



ELSEVIER

Transportation Research Part B 38 (2004) 385–413

TRANSPORTATION
RESEARCH
PART B

www.elsevier.com/locate/trb

Adaptive routing considering delays due to signal operations

Baiyu Yang^{a,1}, Elise Miller-Hooks^{b,*}

^a *Operations Research and Decision Support, American Airlines, 4333 Amon Carter Blvd., MD 5358, Fort Worth, TX 76155, USA*

^b *Department of Civil and Environmental Engineering, 212 Sackett Building, The Pennsylvania State University, University Park, PA 16802, USA*

Received 20 October 2000; received in revised form 5 March 2003; accepted 26 March 2003

Abstract

This work addresses the problem of determining optimal routing decisions in signalized traffic networks, where arc travel times vary over time and are known only probabilistically (i.e. in stochastic, time-varying (STV) networks) and additional delay due to signal operations is explicitly considered. While prior works in the literature address problems of routing in STV networks, none explicitly considers the additional delay that would be incurred due to signal operations at the intersections of the roadway network. In this paper, we consider an adaptive routing problem, where paths are adapted en route based on revealed information concerning the arc travel times and actual signal timings. We first discuss how concepts from existing procedures can be combined to solve the adaptive routing problem in signalized STV networks, where the signal timing plan and actual timings are known a priori. When actual timings or delays due to signal control are known only probabilistically, such techniques will be inefficient. Thus, we propose a more efficient algorithm for solving this latter problem. Results of numerical experiments conducted on a real-world-based signalized street network are presented. These results show that the solutions obtained by explicitly considering delays due to signal operations will likely be significantly different from those solutions generated by techniques that ignore such delays.

© 2003 Elsevier Ltd. All rights reserved.

Keywords: Stochastic; Probabilistic; Time-varying; Dynamic; Shortest paths; Hyperpaths; Signal control

* Corresponding author. Tel.: +1-814-863-2634; fax: +1-814-863-7304.

E-mail addresses: baiyu.yang@aa.com (B. Yang), edm3@psu.edu (E. Miller-Hooks).

¹ Tel.: +1-817-931-6307; fax: +1-817-931-9270.

1. Introduction

This work addresses the problem of determining optimal adaptive path decisions in traffic networks, where arc travel times vary over time and are known only probabilistically and additional delay due to signal operations is explicitly considered.

Future travel times in surface transportation networks, such as vehicular traffic networks, can at best be known a priori with uncertainty. This uncertainty is due in part to variations in driver behavior, acceleration of different vehicle types and the occurrence of non-recurrent congestion due to random events, such as accidents along the roadways. Moreover, patterns in average travel times may exist due to recurrent congestion. Thus, in a network representation of the roadway system (where nodes represent intersections and arcs represent streets connecting the intersections), the arc weights (representing travel times) are given as random variables with probability distribution functions (PDFs) that change over time. Such a network is referred to as a stochastic, time-varying, or STV, network (Miller-Hooks and Mahmassani, 2000). In such a STV network, for a given origin–destination (O–D) pair, one can choose the a priori least expected time (LET) path before travel begins. Exact, but inefficient, procedures were proposed by Hall (1986) and Miller-Hooks and Mahmassani (2000), and an efficient but inexact approach was proposed by Fu and Rilett (1998) for addressing this problem. However, in STV networks, better solutions may result if the path decisions are adapted en route based on experienced travel times on traveled arcs. That is, for example, as the random travel times are revealed, and hence the arrival times at intermediate locations are learned, one can make a more informed decision about how to proceed from the intermediate location toward the destination. Since the travel time on the arcs, and hence the arrival time at each intermediate location, cannot be known before the initial departure from the origin, a better route can be selected by deferring these intermediate path selection decisions until the time at which these nodes are reached.

Recognizing this issue, Hall (1986) formulated the problem of determining the LET path between a single origin and destination in a STV transit network as a dynamic programming problem. He assumed that the random arc travel times have continuous PDFs. This dynamic programming approach is inefficient and is not guaranteed to lead to the optimal solution unless a sufficient number of stages is specified a priori. Miller-Hooks and Mahmassani (2000) proposed an efficient specialized label-correcting algorithm, the expected lower bound (ELB) algorithm, for determining adaptive LET paths in STV networks. Acyclic subnetworks, referred to as hyperpaths, are used to represent the adaptive decisions. These hyperpath solutions provide the optimal next move for each possible arrival time. Thus, upon arrival at a node, and upon learning one's arrival time at that node, one would choose the optimal next move from the hyperpath solutions that corresponds to that location and arrival time. An efficient specialized label-setting algorithm, the stochastic decreasing order of time (SDOT) algorithm, was proposed by Miller-Hooks (2001) for solving the same adaptive problem. See Ahuja et al. (1993) for additional detail on generic labeling algorithms for solving the classical shortest path problem (i.e. in deterministic and time-invariant networks).

While these prior works in the literature address problems of either a priori or adaptive routing in STV networks, none explicitly considers the additional delay that would be incurred due to signal operations at the intersections of the roadway network. One might model this additional delay by incorporating an average delay in the form of a penalty into the arc travel times and then directly use one of these existing techniques. However, in STV networks, the results of employing

such an approach would be erroneous, because the expected times of the LET hyperpaths would be incorrectly computed. Rather than using the arc travel time PDFs at each possible arrival time at the intermediate locations, such computations would employ the arc travel time PDFs at an average arrival time, which may not even be a possible arrival time. A similar argument is made to show that one cannot compute the expected travel time of a path in a STV network by setting each arc weight to its expected value and solving an equivalent deterministic, time-dependent least time path problem (see Hall, 1986; Miller-Hooks and Mahmassani, 2000). Thus, solution techniques that employ such an approach for modeling delay due to signal operations will result in suboptimal solutions. Since, in such a signalized traffic system, delay due to signal control often accounts for a great proportion of the total journey time, a technique that properly accounts for the additional delay due to signal operations is warranted.

A number of works account for turning delays and prohibitions when computing shortest paths in deterministic and time-invariant networks (i.e. where all quantities, including travel times, are static and are exactly known). Wattleworth and Shuldiner (1963) proposed an approach where each intersection is expanded to a subnetwork consisting of eight nodes and as many as 16 arcs. This network expansion requires tremendous additional computation time and storage. An equivalent, but more efficient, method was proposed by Ziliaskopoulos and Mahmassani (1996). In their work, instead of modeling intersection delay in the arc travel times of the subnetwork that is used to represent a signalized intersection, penalties are assigned to intersection movements to account for these delays. A label-correcting algorithm was given, where the penalties are incurred in the label computation. In more recent works, this penalty concept has been adopted to model periodic traffic signals. Chen and Yang (2000) proposed an approach to calculate the intersection penalties given periodic traffic signal control and gave a label-setting version of the Penalty Approach proposed by Ziliaskopoulos and Mahmassani to calculate the least time path in such a network. In a similar context, Ahuja et al. (2002) showed that the time-dependent least time path problem in traffic networks with periodic traffic signal control can be solved in polynomial time as a consequence of the fact that its “auxiliary” network maintains the FIFO property. Unlike the minimum time problem, they showed that when the objective is to minimize cost (a non-time based measure), the problem is generally NP-hard. This is because the problem reduces to a minimum cost path problem with hard time-windows, a problem that is known to be NP-hard (Desrochers and Soumis, 1988).

In this paper, we consider the problem of determining the adaptive LET hyperpaths in signalized STV networks. Paths are adapted en route based on revealed arrival times at intermediate locations arising from revealed information concerning arc travel times and actual signal timings. We first show that in the case where actual signal timings are known a priori and there is no additional uncertain delay at the signalized intersections the delay due to signal control can be modeled as penalties associated with intersection movements. Thus, the method proposed by Ziliaskopoulos and Mahmassani (1996) for modeling turning delays and prohibitions through the use of penalties in deterministic networks and that proposed by Miller-Hooks and Mahmassani (2000) for determining the LET hyperpaths in unsignalized STV networks can be combined and extended to solve the adaptive routing problem in signalized STV networks, where the signal timing plan and actual timings are known a priori. We refer to the resulting procedure as the Penalty Approach. This case is of interest, for example, when the network consists of pretimed signals and the decision-maker is given complete knowledge of the signal timing plan (i.e. actual timings are known), but travel times are uncertain and time-varying quantities.

One can foresee many situations where actual signal timings, or more generally, the added travel delay due to signal operations, might only be known probabilistically. Even if a decision-maker has perfect information of the signal timings, and even if one could perfectly estimate his/her arrival time at an intersection, there may be uncertainty in the amount of time it would take to pass through the intersection due to the effects of queuing and spillback. For example, delay due to queue clearance lost time at an intersection is a function of the vehicle's location in the queue. Since the position in the queue cannot be known with certainty, one can only know the delay incurred at a signalized intersection probabilistically, even if the actual signal timings are known precisely. These lost times could be built into the duration of the red phase by (probabilistically) increasing the penalty or duration of the red phase beyond the actual time of the phase itself. Note that none of the works that address turning delays and prohibited movements in deterministic networks discussed previously considers such uncertain quantities as start-up or queue clearance lost times at the signalized intersections.

Vehicle-actuated signals provide an additional example of the need to consider probabilistic delays due to signal operations. In this case, the cycle lengths will be constant for a given period and the percent of time allocated to each of the phases will vary as a function of the random arrival of vehicles at the corresponding intersections. Thus, even if a decision-maker has perfect information concerning the cycle lengths and signal timing plans, the duration and timing of each phase within a cycle could not be known precisely a priori.

When actual signal timings or delays due to signals are known only probabilistically to the decision-maker, such penalty-based techniques as the Penalty Approach given herein or the approach of expanding a node representing a signalized intersection will be inefficient, and thus, the development of a new method is warranted. We propose a more efficient specialized label-correcting algorithm, the adaptive routing with signal control (ARSC) algorithm, for this problem of determining the adaptive LET hyperpaths in signalized STV networks, where actual signal timings or added delays due to signal operations are known only probabilistically.

A benefit of considering adaptive techniques is that one can exploit the signal timing information regardless of the level of variability in the arc travel times. Since one cannot accurately predict the arrival times at the intermediate nodes en route to the destination when determining an a priori LET path in a signalized network, the delay at these intermediate locations may be computed incorrectly, even if the signal timings are known exactly. However, in an adaptive approach, such imprecise computations are avoided due to the fact that no assumption about the arrival times at intermediate locations is made until travel has been completed. Thus, the correct durations of the delays due to signal control are used in computing the expected hyperpath times.

Numerical experiments were conducted on a real-world-based signalized street network. The results of these experiments, as discussed in Section 5, show that the solutions obtained by explicitly considering delays due to signal operations are often significantly different from those solutions generated by techniques that ignore such delays. Thus, these results demonstrate the necessity for considering signal timings when determining optimal routing decisions in signalized street networks.

2. Notation and definitions

Let $G = (V, A)$ be a directed network, where V is the set of nodes, $|V| = n$, and A is the set of arcs, $|A| = m$. $V = V_1 \cup V_2$, where V_1 is the set of nodes representing intersections with signal

control and V_2 is the set of remaining nodes representing unsignalized intersections. In this paper, a node in V_1 is called a signalized node and a network representation of a traffic system with signal control is called a signalized network. The travel times on each arc are assumed to be discrete random variables with time-varying probability mass functions (PMFs). The period of interest, referred to as the “peak period”, is represented as a set S of discrete departure times, $\{1\delta, 2\delta, \dots, T\delta\}$, where δ is the minimum time unit during which perceptible changes in travel conditions can occur, and T is the largest index of the departure times for the peak period. For departure time $t \in S$, the travel time PMF of each arc (i, j) is given as a set of K possible travel times, $[\tau_{i,j}^k(t)]_{k \in \{1,2,\dots,K\}}$, with corresponding probabilities, $[\rho_{i,j}^k(t)]_{k \in \{1,2,\dots,K\}}$. It is assumed that, for each $k \in \{1, 2, \dots, K\}$ and $(i, j) \in A$, $\tau_{i,j}^k(t) = \tau_{i,j}^k(T\delta)$ and $\rho_{i,j}^k(t) = \rho_{i,j}^k(T\delta)$ for all t occurring after the peak period, i.e. $\forall t > T\delta$. As required in related previous works, the arc traversal times are assumed to be at least as large as the time interval, δ . Without loss of generality, in our discussion, we assume that each departure time interval is of unit duration. Furthermore, the arc traversal time is defined (via a departure time) upon entering the arc and is assumed to be constant for the duration of travel along that arc.

The objective of this work is to develop an efficient solution technique for determining the adaptive LET hyperpaths in signalized STV traffic networks, including additional delays due to signal operations, from all origins to a given destination, D . It is assumed that waiting is permitted (and required) only at signalized intersections at times when the traffic signal governing a movement of interest is red. Finally, the signal cycle consists of only red and green time. Amber time is implicitly considered to be included in the green time. Delays due to deceleration for a red signal or start-up and queue clearance lost times for a green signal are not explicitly considered herein, but can be modeled by extending the delay (or penalty) incurred during a red signal phase.

3. The Penalty Approach

3.1. Modeling signal control

The label-correcting algorithm proposed by Ziliaskopoulos and Mahmassani (1996) for addressing the classical shortest path problem with restricted movements and prohibitions differs from most shortest path algorithms in two regards. First, a penalty is associated with each intersection movement and is used in the label computation. Second, multiple labels are maintained at each node, each of which provides the least time from the origin to the node for a given subsequent movement. In this section, we show how the delay due to signal control can be transformed to penalties for intersection movements. Accordingly, a penalty-based approach that employs the concepts developed in (Ziliaskopoulos and Mahmassani, 1996) is proposed for determining the adaptive LET hyperpaths in signalized STV networks where actual signal timings are known.

At a signalized intersection, traffic streams are temporally separated to avoid or reduce conflict. A signal phase defines a part of a signal cycle allocated to one or more traffic streams simultaneously receiving identical signal indications. A particular movement may only be allowed within a particular phase of a signal cycle. If a particular movement is not permitted in the signal phase

present at the time of arrival, a motorist may not be able to continue traveling until the beginning of the phase in which the movement is permitted. The time between the moment of arriving at the intersection and the beginning of the next phase, in which the movement intended by the motorist is allowed, is called the penalty for this movement. It is assumed in calculating the penalty that there is no spillback effect from downstream intersections and that the vehicle will not have to wait at an intersection through more than one signal cycle. This issue is revisited in Section 4. For a given movement, the quantity of the penalty depends on the arrival time at the intersection. For each movement, the penalties incurred at all arrival times in the entire peak period constitute a penalty vector. For each outgoing arc from the intersection, several penalty vectors exist because different movements involving this arc are allowed in different phases. As a signal cycle repeats itself over time, so do the penalties. The following example is used to illustrate the process of transforming the delay due to signal control into penalty vectors.

A four-approach intersection and a feasible assignment of movements to signal phases are shown in Fig. 1. Phases a , b , c , and d have durations of 3, 3, 2, and 2 time units, respectively, and proceed in alphabetical order. Suppose phase a starts at time 1 and a traveler enters the intersection from the west at time 4 (first time unit of phase b) and plans to continue eastbound (movement 1). This movement is only permitted during phase a . The penalty of this movement for departure time 4 would be the time between time 4 and the beginning of the next phase a , time 11, i.e. 7 time units. The penalty vectors for all movements are shown in Table 1.

The Penalty Approach is presented next for solving the adaptive routing problem in signalized STV networks with known signal timings (and deterministic delays due to signal operations).

3.2. The Penalty Approach for use in signalized STV networks

Let ζ_i be the number of phases at a signalized intersection represented by a node $i \in V_1$. Let $\Gamma^{-1}(i)$ denote the set of predecessor nodes of node i , where element $j : (j, i) \in A$. Similarly, let $\Gamma^{+1}(i)$ denote the set of successor nodes of node i , where element $j : (i, j) \in A$. For each node $i \in V_1$, denote ε_i^l as the l th phase, $l \in [1, \zeta_i]$. For an arc (i, j) and departure time $t \in S$, the penalty associated with an intersection movement when the signal is in phase ε_i^l is denoted $\psi_{ij}^{\varepsilon_i^l}(t)$, $\forall l \in [1, \zeta_i]$, $i \in V_1$, $j \in V$. In order to guarantee that the adaptive routing problem has a finite solution when travel cannot be completed before $T\delta$, it is assumed that all movements are permitted for the last departure time or thereafter, i.e. $\psi_{ij}^{\varepsilon_i^l}(T\delta) = 0$, $\forall l \in [1, \zeta_i]$, $i \in V_1$, $j \in V$. To map a movement from arc (h, i) to arc (i, j) ($(h, i) \rightarrow (i, j)$) to a signal phase ε_i^l and associated penalties, a function $R(h, i, j)$ is defined for each such pair of directed arcs (h, i) and (i, j) . Specifically, $R(h, i, j)$ is set to the index of the phase for which the movement $(h, i) \rightarrow (i, j)$ is permitted and, in turn, implicitly specifies the penalty incurred for this movement. For example, for movement 1 at the intersection shown in Fig. 1, if we let node i represent this intersection and let nodes h and j denote the previously visited intersection and the next intersection to visit, respectively, then $R(h, i, j) = 1$, $\varepsilon_i^1 = a$, and the penalty associated with this movement at departure time 4 is $\psi_{ij}^a(4) = 7$, as shown in Table 1. For all nodes in V_1 without any incoming arcs or without any outgoing arcs and all nodes in V_2 , $R(h, i, j)$ is set to one and the penalties for movements at intersections represented by these nodes are assumed to be zero. To account for a movement whose origin is node i , inferring that $h = i$, $R(h, i, j)$ is set to zero with corresponding penalties of zero.

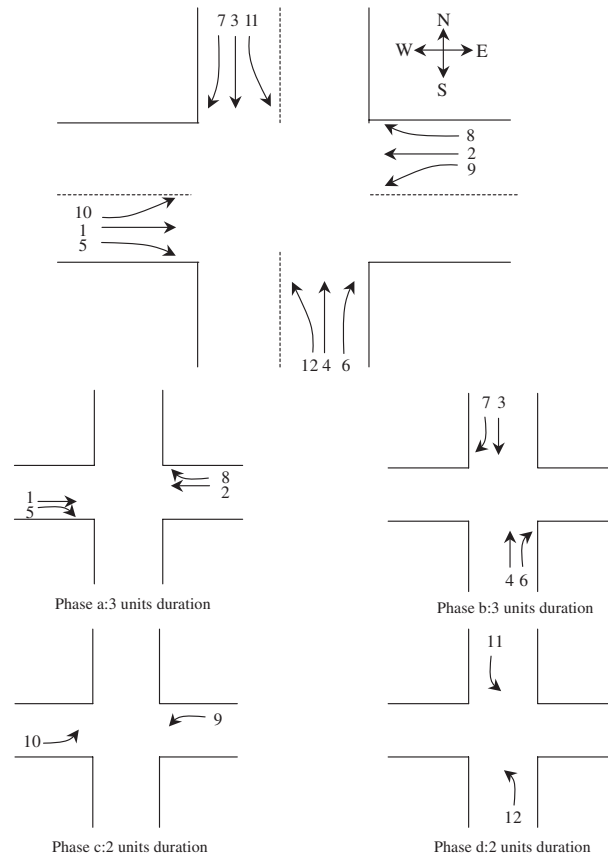


Fig. 1. Movements at a four-approach intersection and a feasible assignment of movements to phases.

Table 1
Signal phases and corresponding penalties for all intersection movements

Time interval	Signal phase	Penalty for movements allowed in phase <i>a</i> (units)	Penalty for movements allowed in phase <i>b</i> (units)	Penalty for movements allowed in phase <i>c</i> (units)	Penalty for movements allowed in phase <i>d</i> (units)
1	a	0	3	6	8
2	a	0	2	5	7
3	a	0	1	4	6
4	b	7	0	3	5
5	b	6	0	2	4
6	b	5	0	1	3
7	c	4	7	0	2
8	c	3	6	0	1
9	d	2	5	8	0
10	d	1	4	7	0
11	a	0	3	6	8
–	–	–	–	–	–

The Penalty Approach can be viewed as a specialized label-correcting algorithm that builds on the ELB algorithm of Miller-Hooks and Mahmassani (2000) and generalizes the penalty concept presented by Ziliaskopoulos and Mahmassani (1996) for additional use in STV networks. A vector label, $\Lambda_i^h = \{\lambda_i^h(t)\}_{t \in S}$, and a vector pointer, $\Pi_i^h = \{\pi_i^h(t)\}_{t \in S}$, are associated with each node i and each predecessor h of i , including $h = i$. Upon termination of the Penalty Approach, each element $\lambda_i^h(t) \in \Lambda_i^h$, at a time $t \in S$, gives the exact LET for the adaptive routing problem from node i to the destination for departure time t given that the motorist arrives at node i by way of node h (note: for travel commencing at node i , $h = i$). It is assumed that, $\forall t > T\delta$, $\lambda_i^h(t) = \lambda_i^h(T\delta)$ for all nodes $i \in V$ and corresponding predecessor nodes h . Furthermore, $\lambda_i^h(t + \vartheta) = \lambda_i^h(t)$ for $0 < \vartheta < \delta$. Each element $\pi_i^h(t) \in \Pi_i^h$ indicates the next node to visit from node i at departure time t , given predecessor h . These pointers are used to track the hyperpaths. Before termination, $\lambda_i^h(t)$ remains an upper bound to the desired value. At each iteration of the Penalty Approach, temporary label components, denoted $\eta_i^h(t)$, are constructed at each predecessor node of the currently scanned node. The value of the label component $\eta_i^h(t)$ is computed as

$$\eta_i^h(t) = \sum_{k=1}^K [(\psi_{ij}^{e_l^i}(t) + \tau_{ij}^k(t + \psi_{ij}^{e_l^i}(t)) + \lambda_j^i(t + \psi_{ij}^{e_l^i}(t) + \tau_{ij}^k(t + \psi_{ij}^{e_l^i}(t)))) \cdot \rho_{ij}^k(t + \psi_{ij}^{e_l^i}(t))],$$

where $\psi_{ij}^{e_l^i}(t)$ is the penalty for the movement $(h, i) \rightarrow (i, j)$ at time t , $\tau_{ij}^k(t + \psi_{ij}^{e_l^i}(t))$ is the k th possible travel time on arc (i, j) at time $t + \psi_{ij}^{e_l^i}(t)$ with associated probability $\rho_{ij}^k(t + \psi_{ij}^{e_l^i}(t))$, and $\lambda_j^i(t + \psi_{ij}^{e_l^i}(t) + \tau_{ij}^k(t + \psi_{ij}^{e_l^i}(t)))$ is the label at node j for the arrival time given parenthetically. $\eta_i^h(t)$ is compared with the most recent value of the label component $\lambda_i^h(t)$. If the temporary label, $\eta_i^h(t)$, has a lower value than the current label, $\lambda_i^h(t)$, then $\lambda_i^h(t)$ is set to the value of the temporary label. The steps of the Penalty Approach for solving the adaptive routing problem in signalized STV networks are presented next. The penalties, $\psi_{ij}^{e_l^i}(t)$, $\forall (i, j) \in A$, $i \in V$, $t \in S$, $l \in [1, \zeta_i]$, are set prior to running the algorithm.

Step 0: Initialization

Initialize the node labels.

$$\lambda_i^h(t) = \infty, \forall i \in V \setminus D, t \in S, \forall h \in \{\Gamma^{-1}(i), i\}.$$

$$\pi_i^h(t) = \infty, \forall i \in V \setminus D, t \in S, \forall h \in \{\Gamma^{-1}(i), i\}.$$

$$\lambda_D^h(t) = 0 \text{ and } \pi_D^h(t) = 0, \forall t \in S, h \in \{\Gamma^{-1}(D), D\}.$$

Initialize the scan eligible (SE) list and insert the destination node D .

Step 1: Choose the current node from the SE list

If the SE list is empty, go to step 3. Otherwise, select the first node j from the SE list.

Step 2: Update labels

For each $i \in \Gamma^{-1}(j)$

For each $h \in \{\Gamma^{-1}(i), i\}$

$$l = R(h, i, j).$$

For each $t \in S$

$$\eta_i^h(t) = \sum_{k=1}^K [(\psi_{ij}^{e_i^k}(t) + \tau_{ij}^k(t + \psi_{ij}^{e_i^k}(t))) + \lambda_j^i(t + \psi_{ij}^{e_i^k}(t) + \tau_{ij}^k(t + \psi_{ij}^{e_i^k}(t))) \cdot \rho_{ij}^k(t + \psi_{ij}^{e_i^k}(t))]. \quad (1)$$

If $\eta_i^h(t) < \lambda_i^h(t)$, then $\lambda_i^h(t) = \eta_i^h(t)$ and $\pi_i^h(t) = j$. $SE = SE \cup \{i\}$.

If all $i \in \Gamma^{-1}(j)$ have been considered, go to step 1.

Step 3: Stop

Optimality and complexity of the Penalty Approach are addressed in the following two propositions. The proofs are omitted due to the similarity with those given by Miller-Hooks and Mahmassani (2000) for the complexity and optimality of the ELB algorithm upon which this procedure is based.

Proposition 1. *The Penalty Approach terminates with the set of adaptive LET hyperpaths for the adaptive routing problem in signalized STV networks with known signal timings.*

Later we will show that the ARSC algorithm presented in Section 4.2 reduces to the Penalty Approach in the case where signal timings are known a priori. We will prove optimality of the ARSC algorithm for a more general case in Section 4.2.2, inferring optimality of the Penalty Approach presented in this section.

Proposition 2. *The Penalty Approach with a basic FIFO SE list has worst-case computational complexity $O(n^4 T^2 K)$, where n is the number of nodes, T is the number of departure time intervals and K is the number of possible travel times for each departure time.*

The ELB algorithm is $O(n^3 T^2 K)$ in the worst-case computational complexity (Miller-Hooks and Mahmassani, 2000). The additional $O(n)$ required in the Penalty Approach is due to the extra loop over predecessor nodes which is not required in the ELB algorithm.

4. The adaptive routing problem with probabilistically known signal timings

4.1. Inherent difficulties of employing the Penalty Approach for the case of uncertain signal timings

Thus far, we have assumed a priori deterministic knowledge of the penalties associated with traffic movements at signalized intersections. Such deterministic knowledge results from complete knowledge of the actual signal timings. The knowledge of the actual signal timings or the added delay due to signal operations may not be known with certainty, as might be the case, for example, where signals are vehicle-actuated, and thus, the duration of each phase depends on the random arrival of vehicles at the intersection. When this knowledge is uncertain, there will be many possible penalties that may be incurred, depending on the possible actual timings or added delays. In computing the label components, all possible penalties associated with the possible signal timings (or added delays) and associated probabilities of occurrence must be considered. For discretely distributed penalties, Eq. (1) in the Penalty Approach description employed in Section 3.2 becomes

$$\eta_i^h(t) = \sum_{p=1}^P \left\{ \sum_{k=1}^K [(\psi_{ij}^{e_i^l,p}(t) + \tau_{ij}^k(t + \psi_{ij}^{e_i^l,p}(t)) + \lambda_j^i(t + \psi_{ij}^{e_i^l,p}(t)) + \tau_{ij}^k(t + \psi_{ij}^{e_i^l,p}(t))) \cdot \rho_{ij}^k(t + \psi_{ij}^{e_i^l,p}(t))] \cdot \theta_{ij}^{e_i^l,p}(t) \right\}, \tag{2}$$

where $\psi_{ij}^{e_i^l,p}(t)$ is the p th possible penalty for departure time t and movements allowed in signal phase e_i^l , and $\theta_{ij}^{e_i^l,p}(t)$ is the probability of occurrence of $\psi_{ij}^{e_i^l,p}(t)$, $p \in \{1, 2, \dots, P\}$, for P possible penalties. When the penalties are continuously distributed, the summations in Eq. (2) must be replaced by integrals to take into account all possible values.

In either the discrete or continuous case, the penalties must be generated. This task can be foreboding, as illustrated in the following example. In the network shown in Fig. 2, a traffic signal at node i has three phases (a , b , and c) in which signals will be green for travelers from predecessor nodes h_1 , h_2 , and h_3 , respectively. The signal phase durations are also shown in Fig. 2 and are assumed to be independent. It is assumed that phase a starts at departure time interval 1. Consider movement $(h_1, i) \rightarrow (i, j)$, which is allowed only in phase a . The penalties for this movement for the first two departure times are zero because the signal is in phase 1. For departure time 3, the signal is red for this movement. It will not be green again until phases b and c have elapsed. There are three possible values for the duration of phases b and c together, $\{1 + 2, 1 + 3, 2 + 2, 2 + 3\} = \{3, 4, 5\}$, as shown in Fig. 2, and thus, there are three possible periods of time before movement $(h_1, i) \rightarrow (i, j)$ is allowed again. The penalties for other departure times can be similarly computed as shown in Fig. 2. When there are more signal phases or the PMF of the duration of a phase has more elements (e.g. where phase durations are continuously distributed, there would be an infinite number of possible values), the complexity of computing penalties increases exponentially because the joint distributions of all phase durations must be considered.

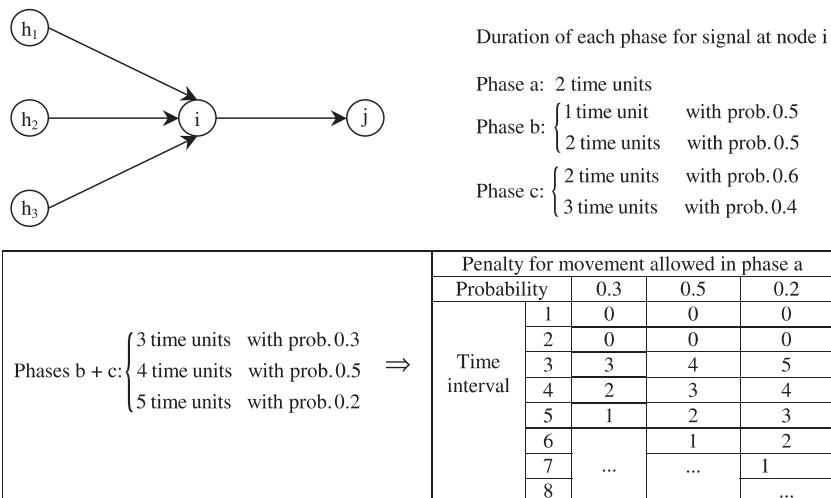


Fig. 2. Probabilistic penalties for movements allowed in phase a for the example problem.

Thus, for the adaptive routing problem in signalized networks with probabilistically known signal timings (or additional delays), the task of converting probabilistic signal timing information into penalties for use in a Penalty Approach similar to that described in the previous section is enormous. This motivates the development of another approach, the ARSC algorithm, described in the next section.

4.2. The ARSC algorithm

4.2.1. Algorithm overview

In this section, we consider the case where the added time for traveling due to signal operations might only be known probabilistically. Such instances arise, for example, where the signals are vehicle-actuated. In this case, the signal timing plans may be known, but the actual allocation of green time to each phase is known only probabilistically. Even for the case where the decision-maker has complete knowledge of the signal timings, the duration of the waiting time at the intersection may be uncertain due to effects of spillback from downstream intersections and queuing. In this section, an efficient specialized label-correcting algorithm, the ARSC algorithm, is proposed for solving the adaptive routing problem when such uncertainty in actual signal timings or delays at signalized intersections exists.

As in the Penalty Approach, in the ARSC algorithm, a label, $\lambda_i^h(t)$, is associated with each node i , each departure time t , and each predecessor node h (for a movement starting from node i , $h = i$) of node i . A pointer, $\pi_i^h(t)$, that indicates the next node to visit from node i given predecessor node h and departure time t is maintained for tracking the hyperpaths. Thus, for each node i , a set of vector labels A_i^h , one for each $h \in \{\Gamma^{-1}(i), i\}$, and corresponding vector pointers Π_i^h are maintained.

When signal timings or added delays are uncertain, two cases need to be considered for an intersection movement. In the first case, the outgoing arc corresponding to the movement in question is available to a traveler on the corresponding incoming arc, i.e. the movement is allowed and the traveler continues through the intersection. In this case, the label components at the successor node (labels are computed backward from the destination node) are used in the label construction at the current node. In the second case, when the outgoing arc is not available at the time of arrival due to a red signal for the desired movement, the traveler must wait and the label of the current node, but at the next departure time, is used in the label component $\lambda_i^h(t)$ computation. As will be seen in the description of the ARSC algorithm, this requires that the label values be computed backward in time. In order to compute these values, the probabilities of the occurrence of these two cases (outgoing arc is available or unavailable) are required. Note that these two probabilities may have different values at different arrival times.

The durations of arc available time and unavailable time (i.e. red and green signals) for a given movement are assumed to be exponentially distributed and a two-state continuous time Markov chain (CTMC), can be employed to compute the probability of the availability of an outgoing arc for a given movement through a signalized intersection for all departure times.

The choice of a CTMC for computing these probabilities was motivated by two factors. The first is that this choice makes the problem tractable. The second is that the model has several attractive features with respect to modeling signal operations in this context. CTMCs are processes that jump from state to state and can remain in each state for an exponentially distributed

length of time. The amount of time that the process remains in a given state is independent of how long the process has been observed to be in that state. Moreover, the use of Markov chains requires that the future be independent of the past, and thus, there is no relationship between how long the process was previously in a given state with how long it will be in that state in the future. This is particularly useful for modeling vehicle-actuated signals, where the time between arrivals of vehicles along the minor approaches, which trigger changes to the phase durations, is likely to be exponentially distributed. Further, we assume that the CTMC has stationary transition probabilities, which is consistent with the fact that, during a given time period (e.g. the PM peak period), cycle lengths are often constant to maintain progression between signals. The CTMC model is described next, followed by an illustrative example.

Define $X_{ij}^{e_i^l}(t)$ as the availability of arc (i, j) at departure time t for movements allowed in phase e_i^l , $\forall (i, j) \in A, i \in V_1, l \in [1, \zeta_i]$. $X_{ij}^{e_i^l}(t) = g$ when the arc is available; otherwise, $X_{ij}^{e_i^l}(t) = r$. The durations of available time and unavailable time of arc (i, j) for movements allowed in phase e_i^l are assumed to be independent and follow exponential distributions with parameters $\phi_{ij}^{e_i^l}$ and $\mu_{ij}^{e_i^l}$, respectively. The process $\{X_{ij}^{e_i^l}(t), t \geq 1\delta\}$ is a CTMC with *rate matrix*

$$Q_{ij}^{e_i^l} = \begin{bmatrix} -\phi_{ij}^{e_i^l} & \phi_{ij}^{e_i^l} \\ \mu_{ij}^{e_i^l} & -\mu_{ij}^{e_i^l} \end{bmatrix}.$$

Let the *initial state vector* $a_{ij}^{e_i^l} = [\text{prob}(X_{ij}^{e_i^l}(1\delta) = g), \text{prob}(X_{ij}^{e_i^l}(1\delta) = r)]$. The *transition probability matrix*

$$P_{ij}^{e_i^l}(t) = \begin{bmatrix} P_{ij}^{gg}(t) & P_{ij}^{gr}(t) \\ P_{ij}^{rg}(t) & P_{ij}^{rr}(t) \end{bmatrix}$$

can be calculated by solving the system of differential equations

$$\frac{d}{dt} P_{ij}^{e_i^l}(t) = P_{ij}^{e_i^l}(t) \cdot Q_{ij}^{e_i^l},$$

where $P_{ij}^{gg}(t) = \text{prob}\{X_{ij}^{e_i^l}(1\delta + t) = g | X_{ij}^{e_i^l}(1\delta) = g\}$. $P_{ij}^{gr}(t), P_{ij}^{rg}(t), P_{ij}^{rr}(t)$ are similarly defined. Given the initial state vector $a_{ij}^{e_i^l}$ and the transition probability matrix $P_{ij}^{e_i^l}(t)$, $Z_{ij}^{e_i^l}(t) = a_{ij}^{e_i^l} \cdot P_{ij}^{e_i^l}(t - 1\delta)$ is the probability that arc (i, j) is available at time t for movements allowed in signal phase e_i^l , based on probabilistic signal phase durations and the initial signal phase. It is assumed that when $h \neq i$, $R(h, i, j) = 1, Z_{ij}^{e_i^l}(t) = 1, \forall t \in S, i \in V_2, j \in V$ and that for all cases where $h = i, R(h, i, j) = 0, Z_{ij}^{e_i^l}(t) = 1, \forall t \in S, i, j \in V$. In other words, for movements at unsignalized nodes and movements originating from signalized nodes, corresponding arcs are always available.

A five-node example network is shown in Fig. 3. It is assumed that nodes 3 and 4 are signalized nodes and each signal is pretimed with two phases. At node 3, the durations of phases a_1 (when only movement $(1,3) \rightarrow (3,4)$ is permitted) and a_2 (when only movement $(2,3) \rightarrow (3,4)$ is permitted) have exponential distributions with parameters 0.4 and 0.5, respectively. At node 4, the durations of phases b_1 (when only movement $(2,4) \rightarrow (4,5)$ is permitted) and b_2 (when only movement $(3,4) \rightarrow (4,5)$ is permitted) have exponential distributions with parameters 0.5 and 0.4, respectively.

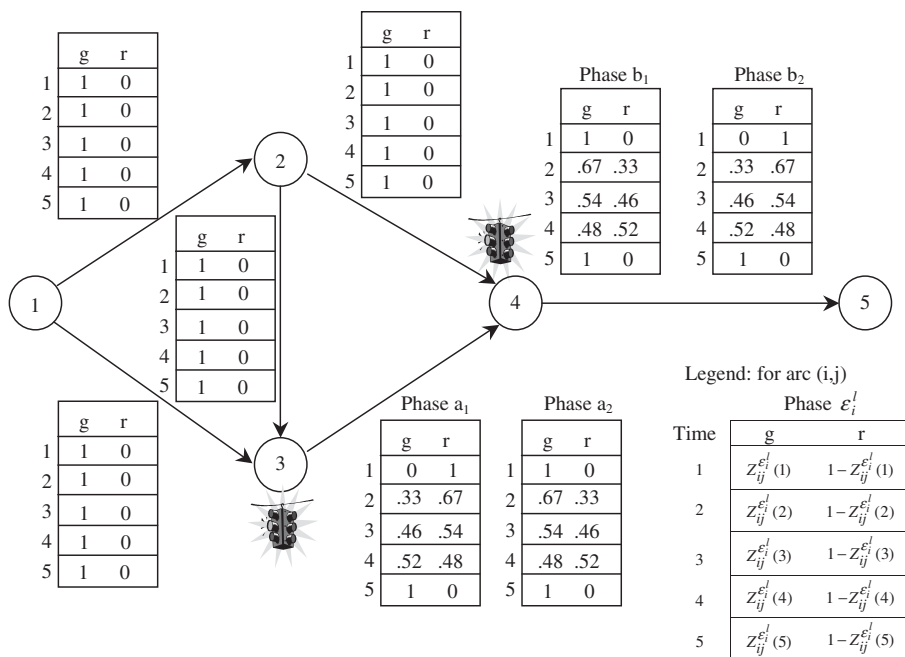


Fig. 3. Arc availability PMFs for the example problem.

Note that these parameter values were chosen arbitrarily and there is no particular relationship between the parameters at the two intersections. Suppose that, at departure time 1, it is known that the signal at node 3 is in phase a_2 (i.e. initial state $a_{34}^{\epsilon_3^1} = [0, 1]$ and $a_{34}^{\epsilon_3^2} = [1, 0]$), and the signal at node 4 is in phase b_1 (i.e. $a_{45}^{\epsilon_4^1} = [1, 0]$ and $a_{45}^{\epsilon_4^2} = [0, 1]$). Thus, for example, the rate matrix and transition probability matrix for arc (3,4) for any $t \in S$, for signal phase $\epsilon_3^1 = a_1$ are given as follows:

$$Q_{34}^{\epsilon_3^1} = \begin{bmatrix} -0.4 & 0.4 \\ 0.5 & -0.5 \end{bmatrix} \quad \text{and} \quad P_{34}^{\epsilon_3^1}(t) = \begin{bmatrix} \frac{4}{9}e^{-0.9t} + \frac{5}{9} & \frac{4}{9}(1 - e^{-0.9t}) \\ \frac{5}{9}(1 - e^{-0.9t}) & \frac{4}{9} + \frac{5}{9}e^{-0.9t} \end{bmatrix}^2.$$

The final arc availability probabilities, $Z_{ij}^{\epsilon_i^l}(t)$, are also shown in Fig. 3.

4.2.2. Algorithm description

The steps of the ARSC algorithm are presented next. The arc availability probabilities $Z_{ij}^{\epsilon_i^l}(t)$, $\forall (i,j) \in A, l \in [1, \zeta_i], t \in S$, are computed prior to running the algorithm.

Step 0: Initialization

Initialize the node labels.

$$\lambda_i^h(t) = \infty, \forall i \in V \setminus D, t \in S, \forall h \in \{\Gamma^{-1}(i), i\}.$$

$$\pi_i^h(t) = \infty, \forall i \in V \setminus D, t \in S, \forall h \in \{\Gamma^{-1}(i), i\}.$$

² The computation of the transition probabilities for CTMCs is discussed in Kulkarni (1995).

$$\lambda_D^h(t) = 0, \pi_D^h(t) = 0, \forall t \in \mathcal{S}, h \in \{\Gamma^{-1}(D), D\}.$$

Initialize the SE list and insert the destination node D .

Step 1: Choose the current node from the SE list

If the SE list is empty, go to step 3. Otherwise, select the first node j from the SE list.

Step 2: Update labels

For each $i \in \Gamma^{-1}(j)$

For each $h \in \{\Gamma^{-1}(i), i\}$

$$l = R(h, i, j).$$

For each $t \in \mathcal{S}$ from $T\delta \rightarrow 1\delta$

$$\eta_i^h(t) = Z_{ij}^{e_i^l}(t) \cdot \left\{ \sum_{k=1}^K [(\tau_{ij}^k(t) + \lambda_j^i(t + \tau_{ij}^k(t))) \cdot \rho_{ij}^k(t)] \right\} + (1 - Z_{ij}^{e_i^l}(t)) \cdot \{\lambda_i^h(t + \delta) + \delta\}.$$

If $\eta_i^h(t) < \lambda_i^h(t)$, then $\lambda_i^h(t) = \eta_i^h(t)$ and $\pi_i^h(t) = j$. $SE = SE \cup \{i\}$.

If all $i \in \Gamma^{-1}(j)$ have been considered, go to step 1.

Step 3: Stop

Proposition 3. Upon termination, given a connected network, the following relations hold for every label component resulting from the ARSC algorithm:

$$\lambda_i^h(t) \leq Z_{ij}^{e_i^l}(t) \cdot \left\{ \sum_{k=1}^K [(\tau_{ij}^k(t) + \lambda_j^i(t + \tau_{ij}^k(t))) \cdot \rho_{ij}^k(t)] \right\} + (1 - Z_{ij}^{e_i^l}(t)) \cdot \{\lambda_i^h(t + \delta) + \delta\} \forall j \in \Gamma^{+1}(i),$$

where $l = R(h, i, j)$.

Proof (by contradiction). Suppose at termination a label component, $\lambda_i^h(t)$, exists such that

$$\lambda_i^h(t) > Z_{ij}^{e_i^l}(t) \cdot \left\{ \sum_{k=1}^K [(\tau_{ij}^k(t) + \lambda_j^i(t + \tau_{ij}^k(t))) \cdot \rho_{ij}^k(t)] \right\} + (1 - Z_{ij}^{e_i^l}(t)) \cdot \{\lambda_i^h(t + \delta) + \delta\} \quad (3)$$

for some $j \in \Gamma^{+1}(i)$. If (3) is true, then node j must not have been scanned (i.e. chosen from the SE list in step 1 for use in improving labels at its predecessor nodes); otherwise, $\lambda_i^h(t)$ would have been updated in step 2 of the algorithm and this inequality could not hold. Given the network is connected, node j must enter the SE list at least once. Since node j has not yet been scanned, it must still be in the SE list, contradicting the assumption of termination. \square

Proposition 4. The ARSC algorithm terminates with the set of adaptive LET hyperpaths for the adaptive routing problem in signalized STV networks with probabilistically known signal timings.

Discussion: Miller-Hooks and Mahmassani (2000) prove by induction that the ELB algorithm solves the adaptive routing problem in unsignalized STV networks. The key of the induction is that once an arc is traversed, the stochastic arc traversal time is realized and the arrival time at the successor node is known. An induction similar to that used to prove the ELB algorithm can be established for proving the optimality of the ARSC algorithm for the adaptive routing problem. First, the label components at the destination node trivially correspond to the adaptive LET hyperpaths from the destination to itself.

In the ARSC algorithm, in addition to the stochastic arc traversal times, the arc availability probability, $Z_{ij}^{c,t}(t)$, is employed when the label is constructed. This probability and the underlying signal timings are independent of arc traversal times. Therefore, we can view the process of traveling through such a network as a sequence of decisions that are made once these two aspects of the routing problem are realized. That is, once a motorist arrives at a node, the arrival time is known (i.e. the arc traversal times on already traveled arcs are realized) and then the signal indication is revealed. The motorist chooses the next node given the revealed arrival time according to the hyperpath pointers. However, these pointers are computed for a given arrival time (i.e. the arrival time is known with certainty), but for probabilistic signal indication (i.e. the signal indication is not yet revealed). This corresponds to a decision that is taken when approaching an intersection but when the signal indication at the moment of arrival is not yet revealed. Note that this decision is based on only probabilistic predictions of the signal indications and arc traversal times along the remaining route to the destination. The process of traveling continues until the destination node is reached. Thus, a backward induction, similar to that given in (Miller-Hooks and Mahmassani, 2000), can be used to show that the resulting pointers provide the adaptive LET hyperpaths to the destination. This is because the only additional element in signalized network problem is the probabilistic delay due to the signal timings which, since decisions are made before these times are realized, can be thought of as simply additional probabilistic travel time on remaining arcs.

Proposition 5. *The ARSC algorithm with a basic FIFO SE list has worst-case computational complexity $O(n^4 T^2 K)$, where n is the number of nodes, T is the number of departure time intervals and K is the number of possible travel times for each departure time.*

Similar to the discussion of Proposition 2 (on the worst-case complexity of the Penalty Approach), the additional $O(n)$ complexity required in the ARSC algorithm as compared with the ELB algorithm ($O(n^3 T^2 K)$) is due to the extra loop for considering all predecessor nodes of the node chosen each time step 1 is repeated.

4.2.3. An illustrative example

We illustrate the ARSC algorithm on the 5-node example problem described in Section 4.2.1 as shown in Fig. 3. In this example, the peak period consists of 5 time intervals. The associated travel time PMFs are given in Table 2. The detailed computational steps for determining the hyperpaths for travel from all nodes to node 5 are shown in Table 3. The final label values are provided in Fig. 4.

Table 2
Travel time PMFs for the example problem for each time interval (TI)

Possible travel times	Arc (1,2)		Arc (1,3)		Arc (2,3)		Arc (2,4)		Arc (3,4)		Arc (4,5)	
	1	2	1	3	1	2	2	4	2	3	2	3
Prob. TI=1	0.4	0.6	0.6	0.4	0.8	0.2	0.9	0.1	0.5	0.5	0.5	0.5
TI=2	0.5	0.5	0.5	0.5	0.7	0.3	0.2	0.8	0.5	0.5	0.5	0.5
TI=3	0.3	0.7	0.9	0.1	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
TI=4	0.6	0.4	0.6	0.4	0.6	0.4	0.7	0.3	0.2	0.8	0.5	0.5
TI=5	0.4	0.6	0.8	0.2	0.9	0.1	0.5	0.5	0.4	0.6	0.5	0.5

Table 3
The steps of applying the ARSC algorithm on the example problem

SE list	Scanned node j	$i \in \Gamma^{-1}(j)$	$h \in \Gamma^{-1}(i)$ $R(h, i, j)$	Time t	Temporary label, $\eta_i^h(t)$	Current label $\lambda_i^h(t)$	Updated label $\lambda_i^h(t)$	Updated pointer $\pi_i^h(t)$
{}	Initialization: $\lambda_i^h(t) = \infty$ and $\pi_i^h(t) = \infty$, for $i \in \{1, 2, 3, 4\}$, $h \in \{1, 2, 3, 4, 5\}$, $t \in [1, 5]$; $\lambda_5^h(t) = 0$ and $\pi_5^h(t) = 0$, for $h \in \{1, 2, 3, 4, 5\}$, $t \in [1, 5]$. SE = {5}							
{5}	$j = 5$	$i = 4$	$h = 2$	$t = 5$	$(1)[(2+0)(0.5) + (3+0)(0.5)] + 0 = 2.5$	∞	2.5	5
{4}			$R(2, 4, 5) = 1$	$t = 4$	$(0.48)[(2+0)(0.5) + (3+0)(0.5)] + (0.52)(2.5+1) = 3.02$	∞	3.02	5
{4}				$t = 3$	$(0.54)[(2+0)(0.5) + (3+0)(0.5)] + (0.46)(3.02+1) = 3.20$	∞	3.20	5
{4}				$t = 2$	$(0.67)[(2+0)(0.5) + (3+0)(0.5)] + (0.33)(3.2+1) = 3.06$	∞	3.06	5
–				$t = 1$	$(1)[(2+0)(0.5) + (3+0)(0.5)] + 0 = 2.5$	∞	2.5	5
–			$h = 3$	$t = 5$	$(1)[(2+0)(0.5) + (3+0)(0.5)] + 0 = 2.5$	∞	2.5	5
–			$R(3, 4, 5) = 2$	$t = 4$	$(0.52)[(2+0)(0.5) + (3+0)(0.5)] + (0.48)(2.5+1) = 2.98$	∞	2.98	5
–				$t = 3$	$(0.46)[(2+0)(0.5) + (3+0)(0.5)] + (0.54)(2.98+1) = 3.30$	∞	3.30	5
–				$t = 2$	$(0.33)[(2+0)(0.5) + (3+0)(0.5)] + (0.67)(3.3+1) = 3.71$	∞	3.71	5
–				$t = 1$	$0 + (1)(3.71+1) = 4.71$	∞	4.71	5
–			$h = 4$	$t = 5$	$(1)[(2+0)(0.5) + (3+0)(0.5)] + 0 = 2.5$	∞	2.5	5
–			$R(4, 4, 5) = 0$	$t = 4$	$(1)[(2+0)(0.5) + (3+0)(0.5)] + 0 = 2.5$	∞	2.5	5
–				$t = 3$	$(1)[(2+0)(0.5) + (3+0)(0.5)] + 0 = 2.5$	∞	2.5	5
–				$t = 2$	$(1)[(2+0)(0.5) + (3+0)(0.5)] + 0 = 2.5$	∞	2.5	5
–				$t = 1$	$(1)[(2+0)(0.5) + (3+0)(0.5)] + 0 = 2.5$	∞	2.5	5
–	$j = 4$	$i = 2$	$h = 1$ and 2	$t = 5$	$(1)[(2+2.5)(0.5) + (4+2.5)(0.5)] + 0 = 5.5$	∞	5.5	4
{2}			$R(1, 2, 4) = 1$	$t = 4$	$(1)[(2+2.5)(0.7) + (4+2.5)(0.3)] + 0 = 5.1$	∞	5.1	4
{2}			$R(2, 2, 4) = 0$	$t = 3$	$(1)[(2+2.5)(0.5) + (4+2.5)(0.5)] + 0 = 5.5$	∞	5.5	4
{2}				$t = 2$	$(1)[(2+3.02)(0.2) + (4+2.5)(0.8)] + 0 = 6.20$	∞	6.20	4
–				$t = 1$	$(1)[(2+3.2)(0.9) + (4+2.5)(0.1)] + 0 = 5.33$	∞	5.33	4
–		$i = 3$	$h = 1$	$t = 5$	$(1)[(2+2.5)(0.4) + (3+2.5)(0.6)] + 0 = 5.1$	∞	5.1	4
{2, 3}			$R(1, 3, 4) = 1$	$t = 4$	$(0.52)[(2+2.5)(0.2) + (3+2.5)(0.8)] + (0.48)(5.1+1) = 5.68$	∞	5.68	4
{2, 3}				$t = 3$	$(0.46)[(2+2.5)(0.5) + (3+2.5)(0.5)] + (0.54)(5.68+1) = 5.91$	∞	5.91	4
{2, 3}				$t = 2$	$(0.33)[(2+2.98)(0.5) + (3+2.5)(0.5)] + (0.67)(5.91+1) = 6.36$	∞	6.36	4
–				$t = 1$	$0 + (1)(6.36+1) = 7.36$	∞	7.36	4

–			$h = 2$	$t = 5$	$(1)[(2 + 2.5)(0.4) + (3 + 2.5)(0.6)] + 0 = 5.1$	∞	5.1	4
–			$R(2, 3, 4) = 2$	$t = 4$	$(0.48)[(2 + 2.5)(0.2) + (3 + 2.5)(0.8)] + (0.52)(5.1 + 1) = 5.72$	∞	5.72	4
–				$t = 3$	$(0.54)[(2 + 2.5)(0.5) + (3 + 2.5)(0.5)] + (0.46)(5.72 + 1) = 5.79$	∞	5.79	4
–				$t = 2$	$(0.67)[(2 + 2.98)(0.5) + (3 + 2.5)(0.5)] + (0.33)(5.79 + 1) = 5.75$	∞	5.75	4
–				$t = 1$	$(1)[(2 + 3.3)(0.5) + (3 + 2.98)(0.5)] + 0 = 5.64$	∞	5.64	4
–			$h = 3$	$t = 5$	$(1)[(2 + 2.5)(0.4) + (3 + 2.5)(0.6)] + 0 = 5.1$	∞	5.1	4
–			$R(3, 3, 4) = 0$	$t = 4$	$(1)[(2 + 2.5)(0.2) + (3 + 2.5)(0.8)] + 0 = 5.3$	∞	5.3	4
–				$t = 3$	$(1)[(2 + 2.5)(0.5) + (3 + 2.5)(0.5)] + 0 = 5.0$	∞	5.0	4
–				$t = 2$	$(1)[(2 + 2.98)(0.5) + (3 + 2.5)(0.5)] + 0 = 5.24$	∞	5.24	4
–				$t = 1$	$(1)[(2 + 3.3)(0.5) + (3 + 2.98)(0.5)] + 0 = 5.64$	∞	5.64	4
–	$j = 2$	$i = 1$	$h = 1$	$t = 5$	$(1)[(1 + 5.5)(0.4) + (2 + 5.5)(0.6)] + 0 = 7.1$	∞	7.1	2
{3, 1}			$R(1, 1, 2) = 0$	$t = 4$	$(1)[(1 + 5.5)(0.6) + (2 + 5.5)(0.4)] + 0 = 6.9$	∞	6.9	2
{3, 1}				$t = 3$	$(1)[(1 + 5.1)(0.3) + (2 + 5.5)(0.7)] + 0 = 7.08$	∞	7.08	2
{3, 1}				$t = 2$	$(1)[(1 + 5.5)(0.5) + (2 + 5.1)(0.5)] + 0 = 6.8$	∞	6.8	2
–				$t = 1$	$(1)[(1 + 6.2)(0.4) + (2 + 5.5)(0.6)] + 0 = 7.38$	∞	7.38	2
–	$j = 3$	$i = 1$	$h = 1$	$t = 5$	$(1)[(1 + 5.1)(0.8) + (3 + 5.1)(0.2)] + 0 = 6.5$	7.1	6.5	3
{1}			$R(1, 1, 3) = 0$	$t = 4$	$(1)[(1 + 5.1)(0.6) + (3 + 5.1)(0.4)] + 0 = 6.9$	6.9	6.9	2
{1}				$t = 3$	$(1)[(1 + 5.68)(0.9) + (3 + 5.1)(0.1)] + 0 = 6.82$	7.08	6.82	3
{1}				$t = 2$	$(1)[(1 + 5.91)(0.5) + (3 + 5.1)(0.5)] + 0 = 7.51$	6.8	6.8	2
–				$t = 1$	$(1)[(1 + 6.36)(0.6) + (3 + 5.68)(0.4)] + 0 = 7.89$	7.38	7.38	2
–		$i = 2$	$h = 1$ and 2	$t = 5$	$(1)[(1 + 5.1)(0.9) + (2 + 5.1)(0.1)] + 0 = 6.2$	5.5	5.5	4
–			$R(1, 2, 3) = 1$	$t = 4$	$(1)[(1 + 5.1)(0.6) + (2 + 5.1)(0.4)] + 0 = 6.5$	5.1	5.1	4
–			$R(2, 2, 3) = 0$	$t = 3$	$(1)[(1 + 5.72)(0.5) + (2 + 5.1)(0.5)] + 0 = 6.91$	5.5	5.5	4
–				$t = 2$	$(1)[(1 + 5.79)(0.7) + (2 + 5.72)(0.3)] + 0 = 7.07$	6.20	6.20	4
–				$t = 1$	$(1)[(1 + 5.75)(0.8) + (2 + 5.79)(0.2)] + 0 = 6.96$	5.33	5.33	4
{1}	$j = 1$	\emptyset	\emptyset	Stop				

The LETs for travel from all nodes to the destination node 5 at departure time t are given by the $\lambda_i^h(t)$ with $h = i$. The final label values and corresponding pointers are displayed in bold in this table.

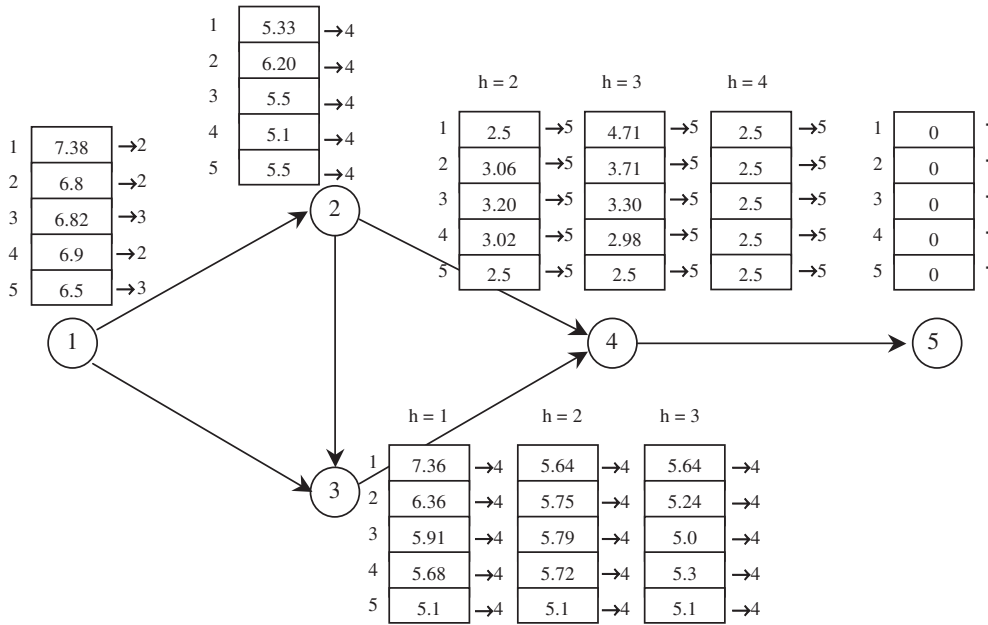


Fig. 4. Final labels and pointers resulting from the ARSC algorithm.

From Fig. 4, one can see that, for example, for node 3 at departure time 3, the LET for the remaining path is 5.91 time units if a traveler comes from node 1, 5.79 units if a traveler comes from node 2, or 5.0 units if a traveler originates at node 3.

The resulting hyperpaths for departure from node 1 at time 1 are shown in Fig. 5. In this figure, circular nodes represent the states when the traveler arrives at the node just prior to revealing the signal indication at the intersection. Rectangular nodes represent the states where the signal indications are revealed. For example, suppose a motorist is traveling from node 1 to node 5 departing node 1 at departure time 1. The arrival time at node 2 could be at either time 2 or 3. If the motorist arrives at node 2 at time 2, the LET of the remaining path is 6.2 units. If arrival is at time 3, the LET of the remaining path is 5.5 units. Because there is no traffic signal control at node 2, i.e. the signal is always green, only one branch exists from node 2 for either possible departure time (recall that no voluntary waiting is permitted). If the traveler leaves for node 4 at time 2, the possible arrival times at node 4 are 4 and 6. Given arrival at node 4 at time 4, the LET of the remaining path is 3.02 units $((0.48)(2.5) + (0.52)(3.5) = 3.02$, where 0.48 is the probability that the signal at node 4 for movement $(2,4) \rightarrow (4,5)$ is green at time 4 with LET of 2.5 units for the remaining path under this condition, and 0.52 is the probability that the signal is red at time 4 with corresponding LET of 3.5 units for the remaining path). For arrival at node 4 at time 6, the LET of the remaining path is 2.5 units $((1)(2.5) = 2.5$, where 1 is the probability that the signal at node 4 for movement $(2,4) \rightarrow (4,5)$ is green, based on the assumption that all movements are permitted at the last departure time or thereafter). The traveler continues until node 5 is reached.

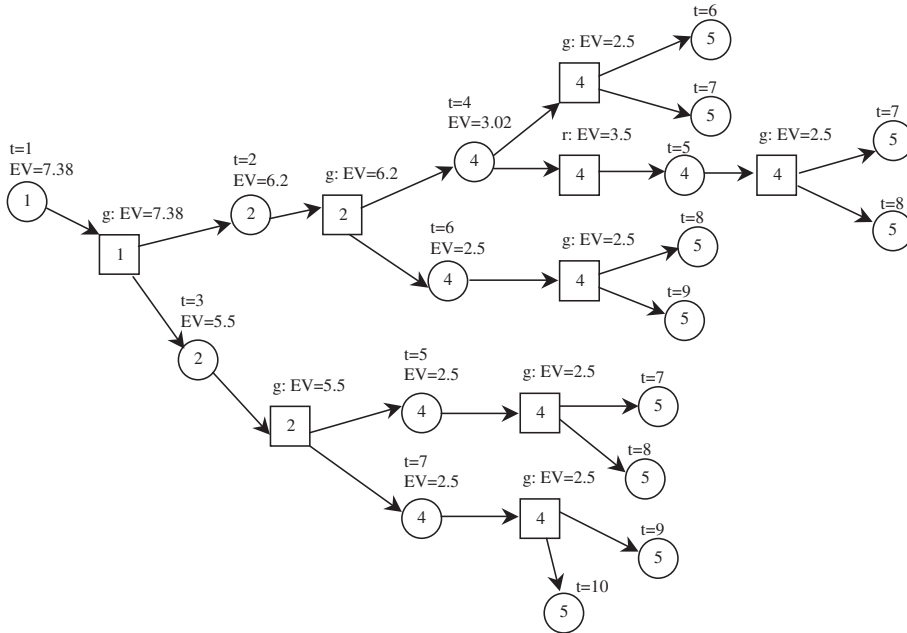


Fig. 5. Hyperpaths generated by the ARSC algorithm for departure time 1 from node 1.

4.3. A special case of the ARSC algorithm

In the ARSC algorithm, a label component $\lambda_i^h(t)$ is constructed from two sources: a successor node j of node i and the label $\lambda_i^h(t + \delta)$ at the next departure time at node i . The availability probability, $Z_{ij}^{e_l}(t)$, of the outgoing arc (i, j) from the signaled intersection at departure time t for the movement $(h, i) \rightarrow (i, j)$ provides the “weight” for each of these two components of the computation. When the signal timings are known a priori, the availability of arc (i, j) is known deterministically. Therefore, when the arc is available, $Z_{ij}^{e_l}(t) = 1$ and the corresponding temporary label at node i is constructed only from the successor node j (i.e. $\eta_i^h(t) = \sum_{k=1}^K [(\tau_{ij}^k(t) + \lambda_j^i(t + \tau_{ij}^k(t))) \cdot \rho_{ij}^k(t)]$). When the arc is unavailable for the movement, $Z_{ij}^{e_l}(t) = 0$ and the temporary label for departure time t is computed as $\eta_i^h(t) = \lambda_i^h(t + \delta) + \delta$, employing only the label component at node i at the next departure time. For this case, it is found that the ARSC algorithm reduces to the Penalty Approach. The equivalence is briefly explained as follows.

In the Penalty Approach, the temporary label is computed using Eq. (1), which can be rewritten as

$$\eta_i^h(t) = \sum_{k=1}^K [(\tau_{ij}^k(t + \psi_{ij}^{e_l}(t)) + \lambda_j^i(t + \psi_{ij}^{e_l}(t) + \tau_{ij}^k(t + \psi_{ij}^{e_l}(t)))) \cdot \rho_{ij}^k(t + \psi_{ij}^{e_l}(t))] + \psi_{ij}^{e_l}(t), \quad (4)$$

because $\psi_{ij}^{e_l}(t)$ is not dependent on k . When arc (i, j) is available for movement $(h, i) \rightarrow (i, j)$ at time t , the penalty $\psi_{ij}^{e_l}(t)$ is equal to 0 and Eq. (4) can be simplified to

$$\eta_i^h(t) = \sum_{k=1}^K [(\tau_{ij}^k(t) + \lambda_j^i(t + \tau_{ij}^k(t))) \cdot \rho_{ij}^k(t)].$$

When arc (i, j) is unavailable at time t , Eq. (4) says that $t + \psi_{ij}^{e_i^j}(t)$ is the next departure time in which arc (i, j) is available and $\eta_i^h(t) = \lambda_i^h(t + \psi_{ij}^{e_i^j}(t)) + \psi_{ij}^{e_i^j}(t)$. In other words, for each departure time interval between t and $t + \psi_{ij}^{e_i^j}(t)$, a penalty of one time interval is incurred. This is equivalent to repeatedly setting $\eta_i^h(t) = \lambda_i^h(t + \delta) + \delta$ for all departure times in $[t, t + \psi_{ij}^{e_i^j}(t) - \delta]$. Thus, the ARSC algorithm reduces to the Penalty Approach when the signal timings are known a priori. Given this equivalence, proof of optimality of the ARSC algorithm, as presented in Propositions 3 and 4, suffices in proving optimality of the Penalty Approach.

5. A real-world illustration and assessment

In this section, the ARSC algorithm is illustrated on a real-world-based signalized street network to measure the value of explicitly considering the added delay due to signal control when determining the LET hyperpaths for the adaptive routing problem addressed herein. We run experiments to compare the resulting hyperpaths of the ARSC algorithm, which explicitly considers the additional time spent traveling due to signal operations, with those generated by the ELB (or SDOT) algorithm that does not consider the effects of the traffic signal control on the optimal solution.

5.1. Experimental design

A network representation of a portion of the State College (SC), Pennsylvania street network is used in illustrating the ARSC algorithm. This network representation was developed by Opasanon and Miller-Hooks (2001) and was used to illustrate an algorithm posed for a related multimodal transportation problem. This network consists of 264 nodes and 802 arcs. The peak period is discretized into 400 time intervals, each of 9 s in duration. This interval size is chosen based on a minimum arc travel time of 9 s that is generally true for the SC network. (Recall that the interval size must be at least as small as the smallest arc travel time in the network.)

Eighteen of the 264 nodes contain traffic signal control, all of which are located along four major corridors (Atherton Street, Park Avenue, College Avenue, and Beaver Avenue) in the study region, as shown in Fig. 6. The signalized intersections are numbered in the figure. The signal settings for these nodes, including offsets and phase timings, were obtained from the Borough of State College. These settings permit signal coordination along the major traffic corridors. While all of the signals are actuated, we assumed that they are pre-timed with settings based closely on the given information. Additionally, it is assumed that complete knowledge of these settings is known. There are an additional eight signalized intersections in the study region for which we could not obtain signal information. In most of these cases, the signals at these intersections function independently of the other signals in the study area and their impact on routing decisions in this area is likely to be small. Since the addition of more signals will only help us to make our

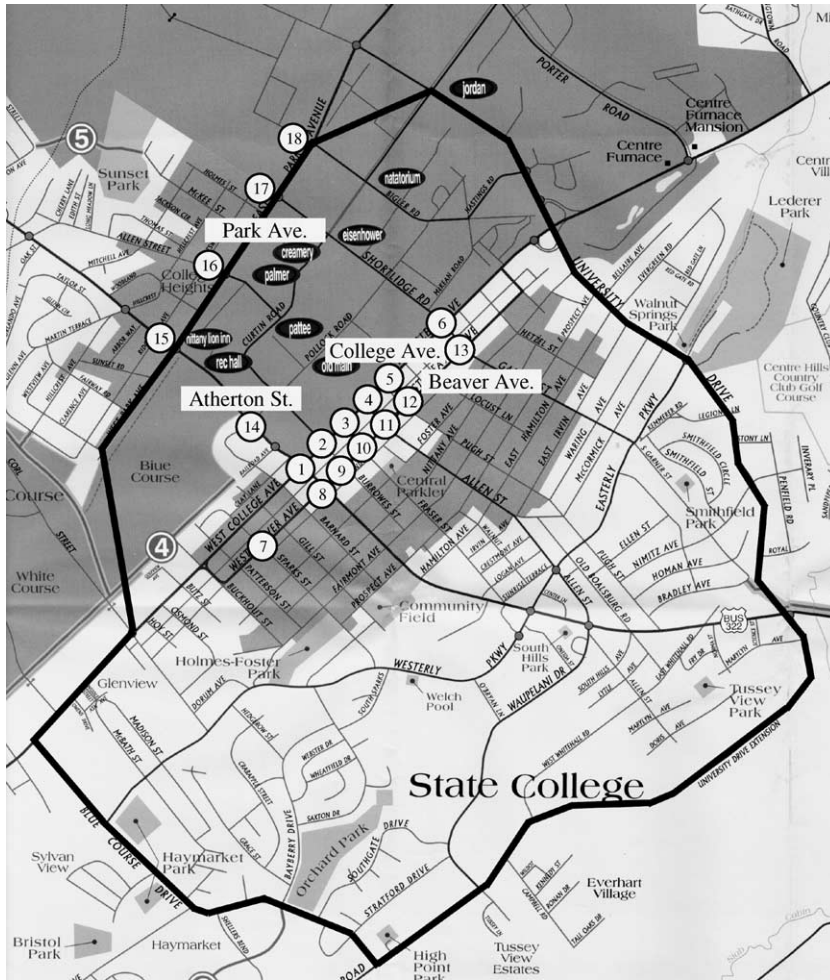


Fig. 6. The State College road network with traffic signals at numbered intersections.

case, because the more signalized intersections the more likely it is that a procedure that ignores signal operations will result in suboptimal solutions, these intersections were treated as if they were unsignalized.

For each arc and departure time, the travel time PMFs were randomly generated by a procedure developed and described in (Miller-Hooks and Mahmassani, 1998). This procedure generates the PMFs from an underlying normal distribution truncated at zero (no negative times are permitted). The method requires the mean and standard deviation of the travel time to be specified. The standard deviation is taken as 0.071 of the mean. As described in (Opasanon and Miller-Hooks, 2001), a Geographical Information System (GIS) database through TransCAD was used to collect arc distances and posted speed limits for each arc in the study region. Speeds on all arcs are assumed to change with time. At the beginning of the peak period, average speeds on all arcs are set to their speed limits. They are assumed to decrease linearly to minimum values (50% of the speed limits for the 133 arcs along the major corridors and 70% for the remaining

arcs) in the first half of the peak period and increase linearly to their speed limits by the end of the peak period. The average speed thereafter is assumed to be constant at the speed limit. The mean travel time was computed from the distance divided by the corresponding mean speed. Three possible arc travel times with corresponding probabilities of occurrence were generated for each arc at each departure time.

Three sets of experiments were conducted. The first set of experiments was designed to assess the impact on the LET of the hyperpaths of explicitly considering delay due to signal control. Specifically, signal delay is ignored when the expected times are computed by the ELB algorithm, but is included in the computation completed by the ARSC algorithm. Thus, these two procedures may result in different solution strategies. For a given origin node and departure time, the ratio of the expected time resulting from the ELB algorithm to that resulting from the ARSC algorithm will provide a measure of how much the ELB algorithm underestimates the actual optimal solution when additional time for traveling due to signal operations should have been considered. The average value of such ratios over all origin nodes and departure times for a select destination was computed and the average over 20 randomly selected destination nodes was then computed for the solutions resulting from the algorithms. By definition, the expected time of the ELB hyperpaths will be no greater than that of the ARSC hyperpaths since the ELB algorithm ignores the additional delay due to the signal operations.

The second set of experiments was designed to quantify changes in the topology of the optimal hyperpaths when signal operations are considered. To quantify the differences in the solution hyperpaths of the ELB and ARSC algorithms (referred to as the ELB and ARSC hyperpaths, respectively, hereafter), their resulting hyperpaths should be compared side by side for each O–D pair at each departure time interval. There are two difficulties in doing this. First, the size of the hyperpath tree increases exponentially with the number of nodes involved in the solution paths. For example, when there are 20 nodes involved in the solution paths, the number of nodes in the hyperpath tree could be greater than $3^{20} \approx 3 \times 10^9$ assuming the number of possible travel times is three for all arcs and departure times. Therefore, it is inefficient to compare the entire hyperpath tree for most O–D pairs in the network. Second, given the tremendous size of the hyperpath tree, it is unlikely that the ELB and ARSC hyperpaths will be identical. In many cases, minor differences exist with significantly sized subtrees that remain nearly identical, if not identical. As a result, the hyperpaths resulting from these two algorithms were compared through the use of a simplified technique described next.

For each O–D pair at each departure time interval, two sets of nodes were created, one resulting from ELB algorithm solutions (the ELB set) and the other from ARSC algorithm solutions (the ARSC set). For each O–D pair at each departure time, the sets were generated by conducting a complete search through the hyperpath solution trees. Once the two sets have been created, their elements are compared with respect to two measures. The first measure asks the question: Of all the nodes in the ELB set, how many are in the true solution found by the ARSC algorithm (i.e. in the ARSC set) as a percentage of the total number in the ELB set? The second measure asks: Of all the nodes in the ARSC set, how many never arose in the ELBs solution, i.e. are not in the ELB set, as a percentage of the total number in the ARSC set? For example, suppose the ELB and ARSC sets are given as follows: $\{1, 3, 5, 6, 8\}$ and $\{2, 3, 4, 5, 7, 8\}$, respectively. Then the answer to the first question is that 3 of 5 (i.e. $\{3, 5, 8\}$ of $\{1, 3, 5, 6, 8\}$) or 60% of the nodes are actually in the true optimal hyperpaths. The answer to the second question is that 3 of 6 (i.e. $\{2, 4, 7\}$ of

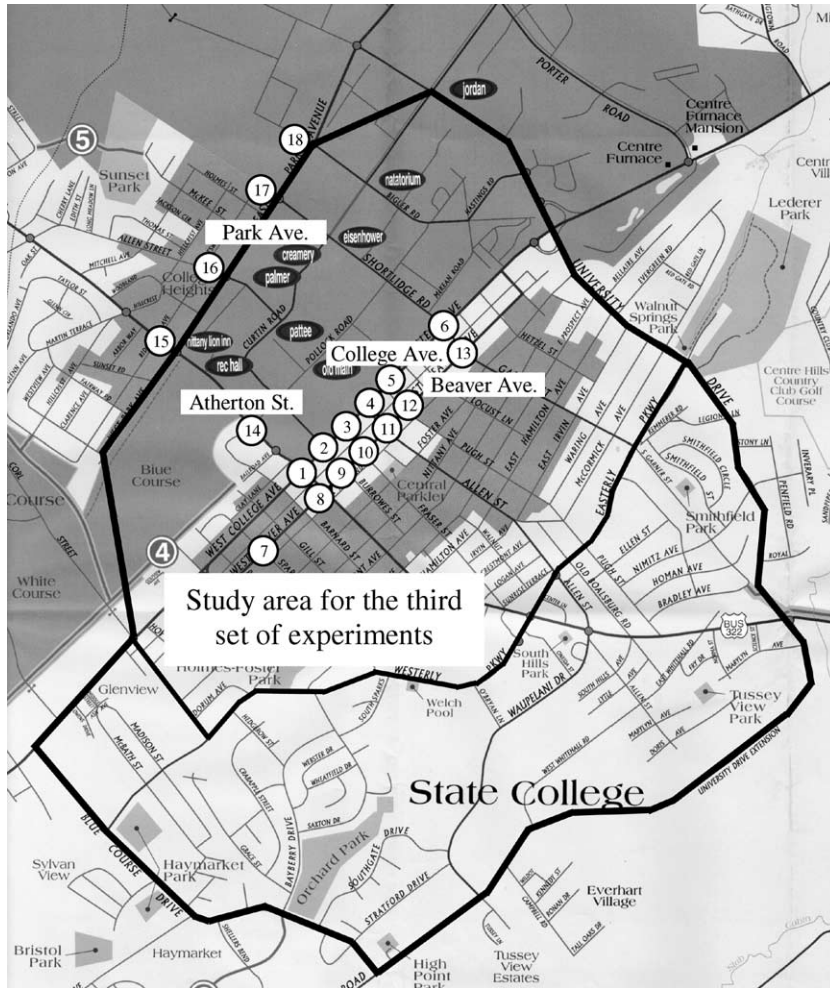


Fig. 7. The subnetwork chosen for the third set of experiments (top portion).

{2, 3, 4, 5, 7, 8}) or 50% of the nodes that are in the optimal hyperpaths were not even in the hyperpaths found by the ELB algorithm. The comparisons were conducted for all 264 origins to each of 20 randomly selected destination nodes. The average value (and standard deviation) for each measure over all origins to the same destination and for all departure times in the peak period was taken.

A third set of experiments is required to measure the value of explicitly considering the delay due to signal control incurred along the LET hyperpaths. Specifically, the ELB algorithm was run (ignoring the signals) and the resulting LET hyperpaths were computed for each origin to a specified destination for the first departure time interval in the peak period. The expected time of these hyperpaths was then properly computed to include the additional delay due to the signals at signalized intersections included in the solution. That is, the solution itself was not changed, only the actual expected value of the solution was recomputed. The resulting value was compared with

Table 4
Comparing the topology of ELB and ARSC solution hyperpaths

Destination	First measure ^a	Standard deviation	Second measure ^b	Standard deviation
1	0.96	0.11	0.06	0.17
2	0.96	0.12	0.07	0.18
3	0.98	0.09	0.57	0.25
4	0.92	0.15	0.17	0.18
5	0.94	0.15	0.03	0.10
6	0.96	0.14	0.43	0.30
7	0.92	0.16	0.09	0.19
8	0.92	0.13	0.09	0.15
9	0.98	0.12	0.62	0.22
10	0.92	0.17	0.04	0.09
11	0.95	0.12	0.06	0.16
12	0.91	0.14	0.04	0.12
13	0.93	0.15	0.04	0.10
14	0.96	0.12	0.21	0.25
15	0.94	0.12	0.04	0.14
16	0.97	0.12	0.61	0.28
17	0.98	0.08	0.20	0.21
18	0.95	0.15	0.59	0.26
19	0.90	0.18	0.02	0.06
20	0.94	0.14	0.10	0.17
Average	0.95	–	0.20	–

^a First measure: Percent average (over all origins) of number of nodes found in both solution sets of total number in ELB set.

^b Second measure: Percent average (over all origins) of number of nodes found in ARSC set not contained in ELB set of total number in ARSC set.

the expected time of the ARSC hyperpaths to obtain the travel time savings (reduction in the expected travel time) gained through explicitly considering the signal timings in choosing the hyperpaths. In this set of experiments, such a comparison was first made for two particular O–D pairs at the first departure time. Then, for a select destination, the expected travel times of the ELB and ARSC hyperpaths were compared for 20 randomly selected origin nodes at the first departure time interval. Due to the tremendous size of the hyperpath tree, these 20 origins were selected from a subnetwork of the study region, as shown in Fig. 7. This chosen area contains all signalized intersections in the region.

The algorithms were implemented in C++ and were run on a DEC Alpha XP1000 professional workstation with 1 gigabyte ram and 2 gigabyte swap, running Digital 4.0E operating system, using Digital's C++ compiler.

5.2. Experimental results and analysis

For 20 randomly selected destinations, from all origins, the first set of experiments showed that the average expected time of the ARSC hyperpaths is 10.2% higher (ranging between 7.6% and 14.3%) than that of the ELB hyperpaths. Thus, the ELB algorithm significantly underestimates the expected time required for the best solution.

Results of the second set of experiments, designed to quantify changes in the topology of the hyperpaths due to consideration of signal control, are summarized in Table 4. The results indicate that, on average over all O–D pairs, 95% (with a range of 90–98% on average for each of the destinations) of the nodes contained in the ELB algorithm are also contained in the true optimal solution. The results further suggest that the 20% (with a range of 2–62% on average for each of the destinations) of the nodes in the true optimal solutions could not be found in the ELB solutions. These findings are not surprising as the solutions will be identical when the signals are green for a given path strategy, but will differ when signals are in the red phase that affect the otherwise optimal path strategies. That is, the ARSC solution strategies will avoid the signalized intersections when the signals are likely to lead to additional travel delay given the likelihood of possible arrival times at the signalized intersections during a red phase. Thus, the results show that there is often considerable difference in the topology of the ELB hyperpaths, which are generated without consideration of the additional delays due to signal operations, as compared with the ARSC hyperpaths, which are generated with explicit consideration for these additional delays.

Results of the third set of experiments show that the hyperpaths chosen by the ARSC algorithm have significantly lower expected time than those chosen by the ELB algorithm when the expected value is properly computed to include the additional delay due to signal control at the signalized intersections included in the solution. First, two particular O–D pairs were considered. For the first O–D pair, A–B, for the majority of the departure times, the solution of the ELB algorithm suggests that the motorist should choose the path highlighted in Fig. 8(a). The corresponding

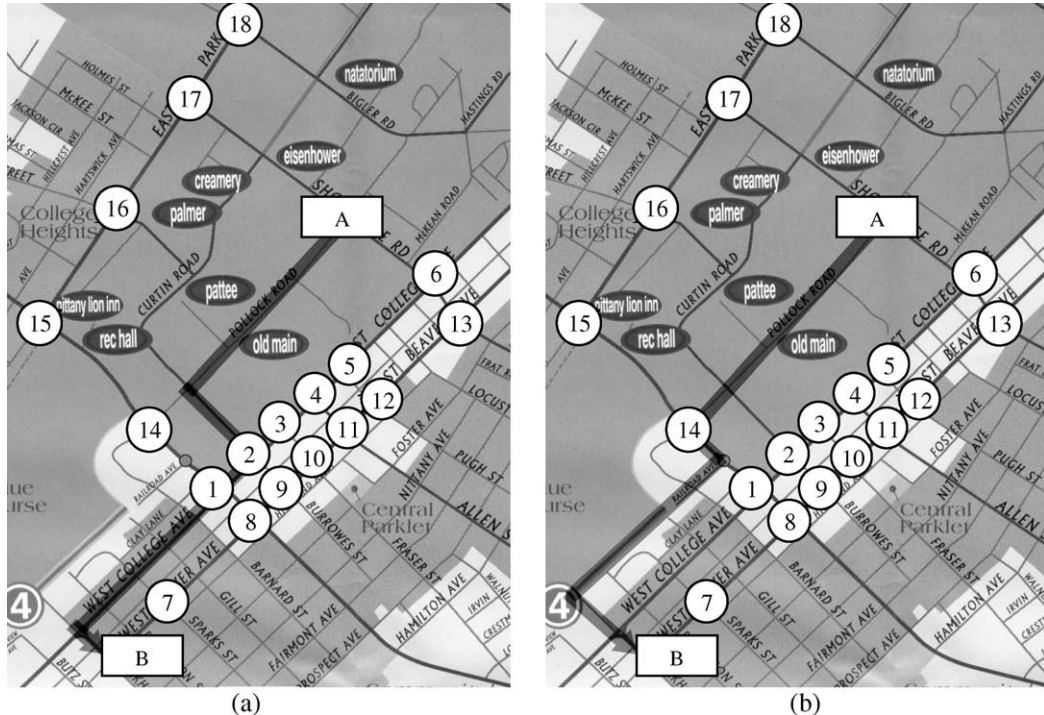


Fig. 8. The primary paths chosen from origin node A to destination node B: (a) ELB, (b) ARSC.

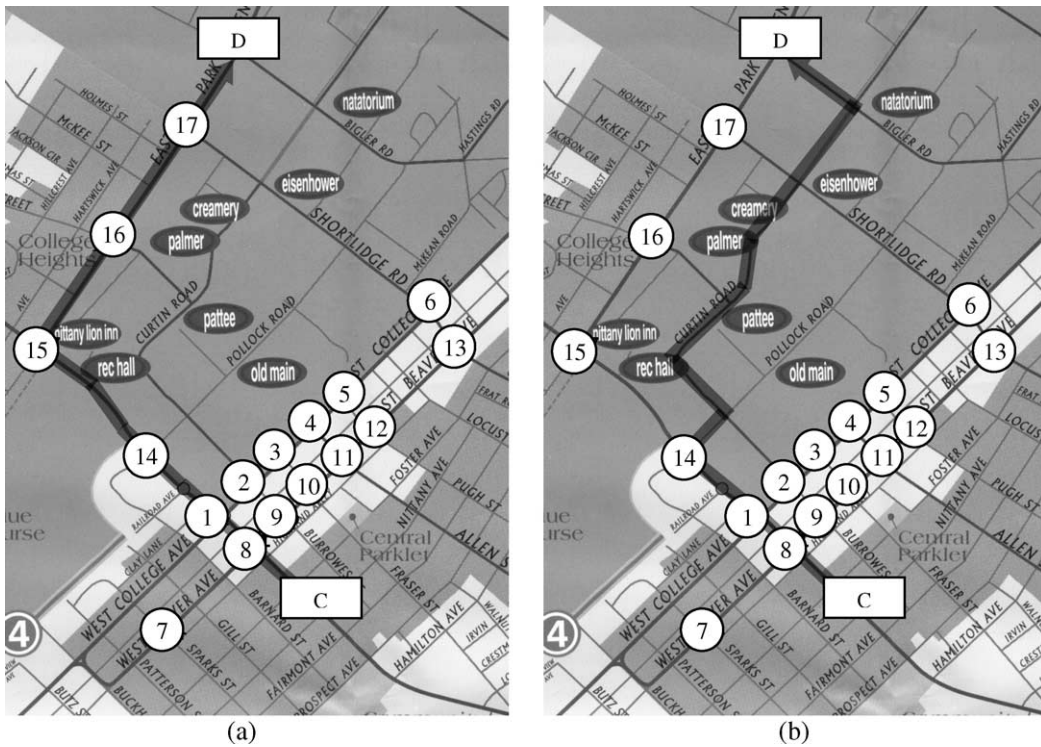


Fig. 9. The primary paths chosen from origin node C to destination node D: (a) ELB, (b) ARSC.

expected travel time of this solution is 273.3 s when the expected time is computed to include the delay due to signal control. The solution hyperpaths of the ARSC algorithm, however, indicate that, for the majority of the departure times, a motorist should choose the path highlighted in Fig. 8(b), which circumvents the traffic signals at nodes 1 and 2. In this case, the expected travel time is 199.1s, which is 27.1% less than that of the ELB hyperpaths. For the second O–D pair, C–D, the ELB hyperpaths indicate that, for the majority of the departure times, a motorist should select the path highlighted in Fig. 9(a), where six intermediate nodes (1, 8, 14, 15, 16, and 17) are signalized nodes. For most departure times en route, the ARSC solution hyperpaths, however, suggest the path that has fewer signalized nodes, as highlighted in Fig. 9(b). In this case, a savings of 28.4% is incurred (the expected travel time decreases from 270.3 s for the ELB hyperpaths to 193.6 s for the ARSC hyperpaths).

One can surmise that the value of considering the delay due to signal operations depends on the relative location of the origin and destination nodes to the traffic signals. To measure the average benefit of considering this delay, we examined the hyperpaths from 20 origin nodes to destination node D (shown in Fig. 9). The results are shown in Table 5, where “ELB time without signals” represents the expected travel time of the ELB hyperpaths when signal control is not considered, “ELB time with signals” represents the expected travel time of the ELB hyperpaths when the delay due to signal control is included for the same solution hyperpaths, and “ARSC time” represents the expected travel time of the ARSC hyperpaths. As is shown in the table, the hy-

Table 5

Expected travel times corresponding to the ELB and ARSC hyperpaths to destination node *D*

Origin node	ELB time without signals	ELB time with signals	ARSC time	Savings of the ARSC time over the ELB time with signals (%)
1	142.4	214.7	190.1	11.4
2	194.2	317.8	204.4	35.7
3	173.0	352.0	207.3	41.1
4	193.9	381.2	220.2	42.2
5	147.4	191.4	186.8	2.4
6	159.3	270.3	193.6	28.4
7	188.2	323.4	222.4	31.2
8	190.6	282.0	205.0	27.3
9	189.2	334.1	262.9	21.3
10	227.1	326.8	267.1	18.3
11	191.6	285.4	226.7	20.6
12	185.7	296.8	221.2	25.5
13	232.1	340.2	247.5	27.2
14	117.6	150.0	140.1	6.6
15	92.5	121.6	97.5	19.8
16	144.9	232.8	147.5	36.6
17	169.9	273.4	204.4	25.2
18	164.1	318.7	197.5	38.0
19	204.1	367.0	277.3	24.4
20	190.4	332.0	209.8	36.8
Average	–	–	–	26.0

perpaths produced by the ARSC algorithm are consistently superior to those produced by the ELB algorithm in terms of expected travel times; thus, indicating that there is a measurable improvement in solutions that were computed by explicitly considering delay due to signal control. In fact, a 26% savings in average travel time was achieved by employing the ARSC algorithm (i.e. explicitly considering signal delays) in place of the ELB algorithm (i.e. ignoring the signal delays).

6. Conclusions and extensions

In this paper, the methodologies proposed by Ziliaskopoulos and Mahmassani (1996) for addressing the classical shortest path problem considering intersection delay and prohibitions and the ELB algorithm of Miller-Hooks and Mahmassani (2000) for determining LET adaptive path strategies in STV unsignalized networks were combined and extended for use in determining LET adaptive path strategies in signalized STV networks, where actual signal timings are known. The resulting procedure is called the Penalty Approach.

For the case of probabilistically known signal timings or uncertain delays due to lost times, it was shown that the Penalty Approach would not be efficient. An efficient label-correcting

algorithm, the ARSC algorithm, was proposed for solving the adaptive routing problem in this context. The ARSC algorithm extends the ELB algorithm to explicitly handle additional delays due to signal operations, where actual signal timings or delays due to signal operations are only known probabilistically.

The ARSC algorithm relies on an assumption that the change in states of the traffic signal (red or green) for a given movement through an intersection can be modeled as a CTMC. Markov chains have numerous properties, some of which might be restrictive for this application. For example, it is assumed that the amount of time the process spends in a given state and the time it spends in the next state visited must be independent random variables. When there are only two states for a given traffic movement and the cycle length is preset, this assumption would not hold. Thus, additional work might be considered to develop another still efficient yet less restrictive solution technique for determining optimal routing strategies in signalized STV networks where the signal timings or delays due to signal operations are uncertain.

In the worst-case, both the Penalty Approach and the ARSC algorithm require additional computation of $O(n)$ (where n is the number of nodes in the network) as compared with the ELB algorithm for solving the same problem but ignoring the effects of signal operations. For most sparse networks, as found in many traffic networks, however, less than fourfold computation of the ELB algorithm is expected for both the Penalty Approach and the ARSC algorithm. Numerical experiments conducted on a real-world-based signalized street network show that significantly improved solutions can be attained by explicitly considering delay to signal control in the path computations and the additional calculations required to explicitly model the extra travel time (i.e. delay) due to signal operations is warranted.

The concepts introduced in this paper for extending the ELB algorithm for use in signalized networks can be directly employed in extending the SDOT (Miller-Hooks, 2001) algorithm, a label-setting approach with better worst-case computational complexity than the ELB algorithm. In certain extensions of these algorithms to real-time routing applications, additional savings in computation time can be achieved through a label-correcting approach. Therefore, our efforts have concentrated on the development of the label-correcting approach (the ELB algorithm) for the problem posed herein.

An extension of this work might consider probabilistic delays, such as queue clearance lost times, where the probability of a vehicle's position in the queue could be forecasted and a PDF for the additional lost time could be estimated. One would need to consider how the algorithm could incorporate these PDFs. When actual signal timings are uncertain, this work assumes independence between signal phases at sequential signalized intersections. One might consider extending the concepts developed here to address more complicated coordinated signal systems, where dependence in delay times at sequential signalized intersections is considered.

Deterministic versions of the combined traffic assignment and control problem have been addressed in several works, including, for example, Allsop and Charlesworth (1977), Smith and Ghali (1990) and Gartner and Al-Malik (1996). While considerable effort would be required to address this problem where the stochasticity in the arc travel times is recognized, we hope that the algorithms proposed in this work will also be helpful in providing suitable path search techniques that might be required for solution of this combined problem by a simulation-based approach or an approach that would employ a feedback mechanism for loading the vehicles onto the network.

Acknowledgements

This work was supported by NSF grant CMS 9875305 and the Weiss Dissertation Fellowship. This support is gratefully acknowledged but implies no endorsement of the findings.

References

- Ahuja, R., Magnanti, T., Orlin, J., 1993. *Network Flows*. Prentice Hall, New Jersey.
- Ahuja, R., Orlin, J., Pallottino, S., Scutellà, M., 2002. Minimum time and minimum cost-path problems in street networks with periodic traffic lights. *Transportation Science* 36, 326–336.
- Allsop, R., Charlesworth, J., 1977. Traffic in a signal-controlled road network: an example of different signal timings inducing different routings. *Traffic Engineering and Control* 18, 262–264.
- Chen, Y., Yang, H., 2000. Shortest path in traffic-light networks. *Transportation Research Part B* 34, 241–253.
- Desrochers, M., Soumis, F., 1988. A generalized permanent labelling algorithm for the shortest path problem with time windows. *INFOR* 26, 191–212.
- Fu, L., Rilett, L., 1998. Expected shortest paths in dynamic and stochastic traffic networks. *Transportation Research Part B* 32, 499–516.
- Gartner, N., Al-Malik, M., 1996. Combined model for signal control and route choice in urban traffic networks. *Transportation Research Record* 1554, 27–35.
- Hall, R., 1986. The fastest path through a network with random time-dependent travel times. *Transportation Science* 20, 182–188.
- Kulkarni, V., 1995. *Modeling and Analysis of Stochastic Systems*. Chapman & Hall, New York (Chapter 6).
- Miller-Hooks, E., 2001. Adaptive least-expected time paths in stochastic, time-varying transportation and data networks. *Networks* 37, 35–52.
- Miller-Hooks, E., Mahmassani, H., 1998. Least possible time paths in stochastic, time-varying networks. *Computers and Operations Research* 25, 1107–1125.
- Miller-Hooks, E., Mahmassani, H., 2000. Least expected time paths in stochastic, time-varying transportation networks. *Transportation Science* 34, 198–215.
- Opasanon, S., Miller-Hooks, E., 2001. Least expected time hyperpaths in stochastic, time-varying multimodal networks. *Transportation Research Record* 1771, 89–96.
- Smith, M., Ghali, M., 1990. The dynamics of traffic assignment and traffic control: a theoretical study. *Transportation Research Part B* 24B, 409–422.
- Wattleworth, J., Shuldiner, P., 1963. Analytical methods in transportation; left-turn penalties in traffic assignment models. *Journal of Engineering Mechanics Division, ASCE* 89, 97–126.
- Ziliaskopoulos, A., Mahmassani, H., 1996. A note on least time path computation considering delays and prohibitions for intersection movements. *Transportation Research Part B* 30, 359–368.