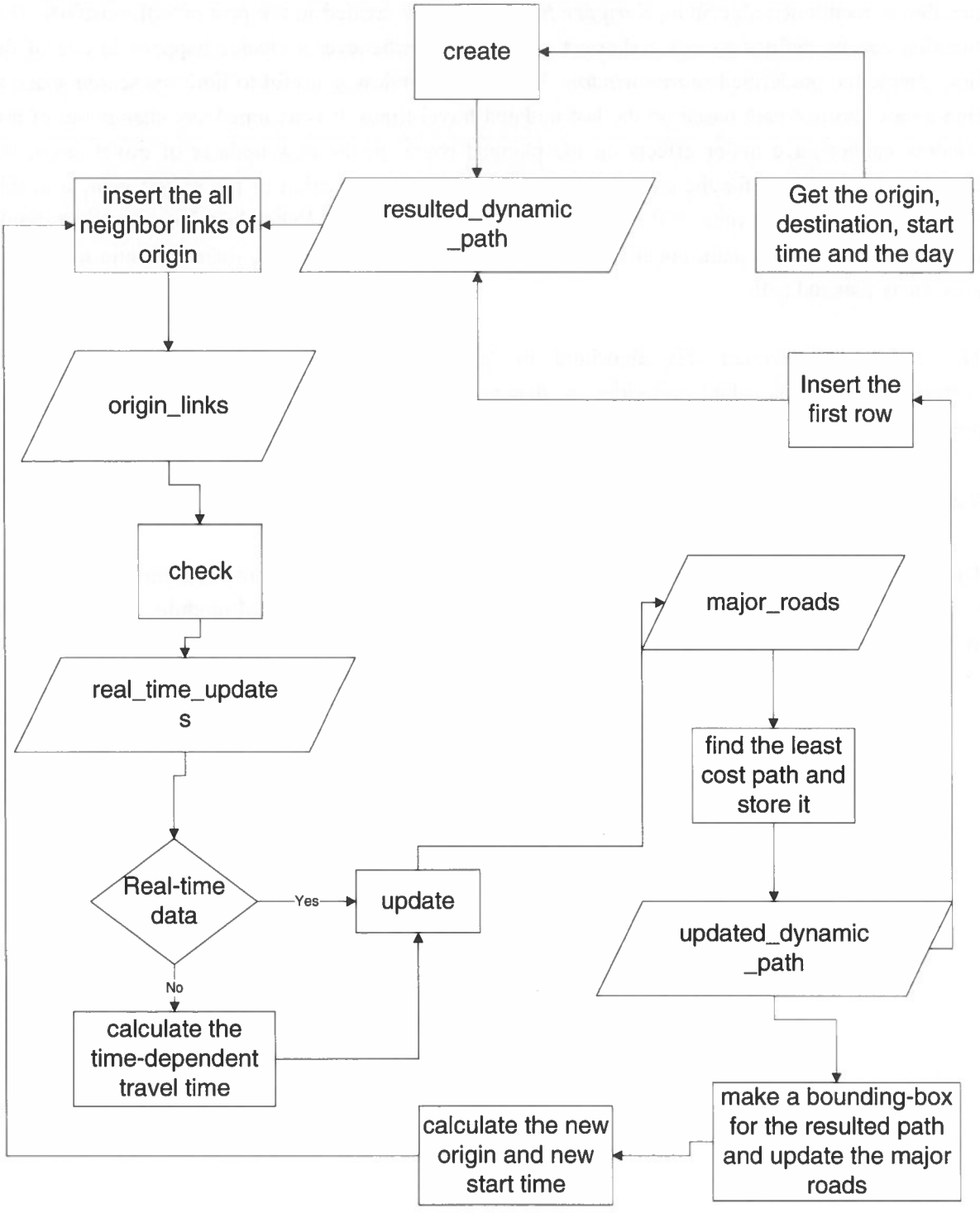

Figure 4.22. The conceptual model of the provided function

To implement the dynamicity of real world road travel-times after calculating the least-cost path in the above- mentioned algorithm, a *trigger function* can be created in the postgreSQL/postGIS. This function can be defined to call a shortest path function whenever a change happens in one of the links inside the predefined *search window*. The search window is useful to limit the search space to find a new shortest path based on the last updated travel-times. It is assumed any change out of this window cannot have major effects on the planned route. In the new updates of travel-times, the function should check for the current time and the estimated location of the routed vehicle in this time, based on the start time. If the vehicle has passed the effected links, the trigger function should not update the planned path, but if the vehicle is en-route to the effected links it should update the previously planned path.

The code to implement this algorithm in '*plpgsql*' language as a new function of the postgreSQL/postGIS, which provides a dynamic and time-dependent routing ability to the postgreSQL/postGIS, is listed in Appendix C.

4.3. Conclusion

The time-dependent (dynamic) paths and the calculated time-dependent (real-time) travel-times resulted from the new function and the trigger could be transferred to MM module and then to DM module to be used in the dynamic dispatching schedule.

5. Discussion and Conclusion

5.1. Chapter Structure

This chapter provides a general summary of the research and tries to answer the questions and objectives given in Chapter One. The advantages and disadvantages of implementing time-dependent (dynamic) routing in a spatial database are explained. In addition the importance of the proposed solution for dynamic routing in a dynamic dispatching problem is investigated.

5.2. Main Contribution and Extension of the Previous Works

The main achievement of this research work is to implement dynamic routing using free spatial databases and its predefined functions. A static routing function is adopted within an iterative function to find a solution for time-dependent (dynamic/ real-time) routing problem. The provided function uses a set of updated travel-times in each iteration to find a time-dependent (real-time) least-cost path from the origin node to a middle node in the final resulted path. It updates the initial resulted path dynamically link by link through the destination and as a final result creates a completely time-dependent (real-time) path. After generating the planned path from the origin to the destination, the proposed solution checks for coming real-time updates using a trigger function in each update in the real-time travel-times table. The trigger function acts until the vehicle arrives to the intended destination node. The result of using trigger function provides an updated and dynamic path even after planning a route.

The available functions in the pgRouting extension of the postgreSQL/postGIS for routing provide static routing solutions. These retrieve a column of cost as the predefined static values for each road link and generate a least-cost path as a solution from the origin to the destination. Although any static cost type can be defined as the values for the column cost but generally it contains the length of the related link or the average travel-time of the link. Due to the changing traffic situation in the real world road networks, have a constant travel-time for each link, which represents a unique travel-time for all days of week and all times of day is not realistic. It is obvious in most of the road links the travel-time in the midnight is not equal to the travel-time in 10 AM in the morning. This is true especially for weekdays.

Using static travel-times to route vehicles for any tow constant pair of origin and destination nodes, provide the same route independent of the time. The resulted path in this way, leads the vehicles through the congested road links if something unseal happens on that road link. Also it avoids routing the vehicles through the links that have high average travel-times stored in the database statically, even in midnight, which links are supposed to have the least travel-times (speed is usually equal to the freeflow speed). These problems cause to have an inefficient route plans.

The provided function in postgreSQL/postGIS can improve the efficiency of the planned routes with compare to the planned routes with static functions. It might not be the optimal solution but it routes the vehicles based on the real world road conditions.

5.3. Achievment of Objectives

In Chapter One the general objective was defined as the formulating of a dynamic dispatching problem and finding a solution for the dynamic routing problem within it. In Chapter Three the dynamic dispatching problem is formulated as a dynamic pick-up and delivery problem with soft time windows. The proposed solution framework is composed of three modules. This gives an overview of the whole problem and Chapter Three describes the interaction between different parts. An objective function was defined, which should be optimised to provide a minimum cost solution for the dynamic dispatching problem. A set of tasks were introduced for each module based on four cases; arrival of a new order, completion of an order, arrival of new travel-time information and order cancelation.

As a core problem of any dynamic dispatching problem, dynamic routing is assigned to be solved in the TTCM module. Because of the spatial nature of transportation data, a free and open source spatial database, postgreSQL/postGIS was chosen to handle the spatial network data, the updates from the real world travel-times, and to implement dynamic routing. Throughout the available static routing functions provided by the pgRouting extension of postgreSQL/postGIS, the A* algorithm was selected, mainly because its running time is less than Dijkstra's for the same origin and destination of the same network.

The dynamicity scenario for dynamic routing problem is defined *as frequently changing real world road travel-times*. A function was defined, which computes a least-cost path link by link and then updates it in the event of significant new travel-time arrivals. The dynamic and time-dependent path that resulted from the new defined function, routes the vehicles through the road links with the real-time travel-times.

In the next step, the least-cost paths and their associated travel-times in the TTCM could be transferred to the DM module to be used in the order assignment procedure. It is clear that in each update occurrence in TTCM the last resulting path and its related travel-times should be transferred to DM to update the schedule. Since in the defined dynamic dispatching problem the DM module should cope with the dynamicity of the real world situations to minimize the costs of the transportation company, the dynamic paths provided by the TTCM module can improve its dynamic functionality and this is the main objective, which is introduced in Chapter One.

5.4. Discussion

Although spatial databases are suitable for handling and manipulating spatial data, in practice they address several constraints in implementation. The limitations that were faced during the implementation of this research can be summarized as follows:

- Converting the format and loading large data sets into the database take time and somehow is restricted using pgAdmin III directly.
- Creating topology and populating length column to be used in the routing procedure for datasets that do not have a defined geometry column is a time consuming progress.
- The routing functions are fast in the first steps, but they become slower through the end.

- There are some limitations in programming to create new functions.
- The source codes are very complicated to be hacked and modified.

As explained in Chapter Four, the first approach employed was to create an algorithm to handle the time-dependency of the real world road networks situations. The A* algorithm was to be modified to be used instead of static algorithm in the pgRouting source code, which is able to calculate road travel-times dynamically based on the travel start-time from that node, and use the calculated time-dependent travel-times to find the least-cost paths.

The source code of the selected static function in pgRouting was investigated to identify the related code for the used static A* algorithm. The result of this investigation addressed the defined A* function in the boost library of the C++ programming language. The main problems occurred when we referred to the boost library of the C++ programming language. Different functions were called in the main body of the written code for the A* algorithm, but it was not clear which function belongs to which header included in the code. Since it seemed to be impossible to find each function called in the source code, was dedicated an alternative approach was needed.

The second approach aimed to use dynamically created link travel-times instead of using a time-dependent function. Referring to the available dataset and available static routing functions via pgRouting, a step-by-step algorithm was created to find the time-dependent link travel-times and apply them dynamically and link-by-link in the final resulted path as the *dynamic least-cost* path. While the first approach above applied just the time-dependent link travel-times, the advantage of the second approach was in applying real-time travel-time information for each link and updating the created path in new travel-time arrivals.

To implement the second approach, custom code was written in the '*plpgsql*' language of postgreSQL/postGIS. The problems faced during the implementation step and the proposed solutions for each are provided in the next section.

5.4.1 Implementing Dynamic Routing in a Spatial Database

After defining the steps for creating a dynamic and time-dependent path out of a static routing function and the provided spatial dataset, several problems occurred during implementation. The problems can be partitioned into two groups.

- The data set related problems :

  i. Most of the road links have unknown or null values for their freeflow speed in the *hsnp* table even for major roads category, which speed profiles are provided as the percentages of them.

  ii. Most of the road links have unknown or null speed profile ids in the *hsnp* table even for major roads category, which restricts the calculation of the time-dependent travel-times for those links.

- The database and programming related problems:

  i. Designing a time-dependent shortest path algorithm is a complicated task. Implementing this unfamiliar language was difficult and time-consuming. Fixing errors and keeping track of all the operations that are required for even a simple time-dependent or dynamic shortest path takes much time.

  ii. There were considerable problems with the running time of complex functions, which have formulas to be calculated.

The first and second problem of the dataset related problems is solved using the *is not null* conditioning statement in the code. The first problem is important to be solved since it creates a null value in the denominator of the function that calculates the time-dependent travel-times. The second dataset related problem is also important to be solved, because the provided id (as a speed profile id) is a part of the composite foreign key of the *hspr* table and it cannot accept a null value when the percentage of freeflow speed should be retrieved from this table.

Having solution for the database related problems is outside the scope of this research work although it is possible to use small road networks to reduce the running time of the created function in this research.

5.4.2 Further Extensions

To extend the functionality of the provided function a search window is provided to limit the search space for finding the least-cost paths. Defining a search window for the provided trigger function can extend it to update the planned path only when a significant change is reported within this window. The criteria to define this search window should consider the fact that happening something unforeseen in a road link does not affect only the travel-time of that link but also affects the neighbour links travel-times and also the travel-times of other links, which are within a specific distance of the congested link. Finding the best way to define this search window is not an easy work since the precise domain of impact cannot be defined simply. One way might be to define a search window as the common area between the results of the forward least-cost path search and the backward least-cost path search, and it should be further investigated.

The proposed dynamic solution might not provide an optimal solution as the least-cost path. The reason can be detected in calculating the least-cost path link-by-link. The proposed solution determines the first link of the final resulted path between an origin and a destination in the first iteration and does not include it in the other steps. While it changes the origin node in the next iterations the same situation is true for the other assigned links which are calculated based on the time-dependent travel-times. It is possible to have another path between the same origin and the same destination, which instead of one of the already assigned links uses another link and provides a path with the cost less than the proposed one by the function. Finding a solution to optimise the provided function can ensure the drivers about the efficiency of the planned route.

It is important to mention that the created function do not get lost in the network while it is running for the next steps after step one. Since it stores the links one by one as the final solution links, and searches for the least-cost path from the current node, so in the middle of the progress the before determined links are stored and the function is searching for the rest, and this search is same as having the current node as an initial origin.

In the provided solution for dynamic routing the road network is assumed to be a FIFO network. Finding the impacts of using a none-FIFO road network instead, can be an interesting problem to be solved.

Also in the case of having large road networks with lots of local and non-major links, it seems wise to apply some kind of *hierarchical* procedure: to make sure function is able to route the vehicles through the major roads as much as possible (the vehicles could be routed using the local and none-major roads to the nearest entrance point to the major roads network and then route through the major roads to arrive to the nearest exit point of this network to the destination node) because of two reasons. First the speed in the major roads for vehicles is more than local and none-major roads if the time does not overlap with rush hours. Second in this case, the majority of the search space can be restricted to only major roads and can result in shorter running times.

5.5 The Dispatching Problem and Dynamic Routing

We started with a definition for a dispatching problem in the Chapter Three and have solved the problem of dynamic and time-dependent routing as the responsibility of the TTCM module. In this section the impact of the provided dynamic least-cost paths and their associated travel-times in the orders assignment procedure in the DM module is investigated. As explained in Chapter Three, MM receives the orders pick-up and delivery locations, and latest pick-up time as the part of request information placed by the customer. This information is delivered to the DM module to assign the most appropriate vehicles to serve the orders in a way that minimizes the costs of the transportation company. To do so, this module should assign a daily dynamic route for each vehicle. To determine dynamic routes and optimum schedules (that are subject to change in new order placement or new information arrival), the DM needs dynamic and time-dependent travel-times, which reflect the real world condition. The required information for updated travel-times is transferred from the TTCM module via the MM. The transferred travel-time information influences the schedule assignment. So travel-time information can be viewed as a control variable. The planned schedule in the DM can be transferred to the MM to give messages to the customers and drivers and to route the assigned vehicles through the least-cost paths calculated by the TTCM.

It is assumed the drivers can be informed about the new pick-up location just after the before assigned order completion, so there is a flexibility to change the schedule for each vehicle before starting its journey. In addition it is assumed that the en-route vehicles cannot be diverted from their routes to serve another order after informing the driver. When a vehicle is en-route to a pick-up or delivery location, the driver only can be informed about the alternative paths, if there is a significant change in the planned routes travel-times are reported by the TTCM.

5.5.1. Possible Limitations and Uncertainties

In the dynamic dispatching problem formulation several constraints are applied to make the problem simpler to be solved. In the reality these constraints might cause different limitations. Also the calculated travel-times force some uncertainties to the dynamic schedule due to providing none-optimal least-cost paths. The constraints, which cause to deviate from the reality, are provided below:

i. The problem is formulated as a *single depot* problem while in the reality the number of depots for transportation companies is more than one.

ii. It is assumed that each vehicle can serve *only one* order in the same time. Having such a constraint results to include unused vehicle capacities in the problem, this is not beneficial cost-wise.

iii. The transportation company *cannot rent vehicles* even if it makes profit.

iv. It does not provide any solution in vehicle *breakdowns*.

Excluding the mentioned constraints and uncertainties from the provided solution can help to improve the reliability of results.

5.6 Conclusion

Understanding the relationship between dynamic dispatching and dynamic routing is essential to solve the problem of dynamic dispatching. The dynamic Pick-up and delivery problem is formulated, and an objective function introduced which should be minimized to give an optimum schedule to serve customers. Three modules are presented each with defined responsibilities to provide a solution for the formulated problem.

Dynamic routing is defined as the core problem of dynamic dispatching problem and two different approaches are provided to solve it. The first approach presents a time-dependent A* algorithm which is adapted to be used with Tele Atlas speed profiles and should be replaced with the existing (static) A* algorithm in the source code of the pgRouting. The second approach presents a step-by-step algorithm to find a dynamic and time-dependent path using the static A* algorithm iteratively. An implementation of this algorithm is given as a function of postgreSQL/postGIS using a static least-cost path function of pgRouting. The function is tested using the Tele Atlas spatial data set and a generated table as a real-time updates table. Comparing routes showed the presented function in the postgreSQL/postGIS provides different travel-times for different times of a day and different days of a week. This result is the one which could be expected, having different travel-times/routes based on the different start times to travel. Although the provided function might not generate an optimum path to be used in the dynamic dispatching problem, but provides improved solutions over purely static ones. This function includes the real world last updated travel-times in the final solution.

References:

Abler, R. F. (1987), The National Science Foundation National Center for Geographic Information and Analysis. International Journal for Geographical Information Systems, vol. 1(4), pp. 303-326.

Ahner, D, K, & A. H. Buss & J. Ruck (2006), Assignment scheduling capability for unmanned aerial vehicles – a discrete event simulation with optimisation in the loop approach to solving a scheduling problem. Proceedings of the Winter Simulation Conference.
http://www.informs-sim.org/wsc06papers/172.pdf

Aho, A. v. & J. E. Hopcroft & J. D. Ullman (1974), The Design and Analysis of Computer Algorithms. Addison Wesley Longman

Alexander, J. R. & C. J. Swanepoel (Accessed in May 2009), Shortest time route through a network with time dependent links. http://www.scs.org/confernc/hsc/hsc02/hsc/papers/hsc056.pdf

Balev, S. & F. Guinand & Y. Pigne (Accessed in May 2009), Maintaining Shortest Paths in Dynamic Graphs. http://www-lih.univ-lehavre.fr/~pigne/papers/BalevGuinandPigneNCP07.pdf

Bast, H. & S. Funke & D. Matijevic (Accessed in April 2009), Transit ultrafast shortest-path queries with linear-time pre-processing. http://www.mpi-inf.mpg.de/~dmatijev/papers/DIMACS06.pdf

Bauer, R. & D. Delling (Accessed in April 2009), SHARC: Fast and Robust Unidirectional Routing. http://www.siam.org/proceedings/alenex/2008/alx08_02bauerr.pdf

Bertsimas, D. J. & G. van Ryzin (1991), A stochastic and dynamic vehicle routing problem in the Euclidean plane. Oper. Res, vol. 39, pp. 601–615

Bertsimas, D. J. & D. Simchi-Levi (1996), A new generation of vehicle routing research: Robust algorithms, addressing uncertainty. Operations Research, Vol. 44, Part. 2, pp. 286–304

Branchini, R.M. & V.A. Armentano & A. Lokketangen (2009), Adaptive granular local search heuristic for a dynamic vehicle routing problem. Computers & Operations Research, Vol. 36(11), pp. 2955-2968

Brakatsoulas, S. & D. Pfoser & N. Tryfona (2005), Practical Data Management Techniques for Vehicle Tracking Data. ICDE, pp. 324-325

Brakatsoulas, S. & D. Pfoser & R. Salas & C. Wenk (2005), On map-matching vehicle tracking data. Proc. 131st VLDB Conference, pp. 853-864

Brakatsoulas, S. & D. Pfoser & C. Wenk (2007), Creating a data mart for floating car data.COOP-CT-2006-032823, Co-operative research project, FP6. http://www.trackandtrade.org

Brodie, M. L. & J. Mylopoulos & J. W. Schmidt (1984), On conceptual modelling. In: Topics in Information Systems. New York: Springer-Verlag.

Chabini, I. (1997), A New Shortest Paths Algorithm for Discrete Dynamic Networks. Proceeding of 8th IFAC Symposium on Transport Systems, pp. 551–556

Chabini, I. (1998), Discrete Dynamic Shortest Path Problems in Transportation Applications: Complexity and Algorithms with Optimal Run Time. Transportation Research Record, vol. 1645, pp. 170–175

Chabini, I. & S. Ganugapati (2002), Parallel algorithms for dynamic shortest path problems. Intl. Trans. in Op. Res., Vol. 9, pp. 279-302

Chang, T-S. & Y-W. Wan & W. Tsang OOI (2008), A stochastic dynamic travelling salesman problem with hard time windows. European Journal of Operational Research, Vol.XXX, pp. xxx-xxx (ARTICLE IN PRESS)

Chon, H.D. & D. Agrawal & A.E. Abbadi (2003), FATES: Finding A Time dependent Shortest path. M.-S. Chen et al. (Eds.): MDM, LNCS 2574, pp. 165–180, Springer-Verlag Berlin Heidelberg

Cooke, K. L. & E. Halsey (1966), The shortest route through a network with time-dependent internodal transit. J. Math. Anal. Appl, Vol. 14, pp. 493–498

Coslovich, L. & R. Pesenti & W. Ukovich (2006), A two-phase insertion technique of unexpected customers for a dynamic dial-a-ride problem. European Journal of Operational Research, vol. 175 , pp. 1605–1615

Couclelis, H. & N. Gale (1986), Space and Spaces. Geografiska Annaler, vol. 68, pp. 1-12

Crainic, T. G. & M. Gendreau (2004), Intelligent fright transportation systems: Assessment and the contribution of operational research. Department informatique et recherche opérationnelle and Centre for Research on Transportation Université de Montréal. http://www.crt.umontreal.ca/~theo/mba8L91/crainic-gendreau.pdf

Daganzo, C. F. (1998), Reversibility of the time-dependent shortest path problem. Research report UCB-ITS-RR-98-14, Department of Civil Engineering and Institute of Transportation Studies, University of California. http://www.its.berkeley.edu/publications/UCB/98/RR/UCB-ITS-RR-98-14.pdf

Dean, B. C. (1999), Continuous-Time Dynamic Shortest Path. Massachusetts Institute of Technology. http://www.cs.clemson.edu/~bcdean/bdean_masters_thesis.pdf

Dean, B. C. (Accessed in June 2009), Shortest Paths in FIFO Time-Dependent Networks: Theory and Algorithms. Massachusetts Institute of Technology. http://www.cs.clemson.edu/~bcdean/tdsp.pdf

Dean, B. C. (2004), Algorithms for Minimum-Cost Paths in Time-Dependent Networks with Waiting Policies. Networks, vol. 44(1), pp. 41-46, Cambridge, USA

Delling, D. & G. Nannicini (2008), Core Routing on Dynamic Time-Dependent Road Networks.http://www.optimization-online.org/DB_FILE/2008/12/2164.pdf

Delling, D. (2008), Time-dependent SHARC-routing. D. Halperin and K. Mehlhorn (Eds.): ESA 2008, LNCS 5193, pp. 332–343. Springer-Verlag Berlin Heidelberg

Du, T. & F. k. Wang & P-L. Lu (2007), A real-time vehicle-dispatching system for consolidating milk runs. Transportation Research Part E, Vol. 43, pp. 565-577

Erwig, M. & R.H. Guting & M. Schneider & M. Vazirgiannis (1999), Spatio-temporal data types: An approach to modelling and querying moving objects in databases. GeoInformatica, vol. 3, pp. 269–296

ESRI copyright.http://www.esri.com/legal/copyright-trademarks.htmlGoldberg, A.V. & C (Accessed in May 2009)

ESRI, GIS and mapping software.http://www.esri.com/industries/transport/business/intermodal.html(Accessed in April 2009)

Fayyad, U. & G. Piatetsky-Shapiro & P. Smyth (1996), From data mining to knowledge discovery: an overview. Fayyad U., Piatetsky-Shapiro G., Smyth P., Uthurusamy R., Editors, Advances in Knowledge Discovery and Data Mining, AAAI/MIT Press, pp. 83-115

Firantas, R. & M. Jusevičius (2006), Automated taxi request, dispatch, and routing: conceptual design. IT University of Copenhagen Rued Langgaards Vej 7 DK-2300 Copenhagen S. http://www.itu.dk/people/martynas/LMA/Taxi.pdf

Fleischmann, B. & E. Sandvoss (2003), Vehicle routing for dynamic order arrivals and dynamic travel times.http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.4.4757

Fleischmann, B. & E. Sandvoß & S. Gnutzmann (2004), Dynamic vehicle routing based on on-line traffic information. Transportation Science, vol. 38(4), pp. 420–433

Fletcher, D. (2000), Geographic information systems for transportation: a look forward. In: Transportation in the new millennium. State of the art and future directions. Washington, DC: Transportation Research Board. http://onlinepubs.trb.org/onlinepubs/millennium/00047.pdf

Floyd, R. W. (1962), Algorithm 97: Shortest path. Commun. ACM, Vol. 5, No. 6

Fu, L. (2001), An adaptive routing algorithm for in-vehicle route guidance systems with real-time information. Transportation Research Part B, vol. 35, pp. 749-765

Gang, Y. & Y. Jian (1998), On the robust shortest path problem. Computers & Operations Research, vol. 25, pp. 457-468

Gendreau, M. & F. Guertin & J-Y. Potvin & R. Seguin (2006), Neighbourhood search heuristics for a dynamic vehicle dispatching problem with pick-ups and deliveries. Transportation Research Part C, Vol. 14, pp. 157-174

Gendreau, M. & J-Y. Potvin (1998), Dynamic vehicle routing and dispatching. Crainic, T., Laporte, G. (Eds.), Fleet Management and Logistics, Kluwer, New York, pp. 115–126

Ghiani, G. & E. Manni & A. Quaranta & C. Triki (2009), Anticipatory algorithms for same-day courier dispatching. Transportation Research Part E, Vol. 45, pp. 96–106

Ghiani, G. & F. Guerriero & G. Laporte & R. Mossman (2003), Real-time vehicle routing: Solution concepts, algorithms and parallel computing strategies. European Journal of Operational Research, Vol. 151, Part. 1, pp. 1–11

Goldberg, A.V. & G. Harrelson (2005), Computing the shortest path: A* search meets graph theory. Proc. SODA 2005, pp. 156–165

Gonzalez, A. (2006), Trends and Predictions in the Transportation Management Systems Market.ARC Advisory Group Report

Gonzalez, M. & M. Machin (2003), In Uruguay, Milk Transportation Uses New GIS and Logistics Tool.http://www.esri.com/news/arcnews/summer03articles/in-uruguay.html

Goodchild, M. F. (Accessed in May 2009), Geographic information systems and disaggregate transportation modelling. National Center for Geographic Information and Analysis, and Department of Geography University of California Santa Barbara, CA 93106-4060

Greenwood, D. & C. Dannegger & K. Dorer (Accessed in May 2009),Dynamic Dispatching and Transport Optimization – Real-World Experience with Perspectives on Pervasive Technology Integration. http://www.whitestein.com/library/WhitesteinTechnologies_Paper_HICSS2008.pdf

Haghani, A. & S. Jung (2005), A dynamic vehicle routing problem with time-dependent travel times. Computers & Operations Research, vol. 32, pp. 2959–2986

Halpern, H. J.(1977), Shortest route with time dependent length of edges and limited delay possibilities in nodes. Operations Research, Vol. 21, pp. 117–124

Hart, P. & N. Nilsson & B. Raphael(1968), A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions on System Sciences and Cybernetics, SSC-4(2), pp. 100-107

http://en.wikipedia.org/wiki/A*_search_algorithm (Accessed in May 2009)

http://en.wikipedia.org/wiki/Dijkstra's_algorithm (Accessed in May 2009)

http://en.wikipedia.org/wiki/Dynamic_programming (Accessed in November 2009)

http://pgrouting.postlbs.org/ (Accessed in September 2009)

Huang, B. & R. L. Cheu & Y. S. Liew (2004), GIS and genetic algorithms for HazMat route planning with security considerations. International Journal of Geographical Information Science, vol. 18, pp. 769–787

Huang, B. & X. Pan (2007), GIS coupled with traffic simulation and optimization for incident response. Computers, Environment and Urban Systems, Vol. 31, pp. 116-132

Ichoua, S. & M. Gendreau & J-l. Potvin (2003), Vehicle dispatching with time-dependent travel times. European Journal of Operational Research, Vol. 144, pp. 379–396

Jayakrishnan, R. & W. K. Tsai & J. Adler & J. S. Oh (1997) Event-based ATIS: Practical Implementation and Evaluation of Optimized Strategies. Event-based ATIS: Practical Implementation and Evaluation of Optimized Strategies, University of California, Irvine. http://www.its.uci.edu/its/publications/papers/CTSS/UCI-ITS-TS-WP-01-1.pdf

Jonker, R. & A. Volgenant (1987), A shortest augmenting path algorithm for dense and sparse linear assignment problems. Computing, vol. 38, pp. 325-340

Kalinowski, N. & C. Wenk (2006), Dynamic Routing. Technical report CS-TR-2007-005, Department of Computer Science, University of Texas at San Antonio. http://www.cs.utsa.edu/dmz/techrep/2007/CS-TR-2007-005.pdf

Kaufmann, D. E. & R. L. Smith (1993), Fastest paths in time-dependent networks for intelligent vehicle-highway systems application. IVHS Journal, vol. 1, pp. 1–11

Kelpin, R. & A. Sohr & M. Umlauft & P. Brosch & G. Schimon & D. Pfoser(2007),Creating a Data Mart for Floating Car Data. Copyright © 2007 TRACK&TRADE consortium, http://www.trackandtrade.org

Lee, W-H. & S-S. Tseng & S-H. Tsai(2009),A knowledge based real-time travel time prediction system for urban network. Expert Systems with Applications, vol. 36, pp. 4239–4247

Li, J-Q. & P .B. Mirchandani & D. Borenstein (2009), Real-time vehicle rerouting problems with time windows. European Journal of Operational Research, Vol. 194, pp. 711–727

MacEachren, A.M. & M. Wachowiez & R. Edsall & D. Huang (1999), Constructing knowledge from multivariate spatiotemporal data: integrating geographical visualization with knowledge discovery in database methods. International Journal of Geographical Information Science, vol. 13, pp. 311–334

Mahmassani, H.S. & X. Zhou & C. Cheng Lu (2005), Toll Pricing and Heterogeneous users. Transportation Research Record, No. 1923, pp. 28-36

Mark, D. M. & A. U. Frank (1989), Concepts of Space and Spatial Language. 9th International Symposium on Computer-Assisted Cartography, Autocrat 9, Baltimore, Maryland, pp. 538-556

Maue, J. & P. Sanders & D. Matijevic (Accessed in March 2009),Goal Directed Shortest Path Queries Using Precomputed Cluster Distances. http://www.mpi-inf.mpg.de/~jensmaue/MaueSandersMatijevic2006.pdf

Miller, H.J. & Y. H. Wu & M. Hung (1999), GIS-based dynamic traffic congestion modelling to support time critical logistics. Copyright IEEE. Proceedings of the Hawai'i International Conference on System Science, Maui, Hawaii. http://www.geog.utah.edu/~hmiller/papers/hicss_web.pdf

Mitrovic - Minic, S. & G. Laporte (2004), Waiting strategies for the dynamic pickup and delivery problem with time windows. Transportation Research Part B, vol. 38, pp. 635–655

MultiNet® Shapefile 4.4 vs 4.3.2.2 Format Specifications, 2008, Tele Atlas

Nannicini, G. & L. Liberti (2008), Shortest paths on dynamic graphs. http://www.lix.polytechnique.fr/~liberti/sppsurvey.pdf

Nual, D. & M. Wlodyka & N.P. Nual (Accessed in May 2009), Real-Time Traffic Information Production and Presentation Using GIS-Based Maps.
http://proceedings.esri.com/library/userconf/proc03/p1208.pdf

Orda, A. & R. Rom (1990), Shortest-path and minimum-delay algorithms in networks with time-dependent edge length. Journal of the ACM, vol. 37(3), pp. 607-625

Patrushev, A.(2007), Shortest path search in real road networks with pgRouting. FOSS4G2007 Presentation.
http://www.comp.dit.ie/pbrowne/Spatial%20Databases%20SDEV4005/Spatialdblecture3.PPT

Pedersen ,J. D. (1998),Robust Communications for High Band width Real-Time Systems.http://www.ri.cmu.edu/pub_files/pub1/pedersen_jorgen_1998_1/pedersen_jorgen_1998_1.pdf

Pfoser, D. & S. Brakatsoulas & P. Brosch & M. Umlauft & N. Tryfona & G. Tsironis (2008),Dynamic travel time provision for road networks. 16th ACM SIGSPATIAL International Symposium on Advances in Geographic Information Systems, Article No.68, ACM-GIS, Irvine, California, USA, ISBN:978-1-60558-323-5

Pfoser, D. & C. S. Jensen (2003), Indexing of Network Constrained Moving Objects. GIS'03, New Orleans, ouisiana, USA. http://dke.cti.gr/pubs/confs/acmgis03.pdf

Pfoser, D. & C. S. Jensen & Y. Theodoridis (2000), Novel Approaches to the Indexing of Moving Object Trajectories. Proceedings of the Int. Conf. on Very Large Data Bases, pp. 395–406

Pfoser, D. & N. Tryfona & A. Voisard (2006), Dynamic Travel Time Maps - Enabling Efficient Navigation. Proc. 18th SSDBM conf., pp. 369–378

Potvin, J. Y. & Y. Xu & I. Benyahia (2006), Vehicle routing and scheduling with dynamic travel times. Computers & Operations Research, vol. 33, pp. 1129–1137

Powell, W. B. & P. Jaillet & A. Odoni (1995), Stochastic and dynamic networks and routing. In: Ball, M.O. & T. L. Magnanti & C. L. Monma & G. L. Nemhauser (eds.), Handbooks in Operations Research and Management Science. Network Routing, Volume Eight. Amsterdam (North-Holland): Elsevier, pp. 141–296.

Psaraftis, H.N. (1995), Dynamic vehicle routing: Status and prospects. Annals of Operations Research 61, pp. 143–164

Sivaram, C. M. S. L & M. N. Kulkarni (Accessed in April 2009), GPS-GIS integrated systems for transportation engineering.GIS development, http://www.gisdevelopment.net/technology/gps/techgp0008.htm

Spira, P. M. & A. Pan (1975),On Finding and Updating Shortest Paths and Spanning Trees .FOCS, Iowa City, Iowa, pp. 82-84

Spivey, M. & W. B. Powell (2004), The dynamic assignment problem. Transportation Science, vol. 38, pp. 399–419

Tele Atlas Speed Profiles user guide, 2008, Tele Atlas

Thill, J.-C. (2000), Geographic information systems for transportation in perspective. Transportation Research Part C, vol. 8, pp. 3-12

Timpf, S. (1992), Conceptual Modelling of Highway Navigation. Master of Science thesis, The University of Maine. http://www.geo.unizh.ch/~timpf/docs/Timpf_thesisMSc.pdf

TNT Logistics, "Transport management Services", Grids:"The TNT Logistics Benelux magazine for logistics & supply chain excellence", Fall 2005

Vazirgiannis, M. & O. Wolfson (2001), A Spatiotemporal Model and Language for Moving Objects on Road Networks. Int. Symposium on Spatial and Temporal Databases, pp. 20–35

Veit Batez ,G. & D. Delling & P. Sanders & C. Vetter(2008), Time-dependent contraction hierarchies.http://www.siam.org/proceedings/alenex/2009/alx09_010_batzg.pdf

Venigalla, M. (2005), GIS in transportation. CEIE 410/510 lecture notes

Wahle, J. & O. Annen & CH. Schuster & L. Neubert & M. Schreckenberg (2001), A dynamic route guidance system based on real traffic data. European Journal of Operational Research, vol. 131, pp. 302-308

Wenk, C. & R. Salas & D. Pfoser (2006), Addressing the Need for Map-Matching Speed: Localizing Global Curve-Matching Algorithms. Proc. 18th SSDBM conf., pp. 379-388

Xiao, J. & Y. Huang & L. Helen H. (2007), A probability distribution estimation based method for dynamic optimization. AIChE journal, vol. 53, no 7, pp. 1805-1816

Yang, J. & P. Jaillet & H. Mahmassani (2004), Real-Time Multivehicle Truckload Pickup and development, Delivery Problems. Transportation science, Vol. 38, No. 2, pp. 135–148

Yoon, J. & B. Nobel & m. Liu (2007), Surface Street Traffic Estimation. San Juan, Puerto Rico, USA.
Copyright 2007 ACM 978-1-59593-614-1/07/0006

Zhan, F. B. & C. E. Noon (1998), Shortest Path Algorithms: An Evaluation using Real Road Networks. Transportation Science, Vol. 32, No. 1

Zhao, L. & T. Oshima & H. Nagamochi (Accessed in April 2009), A* Algorithm for the time-dependent shortest path problem. http://www-or.amp.i.kyoto-u.ac.jp/~liang/research/waac08.pdf

Appendix A; Steps to Execute Dijkstra algorithm:

Using the shp2pgsql -s SRID -i -I "C:\File name.shp" File name > "C:\File name.sql" command the .shp files were converted to the .sql files. The created SQL files were loaded to the database using command prompt window. After loading data to the database the following steps were followed:
(These steps are adopted from http://pgrouting.postlbs.org/ http://pgrouting.postlbs.org/ (Accessed in September 2009)):

Step 1:

```
ALTER TABLE Table-name ADD COLUMN source integer;
ALTER TABLE Table-name ADD COLUMN target integer;
ALTER TABLE Table-name ADD COLUMN length double precision;
```

Step 2:

```
SELECT assign_vertex_id('table name', 0.00001, 'the_geom', 'gid');
UPDATE table name SET length = length(the_geom);
```

Step 3:

```
CREATE INDEX source_idx ON table name(source);
CREATE INDEX target_idx ON table name(target);
CREATE INDEX geom_idx ON t USING GIST(the_geom GIST_GEOMETRY_OPS);
```

Step 4:

```
DROP TABLE IF EXISTS dijsktra_result;
CREATE TABLE dijsktra_result(gid int4) with oids;
SELECT AddGeometryColumn('dijsktra_result', 'the_geom', '4326', 'MULTILINESTRING',
2);
INSERT INTO dijsktra_result(the_geom)
SELECT the_geom FROM dijkstra_sp('table name', origin, destination);
```

Appendix B; Steps to Execute A* algorithm:

Using the shp2pgsql -s SRID -i -I "C:\File name.shp" File name > "C:\File name.sql" command the .shp files were converted to the .sql files. The created SQL files were loaded to the database using command prompt window. After loading data to the database the following steps were followed:
(These steps are adopted from http://pgrouting.postlbs.org/ http://pgrouting.postlbs.org/ (Accessed in September 2009)):

Step 1:

```
ALTER TABLE Table-name ADD COLUMN x1 double precision;
ALTER TABLE Table-name ADD COLUMN y1 double precision;
ALTER TABLE Table-name ADD COLUMN x2 double precision;
ALTER TABLE Table-name ADD COLUMN y2 double precision;
ALTER TABLE Table-name ADD COLUMN length double precision;
```

Step 2:

```
UPDATE Table-name SET x1=x(startpoint(the_geom));
UPDATE Table-name SET y1=y(startpoint(the_geom));
UPDATE Table-name SET x2=x(endpoint(the_geom));
UPDATE Table-name SET y2=y(endpoint(the_geom));
UPDATE Table-name SET Length=Length (the_geom);
```

Step 3:

```
DROP TABLE IF EXISTS astar_result;
CREATE TABLE astar_result(gid int4) with oids;
```

Step 4:

```
SELECT AddGeometryColumn('astar_result', 'the_geom', '4326', 'MULTILINESTRING', 2);
```

Step 5:

```
INSERT INTO astar_result(the_geom)
        SELECT the_geom FROM astar_sp_delta('table name', origin, destination,0.1);
```

Appendix C; The created function in postgreSQL/postGIS for dynamic routing:

```
drop function if exists dynamic_routing1111(integer,integer,double precision,text);
create or replace function dynamic_routing1111(origin integer,destination integer,start_time double precision,the_day text) returns setof major_roads_dec11 as $$

declare
the_time double precision;
relative_sp numeric;
new_travel_time double precision;
new_origin integer;
orlin record;
marod record;
new_profile numeric;
timeslot integer;
min_x1 double precision;
max_x1 double precision;
min_y1 double precision;
max_y1 double precision;

begin
new_origin:=$1;
the_time:=$3;
timeslot:=$3;
drop table if exists resulted_dynamic_path;
create table resulted_dynamic_path(source integer,gid integer,traveltime double precision) with oids;

while new_origin!= $2
loop

drop table if exists origin_links;
create table origin_links(gid integer,id bigint,source integer,target integer,meters double precision, length double precision);
insert into origin_links(gid,id,source,target,meters,length)
select gid,id,source,target,meters,length
from major_roads_dec11
where major_roads_dec11.target=new_origin or major_roads_dec11.source= new_origin;
```

```
for orlin in select * from origin_links order by gid
loop

if (

select real_time_updates.length
from real_time_updates
where(orlin.gid=real_time_updates.gid and real_time_updates.length is not null))is not null
then
update major_roads_dec11
set length= real_time_updates.length
from real_time_updates
where orlin.gid=major_roads_dec11.gid and orlin.gid=real_time_updates.gid;

else

if $4='Sunday'
then
new_profile:= (select profile_1
from hsnp_11
where network_id= orlin.id);

elsif $4='Monday'
then
new_profile:=(select  profile_2
from hsnp_11
where network_id= orlin.id);

elsif $4='Tuesday'
then
 new_profile:=(select profile_3
from hsnp_11
where network_id= orlin.id);

elsif $4='Wednesday'
then
 new_profile:=(select profile_4
from hsnp_11
where network_id= orlin.id);

elsif $4='Thursday'
```

```
then
 new_profile:=(select profile_5
from hsnp_11
where network_id= orlin.id);

elsif $4='Friday'
then
  new_profile:=(select profile_6
from hsnp_11
where network_id= orlin.id);

elsif $4='Saturday'
then
 new_profile:=(select profile_7
from hsnp_11
where network_id= orlin.id);

end if;

if new_profile is not null
then

timeslot:=(select time_slot from hspr where profile_id= new_profile and the_time>time_slot and
the_time<time_slot+300);

relative_sp:=(select rel_sp
from hspr
where profile_id= new_profile AND time_slot= timeslot);
new_travel_time:= ((select meters from major_roads_dec11 where orlin.gid= major_roads_dec11.gid
))/( 10*(select spfreeflow from hsnp_11 where orlin.id=hsnp_11.network_id)*relative_sp);
end if;

if new_travel_time is not null
then
update major_roads_dec11
set length= (3600*new_travel_time)
where major_roads_dec11.gid=orlin.gid;
end if;
end if;
end loop;
```

```
drop table if exists updated_dynamic_path;
create table updated_dynamic_path(gid integer,id_gid integer,traveltime double precision) with oids;
insert into updated_dynamic_path  (select * from shortest_path_astar('
                    select gid as id,
                                source::integer,
                                target::integer,
                                length::double precision as cost,
                                x1, y1, x2, y2
                               from major_roads_dec11',
                new_origin, $2, false, false));

drop table if exists dynamic_astar_geom;
create table dynamic_astar_geom(gid integer,x1 double precision,y1 double precision) ;

insert into dynamic_astar_geom(gid,x1,y1)
select major_roads_dec11.gid,major_roads_dec11.x1,major_roads_dec11.y1
from major_roads_dec11 ,updated_dynamic_path
where major_roads_dec11.gid= updated_dynamic_path.id_gid;

min_x1:=(select MIN(dynamic_astar_geom.x1)
from dynamic_astar_geom);
max_x1:=( select MAX(dynamic_astar_geom.x1)
from dynamic_astar_geom);
min_y1:=( select MIN(dynamic_astar_geom.y1)
from dynamic_astar_geom);
max_y1:= (select MAX(dynamic_astar_geom.y1)
from dynamic_astar_geom);

delete from major_roads_dec11
where major_roads_dec11.x1< (min_x1)-0.1 or major_roads_dec11.x1> (max_x1)+0.1 or
major_roads_dec11.y1< (min_y1)-0.1 or major_roads_dec11.y1> (max_y1)+0.1;



insert into resulted_dynamic_path  (select* from updated_dynamic_path limit 1);
delete from updated_dynamic_path where gid=new_origin;

drop table if exists second_row_dynamic_path;
create table second_row_dynamic_path(source integer,gid integer,traveltime double precision);
insert into second_row_dynamic_path  (select* from updated_dynamic_path limit 1);
```

```
new_origin:=(select source from second_row_dynamic_path);

the_time:= the_time+(select traveltime from second_row_dynamic_path);

end loop;
insert into resulted_dynamic_path  (select * from updated_dynamic_path limit 1);

return;


end;
$$
language'plpgsql'volatile strict
```