AN APPROACH TO TIME-DEPENDENCE AND RELIABILITY IN DYNAMIC ROUTE GUIDANCE

Ioannis Kaparias
Centre for Transport Studies
Imperial College London
Skempton Building
South Kensington Campus
London SW7 2BU, UK
Tel: +44 20 75945952
Fax: +44 20 75946102
Email: ioannis.kaparias@imperial.ac.uk

Michael G.H. Bell
Centre for Transport Studies
Imperial College London
Skempton Building
South Kensington Campus
London SW7 2BU, UK
Tel: +44 20 75946091
Fax: +44 20 75946102
Email: m.g.h.bell@imperial.ac.uk

Klaus Bogenberger
BMW Group
Department of Corporate Quality – Product Monitoring
AQ-11
Munich 80788, Germany
Tel: +49 89 38230764
Fax: +49 89 38230764
Email: klaus.bogenberger@bmw.de

Yanyan Chen
Transportation Research Center
Beijing University of Technology
No. 100 Ping Le Yuan
Beijing 100022, PR China
Tel: +86 10 67391680
Fax: +86 10 67391509
Email: cdyan@bjut.edu.cn

(7350 + 2Figures · 250 = 7850 words)

This paper was submitted to the Transportation Research Board for publication in "Transportation Research Record: Journal of the Transportation Research Board".

8 March 2007

AN APPROACH TO TIME-DEPENDENCE AND RELIABILITY IN DYNAMIC ROUTE GUIDANCE

Ioannis Kaparias[1], Michael G.H. Bell[1], Klaus Bogenberger[2], Yanyan Chen[3]

[1] Centre for Transport Studies, Imperial College London, UK
[2] Department of Corporate Quality – Product Monitoring, BMW Group, Munich, Germany
[3] Transportation Research Center, Beijing University of Technology, Beijing, PR China

ABSTRACT

This paper presents a methodology, in order to increase the reliability of the route suggestions in route guidance systems. Based on the A* path finding algorithm and Chen's link penalty method, the procedure involves penalising links with a high risk of being congested and obtaining a set of reliable route suggestions. Time-dependence of travel times is considered by adapting the Flow Speed Model technique accordingly. Modifications to the structure of the path finding algorithms are also made, so as to account for real road network features. Finally, experiments using simulated travel time and reliability data are carried out on a road network and the results are discussed.

1. INTRODUCTION

In-vehicle information systems are a rapidly growing market, with new systems being continuously developed and existing ones becoming more sophisticated. Having started off from luxury makes and models, they are now gradually spreading through the entire vehicle fleet, offering more and more features aiming to provide the driver with useful information regarding his/her trip. An important feature offered by more advanced systems is Dynamic Route Guidance (DRG).

The objective of DRG is to provide participating drivers with a fast route to their destination, taking current and anticipated traffic conditions into consideration. In accordance to that, Reliable Dynamic Route Guidance (RDRG) is the feature whose objective it is to provide the drivers with a route, which is not only fast, but is also reliable. As reliability is defined as the probability of not encountering delays, RDRG ensures that potential sources of delay in the road network are avoided as much as possible. These can be for example left turns in Continental Europe and North America (right turns in the UK) where the turning vehicle usually has to cross one or more opposing streams, resulting in increased waiting times at junctions, or roads that are frequently congested and are therefore likely to be congested at the time of the user's trip.

RDRG can be provided to the driver by two system architectures, namely Autonomous Reliable Dynamic Route Guidance (ARDRG) and Supported Reliable Dynamic Route Guidance (SRDRG). In ARDRG, which has a decentralised system architecture, a route based on default estimates of link travel times, complemented by traffic information about current incidents broadcast by the Traffic Message Channel (TMC), is suggested to the driver. Default travel time data along with the network data is accessed by the guided vehicle from a static database available on an on-board DVD-ROM. The entire route computation takes place in the vehicle unit itself. In contrast, SRDRG has a more centralised system architecture with two-way data exchange between the guided vehicle and a Traffic Information Centre (TIC), increasing the range and accuracy of information made use of in providing guidance. The entire route computation takes place at the TIC and routes are transmitted to the vehicle in parts, for example at decision points. Only the ARDRG system architecture is considered in the work presented here, as this corresponds to most systems installed in vehicles to date.

The aim of this work is to develop an efficient and robust RDRG algorithm, capable of modelling the reliability element in route guidance and of considering the fact that travel time is not

constant throughout the day, but is continuously changing. The outcome of the algorithm should be a set of good route recommendations, satisfying a number of constraints and user acceptability criteria. A solution strategy based on the A* path finding algorithm (*1*), on Chen's link penalty method (*2-5*) and on Sung's Flow Speed Model (*6*) is presented and is experimentally tested on a part of Munich's road network.

This paper is structured as follows: Section 2 reviews relevant existing literature, while Section 3 presents Chen's link penalty method, which forms the basis of the present study. Section 4 gives an account of the link-based structure adopted, in order to run route guidance algorithms on real road networks, while Section 5 presents the proposed RDRG strategy and the time-dependent reverse A* algorithm. Finally, Section 6 presents the experiments carried out and the results obtained, while Section 7 concludes this paper.

2. LITERATURE REVIEW

In this section, existing literature on a number of relevant fields to this research work is reviewed. These include the A* path finding algorithm, time-dependent path finding techniques and travel time variability and reliability.

2.1. The A* Static Shortest Path Algorithm

Finding the shortest path in a network is one of the most frequently encountered problems, not only in transportation engineering, but also in computer science and operations research. Although various algorithms exist for finding the shortest path, their performance tends to significantly deteriorate with increasing network size. In route guidance, finding the shortest path is a subroutine that needs to be called very often. Due to the fact that the size of transportation networks is usually large, it is of vital importance to have a shortest path algorithm, which is efficient enough to produce accurate results in little computation time.

It should be noted here, that the term shortest path does not necessarily refer to the distance, but to the variable to be optimised. In the case of route guidance, one is usually not interested in the least distance path, but is more concerned with the least travel time path. The link weights used for this computation are travel time values, rather than distance values. Therefore, in the problem formulation, the variable to optimise (minimise) is the total travel time; the path achieving this is referred to as the shortest path in the rest of the paper.

The leading algorithm for finding the shortest path from an origin to a destination is the A* algorithm (*1*). The advantage of A* compared with other shortest path algorithms, such as Dijkstra's algorithm (*7*), is that A* is much more efficient, due to its ability to convert an uninformed search into an informed search. The A* algorithm uses a heuristic function to estimate the cost (travel time) from any point to the destination node of the network. This is often the airline distance to the destination. The shortest path is eventually found, under the condition that the heuristic does not overestimate the actual distance to the destination. The number of nodes expanded is generally much smaller than for other algorithms, making A* more efficient.

The concept of A* is summarised as follows; the algorithm holds two lists, the open list and the closed list. The closed list contains all the nodes of the network that have been expanded, whereas the open list contains all the nodes that may be expanded at the next step. At each step, one node is expanded and moved from the open list to the closed list, while its successors are placed into the open list. For every node n, $f(n) = g(n) + h(n)$ is calculated, where $g(n)$ is the travel time from the origin to n and $h(n)$ is the estimate of the travel time from n to the destination. The node to be expanded at each step is the node with the lowest $f(n)$ value among the nodes in the open list. When $h(n) = 0$ for all nodes n, A* reduces to Dijkstra's algorithm. Moreover, the efficiency of the algorithm increases, the closer $h(n)$ is to the actual value.

2.2. Time-Dependent Shortest Path Algorithms

When computing paths in a road network for DRG, traffic conditions are subject to continuous changes, such that the link weights are not constant anymore, but time-dependent. Therefore, the task to be carried out is finding the shortest path in a dynamic, time-dependent network. This has been extensively studied in the past, not only in the context of transportation, but also in the context of communication systems.

A theoretical study presenting algorithms for solving this problem was carried out by Orda and Rom (*8*), while a study by Ziliaskopoulos and Mahmassani (*9*) suggested the computation of shortest paths for all possible departure times whilst representing the label of a node by a vector of scalars instead of a single scalar, as is the case in static networks. Another approach adopted was to repeatedly apply a static shortest path algorithm over a certain number of time intervals, in order to account for changes in the link travel times, whilst attempting to reduce the computation time for subsequent searches by making use of the computed initial shortest path and only changing the elements of the path, whose weights have changed (*10*).

Nevertheless, these methods do not account for the main problem encountered when seeking the shortest path in a time-dependent network, which is the violation of the so-called first-in-first-out (FIFO) property. The FIFO property can be defined as follows: in a network, link travel times are usually such that vehicles arrive at the end of a link in the same order as they departed from the beginning of that link, or expressed otherwise, the traversal time of any link cannot decrease faster than real time. It has been proven that shortest path algorithms in a time-dependent network only remain optimal if the FIFO property holds (*11*). Based on that, Chabini and Lan (*12*) developed an efficient adaptation of the A* algorithm for finding the shortest path in a time-dependent FIFO network, i.e. a network consisting exclusively of links, on which the FIFO property holds.

In theory, the FIFO property always holds for transport networks, provided continuous travel time distributions exist for all links in the network. In practice however, travel time distributions are expressed in terms of fixed time intervals, which means that it is possible for the FIFO property not to hold. The most important contribution to the field of time-dependent shortest path algorithms was made by Sung et al (*6*), with the Flow Speed Model (FSM) being an efficient method of bypassing this problem and finding the shortest path in any time-dependent network, irrespective of the FIFO property.

The main idea behind Sung's FSM is that, while travel time intervals may result in discontinuities in the link travel time distribution, causing the FIFO property to be violated, flow speed intervals do not affect the corresponding link flow speed distribution. Basically, an abrupt increase in link flow speed results in all vehicles on the link to accelerate, so that no overtaking can take place and thus no vehicle entering the link later than another vehicle can exit the link earlier. A change in the flow speed of a link results in different vehicles experiencing different travel times on the link, according to their location at the time of the change; this is not taken into consideration in any of the previous approaches and is the main advantage of the FSM compared to them.

2.3. Travel Time Variability and Reliability

It has been found, that although travel time is an important factor affecting the traveller's route choice behaviour, travel time variability can be even more important (*13-16*). Travellers are interested in how long it will take them to reach their destination, but are even more concerned with the reliability of their prediction of total travel time. A wrong travel time prediction results in either an early arrival at the destination or in a delay. None of these situations is appreciated by the traveller, with delays having more severe consequences for him/her (e.g. late arrival at the workplace) not being tolerated. Consequently, it is an important requirement of a route guidance system to be able to accurately estimate travel time variability and to avoid potential sources of congestion (unreliable links), in order to avoid early arrivals and delays.

Some studies estimate travel time variance, and consequently reliability, through simulation (*17*). Others present analytical models for estimating the probability distribution of travel time in the

light of normal day to day variations in the travel demand matrix of a road network (*18,19*). Performing sensitivity analysis to determine reliability is another technique used in several studies (*20-22*).

However, reliability studies so far mainly focus on network reliability, with very few studies concentrating on the reliability of single links. The reliability of a link can be defined as the probability of encountering congestion on that link, in other words, the probability that the link's state will be "abnormal", as defined by Bell and Iida (*23*). The "normal" and "abnormal" states of a link are defined in terms of the traversal time of that link: namely, if the link's traversal time exceeds some preset threshold, then the state of the link is set to "abnormal". If there is a high probability that the state of the link will be "abnormal", then the link has a low reliability value. As a consequence of this definition, reliability can only take values between 0 and 1. Extending the definition of the reliability of a link, the reliability of a path is simply the product of the reliabilities of the series of links constituting the path in the absence of "failure dependence" (inter-link abnormality correlations).

3. CHEN'S LINK PENALTY METHOD

Chen's link penalty method, formulated by Chen et al (*2-5*) and further implemented by Kaparias et al (*24,25*), makes use of the A* algorithm, Sung's FSM and the concept of reliability described in the previous section. A methodology for modelling congestion risk and computing reliable paths in a time-dependent network, subject to a number of constraints imposed to ensure acceptability of the resulting path by the driver is presented. The constraints applied are a maximum path duration and a maximum path length constraint.

The constraints introduced are enforced by the application of link weight (travel time) penalties. The main idea lies in performing an initial search in order to find the fastest path without considering reliability, penalising unreliable links by applying travel time penalties and running subsequent searches, each time reducing the penalties, until the computed path satisfies the constraints imposed. The duration and length constraints are expressed in proportion to the duration and length of the fastest path, i.e. for any computed path with duration T and length L, it should be $T < \beta T_0$ and $L < \zeta L_0$, where T_0 and L_0 are the duration and length values of the fastest path and β and ζ are the duration and length permission parameters respectively. More specifically, unreliable links are initially excluded from subsequent path computations, provided this is possible; however, if a path satisfying the acceptability constraints cannot be found, unreliable links are progressively re-included in the search. The sequence in which this occurs depends on their reliability.

The idea of link penalties is also introduced as an attempt to develop an efficient and acceptable method for finding alternate paths in a network, as it has been proven that routing strategies suggesting multiple alternative paths to the driver are superior to single path solutions, because they decrease the risk of the congestion feedback phenomenon. However, it has to be guaranteed that the paths computed will be maximally disjoint, in order to reduce the probability of joint path failure as much as possible. This is enforced by additionally applying penalties to links that are already included in a path that was previously computed and accepted.

A third constraint, associated with the maximally disjoint paths is introduced, and this is the maximum path overlapping ratio constraint, ensuring that a path can only be accepted if it does not overlap too much with any previously computed path. Finally, to ensure that the algorithm will terminate and will not carry on computing an infinite number of paths, a maximum number of paths to be computed is introduced as a constraint. Of course the algorithm terminates earlier if only a smaller number of acceptable paths are found.

The following travel time penalty is used in Chen's link penalty method, for links having a reliability value lower than a preset threshold and for links that are already included in one of the computed paths:

$$\Delta t_i = \alpha^n (1-r_i)^q W_0$$

with $0 < \alpha < 1$, m = number of iterations, $q = 0$ if $m = 0$ otherwise $q = 1$, and W_0 = a value large enough to bring about link exclusion.

An interesting feature of Chen's link penalty method is the fact that for the first run of the A* algorithm, where the fastest path is computed, the algorithm runs in reverse order, that is from the destination to the origin. The advantage of running the A* from the destination to the origin is that after the first run, the computed cost from the destination to every visited node can be used as an estimate for subsequent forward A* runs. The number of nodes visited is thus significantly reduced and consequently the computation time is also significantly reduced.

In order to incorporate time-dependence, Sung's FSM is used by Chen et al (*5*). By slightly modifying the structure of the A* algorithm, the FSM can be implemented fairly easily for the case of the forward run of the algorithm. However, the implementation of the FSM to the reverse run of the A* algorithm is more complex, as the arrival time at the destination is not known at the start of the search, the arrival time at each link is also not known and hence it is not possible to calculate the experienced travel time by a vehicle on any link.

In order to avoid this problem, time-dependent link travel times have only been used for the forward A* runs in Chen's link penalty method so far; when running the reverse A*, static travel times have been used. More specifically, the lowest travel time value from each link's travel time distribution is used, so that the resulting node labels can be used as lower bounds of the remaining travel time to the destination in the subsequent forward A* runs. Nevertheless, besides the fact that these values can be rather conservative in the case where the actual link travel time values are much higher than the lowest ones, the guarantee of finding the fastest path in the initial A* run is lost, as it cannot be guaranteed anymore that the heuristic estimate of the travel time from any point to the origin underestimates the actual travel time.

In Section 5 a modified version of Chen's link penalty method is presented, in order to account for time-dependence when calculating the initial fastest path. Prior to that however, the link-based algorithmic structure developed in order to run path finding algorithms on real road networks for route guidance is described.

4. LINK-BASED STRUCTURE FOR ROUTE GUIDANCE IN REAL ROAD NETWORKS

The traditional algorithmic structure used by label-setting and label-correcting path finding algorithms on artificial grid networks is the one of a directed graph. The network consists of nodes and links, an origin and a destination node are specified and the implemented algorithm proceeds by setting or altering labels, located on each node. Nevertheless, when implementing route guidance algorithms on real road networks, this structure creates problems, mainly with respect to two issues: representation of turn restrictions, and positioning of the vehicle in the network. Therefore, modifications to this structure are made in this study in order to convert it to a more link-based form. The approaches developed in order to account for these two issues are described in the next paragraphs.

4.1. Turn Restrictions and Intersection Representation

Several attempts have been made in the past (*26-28*), each one suggesting different methods to represent turn prohibitions in a node-based structure. The approach adopted here however involves keeping a list of all allowed movements in the network database, where each movement is represented by two links, start-link and end-link. Then, a link-based structure is chosen, in which the labels are placed on the links themselves; each link is thus treated as two parts (start and end), each part holding its own label.

The A* algorithm is adapted accordingly: two open lists (open-start and open-end list) and two closed lists (closed-start and closed-end) are kept, containing links instead of nodes. Each link

part (start or end) is treated as a node in the conventional version of the A* algorithm. Thus, if the start part of a link has its label updated, it is placed in the open-start list and if its end part has its label updated, the link is placed in the open-end list. Similarly, if the start or the end part of the link is expanded, the link is removed from the open-start or open-end list and is placed in the closed-start or closed-end list respectively. This is a very efficient method of representing intersection movements, as it enables the application of label-correcting algorithms without introducing any extra elements to the network.

4.2. Positioning of the Vehicle in the Network

In real road networks, where the nodes correspond to junctions, this approach has a serious drawback, and that is the fact that not only the position of the vehicle is required, but also its direction. It is possible, that a vehicle is located on a road between two junctions, facing towards one of them and not being able to make a U-turn. However, if the position of the vehicle is expressed as the nearest node, it is very likely that a recommended route requires the vehicle to make illegal or impossible movements.

A solution to the problem presented here is to slightly modify the network structure and make it link-based rather than node-based. Namely, when specifying the origin and destination, an origin-link and a destination-link are specified, considering that the origin or the destination respectively are located somewhere on this link.

Using this technique, the setting of the origin link immediately limits the next allowed movements. Similarly, the setting of the destination link results in the direction of approach of the destination being fixed, which is very important in some cases. In any case, when the user is prompted to enter the destination of his/her trip, it is most likely that this will be a street name, therefore only referring to a link.

5. THE TIME-DEPENDENT RDRG ALGORITHM

The main problem when running the time-dependent A* algorithm backwards, as described in Section 3 of this paper, is the fact that the arrival time at the destination is not known and therefore the arrival time at every intermediate point is also not known. The only piece of data available at the start of the entire procedure of providing RDRG is the departure time from the origin. Thus, it is not possible to run the time-dependent A* algorithm backwards, unless there is at least some indication of the arrival time at the destination.

The method described in this paper aims to modify Chen's link penalty method, so that, using Sung's FSM, it will be possible to derive the time-dependent fastest path from the initial A* run and then compute further maximally disjoint time-dependent reliable acceptable paths using subsequent A* runs. For this purpose, the order in which the various A* runs are carried out is reversed here. The procedure is described in the next paragraphs.

5.1. Concept

Unlike the original Chen's link penalty approach, where an initial reverse A* run takes place, in the procedure developed here an initial forward A* run is carried out, yielding the time-dependent fastest path, not making any reliability considerations. From this, taking into account the departure time from the origin, the arrival time (AT) at the destination is obtained. Then, applying the maximum path travel time acceptability criterion, i.e. that any acceptable path's travel time may not be longer than the fastest path's travel time multiplied by the duration permission parameter, and adding the maximum acceptable path travel time to the departure time (DT), the latest acceptable arrival time (AAT) at the destination is obtained.

Following that and applying penalties on all unreliable links, a time-dependent reverse A* algorithm is run starting from the AAT, in order to obtain a reliable path, along with an acceptable departure time (ADT), which is the time point at which one would need to depart from the origin so as

to arrive at the destination before the AAT, using the newly computed reliable path. Having obtained the path, its actual travel time is computed, i.e. the travel time when departing from the origin at time DT rather than ADT, along with its AT. The path is then checked against the constraints, in addition to the condition that the ADT must be later than the DT. If the constraints are met, the path is kept; otherwise, link penalties are reduced and a further reverse A* run is carried out to compute a new path.

The presentation of the time-dependent forward A* algorithm using Sung's FSM is included in Chen et al (*3,5*) and is therefore omitted in this paper. In the next sub-section, the time-dependent reverse A* algorithm is presented.

5.2. The Time-Dependent Reverse A* Algorithm

5.2.1. Definitions and notation

Consider a network G, consisting of a set of nodes N, a set of directional links V and a set of movements M, specifying the allowed turning movements between links. Movements are represented by their start and end links, such that movement (a,b) connects links a and b. If Z is a set of time intervals $[\tau_k, \tau_{k+1})$, $k = 0, 1, \ldots, m-1$ separated by the time points $\tau_0 < \tau_1 < \tau_2 < \ldots < \tau_{m-1} < \tau_m$ and covering one day, then for every time interval $\Delta\tau_k : [\tau_k, \tau_{k+1})$, every link l has a flow speed value $\upsilon_k(l)$ and every movement (a,b) has a delay value of $\delta_k(a,b)$.

o:	The origin link
d:	The destination link
l_x:	Part of link l, s for start, e for end
$t(l)$:	The travel time of link l
$r(l)$:	The reliability of link l
$r(a,b)$:	The reliability of movement (a,b)
$\lambda_k(l)$:	The length of link l
$OP_s \subset V$:	Open-start list
$OP_e \subset V$:	Open-end list
$CL_s \subset V$:	Closed-start list
$CL_e \subset V$:	Closed-end list
$\Gamma^{-1}(l)$:	The set of predecessor links of link l
$s(l)$:	The successor link of l on the optimal path
L:	The list of links in the optimal path
D:	DT
A_{acc}:	AAT
D_{acc}:	ADT
$P_i(D_i,L_i,T_i,\Lambda_i,R_i,A_i)$:	Path i containing the links in list L_i and having a total travel time T_i, a total length Λ_i, a total reliability R_i, departure time D_i and arrival time A_i

5.2.2. Reverse A procedure*

Step 0 (Initialisation): Set $OP_s = \{d_s\}$, $OP_e = CL = L = \varnothing$, $g(d_s) = 0$, $T = 0$, $\Lambda = 0$ and $R = 0$.
$\forall\ l \in V\backslash\{d\}$ *set* $g(l_s) = g(l_e) = \infty$ and input $h(l_s)$ and $h(l_e)$.

Step 1 (Select link part for expansion): $l_x^{exp} = Argmin_{lx \in (OP_s \cup OP_e)} [f(l_x) = g(l_x) + h(l_x)]$.
$CL_x = CL_x + \{l_x^{exp}\}$, $OP_x = OP_x - \{l_x^{exp}\}$.

Step 2 (Branch and update): If $x = e$
If $l_e^{exp} = o_e$ then go to Step 3.
$OP_s = OP_s + \{l_s^{exp}\}$, $g(l_s^{exp}) = g(l_e^{exp}) + t(l^{exp})$,
and, if $l_s^{exp} \in CL_s$, $CL_s = CL_s - \{l_s^{exp}\}$.
Else if $x = s$

$\forall\ u \in \Gamma^{-1}(l^{exp})$:
$OP_e = OP_e + \{u_e\}$.
If $g(u_e) \geq g(l_s^{exp}) + \delta(u,l^{exp})$,
then $g(u_e) = g(l_s^{exp}) + \delta(u,l^{exp})$, $s(u) = l^{exp}$ and,
if $u_e \in CL_e$, $CL_e = CL_e - \{u_e\}$.
Go to Step 1.

Step 3 (Trace path and terminate):
$D_{acc} = A_{acc} - g(o_e)$.
Set $l' = o$.
Repeat until $l'' = d$:
$l'' = s(l')$, $T = T + \delta(l',l'') + t(l'')$, $\Lambda = \Lambda + \lambda(l'')$,
$R = R \cdot r(l',l'') \cdot r(l'')$, $L = L + \{(l'')\}$ and set $l' = l''$.
$A = D + T$.
Output path $P(D,L,T,\Lambda,R,A)$ and stop.

5.2.3. Calculation of time-dependent link travel time and movement delay values

Using Sung's FSM method (*6*), time-dependent link travel time and movement delay values can be calculated, taking into account the arrival time at each point of the reverse A* search. Basically, a vehicle exiting link l, of length $\lambda(l)$, at time $\tau_{ex}(l)$, where $\tau_{ex}(l) \in Z$, will have entered link l at time $\tau_{ent}(l) \in Z$, where $\tau_{ex}(l) - \tau_{ent}(l) = t(l)$, which is the travel time of link l. The time of entry of the vehicle in link l is calculated as follows:

$$
\begin{array}{lll}
\tau_{ent}(l) = \tau_{ex}(l) - \lambda(l) / \upsilon_{m-1}(l) & \text{if} & \lambda(l) / \upsilon_{m-1}(l) < \tau_{ex}(l) - \tau_{m-1} \\
\text{else}\quad \tau_{ent}(l) = \tau_{m-1} - (\lambda(l) - \lambda_{m-1}) / \upsilon_{m-2}(l) & \text{if} & (\lambda(l) - \lambda_{m-1}) / \upsilon_{m-2}(l) < \tau_{m-1} - \tau_{m-2} \\
\text{else}\quad \tau_{ent}(l) = \tau_{m-2} - (\lambda(l) - \lambda_{m-2}) / \upsilon_{m-3}(l) & \text{if} & (\lambda(l) - \lambda_{m-2}) / \upsilon_{m-3}(l) < \tau_{m-2} - \tau_{m-3} \\
\text{else}\quad \tau_{ent}(l) = \tau_{m-3} - (\lambda(l) - \lambda_{m-3}) / \upsilon_{m-4}(l) & \text{if} & (\lambda(l) - \lambda_{m-3}) / \upsilon_{m-4}(l) < \tau_{m-3} - \tau_{m-4} \\
\quad\vdots & & \\
\text{else}\quad \tau_{ent}(l) = \tau_2 - (\lambda(l) - \lambda_2) / \upsilon_1(l) & \text{if} & (\lambda(l) - \lambda_2) / \upsilon_1(l) < \tau_2 - \tau_1 \\
\text{else}\quad \tau_{ent}(l) = \tau_1 - (\lambda(l) - \lambda_1) / \upsilon_0(l) & \text{if} & (\lambda(l) - \lambda_1) / \upsilon_0(l) < \tau_1 - \tau_0
\end{array}
$$

where

$$
\begin{aligned}
&\lambda_{m-1} = \upsilon_{m-1}(l)\,(\tau_{ex}(l) - \tau_{m-1}) \\
&\lambda_{m-2} = \lambda_{m-1} + \upsilon_{m-2}(l)\,(\tau_{m-1} - \tau_{m-2}) \\
&\lambda_{m-3} = \lambda_{m-2} + \upsilon_{m-3}(l)\,(\tau_{m-2} - \tau_{m-3}) \\
&\quad\vdots \\
&\lambda_2 = \lambda_3 + \upsilon_2(l)\,(\tau_3 - \tau_2) \\
&\lambda_1 = \lambda_2 + \upsilon_1(l)\,(\tau_2 - \tau_1)
\end{aligned}
$$

Using this technique and taking into account the fact that the exit time from a link is known as it is shown on the $g(l_e)$ label of the end part of link l, it is possible to calculate the entry time of the vehicle in link l and thus obtain the time-dependent travel time of link l.

When it comes to calculating the time-dependent delay $\delta(a,b)$ of a movement (a,b), the above technique cannot explicitly be applied, because movements do not have length and flow speed values; instead, a delay value is given for each time interval. Therefore, in order to apply Sung's FSM method, the delay value $\delta_k(a,b)$ for each time interval $\Delta\tau_k : [\tau_{k-1}, \tau_k)$ is converted to a fictional speed value $\upsilon_k(a,b)$, such that $\upsilon_k(a,b) = \lambda(a,b) / \delta_k(a,b)$, where $\lambda(a,b)$ is a fictional movement length value.

Thus, a vehicle entering link b (exiting movement (a,b)) at time $\tau_{ent}(b) \in Z$ will have exited link a at time $\tau_{ex}(a) \in Z$, such that $\tau_{ent}(b) - \tau_{ex}(a) = \delta(a,b)$, which is the delay of movement (a,b). The

procedure presented above is then applied, taking into account the fact that $\tau_{ent}(b)$ is known from label $g(a_s)$.

5.3. Formulation of the RDRG Algorithm

The algorithm formulation of the concept described earlier in this section is presented here. Some additional notation is given first, followed by the presentation of the procedure. As the notation used in the previous sub-section also applies here, only newly introduced variables are defined next.

5.3.1. Definitions and notation

m:	Iterations counter
p:	Paths counter
LP:	Set of links to be penalised
PS:	List of p computed paths
T_{max}:	Path travel time threshold ($T_{max} = \beta T_0$)
Λ_{max}:	Path length threshold ($\Lambda_{max} = \zeta \Lambda_0$)
r_{min}:	Link reliability threshold
ε:	Path overlapping ratio
ε_{max}:	Maximum path overlapping ratio threshold
N_{max}:	Maximum number of paths

5.3.2. Procedure

Step 0: Initialisation

Forward A* algorithm for o and d → Fastest path $P_0(D_0,L_0,T_0,\Lambda_0,R_0,A_0)$.
$A_{acc} = D + \beta T_0$.
$\forall\ l_x \in CL_x$ set $h(l_x) = g(l_x)$.
Set $W_0 = \gamma T_0$, where $1.5 \leq \gamma \leq 3$.
Set $m = 0$, $p = 1$, $PS = \varnothing$.

Step 1: Link penalty application

$LP = \varnothing$.
$\forall\ l \in V$:
  If $r(l) < r_{min}$ or $l \in PS$:
    $LP = LP + \{l\}$.
$\forall\ l \in LP$
  $t'(l) = t(l) + \alpha^m (1-r(l))^q W_0$, $0 < \alpha < 1$, $q = 0$ for $m = 0$, $q = 1$ otherwise.
$\forall\ l \in V \setminus LP$
  $t'(l) = t(l)$
$m = m + 1$.

Step 2: Reliable route computation

Reverse A* algorithm using $t'(l)$ for path finding but $t(l)$ for the computation of T_p. → Path $P_p(D_p,L_p,T_p,\Lambda_p,R_p,A_p)$.

Step 3: Check constraints

If $P_p \in PS$, go to Step 4.
Else
  Set $max\varepsilon = 0$.
  $\forall\ P_n \in PS$
    $\varepsilon = \Lambda_{Lp \cap Ln} / (\Lambda_{Lp \setminus \{Lp \cap Ln\}} \cdot \Lambda_{Ln \setminus \{Lp \cap Ln\}})^{1/2}$
    If $\varepsilon > max\varepsilon$, $max\varepsilon = \varepsilon$.
  Set $\varepsilon = max\varepsilon$.
  If $T_p < T_{max}$, $D_{acc} > D$, $\Lambda_p < \Lambda_{max}$, and $\varepsilon < \varepsilon_{max}$
    $PS = PS + \{P_p\}$.
    $p = p + 1$.

If $p < N_{max}$, go back to Step 1 otherwise go to Step 4.
Else go back to Step 1.

Step 4: Termination Output *PS*

It should be noted that by carrying out a forward A* search in the initialisation step of the algorithm, optimality can be guaranteed for the resulting path; this means that the initial path obtained is actually the fastest available path. However, as optimality is not needed when subsequently computing maximally disjoint paths, the heuristic estimates used in the reverse A* algorithm runs do not necessarily need to underestimate the actual travel time from the origin to any point. The values on the *g*-labels, set by the initial forward A* search, may not underestimate the actual cost from any point to the origin, however they are very close approximations to it. Thus, it is acceptable to use them as heuristic estimates in the reverse A* runs, as this reduces computation time.

6. EXPERIMENTAL RESULTS

In order to implement the algorithm presented in the previous section, a simulation experiment using ICNavS, a software tool developed for this purpose, is carried out on part of the road network of Munich, Germany. The test network has 3506 nodes and 7130 links, spreads over the northern suburbs of the city and covers an area of about 10 km length and about 7.5 km width.

Flow speed, junction delay and reliability data is simulated for the test network, in order to carry out the experiment. Regarding flow speed data, the approach used is based on the five road types encountered in a road network (Motorway, Major A-Road, A-Road, B-Road and Minor Road) and on their speed limits. Assuming that vehicles travel at a constant speed for a given time interval and that the maximum speed they can be travelling at is the speed limit, occurring during off peak times, flow speed distributions are obtained for every link type. Speed values are simulated for 15-minute intervals, for weekdays and weekends.

Similarly to flow speed data, junction delay data is simulated based on the type of turn (right, left or straight-on). Right turns are assigned a delay value of 0, as vehicles turning right do not usually come into conflict with any other traffic streams. For left turns, delay values are assigned according to the type of the link from which the turn starts and the type of the link to which the movement is directed. For straight-on movements, delay values are set, depending on the road types of the links that are being crossed. The distribution of junction delays throughout the day is simulated for the same 15-minute intervals as for the flow speed values, taking into account peak and off-peak times. Delay values range from 0 to 45 seconds.

Figure 1 shows the distribution of flow speed values throughout the day for all link types, as well as the delay distributions of left and straight-on movements starting from links of type “Major A Road”. Regarding reliability data, a number of links and movements in the network are assigned reliability values, based on expert knowledge.

The parameters of the simulation experiment are given next. Parameter α, coming into the link penalty application function is set to 0.7, while the value of γ, coming into the same function, is set to 1.9. The duration permission parameter β is set to 1.3, the length permission parameter ζ is set to 2, while the link reliability threshold r_{min} is set to 0.9. Finally, the maximum path overlapping ratio ε_{max} has a value of 2 and the maximum number of paths N_{max} is 5.

The experiment is carried out as follows; the BMW headquarters are chosen as the origin of the trip, while the destination is set to be the northbound direction of the A92 motorway, connecting the city to the airport. The trip takes place during the afternoon peak on a weekday, knowing that the majority of the links on the main artery out of the city, the A9 motorway, are unreliable (have reliability values of 0.6-0.7), while, the links forming the B13 road (Ingolstädter Strasse), which is the main alternative to the A9 motorway, are even more unreliable, with reliability values of 0.3-0.4. The time-dependent RDRG algorithm is run, in order to compute a set of maximally disjoint reliable paths.

The fastest path and three alternative paths to it are computed and are shown on Figure 2. The fastest path (a) is computed first, not taking into account the links and movements with low reliability values. It has a travel time of 22 minutes, a length of 14.5 km and goes through the unreliable A9 motorway, then onto the A99 and finally reaching the A92. The fact that the A9 is used, results in the reliability value of the path being very low (0.003).

In the calculation of the first alternative path however (b), reliability is taken into account, and this is why the A9, as well as the next choice, the unreliable B13 are avoided. Instead, a route along the B304 is chosen, leading onto the A99 and approaching the A92 from a different direction. This route has a slightly higher travel time that the fastest route (24 minutes), a shorter length (11.4 km), but most importantly, a much higher reliability value (0.5).

For the calculation of the second alternative path (c), not only unreliable links and movements are penalised, but also links already included in the first alternative path. Hence, the B304, used by the first alternative path, is avoided. The unreliable A9 is also avoided, and the path goes along the B13, nevertheless still avoiding its unreliable parts, joining the A99 and leading onto the A92. The path has a travel time of 26 minutes, a length of 10.4 km and a reliability value of 0.25.

Finally, the third alternative path (d) avoids the links already included in the previous alternative paths as much as possible. This is why this path, despite sharing some parts of the B304 and the A99 with the first alternative path, approaches the B304 using a different road from the first alternative path and joins it at a different position. It has a travel time of 25 minutes, a length of 11.6 km and a reliability value of 0.31.

7. CONCLUSIONS

In this paper, an algorithm for computing maximally disjoint reliable paths in a network with time-dependent travel times in the context of route guidance was presented. The algorithm, which makes use of the A* path finding algorithm, Chen's link penalty method and Sung's Flow Speed Model method, was also tested on a part of Munich's road network.

Future research will concentrate on extending the approach to networks with variable travel time and reliability values. An improved reliability measure is to be determined, expressing travel time uncertainty in terms of the potential total delay to be encountered. Also, "failure dependence" will be looked into, so as to determine inter-link abnormality conditions. Further experiments will be carried out on a number of different networks of various sizes, using travel time and junction delay values resulting from floating car data, in order to test the efficiency and effectiveness of the algorithm on large-scale networks under real congestion conditions. Finally, the results obtained will be complemented by a series of field trials, in order to compare the proposed algorithm with existing route guidance systems.

ACKNOWLEDGEMENT

The authors would like to thank BMW AG for sponsoring this work.

REFERENCES

(*1*) Hart, PE, Nilsson, NJ and Raphael, B, "A formal basis for the heuristic determination of minimum cost paths", *IEEE Transactions on Systems Science and Cybernetics U39* 4, 1968, pp. 100-107.

(*2*) Chen, Y, Kaparias, I, Bell, MGH and Bogenberger, K, Reliable autonomous route guidance by a constrained A* search considering intersection delays, *The Reliability of Traveling and the Robustness of Transport Systems*, 2005.

(*3*) Chen, Y, Bell, MGH, Wang, D and Bogenberger, K, "Risk-averse time-dependent route guidance by constrained dynamic A* search in decentralized system architecture", *Transportation Research Record* 1944, 2006, pp. 51-57.

(*4*) Chen, Y, Bell, MGH and Bogenberger, K, Reliable pre-trip multi-path planning and dynamic adaptation for a centralized road navigation system, *ITSC 2005 - 8th International IEEE Conference on Intelligent Transportation Systems*, 2005.
(*5*) Chen, Y, Bell, MGH and Wang, D, Risk-averse time dependent route guidance by a constrained dynamic A* search in decenterized structure, *TRB 2006 Annual Meeting CD-ROM*, 2006.
(*6*) Sung, K, Bell, MGH, Seong, M and Park, S, "Shortest paths in a network with time-dependent flow speeds", *European Journal of Operational Research* 121, 2000, pp. 32-39.
(*7*) Dijkstra, EW, "A note on two problems in connexion with graphs", *Numerische Mathematik* 1, 1959, pp. 269-271.
(*8*) Orda, A and Rom, R, "Shortest-path and minimum-delay algorithms in networks with time-dependent edge length", *Journal of the Association for Computing Machinery* 37, 1990, pp. 607-625.
(*9*) Ziliaskopoulos, AK and Mahmassani, HS, "Time-dependent shortest path algorithm for real-time intelligent vehicle highway system applications", *Transportation Research Record* 1408, 1993, pp. 94-100.
(*10*) Kim, J and Jung, S, A dynamic window-based approximate shortest path re-computation method for digital road map databases in mobile environments, 2002.
(*11*) Kaufmann, DE and Smith, RL, "Fastest paths in time-dependent networks for IVHS application", *IVHS Journal* 1, 1993, pp. 1-12.
(*12*) Chabini, I and Lan, S, "Adaptations of the A* algorithm for the computation of fastest paths in deterministic discrete-time dynamic networks", *IEEE Transactions on Intelligent Transportation Systems U18* 3, 2002, pp. 60-74.
(*13*) Adbel-Aty, A, Kitamura, R and Jovanis, PP, "Investigating the effect of travel time variability on route choice using repeated measurement stated preference data", *Transportation Research Record* 1493, 1995, pp. 39-45.
(*14*) Bates, J, Polak, J, Jones, P and Cook, A, "The valuation of reliability for personal travel", *Transportation Research E* 37, 2001, pp. 191-229.
(*15*) Lam, TC and Small, KA, "The value of time and reliability: measurement from a value pricing experiment", *Transportation Research E* 37, 2001, pp. 231-251.
(*16*) Liu, HX, Recker, W and Chen, A, "Uncovering the contribution of travel time reliability to dynamic route choice using real-time loop data", *Transportation Research A* 38, 2004, pp. 435-453.
(*17*) Noland, RB, Small, KA, Koskenoja, PM and Chu, X, "Simulating travel reliability", *Regional Science and Urban Economics* 28, 1998, pp. 535-564.
(*18*) Pattanamekar, P, Park, D, Rilett, LR, Lee, J and Lee, C, "Dynamic and stochastic shortest path in transportation networks with two components of travel time uncertainty", *Transportation Research C* 11, 2003, pp. 331-354.
(*19*) Clark, S and Watling, D, "Modelling network travel time reliability under stochastic demand", *Transportation Research B* 39, 2005, pp. 119-140.
(*20*) Asakura, Y, Hato, E and Kashiwadani, M, Stochastic network design problem: an optimal link investment model for reliable network, *Proceedings 1st International Symposium on Transportation Network Reliability*, 2001.
(*21*) Bell, MGH, Cassir, C, Iida, Y and Lam, WHK, A sensitivity based approach to network reliability assessment, *Proceedings 14th International Symposium on Transportation and Traffic Theory*, 1999.
(*22*) Du, Z-P and Nicholson, A, "Degradable transportation systems: sensitivity and reliability analysis", *Transportation Research B* 32, 1997, pp. 225-237.
(*23*) Bell, MGH and Iida, Y, "Network reliability" in *Transportation network analysis*, Wiley and Sons, Chichester, 1997.
(*24*) Kaparias, I, Bell, MGH, Chen, Y and Bogenberger, K, ICNavS: A tool of reliable dynamic route guidance, *13th World Congress on Intelligent Transportation Systems*, 2006.
(*25*) Kaparias, I, Bell, MGH, Chen, Y and Bogenberger, K, "Autonomous reliable dynamic route guidance with re-routing", *Transportation Research C (Submitted)*, 2006
(*26*) Wattleworth, JA and Shuldiner, PW, "Analytical methods in transportation; left turn penalties in traffic assignment models", *Journal of Engineering Mechanics of the American Society of Civil Engineers* 89, 1963, pp. 97-126.
(*27*) Kirby, RF and Potts, RB, "The minimum route problem for networks with turn penalties and prohibitions", *Transportation Research B* 3, 1969, pp. 397-408.

(*28*) Ziliaskopoulos, AK and Mahmassani, HS, "A note on least time path computation considering delays and prohibitions for intersection movements", *Transportation Research B* 30, 1996, pp. 359-367.

FIGURE 1 (a) Flow speed distributions for (i) weekdays and (ii) weekends, (b) example distributions of left turn delays starting from a "Major A Road" link for (i) weekdays and (ii) weekends, (c) example distributions of straight-on movement delays starting from a "Major A Road" link for (i) weekdays and (ii) weekends.

FIGURE 2 Output of the RDRG algorithm: (a) Fastest path, not considering reliability, (b) first alternative path, (c) second alternative path, (d) third alternative path.

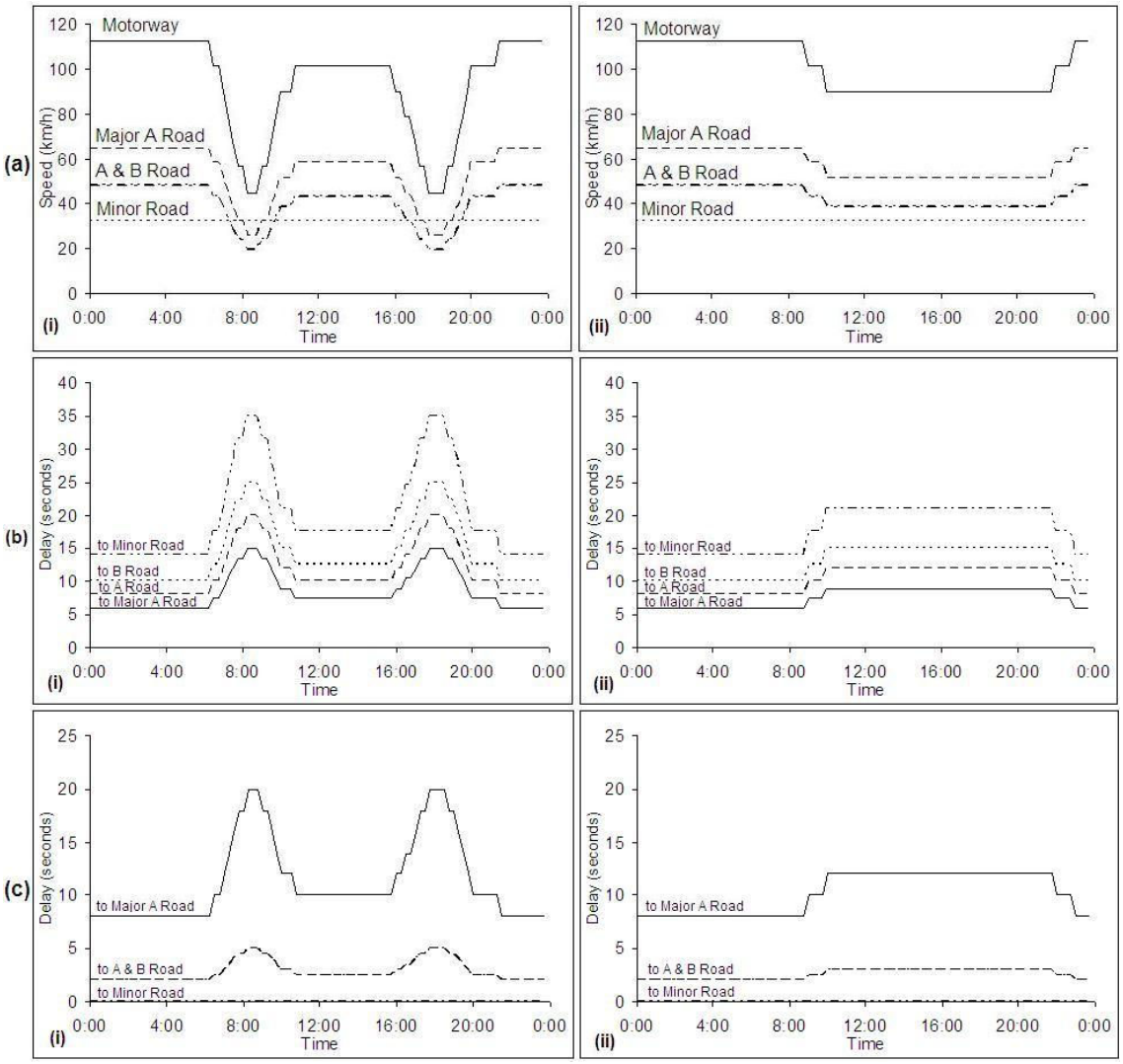

FIGURE 1 (a) Flow speed distributions for (i) weekdays and (ii) weekends, (b) example distributions of left turn delays starting from a "Major A Road" link for (i) weekdays and (ii) weekends, (c) example distributions of straight-on movement delays starting from a "Major A Road" link for (i) weekdays and (ii) weekends.

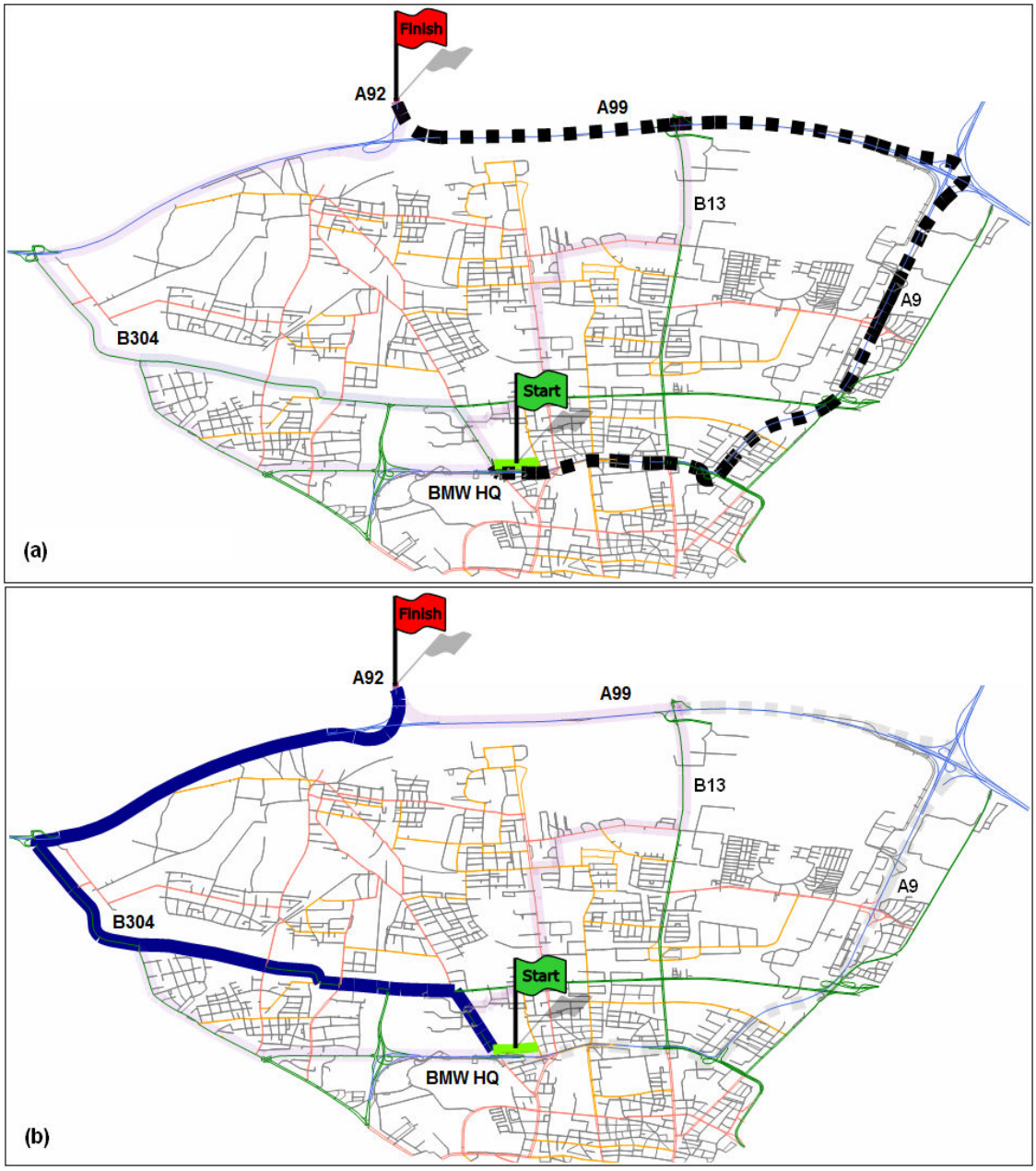

FIGURE 2 Output of the RDRG algorithm: (a) Fastest path, not considering reliability, (b) first alternative path, (c) second alternative path, (d) third alternative path.

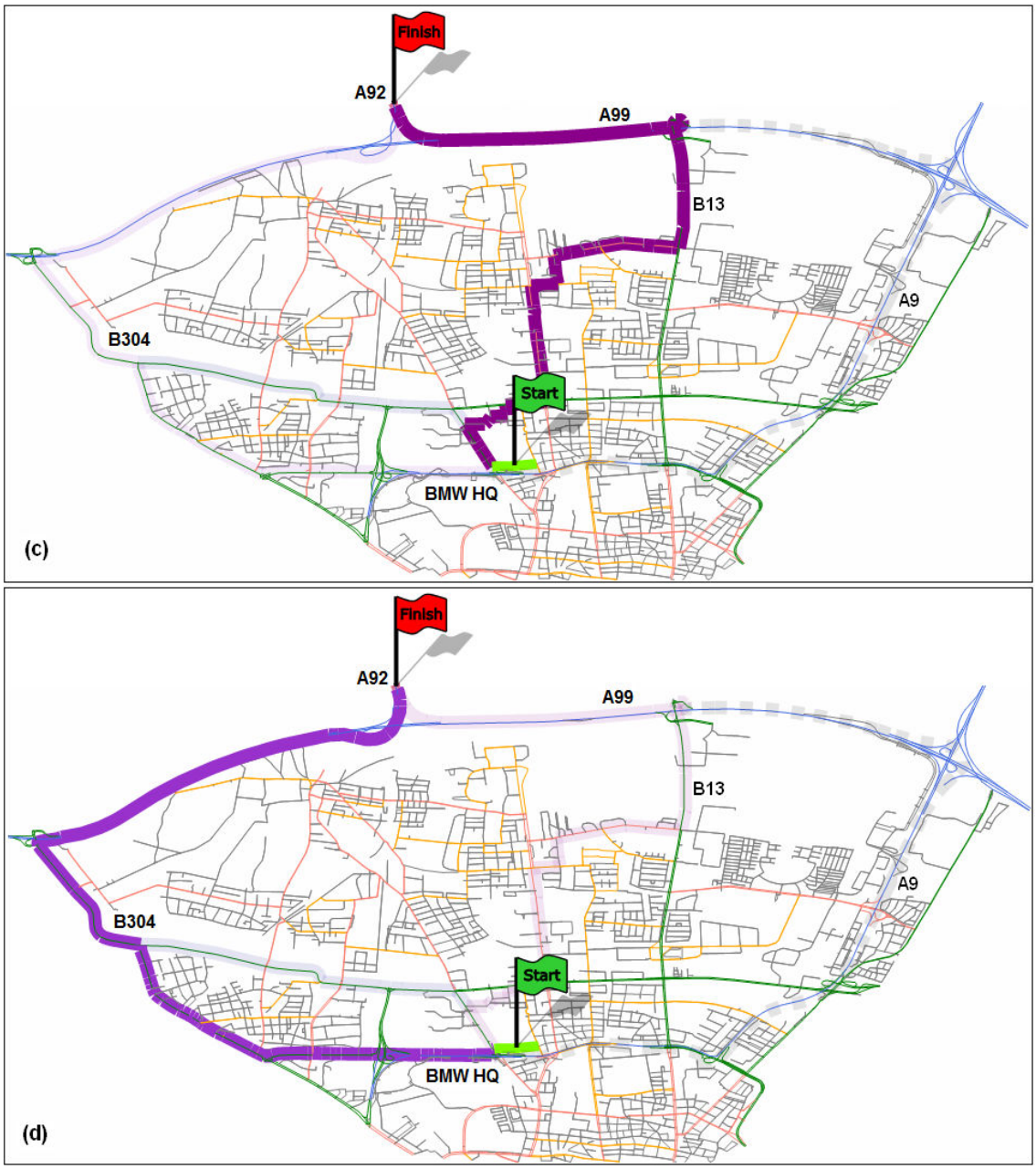

FIGURE 2 (continued)