Time-dependent Hyperstar algorithm for robust vehicle navigation in time-dependent stochastic road networks

Michael G H Bell¹, Valentina Trozzi¹, Solmaz Haji Hosseinloo¹, Guido Gentile² and Achille Fonzone¹

Abstract

The vehicle navigation problem studied in Bell (2009) is revisited and a time-dependent reverse Hyperstar algorithm is presented. This minimises the expected time of arrival at the destination, and all intermediate nodes, where expectation is based on a pessimistic (or risk-averse) view of unknown link delays. This may also be regarded as a hyperpath version of the Chabini and Lan (2002) algorithm, which itself is a time-dependent A* algorithm. Links are assigned undelayed travel times and maximum delays, both of which are potentially functions of the time of arrival at the respective link. The driver seeks probabilities for link use that minimise his/her maximum exposure to delay on the approach to each node, leading to the determination of the pessimistic expected time of arrival. Since the context considered is vehicle navigation where the driver is not making repeated trips, the probability of link use may be interpreted as a measure of link attractiveness, so a link with a zero probability of use is unattractive while a link with a probability of use equal to one will have no attractive alternatives. A solution algorithm is presented and proven to solve the problem provided the node potentials are feasible and a FIFO condition applies for undelayed link travel times. The paper concludes with a numerical example.

Key words: Robust route guidance; Vehicle navigation; Uncertain networks

Introduction

Vehicle navigation systems have made great advances in recent years. Not only has the quality of the electronic maps incorporated in them become more accurate, but also their user friendliness has improved. Current systems allow users to vary the criteria used for building routes (for example, fastest or shortest) or to avoid certain links, route sections or areas (like the congestion charging zone in London). Recent systems also allow the inclusion of on-line traffic information, broadcast with the TMC or more recently the TPEG protocol, enabling drivers to be routed away from congestion once this has been detected.

In addition to such improvements, there has been a move to make navigation available via GPS-enabled cell phones, such as the iPhone from Apple. The rapid spread of such devices has created the possibility of “crowdsourcing” maps and traffic data by tracking the users of such cell phones. Although “crowdsourcing” is in its infancy, the trend is clearly toward the more widespread use of navigation systems by cell phone, both in and out of vehicles. In addition, access to the internet facilitates the provision of a wide range of location-based services to accompany navigation, ranging from locating friends to making reservations.

¹ Centre for Transport Studies, Imperial College London
² Dipartimento di Idraulica Trasporti e Strade, Sapienza Università di Roma
Experience suggests that on-line congestion information alone may lead to unsatisfactory results as not all roads are monitored, the knock-on effects of mass rerouting is not taken into account, and there is often a lag between the onset of congestion and the broadcast of a warning message. As a consequence, systems have been developed to predict congestion (see, for example, the Inrix system). An alternative approach is to seek routes that avoid links with a history of congestion on the grounds that they are likely to prove unreliable in terms of travel time.

Market research has revealed that expected arrival time reliability is a major concern for users of navigation systems (see Park 2009). Recent work designed to generate more reliable routes is reviewed in Bell (2009). In general, most approaches so far proposed involve measuring link travel time reliability and then finding paths which tend to avoid links that are unreliable (see for example, Kaparias 2008, Chen et al. 2007 or Park et al 2007). This may be achieved by penalising unreliable links or by the use of a $k$-shortest path method. It should be noted that a path that excludes unreliable links may still be unreliable in total, particularly where the resulting path is tortuous.

More recently, Bell (2009) has noted that a reinterpretation of the Spiess and Florian (1989) algorithm enables it to be used to generate the set of paths, referred to as the hyperpath, that minimises expected travel time if at each node offering a choice of route drivers assign themselves probabilistically to the exit links so as to minimise their exposure to maximum delay on these exit links. This may be regarded as a robust approach to path finding, as it requires that each link be given a maximum delay but makes no assumption about the distribution of actual link delay, other than it lie between zero and the given maximum.

An important feature of road networks is that link travel times are subject to both regular time-of-day variation, for example peak and off-peak periods, and irregular incidents. There is therefore a strong case, particularly in urban areas, for time-dependent path finding. Because of the large number of links in road networks, efficient path finding algorithms are required. For static networks, the Dijkstra (1959) algorithm, which can be regarded as the mother of all efficient shortest path algorithms, may be sped up by the addition of node potentials (see Wagner and Willhalm, 2006) leading to the well known A* algorithm first proposed by Hart et al (1968). Although all vehicle navigation systems appear to use some form of the A* algorithm, the efficiency in practice depends on the choice of node potentials, and this remains a closely guarded secret. Bell (2009) showed that node potentials may also be applied to the reinterpreted Spiess and Florian algorithm speeding up the generation of hyperpaths. The resulting algorithm is referred to there and here as the Hyperstar algorithm.

Chabini and Lan (2002) have shown that the A* algorithm may be applied to time-dependent networks in a straight forward way provided links conform to a FIFO condition. FIFO implies that it should not be possible for link travel time as experienced by drivers to fall faster than real time, otherwise the sequence of drivers entering a link may differ from the sequence leaving it, implying overtaking or queue jumping within the link. However, a problem in applying the Hyperstar algorithm to time-dependent networks is that it builds the hyperpath from the destination forward to the origin. It is conventionally assumed that link travel time is a function of the time of arrival at a link and not the time of departure from a link. Moreover, Hyperstar provides a pessimistic estimate of the time of departure from the origin given the time of arrival at the destination, whereas a navigation system would normally require a pessimistic estimate of the time of arrival at the destination given the time of departure from the origin.
This paper explores the implications and consequences of reversing the Hyperstar algorithm to generate a hyperpath from the origin to the destination in the context of time-dependent networks. It shows that this reversal implies that drivers minimise their exposure to arrival delays at the destination and all intermediate nodes in the hyperpath, arguably a more rational strategy than basing route choices on departure delays from the origin and all intermediate nodes. It is proved that the reversed Hyperstar algorithm solves the underlying time-dependent problem, provided each link conforms to FIFO operation and all node potentials are feasible. The reversed Hyperstar algorithm is therefore the hyperpath counterpart to the Chabini and Lan (2002) time-dependent A* algorithm. An 8 time 8 node grid network with randomly generated link lengths and maximum link delays is used to illustrate some of the properties of the reversed Hyperstar algorithm.

Hyperpaths

A technical as well as a historical introduction to hypergraphs and hyperpaths may be found in Gallo et al. (1993), cited in Nielsen et al. (2005). A directed hypergraph consists of a set of nodes (vertices) and a set of hyperlinks (hyperarcs). A hyperlink consists of a set of head nodes and a set of tail nodes, either of which (but not both) may be empty. In contrast to a normal link, a hyperlink may have multiple head and tail nodes. Special hypergraphs arise when either the set of head nodes or the set of tail nodes is singular. A forward hypergraph, or F-graph, arises when the hyperlinks have singular head nodes.

Hyperpaths in hypergraphs are an extension to paths in graphs that allow graphs to represent more complex situations (Nielsen et al. 2005). A hypernode can be used to represent a transport interchange and hyperlinks the connections between interchanges, for example. Hall (1986) formulated the minimum expected travel time path problem for random time-dependent networks. He noted that finding the best route requires a strategy that assigns optimal successors to a node as a function of time. Pretolani (2000) shows that directed hypergraphs may be used to model discrete random time-dependent networks and that the minimum expected travel time problem is equivalent to solving the shortest hyperpath problem. Moreover, optimal strategies under different objectives, such as minmax travel time, can also be found by choosing appropriate weights. The shortest hyperpath approach has been proposed for routing problems in discrete random time-dependent networks, where the travel time through a link is a random variable whose distribution depends on the departure time (Miller-Hooks 2001). Problems of this type arise with hazardous materials transportation (Miller-Hooks and Mahmassani 1998) or packet routing in congested communication networks where it is important to allow for real-time routing decisions in response to on-trip events (Miller-Hooks 2001). The bicriterion shortest hyperpath problem has also been investigated. One such application relates to hazmat transportation where both expected travel time and expected risk are minimised.

F-graphs have been studied in the context of urban transit problems (see Nguyen and Pallottino 1988). Transit networks can be represented by a set of nodes corresponding to the stops and links corresponding to in-vehicle line segments, which in turn represent feasible combinations of boarding and alighting stops. A hyperpath represents an attractive set of in-vehicle line segments connecting an origin to a destination. The set of attractive in-vehicle line segments leaving a stop can be grouped to form a hyperlink, with one head node (the boarding stop) and potentially multiple tail nodes (the alighting stops, where the passenger interchanges if the stop is not the destination).
Behind the concept of attractiveness in transit networks are two assumptions. The first is the rule that passengers choose whichever attractive line arrives at the stop first. As a consequence, the attractive in-vehicle line segments are used in proportion to their service frequencies. This means that for each hyperlink the tail node is chosen by each passenger according to the “whichever line arrives first” rule and is therefore used in proportion to the corresponding service frequency. The second is that expected path travel time is minimised. An in-vehicle line segment is therefore only attractive if by excluding it from the choice set (and thus from the hyperlink) the expected travel time would be increased. Spiess and Florian (1989) have shown that, subject to these two assumptions, hyperpaths may be found by solving a linear programming problem.

Hypergraphs and hyperpaths have found applications in many fields, identified in Nielsen et al. (2005).

**Notation**

- \( c_a(t) \) Undelayed travel time on link \( a \) for trips arriving at link \( a \) at time \( t \) (h)
- \( l_a \) Length of link \( a \) (km)
- \( v_a(i) \) Speed on link \( a \) in interval \( i \) (kph)
- \( d_a(t) \) Maximum delay for link \( a \) for trips arriving at link \( a \) at time \( t \) (h)
- \( p_a \) Probability of using link \( a \), interpreted as link attractiveness (primal variable)
- \( q_a \) Pessimistic expectation of maximum delay on link \( a \) (dual variable)
- \( b_i \) 1 if node \( i \) is a destination, -1 if node \( i \) is an origin, and 0 otherwise
- \( w_i \) Pessimistic expectation of delay in arriving at node \( i \), based on the probability of using the links entering that node (primal variable) (h)
- \( u_i \) Pessimistically expected arrival time at node \( i \) (dual variable)
- \( f_i \) Sum of inverse maximum link delays for attractive links entering node \( i \) (see Step 2 of Algorithm \( A_0 \))
- \( y_i \) Probability of using node \( i \)
- \( h_i \) Potential for node \( i \), used to speed up the search for the destination
- \( A \) Set of links
- \( H \) Set of links in the hyperpath from origin \( r \) to destination \( s \)
- \( I \) Set of nodes
- \( A_i^+ \) Set of links entering node \( i \)
- \( A_i^- \) Set of links exiting node \( i \)
Problem formulation

The problem considered here closely resembles the Spiess and Florian (1989) problem, as reinterpreted in Bell (2009), only exposure to maximum delay is minimised on entry to a node rather than on exit from it. The interpretation is that links are subject to an unknown delay up to a predefined maximum. Where there is more than one potentially optimal way (link) to enter node \( i \), we assume that the probability of using each is calculated to minimise the maximum exposure to delay. A potentially optimal link is referred to as an attractive link.

This is a myopic consideration of delay in much the same way that the reinterpreted Spiess and Florian method considers delay myopically. Exposure to delay is minimised on entry to a node rather than over the trip as a whole (the problem of minimising exposure to delay over the trip as a whole in a time-independent context is looked at in Schmoecker et al, 2009). It is conjectured here that the myopic minimisation of exposure to delay coupled with a focus on arrival time is a more natural way of managing uncertainty in the context route guidance, where the traveller is typically concerned about the expected time of arrival. The traveller’s thinking may be expressed as follows: “There may be a delay in arriving at node \( i \) so I will consider the alternatives, and then assign them probabilities of use so as to minimise my maximum exposure to delay”.

Probability of use here may be regarded as a measure of attractiveness. A link or node with a zero probability of use has no possibility of being optimal and is therefore unattractive. Conversely, a link or node with a probability of use equal to one is optimal and therefore has no attractive alternatives. Faced with making a choice, the user of the navigation system, one can presume, would tend to choose the more attractive options but in the knowledge that another option may ex post have been better.

When the expected time of arrival is based on probabilities of use that minimise the maximum exposure to delay myopically, the above principle can be applied recursively. If the expected time of arrival at node \( j \) plus the undelayed travel time from node \( j \) to node \( i \) offers the possibility of arriving earlier at node \( i \) than the expected time of arrival at node \( i \), then the link from \( j \) to \( i \) is attractive. The underlying problem (\( P_0 \)) has the form of the Spiess and Florian (1989) linear program with two differences; link travel times are a function of the pessimistically expected arrival time at the link (the objective function) and link use is calculated to minimise maximum expected delay that would be encountered at the pessimistically expected time of arrival at the link in question:

\[
P_0: \min_{p,w} \left( \sum_{a=(i,j)\in A} p_a c_a(u_i) + \sum_{j\in I} w_j \right)
\]
subject to

\begin{align*}
(1) & \quad \sum_{a=(i,j)\in A^+} p_a - \sum_{a=(j,i)\in A^-} p_a = b_j \text{ for all } j \in I \\
(2) & \quad w_j \geq p_a d_a(u_i) \text{ for all } a = (i,j) \in A_i^+, j \in I \\
(3) & \quad p_a \geq 0 \text{ for all } a \in A
\end{align*}
This formulation of the problem leaves the pessimistically expected time of arrival at each node $i$ undetermined. If $p_a > 0$ then $a = (i, j) \in H$ and

$$ u_i + c_a(u_i) \leq u_j $$

since link $a$ would not be attractive (and therefore not in the hyperpath) if it did not offer the possibility of reaching node $j$ earlier.

Consideration of the following Lagrangean equation makes it clear that $u$ is a dual variable for $P_0$:

$$ L_{p,w,u,q} = \sum_{a=(i,j) \in A} p_a c_a(u_i) + \sum_{j \in I} w_j - \sum_{i \in I} u_i (b_i - \sum_{a=(i,j) \in A^+} p_a + \sum_{a=(j,i) \in A^-} p_a) - \sum_{a=(i,j) \in A} q_a(w_j - p_a d_a(u_i)) $$

$P_0$ is solved when $L_{p,w,u,q}$ is minimised with respect to $p$ and $w$ while being maximised with respect to $u$ and $q$. Equation (5) can be rearranged as follows:

$$ L_{p,w,u,q} = u_r - u_s + \sum_{a=(i,j) \in A} p_a (c_a(u_i) + q_a d_a(u_i) - u_j + u_i) + \sum_{j \in I} w_j (1 - \sum_{a=(i,j) \in A^+} q_a) $$

This leads to the following dual problem

$$ P_1: \max_{u,q} (u_r - u_s) $$

subject to

$$ c_a(u_i) + q_a d_a(u_i) - u_j + u_i \geq 0 \text{ for all } a = (i,j) \in A $$

$$ 1 - \sum_{a=(i,j) \in A^+} q_a = 0 \text{ for all } j \in I - \{r\} $$

$$ q_a \geq 0 \text{ for all } a \in A $$

Without loss of generality, $u_r$ can be set to 0, so according to duality theory at the solution $u_s = \sum_{a=(i,j) \in A} p_a c_a(u_i) + \sum_{j \in I} w_j$, as the objective of the primal problem is equal to minus the objective of the dual problem. Hence at the solution $u_s$ is the pessimistically expected cost of reaching the destination. Equation (7) is equivalent to (4) since

$$ q_a d_a(u_i) \geq 0 \text{ for all } a = (i,j) \in A $$

**Goal oriented search**

It is well known that by transforming link costs through the subtraction of a node potential for the entry node and the addition of a node potential for the exit node for the cost of each link it is possible to speed up Dijkstra’s algorithm when searching for the shortest path from one origin to one destination. The transformation is shown below.

$$ c'_a(u_i) = c_a(u_i) - h_i + h_j \text{ for all } a = (i,j) \in A $$
The resulting algorithm is known as the A* algorithm (Hart et al, 1968). Without loss of generality, we set \( h_s \) equal to 0, where node \( s \) is the destination. It has been shown that in order for the node potential to be \textit{feasible}, it must be less than or equal to the remaining distance (or in this context travel time) to the destination. If the node potential is not feasible, the shortest path may not be found. The closer the node potential is to the remaining distance, the greater the speed-up of the search.

Bell (2009) has shown that node potentials may also be used to speed up the search in the Spiess and Florian (1989) algorithm. This discovery is made use of here.

**Reverse time-dependent Hyperstar algorithm**

As will be proved later, the following algorithm solves the problem:

<table>
<thead>
<tr>
<th>A_0</th>
<th>Reverse time-dependent Hyperstar algorithm</th>
</tr>
</thead>
</table>
| 1. Initialisation | \( u_i \leftarrow \infty, i \in I \setminus \{r\}, u_r \leftarrow 0; \)  
                     | \( f_i \leftarrow 0, i \in I; \)  
                     | \( y_i \leftarrow 0, i \in I \setminus \{s\}, y_s \leftarrow 1; \)  
                     | \( L \leftarrow A; \)  
                     | \( H \leftarrow \emptyset \) |
| 2. Select link a | Find \( a = (i,j) \in L \) with minimum \( u_i + c_a(u_i) + h_j \);  
                    | \( L \leftarrow L \setminus \{a\} \) |
| 3. Update node i | If \( u_j \geq u_i + c_a(u_i) \) then  
                        | if \( u_j = \infty \) and \( f_j = 0 \) then \( \beta \leftarrow 1 \) else \( \beta \leftarrow f_j u_j \),  
                        | \( u_j \leftarrow (\beta + \frac{1}{d_a(u_i)} (u_i + c_a(u_i))) / (f_j + \frac{1}{d_a(u_i)}) \),  
                        | \( f_j \leftarrow f_j + \frac{1}{d_a(u_i)} \) and  
                        | \( H \leftarrow H \cup \{a\}; \)  
                        | if \( L = \emptyset \) or \( u_i + c_a(u_i) > u_s \) then go to Step 3 else go to Step 1 |
| 4. Loading | For every link \( a \in A \) in decreasing order of \( u_i + c_a(u_i) + h_j \),  
                        | if \( a = (i,j) \in H \) then \( p_a \leftarrow \frac{1}{f_j d_a(u_i)} y_j \) and \( y_i \leftarrow y_i + p_a \) else \( p_a \leftarrow 0 \) |

Note that to avoid overflow errors this algorithm works only for positive link delays.
Assumption 1: The node potentials satisfy the triangular inequality property

\[(12) \quad h_i \leq c_a(u_i) + h_j \text{ for all } a = (i, j) \in A\]

In this case the potentials are said to be feasible (Wagner and Willhalm, 2006). The implication of this assumption is that \(h_i - h_s\) is less than or equal to the length of the trip from node \(i\) to destination \(s\). Without loss of generality, we can set \(h_s = 0\), so feasibility requires that \(h_i\) be less than or equal to the remaining distance (or travel time) from node \(i\) to the destination.

Assumption 2: \(u_i > u'_i\) implies \(u_i + c_a(u_i) > u'_i + c_a(u'_i)\). The implication of this assumption is that travel time cannot fall faster than real time, or in other words, it is not possible to enter a link later and exit it earlier. This condition is frequently referred as FIFO (first in, first out). This corresponds to the FIFO condition of Kaufman and Smith (1993).

Proposition 1: At the point at which \(a = (i, j) \in A\) is selected, \(u_i\) is reduced to its final value.

Proof 1: The proof is by induction. For all nodes \(i \in I - \{r\}\), \(u_i\) is set to \(\infty\) (in practice a large number) in Step 0 and then either left unchanged or reduced in Step 2. Suppose that \(u_i\) is greater than its final value. There must be a path to \(i\) not so far considered offering the possibility of a lower travel time. Without loss of generality, suppose this path is via \(a' = (i', i) \in A\) and \(a' \notin H\) (the set of links in the hyperpath). Since \(u_{i'} + c_{a'}(u_{i'}) < u_i\) and as a result of Assumption 1, \(h_i \leq c_a(u_i) + h_j\), so \(u_{i'} + c_{a'}(u_{i'}) + h_i \leq u_{i'} + c_{a'}(u_{i'}) + c_a(u_i) + h_j < u_i + c_a(u_i) + h_j\). This implies that in Step 2, link \(a'\) rather than link \(a\) would have been selected. Since link \(a'\) was not selected, \(u_{i'} + c_{a'}(u_{i'})\) has not been reduced to its final value. By Assumption 2, this implies that \(u_{i'}\) is also greater than its final value. This argument can be repeated until we reach origin \(r\) and conclude that \(u_r = 0\) is greater than its final value, which is clearly untrue. QED

In order to solve \(P_1\), Algorithm \(A_0\) must minimise \(u_s\). Proposition 2 shows that upon termination, \(u_s\) is minimised.

Proposition 2: Upon termination, Algorithm \(A_0\) has minimised \(u_s\).

Proof 2: According to Step 3, Algorithm \(A_0\) terminates when there are no more links to be selected \((L = \emptyset)\) or when the selected link offers no possibility of arriving at the destination earlier \((u_i + c_a(u_i) > u_s)\). In either case, \(u_i\) cannot be minimised further. From Proposition 1 we know that \(u_i + c_a(u_i)\) is minimised when link \(a = (i, j)\) is selected. Since \(h_s = 0\), then if \(u_{i'} + c_{a'}(u_{i'}) \leq u_s\) for any link \(a' = (i', s) \in A \setminus H\), then this link would be selected and added to \(H\). QED

Propositions 1 and 2, however, are not sufficient to ensure that Problem \(P_1\) has been solved by Algorithm \(A_0\), as constraints (8) and (9) must also be satisfied. Proposition 3 shows that constraints (8) and (9) do indeed hold.

Proposition 3: Algorithm \(A_0\) satisfies (8) and (9).

Proof 3: If there is only one link \(a = (i, j) \in H\) entering node \(j\), then by Step 3, \(u_i = c_a(u_i) + d_a(u_i) + u_i\) and \(q_a = 1\). Suppose link \(a = (i, j)\) and \(a' = (i', j)\) are in \(H\), so \(p_a > 0\) and \(p_{a'} > 0\). By (7) we see that \(c_a(u_i) + q_a d_a(u_i) + u_i = c_{a'}(u_{i'}) + q_{a'} d_{a'}(u_{i'}) + u_{i'}\). Step 3 of Algorithm \(A_0\) ensures that \(q_a \geq 0\) and \(q_{a'} \geq 0\) exist such that (8) holds. If this were not so, then
without loss of generality \( c_a(u_i) + d_a(u_i) + u_i < c_{a'}(u_{i'}) + u_{i'} \) would be possible. Step 3, however, ensures that this cannot arise. QED

Note that Proposition 2 requires Proposition 1, which in turn requires Assumptions 1 and 2. At the solution to \( P_1 \) according to duality theory \( P_0 \) is also solved with \( u_s = \sum_{a=(i,j) \in A} p_a c_a(u_i) + \sum_{j \in I} w_j \). Although Algorithm \( A_0 \) does not explicitly calculate \( q_a, a \in A \), Proposition 3 ensures that values satisfying (8) and (9) exist.

Also note that not all potentially optimal paths are included in \( H \), the hyperpath. This is illustrated in Fig. 1. In the brackets, the undelayed travel time and the maximum delay is given for each link. In the absence of any goal orientation \( h_i = 0 \) for all \( i \in I \), the link from node 1 to 2 is selected first, then the link from node 1 to 3, and finally the link from node 2 to 4, leading to the hyperpath indicated by bold arrows. There is only one elemental path from node 1 to 4 in the hyperpath, namely the path through node 3, despite the fact that the path through node 2 is potentially faster, with a travel time of 4 in the absence of delay.

![Fig. 1: Example network (for each link, the first number represents undelayed travel time and the second the maximum delay)](image)

Minimising exposure to maximum delay

The proposed approach may be categorised as a robust approach, in the sense that uncertainty is represented by an interval. For example, the intervals in Fig. 1 are \([2, 7]\) for the link from node 1 to 2, etc. The implication of constraint (2) in \( P_0 \) is that link usage on the approach to every node in the hyperpath is chosen so as to minimise the exposure to maximum delay. The following proposition ensures that if link \( a \) is used, and therefore node \( j \) is in the hyperpath, then \( p_a d_a(u_i) = w_j > 0 \) where \( a = (i, j) \).

**Proposition 4:** If \( p_a > 0 \) then \( p_a d_a(u_i) = w_j > 0 \).
Proof 4: If link $a = (i, j)$ is the only link in the hyperpath leading into node $j$, then $p_a > 0$ and $w_j = p_a d_a (u_i)$. If link $a' = (i', j)$ is also in the hyperpath, then $p_{a'} > 0$ too and Step 4 of Algorithm $A_0$ ensures that $p_a d_a = p_{a'} d_{a'}$. As $w_j$ is minimised, $w_j = p_a d_a = p_{a'} d_{a'} > 0$. QED

Ensuring FIFO

As already mentioned, there is a necessary assumption that link travel time cannot fall faster than real time (Assumption 2). This is equivalent to the FIFO condition, as it implies that it is not possible to delay entry to a link in order to exit it earlier. One way to ensure that this condition is met while allowing link travel times to be time-dependent is to suppose that each link has a speed profile which defines the speed of all vehicles on the link at any point in time (see Sung et al, 2000). Since at any point in time all vehicles are travelling at the same speed, overtaking is ruled out. Fig. 2 shows an illustrative link speed profile. The vehicle is shown arriving at the link entry in the 7th interval at $t_{entry}$. Until time $t_i$ it travels at speed $v_a(7)$. Thereafter it travels at the slower speed of $v_a(8)$ until time $t_{i+1}$, after which it travels at the higher speed of $v_a(9)$ until it reaches the end of the link at time $t_{exit}$. Link travel time is then $t_{exit} - t_{entry}$.

Numerical example

To demonstrate the use of the reverse Hyperstar algorithm, we take an 8 node by 8 node grid network. All arcs can be travelled in either direction, leading to 224 directional links. The length of each arc $l_a, a \in A$, is set equal to $1 + R$, where $R$ is a random number in the range 0 to 1 inclusive, and is the same in either direction. The resulting link lengths are shown in Table 1. The origin $r$ is set near the centre of the network to demonstrate the advantages of goal oriented search. The length of each arc is given in Table 1. The travel time on each arc is time-dependent.
Vehicles are assumed to travel at 50 kph for the first 6 minutes of the trip and then at 20 kph thereafter.

As the length of each link is at least 1 unit, it is straightforward to calculate a reasonable value for $h_i$ which underestimates the travel time from node $i$ to destination $s$. For example, $h_{20} = \frac{5}{50} = 0.1$ (2 links down and 3 across, or any other reasonable route, travelled at the higher speed of 50 kph see Fig. 3). This is a form of what is sometimes referred to in the literature as the Manhattan metric (Wagner and Willhalm, 2006).

Two cases are considered. In the first (see Fig. 3), all links are assigned a near zero maximum delay (in practice, a value of 0.0001 hours). As mentioned earlier, a positive delay is required in order to avoid overflow problems. Since all links are reliable, the hyperpath contains only one elemental path shown by the solid arrows (because link travel times were generated randomly to four decimal places, the chance of two paths having the minimum travel time is small). The diagonal number in Fig. 3 shows the time in hours to reach each node on this elemental path. The transition from the higher speed of 50 kph to the lower speed of 20 kph occurs after 0.1 hours (or 6 minutes), namely between nodes 27 and 19. The introduction of node potentials $h$ as defined above into the algorithm reduces the number of links selected to 157 from 217.

Fig. 3: Fastest path with no risk of delay (solid arrows), showing links selected (dashed arrows)
In the second case (see Fig. 4), maximum link delay is assigned a random value equal to \(0.1R\), where as before \(R\) is a random value in the range 0 to 1 inclusive. Some links are now significantly less reliable than others, although in this example maximum link delay is not treated as time-dependent. The hyperpath now contains numerous elemental paths. The non-zero link usage probabilities defining link attractiveness are shown against the respective links in Fig. 4.

When the risk averse driver reaches node 36, two links are attractive, the link to 28 being more attractive than the link to 35, etc. The diagonal numbers against nodes 36 and s show the pessimistically expected travel times taken to reach the corresponding nodes. In comparison to the first case, the increase in the time to reach node 36 is due to an increase in maximum delay from 0.0001 to 0.0882 hours. As there is only one attractive route to node 36 and the driver is pessimistic, the maximum delay is added to link travel time to obtain the pessimistically expected arrival time. Hence the transition from the higher to the lower speed now occurs on the first link. The expected time to reach the destination has approximately doubled from 0.3857 to 0.7536 hours, in part due to the assumed lower speed now starting at node 36 and in part due to the allowance for delay. The introduction of \(h\) in case 2 leads to a reduction in the number of links selected before termination to 191 from 218.

The advantage of this reverse hyperpath algorithm over single route algorithms is that both time dependency and reasonable detours are taken into account at the route generation stage. The best route in the assumed absence of delays may expose the driver to too high a risk of delay. On the other hand, assuming that every link suffers maximum delay would be too cautious as it neglects the benefits of detouring when the driver receives congestion information in route. Note that when maximum delay is considered, it is more attractive to turn right at node 36 than to continue to node 35 since the maximum delay to arrival at node 35 is 0.0772 hours (a few minutes) compared to 0.0054 hours (a matter of seconds) for node 28.

In practice, maximum delays will rarely be realized, so the driver will typically arrive at each node before the pessimistically estimated arrival time. In this case, the hyperpath search is repeated upon arrival at each node using the actual arrival time rather than the pessimistically expected arrival time. In this context, the use of node potentials to speed up the search becomes particularly important. The amount of acceleration gained by the inclusion of node potentials in this numerical example is relatively modest, in part due to the extent of the underestimate of remaining travel time and in part due to the relatively small size of the numerical example. The estimated time per link is only 0.02 hours, leading to an estimated travel time of only 0.16 hours. Halving of the assumed speed of travel to 25 kph when calculating \(h\) leads to a reduction in the number of links selected in near absence of delay (case 1) from 157 to 69, but unfortunately the algorithm terminates prematurely without finding the fastest path. This is because the actual speed of travel on a significant number of links near the origin is 50 kph, leading to an over-estimate of link travel time in this region.
Conclusions

This paper builds on earlier work published by Bell (2009). In recognition of the prevalence of time-of-day variation in link travel times, an algorithm is presented for building hyperpaths in time-dependent networks from the origin to the destination. Node potentials are included in order to speed up the search, a requirement for vehicle navigation applications, particularly where the search must be repeated upon arrival at each node. The resulting algorithm, referred to in the title as the time-dependent reverse Hyperstar algorithm, may be viewed as the hyperpath version of the Chabini and Lan (2002) algorithm.

Although this algorithm has been presented in the context of risk averse vehicle navigation on road networks, it could also be applied to time tables, opening up the possibility of time-dependent hyperpath searches for transit networks. Allowance for the risk of encountering a maximum delay when connections are missed in transit networks would make the recommendations of public transport journey planners more robust. This is the subject of ongoing research.
Acknowledgements

The authors are grateful for financial support from the UK Engineering and Physical Science Research Council under grant EP/E043569/1 on “Dynamic Hyperpath Search for Transit and Traffic Assignment”. The other members of the Hyperpath Working Group, namely Achille Fonzone, Jinyoung Kim, Fumitaka Kurauchi, Klaus Noekel, Normen Rochau, Jan-Dirk Schmoecker, Hiroshi Shimamoto, Alan Wilson and Rabi Mishalani have contributed to the formulation of the ideas expressed in this paper.

References


Table 1: Link lengths in km

<table>
<thead>
<tr>
<th>From</th>
<th>To</th>
<th>Length</th>
<th>From</th>
<th>To</th>
<th>Length</th>
<th>From</th>
<th>To</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>1.0000</td>
<td>33</td>
<td>34</td>
<td>1.9984</td>
<td>1</td>
<td>9</td>
<td>1.3413</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1.0000</td>
<td>34</td>
<td>33</td>
<td>1.9984</td>
<td>9</td>
<td>1</td>
<td>1.3413</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>1.8610</td>
<td>34</td>
<td>35</td>
<td>1.6762</td>
<td>2</td>
<td>10</td>
<td>1.8170</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1.8610</td>
<td>35</td>
<td>34</td>
<td>1.6762</td>
<td>10</td>
<td>2</td>
<td>1.8170</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>1.2729</td>
<td>35</td>
<td>36</td>
<td>1.0859</td>
<td>3</td>
<td>11</td>
<td>1.0935</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>1.3187</td>
<td>36</td>
<td>37</td>
<td>1.4953</td>
<td>4</td>
<td>12</td>
<td>1.0603</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>1.3187</td>
<td>37</td>
<td>36</td>
<td>1.4953</td>
<td>12</td>
<td>4</td>
<td>1.0603</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>1.3722</td>
<td>37</td>
<td>38</td>
<td>1.5100</td>
<td>5</td>
<td>13</td>
<td>1.6730</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>1.3722</td>
<td>38</td>
<td>37</td>
<td>1.5100</td>
<td>13</td>
<td>5</td>
<td>1.6730</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>1.0820</td>
<td>38</td>
<td>39</td>
<td>1.9532</td>
<td>6</td>
<td>14</td>
<td>1.5704</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>1.0820</td>
<td>39</td>
<td>38</td>
<td>1.9532</td>
<td>14</td>
<td>6</td>
<td>1.5704</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>1.0706</td>
<td>39</td>
<td>40</td>
<td>1.3387</td>
<td>7</td>
<td>15</td>
<td>1.3659</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>1.0706</td>
<td>40</td>
<td>39</td>
<td>1.3387</td>
<td>15</td>
<td>7</td>
<td>1.3659</td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>1.0597</td>
<td>41</td>
<td>42</td>
<td>1.7054</td>
<td>8</td>
<td>16</td>
<td>1.3146</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>1.0597</td>
<td>42</td>
<td>41</td>
<td>1.7054</td>
<td>16</td>
<td>8</td>
<td>1.3146</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>1.9173</td>
<td>42</td>
<td>43</td>
<td>1.7740</td>
<td>9</td>
<td>17</td>
<td>1.6564</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
<td>1.9173</td>
<td>43</td>
<td>42</td>
<td>1.7740</td>
<td>17</td>
<td>9</td>
<td>1.6564</td>
</tr>
<tr>
<td>11</td>
<td>12</td>
<td>1.7747</td>
<td>43</td>
<td>44</td>
<td>1.1977</td>
<td>10</td>
<td>18</td>
<td>1.9030</td>
</tr>
<tr>
<td>12</td>
<td>11</td>
<td>1.7747</td>
<td>44</td>
<td>43</td>
<td>1.1977</td>
<td>18</td>
<td>10</td>
<td>1.9030</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>1.6977</td>
<td>44</td>
<td>45</td>
<td>1.9136</td>
<td>11</td>
<td>19</td>
<td>1.4787</td>
</tr>
<tr>
<td>13</td>
<td>12</td>
<td>1.6977</td>
<td>45</td>
<td>44</td>
<td>1.9136</td>
<td>19</td>
<td>11</td>
<td>1.4787</td>
</tr>
<tr>
<td>13</td>
<td>14</td>
<td>1.7180</td>
<td>45</td>
<td>46</td>
<td>1.5893</td>
<td>12</td>
<td>20</td>
<td>1.7667</td>
</tr>
<tr>
<td>14</td>
<td>13</td>
<td>1.7180</td>
<td>46</td>
<td>45</td>
<td>1.5893</td>
<td>20</td>
<td>12</td>
<td>1.7667</td>
</tr>
<tr>
<td>14</td>
<td>15</td>
<td>1.1626</td>
<td>46</td>
<td>47</td>
<td>1.6894</td>
<td>13</td>
<td>21</td>
<td>1.5520</td>
</tr>
<tr>
<td>15</td>
<td>14</td>
<td>1.1626</td>
<td>47</td>
<td>46</td>
<td>1.6894</td>
<td>21</td>
<td>13</td>
<td>1.5520</td>
</tr>
<tr>
<td>15</td>
<td>16</td>
<td>1.4660</td>
<td>47</td>
<td>48</td>
<td>1.0936</td>
<td>14</td>
<td>22</td>
<td>1.4117</td>
</tr>
<tr>
<td>16</td>
<td>15</td>
<td>1.4660</td>
<td>48</td>
<td>47</td>
<td>1.0936</td>
<td>22</td>
<td>14</td>
<td>1.4117</td>
</tr>
<tr>
<td>17</td>
<td>16</td>
<td>1.8257</td>
<td>49</td>
<td>50</td>
<td>1.2800</td>
<td>15</td>
<td>23</td>
<td>1.5958</td>
</tr>
<tr>
<td>18</td>
<td>17</td>
<td>1.8257</td>
<td>50</td>
<td>49</td>
<td>1.2800</td>
<td>23</td>
<td>15</td>
<td>1.5958</td>
</tr>
<tr>
<td>18</td>
<td>19</td>
<td>1.4818</td>
<td>50</td>
<td>51</td>
<td>1.1612</td>
<td>16</td>
<td>24</td>
<td>1.6689</td>
</tr>
<tr>
<td>19</td>
<td>18</td>
<td>1.4818</td>
<td>51</td>
<td>50</td>
<td>1.1612</td>
<td>24</td>
<td>16</td>
<td>1.6689</td>
</tr>
<tr>
<td>19</td>
<td>20</td>
<td>1.8743</td>
<td>51</td>
<td>52</td>
<td>1.5437</td>
<td>17</td>
<td>25</td>
<td>1.4717</td>
</tr>
<tr>
<td>20</td>
<td>19</td>
<td>1.8743</td>
<td>52</td>
<td>51</td>
<td>1.5437</td>
<td>25</td>
<td>17</td>
<td>1.4717</td>
</tr>
<tr>
<td>20</td>
<td>21</td>
<td>1.7728</td>
<td>52</td>
<td>53</td>
<td>1.1744</td>
<td>18</td>
<td>26</td>
<td>1.1148</td>
</tr>
<tr>
<td>21</td>
<td>20</td>
<td>1.7728</td>
<td>53</td>
<td>52</td>
<td>1.1744</td>
<td>26</td>
<td>18</td>
<td>1.1148</td>
</tr>
<tr>
<td>21</td>
<td>22</td>
<td>1.4925</td>
<td>53</td>
<td>54</td>
<td>1.2681</td>
<td>19</td>
<td>27</td>
<td>1.5486</td>
</tr>
<tr>
<td>22</td>
<td>21</td>
<td>1.4925</td>
<td>54</td>
<td>53</td>
<td>1.2681</td>
<td>27</td>
<td>19</td>
<td>1.5486</td>
</tr>
<tr>
<td>22</td>
<td>23</td>
<td>1.8273</td>
<td>54</td>
<td>55</td>
<td>1.5768</td>
<td>20</td>
<td>28</td>
<td>1.0807</td>
</tr>
<tr>
<td>23</td>
<td>22</td>
<td>1.8273</td>
<td>55</td>
<td>54</td>
<td>1.5768</td>
<td>28</td>
<td>20</td>
<td>1.0807</td>
</tr>
<tr>
<td>23</td>
<td>24</td>
<td>1.1411</td>
<td>55</td>
<td>56</td>
<td>1.2852</td>
<td>21</td>
<td>29</td>
<td>1.8387</td>
</tr>
<tr>
<td>24</td>
<td>23</td>
<td>1.1411</td>
<td>56</td>
<td>55</td>
<td>1.2852</td>
<td>29</td>
<td>21</td>
<td>1.8387</td>
</tr>
<tr>
<td>25</td>
<td>26</td>
<td>1.5008</td>
<td>57</td>
<td>58</td>
<td>1.6098</td>
<td>22</td>
<td>30</td>
<td>1.9270</td>
</tr>
</tbody>
</table>