Dynamic Traveling Salesman Problem in Stochastic-State Network Setting for Search-and-Rescue Application

David Fajardo and S. Travis Waller

The problem presented in this paper was motivated by the need for a solution to be used in a search-and-rescue application and is formulated as a dynamic traveling salesman problem in a stochastic-state network setting. This problem formulation features a full-recourse decision framework and stochastic demands that are revealed only through direct observation. This problem is defined in a stochastic-state network setting, which allows the modeling of implicitly correlated demand stochasticity. The problem is then formulated as a Markovian decision process, and, finally, a heuristic solution is provided. The heuristic solution is based on a two-stage stochastic program with recourse solved on a set of aggregated networks generated by the use of an aggregating function. Subsets of the feasible solutions obtained at each stage are fixed, and the heuristic is used iteratively to further refine the routing policy.

Recent natural disasters have highlighted the need for efficient emergency response operations. Hurricanes Katrina and Rita exposed many flaws in U.S. emergency response plans, and no aspect received more attention than the delays of response to search-and-rescue missions. Soon after the hurricanes, U.S. military leaders requested the creation of a national search-and-rescue plan, recognizing that both natural disasters and terrorist attacks may occur at any moment, leaving search-and-rescue task forces with little time to react (*1*).

An emergency response plan consists of many different operations, which can be analyzed at various levels. Evacuation, search, rescue, delivery of goods, and rebuilding are all critical parts of an emergency response action. An optimal emergency response plan would allocate resources to each of these operations to minimize the impact of an emergency on society. However, because of the large amount of uncertainty and the large number of variables involved in an emergency operation, modeling and determination of the solution to such an allocation problem becomes difficult. As a result, a way to solve the related subproblems as efficiently and accurately as possible must be found.

Search and rescue is one of the subproblems of an emergency response operation. The search-and-rescue problem consists of the detection and rescue of victims whose locations may or may not be known a priori. A proper search-and-rescue plan will assign the appropriate number of search-and-rescue units, the necessary resources to allow the timely rescue of victims, and an efficient and flexible plan for units to follow.

The focus of this paper is the development and determination of the solution to a simplified network-based mathematical model of the search-and-rescue problem that allows description of the characteristics of the problem that have not been explored in the past, namely, the way in which information affects the dynamics of a rescue operation. The two main features that are included in this model are referred to in the literature as "online information" and "late information."

Under an online information setting, problem information is assumed to be disclosed throughout the problem and thereby affects the decisions that are made. The first, key aspect of a problem formulation that allows consideration of online information is the ability of the decision-making process to account not only for information gathered en route but also for the possibility of gathering information in the future. When one explicitly accounts for information that is known to be obtained in the future, better decisions can be made in the present.

The second feature, which is not mutually exclusive to the concept of online information, is the late information setting (*2*). In this setting, some problem variables are assumed to be unknown until they are directly observed or until variables have been observed indirectly as part of the decision-making process. In the case of this problem, the late information assumption refers to the node demands; that is, the number of units to be rescued at each node is not known until a routing unit visits the node itself or the scenario set is reduced to reveal that number. In contrast, a problem that does not account for this late information feature would be one in which the information is revealed at preestablished points that are not affected by the intermediate decisions.

The problem is framed as a traveling salesman problem (TSP), in which the presence of victims at each node can be modeled as part of what is referred to here as a stochastic-state network (SSN). This modeling choice creates an implicit correlation between these demand variables that will affect the routing policies developed. The problem is formulated as a Markovian decision process (MDP). Because of the complexity of the problem, a heuristic based on a two-stage stochastic programming approximation of the problem solved on an aggregated set of network states is provided.

The following problem is addressed in this paper: given a single vehicle and an SSN, determine the optimal online routing policy to minimize the total expected routing costs. An "online policy" is defined to be the set of routing decisions conditional on the information acquired at each stage of the problem.

School of Civil and Environmental Engineering, University of New South Wales, Sydney, New South Wales 2052, Australia. Corresponding author: D. Fajardo, davidfajardo2@gmail.com.

Transportation Research Record: Journal of the Transportation Research Board, No. 2283, Transportation Research Board of the National Academies, Washington, D.C., 2012, pp. 122–130.
DOI: 10.3141/2283-13

LITERATURE REVIEW

The search-and-rescue problem belongs to a more general class of problems known in the logistics literature as vehicle routing problems (VRPs). Of particular interest to the topic evaluated in this paper are variations of TSP and VRP that account for dynamic decisions and stochasticity, especially in the presence or absence of customers.

Stochastic Vehicle Routing

The stochastic VRP is a more general case of the VRP in which parameters of the problem are given as random variables. As uncertainty is introduced into the problem, variations of the problem are defined by the objective, which is defined according to these stochastic variables; the corrective or recourse actions available in the face of uncertainty; and the set of parameters that are stochastic.

Gendreau et al. provide a classification of the different work in the stochastic VRP literature (*3*). In particular, uncertainty is represented as stochastic demands, stochastic travel times, and stochastic customers. A stochastic VRP with stochastic customers can be viewed as a specific instance of a stochastic demand problem in which demands can take on a value of 0. However, in the literature, a distinction is made between these two types of problems. Problems in which stochastic demands can take on a value of 0 and thus allow the customer to be skipped under certain circumstances are referred to as VRPs with stochastic customers and demands. Of particular relevance to the research presented in this paper are those problems that deal with stochastic customers and, in particular, those in which recourse actions are permitted.

From a general perspective, the stochasticity in the parameters can affect the solution structure in two ways: by changes to a vehicle's ability to satisfy customer demand and by changes to the time that it takes a vehicle to complete its route. Under circumstances in which the stochasticity introduced into the problem affects the feasibility of a priori routes, recourse may need to be allowed to define proper solution structures. In general, simple recourse strategies, such as replenishment when capacity is reached or skipping of customers if they are not present, have been the focus of the literature. This focus on simple recourse strategies is due to both the computational simplicity relative to general recourse structures and the ease of implementability of such strategies.

These recourse actions, along with other stochastic elements such as travel times, can also affect the completion time of routes. This change in completion times requires the definition of an appropriate objective function, as the objective functions of deterministic VRPs are not always applicable. Variations included in the literature mostly deal with minimization of the expected routing cost, although some work has considered chance constrained formulations, especially in VRPs with stochastic travel times. Uncertainty in travel and service times for the VRP was first introduced by Lambert et al. (*4*) and Laporte et al. (*5*), respectively. A capacitated version of the same problem was considered by Kenyon and Morton (*6*).

Uncertainty in customer demands was first considered by Tillman (*7*) and was subsequently considered by Jaillet (*8*) and Bertsimas (*9*). Jaillet examined what he refers to as the probabilistic TSP (*8*). This probabilistic TSP is defined in two stages: in the first stage, a Hamiltonian tour through all nodes in the network must be chosen. In the second stage, the sequence of nodes corresponds to the tour developed in the first stage, but any customers not present under the specific realization of the demand will be skipped. The objective is then to find the optimal first-stage Hamiltonian tour that minimizes the expected cost of the routing policy.

In many VRP formulations, an equivalent multistage stochastic program in which the stages can be defined a priori can be written; that is, the number of stages is not a function of the decision variables. Kenyon and Morton summarized the work done in stochastic VRPs from a stochastic programming perspective (*10*). Gendreau et al. generalized the probabilistic TSP, extending it to the case in which the vehicle is capacitated and the demands at each node are not just binary (*11*). Laporte et al. developed a solution framework for the problem based on a stochastic programming approach, formulating the problem as a two-stage stochastic program with recourse and using branch-and-cut methods to solve the problem to optimality (*12*). Dror et al. examined two stochastic programming formulations in the literature for the stochastic VRP with stochastic demands and proposed a new stochastic programming formulation, along with an MDP formulation (*2*). The focus of the formulations presented is quantification of the cost that occurs when the capacity on a route is exceeded under the uncertain realizations of demand at each node. The MDP formulation itself is fairly generic: it incorporates general concepts of stochasticity by defining generalized states, with no specific mention of solution approaches other than recognition that classical MDP methods are ill suited to solve instances of that problem because of the curse of dimensionality, that is, the growth in complexity of MDPs as a function of the number of states and possible control policies.

Dynamic Vehicle Routing

The literature defines the dynamic vehicle routing class of problems as the problem of optimization of a routing operation in which solutions are a function of the information received or acquired over time. Time-varying or time-dependent parameters may not necessarily imply a dynamic VRP, as the solution to dynamic VRPs may be determined a priori. For example, if the objective of a time-varying problem is the set of routes that minimizes the expected cost over all possible scenarios, then the problem is considered static, even though the parameters themselves are dynamic. However, dynamic and stochastic problems are closely related; the probabilistic nature of certain parameters often dictates the dynamic nature of the solution.

Psaraftis provided some insight into the qualities of a problem that indeed make it a dynamic VRP (*13*). Malandraki and Daskin formulated variations of the TSP and VRP in which the link costs are known step functions of time (*14*). Both problems are formulated as integer programs, and heuristics are proposed. The concept of asymmetry of tours is introduced in this paper. Although in the deterministic version of both problems symmetric tours have equal cost, for more complex variations, the direction in which the tour is followed affects the cost of the solution. Bertsimas and Van Ryzin considered a routing problem in which demands are revealed randomly over a bounded Euclidian plane (*15*). Haghani and Jung presented a formulation for a routing problem with real-time service requests and real-time variations in travel times between demands nodes that is solved by the use of genetic algorithms (*16*). Jaillet and Wagner examined an online VRP in which demands are randomly revealed over time on a Euclidean plane (*17*). The authors develop a competitive-ratio algorithm.

MATHEMATICAL FORMULATION

This section focuses on a formal mathematical formulation of the SSN TSP with full recourse.

Stochastic-State Network

Let a network G equal (N, A, c, d), where N is the set of nodes, A is the set of links, $c{:}A \rightarrow R$ is the link cost function, $d{:}N \rightarrow R$ is the node demand function and R is the set of real numbers. Let Ω be the set of all possible networks. Define a stochastic-state network SSN (Ω, P) as a probability space $(\Omega, E = \{H{:}H \subseteq \Omega\}, P)$, that is, the probability space in which the sample space is equal to Ω, the set of events is equal to all subsets of Ω, E is the set of all events, and P probabilities are assigned to E.

The focus of attention in this paper is on an SSN such that a finite number of outcomes with positive probability exists and P is therefore a discrete probability distribution with finite support. An SSN is intuitively a discrete, finite set of networks $\omega \in \Omega$, each of which occurs with probability $P(\omega)$. This stochastic-state network structure can be used to introduce demand stochasticity in the form of discrete demand scenarios, which are formalized below.

Variable Definition

This section defines the variables needed to formulate the problem as an MDP. Define a stochastic-state network $\text{SSN}_d(\Omega_d, P)$, where the set Ω_d is a finite set of elements ω equal to (N, A, c, d^ω), where N is the set of nodes, A is the set of links, c is the link cost vector, and d is the demand vector under scenario ω. Assume that N, A, and c are the same across all network states.

MDP Formulation

To model the present search-and-rescue problem as an MDP, three main concepts must be defined: state, control policy, and transition cost.

State

Define a state k, s^k, which is equal to $(i^k, \Omega^k, \gamma^k)$, to be the combination of three variables: the current position of the routing unit, i^k; the current subset of feasible demand scenarios Ω^k, where $\Omega^k \subseteq \Omega$; and the current demand state at each node, γ^k, which is equal to 0 if the node has already been visited or if its demand is 0 under all remaining feasible demand scenarios and 1 otherwise. Define the initial state s^0, which is equal to $(0, \Omega, \vec{1})$; that is, the routing unit is at the origin; the full set of demand scenarios is feasible; and all nodes may need to be visited, which is represented by the vector $\vec{1} = \{1, 1, \ldots, 1, 1\}$. Define a unique terminal state with a cost of 0 as $s^f = (0, \omega^f, \vec{0})$; that is, a state in which the routing unit is at the origin; some subset of scenarios is feasible (which can be shown to have a cardinality of 1 if duplicate demand scenarios are not allowed); and no nodes that need to be visited, represented by the vector $\vec{0} = \{0, 0, \ldots, 0, 0\}$, remain.

The set of feasible scenarios Ω^k represents the set of scenarios that is consistent with the information that has been gathered through all previous stages of routing up to stage k. For instance, assume that the demand values observed at the first three nodes visited are 0, 1, and 0, respectively. If this assumption is true, the set of feasible scenarios Ω^3 will contain all demand scenarios such that the demands of those nodes are consistent with the observed values and will exclude all scenarios such that the demand value for any of the nodes visited differs from the observed value.

Control Policy

Define a control policy μ as a mapping from each state s^k to a node $i \in N$. That is, for every state, the control policy determines the next node to be visited. This solution structure is a manifestation of the concept of online information, as the information gathered en route directly affects every decision made; the decisions not only adapt to the information gathered, but the decisions made account for the information that may be gathered in the future.

Given a state s^k and a control policy $\mu(s^k)$, transitions to at most two states with nonzero probability will exist:

- Transition node. If $\gamma^k(\mu(s^k)) = 1$, that is, the node may need to be visited under one of the remaining feasible demand scenarios, transitions to two states with nonzero probabilities are possible and correspond to encountered demands of 0 or 1 at $\mu(s^k)$. Given a state s^k and control policy μ, let these states be $s_0^{k+1}(s^k, \mu)$ and $s_1^{k+1}(s^k, \mu)$, respectively:

$$s_\alpha^{k+1}\left(s^k, \mu\right) = \left(\mu\left(s^k\right), \Omega_\alpha^{k+1}, \gamma_\alpha^{k+1}\right) \tag{1}$$

where α represents the outcome at the transition node, that is, either 0 or 1, and where

$$\Omega_\alpha^{k+1} = \left\{\omega \in \Omega^k : d^\omega\left(\mu\left(s^k\right)\right) = i\right\} \tag{2}$$

$$\gamma_\alpha^{k+1}(i) = \begin{cases} \min\left\{\gamma^k(i), \max\limits_{\omega \in \Omega_\alpha^{k+1}}\left\{d^\omega(i)\right\}\right\} & i \neq \mu\left(s^k\right) \\ 0 & i = \mu\left(s^k\right) \end{cases} \tag{3}$$

That is, the new set of feasible scenarios for $s_0^{k+1}(s^k, \mu)$ and $s_1^{k+1}(s^k, \mu)$ is the set of scenarios under which the demands at node $\mu(s^k)$ are 0 and 1, respectively. The value of γ for node i is changed from 1 to 0 if the node has a 0 demand for all remaining scenarios or if it has already been visited and remains at its original value otherwise. The transition probability into each state is equal to

$$P\left(\Omega_i^{k+1}\right) = \sum_{\omega \in \Omega_i^{k+1}} P(\omega).$$

- Nontransition node. If $\gamma^k(\mu(s^k)) = 0$, that is, the node does not need to be visited under any remaining scenario or it has already been visited, a transition into the following state will occur with a probability of 1:

$$s^{k+1}\left(s^k, \mu\right) = \left(\mu\left(s^k\right), \Omega^k, \gamma^k\right) \tag{4}$$

Thus, a state with the same set of demand scenarios and node demand variables but with a node location variable equal to $\mu(s^k)$ will exist.

Transition Cost

The transition cost at each stage is equal to the cost of link $(i, \mu(s^k))$, that is, the cost of traveling on the link between the current node and the node chosen through the chosen control policy. This cost can be taken to be the travel time in more straightforward routing applications, but the formulation can be used to represent any constant link cost.

Solution Structure Properties

Several aspects of this solution structure are of interest in the analysis of the problem:

- The solution to online problems is not a single tour but, rather, a family of tours that is dependent on the demand events encountered during the routing procedure. Alternatively, the routing policy can be described as a binary decision tree.
- Although the problem has several routing stages, the number of routing stages is dependent on the realization of demands and the routing decisions themselves.

These properties are further explored in this paper in the determination of appropriate solution methods.

EXACT SOLUTION ALGORITHM

To solve the MDP formulation proposed in the previous section, standard MDP techniques such as dynamic programming can be used. Given the initial and terminal states defined above, the objective is to determine an optimal control policy for each state with nonzero probability that is reached from the start of the initial state.

What follows is the dynamic programming algorithm. The recursion function can be defined as

$$\min_{\mu} V(0, \Omega, 1) \tag{5}$$

where

$$V(s^k) = \min_{\mu(s^k) \in N} c(i^k, \mu(s^k)) + \begin{cases} \sum P_{\alpha}^{k+1} V(s_{\alpha}^{k+1}(s^k, \mu)) & \text{if } d^k(\mu(s^k)) = 1 \\ V(s^{k+1}(s^k, \mu)) & \text{if } d^k(\mu(s^k)) = 0 \end{cases} \tag{6}$$

where

$$P_{\alpha}^{k+1} = \sum_{\forall h: \omega \in \Omega_{\alpha}^{k+1}} P(\omega) \tag{7}$$

Furthermore, define the set of terminal states, each with a cost of 0:

$$V(0, \omega^F, 0) = 0 \tag{8}$$

where ω^F is the final set of scenarios. Each terminal state has three elements: the first element is the position of the vehicle, which must be 0, that is, the origin; the second element is the final, true realization of the demand vector; and the final element is the demand left at each node, which must be 0, as all nodes with positive demand must be visited. The reason why the second element consists only of the true demand vector is because all network states are different and all demands for each instance of the problem must be satisfied. As such, once all demands have been satisfied, the true value for all nodes must have been revealed either directly or indirectly. Furthermore, because of the structure of the terminal states, exactly $|\Omega|$ possible terminal states exist.

Properties of MDP

This section discusses the properties of the MDP formulation that allow a more efficient solution algorithm to be developed.

Assume a complete network with nonnegative link costs. Furthermore, assume triangular link costs, that is, $c(i, j) \le c(i, k) + c(k, j)$. Although this may seem restrictive, one can transform any network with nonnegative costs to satisfy this requirement by setting $c(i, j)$ to be the cost of the shortest path between i and j, which is a reasonable conversion in networks that represent physical distances.

These two assumptions, nonnegativity and triangular link costs, allow several important problem properties to be established.

Lemma 1. For any states equal to $(i, \Omega^*, 0)$, such that $i \neq 0$ and $|\Omega^*| = 1$, an optimal control policy, μ^*: $\mu^*(s) = 0$, exists.

Proof. Assume an optimal control policy exists such that $\mu^{**}(s)$ is equal to $j \neq 0$. The lemma can be proved by construction of a feasible solution of equal or lower cost to this control policy. Because all remaining transitions are given by Equation 4, the only terminal state that can be reached from state s is $s_f = (0, \Omega^*, 0)$. As such, any control policy that does not arrive at state s_f with a probability of 1 will have an unbounded cost and cannot be optimal. Therefore, for any optimal control policy, $V(i, \Omega^*, 0) = c(i, j) + C + c(l, 0)$, where C represents the cost of all transitions between state s and state s_l, which is equal to $(l, \Omega^*, 0)$.

Consider control policy μ^*: $\mu^*(s) = 0$. The recursive function value $V(s) = c(i, 0)$, as $V(0, \Omega^*, 0) = 0$. Since $c(i, j) + C + c(l, 0)$ represents the cost of a path between i and 0 and $c(i, 0)$ represents the cost of the shortest path between i and 0 because of the triangular cost structure assumption, $V_{\mu^*}(s) = c(i, 0) < c(i, j) + C + c(l, 0) = V_{\mu^{**}}(s)$. ■

Lemma 2. An optimal control policy that at each state s^k : $|\Omega^k| > 1$ chooses only nodes j : $\gamma^k(j) = 1$ exists.

Proof. Consider an optimal policy μ^* : $\exists s^k$: $\gamma^k(\mu(s^k) = j) = 0$, which implies a transition to state $s^{k+1} = (j, \Omega^k, \gamma^k)$, given by Equation 5. Let the routing decision at this state be $l = \mu(s^{k+1})$. Two cases are possible:

$$\gamma^k(l) = 0, \text{ in which case } V(s^{k+1}) = v$$

$$\gamma^k(l) = 1, \text{ in which case } V(s^{k+1}) = c(j, l) + P(\Omega_0) V(l, \Omega_0, \gamma_0) + P(\Omega_1) V(l, \Omega_1, \gamma_1)$$

where $\Omega_{\alpha} = \omega \in \Omega^k$: $\omega(l) = \alpha$ and

$$d_{\alpha}(i) = \begin{cases} \min\{d^k(i), \max_{\omega \in \Omega_{\alpha}} \{\omega(i)\}\} & i \neq l \\ 0 & i = l \end{cases} \tag{9}$$

Because no transition occurs at node j, the probability distribution of states at node l is given by the feasible set of scenarios at state s^k and the vector d is updated accordingly. As such, $V(s^k) = c(i, j) + V(s^{k+1})$, and expansion of the second term of the right-hand side gives

$$V(s^k) = c(i,j) + c(j,l) + \begin{cases} V(l, \Omega^k, d^k) & \text{if } d^k(l) = 0 \\ P(\Omega_0)V(l, \Omega_0, d_0) + P(\Omega_1)V(l, \Omega_1, d_1) & \text{if } d^k(l) = 1 \end{cases} \quad (10)$$

To establish a proof of the lemma, a feasible solution μ** can be constructed by bypassing node j, which is of equal or lower cost than μ*. Let μ**(s^k) equal l; that is, nontransition node j and state s^{k+1} are bypassed in the path between i^k and l. In this case, the recursive function at state s^k is given by

$$V(s^k) = c(i,l) + \begin{cases} V(l, \Omega^k, \gamma^k) & \gamma^k(l) = 0 \\ P(\Omega_0)V(l, \Omega_0, \gamma_0) + P(\Omega_1)V(l, \Omega_1, \gamma_1) & \gamma^k(l) = 1 \end{cases} \quad (11)$$

The expressions under policies μ* and μ** differ only in the transition costs. Because of the triangular cost assumption, $c(i, l) \le c(i, j) + c(j, l)$, the cost of policy μ* is less than or equal to the cost of μ**, which implies that μ** is also optimal. ■

This lemma allows a simplified version of the recursive function to be written:

$$V(s^k) = \min_{\mu(s^k) \in N: \gamma^k(\mu(s^k))=1} c(i^k, \mu(s^k)) + \sum_{\alpha=0,1} P(\Omega_\alpha^{k+1}) V(s_\alpha^{k+1}) \quad (12)$$

where

$$V(0, \omega^F, 0) = 0 \quad (13)$$

Lemma 3. For a state $s^k : |\Omega^k| = 1$, the value of the recursive function $V(s^k)$ is equal to the minimum-cost Hamiltonian path on the graph created by use of only the vertices $j \in N : \gamma^k(j) = 1$ with origin i^k and destination 0.

Proof: In the case of a state in which only one demand scenario is possible, the expression

$$\gamma_\alpha^{k+1}(i) = \begin{cases} \min\{\gamma^k(i), \max_{\omega \in \Omega_\alpha^{k+1}} \{d^\omega(i)\}\} & \text{if } i \ne \mu(s^k) \\ 0 & \text{if } i = \mu(s^k) \end{cases} \quad (14)$$

can be rewritten as

$$\gamma_\alpha^{k+1}(i) = \begin{cases} \gamma^k(i) & i \ne \mu(s^k) \\ 0 & i = \mu(s^k) \end{cases} \quad (15)$$

That is, because the set of scenarios cannot be further narrowed, the vector γ will change only if a node, $i : \gamma(i) = 1$, is visited, and only $\gamma(i)$ will be changed. As such, to generate a feasible solution, that is, one that arrives at a terminal state, all nodes $i : d^k = 1$ must be visited. Furthermore, as was shown previously, an optimal policy exists, such that nodes $j : \gamma^k = 0$ are not visited. Therefore, the optimal policy needs to visit a node i if and only if $\gamma^k(i) = 1$. ■

Determination of a solution to meaningful problems by use of the exact algorithm is infeasible because of computational complexity. As such, in the next section the focus is shifted to development of a heuristic for the problem on the basis of establishment of feasible solutions with upper bounds derived from a stochastic programming approach.

HEURISTIC

The heuristic presented in this section is based on the concept of use of an aggregating function to obtain a simplified problem that can generate a feasible solution. This approach is used recursively to develop a routing policy that at each stage chooses the lowest upper-bound subroute.

Problem Properties

This section establishes problem properties that will aid with building toward the stochastic programming formulation for the subproblem that is the basis for the heuristic presented in this section.

Lemma 4. Consider a state $s^k = (i^k, \Omega^k, d^k)$. Let ω^{max}: $d^{\omega^{max}}(i) = \max_{\omega \in \Omega^k} \omega(i)$. The optimal route that is the solution to the problem arg min $V(s^{max})$, where $s^{max} = (i^k, \omega^{max}, d^k)$, is a feasible solution to the original problem and, as such, is an upper bound to the original problem.

The demand scenario ω^{max} corresponds to a single scenario aggregated over all remaining scenarios in which, if node j must be visited under any remaining feasible scenario, it must be visited in scenario ω^{max}. Therefore, the optimal solution to the subproblem with a single feasible demand scenario remaining consists of a route that satisfies the node demands for all remaining nodes under all remaining scenarios. Because it satisfies the demands under all scenarios, it is a feasible solution; because it is not guaranteed to be optimal for the original problem, it constitutes an upper bound.

Corollary 1. Consider a transition node j reached from state s^k, and replace the set Ω^k with two demand scenarios, S_0 and S_1, such that

$$S_0 : d^{S_0}(i) = \max_{\omega \in \Omega^k : d^\omega(j) = 0} d^{\omega(i)} \quad (16)$$

$$S_1 : d^{S_1}(i) = \max_{\omega \in \Omega^k : \omega(j) = 1} \omega(i) \quad (17)$$

$$P(S_0) = \sum_{\omega \in \Omega^k : \omega(j) = 0} P(\omega) \quad (18)$$

$$P(S_1) = \sum_{\omega \in \Omega^k : \omega(j) = 1} P(\omega) \quad (19)$$

The optimal solution to this new problem is a feasible solution to the original problem and as such provides an upper bound.

The solution to this modified problem consists of two routing stages: a path between nodes i and j and a second-stage path that corresponds to either demand scenario S_0 or demand scenario S_1. Because of the recursive nature of the problem, the problem of determination of the optimal second-stage paths given a specific first-stage path has the same property derived in Lemma 4, and so the optimal route for a single aggregated state is a feasible solution to the problem. Therefore, the overall routing strategy is feasible as well and constitutes an upper bound.

Corollary 2. The solution to $\arg\min_\mu V(s^{\max})$ is the minimum-cost Hamiltonian path that visits all the nodes ($j : \gamma^k(j) = 1$.

As was established in Lemma 3, the optimal solution to the subproblem in which the cardinality of the set of feasible demand scenarios is 1 is the least-cost Hamiltonian path in the network that comprises the origin equal to i^k, the destination equal to 0, and all nodes j such that $d(j) = 1$.

Stochastic Programming Formulation for Two-Scenario States

This section presents a two-stage stochastic program with recourse formulation for the two-scenario-state problem, in which the first-stage decision variable is the route to be followed to the transition node and the second-stage variables correspond to the route from the transition node back to the origin for each of the two possible outcomes at the transition node. Consider a state $s^k = (i^k, \Omega^k, d^k)$ such that $|\Omega^k| = 2$. The optimal solution to the subproblem $\arg\min_\mu V(s^k)$ can be found by solution of the following integer program where SSTSP stands for the stochastic state traveling salesman problem:

$$\text{SSTSP}(s^k, t) = \min_{x, y_\omega} \sum_{(i,j)\in A} c(i,j)x(i,j) + \sum_{\omega\in\Omega^k} P(\omega)c(i,j)y_\omega(i,j) \tag{20}$$

such that

$$\sum_j x(i^k, j) = 1 \tag{21}$$

$$\sum_j y_\omega(j, 0) = 1 \qquad \forall\omega \in \Omega^k \tag{22}$$

$$\sum_j x(j, t) = 1 \tag{23}$$

$$\sum_j y_\omega(t, j) = 1 \qquad \forall\omega \in \Omega^k \tag{24}$$

$$\sum_j y_\omega(i, j) = 1 \qquad \forall\omega \in \Omega^k, i \in S_\omega \tag{25}$$

$$\sum_j y_\omega(j, i) = 1 \qquad \forall\omega \in \Omega^k, i \in S_\omega \tag{26}$$

$$\sum_j x(i, j) + y_\omega(i, j) = 1 \qquad \forall\omega \in \Omega^k, i \in R \tag{27}$$

$$\sum_j x(j, i) + y_\omega(j, i) = 1 \qquad \forall\omega \in \Omega^k, i \in R \tag{28}$$

$$\sum_j x(i, j) - \sum_j x(j, i) = 0 \qquad \forall\omega \in \Omega^k, i \in R \tag{29}$$

$$\sum_j y_\omega(i, j) - \sum_j y_\omega(j, i) = 0 \qquad \forall\omega \in \Omega^k, i \in R, y_\omega \in \{0, 1\} \tag{30}$$

$$\sum_{i\notin S}\sum_{j\notin S} x(i, j) + y_\omega(i, j) \geq 2 \qquad \forall S \subset \{S_\omega \cup R \cup i^k \cup 0\} \tag{31}$$

$$\sum_{i\in S}\sum_{j\notin S} x(i, j) + y_\omega(i, j) \geq 2 \qquad \forall S \subset \{S_\omega \cup R \cup i^k \cup 0\} \tag{32}$$

where

$$N' = \{i : i \in N, d^k(i) = 1\} \tag{33}$$

$$R = \{i : i \in N', \omega(i) = 1, \forall\omega \in \Omega^k, i \neq t\} \tag{34}$$

$$S_\omega = \{i : i \in N', \omega(i) = 1, i \notin R, i \neq t\} \tag{35}$$

The set N' represents the set of nodes that may need to be visited under at least one of the remaining feasible demand scenarios. The set R represents the set of nodes that must be visited under both scenarios, whereas the set S_ω represents the set of nodes that needs to be visited only under demand scenario ω.

Equation 21 ensures that one link departs from the origin in the first stage, whereas Equation 22 ensures that one link will arrive at the destination under each demand scenario. Equation 23 ensures that one link will enter the transition node in the first stage, whereas Equation 24 guarantees that one link will leave the transition node under each demand scenario. Equations 25 and 26 guarantee that each node that needs to be visited only under a particular demand scenario will be visited as part of that scenario's second-stage route. Equations 27 and 28 guarantee that each node in R will be visited in either the first or the second stage for each demand scenario, whereas Equations 29 and 30 are the flow balance constraints in x and y_ω. Finally, Equations 31 and 32 prevent the formation of subtours in the solution in that they ensure that every subset of nodes has one link entering it and one link leaving it.

This mathematical programming formulation intuitively finds a minimum-cost solution combination of a first-stage route from node i^k to node t by use of nodes only in R and a set of second-stage routes from t to the origin, one for each possible demand scenario, that visits nodes only in R and S_ω. The set of variables $x(i, j)$ represents the first-stage routing decision, that is, the decision to travel from i to node j, whereas the set of variables y_ω represents the second-stage routing decisions under the now-known demand scenario.

The reasoning behind the formulation is that once the first transition node is encountered, the demand scenario becomes known. Once this demand scenario becomes known, the optimal solution is the Hamiltonian path between the transition node t and the origin 0. As such, the optimal combination of a single path to the transition node and one path from the transition node to the origin for each possible demand scenario must be found.

HEURISTIC ALGORITHM

This section outlines the algorithm for the heuristic based on the use of the two-stage stochastic program described in the previous section. A summarized outline of the algorithm is first provided, and then an in-depth description of the algorithm is presented.

Algorithm Outline

The algorithm is based on three main steps: aggregation of demand scenarios, solution of a two-stage stochastic program, and use of the first-stage routing solution of the stochastic program as a subpath for the main solution. The formal definition of the algorithm follows.

Main recursion:
$V(\iota, \Omega, \gamma)$
Inputs: root node ι, network states Ω, demand state vector γ
optimalValue = INF
for all transition nodes i do
  aggregate$(i, \Omega, \gamma) \rightarrow S_0$ and S_1
  stochasticProgram$(s = (i, \{S_0, S_1\}, \gamma)), t) \rightarrow$ path $= x$
  $\alpha = \text{cost}(x)$
  for all $j \in N, j \neq i$ do
    $\gamma_0(j) = \min\{\gamma(j), S_0(j)\}$
    $\gamma_1(j) = \min\{\gamma(j), S_1(j)\}$
  end for
  for all $j \in x$ do
    $\gamma_0(j) = 0$
    $\gamma_1(j) = 0$
  end for
  $\beta[0] = V(\iota = t, \Omega = S_0, \gamma_0)$
  $\beta[1] = V(\iota = t, \Omega = S_1, \gamma_1)$
  if alpha + $P(S_0)\beta[0] + P(S_1)\beta[1]$ < optimalValue then
    optimalValue ← alpha + $P(S_0)\beta[0] + P(S_1)\beta[1]$
    $\mu(\iota, \Omega, \gamma) = x$
  end if
end for
Output: optimalValue

The following sections focus on the specific details on the implementation of the three main sections of the algorithm.

Demand Aggregation

The demand aggregation step generates two demand scenarios, S_0 and S_1, given a transition node i, a set of feasible demand scenarios Ω, and a set of demand states d. All states with equal demand values for the transition node states S_0 and S_1 are aggregated. The procedure is simple: group demand scenarios such that $\omega(i) = 0$ into the set Ω_0, and group demand scenarios such that $\omega(i) = 1$ into the set Ω_1. Then, generate the aggregated demand scenarios according to the following relationship:

$$S_\beta(j) = \begin{Bmatrix} \max_{\omega \in \Omega_\beta} \omega(j) & \text{if } d(i) = 1 \\ 0 & \text{if } d(i) = 0 \end{Bmatrix} \quad (36)$$

The relationships presented above aggregate the scenarios in the sets Ω_0 and Ω_1 by use of a procedure that takes the maximum value for each node among all elements of the set for nodes $i : \gamma(i) = 1$ and assignment of a value of 0 otherwise. As indicated in Corollary 1, solution of the problem in which the demands are aggregated by the procedure represented by Equation 36 provides a feasible solution to the problem and an upper bound. The following section discusses the stochastic program used to solve the problem.

Stochastic Program

The stochastic program to be solved is based on the formulation developed earlier in this paper, in which the inputs are given as the state information, that is, the current node, the two remaining feasible demand scenarios, and the demand state vector. The output taken from this procedure is the chosen transition node, as well as the path from the current node to the transition node.

The integer programming formulation described in the previous section includes subtour elimination constraints, the number of which grows combinatorially according to the size of the network, which makes solution of the problem directly infeasible. As such, an iterative subtour elimination procedure based on the branch-and-bound algorithm developed elsewhere for deterministic TSPs is used (*18*).

A relaxed version of the problem is solved with no subtour constraints. If no subtours are found, the solution is optimal. If subtours are found, a branching procedure is used to generate new problems in which the smallest subtour is found, and for each link in said subtour, a new problem in which that link is removed from the network is generated. This branch-and-bound procedure eventually leads to a set of feasible solutions, of which the one with the least cost is chosen.

The relaxed version of the problem, RSSTSP, is presented below:

$$\text{RSSTSP}(s^k, t, \text{RL}) = \min_{x, y_\omega} \sum_{(i,j) \in A} c(i, j) x(i, j) + \sum_{\omega \in \Omega^k} P(\omega) c(i, j) y_\omega(i, j) \quad (37)$$

such that

$$\sum_j x(i^k, j) = 1 \quad (38)$$

$$\sum_j y_\omega(j, 0) = 1 \qquad \forall \omega \in \Omega^k \quad (39)$$

$$\sum_j x(j, t) = 1 \quad (40)$$

$$\sum_j y_\omega(t, j) = 1 \qquad \forall \omega \in \Omega^k \quad (41)$$

$$\sum_j y_\omega(i, j) = 1 \qquad \forall \omega \in \Omega^k, i \in S_\omega \quad (42)$$

$$\sum_j y_\omega(j, i) = 1 \qquad \forall \omega \in \Omega^k, i \in S_\omega \quad (43)$$

$$\sum_j x(i, j) + y_\omega(i, j) = 1 \qquad \forall \omega \in \Omega^k, i \in R \quad (44)$$

$$\sum_j x(j, i) + y_\omega(j, i) = 1 \qquad \forall \omega \in \Omega^k, i \in R \quad (45)$$

$$\sum_j x(i, j) - \sum_j x(j, i) = 0 \qquad \forall \omega \in \Omega^k, i \in R \quad (46)$$

$$\sum_j y_\omega(i, j) - \sum_j y_\omega(j, i) = 0 \qquad \forall \omega \in \Omega^k, i \in Rx, y_\omega \in \{0, 1\} \quad (47)$$

$$x(i, j) = 0, \forall x(i, j) \in \text{RL} \quad (48)$$

$$y_\omega(i, j) = 0, \forall y_\omega(i, j) \in \text{RL} \tag{49}$$

where

$$N' = \{i : i \in N, d^k(i) = 1\} R = \{i : i \in N', \omega(i) = 1 \qquad \forall \omega \in \Omega^k, i \neq t\} \tag{50}$$

$$S_\omega = \{i : i \in N', \omega(i) = 1, i \notin R, i \neq t\} \tag{51}$$

Because the only nodes that can have both x and y flows through them are the transition node and the origin (in the case in which the origin is also the destination), all subtours that are formed will form within the same routing stage. For practical purposes, this does affect the efficiency of the algorithm, as breaking of a subtour in one routing stage is not likely to break a subtour in a different stage. The formal definition of the algorithm for solution of the stochastic program follows.

Stochastic program solution:
function: stochasticProgram(s^{in}, t)
Inputs: states^{in}, transitionNode t
RL ← ∅ is the set of removed links
z^* ← ∞ is the optimal cost
x^* is the optimal first-stage path
problemList = {(s^{in}, t, RL)}
while problemList ≠ ∅ do
  Solve RSSRP(s^{in}, t, RL)
  if RSSRP(s^{in}, t, RL) < z^* then
    if no subtours in solution then
      z^* ← RSSRP(s^{in}, t, RL)
      x^* ← first stage path of RSSRP(s^{in}, t, RL)
    or else
      L = smallest cycle or path
      for all links (a, b) ∈ L
        add problem = (s^{in}, t, RL + {(a, b)})
      end for
    end if
  end if
end while
Output: optimalValue = z^*, firstStagePath = x

NUMERICAL RESULTS

To test the performance of the algorithms presented in this paper and because of the extremely high degree of complexity of the exact solution method, the performance of the heuristic is first compared with that of the exact method for small networks. The comparison shows that the heuristic can provide good results for such cases. Results for larger networks solved by use of the heuristic developed earlier are then presented.

To contrast the performance of the exact and heuristic methods, their performance was tested on a set of randomly generated networks. Two sets of networks were tested:

- A complete network with $N = 10$ and $|\Omega| = 5$ and
- A complete network with $N = 10$ and $|\Omega| = 10$.

Tables 1 and 2 compare the performance of the exact method and the heuristic. For the set of networks in which $|\Omega| = 5$ (Table 1), both the solution and computational run times are similar. This similarity of run times can be attributed to the relatively low number of recursive calls in the dynamic program, which allows the stochastic programming subproblem to estimate the true value of the recursion function relatively well. However, because of the low number of stages, the computational efficiency gain is marginal.

TABLE 1 Results for Test Networks with 10 Nodes and Five Network States

Exact		Heuristic		
Value	Run Time (s)	Value	Run Time (s)	Percent from Optimal
58	74	58	54	0
60	74	60	72	0
63	79	63	65	0
64	75	64	58	0
74	80	74	78	0
77	79	77	69	0
79	79	80	51	1.3
79	74	79	90	0
79	77	81	67	2.5
86	75	86	65	0
87	77	87	69	0
87	80	87	64	0
88	79	88	94	0
97	76	97	72	0
101	79	101	53	0
101	76	103	78	2
104	73	104	71	0
135	77	135	61	0

However, as the number of network states is increased to 10 in the second set of networks (Table 2), the computational effort required to solve the problem exactly increases rapidly. Only a small number of networks was examined, as the time required to solve the problem to optimality exceeded 2 h. The heuristic, however, was able to achieve relatively good results with much shorter run times.

Despite the relatively small difference in run times for small instances of the problem, the computational complexity of the exact method is such that an increase in the size of the network past this point makes the problem too computationally expensive to solve. As a frame of reference, for a complete network with 15 nodes and five scenarios, the program causes a stack overflow; that is, memory

TABLE 2 Results for Test Networks with 10 Nodes and 10 Network States

Exact		Heuristic		
Value	Run Time (s)	Value	Run Time(s)	Percent from Optimal
169	52.7	175	3.6	2.2
209	50.8	211	1	2.2
224	56.2	227	1.3	2
207	54.4	219	5.8	1.3
202	53.4	209	3.5	2.7

TABLE 3 Results for Test Networks with 15 Nodes and Five Network States

Value	Run Time (min)	Lower Bound	Upper Bound
71	15.3	42	84
122	21.3	62	125
83	16.8	42	100
86	11.8	32	95
105	13.2	56	115
87	9.7	40	105
78	13.5	30	91
90	12.6	45	105
90	14.4	48	109
108	6.6	62	111
115	16.3	50	125
103	17.1	48	121

is insufficient to continue to evaluate the program. In contrast, the heuristic can solve these problems in relatively reasonable amounts of time, which are presented in Table 3.

Further discussion of the results shown for the set of problems with the 15-node, five-network state is difficult to provide, because in general, the lower bound, which was obtained by generation of a tour that visits only nodes that must be visited under all feasible scenarios, was not tight.

DISCUSSION OF RESULTS AND CONCLUSIONS

The problem presented in this paper addresses issues in the literature that have been ignored from a modeling framework perspective. Furthermore, the heuristic presented in this paper, which exploits the aggregation of the network state to solve an approximation to the problem, allows good feasible solutions to be obtained significantly more efficiently.

The computational complexity of the heuristic is not affected nearly as dramatically by the increase in the number of possible network states, which is of particular importance. This less dramatic effect on computational complexity is a result of the procedure of aggregation of the network state used as part of the heuristic, because it allows the heuristic to limit the effective computational work required to generate a feasible solution.

However, the computational complexity of the problem is such that even a heuristic such as the one presented in this paper will still be tractable only for problems relatively modest in size. As such, the need to develop solution methodologies that can generate feasible solutions in larger networks is recognized. Of particular interest are other forms of aggregation, such as node aggregation; rather than consideration of each node's individual demand, heuristics could be devised by grouping of nodes that share similar demand profiles across network states or development of clusters on the basis of network connectivity properties.

As an alternative, other heuristics can be devised on the basis of decomposition of the problem into different decision stages. For example, an interesting related problem, which falls out of the scope of this research, is the ability to determine routes that allow the true demand realization of the network to be determined. The ability to develop such a set of routes, optimized according to some metric, would allow a reformulation of the problem as a two-stage stochastic problem with recourse, in line with several formulations developed in the literature. In every instance of the problem, regardless of the realization of the network, the problem can be separated into two stages. In the first stage, the real demand realization is determined through some chosen route. In the second stage, the remaining demand is satisfied along minimum-cost routes. The second stage of this problem can be formulated as a deterministic VRP. Although, in general, the true realization of the demand may not be found until the last node is visited, this may happen at an earlier stage of the problem. Although this reformulation of the problem itself may not generate any savings in computational time, it may be possible to devise viable heuristics by use of this problem property to develop a decomposition scheme.

REFERENCES

1. Associated Press. *Military to Bush: U.S. Needs Search-Rescue Plan.* MSNBC, 2005. http://www.msnbc.msn.com/id/9477781/.
2. Dror, M., G. Laporte, and P. Trudeau. Vehicle Routing with Stochastic Demands: Properties and Solution Frameworks. *Transportation Science,* Vol. 23, No. 3, 1989, pp. 166–176.
3. Gendreau, M., G. Laporte, and R. Séguin. Stochastic Vehicle Routing. *European Journal of Operational Research,* Vol. 88, No. 1, 1996, pp. 3–12.
4. Lambert, V., G. Laporte, and F. Louveaux. Designing Collection Routes Through Bank Branches. *Computers and Operations Research,* Vol. 20, No. 7, 1993, pp. 783–791.
5. Laporte, G., F. Louveaux, and H. Mercure. The Vehicle Routing Problem with Stochastic Travel Times. *Transportation Science,* Vol. 26, No. 3, 1992, pp. 161–170.
6. Kenyon, A. S., and D. P. Morton. Stochastic Vehicle Routing with Random Travel Times. *Transportation Science,* Vol. 37, No. 1, 2003, p. 69.
7. Tillman, F. A. The Multiple Terminal Delivery Problem with Probabilistic Demands. *Transportation Science,* Vol. 3, No. 3, 1969, p. 192.
8. Jaillet, P. A Priori Solution of a Traveling Salesman Problem in Which a Random Subset of the Customers Are Visited. *Operations Research,* Vol. 36, No. 6, 1988, pp. 929–936.
9. Bertsimas, D. J. A Vehicle Routing Problem with Stochastic Demand. *Operations Research,* Vol. 40, No. 3, 1992, pp. 574–585.
10. Kenyon, A. S., and D. P. Morton. A Survey on Stochastic Location and Routing Problems. *Central European Journal of Operational Research,* Vol. 9, 2002, pp. 277–328.
11. Gendreau, M., G. Laporte, and R. Séguin. An Exact Algorithm for the Vehicle Routing Problem with Stochastic Demands and Customers. *Transportation Science,* Vol. 29, 1995, pp. 143–155.
12. Laporte, G., F. V. Louveaux, and H. Mercure. A Priori Optimization of the Probabilistic Traveling Salesman Problem. *Operations Research,* Vol. 42, No. 3, 1994, pp. 543–549.
13. Psaraftis, H. N. A Dynamic Programming Solution to the Single Vehicle Many-to-Many Immediate Request Dial-a-Ride Problem. *Transportation Science,* Vol. 14, 1980, pp. 130–154.
14. Malandraki, C., and M. Daskin. Time Dependent Vehicle Routing Problems: Formulations, Properties and Heuristics Algorithms. *Transportation Science,* Vol. 26, No. 3, 1992, p. 185.
15. Bertsimas, D. J., and G. Van Ryzin. A Stochastic and Dynamic Vehicle Routing Problem in the Euclidian Plane. *Operations Research,* Vol. 39, No. 4, 1993, pp. 60–76.
16. Haghani, A., and S. Jung. A Dynamic Vehicle Routing Problem with Time-Dependent Travel Times. *Computers and Operations Research,* Vol. 32, 2005, pp. 2959–2986.
17. Jaillet, P., and M. Wagner. Online Routing Problems: Value of Advanced Information as Improved Competitive Ratios. *Transportation Science,* Vol. 20, No. 2, 2006, pp. 200–210.
18. Carpaneto, G., and P. Toth. Some New Branching and Bounding Criteria for the Asymmetric Travelling Salesman Problem. *Management Science,* Vol. 26, No. 7, 1980, p. 736.

The Transportation Network Modeling Committee peer-reviewed this paper.