Restricted Recourse Strategies for Dynamic Networks with Random Arc Capacities

WARREN B. POWELL

Princeton University, Princeton, New Jersey 08544

LINOS F. FRANTZESKAKIS

AT & T Bell Laboratories, Holmdel, New Jersey 07733-1988

We consider a class of multistage stochastic programming problems that can be formulated as networks with random arc capacities. Large problems have proved intractable using exact methods and hence various approximations have been proposed, ranging from approximating the recourse function to sampling a small number of scenarios to capture future uncertainties. We explore the use of specialized recourse strategies that are not as general as network recourse but nonetheless capture some of the important tradeoffs. These new recourse strategies allow us to develop approximations to the recourse function that can be used to solve problems with thousands of random variables. Given these approximations, classical optimization methods can be used. The concept of hierarchical recourse is introduced and used to synthesize and generalize earlier notions of nodal recourse and cyclic recourse.

Stochastic programming represents a powerful framework for formulating dynamic optimization problems in the presence of forecasting uncertainties. Such problems arise in transportation in the form of dynamic fleet management, dynamic vehicle routing and logistics. In these problems, we are faced with the problem of routing vehicles or managing a fleet in the face of uncertain future demands. For example, we may have to decide whether to hold vehicles in a particular region in anticipation of future demands, or reposition them empty to another region where there is more potential. Problems in logistics require making production and inventory planning decisions in the face of uncertain future demands. Stated formally, these problems often result in large scale stochastic programming problems that are computationally intractable.

The prototypical (two-stage) stochastic programming problem is often written as:

$$\min_x c^T x + \overline{Q}(x) \qquad (1)$$

subject to $Ax = b$, $x \geqslant 0$, where $\overline{Q}(x)$ is the *expected recourse function*, defined by:

$$\overline{Q}(x) = E_\xi\left[\min_y \{q^T y | Wy = \xi - Tx,\ y \geqslant 0\}\right] \qquad (2)$$

Here, x and y represent, respectively, first and second stage decisions and ξ is a vector of random variables. c and q are cost vectors, and T and W determine the effect of first stage decisions on second stage constraints (T is generally referred to as the technology matrix, and W is the recourse matrix). Decisions must be made in the first stage, after which the random vector ξ is realized, at which time we are allowed to find the optimal y in the second stage. Our ability to respond to the random variables is defined as the recourse, and $\overline{Q}(x)$ captures the effects of this recourse. In the applications that we consider, the optimization problems require solving network problems, and hence this is called a problem with *network recourse*. Section 1 reviews specific problems that arise in transportation and logistics that can be formulated within this framework.

The difficulty with problem (1) is that $\overline{Q}(x)$ cannot be written analytically as a function of x. We

Transportation Science
Vol. 28, No. 1, February 1994

0041-1655/94/2801-0003 $01.25

have to solve a network problem for every realization of ξ in the second stage. This process usually makes finding $\overline{Q}$ analytically intractable. Transportation often exhibits random vectors ξ with high dimensionality, creating an exponentially large number of outcomes.

In recent years, several approaches have been developed with tremendous promise in terms of being able to provide at least good approximations to these large problems. This is particularly true in the case of problems such as networks which offer special structure. Even stronger results can be obtained for special types of networks such as transportation problems or transportation networks with random arc capacities. In this research we consider a specific set of stochastic network problems that arise in transportation and logistics that can be formulated as a network with random arc capacities. In all cases, these problems can be written as either a two or N-stage stochastic program with network recourse. We then propose a range of more restrictive recourse strategies that approximate network recourse but which are analytically more tractable. Thus, we might define an approximate function $\hat{Q}(x\,|\,\xi)$ which satisfies:

$$\hat{Q}(x\,|\,\xi) \geqslant \hat{Q}(x\,|\,\xi)$$

where $\hat{Q}(x\,|\,\xi)$ is obtained by heuristically optimizing the conditional recourse function $\overline{Q}(x\,|\,\xi)$. The goal is to find a heuristic so that $\hat{Q}(x\,|\,\xi)$ reasonably approximates $\overline{Q}(x\,|\,\xi)$, and exhibits sufficient structure so that its expectation can be easily found. If this is possible, then we may replace the general recourse function $\overline{Q}(x)$ with an approximation that would allow problem (SP) to be solved using classical means. The specific approach used in this paper is to explore recourse strategies that are simpler than network recourse that allow the recourse problem (1) to be solved in an analytically tractable way.

There is an extensive literature on bounds and approximations to stochastic programs (see, for example, the excellent review by BIRGE and WETS[4]). One approach involves sampling a small number of scenarios that describe future possible outcomes. Using these scenarios, a much larger optimization problem is formulated which recognizes these different outcomes, but forces the model to recommend a *single* decision for the first stage (a process known as scenario aggregation). In practice, the resulting optimization problem can be extremely large, and any network structure that might exist in the original problem is lost. VAN SLYKE and WETS[25] propose an outer linearization scheme for solving this problem. More recently, ROCKAFELLAR and WETS[23] propose a decomposition scheme that may be computationally attractive for large problems with many scenarios. This approach is particularly well suited in strategic planning applications where future uncertainties are often represented using a small set of scenarios, but it raises an important issue for problems where the number of scenarios may be extremely large. At this time, there has been very little theoretical or experimental work analyzing the effects of these sampling problems, but recent research on importance sampling[9] may offer some key insights into this problem.

A second approach involves developing approximations to the recourse function. Simple recourse is one such approximation, whereby complex, multivariate expectations are reduced to a series of expectations involving a single variable. In some applications such as the newsboy problem, simple recourse is an exact model, where the penalties for providing too many or too few newspapers are determined by the parameters of the problem (such as the salvage value of a newspaper or the estimated cost of lost demand). However, simple recourse can be used to approximate more general stochastic programs. BIRGE and WETS[4] use ray approximations to estimate overage and underage penalties for general stochastic programs, replacing them with approximate simple recourse problems. The overage and underage penalties are found by perturbing, in both directions, each random variable individually (if there are m random variables, this requires solving $2m$ linear programs, one for each random variable perturbed in both the positive and negative directions). This idea is further generalized in BIRGE and WETS,[5] which solves the stochastic program by perturbing the random vectors in m dimensions simultaneously, instead of one at a time. As with the ray approximations, $2m$ optimization problems are solved (as opposed to other methods that require a number of problems to be solved that is exponential in the number of random variables). However, general perturbation vectors (rather than unit vectors) create significantly greater computational requirements. BIRGE[6] indicates how this approach can be extended to multistage problems.

WALLACE[26] introduces a piecewise-linear upper bound for stochastic programs with network recourse. This method involves identifying cycles in a network with random arc capacities, and then calculating the effect random perturbations in the upper bounds have on the flow in each cycle. In order to perform these calculations efficiently, however, it is necessary to introduce some strong approximations that have the effect of inducing separability among the random variables in the recourse function. This concept is extended in BIRGE and WALLACE[3] to general linear programs. It is important to emphasize, however, that this work has focussed on developing computable *bounds* on the recourse function for a given set of first stage decisions, as opposed to calculating *functions* which approximate the recourse function. Just the same, the insights behind this work may assist in the development of better approximations.

BEALE et al.[1] suggest a response surface methodology where the recourse function is replaced with a simple quadratic or exponential approximation involving a few parameters. These parameters are fitted statistically using observations derived from repeated sampling. An entirely different approach is based on the idea of stochastic gradients where gradients of the recourse function are sampled and then used to identify search directions (see ERMOLIEV[11] and RUSZCZYNSKI[24]). These methods have the advantage of proven convergence properties, but do not take advantage of the structure of the problem.

The research in this paper considers a specific set of stochastic programs that arises in both logistics and carrier operations. The work draws on prior research in the area by the authors (FRANTZESKAKIS and POWELL[13, 14]) and WALLACE[26] where special recourse strategies were found to yield analytically tractable approximations to the recourse function. These approximations allow large scale stochastic programming problems to be solved using standard techniques. The effect is not unlike using simple recourse to approximate more general stochastic programs. The idea is to replace the minimization problem in (2) with a simpler optimization problem which allows the expected recourse function $\overline{Q}(x)$ to be written analytically as a function of x. For example, FRANTZESKAKIS and POWELL[12] introduced the notion of *nodal recourse* and showed how, for transportation networks with random arc capacities, the transportation problem could be replaced with a much simpler optimization problem that yields a good approximation of the recourse function. The limitation of this work is that it only applies to a very special class of stochastic networks.

In this paper, we introduce a much broader set of nodal recourse strategies, and show how these fit into a more general class of recourse strategies we call *hierarchical* recourse. All of these are special cases of network recourse, but they introduce simplifications that either provide useful approximations to expected recourse functions, or at least provide efficient bounds (along the same lines as WALLACE.[26]) The objective of this research is to introduce and explore these special recourse strategies and to show how more accurate approximations can be developed.

We begin in section 1 by reviewing three particular types of stochastic programs that serve as a basis for discussion. Section 2 outlines the basic strategy for approximating recourse functions by using restricted recourse strategies. Next, section 3 discusses a recently developed class of restricted recourse strategies known as nodal recourse which have been proven effective for dynamic transportation problems with random arc capacities. Finally, section 4 introduces a general class of recourse strategies referred to here as hierarchical strategies. The goal here is not to present specific new algorithms but rather to present a fresh perspective on recourse strategies that may lead to new bounds and approximations.

1. STOCHASTIC PROGRAMMING PROBLEMS

THERE IS, of course, an extremely wide variety of stochastic programming formulations, each exhibiting its own special structure. For the purpose of our presentation, three closely related problems are presented which serve as a basis for discussion for recourse strategies. The first is a two-stage transportation problem with random demands. The second is an N-stage transportation-type problem (each stage involves a bipartite graph) with random arc capacities. The third is a general N-stage dynamic (transshipment) network with random arc capacities. We begin the presentation with a brief discussion of notation and the basic framework we are working in. At the end of the section we discuss some of the contexts in which each of these situations may arise in practice.

1.1. Notation

We consider stochastic, dynamic network problems which involve determining flows between different points in space and time. We refer to points in space as cities, which may refer to regions, terminals, warehouses or ports. Flows between cities are assumed to move forward in time. Let $\mathbf{R}$ be the set of cities and $t = 1, 2, \ldots, P$ be the time periods where P is the planning horizon. We do not consider issues associated with the truncation of the planning horizon. For a given city $i \in \mathbf{R}$, we denote a node in the network by (i, t), representing a particular city at a point in time t. If t_{ij} is the travel time (in units of time periods) from city i to city j, then we denote the link from (i, t) to $(j, t +$

t_{ij}) by (i, t, j). In some instances, we allow the travel time between cities to be zero.

Decision variables are denoted by $x_{ij}(t)$ which gives the flow from city i to city j in time period t. The argument (t) is used to denote the stage in which a variable is determined. $S_i(t)$ is used to denote the flow through node (i, t) determined by decisions made before time period t. $R_i(t)$ represents the exogenous demands placed on the network, representing flows entering or leaving the network at (i, t).

An important concept in stochastic programming is the notion of a stage. Random variables in a given stage are all realized at the same time, and after those in any previous stage. A stage may consist of multiple time periods, but we assume each stage consists of a single time period, and use the terms time period and stage interchangeably.

1.2. The Two-Stage Stochastic Transportation Problem

Perhaps the most widely formulated stochastic programming problem is the two-stage transportation problem. An initial vector of supplies $R(1)$ must be allocated from one set of cities to another (we will use the same sets of cities for supplies and demands, but clearly it is possible that some cities may serve as supply points and other cities may serve as demand points). We have to move flow from supplies to demands in the first stage, prior to realizing demands in the second stage. Once demands are realized, we may again allocate flow from city to city. The formulation includes the situation where the set of cities can be divided into plants, warehouses and customers. Supplies originate only at plants, and can only be shipped to warehouses in the first period. In the second period, goods may be moved from warehouses to markets. For notational simplicity, we use one set of cities in all periods.

This problem is given by:

$$\min_{x(1),\, S(1)} \; c^T x(1) + \bar{Q}(S(1)) \tag{3}$$

subject to:

$$\textstyle\sum_{j \in \mathbf{R}} x_{ij}(1) = R_i(1) \quad \forall i \in \mathbf{R} \tag{3a}$$

$$\textstyle\sum_{i \in \mathbf{R}} x_{ij}(1) - S_j(1) = 0 \quad \forall i \in \mathbf{R} \tag{3b}$$

$$x_{ij}(1) \geqslant 0 \quad \forall i, j \in \mathbf{R} \tag{3c}$$

where $\bar{Q}(S(1))$ is the expected recourse function defined by:

$$\bar{Q}(S(1)) = E_{\xi(2)}[Q(S(1), \xi(2)] \tag{4}$$

and where $Q(S(1), \xi(2)$ is defined by:

$$Q(S(1), \xi(2)) = \min_{x(2)} c^T x(2) \tag{5}$$

subject to:

$$\textstyle\sum_{j \in \mathbf{R}} x_{ij}(2) = S_i(1) \quad \forall i \in \mathbf{R} \tag{5a}$$

$$\textstyle\sum_{k \in \mathbf{R}} x_{kj}(2) \geqslant \xi_j(2) \quad \forall j \in \mathbf{R} \tag{5b}$$

$$x_{ij}(2) \geqslant 0 \quad \forall i, j \in \mathbf{R} \tag{5c}$$

Here and throughout the paper, the decision variable $x(2)$ for the conditional recourse function $Q(S(1), \xi(2))$ is assumed to be conditional on $\xi(2)$. Within the conditional recourse function, we choose $x(2)$ after $\xi(2)$ is realized. It is generally apparent when this is the case, so we do not explicitly write the dependency of $x(2)$ on $\xi(2)$.

Equation (5a) constrains the flow out of region i in the second time period to the total flow coming into this region from the first period. Equation (5b) is the demand constraint, which requires that the total flow into region j meets or exceeds the market demand for that region. Note that here and in the remainder of the paper, we use $x(2)$ within the conditional recourse problem $Q(S(1), \xi(2))$ without explicitly indexing $x(2)$ by the realization of $\xi(2)$. Thus, we have to choose a vector $x(2)$ for each realization of $\xi(2)$.

This problem can be visualized using the network in Figure 1. In this figure, each node is identified by the city and stage. Thus, cities 1 through 4 are represented at the beginning of stage 1, and then

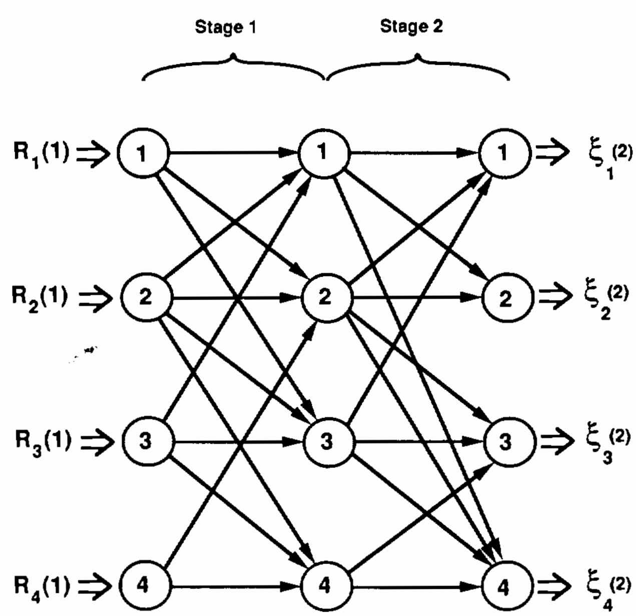


Fig. 1. Two-stage stochastic transportation problem with network recourse.

twice within stage 2 (before and after the transportation decision has been made). In the first stage, we solve a one-sided transportation problem with supplies $R(1)$ at the beginning of the period but no particular demands at the end of the period, and no a priori knowledge of the demands $\xi(2)$ at the end of the second period. Flows $x(1)$ in the first period create supplies $S(1)$ at the beginning of the second period. For a given set of supplies $S(1)$, we must now solve a second transportation problem for a given *realization* of the demands $\xi(2)$.

One difficulty with (3) is that it may not be feasible for a given vector $\xi(2)$. This situation can be avoided by modifying the problem to allow for underage or overage with a (possibly high) penalty.

1.3. *N*-Stage Transportation Problems With Random Arc Capacities

The second problem is the N-stage stochastic transportation problem with random arc capacities depicted in Figure 2. This problem arises in dynamic fleet management, where the limit on the number of vehicles that can move loaded between two regions is limited by a forecasted (and therefore uncertain) demand. These demands are modeled as random arc capacities (see POWELL[20]). As before, each node is identified by the city it represents. We assume that the entire set of cities is replicated over time. This is stated as:

$$\min_{x(1),S(1)} G(x(1)) = c^T x(1) + \bar{Q}(S(1)) \tag{6}$$

subject to:

$$\textstyle\sum_{j\in\mathbf{R}} x_{ij}(1) = R_i(1) \quad \forall i \in \mathbf{R} \tag{6a}$$

$$\textstyle\sum_{k\in\mathbf{R}} x_{kj}(1) - S_j(1) = 0 \quad \forall i \in \mathbf{R} \tag{6b}$$

$$x_{ij}(1) \geqslant 0 \quad \forall i,j \in \mathbf{R} \tag{6c}$$

$$x_{ij}(1) \leqslant u_{ij}(1) \quad \forall i,j \in \mathbf{R} \tag{6d}$$

where

$$\bar{Q}(S(t-1)) = E_{\xi(t)}[Q(S(t-1),\xi(t)] \qquad t = 2,\ldots,P \tag{7}$$

The function $\bar{Q}(S(t-1))$ is the recourse function and is defined by the following recursion:

$$Q(S(t-1),\xi(t)) = \min_{x(t),S(t)} c(t)^T x(t) + \bar{Q}(S(t)) \tag{8}$$

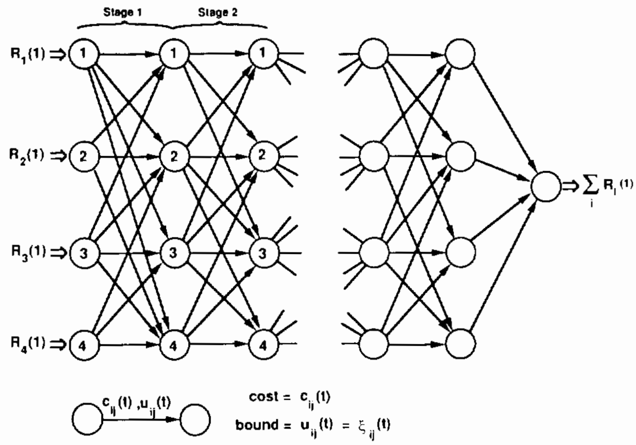


Fig. 2. N-stage dynamic network with random arc capacities.

subject to:

$$\textstyle\sum_{j\in\mathbf{R}} x_{ij}(t) = S_i(t-1) + R_i(t) \quad \forall i \in \mathbf{R} \tag{8a}$$

$$\textstyle\sum_{k\in\mathbf{R}} kx_{kj}(t) - S_j(t) = 0 \quad \forall j \in \mathbf{R} \tag{8b}$$

$$x_{ij}(t) \geqslant 0 \quad \forall i,j \in \mathbf{R} \tag{8c}$$

$$x_{ij}(t) \leqslant \xi_{ij}(t) \quad \forall i,j \in \mathbf{R} \tag{8d}$$

The recursion (7)–(8d) is defined over the planning horizon $t = 2,3,\ldots,P$ with

$$\bar{Q}(S(P)) = 0 \tag{8e}$$

In this problem, flows must be moved in stage t in anticipation of future events. There are no demands at any nodes, and hence flow does not leave the network at any point. Instead, there are random arc capacities in each stage.

1.4. General *N*-Stage Networks With Random Arc Capacities

The last and most general class of problems is the N-stage general network problem with random arc capacities, depicted in Figure 3. The problem is similar to (6) with the addition of flow conservation constraints at transshipment nodes within each period. In this figure, the same 12 cities are replicated in each stage. However, a city might represent a plant, warehouse or customer.

Notationally, the problem can be described in a similar fashion to the stochastic transportation problem with random arc capacities. We assume, as before, that each time period represents a stage and that a general network connects the cities within a stage. The important characteristic is that all the random variables within a stage are realized simultaneously, and the network within each stage has a relatively general structure. Unlike the other

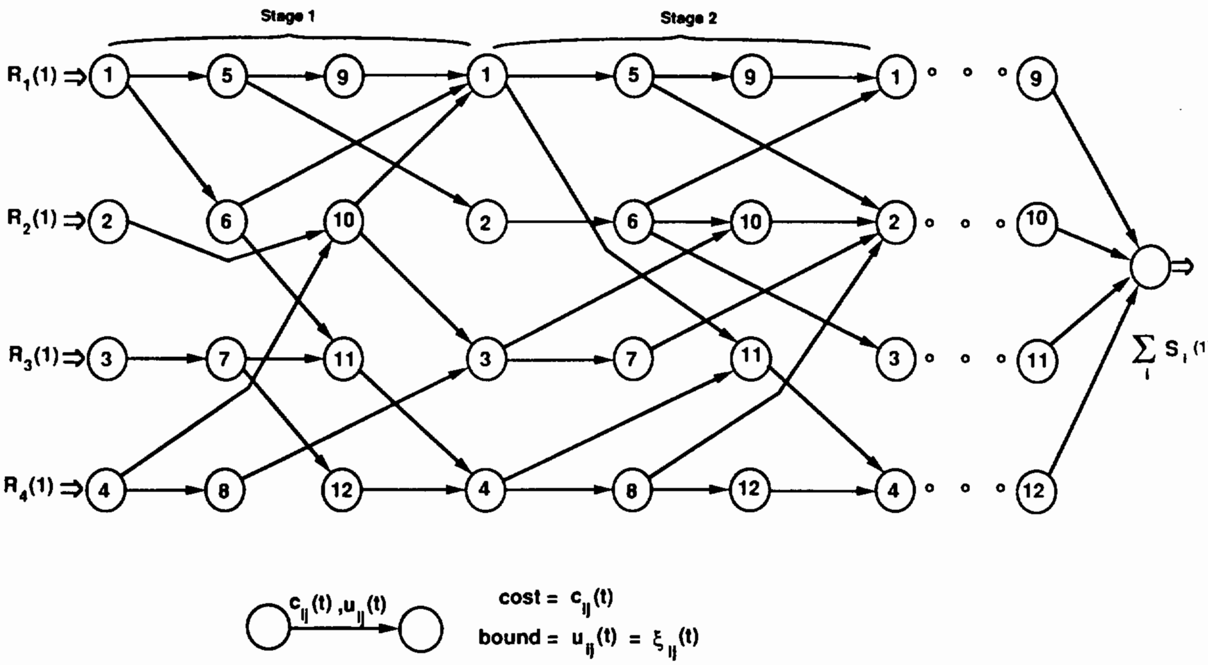


Fig. 3. N-stage dynamic network, with transshipment networks in each stage.

two models, we now need to distinguish between flows within a stage and flows between stages. Define:

$$x_{ij}(t, t') = \text{flow from city } i \text{ in stage } t \text{ to city } j \text{ in stage } t', \text{ where we assume } t' \geqslant t)$$

In this context we use $S_i(t)$ as the flow into city i from decisions made in stages t and earlier, which can be moved in period $t+1$. Let $x(t)$ be the vector of link flows originating in stage t to stages $t, t+1, \ldots,$ and let $c(t)$ be the corresponding vector of link costs. Then the problem can be stated as:

$$\min_{x(1), S(1)} c(1)^T x(1) + \overline{Q}(S(1)) \tag{9}$$

subject to:

$$\sum_{t' \geqslant 1} \sum_{j \in \mathbf{R}} x_{ij}(1, t') - \sum_{k \in \mathbf{R}} x_{ki}(1,1) = R_i(1) \quad \forall i \in \mathbf{R} \tag{9a}$$

$$\sum_{i \in \mathbf{R}} x_{ij}(1,2) - S_j(1) = 0 \quad \forall j \in \mathbf{R} \tag{9b}$$

$$x_{ij}(1) \leqslant u_{ij}(1) \quad \forall i, j \in \mathbf{R} \tag{9c}$$

$$x_{ij}(1) \geqslant 0 \quad \forall i, j \in \mathbf{R} \tag{9d}$$

The expected recourse function is given by:

$$\overline{Q}(S(t-1)) = E_{\xi(2)}[Q(S(t-1), \xi(t))]$$

where

$$Q(S(t-1), \xi(t)) = \min_{x(t), S(t)} c(t)^T x(t) + \overline{Q}(S(t)) \tag{10}$$

subject to:

$$\sum_{t' \geqslant t} \sum_{j \in \mathbf{R}} x_{ij}(t, t') - \sum_{k \in \mathbf{R}} x_{ki}(t,t) = R_i(t) + S_i(t-1) \quad \forall i \in \mathbf{R} \tag{10a}$$

$$\sum_{t' \leqslant t} \sum_{i \in \mathbf{R}} x_{ij}(t', t+1) - S_j(t) = 0 \quad \forall j \in \mathbf{R} \tag{10b}$$

$$x_{ij}(t) \leqslant \xi_{ij}(t) \quad \forall i, j \in \mathbf{R} \tag{10c}$$

$$x_{ij}(t) \geqslant 0 \quad \forall i, j \in \mathbf{R} \tag{10d}$$

Equation (10a) defines the flow conservation constraint for each node within a stage. Note that we use the supply vector $S_i(t-1)$ to summarize inputs from previous stages. Equation (10b) is the definitional constraint for $S(t)$.

It should be apparent that problem (3), in addition to (6), can also be modeled as a network with random arc capacities by using standard network tricks, as depicted in Figure 4. Here, stage 2 has been modified by the addition of a supersink with links from each demand node into the super sink. This new network can be viewed as one where the second stage is broken into two time periods. The first of these, time period 2, is similar to time period 1, and consists only of a deterministic trans-

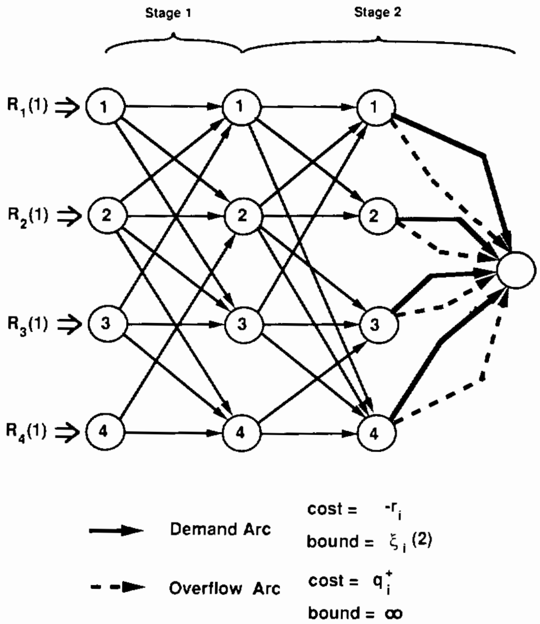


Fig. 4. Two-stage stochastic transportation problem, modeled as a network with random arc capacities.

portation problem. The third time period consists of *demand arcs* and *overflow arcs*. The upper bound on a demand arc out of city i is the random variable $\xi_i(2)$, which in (3) was the market demand. However, in (3) we were constrained to satisfy the market demand, whereas here we assume there is a revenue associated with the market demand (given as a negative cost) which is high enough to encourage the optimization to satisfy some or all of the demand. Thus if:

r_i = revenue derived from satisfying the demand in city i

then we would put a cost of $-r_i$ on the demand arcs out of city i (it is common to include not only the cost of lost revenue but also a penalty for unsatisfied demand). The coefficient on an overflow arc is the cost of having excess supply minus any salvage value. Note that the network representation depicted in Figure 4 overcomes one limitation of the classical two-stage stochastic optimization problem, namely that the optimization in (4) is potentially infeasible.

Figure 4 models the two-stage stochastic transportation problem as a network with random arc capacities. It is important to note that the network in Figure 4 is more similar to the general problem (9) than it is to the stochastic transportation problem with random arc capacities (6). Specifically, (6) has the property that the links with random arc capacities are directly incident to the nodes into which flow is supplied. In Figure 4, flow enters stage two at cities 1–4 at the beginning of the stage, but the random arc capacities are on links that are emanating from cities 1–4 at the end of the stage.

1.5. Applications

The generic two-stage stochastic transportation problem (3) exists more as an abstract model than as a true engineering application, but the classical uses of this model are generally obvious. For example, an automobile importer will distribute a batch of cars among a group of dealers in the region based on forecasted demand (this is the first stage problem). After customer demands for a particular class of car are realized, dealers may move cars between themselves to better satisfy these known demands. In another instance, goods are moved from plant to warehouse and then, as demands are realized, moved from warehouse to the customer. Large appliance manufacturers often work in this mode, where customer demands are satisfied from a warehouse as opposed to providing inventory at individual stores. This allows the final movement of appliances from warehouse to market to be done after the demands are known.

Transportation networks with random arc capacities (9) have been used in the formulation of the stochastic dynamic vehicle allocation problem. The decision variables $x_{ij}(t)$ represent the flow of vehicles from i to j in period t. The random arc capacities $\xi_{ij}(t)$ are used to model the market demand from i to j in period t. In the dynamic vehicle allocation problem we distinguish between two types of movements: loaded movements, which produce a negative cost (actually a positive profit) and are restricted by $\xi_{ij}(t)$, and empty movements which move at a positive cost and have no upper bound. Note that the formulation (9) implies travel times between cities of one period. There is no need to make this assumption and the discussions that follow do not require this assumption.

Instances of dynamic fleet management problems arise in truckload trucking, rail and container traffic, as well as (one-way) auto and truck rental. In trucking, it is common to represent each day as a different stage, where decisions must be made one day in anticipation of, but without actually knowing, future demands. The problem is often formulated over perhaps a 7–14-day planning horizon, so that decisions made today take into account possible downstream activities. For example, the decision to accept a load from Chicago to Phoenix must take into account that once the truck arrives in Phoenix, it may have to move empty to Los Angeles before picking up a load back to the east cost. If the model is trying to decide between this load and another one going to Cleveland (a much shorter distance) it is necessary to compare the two options over the same length planning horizon.

By contrast, international container applications are more effectively modeled on a week-by-week basis, since ships typically depart from a port on a weekly schedule. However, decisions made one week may have to anticipate demands over the next 4 to 6 weeks, given the long distances involved. Thus, the decision to move a container empty from Tokyo to Chicago may require anticipating market demands out of Chicago 4 weeks into the future.

The transportation problem with random arc capacities exhibits a special property of a bipartite network within each stage, where decisions must be made regarding the amount of flow from one node to the next, and where the random variables appear as arc capacities on links out of the supply nodes. A problem as simple as (3), however, does not exhibit this property and hence the concept cannot be used. The N-stage network with random arc capacities, (9), is introduced as a much more general model that would arise in situations where, within a stage, multiple moves over a transshipment network can be made after all the random variables within the stage have been realized. Using standard network tricks, a broad range of stochastic network problems can be formulated in this way.

2. A STRATEGY FOR APPROXIMATING RECOURSE FUNCTIONS

THE challenge of stochastic programming problems is the analytical intractability of the recourse function, where an optimization problem is contained within an expectation. Notwithstanding the difficulty of solving this problem for a particular set of first stage decisions $x(1)$, we would like ideally like to find recourse functions expressed directly as a function of $x(1)$. This goal is generally difficult because of the number of dimensions of both the decision vector $x(t)$ and the random vector $\xi(t)$, combined with the property that $Q(x, \xi)$ is typically a nonseparable function of both variables.

The development of an approximate recourse function requires two steps. First, the imbedded optimization within the recourse function must be replaced with a much simpler search procedure. Second, the probabilistic structure of the resulting optimal solution, conditioned on the random vector ξ, must possess sufficient structure so that taking its expectation is computationally tractable. We propose to pursue this approach by replacing the imbedded optimization with a restricted optimization problem that is easier to solve. Thus, instead of using full network recourse (which implies the solution of a network optimization problem within the expectation), we would use restricted recourse strategies that would approximate a network optimization problem. However, the simple identification of a restricted recourse strategy that is easy to solve is not enough. The resulting solution must also exhibit sufficient structure to allow the expectation to be taken easily.

If we are to develop a functional approximation to the expected network recourse function, it is necessary to make assumptions about the structure of the approximation. In our case, the most natural structure is to assume that the function is separable in the vector $S(1)$. Thus, we would like to find:

$$\overline{Q}(S(1)) \cong \hat{Q}(S(1)) = \sum_j \hat{Q}_j\big(S_j(1)\big) \qquad (11)$$

The problem now is to develop approximations $\hat{Q}_j(S_j(1))$ which capture the marginal effects of changing $S(1)$. If a separable approximation proves accurate, then the original optimization problem can be solved as a pure network with possibly nonlinear (or piecewise linear) costs. While it is possible that more complex recourse functions may prove necessary, the attractiveness of solving the combined problem as a pure network (as long as the first stage problem is a pure network) is enough to motivate this line of investigation.

The basic strategy for developing the approximations $\hat{Q}_j(S_j(1))$ is as follows. For a fixed vector $S(1)$, we wish to parametrically change $S_j(1)$ over the range $0, 1, \ldots, n$, where n is a suitably chosen maximum value. We wish to then find $q_j(s) = \overline{Q}_j(s) - \overline{Q}_j(s-1)$, which gives the expected incremental value of the s^{th} unit of flow. If we denote $\hat{x}_j(s, 2)$ the optimal expected flows in the second stage, given a vector of input flows $S(1)$ with the j^{th} element equal to s, then $q_j(s) = c^T[\hat{x}_j(s, 2) - \hat{x}_j(s - 1, 2)]$.

The biggest challenge in developing an approximation to the recourse function is the complex interactions between the random variables as a result of the optimization. For this reason, it is likely we will need to combine the use of restricted recourse strategies with other approximations .

The development of separable approximations requires simplifying the imbedded optimization within the recourse function so that the expectation can be handled easily. The goal is to be able to estimate the marginal impact of increasing s in $\hat{Q}_j(s)$ for integer values of s over a specified range. This can be viewed as *optimizing* the assignment of each incremental unit of flow given the assignment of the first s units of flow. The nonseparability of $Q(s)$, however, combined with the often difficult probabilistic structure of the problem, requires the use of various approximations to simplify the problem. Several methods can be used, sometimes in combination, to achieve this, including:

1) relaxation methods,
2) linearization approximations,
3) probabilistic decomposition (variable splitting).

Relaxation methods induce separability by unbundling decisions when key constraints are eliminated. Linearization approximations represent a different mechanism that achieves the same affect, by replacing certain nonlinear functions with linear ones. Linearization has proved useful in the solution of multistage stochastic programs. Probabilistic decomposition reduces or eliminates interactions by redefining the random variables in such a way so as to reduce or eliminate interactions between them.

The next section reviews a specific set of restricted recourse strategies for stochastic networks, with an emphasis on stochastic transportation problems with random arc capacities (6). After this, section 4 introduces a new class of recourse strategies that are more amenable to general networks.

3. APPROXIMATE RECOURSE STRATEGIES FOR STOCHASTIC NETWORKS

In this section, we focus on a family of restricted recourse strategies that make the expectation of the recourse function analytically tractable. Our focus is on transportation problems with random arc capacities (6).

We begin by reviewing simple recourse in some depth because it is often used in the research literature, due to its analytical simplicity. Our presentation looks at simple recourse as an approximation of network recourse. Next we discuss null recourse and nodal recourse which successively generalize simple recourse for certain applications. Nodal recourse is the first interesting example of a restricted recourse strategy, but it is highly specialized to a particular network structure. We close the section with a presentation of a new set of strategies which represent variations of extended nodal recourse, which illustrates both the potential and the pitfalls of this approach.

3.1. Simple Recourse

Simple recourse arises in a number of settings where there is effectively no recourse once the random vector is realized, aside from incurring penalties. Simple recourse can also be used in more complex problems to simplify the interaction between the expectation and the imbedded optimization problem within the recourse function. The heart of simple recourse is the inclusion of a set of *recourse variables* which absorb the effects of randomness and therefore eliminate the interactions between random variables. For example, consider the expected recourse function for the two stage stochastic transportation problem.

The classical stochastic transportation problem in (3)–(5) can be approximated using simple recourse by replacing the constraint $\sum_k x_{kj}(2) = \xi_j(2)$ with

$$x_{ii}(2) + x_{ii}^{-}(2) - x_{ii}^{+}(2) = \xi_i(2) \qquad (12)$$

$$x_{ij}(2) = 0 \quad i \neq j \qquad (13)$$

where $x_{ii}^{+}(2)$ and $x_{ii}^{-}(2)$ are the *recourse variables*, representing, respectively, overage or underage, which are given by:

$$x_{ii}^{+}(2) = \max[x_{ii}(2) - \xi_i(2), 0] \qquad (14)$$

$$x_{ii}^{-}(2) = \max[\xi_i(2) - x_{ii}(2), 0] \qquad (15)$$

Let $q_{ij}^{+}(t)$ and $q_{ij}^{-}(t)$ be the coefficients of $x_{ij}^{+}(t)$ and $x_{ij}^{-}(t)$. In the transportation problem, q^{+} is the salvage value of excess supply while q^{-} is typically a penalty for unsatisfied demand. In other applications, however, the recourse variables can take on very different interpretations. The effect of simple recourse in this context is to force the problem to satisfy the random demand ξ_i in period 2 with the flow sent to city i in time period 1, as illustrated in Figure 5. Adding the recourse variables to the ob-

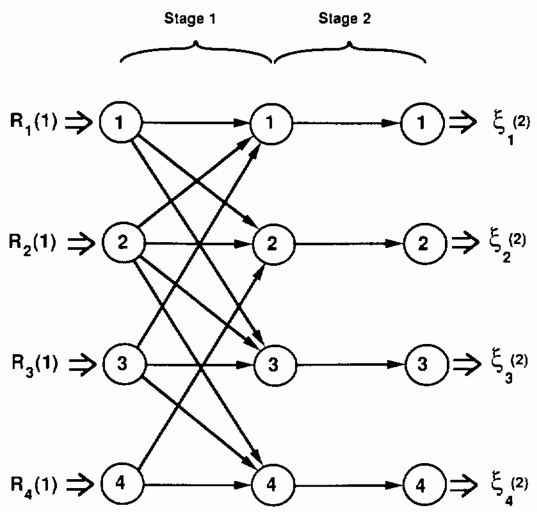


Fig. 5. Stochastic transportation problem with simple recourse.

jective function gives:

$$\Phi(S(1), \xi(2)) = \sum_i c_{ii} S_i(1) + \sum_i q_i^+ \max[S_i(1) - \xi_i(2), 0] + \sum_i q_i^- \max[\xi_i(2) - S_i(1), 0] \quad (16)$$

which is the optimal solution of (16) parameterized by $S(1)$ and $\xi(2)$. The right-hand side of (16) is a separable function of ξ_i and hence it is fairly easy to take expectations of $\Phi(S(1), \xi(2))$. If $\xi(2)$ is a continuous (discrete) random variable then $\Phi(S(1)) = E_\xi[\Phi(S(1), \xi(2))]$ becomes a nonlinear (piecewise linear), separable function of $x(1)$.

It is unlikely that anyone would actually use simple recourse to solve a transportation problem with network recourse. The development does illustrate, however, how a restricted recourse strategy (achieved by the addition of the recourse variables) can simplify the development of the expected recourse function.

For networks with random arc capacities, the notion of simple recourse can be given a somewhat richer interpretation. Constraints (12)–(13) force flows from i to $j, i \neq j$ to equal zero. These constraints can be replaced with

$$x_{ij}(t) + x_{ij}^-(t) - x_{ij}^+(t) = \xi_{ij}(t) \quad (17)$$

Again

$$x_{ij}^+(t) = \max\left[x_{ij}(2) - \xi_{ij}(2), 0\right] \quad (18)$$

$$x_{ij}^-(t) = \max\left[\xi_{ij}(2) - x_{ij}(2), 0\right] \quad (19)$$

The assumption is made that $x_{ij}(t)$ must be chosen prior to knowing $\xi_{ij}(t)$ while $x_{ij}^-(t)$ and $x_{ij}^+(t)$ must be "optimized" for a given realization of $\xi_{ij}(t)$ (these optimal solutions are given in (18) and (19)).

The recourse variables can be given simple interpretations. $x^-(t)$ is interpreted as lost demand and can be assigned a penalty q^-. $x^+(t)$ can be interpreted as a nonrevenue producing movement of flow. In the dynamic vehicle allocation problem, $x_{ij}^+(t)$ represents moving vehicles empty as a result of insufficient demand. Thus $x_{ij}(t)$ is the total number of vehicles allocated to move from i to j in period t, with $x_{ij}(t) - x_{ij}^+(t)$ moving "loaded" (producing revenue, with $c_{ij}(t) < 0$) and $x_{ij}^+(t)$ moving "empty" (at a positive cost). Most important is the implicit assumption that $x_{ij}(t)$ must be chosen prior to knowing $\xi_{ij}(t)$. This can be visualized as breaking stage t into two stages, t and t', where $x(t)$ is chosen in stage t, while $x^+(t')$ and $x^-(t')$ are chosen in stage t'. Since $x^+(t')$ and $x^-(t')$ do not appear in any constraints for stages other than t, let $\bar{\phi}(x_{ij}(t))$ be the expected recourse function for stage t', given by:

$$\bar{\phi}\left(x_{ij}(t)\right) = E_{\xi_{ij}(t)}\left[q_{ij}^- x_{ij}^-(t) + q_{ij}^+ x_{ij}^+(t)\right] \quad (20)$$

where $x^+(t)$ and $x^-(t)$ are given by (18)–(19). $\bar{\phi}(x_j(t))$ is a separable, nonlinear convex function in $x_{ij}(t)$. Adding these imbedded recourse functions to the problem (6) with simple recourse may be written:

$$\min_{x(1), S(1)} G(x(1)) = c(1)^T x(1) + \bar{\Phi}(S(1)) \quad (21)$$

subject to (6a)–(6d) with

$$\bar{\Phi}(S(t)) = E_{\xi(t+1)}[\Phi(S(t), \xi(t+1))] \quad (22)$$

where

$$\Phi(S(t), \xi(t+1)) = \min_{x(t), S(t)} \sum_i \sum_j c_{ij}(t) x_{ij}(t) + \bar{\phi}_{ij}\left(x_{ij}(t)\right) + \bar{Q}(S(t)) \quad (23)$$

subject to:

$$\sum_{j \in \mathbf{R}} x_{ij}(t) = S_i(t-1) \quad \forall i \in \mathbf{R} \quad (23a)$$

$$\sum_{i \in \mathbf{R}} x_{ij}(t) = S_j(t) \quad \forall j \in \mathbf{R} \quad (23b)$$

$$x_{ij}(t) \geqslant 0 \quad \forall i, j \in \mathbf{R} \quad (23c)$$

Note that (23)–(23c) is no longer a function of $\xi(t)$, having been incorporated into $\phi_{ij}(x_{ij}(t))$. Thus $\bar{\Phi}(S(t)) = \Phi(S(t), \xi(t+1))$, implying we may rewrite (21)–(23) as a single optimization problem:

$$\min_{x(1), \ldots, x(P)} \sum_{t=1}^{P} c(t)^T x(t) + \bar{\Phi}(x(t)) \quad (24)$$

subject to (23a)–(23c). Constraints (23a) and (23b) can be combined to eliminate $S_i(t)$, producing a single flow conservation constraint. This problem is now a classical convex, nonlinear network flow problem which can be solved using standard techniques (see, for example, KENNINGTON and HELGASON.[15])

The purpose of this section has been to review simple recourse strategies in the context of two important, related problems. For the two stage stochastic transportation problem, it is shown that simple recourse is equivalent to simply eliminating the network options in the second stage. For the N-stage network with random arc capacities, simple recourse is equivalent to splitting each stage into two stages, where the first half-stage sets the flow variables while the second stage sets the recourse variables. As long as flow is allowed to exceed an arc bound, at a cost (representing a nonrevenue-producing movement of flow), the resulting problem is a deterministic nonlinear network.

The real purpose of simple recourse is to replace a complex, nonseparable recourse function with a simpler, separable one. Simple recourse accomplishes this via exceptionally strong assumptions, producing models that are unlikely to succeed in most practical applications (where network recourse is the appropriate model). In the remainder of this paper, we review alternative recourse strategies that produce computationally feasible algorithms without the strong assumptions required by simple recourse.

3.2. Null Recourse

For problem (6), simple recourse can be viewed as replacing each link with flow $x_{ij}(t)$ with two links carrying flow $x_{ij}(t) - x_{ij}^{+}(t)$ and $x_{ij}^{+}(t)$, respectively. If $\xi_{ij}(t) < x_{ij}(t)$, flow is pushed in response from the primary arc onto the overflow arc. In the context of the dynamic vehicle allocation problem, this is equivalent to saying that if a vehicle cannot move loaded over a link, it will move empty anyway. It is generally the case that $q_{ii}^{+} \ll q_{ij}^{+}$, meaning that holding flow in a city (on the overflow arc) may be much less expensive than the overflow cost for a link moving to another city j. A more efficient strategy, then, might be to let the overflow fall onto the (unbounded) inventory link. Null recourse can be illustrated as shown in Figure 6 as a process whereby flow "spills" from a bounded link onto an inventory link. The total "spilled" flow on the inventory link, $x_{ii}^{+}(t)$, is given by:

$$x_{ii}^{+}(t) = \sum_{j \in \mathbf{R}} \max\left[x_{ij}(t) - \xi_{ij}(t), 0\right] \quad (25)$$

The original recourse function:

$$\bar{Q}(S(t-1)) = E_{\xi(t)}\left[\min_{x(t)} \sum_{i \in \mathbf{R}} \sum_{j \in \mathbf{R}} x_{ij}(t) c_{ij}\right]$$

subject to:

$$\sum_{j \in \mathbf{R}} x_{ij}(t) = S_i \quad \forall i \in \mathbf{R}(t-1)$$
$$x_{ij}(t) \leqslant \xi_{ij}(t) \quad \forall i, j \in \mathbf{R}$$

can be replaced by the null recourse approximation:

$$Q(S(t-1), x(t))$$
$$= E_{\xi(t)}\Big[\sum_{i \in \mathbf{R}} \sum_{j \in \mathbf{R}} \min\big(x_{ij}(t), \xi_{ij}(t)\big) c_{ij} + \sum_{i \in \mathbf{R}} \sum_{j \in \mathbf{R}} \max\big(x_{ij}(t) - \xi_{ij}(t), 0\big) q_{ii}^{+}\Big]$$

The imbedded optimization is handled by the recourse variables, leaving a problem that is still a function of $x(t)$, but outside of the expectation. The expectation is now fairly simple, leaving a nonlinear function of $x(t)$ which can be optimized using standard methods.

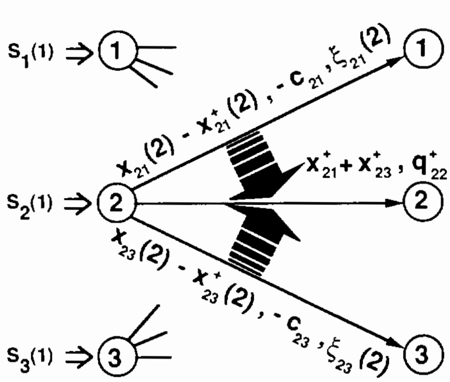


Fig. 6. Illustration of the null recourse strategy, with overflow spilling to the inventory link.

Now, the total flow moving from i to j is $x_{ij}(t) - x_{ij}^{+}(t)$, which is a random variable. This implies that $S(2)$ is a random vector, making a three (or more) stage problem extremely difficult to solve exactly. The nested expectations and optimizations implicit in (6)–(8) make N-stage problems computationally intracable. POWELL [19] presents an approximation whereby the random variables $S_i(t)$, $i = 1, \ldots, R$ are treated as independent with approximate distributions fit around the mean and variance of each random variable. This approach produces a nonlinear programming problem that can be solved efficiently.

Null recourse is a slightly more realistic strategy than simple recourse, but still represents a very strong assumption compared to full network recourse. The next section further generalizes null recourse.

3.3. Nodal Recourse

Simple and null recourse are both policies where realizations of the random vector $\xi(t)$ are handled by using an overflow option. An overflow option is a simple example of a hierarchical recourse strategy, where we try to put flow on one link but, if it is restricted by a random capacity, specify that any excess be put on a specified overflow link. This constitutes a two-level hierarchy (for each link). Most importantly, these strategies are examples of replacing a difficult optimization problem with a much more restrictive one which simplifies taking the expectation of the recourse function.

Nodal recourse generalizes both simple and null recourse by providing for multiple overflow options, which can be viewed as a multilevel hierarchy. The concept of nodal recourse is developed in the context of problem (9), and does not extend easily to more general stochastic networks. It does, however, illustrate another example of a restrictive recourse strategy which allows a recourse function to be solved directly.

Let $\delta_i(t)$ denote the hierarchy of options for flow out of region i at time t, where:

$$\delta_i(t) = \{\delta_{i1}(t), \delta_{i2}(t), \ldots, \delta_{iN}(t)\}$$

In this vector, $\delta_{in}(t)$ is the n^{th} option for a unit of flow, which is used if the first $n - 1$ options are unavailable. Throughout this section we use N to denote the number of options available in any given list. An option represents the ability to move over a link, and the nodal recourse vector captures the ability to spill from one link to the next. However, eventually it is necessary to provide an overflow option to guarantee feasibility. Let:

$\Delta_{ij}(t)$ = symbol representing the option to move over the link (i, t, j)

$E_{ij}(t)$ = symbol representing the option to move over the overflow arc for link (i, t, j)

The move over the link (i, t, j) occurs at a cost $c_{ij}(t)$, while the overflow option carries a penalty $q_{ij}^+(t)$. A simple nodal recourse strategy might be:

$$\delta_i(t) = \{\Delta_{i1}(T), \Delta_{i2}(t), \Delta_{i4}(t), E_{i3}(t)\}$$

This policy would read: "move over the link from i to 1, if possible; otherwise, move over the link from i to 2, if possible; otherwise, move from i to 4, if possible; otherwise, overflow from i to 3." The spilling process from one link to the next is illustrated in Figure 7.

The optimization problem posed by simple nodal recourse involves finding a suitable permutation of options in the vector $\delta_i(t)$. One simple method for solving this is to find a set of values $w_{in}(t)$ where: $w_{in}(t)$ = conditional marginal value of a unit of flow using the n^{th} option out of (i, t).

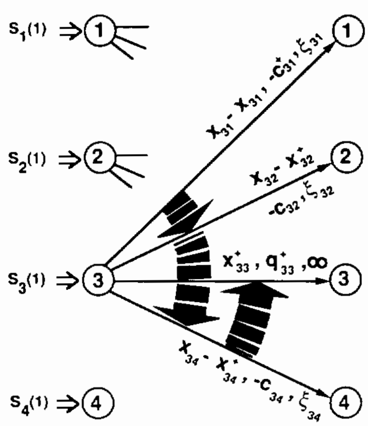


Fig. 7. Illustration of the nodal recourse strategy, with flow spilling from one link to the next out of a given node.

One scheme for calculating these values is:

$$w_{in}(t) = \begin{cases} c_{ij}(t) + p_j(t+1) & \text{if the } n^{\text{th}} \text{ option is to move over link } (i, t, j) \\ q_{ij}(t) + p_j(t+1) & \text{if the } n^{\text{th}} \text{ option is to move over the overflow} \end{cases}$$

where $p_j(t + 1)$ is an estimate of the marginal value of another unit of flow to $(j, t + 1)$. The option vector $\delta_i(t)$ should now be chosen so that:

$$w_{i1}(t) \leqslant w_{i2}(t) \leqslant \cdots \leqslant w_{iN}(t) \tag{26}$$

Nodal recourse is the first interesting example of a strategy we call hierarchical recourse, which is discussed more thoroughly in section 4. Here, we rank a set of options over which flow can be moved using some criterion, and then we incrementally add flow, moving down the list of options as specified in the traversal list $\delta_i(t)$. The probability the k^{th} unit of flow moves over a given option depends on the amount of capacity jointly available for all the options after the first $k - 1$ units of flow have been assigned. To determine these probabilities let:

$$U_{in}(t) = \text{capacity of the } n^{\text{th}} \text{ option.}$$
$$= \begin{cases} \xi_{ij}(t) & \text{if } \delta_{in}(t) = \Delta_{ij} \\ \infty & \text{if } \xi_{in}(t) = E_{ij} \end{cases} \tag{27}$$

$\mathbf{U_i(t)} = \{U_{i1}(t), \ldots, U_{iN}(t)\}$

$Y_{in}^k(t)$ = flow allocated to the n^{th} option after k units of flow have been moved through node (i, t)

$\mathbf{Y_i^k(t)} = \{Y_{i1}^k(t), \ldots, Y_{iN}^k(t)\}$

$V_{in}(t)$ = available capacity on the n^{th} option after k units of flow have been assigned $= U_{in}(t) - Y_{in}^k(t)\mathbf{V_i^k(t)} = \{V_{i1}^k(t), \ldots, V_{iN}^k(t)\}$

Now let

$d_{in}^k(t)$ = probability the k^{th} unit of flow is dispatched on the n^{th} option

$$= \text{Prob}\left[\sum_{l=1}^{n-1} V_{il}^k(t) = 0 \cap V_{in}^k(t) > 0\right] \tag{28}$$

Thus the *kth* unit of flow is dispatched on the n^{th} option if there is capacity remaining on the n^{th} option but no capacity on the first $n - 1$ options. The critical quantity here is the residual capacity, $V_i^k(t)$, which gives the amount of capacity remaining after k units of flow have been assigned. It is the complexity of the probabilistic structure of this vector that makes many problems intractable, forcing the use of recourse strategies that have special structure.

The general use of restricted recourse strategies to develop approximations of recourse functions appears to be relatively new. The notion of nodal recourse was first introduced by POWELL[20] for the stochastic, dynamic vehicle allocation problem. More recently, FRANTZESKAKIS and POWELL[12] show how nodal recourse can be used to approximate the recourse function $Q(S(2))$ as a separable, piecewise linear function, allowing (6) to be solved as a pure network. The key to nodal recourse is that realizations of random arc capacities for arcs incident to node (i, t) control the movement of flow out of node (i, t). This approach is particularly well suited to dynamic transportation problems with the form of problem (6).

Nodal recourse is a classical example of a restricted recourse strategy where the imbedded optimization within the recourse function is forced to consider a narrower set of options. Thus, instead of considering all possible solutions to a network problem, we constrain ourselves to the set of permissible flows allowed by our restricted set of recourse strategies. Since we are constraining our search, our approximate recourse function will produce higher overall costs. The goal is to find a set of recourse strategies that yields a good approximation of the actual recourse strategy but still allows the expected recourse function to be expressed directly as a function of its arguments.

Nodal recourse is well suited to problems such as (6), which involves bipartite networks in each stage. Especially important is the property that links with random capacities are incident to the nodes at the beginning of each stage. For these problems, we need to find a suitable permutation of the option vector $\delta_i(t)$ for each city i that minimizes costs for periods (stages) $t, t + 1, \ldots, P$. A second problem is showing that for a given option vector $\delta(t)$, we can express $\overline{Q}(S(t - 1))$ as a function of the vector $S(t - 1)$. This can be accomplished by replacing the expected recourse function $\overline{Q}(S(t))$ for period $t + 1$ with a linear approximation. This linearization approximation serves to decouple the choice of the option vector $\delta_i(t)$ for each city i (inducing separability) and greatly simplifies the calculation of $\overline{Q}(S(t - 1))$.

3.4. Extensions of Nodal Recourse

The notion of using a restricted set of recourse strategies to develop approximate recourse functions offers the possibility of addressing more difficult problems. In addition, we would like to improve on the accuracy of the approximation provided by simple nodal recourse. In this section, we propose several extensions to nodal recourse, including *generalized nodal recourse*, *partitioned nodal recourse*, and *extended nodal recourse*. Like nodal recourse, these policies are all restricted to problems like (6). However, their description illustrates how different approximations may be built. The presentation begins with generalized nodal recourse, which illustrates how apparently simple heuristics for optimizing the problem can lead to intractable solutions from the perspective of taking expectations.

An obvious limitation of nodal recourse is that it imposes the restriction that every unit of flow through node (i, t) is dispatched with the same policy vector $\delta_i(t)$. An extension, referred to here as *generalized nodal recourse*, uses a different policy $\delta_i^k(t)$ for the k^{th} unit of flow. The motivation is that the marginal value of a unit of flow in region i, time $t + 1$ declines with each unit of flow, suggesting that a declining set of values $p^k(t + 1)$ should be used for the k^{th} unit of flow instead of the constant values used in simple nodal recourse.

The complication with generalized nodal recourse is that it is very difficult to calculate the dispatch probabilities $d_i^k(t)$ when the nodal recourse vectors $\delta_i^k(t)$ are not the same for all k. To see the difficulty, assume:

$$\delta_i^1(t) = \{\Delta_{12}, \Delta_{13}, \Delta_{14}, E_{12}\}$$

$$\delta_i^2(t) = \{\Delta_{14}, \Delta_{13}, \Delta_{12}, E_{12}\}$$

For policy $\delta_i^2(t)$ above, Δ_{12} is now a lower ranked option but for the first unit of flow was a higher ranked option, destroying the probabilistic structure of the residual capacities. Thus, we have to be careful that a restricted recourse strategy exhibits enough structure to allow the expectation to be found easily.

This problem is avoided using a policy termed *partitioned nodal recourse*. Divide the options Δ_{ij}, $j = 1, \ldots, R$ into mutually exclusive sets. Then, we define restricted option policy vectors, $\delta_i^{(l)}(t)$ which must draw from one of these sets. For example, we might have:

$$\delta_i^{(1)}(t) = \{\Delta_{12}, \Delta_{15}, E_{12}\}$$

$$\delta_i^2(t) = \{\Delta_{14}, \Delta_{11}, \Delta_{16}, E_{12}\}$$

$$\delta_i^3(t) = \{\Delta_{17}, \Delta_{13}, E_{14}\}$$

Now, we would set $\delta_i^k(t) = \delta_i^{(l)}(t)$ for some l. The resulting policy structure would be very easy to analyze probabilistically, as long as we remain within a single stage.

Note that simple and null recourse strategies can be viewed as extreme examples of partitioned nodal recourse. Simple recourse uses policy vectors of the form:

$$\begin{aligned}\delta_i^k(t) &= \delta_i^{(l)}(t)\\ &= \{\Delta_{ij}, E_{ij}\}\end{aligned}$$

while null recourse uses:

$$\begin{aligned}\delta_i^k(t) &= \delta_i^{(l)}\\ &= \{\Delta_{ij}, E_{ii}\}\end{aligned}$$

For both simple and null recourse, we have the problem of deciding in what order to sequence the policy vectors $\delta_i^{(l)}(t)$.

A different approach to avoiding the difficulties of generalized nodal recourse is to use *extended nodal recourse*, which works as follows. In nodal recourse, each option may appear only once in the policy vector, and the capacity of each option is defined by $U_i(t)$ in (27). Assume now that we allow options to appear multiple times, such as:

$$\delta_i^e(t) = \{\Delta_{14}, \Delta_{13}, \Delta_{14}, E_{12}, \Delta_{13}, E_{14}, E_{12}\}$$

Associated with this extended option vector are extended option capacities:

$$\begin{aligned}U_i^e(t) = \{&U_{i1}^e(t), U_{i2}^e(t), U_{i3}^e(t), U_{i4}^e(t),\\ &U_{i5}^e(t), U_{i6}^e(t), U_{i7}^e(t)\}\end{aligned}$$

These new option capacities will, in general, be random variables, and may be built in a variety of ways. To produce a computationally tractable procedure, the random variables should be independent. Also, the last option should be an infinite capacity overflow option to insure feasibility.

Extended nodal recourse creates the effect of distributing flow among a broader range of options. Simple nodal recourse can produce an extreme solution if a large amount of flow is pushed over a link with a large upper bound (or onto the highest ranked overflow option). Furthermore, if an effective strategy can be developed to calculate the capacities $U_{ij}^e(t)$, then the resulting policy is computationally very easy.

4. HIERARCHICAL RECOURSE STRATEGIES

NODAL recourse, and its various extensions, is fundamentally a policy which induces separability in the recourse function for stage t, $\overline{Q}(S(t))$, by restricting possible choices to a prespecified option vector. The approach is well suited to problem (6), but is limited in its application to other problems, since the approach defines a recourse strategy at a node dependent on the random capacities of links incident to that node. To see the limitation of this approach, consider using the policy on the classical two-stage stochastic transportation problem. Referring to Figure 4, note that the links emanating from nodes at the beginning of stage 2 are unbounded, implying that $\xi_{ij}(2) = \infty$ for these links. Because the arcs with random capacities are removed one link away, the nodal recourse policy at the beginning of the stage is ineffective. Furthermore, applying nodal recourse to the nodes from which these random links do emanate is equivalent to simple recourse since there is only one link out of each node. This relatively extreme behavior carries forward to general stochastic network problems with random arc capacities.

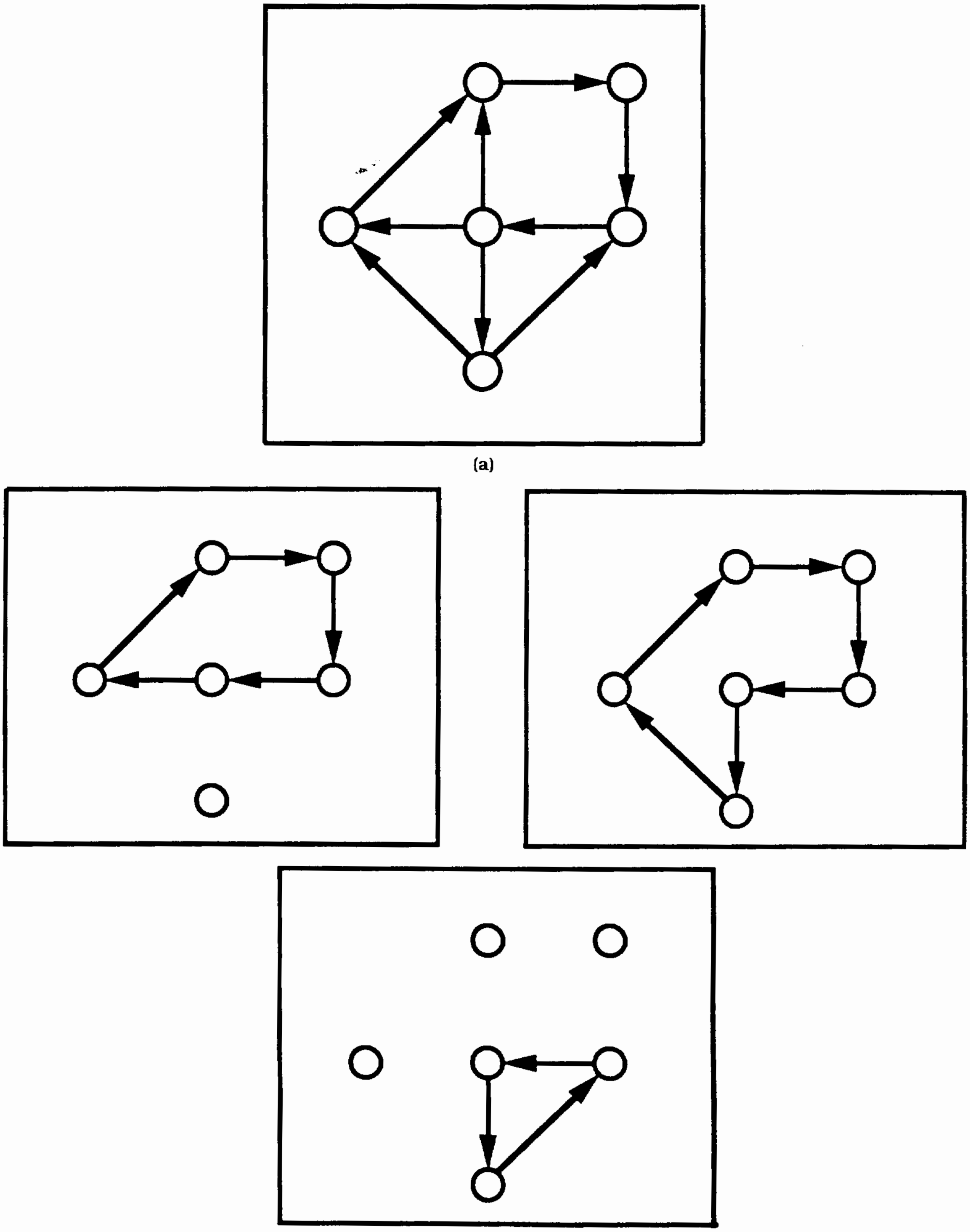
Nodal recourse has two fundamental shortcomings:

i) Information about the *expected* behavior of stage $(t + 1)$, given $S_i(t)$ and policies $\delta_i(t)$, needs to be communicated back to stage t (the linearization of the recourse function at time t destroys this).
ii) Information about *realizations* of random variables on downstream links within stage t needs to be communicated back to links at the beginning of the stage.

We begin by addressing issue (ii) above in the context of a two-stage general network with random arc capacities. Section 4.1 reviews an idea originated by WALLACE[26], which was used to develop piecewise linear bounds for stochastic networks. This section introduces the notion of cyclic recourse which is represented as a special case of what are referred to here as *hierarchical recourse strategies*. Sections 4.2 and 4.3 then present different types of hierarchical recourse strategies for dynamic, acyclic networks.

4.1. Cyclic Recourse

In WALLACE,[26] a simple approach is presented which can be used to develop piecewise linear bounds for general networks with random arc capacities. Consider the sample network shown in Figure 8a which represents the second stage of a two-stage general network with random arc capacities. Let $G = \{N, A\}$ represent the graph and let $L = |A|$ be the number of links. We begin by solving (9) using $\overline{Q}(S(1)) = Q(S(1), \hat{\xi}^0(2))$, where we are fixing the market demands to a constant vector $\hat{\xi}^0(2)$ (for example, we might use $\hat{\xi}^0(2) = E[\xi(2)]$). Let $x^0(2)$ be the optimal flows for this problem, where these flows must of course satisfy (6a–d). Now define $z(2) = x(2) - x^0(2)$ to be the changes in the flows in the second stage relative to the base



(a)

(b)

Fig. 8. (a) Sample second-stage transshipment network. (b) Decomposition of second-stage network into cycles.

solution. The general recourse function may now be written as:

$$\Psi(\xi(2)) = \min_{z(2)} c^T(z(2) + x^0(2)) \tag{29}$$

subject to:

$$\sum_i z_{ij}(2) = 0 \quad \forall j \in \mathbf{R} \tag{29a}$$

$$z_{ij}(2) \leqslant \xi_{ij}(2) - x^0_{ij}(2) \quad \forall i, j \in \mathbf{R} \tag{29b}$$

$$z_{ij}(2) \geqslant -x^0_{ij}(2) \quad \forall i, j \in \mathbf{R} \tag{29c}$$

Any flow vector for the network in Figure 8a can be decomposed into a set of cycles shown in Figure 8b. Let $d^{(n)}$ be the L-dimensional incidence vector giving the links in the m^{th} cycle, and let M_C be the set of cycles (we do not assume that all the cycles in the network have been enumerated). Finally let $\beta_m = c^T d^{(m)}$ be the cycle cost, and let F_m be a scalar random variable denoting the cycle flow. We can let $x^{(m)} = F_m d^{(m)}$ be the set of link flows induced by cycle $d^{(m)}$.

The problem is in finding the cycle flows $x^{(m)}$ taking into account the potentially complex interactions that may exist among the cycles.

WALLACE solved this problem as follows. First, assume that the cycles are ordered so that

$$\beta_1 \leqslant \beta_2 \leqslant \cdots \leqslant \beta_M \tag{30}$$

Since cycle 1 has the most negative reduced cost, we give it the highest priority for receiving flow (that is, the top of the hierarchy). Now assume that the random variables ξ_{ij} satisfy

$$0 \leqslant \xi_{ij} \leqslant \xi^{max}_{ij} \tag{31}$$

and let

$$\xi^{(m)} = \min\left\{ \xi^{max}_{ij} | d^{(m)}_{ij} = 1 \right\} \tag{32}$$

Thus $\xi^{(m)}$ is the smallest maximum upper bound of all the links in cycle m, implying

$$0 \leqslant F_m \leqslant \xi^{(m)} \tag{33}$$

Consider another link appearing in the highest priority cycle with upper bound ξ_{ij} with a distribution as shown in Figure 9a. We can create a new random variable

$$\xi^{(1)}_{ij} = \min\left\{ \xi_{ij}, \xi^{(1)} \right\} \tag{34}$$

which is the highest possible *binding* capacity for link (i, j) for cycle 1. Now let $\hat{\xi}^{(1)}_{ij} = \xi_{ij} - \xi^{(1)}_{ij}$ be the slack capacity. The distributions of $\xi^{(1)}_{ij}$ and $\hat{\xi}^{(1)}_{ij}$ are depicted in figures 9b and 9c. Note that $\xi^{(1)}_{ij}$ and $\hat{\xi}^{(1)}_{ij}$ are *complementary* random variables in that they satisfy

$$\left(\xi^{(m)} - \xi^{(1)}_{ij} \right)\left(\hat{\xi}^{(1)}_{ij} \right) = 0 \tag{35}$$

Thus, keeping in mind that $\xi^{(1)}_{ij}$ and $\hat{\xi}^{(1)}_{ij}$ are not independent, we note that

$$\xi_{ij} \sim \xi^{(1)}_{ij} + \hat{\xi}^{(1)}_{ij} \tag{36}$$

Equation (36) implies that this approximation is equivalent to splitting link (i, j) into two links, where the first link has upper bound $\xi^{(1)}_{ij}$ and is devoted solely to cycle 1. The other link contains the residual capacity $\hat{\xi}^{(1)}_{ij}$, which may now be allocated to cycle 2 (let $\xi^{(2)}_{ij} = 0$ if cycle 2 does not use this link). Applying this procedure for each cycle in order, and for all links within each cycle, produces a new network where none of the cycles interact, illustrated in Figure 10. Now it is quite easy to find the expected flows around the network, giving us at least a bound on $\overline{Q}$.

We call this method *cyclic recourse* because realizations of random variables are handled by pushing flow around a cycle. The recourse available to handle a specific realization of ξ is to move flow around prespecified cycles in a fixed order. Furthermore, we place this particular solution approach into a class of recourse strategies we call *hierarchical recourse strategies* because we rank each cyclic response using (30) and greedily allocate capacity based on this rule. Finally, we simplified the calculation of the expected flows using a particular *variable splitting* technique that decomposed random variables into summable parts. Variable splitting is a special case of a class of procedures we refer to as *probabilistic decomposition*, whose function is only to simplify certain difficult probability calculations.

Note that it is not necessary that all the cycles of a network be enumerated. Any subset will provide a valid bound, although more complete sets will produce tighter bounds. A variety of strategies can be used to find cycles. One suggested by WALLACE is to solve (29) with $\xi = \xi^{max}$, with optimal flows x^{max}. Now we may find a conformal realization of x^{max} (ROCKAFELLAR[22]) which decomposes x^{max} into a (nonunique) set of cycle flows.

While WALLACE's approach is a powerful and creative concept, under certain circumstances it produces very weak bounds (FRANTZESKAKIS and POWELL[14]) More importantly, from the perspective of this paper, it falls short of realizing the full potential of the approach. First, a single conformal realization of a vector of link flows is likely to produce a relatively small number of cycles. Several procedures could be used to generate a wider range of cycles (for example, solve $\Psi(\xi)$ for many realiza-

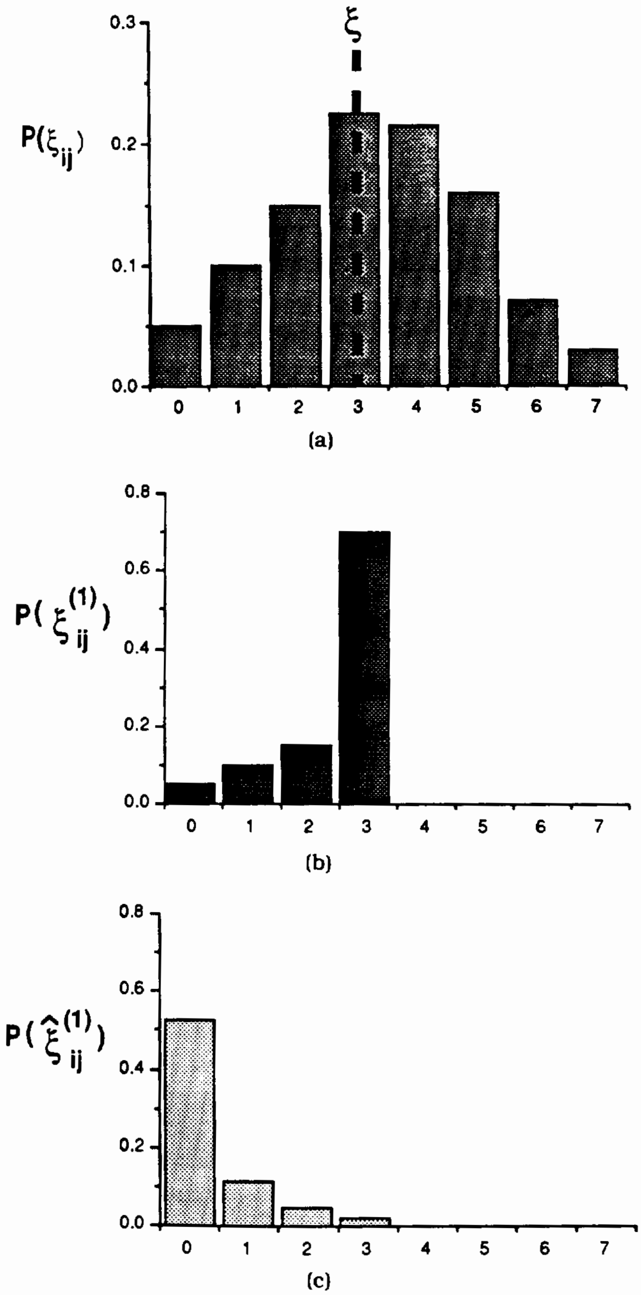


Fig. 9. (a) Distribution of a random arc capacity for an arc which shares two cycles. The maximum possible flow on the first cycle is 3. (b) Distribution of the capacity allocated to the first cycle, bounded by 3. (c) Distribution of the residual capacity for the second cycle.

tions of ξ and accumulate the cycles produced by each set of conformal realizations). Using this larger set M of cycles, again rank them using (30), creating a hierarchical set of cyclic recourse policies. Now, assign flow to the highest ranked cycle. Next, put as much flow on the second ranked cycle, taking into account the residual capacity from the first cycle. If ξ_{ij}^m is the residual capacity after cycles $1, \ldots, m-1$ have been assigned, then realizations of ξ_{ij}^m are accounted for by changes in flows on cycles strictly lower in the hierarchy.

The problem of working out the distributions of the link capacities ξ_{ij}^m after m cycles have been assigned is a separate (albeit nontrivial) issue. The variable splitting approach used by WALLACE is a highly restrictive approach which can result in very weak bounds. For example, it is likely that the highest ranked cycle on a link will take all the capacity of that link, leaving no residual capacity for lower ranked cycles. FRANTZESKAKIS and POWELL[14] present alternative approaches which are computationally efficient with significantly better results. The important issue is the notion of a predefined set of cycles which have a fixed hieararchy which determines the order in which each cycle is given the opportunity to receive flow.

This approach is used by WALLACE to provide a bound for the recourse function for a given vector $S(2)$. This bound does not provide a tool that would help with optimizing (9) since it does not yield an approximation of the recourse function. The concept of cyclic recourse, however, is a powerful one because it provides a far more flexible and realistic model of the true recourse function. Below, we extend this concept to develop procedures that can be used to actually optimize (9).

4.2. Path Recourse

For acyclic networks (which are more typical of true dynamic models) it is more natural to think of paths rather than cycles. Consider, for example, the network shown in Figure 11a, representing the second stage of a general network with random arc capacities. This network may be at least partially decomposed into a set of paths from the initial nodes (with supplies $S(2)$) to a supersink. A sample of such paths is depicted in Figure 11b. A variety of schemes can be formulated to develop this list of paths. Let d^m be the arc-length incidence vector denoting path m, and let $\beta_m = c^T d^m$ be the cost of each path (we use the same notation as for cyclic recourse because the concepts are not fundamentally different). Ranking the paths from least to most cost, we now have another hierarchical recourse strategy. As with cyclic recourse, we again have a difficult probability problem determining the expected flow on each path (the difficulty is that we need to know the *distribution* of the residual capacity after the first $m-1$ paths have been assigned). However, this is a separate task and different schemes can be devised to simplify these calculations. For example, we can use variable splitting to ensure that the path capacities are

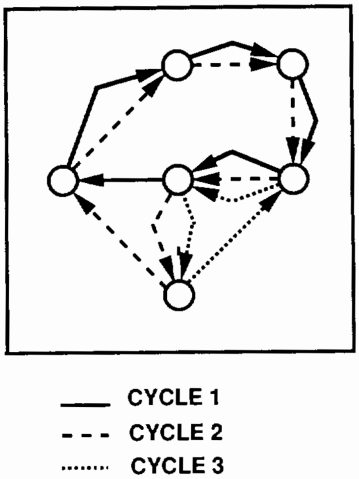


Fig. 10. Decomposition of network into three cycles over split links to eliminate bundling of the cycle flows.

independent random variables. Alternatively, if the problem has the structure of a transportation problem, then path recourse is equivalent to nodal recourse, which has a very simple probabilistic structure. As long as we can work out the probabilistic structure of the residual capacities on the paths, we can find expected path flows (which provides a bound on the recourse function) and, possibly, an approximation to the recourse function itself.

Path recourse can be viewed as a different form of cyclic recourse, since we can form a cycle with any path by simply adding the inventory links. In this interpretation, flow added to a path represents flow subtracted from an inventory link. In cyclic recourse, flow conservation is maintained at all nodes, which means the concept is only useful for developing bounds on the recourse function for a fixed first stage vector $x(1)$. Path recourse, however, can be used to approximate the recourse function $Q(S(1))$ by incrementally increasing $S_i(1)$ for a particular region i. Path recourse is used to approximately optimize how this incremental unit of flow is routed over the network in the second stage. This process can be used to build a convex, separable approximation of $Q(S(1))$ which can then be used in the optimization of the first stage problem.

4.3. Stochastic Path Recourse

Hierarchical cyclic and path recourse are effective concepts for a two-stage program but do not extend to N stages. Figure 12a depicts stages 1 through 4 of a four-stage network with random arc capacities. Assume that we wish to decompose this into paths such as that depicted in Figure 12b.

Let F_1 be the flow on this path, and let $\hat{\xi}^1$, ξ^2, ξ^3, and ξ^4 be the upper bounds on the first four

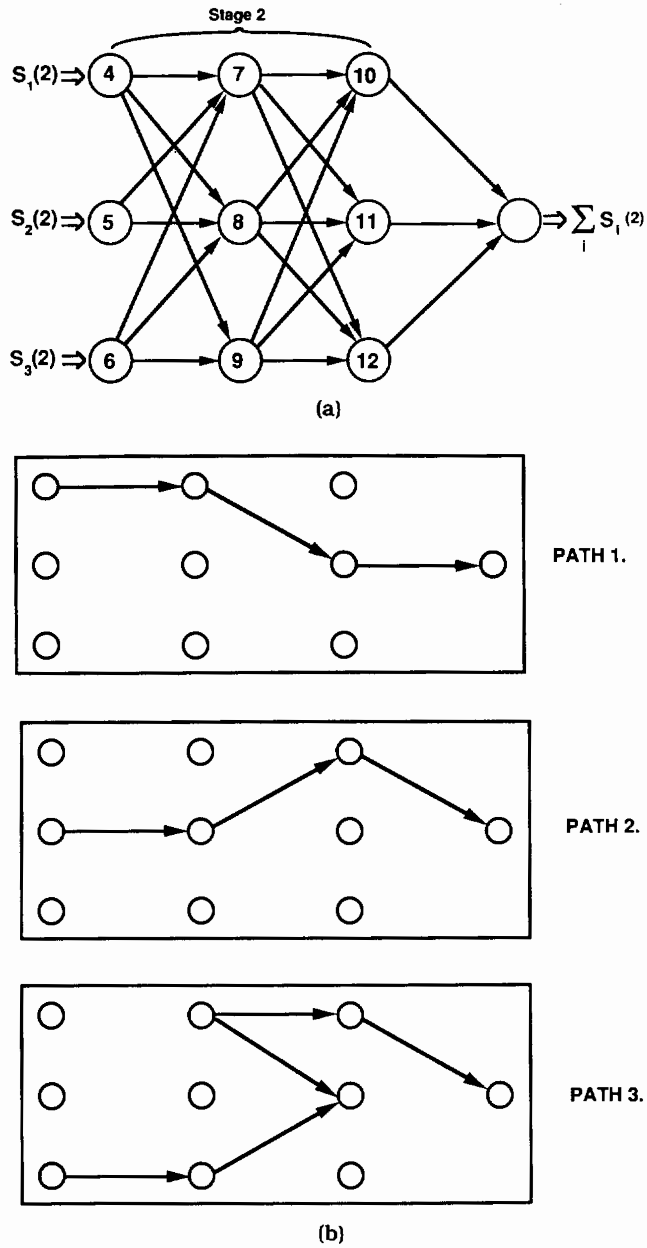


Fig. 11. (a) Transshipment network for the second stage of a two-stage problem. (b) Decomposition of second stage into path flows.

links of the path, as shown. $\hat{\xi}^1$ is the realization of the demand in the first stage. Then we might expect the flow on the path to be given by:

$$F_1 = \min\{\hat{\xi}^1, \xi^2, \xi^3, \xi^4\} \tag{37}$$

However, this expression implies that the flow on the first leg of the path must anticipate the bounds on the second and third legs which fall in later stages, thus violating the *nonanticipativity* condition of stochastic programming. Decisions made in period t can use only distributional information about later periods.

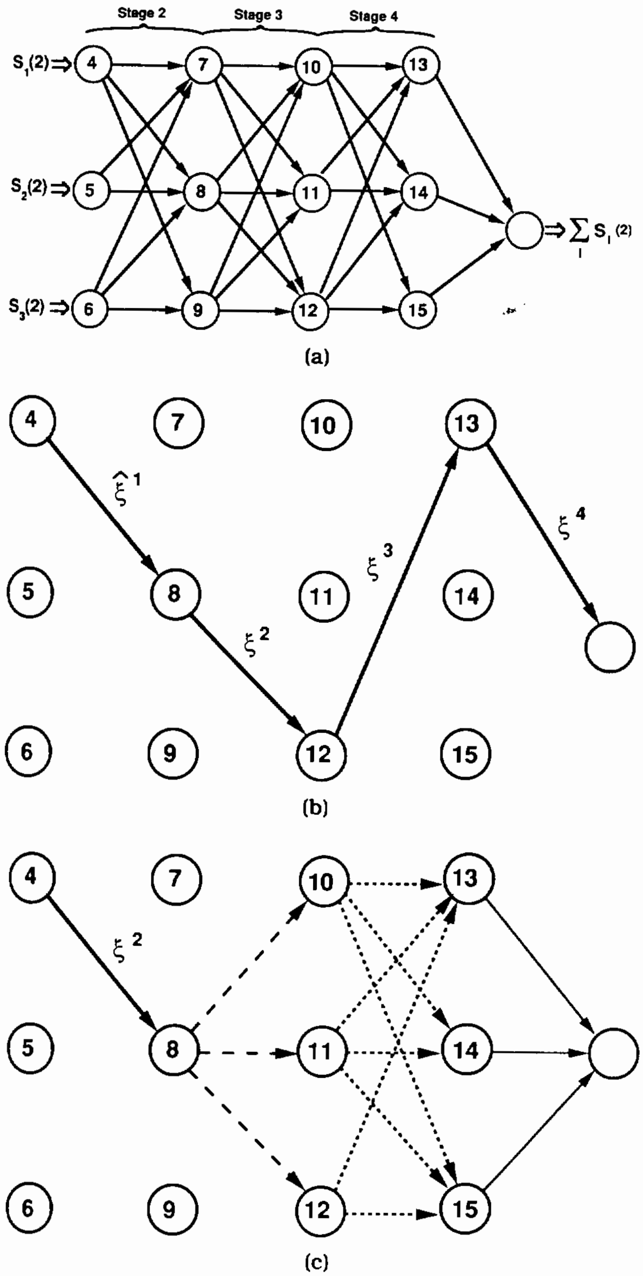


Fig. 12. (a) Three-stage stochastic network. (b) Single path through three-stage network. (c) Path splitting through a multi-stage network, showing multiple trajectories.

To circumvent this problem, we introduce *stochastic path recourse*, which decouples each stage in the following way. We begin by choosing a single path through stage 1 (in the case of the network in Figure 12a, this path is comprised of a single link, but in a more general network may move through several transshipment points). Let $R^m(1)$ be the vector of supplies we are trying to push over the m^{th} path in period 1, where $\sum_m R^m(1) = R(1)$, the initial input vector (if the single path departs from city i, then $R_j^m(1) = 0$ for $j \neq i$). Now let $S^m(1)$ be the vector of supplies created for stage 2 as a result of flows moving over the m^{th} path in stage 2. We next have to push $S^m(1)$ units of flow over paths in stage 2, which means we must have a range of options available in period 2, as well as in all later periods (we are not allowed to let realizations of link capacities in period 2 to affect the flows in period 1). We may choose a set of possible paths in period 2 which will then be hierarchically ranked, and flow will then be allocated to each of these paths in order. For the network in Figure 12a, this process is equivalent to nodal recourse (since each path is one link long). The resulting "stochastic path" is illustrated in Figure 12c. Note that this is a single path, and the cost of the path requires assigning probabilities to the possible paths in later stages. Again let β_m be a measure of the value of path m over all N stages. We would like β_m to be the expected cost of the path, but at this point we are unable to calculate the probability each link will be used since it depends on the other flows in the network. We could calculate path probabilities on an empty network, so that β_m would be the expected value of a path if it receives the first unit of flow, but it is only an approximate measure of the value of a path for subsequent units of flow. Alternatively, β_m might be the cost of the path in stage 1 plus an approximate value of another unit of flow at the appropriate node at the beginning of stage 2.

After a set of stochastic paths are enumerated, they are ranked on the basis of β_m, and flow is assigned to each path in order. In general, we will not be able to calculate the expected flow using each path because the probabilistic structure of the residual capacities after $m - 1$ units of flow have been assigned will be very complex. However, we may be able to impose restrictions on the structure of the stochastic paths, similar to those used in the various nodal recourse strategies. For example, the various schemes for ranking options in nodal recourse, whereby outbound links are kept in the same order for each additional unit of flow, represent a mechanism for simplifying these calculations.

A final remark on stochastic path recourse is suggested by Figure 12c. It is possible, under certain stationarity assumptions, that the path stage vector, $S^m(t)$, may possess a limiting distribution as $t \to \infty$. If this is the case, then it may be possible to eliminate the need to specify a fixed forecast horizon N, thereby avoiding classical truncation errors.

4.4. Using Hierarchical Recourse in Optimization

The work by WALLACE[26] as well as many others (DULA[10], BIRGE and WALLACE[3]) is oriented toward providing bounds for the recourse function $\overline{Q}(S(2))$ but does not directly address the problem of *optimizing* the original problem (3, 6, or 9). For this purpose, one approach is to develop an approximation of $\overline{Q}(S(2))$ which can then be used with classical optimization techniques. Simple recourse strategies easily yield analytical expressions. FRANTZESKAKIS and POWELL[12] show how nodal recourse can be used to develop a separable, piecewise linear approximation of $\overline{Q}(S(2))$, yielding an equivalent network formulation of the original optimization problem.

Stochastic path recourse can be used to directly optimize the network by incrementally loading the network with flow, providing what may be an effective heuristic for solving large stochastic programs. The approach is a kind of stochastic generalization of the BUSACKER and GOWEN[7] procedure for solving minimum cost flow problems, whereby flows are incrementally added to least-cost flow augmenting paths in a network.

5. SUMMARY

THE PURPOSE of this paper has been to expose a new class of restricted recourse strategies that may be used to solve stochastic dynamic networks. The goal of this research is to develop approximations of the recourse function that may be used to directly solve the original optimization problem subject to recourse. Up to now, simple recourse has been the primary mechanism for accomplishing this, whereas other research has focussed primarily on bounds for the recourse function. These restricted recourse strategies can be used to develop bounds (as was done by WALLACE[26]), but the real value is in developing approximations of the recourse function which allows the first stage problem to be solved in a simple and straightforward manner.

ACKNOWLEDGMENT

RESEARCH WAS supported by the National Science Foundation Presidential Young Investigator Award No. ECE-8481466.

REFERENCES

1. E. M. L. BEALE, G. B. DANTZIG AND R. D. WATSON, "A First Order Approach to a Class of Multi-Time Period Stochastic Programming Problems," *Math. Programming Stud.* **27**, 103–117 (1986).
2. C. BES AND S. P. SETHI, "Concepts of Forecast and Decision Horizons: Applications to Dynamic Stochastic Optimization Problems," *Math. Oper. Res.* **13**, 295–310 (1988).
3. J. R. BIRGE AND S. W. WALLACE, "A Separable Piecewise Linear Upper Bound for Stochastic Linear Programs," *SIAM J. Control Optim.* **26**, 1–14 (1988).
4. J. R. BIRGE AND R. J-B WETS, "Designing Approximation Schemes for Stochastic Optimization Problems, In particular for Stochastic Programs with Recourse," *Math. Prog. Stud.* **27**, 54–102 (1986).
5. J. R. BIRGE AND R. J-B WETS, "Sublinear Upper Bounds for Stochastic Programs with Recourse," *Math. Programming.* **43**, 131–149 (1989).
6. J.R. BIRGE, "Multistage Stochastic Planning Models using Piecewise Linear Response Functions," University of Michigan report, Department of Industrial and Operations Engineering, 1990.
7. R. G. BUSACKER AND P. J. GOWEN, "A Procedure for Determining a Family of Minimal Cost Network Flow Patterns," ORO Technical Report, **15**, Johns Hopkins University, 1961.
8. L. COOPER AND L. J. LEBLANC, "Stochastic Transportation Problems and Other Network Related Convex Problems," *Naval Res. Logist. Quart.* **24**, 327–337 (1977).
9. G. B. DANTZIG AND P. W. GLYNN, "Parallel Processors for Planning Under Uncertainty," Department of Operations Research, Stanford University, Report SOL-88-8R, 1988.
10. J. H. DULA, "An Upper Bound on the Expectation of Sublinear Functions of Multivariate Random Variables," Department of Operations Research and Engineering Management, Southern Methodist University, Technical Report 87-OR-9, 1987.
11. Y. ERMOLIEV, "Stochastic Quasigradient Methods," in *Numerical Methods in Stochastic Programming*, Y. Ermoliev and R. J-B Wets (eds.), Springer-Verlag, New York, 1988.
12. L. F. FRANTZESKAKIS AND W. B. POWELL, "A Successive Linear Approximation Procedure for Stochastic Dynamic Vehicle Allocation Problems," *Trans. Sci.* **24**, 40–57 (1990).
13. L. F. FRANTZESKAKIS AND W. B. POWELL, "Bounding Procedures for Dynamic Networks with Random Link Capacities, with Application to Stochastic Programming," (submitted to *Networks*) (1989a).
14. L. F. FRANTZESKAKIS AND W. B. POWELL, "An Improved Polynomial Bound for the Expected Network Recourse Function," Department of Civil Engineering and Operations Research, Princeton University, Working paper SOR-89-23, 1989b.
15. J. L. KENNINGTON AND R. V. HELGASON, *Algorithms for Network Programming*, Wiley & Sons, New York, 1980.
16. A. L. KORNHAUSER AND V. J. MASLANKA, "A Scenario Aggregation Approach to the Dynamic Vehicle Allocation Problem Under Uncertainty," Department of Civil Engineering and Operations Research, Princeton University, 1989.
17. J. M. MULVEY AND H. VLADIMIROU, "Solving Multi-

stage Stochastic Networks: An Application of Scenario Aggregation," Statistics and Operations Research Series, SOR- 88-1 (1989).

18. W. B. POWELL, Y. SHEFFI AND S. THIRIEZ, "The Dynamic Vehicle Allocation Problem with Uncertain Demands," *Ninth International Symposium on Transportation and Traffic Theory*, 1984.
19. W. B. POWELL, "A Stochastic Model of the Dynamic Vehicle Allocation Problem," *Transportation Sci.* **20**, 117–129 (1986).
20. W. B. POWELL, "A Comparative Review of Alternative Algorithms for the Dynamic Vehicle Allocation Problem," in *Vehicle Routing: Methods and Studies*, North Holland, New York, 1988.
21. W. B. POWELL, "An Operational Planning Model for the Dynamic Vehicle Allocation Problem with Uncertain Demands," *Transportation Res., Part B* **21**, 217–232 (1987).
22. T. R. ROCKAFELLAR, *Network Flows and Monotropic Programming*, Wiley & Sons, New York, 1984.
23. T. R. ROCKAFELLAR AND R. J-B WETS, "Scenarios and Policy Aggregation in Optimization Under Uncertainty," *Math. Oper. Res.* **16**, 119–147 (1991).
24. A. RUSZCZYNSKI, "A Linearization Method for Nonsmooth Stochastic Programming Problems," *Math. Oper. Res.* **12**, 32–49 (1987).
25. R. VAN SLYKE AND R. J-B WETS, "L-Shaped Linear Programs with Applications to Optimal Control and Stochastic Programming," *SIAM J. Appl. Math.* **17**, 638–663 (1969).
26. S. W. WALLACE, "A Piecewise Linear Upper Bound on the Network Recourse Function," *Math. Programming* **38**, 133–146 (1987).

(Received, July 1991; revision received January 1992, October 1992; accepted October 1992)