

Time Dependency in Multiple Objective Dynamic Programming

MICHAEL M. KOSTREVA AND MALGORZATA M. WIECEK

*Department of Mathematical Sciences, Clemson University,
Clemson, South Carolina 29634-1907*

Submitted by E. Stanley Lee

Received May 1, 1991

1. INTRODUCTION AND PRELIMINARIES

The problem of planning paths in a network structure is important for many applications. Interest in path planning is strong in transportation, telecommunications, computer design, and fire hazard analysis. Such a problem is sometimes called a routing problem, or a shortest path problem. One of the earliest solutions to the problem was given by Bellman [1]. Under the assumptions of constant travel times on each link, dynamic programming was applied to compute the path of minimum travel time through the network, from any node to a given destination node. Bellman applied the functional equations approach to devise an iterative algorithm which converges to the solution in at most $N - 1$ steps for a network with N nodes.

Interest in the problem arose recently from fire hazard analysis [12]. In particular, it is desired to construct realistic models of the egress of humans from a residential building which is involved in fire. To add realism to the models, it was suggested to consider dynamic networks (costs on links are functions of time) and multi-objective behavior of humans. It seems clear that a human, taking a quick look at a scene in a burning building, will not consider only the time required to travel along a path, but also whether the path is cluttered with obstructions, whether smoke is present, whether fire is present, the distance of the path, the sound of someone calling, and so on. Human integration and automatic consideration of trade-offs are evident in decisions that they make. Although it is not evident that the complete data needed to solve a complex model such as will be proposed is actually available to humans in egress situations, the models are viewed as idealizations to use for standards or benchmarks. Once the model solutions

289 - 307

0022-247X/93 \$5.00

Copyright © 1993 by Academic Press, Inc.
All rights of reproduction in any form reserved.

are known, it is unlikely that humans will be able to do real-time path planning which is as good as these solutions. Hence, they are considered as lower bounds on what can be expected in real egress scenarios. If these lower bound or ideal values represent unsatisfactory outcomes, then decision makers should be made aware of the potential hazards and take appropriate remedial actions.

Work on dynamic programming for path planning in networks with dynamic cost functions has its roots in the paper of Cooke and Halsey [5]. Travel times on links were considered as general functions of time, and a grid of discrete values of time was superimposed. The functions were to be evaluated at the arrival time at a node. This type of evaluation is now known as the "frozen link" model of cost evaluation for dynamic networks [14]. This dynamic cost approach considers only travel times, and indeed, cannot comprehend other types of costs such as distances, etc. It will be shown in this paper how to modify the dynamic programming approach to include other types of cost functions in a multiple objective dynamic programming approach to path planning in networks.

Other related research on dynamic programming considering time dependent parameters includes Sebastian [16] and Li and Haimes [13] and the recent analysis of Orda and Rom [14]. Sebastian and Li and Haimes are concerned with discrete dynamical systems and their control under time varying constraints and parameters. The state equations they consider are replaced by the network structure in path planning. Hence, the results do not transfer easily to path planning in networks. Orda and Rom [14] are more concerned with computational complexity of path planning and other alternative algorithms under time varying costs. Their focus is restricted to the construction of a single path from one origin to a single destination. Hence the complexity results they obtain must be scaled up accordingly when computing all paths under the time varying assumption. Two separate objectives are considered, time and distance, but only one of these objective functions is present in any one model. Bertsekas and Gallager [2] ask a question (in the exercises) about how to handle a path planning problem in which one cost on one link increases at a given time. They suggest that there is a simple modification to an existing algorithm to handle such a situation. Very recent work by Kaufman and Smith [11] and Evans *et al.* [8] suggest that there are economical computational strategies available in the case of specially structured dynamic programming problems. For path planning, there are similar observations about structure and how to compute more effectively. Algorithms to effectively compute time dependent path planning solutions with multiple objectives will be introduced in this paper, under some assumptions about the time varying cost structure which are motivated by the applications. These assumptions are similar to those made by Kaufman and Smith for

the single objective case. It is interesting to note that these assumptions arose independently and simultaneously from two separate applications, transportation planning in congested road networks and fire hazard analysis. It is quite likely that the assumptions are widely applicable in many diverse settings.

A new development in dynamic programming theory was presented by Ibaraki [10], who introduced the concept of congener relaxation and related technique that he called successive sublimation dynamic programming. The original dynamic programming problem is relaxed and subsequent sublimateations of the relaxation are examined until one of them is congener to the original problem, which guarantees availability of its solution. Relationships of this technique to our results are yet unknown.

Multiple objective dynamic programming developed concurrently to dynamic programming with time varying costs, but more research exists on the latter topic. Initially, Brown and Strauch [3] considered multiple objective functions with a latticial order. Daellenbach and De Kluyver [7], seemed to be unaware of Brown and Strauch, and they showed how to compute the set of nondominated paths in a network under the ordering of the nonnegative orthant. The theory behind the computation was not included in paper [7]. Multiple objective analysis of discrete dynamical systems is considered by Perevozchikov [15], while papers [6, 9] consider vector routing problems in networks with constant costs on the links. Carraway and Morin [4] consider a generalization of dynamic programming, but their work does not handle the time dependent cost structure of interest. Finally, Li and Haimes [13] consider the discrete dynamical systems control problem with multiple objectives and time dependent constraints. Again, the network structure precludes such work from application to the path planning problem of this paper.

The above discussion of progress to date in dynamic programming with time varying cost structure and multiple objective functions demonstrates that many related papers exist, but that evidently none will handle the object of interest: path planning in networks which comprehends multiple time varying objective functions. From this brief introduction to the problem and the existing literature, we now proceed to the formal framework for our research.

The mathematical framework we consider is a general network, not assumed to be acyclic. It consists of a set of nodes $\{1, 2, \dots, N\}$ and a set of links which indicate connections between nodes, i.e., $\{(i_1, i_2), (i_3, i_4), \dots\}$. The links' directions are indicated by the order of the indices. So, $(3, 4)$ is the link from node 3 to node 4, while $(4, 3)$ is the link from node 4 to node 3. A path from node i_0 to node i_p is a sequence of links $P = \{(i_0, i_1), (i_1, i_2), \dots, (i_{p-1}, i_p)\}$ in which the initial node of each arc is the same as the terminal node of the preceding arc in the sequence and i_0, \dots, i_p

are all distinct nodes. Let Π be the set of all feasible paths in the network which have the form

$$\{(i_1, i_2), (i_2, i_3), (i_3, i_4), \dots, (i_{S-1}, i_S)\}, \quad \text{where } 1 \leq i_1, i_S \leq N.$$

Each link carries one or more attributes (i.e., time to travel, distance to travel, etc.) which we think of as cost functions. The cost (vector) of a link (i, j) applies to all paths which include link (i, j) . The cost functions $(c_{ij}: R^+ \rightarrow R^{m+})$ are assumed to be positive vector valued functions of time, and are not assumed to be continuous. Let $[c_{ij}(t)]_1$ be the time to travel from node i to node j , given that travel starts at time t . The cost to traverse a path p in Π is defined to be

$$[c(p)] = \sum_{(i,j) \in p} [c_{ij}(t)].$$

A path in Π is a nondominated path if there is no other path p' in Π with $[c(p')] \leq [c(p)]$ and $[c(p')]_r < [c(p)]_r$ for some $r \in \{1, \dots, m\}$, where symbol \leq in the vector inequality denotes $[c(p')]_r \leq [c(p)]_r$ for $r = 1, \dots, m$.

The organization of this paper is as follows. Section 2 includes all the theoretical results developed in the paper. In two subsections, two different approaches and algorithms to solving time dependent multiple criteria routing problems are presented. The first applies backward dynamic programming and solves the routing problem generating all nondominated paths leading from every node in the network to the destination node. The second approach solves the routing problem for the set of feasible paths which lead from the origin node to all other nodes in the network. The analysis shows how adoption of forward dynamic programming leads to a new version of the principle of optimality that can deal with a general class of dynamic multiple objective networks. The relationship between the forward and backward case is also explored. Examples that apply the two algorithms are included in Section 3.

2. MULTIPLE CRITERIA TIME DEPENDENT DYNAMIC PROGRAMMING ALGORITHMS

2.1. Backward Dynamic Programming Case

In this section we present an algorithm for computing the set of all nondominated paths in the network, as introduced in the previous section. The algorithm is based on Bellman's principle of optimality [1]. We generalize the approach to finding the shortest (fastest) route through a network developed by Cooke and Halsey [5] to the multiple criteria case.

Assume a discrete time scale $S_T = \{t_0, t_0 + 1, t_0 + 2, \dots, t_0 + T\}$, $t_0 > 0$. Accordingly, assume that all the functions $[c_{ij}(t)]_k$, $i, j = 1, 2, \dots, N$, $i \neq j$, $k = 1, 2, \dots, m$, have positive values for $t \in S_T$. $[c_{ij}(t)]_1$ is the travel time from node i to node j , given that the arrival time at node i is t and it is assumed to be a positive integer. The number T is taken to be an upper bound on the total travel time required to go from any node in the network to node N . For example, $\max_{1 \leq i \leq N} \{[c_{iN}(t_0)]_1\}$ is an upper bound. If the nodes are not all connected to node N , then the number T is only implicitly defined. Since there are only a finite number of paths with finite cost vectors which are feasible, the value of T exists.

Now we introduce the following sets that are defined for $t \in S_T$ and $i = 1, 2, \dots, N - 1$:

$\{E_i(t)\}$ —the set of all paths in the network which leave node i at time $t \in S_T$ and reach node N ;

$\{\text{Eff}(E_i(t))\}$ —the set of all nondominated paths which leave node i at time $t \in S_T$ and reach node N ;

$\{[F_i(t)]\}$ —the set of vector costs of all nondominated paths in $\{\text{Eff}(E_i(t))\}$;

$\{E_i(t)^{(k)}\}$ —the set of all paths of at most k links, leaving node i at time t and reaching node N at or before time $t_0 + T$, $k = 2, 3, \dots$;

$\{[F_i(t)^{(k)}]\}$ —the set of vector costs of all nondominated paths in $\{E_i(t)^{(k+1)}\}$. Obviously, $\{\text{Eff}(E_i(t))\} \subseteq \{E_i(t)\}$.

According to the frozen link model, we also assume that upon the arrival time at node i the vector cost of link (i, j) for all j such that (i, j) exists, is an easily computed constant. Thus, the arrival time at node j is $t + [c_{ij}(t)]_1$.

Let $[\infty]$ and $\{\infty\}$ denote the vector and the set of m -component vectors such that each component is equal to infinity, respectively. Let $\{0\}$ denote a set containing the zero vector.

Bellman's principle of optimality [1] has been applied to multiple objective dynamic programming [3, 7, 9]. For completeness we present the principle of optimality in the form relevant to the path planning problem in this paper. Following Cooke and Halsey [5] and Kaufman and Smith [11], we consider the expanded static multiple objective network in which the state incorporates the current location in the network (node) at the current time.

THEOREM 1. (Principle of Optimality for Static Multiple Objective Networks). *A nondominated path p , leaving node i at time $t \in S_T$ and reaching node N at or before time $t_0 + T$, has the property that for each node*

j lying on this path, a subpath p_1 , that leaves node j at time $t_j \in S_T$, $t_j > t$, and arrives at node N at or before time $t_0 + T$, is nondominated.

Proof. By contradiction. Assume to the contrary that p_1 is dominated (not nondominated), i.e., there exists path p^* leaving node j at time t_j and arriving at node N at or before time $t_0 + T$, such that

$$[c(p^*)] \leq [c(p_1)]$$

and

$$[c(p^*)]_r < [c(p_1)]_r \quad \text{for some } r \in \{1, \dots, m\}. \quad (1)$$

Let p_2 be a subpath that leaves node i at time t and arrives at node j at time t_j . Thus we have two paths from node i to node N such that their total cost is, respectively,

$$[c(p_1)] + [c(p_2)]$$

and

$$[c(p^*)] + [c(p_2)].$$

Applying (1) we get

$$[c(p^*)] + [c(p_2)] \leq [c(p_1)] + [c(p_2)],$$

which implies that path $p = (p_2, p_1)$ consisting of subpath p_2 and p_1 is dominated.

By the principle of optimality [1] and Theorem 1, we establish that for $t \in S_T$,

$$\begin{aligned} \{[F_i(t)]\} &= VMIN\{[c_{ij}(t)] + \{[F_j(t + [c_{ij}(t)]_1)]\}\}, \\ i &= 1, 2, \dots, N-1, \end{aligned} \quad (2)$$

$$\{[F_N(t)]\} = \{0\},$$

where operation $VMIN$ computes vector costs of nondominated paths in the set being the algebraic sum of the cost vector $[c_{ij}(t)]$ and the set of vector costs of all nondominated paths that leave node j at time $t + [c_{ij}(t)]_1$. Computing all nondominated paths in $\{Eff(E_i(t))\}$ requires applying an iteration scheme on the system of equations above.

We present now Algorithm One which includes the iterative procedure and finds $\{Eff(E_i(t))\}$, $i = 1, 2, \dots, N-1$, in a finite number of steps.

ALGORITHM ONE.

Step 1. Establish a limited grid of discrete values of time $S_T = \{t_0, t_0 + 1, t_0 + 2, \dots, t_0 + T\}$. (Choice of T is discussed above). For $t \in S_T$ and $i, j = 1, 2, \dots, N, i \neq j$, compute $[c_{ij}(t)]$.

Step 2. Modify the vectors $[c_{ij}(t)]$, $t \in S_T$ as follows:

$$[c_{ij}(t)]' = \begin{cases} [c_{ij}(t)] & \text{if } t + [c_{ij}(t)]_i \leq t_0 + T \\ [\infty] & \text{if } t + [c_{ij}(t)]_i > t_0 + T \end{cases} \quad i, j = 1, 2, \dots, N, i \neq j. \tag{3}$$

Step 3. Construct an "initial guess" array $[\{[F_i(t)^{(0)}]\}]$, $i = 1, 2, \dots, N$, $t \in S_T$, where $\{[F_N(t)^{(0)}]\} = \{0\}$, and $\{[F_i(t)^{(0)}]\} = [c_{iN}(t)]'$ for $i = 1, 2, \dots, N - 1$.

Step 4. Calculate the arrays $[\{[F_i(t)^{(k)}]\}]$, $i = 1, 2, \dots, N$, $t \in S_T$, for $k = 1, 2, 3, \dots$ as follows:

$$\begin{aligned} \{[F_i(t)^{(k)}]\} &= VMIN\{[c_{ij}(t)]' + \{[F_j(t + [c_{ij}(t)]'_i)^{(k-1)}]\}\}, \\ &\quad i = 1, 2, \dots, N - 1, \tag{4} \\ \{[F_N(t)^{(k)}]\} &= \{0\}. \end{aligned}$$

The *VMIN* operation in the equation above will lead to $\{\infty\}$ if $[c_{ij}(t)]' = [\infty]$ for all j or if $t + [c_{ij}(t)]'_i \notin S_T$. Otherwise, if both $[c_{ij}(t)]'$ and $\{[F_j(t + [c_{ij}(t)]'_i)^{(k-1)}]\}$ are finite for some j , compute $[c_{ij}(t)]'$, extract $\{[F_j(t + [c_{ij}(t)]'_i)^{(k-1)}]\}$ from the array $[\{[F_i(t)^{(k)}]\}]$ and perform the *VMIN* operation (over j) of their algebraic sum.

Step 5. The sequence of sets $\{[F_i(t_0)^{(k)}]\}$, $k = 1, 2, \dots$ converges to $\{[F_i(t_0)]\}$. The set $\{Eff(E_i(t_0))\}$ is obtained by keeping track of the indices of paths' links that contribute to $\{[F_i(t_0)^{(k)}]\}$.

THEOREM 2. *The method of Algorithm One is well-defined.*

Proof. We start (step $k=0$) with all paths and corresponding costs from node i to node N set equal to the single link costs connecting each node to N . That is, we start with nondominated paths of one link each. Each set $\{[F_i(t)^{(0)}]\} \neq \emptyset$, and contains exactly one cost vector. As k increases, the set $\{[F_i(t)^{(k)}]\}$ accumulates the nondominated cost vectors, while always dropping dominated cost vectors. Hence, $\{[F_i(t)^{(k)}]\}$ remains nonempty for any k . Therefore the *VMIN* operation may be applied for any k .

THEOREM 3. *The iterative step of Eq. (4) computes the set of all vector costs corresponding to all nondominated paths with at most k links connecting node i to node N with departure time t .*

Proof. The proof is by induction on k . Assume that $k = 1$ and define for $t \in S_T$:

$$\{[F_i(t)^{(1)}]\} = \begin{cases} \text{VMIN over the vector costs of paths in } \{E_i(t)^{(2)}\}, \\ \quad \text{if } \{E_i(t)^{(2)}\} \neq \emptyset, \\ \{\infty\} \quad \text{if } \{E_i(t)^{(2)}\} = \emptyset, \end{cases}$$

$$i = 1, 2, \dots, N-1,$$

$$\{[F_N(t)^{(1)}]\} = \{0\}.$$

If $\{E_i(t)^{(2)}\} = \emptyset$, then the one-link path from node i to node N , leaving at time t reaches node N after time $t_0 + T$, so that $[c_{ij}(t)]' = [\infty]$. There is also no two-link path from node i through node j to node N that reaches node N by time $t_0 + T$, so that no one-link path leaving node j at time $t + [c_{ij}(t)]'_1$ reaches node N by time $t_0 + T$. Therefore either $[c_{ij}(t)]' = [\infty]$ or $\{[F_j(t + [c_{ij}(t)]'_1)^{(0)}]\} = \{\infty\}$ and $\{[F_i(t)^{(1)}]\} = \{\infty\}$.

If $\{E_i(t)^{(2)}\} \neq \emptyset$, then there exists at least one nondominated path of one or two links leaving node i at time t and reaching node N by time $t_0 + T$. A one-link nondominated path may have been obtained from initialization and still be nondominated. A two-link nondominated path leads from node i to node j , and then follows the one-link path from node j to node N with the arrival time at node j equal to $t + [c_{ij}(t)]'_1 = t + [c_{ij}(t)]_1$.

Now assume that the iterative step is valid for $k > 1$ and we will show its validity for $k + 1$. Again define for $t \in S_T$:

$$\{[F_i(t)^{(k+1)}]\} = \begin{cases} \text{VMIN over the vector costs of paths in } \{E_i(t)^{(k+2)}\}, \\ \quad \text{if } \{E_i(t)^{(k+2)}\} \neq \emptyset, \\ \{\infty\} \quad \text{if } \{E_i(t)^{(k+2)}\} = \emptyset, \end{cases}$$

$$i = 1, 2, \dots, N-1,$$

$$\{[F_N(t)^{(k+1)}]\} = \{0\}.$$

If $\{E_i(t)^{(k+2)}\} = \emptyset$, then there is no path of at most $k + 2$ links that leaves node i at time t and reaches node N by time $t_0 + T$. Therefore $\{[F_i(t)^{(k+1)}]\} = \{\infty\}$.

If $\{E_i(t)^{(k+2)}\} \neq \emptyset$, then there exists at least one nondominated path of at most $k + 2$ links leaving node i at time t and reaching node N by time $t_0 + T$. A nondominated path of at most $k + 1$ links leaving node i at time

t and reaching node N by time $t_0 + T$ may have been obtained in the k th or an earlier iteration of the algorithm. In the $k + 1$ iteration one may obtain a nondominated path of at most $k + 2$ links leading from node i to node j , and then following a path of at most $k + 1$ links from node j to node N , that has been found in the k th or an earlier iteration on the algorithm.

THEOREM 4. *After a finite number of steps Algorithm One generates all nondominated paths that leave node i , $i = 1, 2, \dots, N - 1$, at time t_0 and reach node N .*

Proof. By contradiction. Assume that the method generated all nondominated paths, that leave node i at time t_0 and reach node N , except one. Then either the set of nondominated paths is incomplete or it includes at least one path that is dominated by the missing path. The former implies that the algorithm missed one nondominated path which contradicts the fact that each node of the network was visited and all links leading to it were examined. The latter indicates that the procedure did not identify the true status of a path, namely dominated. Hence arises a contradiction with performing the *VMIN* operation at node i , $i = 1, 2, \dots, N - 1$, of the network and discovering all nondominated paths leaving this node and reaching node N .

2.2. Forward Dynamic Programming Case

In the dynamic programming literature two problem formulations are commonly considered. For the multiple objective network there are: (1) find all nondominated paths from every node in the network to the destination node, or (2) from the origin node find all nondominated paths to every node in the network. While the former formulation gave rise to the development of Algorithm One, the latter will be considered in this section. The theoretical background of the multiple criteria dynamic network analysis will now be adapted for the second formulation. We relate the second formulation to the first as follows: the destination node is still the main focus. The forward solution is to be obtained with each other node in the network as the origin, and nondominated paths which reach the destination node are computed. The extra computations required may be compensated by other structural simplifications of the forward approach.

In this forward approach, feasible paths are those which start at the origin node. Assume that time t is a continuous variable, that is, $t \geq 0$, and hence allow $[c_{ij}(t)]_1$ to take any positive value. We normalize so that the departure time from the origin node is $t = 0$. Finally an assumption is introduced which allows the formulation of the principle of optimality for dynamic multiple objective networks.

Assumption 1. For any link (i, j) in the network and all $t_1, t_2 \geq 0$, if $t_1 \leq t_2$, then

- (a) $t_1 + [c_{ij}(t_1)]_1 \leq t_2 + [c_{ij}(t_2)]_1$, and
 (b) $[c_{ij}(t_1)]_r \leq [c_{ij}(t_2)]_r$ for all $r \in \{2, \dots, m\}$.

We also introduce the following sets defined for $j=2, 3, \dots, N$, and vectors defined for $t > 0$ and $j=2, 3, \dots, N$:

$\{D_j\}$ —the set of all paths in the network which leave the origin node at time $t=0$ and lead to node j ;

$\{\text{Eff}(D_j)\}$ —the set of all nondominated paths which leave the origin node at time $t=0$ and lead to node j ;

$\{D_j^{(k)}\}$ —the set of all paths of at most k links leaving the origin node at time $t=0$ and leading to node j , $k=2, 3, \dots$;

$[G_j^u(t^u)^{(k)}]$ —the vector cost of the nondominated path u in $\{D_j^{(k+1)}\}$, where t^u is the arrival time of this path at node j ;

$\{[G_j^{(k)}]\} = \{[G_j(t^u)^{(k)}], u=1, \dots, N_j\}$ —the set of vectors costs of all nondominated paths in $\{D_j^{(k+1)}\}$, where N_j is the number of the nondominated paths;

$[G_j^u(t^u)]$ —the vector cost of the nondominated path u leaving the origin node at time $t=0$ and arriving at node j at time t^u ;

$\{[G_j]\} = \{[G_j(t^u)], u=1, \dots, N_j\}$ —the set of vector costs of all nondominated paths in $\{\text{Eff}(D_j)\}$, where N_j is the number of the nondominated paths.

Obviously, $\{\text{Eff}(D_j)\} \subseteq \{D_j\}$.

THEOREM 5 (Principle of Optimality for Dynamic Multiple Objective Networks). *Under Assumption 1(a) and (b), a nondominated path p , that leaves the origin node at time $t=0$ and arrives at node j at time t_j , has the property that for each node i lying on this path, a subpath p_1 , that leaves the origin node at time $t=0$ and arrives at node i at time t_i , $t_i \leq t_j$, is nondominated.*

Proof. By contradiction. Assume to the contrary that p_1 is dominated (not nondominated), i.e., there exists path p^* that leaves the origin node at time t and arrives at node i at time $t_i^* \leq t_i$ such that

$$[c(p^*)] \leq [c(p_1)],$$

and

(5)

$$[c(p^*)]_r < [c(p_1)]_r \quad \text{for some } r \in \{1, \dots, m\}.$$

Both paths, p_1 and p^* , leave the origin node at the same time $t=0$ and lead to node i . Hence there are two paths from node i to node j such that one of them leaves node i at time t_i and arrives at node j at time t_j , and the other leaves node i at time t_i^* and arrives at node j at time t_j^* . Let us call those paths p_2 and p_2^* , respectively. Thus we have two paths from the origin node to node j such that their total cost is respectively:

$$[c(p_1)] + [c(p_2)] = [c(p_1)] + \sum_{\substack{k=i \\ (k,l) \in p_2}}^{l=j} [c_{kl}(t_k)] \tag{6}$$

and

$$[c(p^*)] + [c(p_2^*)] = [c(p^*)] + \sum_{\substack{k=i \\ (k,l) \in p_2^*}}^{l=j} [c_{kl}(t_k^*)], \tag{7}$$

where t_k and t_k^* are the arrival times at node k ($k = i, \dots, l = j$) on paths p_2 and p_2^* , respectively.

We also have that $t_i^* \leq t_i$, then by Assumption 1(a)

$$t_i^* + [c_{is}(t_i^*)]_1 \leq t_i + [c_{is}(t_i)]_1$$

for every link (i, s) in the network. Applying Assumption 1(a) on every link (k, l) in path p_2 and p_2^* , and substituting arrival times at each node, we reach node j and get

$$t_i^* + \sum_{\substack{k=i \\ (k,l) \in p_2^*}}^{l=j} [c_{kl}(t_k^*)]_1 \leq t_i + \sum_{\substack{k=i \\ (k,l) \in p_2}}^{l=j} [c_{kl}(t_k)]_1,$$

which by definition means

$$t_j^* \leq t_j. \tag{8}$$

Applying Assumption 1(b) and summing over all links on p_2 and p_2^* we have

$$\sum_{\substack{k=i \\ (k,l) \in p_2^*}}^{l=j} [c_{kl}(t_k^*)]_r \leq \sum_{\substack{k=i \\ (k,l) \in p_2}}^{l=j} [c_{kl}(t_k)]_r, \quad r \in \{2, \dots, m\}. \tag{9}$$

Note that inequalities (5), (8), and (9) lead to

$$[c(p^*)] + [c(p_2^*)] \leq [c(p_1)] + [c(p_2)],$$

which implies that path $p = (p_1, p_2)$ consisting of subpaths p_1 and p_2 is dominated.

By the principle of optimality [1] and Theorem 5, we establish that for $t' > 0$ and $t'' > 0$:

$$\{[G_j'(t')], \ell = 1, \dots, N_j\} = VMIN\{[G_i^n(t'')] + [c_{ij}(t'')], \quad n = 1, \dots, N_i\},$$

$$j = 2, 3, \dots, N, \quad (10)$$

$$\{[G_j'(t')], \ell = 1\} = \{0\},$$

where operation *VMIN* computes vector costs of nondominated paths in the set for which each element is a vector sum of the vector cost of the nondominated path n leaving the origin node at time 0 and arriving at node i at time t'' , and the cost vector of link (i, j) with the arrival time t'' at node i . Computing all nondominated paths in $\{\text{Eff}(D_j)\}$ requires again applying an iteration scheme on the system of equations above.

We now present Algorithm Two which includes the iterative procedure and finds $\{\text{Eff}(D_j)\}$, $j = 2, 3, \dots, N$, in a finite number of steps. (Assume without loss of generality that node 1 is the origin node.)

ALGORITHM TWO.

Step 1. Construct an "initial guess" vector $\{[G_j^{(0)}]\}$, $j = 1, 2, \dots, N$, where

$$\{[G_1^{(0)}]\} = \{0\},$$

$$\{[G_j^{(0)}]\} = [c_{1j}(0)], \quad j = 2, 3, \dots, N.$$

Step 2. Calculate the vectors $\{[G_j^{(k)}]\}$, $j = 1, 2, \dots, N$, for $k = 1, 2, 3, \dots$, as follows:

$$\{[G_j'(t')^{(k)}], \ell = 1, \dots, N_j\} = VMIN\{[G_i^n(t'')^{(k-1)}]$$

$$+ [c_{ij}(t'')], n = 1, \dots, N_i\}, \quad j = 2, 3, \dots, N,$$

$$\{[G_j'(t')^{(k)}], \ell = 1\} = \{0\}. \quad (11)$$

The *VMIN* operation in the equation above will lead to $\{\infty\}$ if $[c_{ij}(t'')] = [\infty]$ for all nondominated paths leading to node i ($n = 1, \dots, N_i$). Otherwise, if both $[c_{ij}(t'')]$ and $[G_i^n(t'')^{(k-1)}]$ are finite for some nondominated path n , then compute their vectors sum for each such path, and perform the *VMIN* operation (over i) on the set just obtained.

Step 3. The sequence of sets $\{[G_j^{(k)}]\}$, $k = 1, 2, \dots$, converges to $\{[G_j]\}$. The set $\{\text{Eff}(D_j)\}$ is obtained by keeping track of paths' links that contribute to $\{[G_j^{(k)}]\}$.

The proofs of the following theorems are similar to the proofs of Theorems 2, 3, and 4, and are thus omitted.

THEOREM 6. *The method of Algorithm Two is well defined.*

THEOREM 7. *The iterative step of Eq. (11) computes the set of all vectors costs corresponding to all nondominated paths of at most k links connecting node 1 and node j with departure time $t=0$.*

THEOREM 8. *After a finite number of steps Algorithm Two generates all nondominated paths that leave node 1 at time $t=0$ and reach all other nodes j , for $j=2, 3, \dots, N$.*

COROLLARY 1. *If $[c_{ij}(t)]_r$, $i, j=1, 2, \dots, N$, and $i \neq j$, $r=1, 2, \dots, m$, is a continuous monotone increasing function on $[0, \infty)$, then Algorithm Two finds all nondominated paths from the origin node to any other node.*

Proof. A continuous monotone increasing function satisfies Assumption 1(a) and (b) and thus the Principle of Optimality for Dynamic Multiple Objective Networks holds, and by Theorem 8 all nondominated paths from the origin node to any other node are generated.

COROLLARY 2. *If $[c_{ij}(t)]_r$, $i, j=1, 2, \dots, N$, and $i \neq j$, $r=1, 2, \dots, m$, is a monotone increasing step function on $[0, \infty)$, then Algorithm Two finds all nondominated paths from the origin node to any other node.*

Proof. Proof follows the proof of Corollary 1 since a monotone increasing step function satisfies Assumption 1(a) and (b).

COROLLARY 3. *If $[c_{ij}(t)]_r$, $i, j=1, 2, \dots, N$, and $i \neq j$, $r=1, 2, \dots, m$, is a continuous monotone increasing function on $[0, \infty)$, then all nondominated paths from node i , $i=1, 2, \dots, N-1$, starting at time $t=0$ and leading to the destination node N may be computed with at most $N-1$ applications of Algorithm Two.*

COROLLARY 4. *If $[c_{ij}(t)]_r$, $i, j=1, 2, \dots, N$, and $i \neq j$, $r=1, 2, \dots, m$, is a monotone increasing step function on $[0, \infty)$, then all nondominated paths from node i , $i=1, 2, \dots, N-1$, starting at time $t=0$ and leading to the destination node N may be computed with at most $N-1$ applications of Algorithm Two.*

As it was mentioned above, computing all nondominated paths from any node to the destination node is the main interest. Corollaries 3 and 4, that result immediately from Corollaries 1 and 2 and thus are presented without proofs, show that the forward approach can be applied to solving that problem. Although such a method becomes more complex computationally

(at most $(N-1)$ Algorithm Two solutions), it is competitive with Algorithm One, which requires storing and computing a large amount of data for the expanded static multiple objective network.

3. EXAMPLES

The algorithms presented in the previous section are now applied to solve two dynamic routing problems. The notation in each of the subsections below agrees with symbols previously used in Sections 2.1 and 2.2, respectively.

3.1. Backward Dynamic Programming Case

We will use Algorithm One to solve a dynamic routing problem with two criteria ($m=2$) for the network depicted in Fig. 1. The example is related to a somewhat simpler example in Kaufman and Smith [11]. Remarks there indicate that a naive approach will fail on their example. Similarly, one needs the expanded static network to solve the network.

A grid of discrete values of time $S_{19} = \{1, 2, \dots, 20\}$ for $t_0 = 1$ is established, and vectors $[c_{ij}(t)]$ for $t \in S_{19}$ are modified according to Step 2 of the algorithm, which results in the expanded static network. An initial guess array $[\{[F_i(t)^{(0)}]\}]$, $i = 1, \dots, 4$, $t \in S_{19}$, is constructed, and the arrays $[\{[F_i(t)^{(1)}]\}]$ and $[\{[F_i(t)^{(2)}]\}]$ are calculated as Step 4 of the algorithm indicates. Figure 2 shows the initial array and the two subsequent arrays calculated in Step 4 of the algorithm (each array has 20 rows and 4 columns that are shown in Fig. 2 in a truncated form). $[\{[F_i(t_0)^{(2)}]\}]$ is given by the first row of array $[\{[F_i(t)^{(2)}]\}]$ and contains vector costs of all nondominated paths that leave node i , ($i = 1, \dots, 4$), at time $t_0 = 1$ and reach node 4.

The sets $\{\text{Eff}(E_i(t_0))\}$ of all nondominated paths that leave node i , $i = 1, 2, 3$, at time $t_0 = 1$ are given as

$$\{(1, 2), (2, 3), (3, 4)\},$$

$$\{(2, 3), (3, 4)\},$$

$$\{(3, 4)\},$$

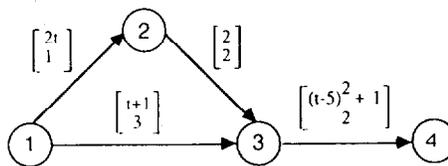


FIG. 1. Two-criteria dynamic network.

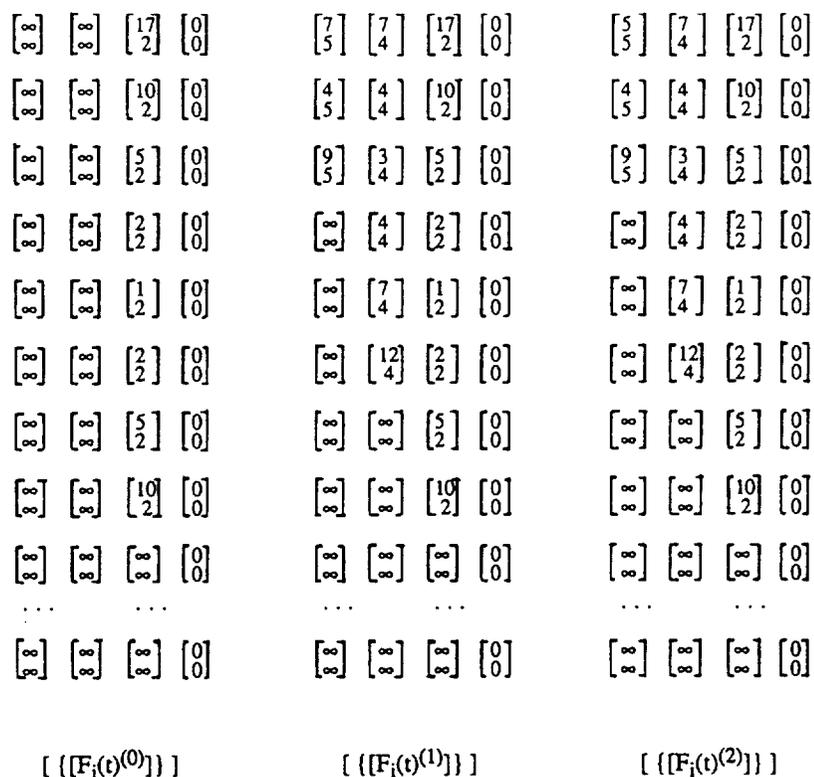


FIG. 2. Algorithm One, Step 4 calculations.

and are obtained by keeping track of the indices of paths' links that contribute to $\{ \{ [F_i(t_0)^{(2)}] \} \}$.

Observe that because the cost functions are not all monotone increasing functions of time, it is possible to have total cost behave in a non-monotonic way. In such a network, one may also experience "passing" by which one traveling unit overtakes another on a link.

3.2. Forward Dynamic Programming Case

We solve a two-criteria dynamic routing problem presented by the network in Fig. 3 and apply Algorithm Two. Cost functions here satisfy Assumption 1. Therefore, a "no-passing" convention is in effect. For comparison, the cost functions are chosen as a combination of constant functions and monotone increasing step functions.

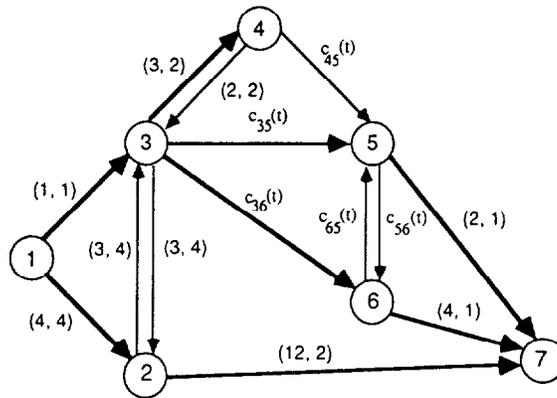


FIG. 3. Two-criteria dynamic network with step cost functions and its nondominated paths.

The step cost functions are given as

$$c_{35}(t) = \begin{cases} \begin{bmatrix} 6 \\ 8 \end{bmatrix} & t < 3 \\ \begin{bmatrix} 10 \\ 12 \end{bmatrix} & t \geq 3 \end{cases}, \quad c_{45}(t) = \begin{cases} \begin{bmatrix} 4 \\ 4 \end{bmatrix} & t < 3 \\ \begin{bmatrix} 10 \\ 10 \end{bmatrix} & t \geq 3 \end{cases}, \quad c_{65}(t) = \begin{cases} \begin{bmatrix} 8 \\ 10 \end{bmatrix} & t < 3 \\ \begin{bmatrix} 10 \\ 12 \end{bmatrix} & t \geq 3 \end{cases}, \\ c_{36}(t) = \begin{cases} \begin{bmatrix} 5 \\ 7 \end{bmatrix} & t < 5 \\ \begin{bmatrix} 10 \\ 12 \end{bmatrix} & t \geq 5 \end{cases}, \quad c_{56}(t) = \begin{cases} \begin{bmatrix} 5 \\ 10 \end{bmatrix} & t < 5 \\ \begin{bmatrix} 12 \\ 12 \end{bmatrix} & t \geq 5 \end{cases}.$$

All the other links of the network have constant vector costs, as Fig. 3 shows. We start with an initial guess vector $[\{G_j^{(0)}\}]$, $j = 1, 2, \dots, 7$, and calculate vectors $[\{G_j^{(k)}\}]$, $j = 1, 2, \dots, 7$, according to Step 2 of the algorithm. The vectors $[\{G_j^{(k)}\}]$, $j = 1, 2, \dots, 7$, for $k = 0, 1, \dots, 4$ are shown in Fig. 4.

The sequence of sets $\{[G_j^{(k)}]\}$ converges to $\{[G_j]\}$ in the second iteration of the algorithm. The sets $\{\text{Eff}(D_j)\}$ for $j = 2, 3, \dots, 7$, are obtained by keeping track of paths' links that contribute to $\{[G_j^{(k)}]\}$ and include the following paths:

- $\{(1, 2)\}$,
- $\{(1, 3)\}$,
- $\{(1, 3), (3, 4)\}$,
- $\{(1, 3), (3, 5)\}$,
- $\{(1, 3), (3, 6)\}$,
- $\{(1, 2), (2, 7)\}$, $\{(1, 3), (3, 5), (5, 7)\}$, $\{(1, 3), (3, 6), (6, 7)\}$.

	k=0	k=1	k=2	k=3
$\{G_1^{(k)}\}$	$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$
$\{G_2^{(k)}\}$	$\begin{bmatrix} 4 \\ 4 \end{bmatrix}$	$\begin{bmatrix} 4 \\ 4 \end{bmatrix}$	$\begin{bmatrix} 4 \\ 4 \end{bmatrix}$	$\begin{bmatrix} 4 \\ 4 \end{bmatrix}$
$\{G_3^{(k)}\}$	$\begin{bmatrix} 1 \\ 1 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 1 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 1 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 1 \end{bmatrix}$
$\{G_4^{(k)}\}$	$\begin{bmatrix} \infty \\ \infty \end{bmatrix}$	$\begin{bmatrix} 4 \\ 3 \end{bmatrix}$	$\begin{bmatrix} 4 \\ 3 \end{bmatrix}$	$\begin{bmatrix} 4 \\ 3 \end{bmatrix}$
$\{G_5^{(k)}\}$	$\begin{bmatrix} \infty \\ \infty \end{bmatrix}$	$\begin{bmatrix} 7 \\ 9 \end{bmatrix}$	$\begin{bmatrix} 7 \\ 9 \end{bmatrix}$	$\begin{bmatrix} 7 \\ 9 \end{bmatrix}$
$\{G_6^{(k)}\}$	$\begin{bmatrix} \infty \\ \infty \end{bmatrix}$	$\begin{bmatrix} 6 \\ 8 \end{bmatrix}$	$\begin{bmatrix} 6 \\ 8 \end{bmatrix}$	$\begin{bmatrix} 6 \\ 8 \end{bmatrix}$
$\{G_7^{(k)}\}$	$\begin{bmatrix} \infty \\ \infty \end{bmatrix}$	$\begin{bmatrix} 16 \\ 6 \end{bmatrix}$	$\begin{bmatrix} 16 \\ 6, 9, 10 \\ 9 \end{bmatrix}$	$\begin{bmatrix} 16 \\ 6, 9, 10 \\ 9 \end{bmatrix}$

FIG. 4. Algorithm Two, Step 2 calculations.

Note that this calculation produces the set of nondominated paths from node 1 to node 7 (destination node) as well as to all other nodes. A similar computation is required for nodes 2, 3, 4, 5, and 6 to compute all nondominated paths from these nodes to the destination node. Applying Algorithm Two for each of these nodes gives the following nondominated paths:

$$\begin{aligned}
 &\{(2, 7)\}, \\
 &\{(3, 2), (2, 7)\}, \quad \{(3, 6), (6, 7)\}, \quad \{(3, 5), (5, 7)\}, \\
 &\{(4, 5), (5, 7)\}, \\
 &\{(5, 7)\}, \\
 &\{(6, 7)\}.
 \end{aligned}$$

4. CONCLUSIONS

This paper presents for the first time a theoretical and algorithmic development for the problem of path planning in networks including multiple time dependent costs on the links. Throughout, the goal is to compute all nondominated paths in an efficient computational procedure. Applications abound in fire safety science, general transportation, and telecommunications.

Our study has produced two distinct algorithms, each one tailored to a particular class of cost functions. For general costs functions, we extend the work of Cooke and Halsey [5] to handle multiple objective functions. We observe that their paper solved only the minimum travel time path planning problem, so this paper seems to be the first to handle cost functions other than travel time. This algorithm should be used only when required (by cost functions which are not monotone increasing) because it has a greater overhead and computational cost than the other algorithms.

For monotone increasing cost functions satisfying one additional assumption, a forward dynamic programming algorithm which generalizes the paper of Kaufman and Smith [11] is presented. Such an algorithm seems more computationally effective than the one for general cost functions and it seems to be independent of the time horizon with respect to its computational complexity. It has the disadvantage that it must be applied to each origin node independently in order to get a complete set of nondominated paths.

ACKNOWLEDGMENTS

This research was supported in part by Grant 60NANB0D1023 from the Building and Fire Research Laboratory, National Institute of Standards and Technology, Gaithersburg, Maryland.

REFERENCES

1. R. BELLMAN, On a routing problem. *Quart. Appl. Math.* **16** (1958), 87-90.
2. D. BERTSEKAS AND R. GALLAGER, "Data Networks," Prentice-Hall, Englewood Cliffs, NJ, 1987.
3. T. A. BROWN AND R. E. STRAUCH, Dynamic programming in multiplicative lattices, *J. Math. Anal. Appl.* **12** (1965), 364-370.
4. R. L. CARRAWAY AND T. L. MORIN, Theory and applications of generalized dynamic programming: An overview, *Comput. Math. Appl.* **16**, No. 10/11 (1988), 779-788.
5. K. L. COOKE AND E. HALSEY, The shortest route through a network with time-dependent internodal transit times, *J. Math. Anal. Appl.* **14** (1966), 493-498.
6. H. W. CORLEY AND I. D. MOON, Shortest paths in networks with vector weights, *J. Optim. Theory Appl.* **46** (1985), 79-86.
7. H. G. DAELLENBACH AND C. A. DE KLUYVER, Note on multiple objective dynamic programming, *J. Oper. Res. Soc.* **31** (1980), 591-594.
8. J. R. EVANS, C. SAYDAM, AND M. MCKNEW, A note on solving the concave cost dynamic lot-sizing problem in almost linear time, *J. Oper. Management* **8**, No. 2 (1989), 159-167.
9. R. HARTLEY, Vector optimal routing by dynamic programming, in "Mathematics of Multiobjective Optimization" (P. Serafini, Ed.) pp. 215-224, Springer-Verlag, Vienna, 1985.
10. T. IBARAKI, Enumerative approaches to combinatorial optimization, Part II, *Ann. Oper. Res.* **11** (1987), 343-440.

11. D. E. KAUFMAN AND R. L. SMITH, "Minimum Travel Time Paths in Dynamic Networks with Application to Intelligent Vehicle-Highway Systems," University of Michigan, Transportation Research Institute, IVHS Technical Report-90-11, 1990.
12. M. M. KOSTREVA, M. M. WIECEK, AND T. GETACHEW, Optimization models in fire egress analysis for residential buildings, in "Proceedings of the Third International Symposium on Fire Safety Science, Edinburgh, United Kingdom, July 8-12, 1991," 805-814, (updated).
13. D. LI AND Y. Y. HAIMES, Multiobjective dynamic programming: The state of the art, *Control Theory Adv. Tech.* 5, No. 4 (1989), 471-483.
14. A. ORDA AND R. ROM, Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length, *J. A. C. M.* 37, No. 3 (1990), 607-625
15. A. G. PEREVOZCHIKOV, Dynamic programming in multistep vector optimization problems, *Eng. Cybernet.* 22, No. 3 (1984) 21-24.
16. H.-J. SEBASTIAN, Dynamic programming for problems with time-dependent parameters, *Differential Equations* 14, No. 2 (1978), 242-249.