Stochastic Shortest Path Problems with Recourse

George H. Polychronopoulos*

Operations Research Center, M.I.T., Cambridge, Massachusetts 02139

John N. Tsitsiklis†

Laboratory for Information and Decision Systems and the Operations Research Center, M.I.T., Cambridge, Massachusetts 02139

We consider shortest path problems defined on graphs with random arc costs. We assume that information on arc cost values is accumulated as the graph is being traversed. The objective is to devise a policy that leads from an origin to a destination node with minimal expected cost. We provide dynamic programming algorithms, estimates for their complexity, negative complexity results, and analysis of some possible heuristic algorithms. © *1996 John Wiley & Sons, Inc.*

1. INTRODUCTION

The deterministic shortest path problem has been studied extensively and has been found to be a very useful tool in a great variety of contexts. In this problem, one looks for a path joining two given nodes of a graph while minimizing the sum of the costs of the traversed arcs, assuming that these arc costs are known. On the other hand, there are many application areas in which it is natural to model arc costs as random variables. One example is the problem of vessel routing in the presence of uncertain weather conditions. Other examples concern the routing of automobiles in the presence of partially known and stochastically changing road congestion levels. Indeed, the recent interest in Intelligent Vehicle Highway Systems (IVHS) and in automated driver assistance technologies seems to lead naturally to variations of the shortest path problem that involve random arc costs.

The shortest path problem with random arc costs admits a few different formulations depending on the assumptions made regarding the time at which the realized values of the arc costs are learned (for a more detailed discussion, see [1]). For example:

(a) We may assume that the values of the arc costs are learned before the graph is traversed, in which case a shortest path (with respect to the realized values) should be followed. As the shortest path length is a random variable, it can become an object of study, as in [7, 9–11].

(b) At the other extreme, we may assume that the values of the arc costs are never learned or become known after a path is chosen. In this case, we should choose a path which is shortest with respect to the expected values of the arc lengths. [12, 13] deal with the more difficult problem of finding a path that minimizes the expectation of a utility function that depends nonlinearly on the arc lengths.

(c) In an intermediate formulation, which is the one adopted in this paper, the realization of the arc costs is learned progressively, as the graph is traversed. In particular, if learning occurs by direct observation, we may postulate that the cost of an arc is learned at the first time that an end-node of that arc is visited. This is the case, e.g., for a vehicle that learns the congestion level in a particular road

* Current address: Salomon Brothers International, 111 Buckingham Palace Rd., London SW1W OSB, United Kingdom.

† E-mail: jnt@athena.mit.edu.

CCC 0028-3045/96/020133-11

by getting to an intersection of that road. This framework is also suitable for modeling a robot that attempts to find a path to a destination through a random environment. For this class of problems, one should not be looking for a best path, but rather for a best *policy,* i.e., a rule for deciding where to go next given currently available information. Alternatively, the problem arises when we consider the set of decisions facing a vehicle that starts moving toward the destination along a certain path, with the recourse option of choosing a new path whenever new information is obtained.

[6] appears to be the first to have studied a model of this type, in a fairly restricted setting. Finally, [2] considered a model in which arcs can be active or inactive. One starts along a "ground path" which is followed until an inactive arc is encountered. At that point, an alternative recourse path is chosen and followed until the destination is reached. The model in the present paper is more general in several of respects (e.g., we allow for several recourse actions, each time that new information is obtained). The dynamic programming algorithms given in [2] and in this paper are based on similar ideas.

In all of the above cases, we have assumed that the value of the arc costs is random but does not change with time. In an alternative set of models, briefly discussed in the last section, we may assume that the arc costs can change randomly with time, as in [18].

We also note that there is some related literature in which information is obtained as the graph is traversed, and the arc costs are modeled as deterministic but unknown [3, 16]. This formulation is natural in unstructured environments for which minimal prior information is available. On the other hand, our probabilistic framework could be more suitable for environments that have some statistical regularity, e.g., street congestion levels.

Regarding the objectives of this paper, although this study has been motivated from certain practical contexts, we do not claim to be studying fully realistic or complete models of real-world situations. Rather, in view of the practical significance of routing problems under uncertainty, we are interested in understanding the assumptions necessary for such problems to be computationally tractable. Knowing which formulations result to tractable problems is often a key factor in deciding how to pose and approach practical problems.

The remainder of this paper is organized as follows: In Section 2, we specify the details of the models to be employed; in particular, two alternative models are proposed in which the arc costs are modeled as independent or dependent, respectively, random variables. Sections 3 and 4 present dynamic programming algorithms for these two models, together with upper bounds on their complexity. Section 5 contains some negative complexity results indicating that our algorithms cannot be improved much. Section 6 discusses and analyzes a few natural heuristics and provides some bounds on the "value of information." Section 7 contains a discussion of further variations of our model.

2. THE MODEL AND NOTATION

We define a *random network* $G = (\mathcal{N}, \mathcal{A}, \mathcal{P})$, by a triple consisting of a set $\mathcal{N}$ of nodes ($|\mathcal{N}| = n$), a set $\mathcal{A}$ of arcs ($|\mathcal{A}| = m$), and a probability distribution $\mathcal{P}$ describing the statistics of the arc costs. In particular, the cost C_{ij} of each arc (i, j) is assumed to be a random variable and $\mathcal{P}$ specifies the joint probability distribution of these random variables. The arcs could be either directed or undirected; in the undirected case, it is assumed that the cost is the same for both directions of travel through an arc.

Note that the sets $\mathcal{N}$ and $\mathcal{A}$ are assumed to be the same under every realization. On the other hand, the distribution of the arc costs can be chosen so that under a particular realization the cost of some arc is prohibitively large. Thus, in effect, our model encompasses the situation where certain arcs are absent with some probability, as in the model of [2]. The same comment applies to the case where certain nodes are absent with some probability.

We propose two different ways of specifying the probability distribution $\mathcal{P}$. In the first model, we assume that there is a set $\mathcal{R} = \{1, \ldots, R\}$ of possible realizations of the vector of arc costs, with the rth realization having probability p^r. We use c_{ij}^r to denote the cost of arc (i, j) under the rth realization. It is clear that under this model the costs of different arcs will be, in general, dependent random variables. The stochastic shortest path problem that is obtained under this model will be referred to as R-SSPPR.

In the second model, we assume that the costs of different arcs are independent random variables. The probability distribution $\mathcal{P}$ can be then determined by specifying the statistics of each arc cost C_{ij}. We assume that the range of C_{ij} is finite and of cardinality K_{ij}. We are then given, for each $(i, j) \in \mathcal{A}$, the possible values c_{ij}^r, $r = 1, \ldots, K_{ij}$, of C_{ij}, together with the associated probabilities p_{ij}^r. The stochastic shortest path problem that is obtained under this model will be referred to as i-SSPPR.

We note that the i-SSPPR can be viewed as a special case of the R-SSPPR, except that a large number $R = \prod_{(i,j)\in\mathcal{A}} K_{ij}$ of realizations is needed. In practice, the different arc costs are usually dependent; e.g., high congestion on one arc (or road) might imply high congestion on other arcs as well. For this reason, the R-SSPPR with a moderate value of R could sometimes be a realistic formulation.

Besides the random network G, we are given an origin node $s \in \mathcal{N}$ and a destination node $t \in \mathcal{N}$. We consider a vehicle that starts at node s, travels through the network, and ends up at node t. In doing so, the vehicle incurs a cost equal to the sum of the costs of the arcs that it traverses. The key difference from the classical shortest path problem is that when the vehicle starts traveling it does not know the realized values of the arc costs; it only knows their statistics, as summarized by the probability distribution $\mathcal{P}$. As the vehicle moves through the network, its information increases. In particular, we assume that whenever the vehicle visits a new node i it learns (and remembers) the realization of C_{ij} for every arc (i, j) emanating from i.

A *policy* of the vehicle is defined as a set of rules that, given the location of the vehicle and the information that it has collected, determines the arc that should be traversed next. The cost of a policy is the sum of the costs of the arcs traversed until the vehicle reaches the destination node t. As the cost of a policy depends on the realization of the arc costs, it is a random variable. Our objective is to find a policy that has the smallest possible expected cost.

The SSPPR is a stochastic problem, because of the randomness of the arc costs, and dynamic, because the vehicle's information changes dynamically and, in fact, the information acquired also depends on the vehicle's decisions. The SSPPR can be viewed as a stochastic programming problem with recourse; the readjustment of the vehicle's path based on any newly acquired information can be viewed as a recourse action. It is more useful, however, to view the SSPPR as a stochastic control problem with imperfect information [4]. Such problems can be solved, in principle, using the dynamic programming methodology, although the resulting algorithms typically have prohibitively high complexity [15]. One of the objectives of this paper was to study the extent to which the SSPPR is an intractable problem and to determine conditions under which it can be solved realistically.

We end this section with an assumption which will be in effect throughout the remainder of the paper. This assumption is introduced in order to guarantee that the expected cost of an optimal policy is not equal to $-\infty$ and is a natural generalization of the assumptions commonly made for deterministic shortest path problems.

Assumption. *Under every possible realization of the network, the cost of every cycle is nonnegative.*

3. A DYNAMIC PROGRAMMING ALGORITHM FOR THE R-SSPPR

The R-SSPPR is a stochastic control problem with imperfect information because the actual realization of the random network is not known by the vehicle. On the other hand, as is customary with imperfect information problems, it can be converted to a problem with perfect information by suitably redefining the state vector. In particular, the state vector should encompass whatever information is relevant to the future decisions of the vehicle: its current location and the arc cost information collected thus far. In this section, we describe how this can be accomplished and we bound the complexity of the resulting algorithm.

Recall that $\mathcal{R} = \{1, \ldots, R\}$ is the set of possible realizations of the network. Any subset of $\mathcal{R}$ will be called an *information set*. Initially, the vehicle may only know that the actual realization of the network belongs to $\mathcal{R}$. As the vehicle travels through the network, it can eliminate those realizations that are incompatible with the observed values of the C_{ij}'s. In particular, at any time, the vehicle possesses an information set I (or "has information I") which is the set of all $r \in \mathcal{R}$ such that C^r_{ij} is equal to the observed value of C_{ij}, for all arcs (i, j) such that the vehicle has visited node i. If I is a singleton, then there is a single realization compatible with the vehicle's observations which means that the vehicle knows (or can infer) the value of C_{ij} for every arc of the network. Initially, when $I = \mathcal{R}$, the probabilities of the different elements of I are equal to the prior probabilities of the different realizations. Later on, when some of the realizations are eliminated, the probabilities of the remaining possibilities can be easily evaluated: If I is the current information set, then

$$\Pr(r|I) = \frac{p^r}{\sum_{q \in I} p^q}. \tag{1}$$

Given a current information set I, the set $\mathcal{A}$ can be divided into two subsets:

(a) The set $\mathcal{A}^d_I$ of arcs for which the value of C_{ij} can be inferred from the available information; formally, $\mathcal{A}^d_I = \{(i,j) \in \mathcal{A} \mid c^{r_1}(i,j) = c^{r_2}(i,j), \forall r_1, r_2 \in I\}$. We call these arcs *deterministic (given I)*.

(b) The set $\mathcal{A}^u_I$ of arcs for which the value of C_{ij} cannot be inferred from the available information; formally, $\mathcal{A}^u_I = \{(i, j) \in \mathcal{A} \mid \exists r_1, r_2 \in I, c^{r_1}(i, j) \neq c^{r_2}(i, j)\}$. We call these arcs *uncertain (given I)*.

For some more terminology, if i is a node and at least one of its outgoing arcs (i, j) is uncertain given I, we say that node i is *an information collection node* (given I); let $\mathcal{N}^c_I$ be the set of such nodes. The rationale behind this definition is the following: If the vehicle has information I and visits next a node i which is not an information collection node, then the values of C_{ij} are already known for all arcs emanating from i, no new information is provided to the vehicle, and the information set I remains the same. If, on the other hand, i is an information collection node, then the vehicle will learn the realization of C_{ij} for some uncertain arc and will be able to eliminate at least one of the realizations. Thus, the cardinality of the information set decreases each time that an information collection node is visited.

We now describe the "dynamics" of the changes of the information set. Suppose that the vehicle has information I and visits next a node $i \in \mathcal{N}_I^c$. The new information set, after i is visited, is some proper subset I' of I. Given the value of I and i, I' is a random variable because it depends on the random (and uncertain) costs of some of the arcs emanating from i. In particular, I' can be expressed as a function $h(i, I, r)$ of i, I, and the actual realization r of the network. This functional dependence can provide us with the statistics of I' given I and i.

We are now ready to present a dynamic programming algorithm for the R-SSPPR. Let $V(i, I)$ be the expected cost-to-go (until the vehicle reaches the destination node t), under an optimal policy, assuming that the vehicle starts at node i and has information I. If I is a singleton, then $V(i, I)$ is the shortest path length from i to t in a graph without any arc cost uncertainty and can be determined by running a shortest path algorithm. Suppose now that we have already determined $V(i, I)$ for all i and all I of cardinality $k - 1$ or less. Consider some node i and some I of cardinality k. We will show how $V(i, I)$ can be evaluated. The cost of the vehicle starting at node i with information I and until it reaches node t can be broken up as follows: The vehicle goes to an information collection node $j \in \mathcal{N}_I^c$, acquires a new (and smaller) information set I', and then, starting from node j with information I', goes to node t. The only other possibility is that the vehicle goes directly from i to t without passing through an information collection node. Using the principle of optimality, once the vehicle reaches node j and acquires information I', its expected cost-to-go should be the optimal cost-to-go $V(j, I')$. This is equivalent to deleting all arcs emanating from j (for every information collection node j) and replacing them with a single arc (j, t) with cost $E[V(j, I')]$. Having done so, $V(i, I)$ is simply the shortest path length from i to t in this newly defined graph. A few remarks are in order regarding the arc costs in this new graph: If j is not an information collection node given I, the costs of all arcs emanating from j are uniquely determined by I. If, on the other hand, $j \in \mathcal{N}_I^c$, we only need to know the value of $E[V(j, I')]$. We note that I' has cardinality smaller than that of I and, according to our earlier assumption, that $V(j, I')$ is available for every relevant value of I'. The expectation is necessary because, as discussed earlier, I' is a random variable whose statistics are determined from j and I.

We now estimate the complexity of this algorithm: Since there are R realizations, there are $2^R - 1$ choices for I and we have to solve $2^R - 1$ all-origin single-destination shortest path problems. This can be done in time $O(2^R n^3)$ or, if all arc costs are nonnegative, in time $O(2^R n^2)$. There is some additional work required in order to determine the values of $E[V(j, I')]$. For each I, there are $O(n)$ information collection nodes j that need to be considered. Given I and j, the probability of I' can computed using the formula

$$\Pr(I' \mid j, I) = c \cdot \sum_{\{r \in \mathcal{R} \mid h(j, I, r) = I'\}} p^r, \qquad (2)$$

where c is a normalizing factor, so that $\sum_{I'} \Pr(I' \mid j, I) = 1$. We note that the union of all information sets I' that can result from a given I and j has cardinality $|I|$, which is bounded by R. In particular, the summations in (2) can be carried out for all relevant values of I' in $O(R)$ time. We conclude that this additional work is of the order of $O(R2^R n)$. We also note that once $V(i, I)$ has been computed for every i and I an optimal policy is easily determined.

We summarize this discussion in the following:

Theorem 1. *The R-SSPPR can be solved in time $O(2^R(Rn + n^3))$, in general, and in time $O(2^R(Rn + n^2))$ if all arc costs are nonnegative.*

Remarks:

1. A path followed under an optimal policy might contain a cycle even if the arc costs are positive under every realization, as can be demonstrated by simple examples [2, 17]. This is because information gathering could be an important feature of an optimal policy. For example, it might be profitable in the expected value sense to find out the realization of a certain arc, and if its realization is not what was hoped for, to backtrack to the origin node and try an alternate path. Let us now assume that every cycle has positive length under every realization. Then, it is easily seen that any cycle resulting from an optimal policy is traversed only once. The reason is the following: The path resulting from an optimal policy consists of a sequence of shortest paths obtained from certain deterministic shortest path problems. One such shortest path ends and another starts when an information collection node is reached. Since shortest paths in deterministic networks (under the positive cost cycle assumption) do not contain cycles, any cycle must contain an information collection node. But if a cycle has been traversed once, its nodes cease to be information collection nodes and therefore the same cycle will not be traversed again.
2. Using the observations in the preceding remark, and assuming that cycle costs are positive, under a path followed by an optimal policy, there must be a visit to an information collection node between any two visits

to the same node. Since visits to information collection nodes result in a reduction in the cardinality of the information set and in a reduction of the number of information collection nodes, we conclude that no node can be visited more than $\min(n, R)$ times. In particular, the number of arcs in the path followed by an optimal policy is bounded by $n \min(n, R)$ under every realization. If the positive cost cycle is relaxed and we only require cycle costs to be nonnegative, the same reasoning shows that there still exists an optimal policy under which no more than $n \min(n, R)$ arcs are traversed. (However, not all optimal policies need to have this property.) This upper bound on the number of arcs that may have to be traversed turns out to be tight within a constant factor, as shown by an example on p. 58 of [17], for the case of directed networks.

3. When actually solving an instance of R-SSPPR, many of the information sets I might never be reached and the computation of the corresponding $V(i, I)$ might be unnecessary. One way of exploiting this, in order to reduce the computational burden, is to use a forward implementation of the algorithm: When trying to compute some $V(i, I)$, if the value of some $V(j, I')$ with $|I'| < |I|$ is needed, pause to compute $V(j, I')$, using the same method (recursively). This variation of the basic algorithm can reduce substantially the run time in practice; however, its theoretical worst-case performance is the same.
4. The R-SSPPR belongs to the class of "stochastic shortest path problems," in the terminology of [5], i.e., it is a controlled Markov chain [in our case, the state is (i, I)] and the objective is to reach a terminal state [in our case, any state of the form (t, I)], with minimal total (undiscounted) expected cost. While there are general purpose dynamic programming algorithms for stochastic shortest path problems, their complexity is substantially larger than is the complexity of the algorithm proposed here (cf. Theorem 1). The reason for our better complexity estimate is the special structure of the R-SSPPR: Its state space consists of a sequence of R layers (with successive layers associated with a smaller cardinality of the set I); the state can only move from one layer to a random state in a next layer or it can move to another state in the same layer; and in the latter case, the next state is a deterministic function of the decision variable. It is not hard to see that any Markov decision problem with such a structure can be solved by computing the cost-to-go function in one layer at a time using a deterministic shortest path algorithm at each layer.
5. This algorithm is easily amenable to parallel implementation, with different processors in charge of computing $V(i, I)$ for different choices of I.

4. A DYNAMIC PROGRAMMING ALGORITHM FOR THE i-SSPPR

In this section, we present an algorithm for the i-SSPPR, similar to the one proposed for the R-SSPPR. The main difference between the two algorithms is in the manner in which the information component of the state vector is defined.

Note that our algorithm for the R-SSPPR can be applied to the i-SSPPR since the latter is a special case of the former. However, we would need to let $R = O(K^m)$, where K is a bound on the cardinality of the range of each C_{ij} and m is the number of arcs. Since the algorithm for the R-SSPPR is exponential in R, we would obtain an algorithm for the i-SSPPR which is doubly exponential in m. We will show shortly that a singly exponential algorithm is possible.

As the state vector in our dynamic programming algorithm for the i-SSPPR, we take the present location of the vehicle together with an information component I which for every arc provides the value of its cost, if its cost has been already learned by the vehicle, or an indication that this particular arc has not been observed yet. With m arcs, a typical state vector will be an $(m + 1)$-tuple; there are n choices for the first entry in the state vector and at most $K + 1$ choices for each one of the remaining entries. (There are up to K possible realizations of C_{ij} and an additional possibility that the realization has not been learned yet.) We conclude that the size of the state space is $O(n(K + 1)^m)$.

Once the state space has been defined as above, an algorithm is obtained in pretty much the same way as for the R-SSPPR. We say that an arc (i, j) is deterministic, given information I, if its cost is uniquely determined from I; otherwise, the arc is said to be uncertain. We say that a node is an information collection node if at least one of the arcs emanating from that node is uncertain. When an information collection node is reached (and only then), I changes and the number of uncertain arcs strictly decreases. If I' is the new information, it includes the costs of the newly observed arcs, and these costs are selected randomly (according to their prescribed probability distributions) and independently of everything else.

Let $V(i, I)$ be the expected cost-to-go starting from state (i, I). Similarly with the R-SSPPR, $V(i, I)$ can be computed recursively for all (i, I) by solving $O((K + 1)^m)$ deterministic single-destination shortest path problems. [In this recursion, we start by computing $V(i, I)$ for those I for which all arc costs are known and proceed to the computation of $V(i, I)$ with fewer known arc costs.] Finally, for every I and every information collection node j (under I), we need to compute $E[V(j, I') \mid j, I]$, where I' is the new information after j is reached. We note that given I and j there are $O(K^n)$ possible values for I' [we have $O(n)$ arcs outgoing from j and K possibilities for each]. The probability of each possibility for I' is determined by multiplying the probability of each choice for every particular arc (the independence assumption is used here). These latter multiplications need only be done once for each node j; so, the total work spent for such multiplications (summed over all nodes in the network) is $O(nK^n)$. Once these probabilities are available, each expectation $E[V(j, I')|j, I]$ can be computed in time $O(K^n)$. Thus, the amount of computation needed besides running shortest path algorithms is $O(n(K+1)^m K^n)$.

We summarize our discussion of the i-SSPPR in the following theorem:

Theorem 2. *The i-SSPPR can be solved in time $O((K+1)^m(nK^n + n^3))$, in general, and in time $O((K+1)^m(nK^n + n^2))$ if the arc costs are nonnegative.*

5. COMPLEXITY RESULTS

Theorem 1 shows that the R-SSPPR can be solved in time polynomial in n, if R is held fixed. On the other hand, our algorithm is exponential in R. Similarly, for the i-SSPPR, our algorithm is exponential in the number of arcs. In this section, we show that polynomial time algorithms for these problems are unlikely to exist. Some indication of the difficulty of these problems has been provided in [1] where it is observed that a full description of an optimal policy may involve an exponentially long table. Our results go beyond this observation and establish that the problem is difficult even if such a description in the form of a table is not required.

We now state formally the recognition version of the R-SSPPR:

INSTANCE: A graph $(\mathcal{N}, \mathcal{A})$; two nodes $s, t \in \mathcal{N}$; a positive integer R; positive integers C_{ij}^r, for $(i, j) \in \mathcal{A}$ and $r = 1, \ldots, R$; positive rational numbers $p^1, \ldots, p^R$ that sum to 1; a positive rational number B.

QUESTION: Does there exist a policy whose expected cost is less than or equal to B?

Theorem 3. *The recognition version of the R-SSPPR is NP-complete, for both directed and undirected networks.*

Proof. We first show that R-SSPPR $\in$ NP. Suppose that we have a "YES" instance of R-SSPPR and let μ^* be an optimal policy. Such a policy can be described by specifying the R paths that are followed by the vehicle under each possible realization of the network; let π_r^* be the path followed under the rth realization. We claim that a policy described as above is a certificate that can be used to verify in polynomial time that we are dealing with a "YES" instance.

We first note that (see Remark 2 in Section 3) each path need not involve more than $n \min\{n, R\}$ arc traversals; thus, this description of a policy is of polynomial length. The next step is to verify that the given paths do correspond to an admissible policy. In particular, we must check that two paths π_r^* and $\pi_{r'}^*$ separate only after an information collection node is reached at which the vehicle realizations r and r' lead to observable differences. Given the bound on the number of arcs in each path, this can be also checked in polynomial time. Finally, we need to evaluate the cost L_r of each path π_r^* (with respect to the arc costs C_{ij}^r) and then compute the expected cost of the policy which is $\sum_{r=1}^R p^r L_r$. All these computations can be done in polynomial time, which establishes that the problem belongs to NP.

We will now reduce the undirected Hamiltonian path problem to the R-SSPPR for undirected networks. In the undirected Hamiltonian path problem, we are given an undirected graph $G' = (\mathcal{N}', \mathcal{A}')$ and we are asked whether there exists a path which visits every node exactly once. This problem is known to be NP-complete (see p. 199 of [8]).

Assume that we are given an instance $G' = (\mathcal{N}', \mathcal{A}')$ of the Hamiltonian path problem and that $\mathcal{N}' = \{1, \ldots, n\}$. We construct from G', a graph $G = (\mathcal{N}, \mathcal{A})$, as follows: We let $\mathcal{N} = \mathcal{N}' \cup \{s, t\}$ and $\mathcal{A} = \mathcal{A}' \cup \{(s, i), (i, t) | i = 1, \ldots, n\}$. We assume that there are $R = n$ possible realizations for the arc costs and $\mathcal{R} = \{1, \ldots, R\}$. The arc costs in G take the following values under the corresponding realizations:

$$c_{ij}^r = 1, \quad (i, j) \in \mathcal{A}', r \in \mathcal{R},$$

$$c_{si}^r = 0, \quad i \in \mathcal{N}', \quad r \in \mathcal{R},$$

$$c_{it}^r = \begin{cases} 0, & \text{if } r = i, \\ \infty, & \text{if } r \neq i, \end{cases} \quad i \in \mathcal{N}', \quad r \in \mathcal{R}.$$

Let $B = (R-1)/2$ and let $\Pr(r|\mathcal{R}) = 1/R$ for all $r \in \mathcal{R}$. This completes the construction of an instance of R-SSPPR.

Assume that we have a "YES" instance of the Hamiltonian path problem. Based on a Hamiltonian path for G, we construct the following policy μ^*: Starting from node s, go to the first node of the Hamiltonian path and then follow that path. When at node r, if $c_{rt} = 0$, we can conclude that the realization of the network is r and that node t should be reached following arc (r, t); otherwise, continue to the next node in the Hamiltonian path. We notice that policy μ^* ensures that t is reached, and since G' is a "YES" instance for the Hamiltonian path problem, and $c_{ij}^r = 1$ for all $r \in \mathcal{R}$ and all $(i, j) \in \mathcal{A}'$, the expected cost of this policy is

$$\frac{1}{R} \sum_{i=0}^{R-1} i = \frac{R-1}{2}.$$

This shows that we have a "YES" instance of R-SSPPR.

Suppose now that we have a "NO" instance of the Hamiltonian path problem. We observe that any policy with finite expected cost visits the nodes of G' in some order, and when a node $r \in \mathcal{N}'$ is visited with $c_{rt} = 0$, the vehicle reaches t by traversing arc (r, t). Since no Hamiltonian path exists, and since all nodes have to be visited under the worst possible realization, some node will have to be visited twice with positive probability. This results in the expected cost of the policy being strictly larger than $(R-1)/2$ and we have a "NO" instance of the R-SSPPR.

A similar reduction also works for the case of directed networks, since the Hamiltonian path problem is NP-complete for directed graphs as well. ■

We now provide a formal definition of the i-SSPPR and a corresponding complexity result:

INSTANCE: A graph $(\mathcal{N}, \mathcal{A})$; two nodes $s, t \in \mathcal{N}$; a positive integer R; positive integers C^r_{ij}, for $(i, j) \in \mathcal{A}$ and $r = 1, \ldots, K$; positive rational numbers $p^1_{ij}, \ldots, p^K_{ij}$ such that $\sum^K_{k=1} p^k_{ij} = 1$; a positive rational number B.

QUESTION: Does there exist a policy whose expected cost is less than or equal to B, assuming that the costs of the different arcs are independent random variables?

Theorem 4. *The i-SSPPR is #P-hard, and can be solved in polynomial space, for both directed and undirected networks.*

Proof. We first discuss membership in PSPACE. [14] defines a class of decision-making problems under uncertainty which is called SAP-TIME (stochastic alternating polynomial time). In this class of problems, one deals with a controlled Markov chain. The goal is to minimize the expectation of some functional depending on the history of states and decisions. The problem i-SSPPR can be viewed as such a controlled Markov chain with state (i, I), where the information vector I is as defined in the proof of Theorem 2. For this reason, i-SSPPR belongs to SAPTIME and [14] establishes that SAPTIME = PSPACE. In fact, the definition of SAPTIME given in [14] is more precise; for a problem to belong to SAPTIME, there must be certain bounds on the time horizon, the size of the state space, the size of the control space, and the size of the numbers involved. It is not hard to show that instances of i-SSPPR do satisfy all of these requirements, after a minor reformulation; the details can be found in [17].

We now present a polynomial-time reduction of the reliability problem to i-SSPPR. Reliability, which is #P-hard [19], for both cases of directed and undirected graphs is formally defined as follows:

INSTANCE: A graph $G' = (\mathcal{N}', \mathcal{A}')$ in which arcs fail independently with rational probability $p \in [0, 1]$ and two nodes $s, t \in \mathcal{N}'$.

OUTPUT: The probability $f(G', s, t; p)$ that there is a path from s to t without failed edges.

We first consider the undirected case, and then extend the reduction to the case of directed networks. Given an instance $(\mathcal{N}', \mathcal{A}', s, t, p)$ of the reliability problem, we construct an instance of i-SSPPR with node set $\mathcal{N} = \mathcal{N}'$ and arc set $\mathcal{A} = \mathcal{A}' \cup \{(s, t)\}$. Regarding the arc costs, we assume that $C_{st} = 1$ with probability 1 and

$$C_{ij} = \begin{cases} 0, & \text{with probability } 1-p, \\ 1, & \text{with probability } p, \end{cases} \qquad \forall (i, j) \in \mathcal{A}'.$$

The following policy is easily seen to be optimal: While using only zero-cost arcs, explore as much of the graph as possible. (Given that the graph is undirected, the vehicle can always backtrack and therefore the exploration strategy is immaterial). If the destination t is reached, stop. If the destination t cannot be reached, backtrack to node s and traverse arc (s, t). The expected cost of this policy is equal to the probability that there exists no path from s to t consisting of zero-cost arcs, which is $1 - f(G', s, t; p)$. As a consequence, i-SSPPR is #P-hard, for the undirected case.

The reduction given for the directed the case does not work for the directed case because the vehicle may be unable to backtrack. This can be remedied by introducing additional arcs of the form (i, s), $i \in \mathcal{N}$, whose cost is zero with probability 1. With this modification, our reduction goes through and establishes that i-SSPPR is #P-hard for the directed case as well. ■

We note that the proof of Theorem 4 shows that the i-SSPPR remains #P-hard even for the case where each arc cost can take at most two different values.

6. HEURISTICS AND BOUNDS ON THE OPTIMAL COST

In this section, we compare the cost of an optimal policy for the SSPPR with the cost resulting from the application of a heuristic policy (i.e., an easily computable but non-optimal policy). A few different heuristic policies are considered and a set of results (mostly negative) are derived. Our results are presented separately for the R-SSPPR and the i-SSPPR, and we also distinguish between directed and undirected networks for each case. Throughout this section, we assume that arc costs are nonnegative with probability 1.

Before proceeding to the detailed development, we introduce some notation. We use C_H to denote the expected cost corresponding to some heuristic policy H, and C_{OPT} to denote the expected cost of an optimal policy. We also define C_{FI} (for "full information") as the expected cost had the realization of all arc costs been known when at s. (The expectation is taken again with respect to the probabilities of the different realizations.) It is clear that the following inequalities hold for every instance of SSPPR and for every heuristic policy H:

$$C_{FI} \leq C_{OPT} \leq C_H. \tag{3}$$

We will be interested in the ratio C_H/C_{OPT} which characterizes the quality of a heuristic and in the ratio C_{OPT}/C_{FI} which measures the value of information. (We also note that the ratio C_H/C_{FI} is similar to the ratio that was studied in [16] and which is sometimes called the "competitive ratio.")

6.1. The Certainty Equivalent Heuristic

An easy way of obtaining a policy is to replace the arc costs by their expectations, obtain a shortest path under these new (deterministic) arc costs, and let the vehicle follow such a shortest path no matter what information it acquires along the way. We call this heuristic the certainty equivalent heuristic and we denote the cost of the resulting policy by C_{CE}. A main drawback of this heuristic is that it makes no use of available information, and for this reason, its cost can be quite high compared to the optimal cost, as we now demonstrate:

Consider an instance of the R-SSPPR, involving an undirected graph. Assume that there exist only two paths from s to t (an "upper" path and a "lower" path) and two equally likely realizations. All arc costs are zero except that under the first (respectively, the second) realization, the lower (respectively, the upper) path has an arc with a cost of 1. The certainty equivalent heuristic (as any other policy that does not use the available information) results in an expected cost of 1/2, whereas the optimal cost is zero. Essentially the same example would lead to the same conclusion for the case of directed graphs as well.

The cost C_{CE} can be arbitrarily worse than the optimal for the i-SSPPR as well (for both directed and undirected networks), as we now show. Consider the undirected graph in Figure 1. Let the arcs of the form (s, i) have zero length. The length of each arc of the form (i, t) is equal to 1 with probability p and equal to 0 with probability $1 - p$. Then, the cost C_{CE} is equal to p and the optimal cost C_{OPT} is p^k. As a consequence, the ratio C_{CE}/C_{OPT} can become arbitrarily large if we keep p constant and we increase k or if we keep k constant and decrease p. An example involving directed graphs leading to the same conclusion is be obtained if we take the same instance, assign a direction from left to right to all arcs, and add zero-cost arcs of the form (i, j), $i, j = 1, \ldots, n$.

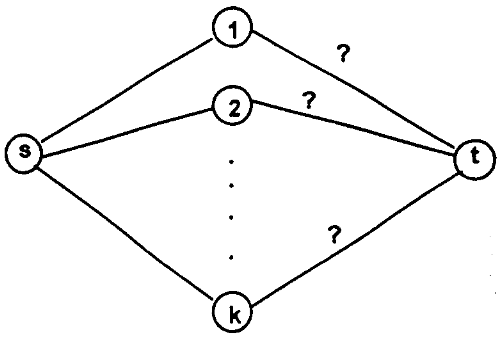

Fig. 1. An undirected graph.

Having obtained these negative results for the certainty equivalent heuristic, we now turn our attention to heuristics that try to make some use of new information when it becomes available.

6.2. Heuristics for the R-SSPPR

A Naive Adaptive Heuristic

We start by considering the following "naive adaptive heuristic" H_N for the R-SSPPR. Let L_r be the shortest path length from s to t, under the rth realization of the network, and let π_r be a corresponding shortest path. Assume that the realizations have been indexed so that $L_1 \leq L_2 \leq \cdots \leq L_R$. The heuristic proceeds as follows: The vehicle first behaves as if the realization of the network is 1 and follows π_1. More generally, it behaves as if the realization is r and follows π_r. If, along the way, it finds out that the realization is not r, it returns to s and assumes that the realization is the lowest indexed realization compatible with the information collected thus far. The vehicle will eventually reach t, because once it guesses the correct realization r, it will follow the path π_r to the end.

Theorem 5. *Consider the R-SSPPR for undirected networks. For every instance, we have*

(*a*) $C_{H_N}/C_{FI} \leq 2\ min\{n, R\}$.
(*b*) $C_{H_N}/C_{OPT} \leq 2\ min\{n, R\}$.
(*c*) $C_{OPT}/C_{FI} \leq 2\ min\{n, R\}$.

Furthermore, there exist instances for which:

(*d*) $C_{H_N}/C_{FI} = min\{n, R\}$.
(*e*) $C_{H_N}/C_{OPT} = min\{n, R\}$.
(*f*) $C_{OPT}/C_{FI} = min\{n, R\}$.

Proof. Assume first that $R \le n$. Suppose that the true realization is some r^*. (In particular, C_{FI} is the expectation of L_{r_*}.) Whenever the vehicle assumes that r is the true realization and tries the path π_r, we must have $r \le r^*$. Thus, the total length traversed by the vehicle, including backtracking to s, is bounded by $2 \sum_{r=1}^{r^*-1} L_r + L_{r*} \le 2RL_{r*}$, from which we obtain $C_{H_N} \le 2RC_{FI}$. Notice that the vehicle can make incorrect assumptions on the true value of r at most n times, because the assumption on the true value of r can only change when the vehicle visits a new information collection node. For this reason, the bound of $2RL_{r*}$ can be improved to $2 \min\{n, R\} L_{r*}$, which establishes part (a) of the result. Parts (b) and (c) follow immediately.

To prove part (f), we construct an instance for which $C_{OPT}/C_{FI} = R$. The underlying graph is the one shown in Figure 1. There are $R = k$ possible realizations which are equally likely. Arcs of the form (s, i) have unit length. Arcs of the form (i, t) have length M, with $M > 2R$, except that under realization r, arc (r, t) has zero length. Clearly, $C_{FI} = 1$. For this instance, the naive adaptive heuristic behaves as follows: Go to node 1; if $c_{1t} = 0$, then go to node t; otherwise, go back to node s and then to node 2, etc. The expected cost of this policy is

$$C_{OPT} = \frac{1}{R} \sum_{i=1}^{R} (1 + 2(i - 1)) = R,$$

which establishes part (d). It is easily seen that, for this instance, the heuristic H_N is also an optimal policy which establishes part (f) as well.

To prove part (e), we modify the instance of Figure 1 as in Figure 2. Here, the cost of arc (a, b) is random and takes one of R different values and its value provides full information on the actual realization. [The statistics of the arcs (i, t) are the same as before.] The same calculation as in the preceding paragraph shows that $C_{H_N} = R$. On the other hand, an optimal policy would first visit node a, learn the true realization, and then follow a shortest path to the destination t, for a total cost of 1. ■

It should be clear that parts (c) and (f) of this theorem are not related to the particular heuristic being studied. Instead, they establish that information is valuable in that it can improve costs by a factor of R but no more than $2R$.

A further improvement to the heuristic is obtained as follows: Once an assumed value of r is invalidated and a new value of r is assumed, instead of returning to s and following π_r, find a shortest path from the current node to node t (with respect to the arc costs under the newly assumed realization) and follow it until the new assumption about r is invalidated. We might expect this modification to lead to better performance, although the results of Theorem 5 would not be changed.

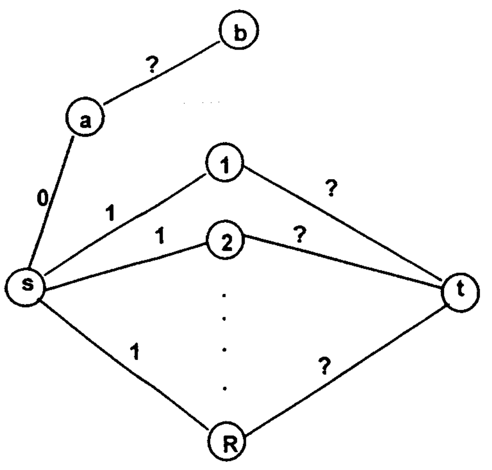

Fig. 2. An instance of R-SSPPR for the proof of part (e) of Theorem 5.

The naive adaptive heuristic is not useful in the case of directed networks because, once the vehicle leaves the origin s along a certain path, it might be impossible to return. For this reason, both C_{H_N}/C_{OPT} and C_{OPT}/C_{FI} are unbounded above, for the directed case.

An Open Loop Feedback Heuristic

The "open loop feedback" heuristic is a general purpose heuristic for stochastic problems. The idea is to start by following a path which is shortest with respect to the expected values of the arc costs, as in the certainty equivalent heuristic. However, as soon as the vehicle acquires some new information, the probability distribution of the arc costs is replaced by the conditional distribution given the new information; then, a new path with the least expected cost is computed and followed, and so on. This procedure is repeated until node t is reached. We notice that the amount of computation needed to implement this heuristic is polynomial in R and n. (We basically need to solve at most $\min\{R, n\}$ deterministic shortest path problems, for each particular realization.)

Unfortunately, this heuristic does not enjoy any favorable performance guarantees. As for the certainty equivalent heuristic, it is not hard to construct examples for which the optimal cost is zero and the cost of this heuristic is positive.

6.3. Heuristics for the i-SSPPR

We restrict our discussion to the undirected case. Since the i-SSPPR is a special case of the R-SSPPR, it follows from Theorem 5 that the naive adaptive heuristic satisfies $C_{H_N}/C_{FI} \le 2n$. It might appear that implementing this heuristic could require an exponential amount of computation, since we need to find a shortest path under each realization and then sort these shortest paths in order of increasing lengths. However, a more efficient implementation is possible which we now describe: At any point in time, assume that all arc lengths are equal to the lowest possible values, given the information observed so far, and follow a path to node t that would be shortest if this assumption were true. Thus, implementation of this heuristic only requires that we solve a new shortest path problem each time that we encounter an arc whose value is larger than the value that was assumed. Therefore, at most n shortest path problems need to be solved and the total computational requirements are polynomial. (On the other hand, evaluating the expected cost C_{H_N} associated with this heuristic appears to be much more difficult, because the expectation is the sum of an exponential amount of terms. In fact, it can be shown that the evaluation of C_{H_N} is a #P-complete problem. The proof is fairly similar to the proof of Theorem 4.)

Using Theorem 5(c), we obtain $C_{OPT}/C_{FI} \leq 2n$. We conjecture that this bound is pretty tight, i.e., there are instances of i-SSPPR for which $C_{OPT}/C_{FI} = \Omega(n)$.

7. EXTENSION AND CONCLUSIONS

We discuss here a few variations of our model:

There are several possible variations of the mechanism whereby new information on arc costs is acquired. For example, we could have assumed that the length of an arc is learned only after traversing it. Under this variation, the certainty equivalent solution is optimal when arc costs are independent random variables (i-SSPPR); for the R-SSPPR, a dynamic programming algorithm similar to the one in Section 3 is possible.

In a more complicated variant, we can allow for an option of obtaining information on the costs of remote arcs, except that a price has to be paid whenever such information is to be obtained. (This could represent an "intelligence gathering" activity.) The dynamic programming methodology easily extends to this setting [17].

In another variant, we may assume that global information on the realization of the arc costs is obtained at some time τ, after the vehicle starts its journey. (The time τ could be either a deterministic constant or an exponentially distributed random variable.) For the model to be complete, we need to specify the relation, if any, between arc costs and travel times. For example, we may assume that each arc traversal takes unit time and that there is a penalty for waiting in place. Alternatively, we may assume that travel times are proportional to arc costs. Both alternatives can be approached via dynamic programming [17] similarly with Section 3.

Another possible direction involves the case in which arc costs change with time, according to a stochastic process. As in the R-SSPPR, we can assume that there are R possible realizations and that the actual realization r changes according to a Markov chain. If the value of r is observed perfectly, we obtain a Markov decision problem with state (i, r), where i is the current location of the vehicle. In the case where r is not perfectly known, but we attempt to infer r from the observed arc costs and knowledge of underlying statistics, the problem can be still handled, in principle, through dynamic programming, but its computational requirements are much more substantial.

In a model similar to the i-SSPPR, we can assume that each arc evolves independently as a Markov chain. This problem can be solved in polynomial time for the case of directed acyclic graphs [18], but seems to be quite difficult for general graphs.

Overall, this paper has analyzed and discussed a number of stochastic shortest path problems in the absence of full arc cost information. Our main interest was in understanding the conditions under which problems of this type can be considered to be solvable. The distinction between the R-SSPPR and the i-SSPPR and the results obtained for these problems highlight the fact that modeling uncertainty is crucial for problems of this type and that there may be a significant tradeoff between modeling accuracy and solvability, e.g., the R-SSPPR with a small value of R is easiest to solve but could be a poor model in some settings.

This research was supported by the C. S. Draper Laboratory under Contract DL-H-441625 and by a grant from Siemens A. G.

REFERENCES

[1] G. Andreatta, Shortest path models in stochastic networks. *Stochastics in Combinatorial Optimization* (G. Andreatta, F. Mason, and P. Serafini, Eds.). World Scientific, Singapore (1987).

[2] G. Andreatta and L. Romeo, Stochastic shortest paths with recourse. *Networks* **18** (1988) 193–204.

[3] A. Bar-Noy, and B. Schieber, The Canadian traveller problem. *Proceedings of the 2nd Annual ACM-SIAM Symposium on Discrete Algorithms,* San Francisco, CA (1991).

[4] D. P. Bertsekas, *Dynamic Programming, Deterministic and Stochastic Models.* Prentice-Hall, Englewood Cliffs, NJ (1987).

[5] D. P. Bertsekas, and J. N. Tsitsiklis, An analysis of stochastic shortest path problems. *Math. Oper. Res.* **16**(3) (1991) 580-595.

[6] J. S. Croucher, A note on the stochastic shortest route problem. *Nav. Res. Log. Q.* **25** (1978) 729–732.

[7] H. Frank, Shortest paths in probabilistic graphs. *Oper. Res.* **17** (1969) 583–599.

[8] M. R. Garey and D. S. Johnson, *Computers and Intractability, A Guide to the Theory of NP-Completeness.* W. H. Freeman, New York (1979).

[9] R. Hassin and E. Zemel, On shortest paths in graphs with random weights. *Math. Oper. Res.* **10** (1985) 557–564.

[10] R. C. Larson and A. R. Odoni, *Urban Operations Research,* Prentice Hall, Englewood Cliffs, NJ (1981).

[11] P. B. Mirchandani, Shortest distance and reliability of probabilistic networks. *Comp. Oper. Res.* **3** (1976) 347–355.

[12] P. B. Mirchandani and H. Soroush, Optimal paths in probabilistic networks: A case with temporary preferences. *Comput. Oper. Res.* **12**(4) (1985) 365–381.

[13] P. B. Mirchandani, and H. Soroush, Routes and flows in stochastic networks. *Stochastics in Combinatorial Optimization.* (G. Andreatta, F. Mason, and P. Serafini, Eds.). World Scientific, Singapore (1987).

[14] C. H. Papadimitriou, Games against nature. *J. Comput. Syst. Sci.* **3** (1985) 288–301.

[15] C. H. Papadimitriou and J. N. Tsitsiklis, The complexity of Markov decision processes. *Math. Oper. Res.* **12**(3) (1987) 441–450.

[16] C. H. Papadimitriou and M. Yannakakis, Shortest paths without a map. *Proc. ICALP* (1989) 611–620.

[17] G. H. Polychronopoulos, Stochastic and dynamic shortest distance problems. PhD Thesis, Operations Research Center, M.I.T. (1992).

[18] H. N. Psaraftis and J. N. Tsitsiklis, Dynamic shortest paths with Markov arc costs. *Oper. Res.* **41**(1) (1993) 91–101.

[19] L. G. Valiant, The complexity of enumeration and reliability problems. *SIAM J. Comput.* **8** (1979) 410–421.

Received December 1993
Accepted August 4, 1995