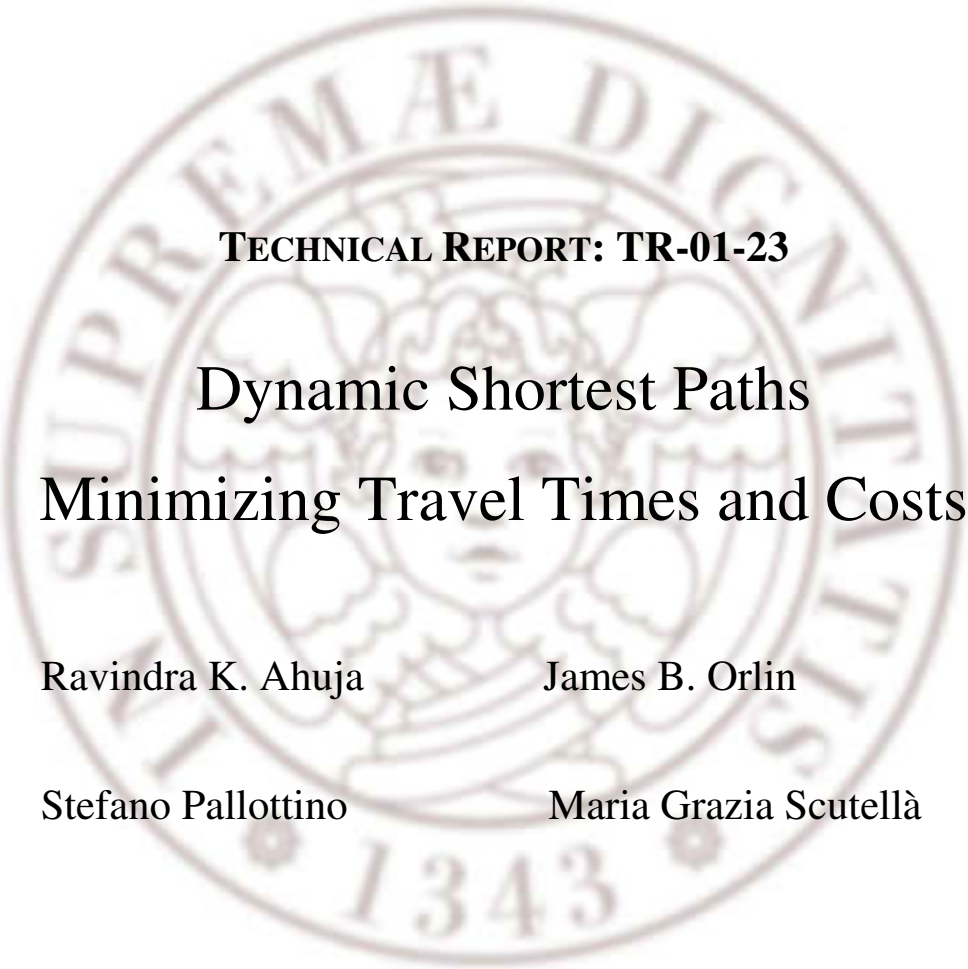
UNIVERSITÀ DI PISA

DIPARTIMENTO DI INFORMATICA



TECHNICAL REPORT: TR-01-23

Dynamic Shortest Paths Minimizing Travel Times and Costs

Ravindra K. Ahuja James B. Orlin

Stefano Pallottino Maria Grazia Scutellà

October 30, 2001

ADDRESS: Corso Italia 40, 56125 Pisa, Italy. TEL: +39 050 2212700, FAX: +39 050 2212726

Dynamic Shortest Paths Minimizing Travel Times and Costs

Ravindra K. Ahuja
Department of Industrial & Systems Engineering, University of Florida, Gainesville, FL 32611, USA
ahuja@ufl.edu

James B. Orlin
Sloan School of Management, Massachusetts Institute of Technology, Cambridge, MA 02139, USA
jorlin@mit.edu

Stefano Pallottino
Dipartimento di Informatica, Università di Pisa, 56125 Pisa, Italy, pallo@di.unipi.it

Maria G. Scutellà
Dipartimento di Informatica, Università di Pisa, 56125 Pisa, Italy, scut@di.unipi.it

Abstract

In this paper, we study dynamic shortest path problems, which is to determine a shortest path from a specified source node to every other node in the network where arc travel times change dynamically. We consider two problems: the minimum time walk problem (which is to find a walk with the minimum travel time) and the minimum cost walk problem (which is to find a walk with the minimum travel cost). The minimum time walk problem is known to be polynomially solvable for a class of networks called *FIFO networks*. This paper makes the following contributions: (i) we show that the minimum cost walk problem is an NP-complete problem; (ii) we develop a pseudopolynomial-time algorithm to solve the minimum cost walk problem (for integer travel times); and (iii) we develop a polynomial-time algorithm for the minimum time walk problem arising in road networks with traffic lights.

1. Introduction

Consider a directed network G = (N, A) with node set N and arc set A. Let n = |N| and m = |A|. Let node s denote a specified node, called the *source node*. In the network G, a *directed walk* W is a sequence of nodes i_1- i_2 - ... - i_{r-1}- i_r satisfying the property that for all $1 \le k \le r-1$, $(i_k, i_{k+1}) \in A$. In a directed walk, some nodes and some arcs can be visited more than once. A directed walk is a *directed path* if no node is repeated. In this paper, for brevity we shall refer to directed walks by walks and directed paths by paths.

In this paper, we consider shortest path problems in dynamic networks, that is, networks where travel times change with time. We consider the problem of determining a shortest path from the source node to every other node in the network. Dynamic path problems have been extensively investigated in the literature. Some selected references on these problems are: Chabini [1997, 1998, 2000], Dreyfus [1969], Kaufmann and Smith [1993], Kirby and Potts [1969], Orda and Rom [1990, 1991], Pallottino and Scutellà [1998], Ziliaskopoulos [1994], Ziliaskopoulos and Mahmassani [1993, 1996], Wardell and Ziliaskopoulos [2000].

Let $d_{ij}(t)$ denote the travel time of arc (i, j) starting at time t; that is, the time spent in traversing arc (i, j) if we start traversing the arc at time t. If one starts at node i at time t and travels on arc (i, j), then one arrives at node j at time $d_{ij}(t) + t$. We assume that if t is integer, then $d_{ij}(t)$ is also integer. We refer to this assumption as the *integer travel time* assumption. We also assume that all arcs have non-negative travel times. Although we permit arcs with a travel time of 0, we assume that all directed cycle in G have a travel times at least 1.

We say that arc (i, j) satisfies the *FIFO property* if for each pair t, t' of times with $t < t'$, $d_{ij}(t) + t \leq d_{ij}(t') + t'$. This implies that a person traveling on arc (i, j) starting at time t arrives at node j no later than a person traveling in arc (i, j) starting at time t'. The FIFO property further implies that it will not save time for a traveler to wait at node i before traversing the arc. For example, if one waits at node i for five minutes and then starts traversing arc (i, j), then the travel time of the arc might be less, but by no more than five minutes. We say that a network is a FIFO network if each arc in it satisfies the FIFO property. Our results in Section 4 assume that the network G is a FIFO network.

The minimum time walk problem in a FIFO network is polynomially solvable (Kaufmann and Smith [1993]). The minimum time walk problem in a non-FIFO network is NP-hard (Orda and Rom [1990]).

Let d^*_{ij} denote the minimum possible travel time on arc (i, j), that is, $d^*_{ij} = \min\{d_{ij}(t)$: all t$\}$. Let $e_{ij}(t) = d_{ij}(t) - d^*_{ij}$ denote the *excess time* for traversing arc (i, j) starting at time t; it is the difference between the actual travel time and the minimum possible travel time. Let e* denote the maximum excess time on any arc, that is,

$$e^* = \max\{d_{ij}(t) - d^*_{ij}: \text{all } (i, j) \in A, \text{ and all } t\}. \quad (1)$$

For each arc $(i, j) \in A$, and for each time t, we define the cost of arc (i, j) as

$$c_{ij}(t) = \alpha d^*_{ij} + \beta e_{ij}(t),$$

where α and β are specified non-negative parameters. In other words, we assume that the cost of an arc depends linearly on the minimum possible travel time for the arc and the excess time for the arc. This also can be used to model the situations in which the arc cost depends linearly on $d_{ij}(t) = d^*_{ij} + e_{ij}(t)$ and on $e_{ij}(t)$.

If $\beta = \alpha$, then the cost of an arc is α times its travel time. If $\beta = 0$, the cost of an arc is the minimum travel time of the arc. If $\alpha = 0$, then the cost of an arc is its excess time. If $0 < \beta < \alpha$, this reflects a situation in which excess time is preferable to driving time. For example, waiting time in a cab is less costly than driving time. If $\alpha < \beta$, this reflects a situation in which driving time is preferable to excess time. For example, many drivers prefer uncongested paths to congested paths even if both lead to the same overall driving time. Hence by choosing α and β appropriately, we can model several practical utility functions.

Consider a walk $W = i_1\text{-}i_2\text{-}i_3\text{-} \ldots \text{-} i_k$. We denote by d(W) the travel time of W starting at time 0. In other words,

$$d(W) = d_{i_1i_2}(0) + d_{i_2i_3}(d_{i_1i_2}(0)) + d_{i_3i_4}(d_{i_1i_2}(0) + d_{i_2i_3}(d_{i_1i_2}(0))) + \ldots.$$

We denote by d*(W) as the sum of the minimum possible travel times of arcs in W, that is,

$$d^*(W) = \sum_{(i,j)\in W} d^*_{ij}.$$

We point out that d*(W) is a lower bound on the time taken to traverse the walk W and may be a strict lower bound. There may not be any W with travel time equal to d*(W). We denote by e(W) the excess time of W starting at node 0. Thus,

$$e(W) = d(W) - d^*(W)$$

The cost of a walk starting at time 0 is the sum of the arc costs in the walk:

$c(W) = \alpha d^*(W) + \beta e(W)$.

We study the following problems in this paper:

Minimum Time Walk Problem: Determine a walk W from node s to every other node starting at time 0 with the minimum value of d(W).

Minimum Excess Time Walk Problem: Determine a walk W from node s to every other node starting at time 0 with the minimum value of e(W).

Minimum Cost Walk Problem: Determine a walk W from node s to every other node starting at time 0 with the minimum value of c(W).

As stated above, the minimum time walk problem and the minimum excess time walk problem are both special cases of the minimum cost walk problem. In this paper, we study walk problems rather than path problems which are more popular in literature because in general a minimum cost walk might have a lower cost than a minimum cost path. To illustrate, suppose we wanted to get from our own driveway to someone's house, and that we had a circular driveway, and that there was a traffic light at the end of the driveway. If we disliked waiting at red lights (and accordingly, $\beta >> \alpha$), we might circle our driveway (possibly many times) until the light was about to turn green, and then exit.

The minimum time walk problem in a FIFO network can be solved by a slight modification of Dijkstra's algorithm, provided $d_{ij}(t) \geq 0$ for each arc $(i, j) \in A$ and every t. This algorithm consists of applying the standard Dijkstra's algorithm (see, for example, Ahuja, Magnanti and Orlin [1993]) with the modification that when node i is permanently labeled and receives a permanent label of b(i), then the travel time of any arc (i, j) emanating from this node is taken to be $d_{ij}(b(i))$. This algorithm can be implemented in $O(m + n \log n)$ time using Fibonacci heap implementation (Fredman and Tarjan [1984]). Kaufmann and Smith [1993] showed that in a FIFO network there always exists a minimum time walk that is a path, and thus it does not pay to repeat nodes.

In this paper, we make the following contributions:

1. We show that the minimum excess time walk problem (and hence the minimum cost walk problem) in a FIFO network is NP-complete if travel times are expressed using interval encodings.

2. We show that we can solve the minimum cost walk problem in a network (not necessarily a FIFO network) in $O(\beta nme^*/\min(\alpha, \beta))$ time if $d_{ij}(t) \geq 0$ for all (i, j) and if there are no directed cycles whose travel time is 0. These are pseudo-polynomial algorithms to solve the minimum cost walk problem. In particular, the running time depends on e* rather than on the travel times for the arcs.

3. We consider an important shortest walk problem arising in road networks with traffic lights and transform it to the minimum time walk problem, thereby allowing it to be solved in O(n log n) time. We show that the minimum excess time walk problem in this road networks is strong NP-complete; that it, it is NP-complete even if all excess times (waiting times) at traffic lights are bounded by a polynomial and if traffic times in any arc are independent of the starting times.

4. We consider the generalization of the dynamic minimum cost walk problem to road networks with traffic lights in which the maximum cycle time for a traffic light phase is p*. We show that the minimum cost problem is solvable in $O((\beta nm(p^* + e^*)/\min(\alpha, \beta))$ time. We also show that if $\alpha = 0$, then there is no algorithm that runs in time polynomial in n, m, e*, and p* unless P = NP.

2. Complexity of the Minimum Excess Time Walk Problem

In this section, we show that the minimum excess time walk problem is NP-complete. To establish this result, we must first describe how do we describe in polynomial space an instance of the minimum excess

time walk problem. To describe an instance of the minimum excess time walk problem, we need to describe the network G = (N, A) which can be done in O(m) space. We also need to describe $d_{ij}(t)$ for each (i, j) and for all times t. We assume that there is an upper bound T on the travel time between any pair of nodes in the network. Hence, the time interval [0, T] can be regarded as the range of our time horizon. In this case, we can define $d_{ij}(t)$ for all t, $0 \le t \le T$, using a scheme called *interval encoding*, where we list travel times as a set of quintuples <i, j, a, b, f> with the meaning that $d_{ij}(t) = f$ if $a \le t \le b$. We need at most T+1 quintuples to describe $d_{ij}(t)$ for all t, $0 \le t \le T$. We can thus describe an instance of the minimum excess time walk problem in O(mT) time, but often it can be represented much more efficiently. Let $Q \le mT$ denote the number of quintuples needed to state the problem. Observe that $Q \ge m$ since we need at least one quintuple for every arc. We can also assume that $Q \ge m \ge n$. Hence Q denotes the input size of the problem.

The NP-completeness result is established by transforming the number partition problem to the minimum cost walk problem. We define the number partition problem next.

Number Partition Problem. Given n integer numbers a_1, a_2, ..., a_n, and another integer number b, does there exist a 0-1 vector x such that $\sum_{i=1}^{n} a_i x_i = b$?

Theorem 1. *The minimum excess time walk problem is NP-complete if the problem is specified using an interval encoding.*

Proof. Given a number partition problem, we will transform it into a minimum excess time walk problem on the network shown in Figure 1. We construct the network G = (N, A) with nodes N = {1, 2, ..., n+2}. For each i = 1 to n, we introduce two arcs from i to i+1, one with $d_{i,i+1}(t) = 0$ and one with $d_{i,i+1}(t) = a_i$. Further, there is one arc (n+1, n+2) with $d_{n+1,n+2}(t) = 1$ for $t \ne b$ and $d_{n+1,n+2}(b) = 0$. Notice that only one arc (n+1, n+2) has time-dependent travel time; all other arcs have fixed travel times. We observe that this network can be represented by O(n) quintuples using interval encodings; hence Q = O(n).

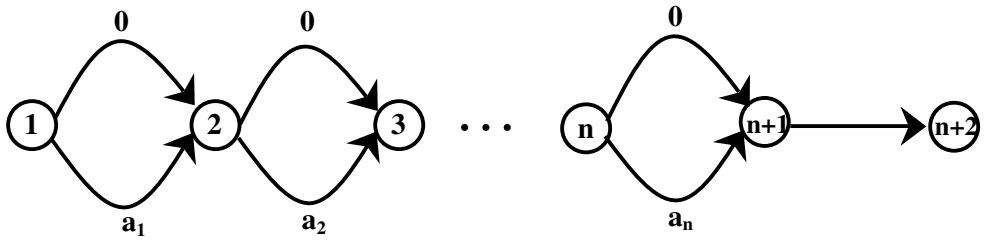


Figure 1. Transforming a number partition problem into a minimum excess time walk problem.

We claim that the number partition problem has a feasible solution if and only if there is a walk in G from node 1 to node n+2 with no excess time. First assume that there is a feasible solution x for the number partition problem. Let W be a walk from node 1 to node n+2 defined as follows: (i) if $x_i = 1$, then $(i, i+1) \in W$ with travel time a_i; and (ii) if $x_i = 0$, then $(i, i+1) \notin W$. The walk W also contains the arc (n+1, n+2). Observe that W takes b time units to arrive node n+1, and then arrives at node n+2 with no excess time. This establishes the first part of the claim. Conversely, let W be a path from 1 to n+2 with no excess time. Then W must include a sub-walk W′ from node 1 to node n+1 that takes b units of time. Let $x_i = 1$ if there is an arc (i, i+1) in W′ with travel time a_i, and let $x_i = 0$ otherwise. Thus, x is a feasible solution to the number partition problem. This establishes the second part of the claim, and completes the proof of the theorem. ♦

We can make the following observations from the proof of the above theorem:

Observation 1: *The proof of Theorem 1 implies that the minimum excess time walk problem is NP-complete even if we restrict to problems in which e* = 1.*

Observation 2: *The network we construct to transform the number partition problem into the minimum excess time walk problem is a FIFO network. This implies that the minimum excess time walk problem in a FIFO network is also NP-complete.*

Observation 3: *The number partition problem is not a strong NP-complete problem. Hence, the minimum excess time walk problem is also not a strong NP-complete problem. Consequently, we may develop a pseudo-polynomial time algorithm to solve it. We describe one such algorithm is Section 3.*

Observations 4: *The minimum excess time walk problem is a special case of the minimum cost walk problem. Hence the minimum cost walk problem is also NP-complete.*

3. A Pseudo-Polynomial Time Algorithm for the Minimum Cost Walk Problem

In the previous section, we have shown that the minimum excess time problem is an NP-hard problem by showing that those instances which can be represented in O(n) space cannot be solved by algorithms with running time bounded by a polynomial in n unless P = NP. In this section, we shall study the general minimum excess time problem which may take as much as mT space to specify the problem data. Alternatively, Q = O(mT).

In the previous section, we also established that the minimum cost walk problem is NP-complete if $\alpha = 0$ and $\beta > 0$, in which case the problem is the minimum excess time walk problem. If $\beta = 0$ and $\alpha > 0$, then the minimum cost walk problem is a shortest path problem on a static network using costs d*, and thus can be solved using Dijkstra's algorithm. In this section, we consider the remaining case, that is, we assume that $\alpha > 0$ and $\beta > 0$. If we view both α and β as fixed, the algorithm that we develop to solve the minimum cost walk problem runs in time that is polynomial in n, m, and e*. In the case that e* itself is polynomial in n and m, this algorithm is polynomial regardless of the magnitudes of the actual travel times.

Working with the Time Expanded Network

A standard approach for dealing with dynamic network problems is to construct a time-expanded network. We denote the time-expanded network by G^T. For a given network G = (N, A), the network $G^T = (N^T, A^T)$ is as follows. For each node $i \in N$, there are T+1 copies $i_0, i_1, i_2, ..., i_T$ in N^T, where node i_k represents node i in G at time k. For each arc $(i, j) \in A$, there are multiple arcs (at most T) corresponding to different starting times. There is an arc (i_k, j_l) in the time-expanded network whenever $(i, j) \in A$, $l - k = d_{ij}(k)$, and $l \leq T$. The arc (i_k, j_l) in the time-expanded network represents traveling along arc (i, j) starting from node i at time k and arriving at node j at time $l = k + d_{ij}(k)$. We set the cost of arc (i_k, j_l) in the time-expanded network equal to $\alpha d^*_{ij} + \beta(d_{ij}(k) - d^*_{ij})$. By assumption, $d_{ij}(k) \geq 0$ for each (i, j) and every k, and G has no directed cycles whose travel time is 0. This will imply that G^T is an acyclic network. The following property is a direct outcome of the construction of G^T.

Property 1. *For each walk in G from node i to node j with travel time $l \leq T$, there is a path in G^T from node i_0 to node j_l. Moreover, the walk and path have the same cost. Similarly, for each path in G^T from node i_0 to node j_l, there is a walk in G from node i to node j with travel time l, and these walks have the same cost.*

It is well known that the shortest path problem (that is, determining minimum cost path from node s to every other node) can be solved in time linear in the number of arcs in the network. Since G^T has O(mT) arcs, we can solve the shortest path problem in O(mT) time. In this section, we will develop an algorithm that runs much faster than O(mT) time.

Working with Reduced Travel Times

Suppose that we solve the shortest path problem in G with respect to the travel times d^*_{ij}. Let P(k) denote the shortest path from node s to node k and $\pi(k)$ denote its travel time. It follows from the shortest path optimality conditions (see, for example, Ahuja, Magnanti and Orlin [1993]) that

$$\pi(j) \le \pi(i) + d^*_{ij} \text{ for each arc } (i, j) \in A, \text{ and} \tag{2a}$$

$$\pi(j) = \pi(i) + d^*_{ij} \text{ for each arc } (i, j) \in P(k) \text{ and each node } k \in N\backslash\{s\}. \tag{2b}$$

It is easy to see that $\pi(i)$ is a lower bound on the travel time from node s to node i (see, for example, Ahuja, Magnanti and Orlin [1993]). We now define the *reduced minimum travel time*, $d^{\pi*}_{ij}$, with respect to π as follows: $d^{\pi*}_{ij} = d^*_{ij} + \pi(i) - \pi(j)$. It follows from (2a) that $d^{\pi*}_{ij} \ge 0$ for each arc $(i, j) \in A$, and $d^{\pi*}_{ij} = 0$ for each arc $(i, j) \in P(k)$ and each node $k \in N\backslash\{s\}$. Let $d^{\pi}_{ij}(t) = d_{ij}(t) + \pi(i) - \pi(j)$ denote the reduced travel time of arc (i, j) and $c^{\pi}_{ij}(t) = \alpha d^{\pi*}_{ij} + \beta e_{ij}(t)$ denote the reduced cost of arc (i, j) at time t. The following lemma shows that solving the minimum cost walk problem with respect to c is equivalent to solving it with respect to the reduced costs c^{π}.

Lemma 1. *A walk W in G from node s to node k is a minimum cost walk with respect to the costs $c_{ij}(t)$ if and only if it is a minimum cost walk with respect to the reduced costs $c^{\pi}_{ij}(t)$.*

Proof. Consider a walk W starting at node s and terminating at node k in G. Then,

$$c(W) - c^{\pi}(W) = \alpha(d^*(W) - d^{\pi*}(W)) = \alpha \sum_{(i,j)\in W} d^*_{ij} - \alpha(\sum_{(i,j)\in W}(d^*_{ij} + \pi(i) - \pi(j)) = \alpha\pi(k),$$

which is a constant for each node k. Therefore, a minimum cost walk from s to k with respect to c will also be a minimum cost walk from s to k with respect to c^{π}. ♦

Lemma 2. *Let $p = \beta ne^*/\min(\alpha, \beta)$. Then, every minimum cost walk from node s to any other node k will have reduced travel time no more than p.*

Proof. Recall that P(k) denotes the shortest path from node s to node k with d^*_{ij} as travel times. We will henceforth refer to P(k) as P. The reduced cost of the path P is

$$c^{\pi}(P) = \alpha d^{\pi*}(P) + \beta e(P) = \beta e(P) \le \beta(n-1)e^*, \tag{3}$$

where the second equality follows from (2b) and the last inequality follows from the fact that P has at most n-1 arcs, and each arc has an excess time of at most e*. We will now show that any walk P' with reduced travel time at least p will have reduced cost greater than $c^{\pi}(P)$, and therefore will not be better than the path P. We will prove this result for two cases: $\alpha \ge \beta$ and $\alpha < \beta$.

Suppose first that $\alpha \ge \beta$. In this case, $p = \beta ne^*/\min(\alpha, \beta) = ne^*$. Any walk P' with reduced travel time at least p has a reduced cost at least $\alpha p \ge \beta p$. Since $p = ne^*$, then the walk P' has a reduced cost of at least $\beta ne^* > c^{\pi}(P)$ (from (3)). Next consider the case when $\alpha < \beta$. In this case, $p = \beta ne^*/\min(\alpha,\beta) = \beta ne^*/\alpha$. Any walk P' with reduced travel time at least p has a reduced cost of at least αp. Since $p = \beta ne^*/\alpha$, the reduced cost of the walk P' is at least $\beta ne^* > c^{\pi}(P)$ (from (3)). So, in both cases, the reduced cost of P = P(k) strictly dominates the reduced cost of any walk P' with reduced travel time at least $\beta ne^*/\min(\alpha, \beta)$. This proves the theorem. ♦

The Algorithm

We are now in a position to describe our algorithm for the minimum cost walk problem.

Step 1. Compute the shortest paths from node s to all other nodes using travel times d*. Let $\pi(j)$ denote the least travel time from node s to node j.

Step 2. Compute the reduced travel times $d_{ij}^{\pi*}(t)$ as well as the reduced costs $c_{ij}^{\pi}(t)$. Construct the time-expanded network G^p with $p = \beta ne^*/\min(\alpha, \beta)$ in a manner as described earlier but using reduced travel times $d_{ij}^{\pi*}(t)$ instead of the travel times $d_{ij}(t)$, and using cost of arc (i_k, j_l) in the time-expanded network G^p equal to $\alpha d_{ij}^{\pi*} + \beta(d_{ij}(t) - d_{ij})$. Find the minimum cost path from node s_0 to every other node in G^p.

It follows from Lemma 1 that in G each directed cycle has a reduced travel time of at least 1, and thus the time expanded network G^p is acyclic. Since G^p has O(np) nodes and O(mp) arcs, we can determine minimum cost path from node s_0 to all other nodes in $O(mp) = O(\beta nme^*/\min(\alpha, \beta))$ time. Hence the following theorem.

Theorem 2. *The minimum cost walk problem in a network G = (N, A) with integer travel times, where $d_{ij}(t) \geq 0$ for all (i, j) and for all t and where there is no 0-time cycle, can be solved in $O(\beta nme^*/\min(\alpha, \beta))$ time.*

4. Shortest Routes in Networks with Traffic Lights

The research in this paper was motivated by the following application. Suppose that we want to go from an origin node s to a destination node t in a (road) network G = (N, A) with n = |N| and m = |A|. We assume that G is a directed network where a two-way street is represented by two arcs representing movement in both directions. We assume (as is typical for any road network) that the number of arcs incident on any node are bounded by a constant number. For example, if there is a four-road junction with two-way traffic on each road, then there will be eight arcs incident on the node representing this junction. Thus, we assume that m = O(n). Let d(a, t) denote the travel time on arc "a" starting at time t. For notational convenience in this section, we refer to an arc as "a" instead of (i, j). We assume that d(a, t) satisfies the FIFO property for every arc a.

At some intersections in the road network, there are traffic lights. For simplicity, we assume that each traffic light alternates between red and green lights in a specified fixed pattern. Our approach, however, can be easily extended to the case when traffic light pattern changes with time. As a person travels in this network, travel times depend upon two things: d(a, t) the time needed to traverse arc a, and the *waiting time* due to the red light before we start traversing the arc. This problem has been studied by Ahuja et al. [2001] in the past; in this paper we give a different algorithm to solve this problem. We also prove new complexity results about this problem.

We study the following three problems:

Minimum Time Route Problem. From node s to every other node in the network, determine a walk with the least travel time in the network.

Minimum Excess Time Route Problem. From node s to every other node in the network, determine a walk with the least waiting time.

Minimum Cost Route Problem. From node s to every other node in the network, determine a walk with the least cost, which is a linear function of the minimum travel times and the waiting times.

The difference in these problems versus the problems treated in the previous section is that any waiting time at a traffic light is treated as the excess time, but this waiting is not associated with an arc. We will show in this section that the minimum time route problem can be transformed to the minimum time walk problem in an appropriately defined network and hence can be solved very efficiently. We will also show that the minimum cost route problem can be transformed to the minimum cost walk problem and hence can be solved in pseudo-polynomial time using the algorithm described in Section 3.

We show in Figure 2(a) an example of a traffic junction created by the intersection of a one-way street with a two-way street. Figure 2(b) shows the corresponding graph at this node. We say that a pair of arcs a and b define a *turn* if the head node of arc a is the same as the tail node of arc b; we denote this turn as <a, b>. We say that a turn is a *valid turn* if one can legally take that turn. For example, the situation depicted in Figure 2(a) permits the following seven valid turns: <a, b>, <a, c>, <a, d>, <b, c>, <b, d>, <c, b> and <c, d> (hence there are seven arcs incident to the node representing this junction in the corresponding network). We assume that each valid turn has a repetitive pattern of green and red lights phases. We specify this sequence for each turn by defining three values for each valid turn:

g(a, b): the duration of the green phase;
r(a, b): the duration of the red phase; and
s(a, b): the latest time before time 0 when the green phase begins.

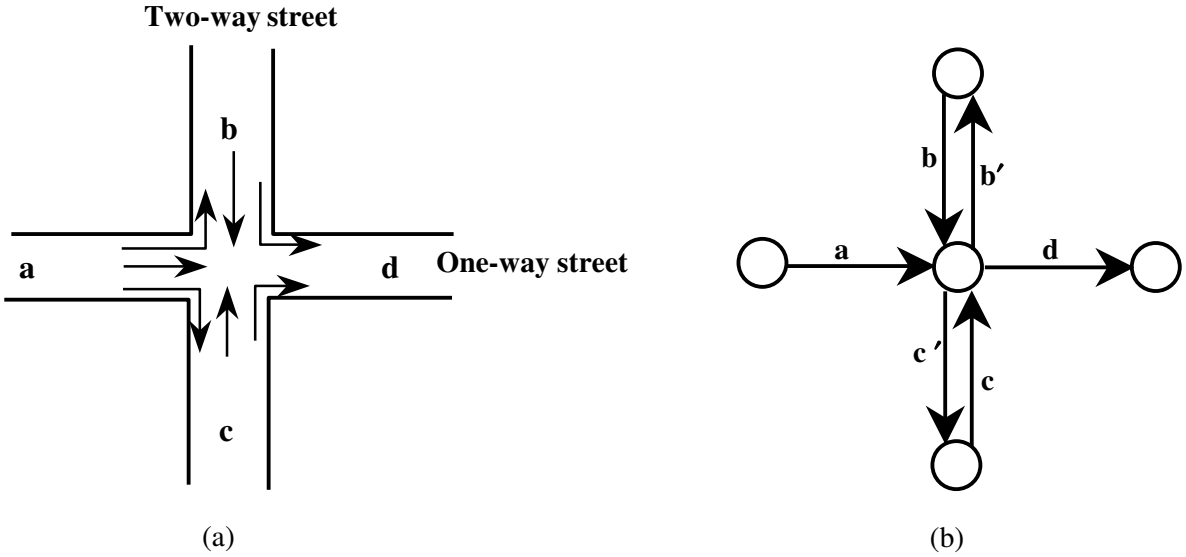


Figure 2. (a) An example of a traffic junction.
(b) The corresponding network.

The value (r(a, b) + g(a, b)) denotes the *cycle time*, the time after which the light pattern repeats itself. The value s(a, b) allows us to identify for what time intervals lights will be red and green. We need to explain further what do we mean by s(a, b). We illustrate it using the two examples shown in Figure 3. For the situation depicted in Figure 3(a), s(a, b) = 8, and for the situation shown in Figure 3(b), s(a, b) = 4. For both the situations, g(a, b) = 6, and r(a, b) = 4.

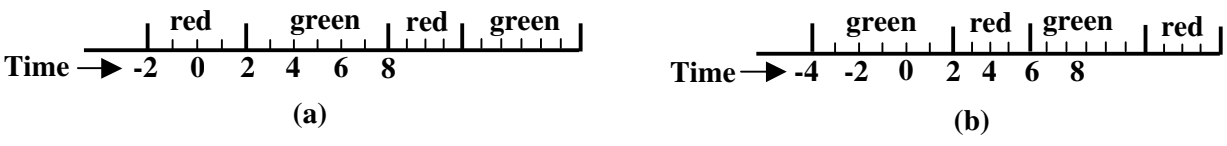


Figure 3. Illustrating the definitions of g(a, b), r(a, b), and s(a, b).

Now suppose that we know the triplet {g(a, b), r(a, b), s(a, b)} for each valid turn <a, b>. Also suppose that we arrive at the head node of arc a at time t and we need to turn into arc b. We want to determine that for how long we need to wait before there is a green light. The waiting time for any turn <a, b> is a function of both a and b and when we are ready to take the turn. We will now derive the expression for the waiting time w(a, b, t) which denotes the time we need to wait at turn <a, b> starting at time t (when we are ready to take the turn). Let $\bar{t}$ = t mod(r(a, b) + g(a, b)). It is easy to see that $\bar{t} \leq$ (r(a, b) + g(a, b)) and the green or red phase at time t will be the same as at time t.

Now suppose that s(a, b) = 0. In this case, the green phase begins at time 0. It is easy to see that the waiting time in this case will be given by the following expression:

$$w(a,b,t) = \begin{cases} 0 & \text{if } \bar{t} \leq g(a, b), \\ \bar{t} - g(a, b) & \text{otherwise.} \end{cases} \quad (4)$$

Next consider the case when s(a, b) > 0. In this case, the green phase starts at time –s(a, b). Let $\bar{t}$ = (t + s(a, b)) mod(r(a, b) + g(a, b)). It is easy to see that $\bar{t}$ gives the time elapsed since the latest green phase before time t. Hence using (4) we can determine the waiting time for the turn.

In order to capture the effects of traffic lights through the model described in Section 3, we will transform the graph G to a graph G′ = (N′, A′) that satisfies the assumptions stated for Section 3. Our transformation is similar to that for the shortest paths with turn penalties (see, for example, Ahuja, Magnanti and Orlin [1993], pp. 130). For the graph in Figure 2(a), we show this transformation in Figure 4. In the transformed network, we take a node with k incident arcs and "split" it into k nodes so that each incoming arc ends at a separate node and each outgoing arc emanates from a separate node. For each valid turn <a, b>, we create a "connection" arc (a, b) in the transformed network connecting the inbound arc a to the outbound arc b. Hence, the network G′ has two types of arcs – arcs that are also in G, which we refer to them as *regular arcs*, and *connection arcs* that are not in G. Observe that at each junction node, there are as many connection arcs as the number of turns allowed at the traffic junction. This transformation increases the size of the network. However, since the number of arcs in a traffic network incident to a node is a constant number (typically, no more than 6), the number of arcs and nodes are multiplied by a constant number to obtain G′. Consequently, n′ = |N′| = O(n) and m′ = |A′| = O(n).

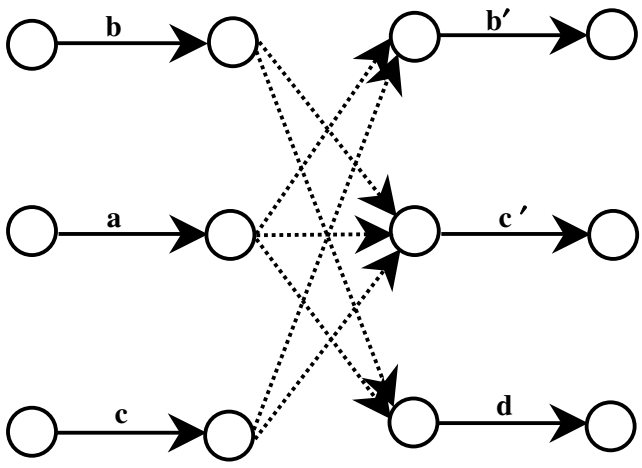


Figure 4. Transforming the minimum time route problem into a minimum time walk problem.

In the transformed network G′, we set the travel time of a regular arc a at time t equal to d(a, t). We set the travel time of a connection arc (a, b) at time t equal to w(a, b, t). (Note that the minimum times for these arcs are 0, and so w(a, b, t) is viewed as excess time, as we stated earlier.) It is easy to observe that there is one-to-one correspondence between paths in G and G′ and they have the same travel times. By

assumption, regular arcs in A′ satisfy the FIFO property. It is also easy to see that the connection arcs too satisfy the FIFO property. Hence the graph G′ satisfies the FIFO property, and we can solve the shortest path problem in it O(m + n log n) time = O(n log n) time (Kirby and Potts [1969], Kaufmann and Smith [1993], and Pallottino and Scutellà [1998]). We summarize our discussion with the following theorem.

Theorem 3. *The minimum time route problem in a traffic network can be solved in O(n log n) time.*

Let w^* be the maximum phase for a light, that is $w^* = \max\{r(a, b) + g(a, b): \langle a, b\rangle \text{ is a valid turn}\}$. Then w^* is an upper bound on the excess time at all connection arcs. As before, e^* is an upper bound on the excess time at all regular arcs. We can use the results of Section 3, and obtain the following theorem.

Theorem 4. *If $\alpha, \beta > 0$, then the minimum cost route problem in a traffic network can be solved in $O(\beta nm(e^*+w^*)/\min(\alpha,\beta))$ time.*

It follows from Theorem 1 that the minimum cost route problem is a NP-complete problem even if there is traffic light at just one junction. We will strengthen this result to show that the minimum cost route problem is NP-complete even if $e^* = 0$, and w^* is bounded by a polynomial in n. In other words, the NP-completeness for the traffic light problem can be derived from the effects of traffic lights rather than the effects of different travel times on the regular arcs, and does not rely on long traffic signal phases.

Theorem 5. *The problem of minimizing waiting times at traffic lights is NP-complete even if restricted to problems in which $e^* = 0$ and $w^* \le n^2$.*

Proof. We will carry out a transformation as in Theorem 1 from the number partition problem. The number partition problem remains NP-complete even if we restrict attention to instances in which $b \le 2^n$. Let $q_1, q_2, ..., q_k$ denote the prime numbers that are less than n^2. By the prime number theorem, k is approximately $n^2/(\ln n^2) > n^{3/2}$. Therefore, $\prod_{i=1}^{k} q_i > 2^n > b$.

We first create nodes 1 to n+1 as in the transformation used in the proof of Theorem 1. We also create nodes n+2, ..., n+k+1 such that the travel time (ignoring lights) to get from node n+j to n+j+1 is 0 for each j = 1 to k. We also create a sequence of k traffic lights, one on each of n+1, ..., n+k such that the time between the start of two successive green phases at the light at node n+j is q_j. This light is green for exactly one time unit per phase followed by a red interval of q_j - 1 time units. Finally, the initial green interval is chosen so that each light is green at time b. Let $g_j \equiv b \pmod{q_j}$ for j = 1 to k.

By construction, if one arrives at time b at traffic node n+1, then one can travel from node n+1 to node n+k+1 in 0 time units, since each traffic light will be green and all travel times are 0. Moreover, if one arrives at node n+1 at any time $t < 2^n$ such that $t \ne b$, then at least one of the traffic lights on nodes n+1, ..., n+k will be red. To see this, note that if all lights are green at time t, then $t \equiv g_j \pmod{q_j}$ for all j. But since $\prod_{i=1}^{k} q_i > 2^n$, there is at most one value t satisfying these k congruences such that $t < 2^n$. So, it must be the case that t = b.

Since nodes 1 to n+1 have the same construction as in the proof of Theorem 1, there is a solution to the number partition problem if and only if there is a path from node 1 to node n+1 of travel time b. This occurs if and only if there is a path from node 1 to node n+k+1 with no waiting time at traffic lights. This completes the proof of the theorem. ♦

Acknowledgements

The first author gratefully acknowledges the support of NSF Grant DMI-9900087. The research of the second author was supported by the NSF Grant DMI-9820998 and the Office of Naval Research Grant ONR

N00014-98-1-0317. The other two authors have been supported by the MURST-Cofinanziato and INDAM-GNAMPA.

References

Ahuja, R.K., T.L. Magnanti and J.B. Orlin. 1993. *Network flows: Theory, Algorithms, and Applications*, Prentice Hall, Englewood Cliffs, NJ.

Ahuja, R.K., J. B. Orlin, S. Pallottino, and M. G. Scutellà. 2001. Minimum time and minimum cost path problems in street networks with periodic traffic lights. To appear in *Transportation Science*.

Chabini, I. 1997. A new short path algorithm for discrete dynamic networks. *Proceedings of the 8th IFAC Symposium on Transport Systems*, Chania, Greece, June 16-17, 551-556.

Chabini, I. 1998. Discrete dynamic shortest path problems in transportation applications: Complexity and algorithms with optimal run time. *Transportation Research Record* **1645**, 170-175.

Chabini, I. 2000. Fastest paths in dynamic networks. To appear in *Transportation Science*.

Dreyfus, S.E. 1969. An appraisal of some shortest-path algorithms. *Operations Research* **17**, 395-412.

Fredman, M.L., and R.E. Tarjan. 1984. Fibonacci heaps and their uses in improved network optimization algorithms. *Proceedings of the 25th Annual IEEE Symposium on Foundations of Computer Sci*ence, pp. 338-346. Full paper in *Journal of ACM* **34**(1987), 596-615.

Kaufmann, D.E. and R.L. Smith. 1993. Fastest paths in time-dependent networks for intelligent vehicle-highway systems application. *IVHS Journal* **1**, 1-11.

Kirby, R.F. and R.B. Potts. 1969. The minimum route problem for networks with turn penalties and prohibitions. *Transportation Research* **3**, 397-408.

Orda, A. and R. Rom. 1990. Shortest-path and minimum-delay algorithms in network with time-dependent edge length. *Journal of the ACM* **37**, 607-625.

Orda, A. and R. Rom. 1991. Minimum weight paths in time-dependent network. *Networks* **21**, 295-320.

Pallottino, S. and M.G. Scutellà. 1998. Shortest path algorithms in transportation models: Classical and innovative aspects. *Equilibrium and Advanced Transportation Modelling*, P. Marcotte and S. Nguyen (Eds.), Kluwer Academic Publishers, 245-281.

Wardell, W. W., and A. K. Ziliaskopoulos. 2000. A intermodal optimum path algorithm for dynamic multimodal networks. *European Journal of Operational Research* **125**, 486-502.

Ziliaskopoulos, A.K. 1994. Optimum path algorithms on multidimensional networks: Analysis, design, implementation and computational experience. Ph.D. Dissertation, University of Texas at Austin.

Ziliaskopoulos, A.K. and H.S. Mahmassani. 1996. On finding least time paths considering delays for intersection movements. *Transportation Research*, Part B, **30B**, 359-367.