An Algorithm for Multistage Dynamic Networks with Random Arc Capacities, with an Application to Dynamic Fleet Management

Raymond K. Cheung

Iowa State University
Department of Industrial and Manufacturing Systems Engineering
Ames, IA 50010

Warren B. Powell

Princeton University
Department of Civil Engineering and Operations Research
Princeton, NJ 08544

September, 1995 (Revised)

Abstract

We consider the class of multistage dynamic networks with random arc capacities, a framework that is well suited to model dynamic fleet management problems. We propose a successive convex approximation approach that produces an approximation to the expected recourse function which captures the future effects of current decisions under uncertainty. This method decomposes the network in each stage into tree subproblems, whose expected recourse functions are easy to obtain. We also compare this method with two alternative methods on a set of dynamic fleet management problems. The numerical results show that this method is superior than the two alternative methods.

We consider the problem of managing a fleet of vehicles over space and time to serve current and forecasted demands. We assume the vehicles are homogeneous, and demands must be served at a specific instant in time, or be lost (with a loss in contribution to the company). Our problem can be viewed as a type of spatial, dynamic inventory management problem with reusable resources (vehicles). At any point in space and time, a vehicle may be assigned to satisfy a revenue generating activity, it may be repositioned empty to another point in space-time (presumably with higher revenue potential) or held in inventory.

We use as our model a dynamic network with random arc capacities, first introduced by Powell [11]. A market demand in the future is represented using an arc with a negative cost (representing the contribution, or reward, offered by a task) and an upper bound representing the market demand. When demands are forecasted, these upper bounds can be modeled as discrete random variables, creating a very large, multistage stochastic program, with considerable structure.

Dynamic networks have long proved to be a valuable technique for modeling dynamic fleet management problems. An early application is by Dantzig and Fulkerson [3] to minimize the number of tankers required to meet a given schedule. White and Bomberault [17] and White [16] formulate the railcar management problem using a deterministic dynamic network, and present an inductive version of the out-of-kilter algorithm for solving the resulting transshipment problem. Magnanti and Simpson [9] review a range of dynamic network models for fleet management, with special emphasis on decomposition methods for solving multicommodity flow problems. For a thorough survey of other applications of dynamic networks, see the review by Aronson [1].

The large majority of prior research has focused on deterministic networks. Jordan and Turnquist [6] and Powell [10] present a specialized method for approximating stochastic flows over dynamic networks, where the supplies of inventories in future time periods are approximated using continuous probability distributions characterized by the first two moments. Powell [11] provides the first formulation of this problem as a stochastic program, and Frantzeskakis and Powell [5] present a method called the Succes-

sive Linear Approximation Procedure (SLAP) for approximating the expected recourse function for the multistage dynamic network with random arc capacities.

This research is an extension of Powell and Cheung [13] which introduces the notion of *tree recourse* and Powell and Cheung [12] which addresses two-stage problems using a procedure called *network recourse decomposition*. The numerical experiments in [12] show that the network recourse decomposition approach can produce a tight lower bound for the expected recourse function. More importantly, the results show that this approach can produce an accurate approximation of the expected recourse function.

This paper shows how the network recourse decomposition can be extended to solving multistage networks. It also represents an alternative approach to SLAP for solving the same class of problems. We generalize SLAP by using convex, instead of linear, approximations of the expected recourse function. Furthermore, this new approach can be applied to a wider class of problems where SLAP may not be applied. Finally, the experimental results support even more strongly the conclusions in [5] that solving an approximation of the stochastic problem can produce better results than an optimal solution of a deterministic approximation.

We begin in section 1 by providing a stochastic programming formulation for dynamic networks with random arc capacities. Next, section 2 presents the successive convex approximation method for solving this class of problems. This method is a decomposition method involving tree subproblems whose solution can be obtained quite efficiently by the method of [13]. In section 3, we review two alternative methods for the same class of problems. Section 4 compares the performance of the new method with the alternative methods on a set of test problems, which include real dynamic fleet management problems. The results in section 4 show the superiority of the new method over the alternative methods. Finally, section 5 gives some concluding remarks. For completeness, the appendix provides an example to illustrate the approximation method.

1 Problem Formulation

Consider a dynamic fleet management problem as an N-stage dynamic network where arc capacities are used to model random market demands. In such a network, a node represents a particular city at a particular time, and an arc represents the movement (loaded, empty or inventory) of vehicles between a pair of cities in a particular stage. In each stage, we need to determine the positioning of a fleet of vehicles while considering the downstream impacts of the current moves under future uncertainty. These impacts are commonly measured by the expected total profits (or total costs) of the decisions made in later stages. Therefore, in stage t, our objective is to maximize both the total profit (revenue minus cost) in stage t and the expected profits in later stages.

Before giving the mathematical formulation, we first state our assumptions:

A1. Travel time between each pair of cities is exactly one stage(period);

A2. Decisions are made only once per time period;

A3. There is always an uncapacitated link out of a city with zero cost;

A4. The random market demands are independent discrete random variables;

A5. All unsatisfied demands are considered lost.

The first assumption is made only to simplify our presentation and is not a model restriction. As we see later in section 4, we may allow multi-period travel times. The second assumption avoids the details of the decisions made within a time stage, keeping the network size manageable. The third assumption says that vehicles can be held in a city over time with zero cost. Such an assumption ensures the feasibility of the problem by having an uncapacitated link (inventory link) out of each node in the network. The rationale behind the fourth assumption is that in a real world application, a common carrier often serves an enormous number of independent customers. The last assumption rules out the case of backlogging which may significantly complicate the problem. Finally, by convention, we write this problem as a minimization problem.

Let $\tilde{\xi}$ be a random vector (market demands) defined over a probability space $(\Omega, \mathcal{F}, P)$ with elementary outcomes $\Omega = \{\omega : \omega = (\omega_2, ..., \omega_t, ..., \omega_N)\}$ where ω_t is the outcome in

stage t (we assume no uncertainty in stage 1). Define :

$\mathcal{N}^t$ = set of cities at time t,

$\mathcal{A}^t$ = set of arcs originating at time t,

$\tilde{\xi}_{ij}^t$ = (random) market demand for vehicles moving from city i to city j starting at time t

ξ_{ij}^t = a realization of $\tilde{\xi}_{ij}^t$,

$\tilde{x}_{ij}^t$ = flow of loaded vehicles from city i to city j starting at time t,

x_{ij}^t = flow of loaded vehicles from city i to city j for realization ξ_{ij}^t,

$\tilde{u}_{ij}^t$ = flow of empty vehicles from city i to city j starting at time t,

u_{ij}^t = flow of empty vehicles from city i to city j for realization ξ_{ij}^t,

r_{ij}^t = net profit per loaded vehicle from city i to city j at time t,

c_{ij}^t = cost per vehicle for moving from from city i to city j at time t,

S_j^t = number of vehicles available in city j at time $t+1$

= internal supply to node $j \in \mathcal{N}^{t+1}$ resulting from decisions made in stages t,

R_i^t = external supply of vehicles to city i at time t.

Our convention is that a variable with no subscript represents a vector. For example, S^t is the vector of supplies to nodes in stage $t+1$ (that is, the vehicles available at time $t+1$).

By assumption (A.1), we know that there is no pure transshipment node in this network since, in stage t, vehicles enter a city i at time t and end up in another city j at time $t+1$. Therefore, all nodes in the dynamic network can be treated as supply nodes, meaning that:

$$\mathcal{N} = \bigcup_{t=1}^{N+1} \mathcal{N}^t, \quad \text{and} \quad \mathcal{N}^t \cap \mathcal{N}^{t'} = \emptyset, \quad \text{if } t \neq t'. \tag{1}$$

The decision variables in stage t are actually dependent on the history up to and including stage t. However, the internal supply S^{t-1} is a state vector which communicates the past information with the decision making in stage t. Therefore, x^t can be written as $x(t, \omega_2, \ldots, \omega_{t-1}, \omega_t)$ or $x(t, S^{t-1}, \omega_t)$. Nevertheless, we retain the notation x^t for brevity. Finally, we use $E_t(\cdot)$ to denote an expected value relative to $\tilde{\xi}^t$.

The mathematical formulation of a dynamic fleet management problem is given by:

$$\min_{x^1,u^1,S^1} \quad -r^{1T}x^1 + c^{1T}u^1 + E_2Q_2(S^1,\tilde{\xi}^2) \tag{2}$$

subject to:

$$\sum_{j\in\mathcal{N}^2}(x^1_{ij}+u^1_{ij}) = R^1_i \qquad \forall\, i\in\mathcal{N}^1 \tag{3}$$

$$\sum_{i\in\mathcal{N}^1}(x^1_{ij}+u^1_{ij}) - S^1_j = 0 \qquad \forall\, j\in\mathcal{N}^2 \tag{4}$$

$$0 \le x^1_{ij} \le \xi^1_{ij} \qquad \forall\, i\in\mathcal{N}^1,\ j\in\mathcal{N}^2 \tag{5}$$

$$u^1_{ij} \ge 0 \qquad \forall\, i\in\mathcal{N}^1,\ j\in\mathcal{N}^2 \tag{6}$$

where for a given supply vector S^1 and a particular realization ξ^2, $Q_2(S^1,\xi^2)$ is the value of a minimization problem which is defined recursively as follows:

$$Q_t(S^{t-1},\xi^t) = \min_{x^t,u^t,S^t} -r^{tT}x^t + c^{tT}u^t + E_{t+1}Q_{t+1}(S^t,\tilde{\xi}^{t+1}) \tag{7}$$

subject to:

$$\sum_{j\in\mathcal{N}^{t+1}}(x^t_{ij}+u^t_{ij}) = S^{t-1}_i + R^t_i \qquad \forall\, i\in\mathcal{N}^t \tag{8}$$

$$\sum_{i\in\mathcal{N}^t}(x^t_{ij}+u^t_{ij}) - S^t_j = 0 \qquad \forall\, j\in\mathcal{N}^{t+1} \tag{9}$$

$$0 \le x^t_{ij} \le \xi^t_{ij} \qquad \forall\, i\in\mathcal{N}^t,\ j\in\mathcal{N}^{t+1} \tag{10}$$

$$u^t_{ij} \ge 0 \qquad \forall\, i\in\mathcal{N}^t,\ j\in\mathcal{N}^{t+1} \tag{11}$$

where $t = 2,3,\ldots,N$ and $Q_{N+1}(\cdot,\cdot) = 0$.

The assumption $Q_{N+1}(\cdot,\cdot) = 0$ means that all vehicles arriving at time beyond the planning horizon are considered to have a salvage value of 0. In fact, the study of end effects deserves further research and is beyond the scope of this study.

The problem defined by (2) – (11) is a typical multistage stochastic program with network recourse (see Wallace[15]) where (2) – (6) is called the *first stage* problem and (7) – (11) is called the *stage t recourse* problem. The expectation functional $E_{t+1}Q_{t+1}(S^t,\tilde{\xi}^{t+1})$ is called the *stage t expected recourse function.*

The nested expectation functionals which appear in the objective functions make the problem extremely complicated. In order to solve the first stage problem, we need to determine the expected recourse function $E_2Q_2(S^1, \tilde{\xi}^2)$ as a function of S^1. However, in stage 2, again we need to evaluate the function $E_3Q_3(S^2, \tilde{\xi}^3)$ and so forth. Except for a few special cases, exact calculation of the expected recourse functions is numerically intractable, even for moderate problems with small number of stages. Therefore, we rely on approximation methods. An overview of various approximation schemes for general stochastic programs can be found in Birge and Wets[2] and Kall *et al.* [7] and the references cited there. In our problem, we have a very special structure in each stage, namely a transportation problem with random arc capacities where no specific amount of flow must be shipped to the demand points. This special structure allows us to develop specialized method to approximate the expectation functionals using some simpler functions. Therefore, we do not consider general stochastic programming techniques in this paper.

For simplifying our presentation in the rest of this paper, we assume that we have only a link between a pair of nodes. We can satisfy this assumption easily by adding additional nodes and links to the network. If x_{ij}^t represents loaded movement, then $c_{ij}^t < 0$ and $\tilde{\xi}_{ij}^t$ is a finite discrete random variable. If x_{ij}^t represents empty movement (we use u_{ij} previously), then $c_{ij}^t \geq 0$ and $\tilde{\xi}_{ij}^t = \infty$. Finally, we consider the underlying problem as a minimization problem.

2 Successive convex approximation methodology

Powell and Cheung[13] introduces a specialized algorithm to find the expected objective value of a directed tree with random arc capacities, as a function of the scalar supply. Based on this result, a backward recursion procedure is proposed to find the expected recourse function exactly for a multistage stochastic network where each stage consists of entirely independent trees. The procedure is quite efficient: exact expected recourse functions for trees with several thousand random variables can be obtained in a few

seconds. Our ability to solve tree problems motivates the use of decomposition when we encounter a harder problem. Powell and Cheung[12] studies a class of more general two-stage stochastic networks. It develops an iterative procedure to provide approximations of the expected recourse function by structurally decomposing the underlying network into tree subproblems. The expected objective values of these trees can be found parametrically as functions of supplies, using the algorithm in [13]. In this paper, we extend these results to general, multistage networks. We develop a backward recursion method to successively provide convex approximations of the expected recourse functions in each stage, where a structural decomposition is involved. We refer to it as the *Successive Convex Approximation Method* (SCAM).

In SCAM, we successively generate a sequence of functions starting from the last stage, that is $\hat{Q}_N(S^{N-1}),\ \cdots,\ \hat{Q}_{t+1}(S^t),\ \hat{Q}_t(S^{t-1}),\ \cdots,\ \hat{Q}_2(S^1)$ where the function $\hat{Q}_{t+1}(S^t)$ is used to approximate the expected total cost for stages from t to N as a parametric function of S^t (internal supply to stage $t+1$). Furthermore, each function $\hat{Q}_{t+1}(S^t)$ is convex and has the form of

$$\hat{Q}_{t+1}(S^t) = \sum_{j \in \mathcal{N}^{t+1}} \hat{Q}_{t+1,j}(S^t_j) + K^t \tag{12}$$

with the properties:

(P1) $\hat{Q}_{t+1,j}(S^t_j)$ is piecewise linear and convex

(P2) $\hat{Q}_{t+1}(S^t)$ is a lower bound of $E_{t+1}Q_{t+1}(S^t, \tilde{\xi}^{t+1})$

(P3) K^t is some constant (it can depend on t).

In section 2.1, we show how to obtain $\hat{Q}_t(S^{t-1})$ by given an approximation $\hat{Q}_{t+1}(S^t)$. In section 2.2, we summarize the steps of SCAM.

2.1 Approximation by decomposition

The process of obtaining $\hat{Q}_t(S^{t-1})$ for a given approximation $\hat{Q}_{t+1}(S^t)$ (with the form of (12)) consists of the following steps:

1. Approximate the stage t problem by replacing $E_{t+1}Q_{t+1}(S^t, \tilde{\xi}^{t+1})$ (the true expected recourse function) with the approximation $\hat{Q}_{t+1}(S^t)$. Let us denote the modified stage t problem as $[MP_t]$.

2. Represent $\hat{Q}_{t+1}(S^t)$ by a set of links such that $[MP_t]$ is a network with random arc capacities (that is, augment the original stage t network with additional links).

3. Reformulate $[MP_t]$ as a multicommodity flow problem with bundle constraints.

4. Use a relaxation method to produce a function, denoted by $\hat{Q}_t(S^{t-1})$, that approximates the expected cost of $[MP_t]$ as a function of S^{t-1}.

We show later that the function $\hat{Q}_t(S^{t-1})$ also has the form of (12). This function will then be used to augment the stage $t-1$ for obtaining $\hat{Q}_{t-1}(S^{t-2})$. We describe in detail of the network augmentation (steps 1 and 2), the reformulation (step 3), and the relaxation (step 4) in sections 2.1.1, 2.1.2 and 2.1.3 respectively.

2.1.1 Network augmentation

By replacing $E_{t+1}Q_{t+1}(S^t, \tilde{\xi}^{t+1})$ with the approximation $\hat{Q}_{t+1}(S^t)$, the modified stage t problem becomes:

$$[MP_t] \quad \tilde{Q}_t(S^{t-1}, \xi^t) = \min_{x^t, S^t} \left\{ c^{tT}x^t + \sum_{j \in \mathcal{N}^{t+1}} \hat{Q}_{t+1,j}(S_j^t) + K^t \text{ subject to: (8) - (11)} \right\}. \tag{13}$$

where we let $\tilde{Q}_t(S^{t-1}, \xi^t)$ be the value of problem $[MP_t]$. From this formulation, we see that $\hat{Q}_{t+1,j}(S_j^t)$ has an intuitive interpretation: it captures the expected marginal value of having S_j^t vehicles in city j at time $t+1$ (that is, S_j^t units of flow in node $j \in \mathcal{N}^{t+1}$). Assume these vehicles are ordered such that the first one takes the most profitable movement, the second takes the second best and so forth. Let q_{jk}^{t+1} be the expected marginal contribution of the k^{th} vehicle. Then, when $S_j^t = s$, $\hat{Q}_{t+1,j}(S_j^t)$ can be written as:

$$\hat{Q}_{t+1,j}(s) = \sum_{k=1}^{s} q_{jk}^{t+1}. \tag{14}$$

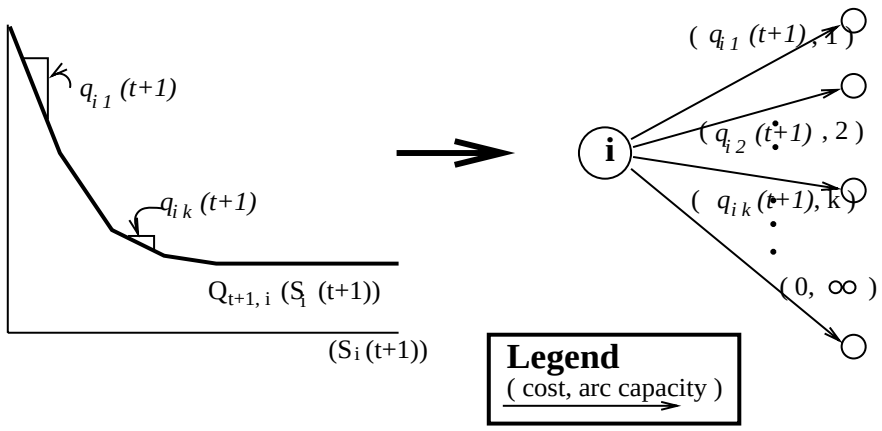

Figure 1: Representation of convex, piecewise linear function by a set of links

As shown later in proposition 2.1, $\hat{Q}_{t+1,j}(s)$ is convex on s, that is

$$q_{jk}^{t+1} \leq q_{jk'}^{t+1} \quad \text{if} \quad k < k'. \tag{15}$$

This inequality reflects the diminishing return of the incremental unit of supply. In other words, the first vehicle available in city j should have a higher expected marginal value (or lower expected marginal cost) than the second one.

Since each function, $\hat{Q}_{t+1,j}(S_j^t)$, is convex and piecewise linear, it can be represented by a set of deterministic links, which we call "recourse" links (see figure 1). The k^{th} recourse link out of node $j \in \mathcal{N}^{t+1}$, except the last one, has an upper bound of one and a link contribution of q_{jk}^{t+1}. The last recourse link, with zero cost and no upper bound, is added for the case where a large number of vehicles are being sent to node j. As a result, the impacts of current decisions are captured by the recourse links. Mathematically speaking, we let

y_{jk}^{t+1} = flow on the k^{th} recourse link out of node $j \in \mathcal{N}^{t+1}$ for realization ω_t,
$\mathcal{Z}_j$ = the set of recourse links out of node j
$\mathcal{Z}_j'$ = $\mathcal{Z}_j$ excluding the feasibility link.

Then, problem (13) is equivalent to:

$$\tilde{Q}_t(S^{t-1}, \xi^t) = \min_{x^t, y^t} \sum_{i \in \mathcal{N}^t} \sum_{j \in \mathcal{N}^{t+1}} \left(c_{ij}^t x_{ij}^t \; + \; \sum_{k \in \mathcal{Z}_j} q_{jk}^{t+1} \; y_{jk}^{t+1} \right) + K^t \tag{16}$$

subject to:

$$\sum_{j \in \mathcal{N}^{t+1}} x_{ij}^t = \; S_i^{t-1} \qquad \forall \; i \in \mathcal{N}^t \tag{17}$$

$$\sum_{i \in \mathcal{N}^t} x_{ij}^t - \sum_{k \in \mathcal{Z}_j} y_{jk}^{t+1} = \; 0 \qquad \forall \; j \in \mathcal{N}^{t+1} \tag{18}$$

$$0 \; \leq \; y_{jk}^{t+1} \leq \; 1 \qquad \forall \; j \in \mathcal{N}^{t+1}, k \in \mathcal{Z}_j' \tag{19}$$

$$0 \; \leq \; x_{ij}^t \; \leq \; \xi_{ij}^t \qquad \forall \; i \in \mathcal{N}^t, \; j \in \mathcal{N}^{t+1} \tag{20}$$

Clearly, problem (16) – (20) is an acyclic network since we augment the original stage t problem by adding the recourse links.

2.1.2 Multicommodity network formulation

Once again, our goal is to approximate the expected value of problem (16) – (20) as a separable function of the supply (meaning that the number of vehicles available in each city). Therefore, we would like to decompose this network by their origins so that each component of the separable function can be obtained individually. This leads to the notion of multicommodity flows where a commodity is defined as follows:

Definition 2.1 *A unit of flow originating from node $i \in \mathcal{N}^t$ is defined as* commodity i.

For example, consider the two-origin network depicted in figure 2. The flow entering node 1 is considered as commodity 1 and the flow entering node 2 is considered as commodity 2. That is, the flows in the network are differentiated by their origins. We let

$y_{jk}^{t+1,i}$ = flow of commodity i on the k^{th} recourse link out of node $j \in \mathcal{N}^{t+1}$ for realization ξ^t.

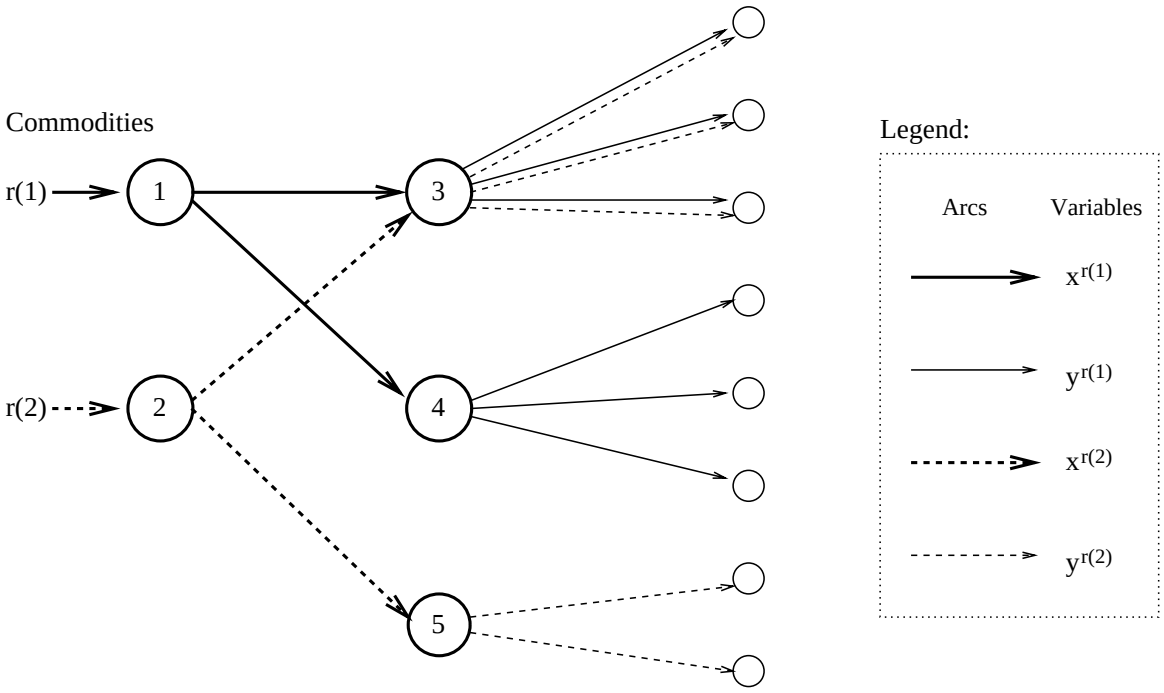

Figure 2: Two-level trees.

Therefore, the multicommodity formulation of problem (16) – (20) can be written as:

$$\tilde{Q}_t(S^{t-1}, \xi^t) = \min_{x^t, y^t} \sum_{i \in \mathcal{N}^t} \sum_{j \in \mathcal{N}^{t+1}} \left(c_{ij}^t x_{ij}^t + \sum_{k \in \mathcal{Z}_j} q_{jk}^{t+1} y_{jk}^{t+1,i} \right) + K^t \tag{21}$$

subject to:

$$\sum_{j \in \mathcal{N}^{t+1}} x_{ij}^t = S_i^{t-1} \quad \forall\, i \in \mathcal{N}^t \tag{22}$$

$$x_{ij}^t - \sum_{k \in \mathcal{Z}_j} y_{jk}^{t+1,i} = 0 \quad \forall\, i \in \mathcal{N}^t,\ j \in \mathcal{N}^{t+1} \tag{23}$$

$$0 \le x_{ij}^t \le \xi_{ij}^t \quad \forall\, i \in \mathcal{N}^t,\ j \in \mathcal{N}^{t+1} \tag{24}$$

$$0 \le y_{jk}^{t+1,i} \le 1 \quad \forall\, i \in \mathcal{N}^t,\ j \in \mathcal{N}^{t+1}, k \in \mathcal{Z}_j' \tag{25}$$

$$\sum_{i \in \mathcal{N}^t} y_{jk}^{t+1,i} \le 1 \quad \forall\, j \in \mathcal{N}^{t+1}, k \in \mathcal{Z}_j' \tag{26}$$

In this formulation, constraints (25) are redundant since (26) implies (25). However, if we relax the constraints (26), then this problem becomes separable by commodity with the looser constraints (25). Furthermore, constraints (22) – (25) indicate that each commodity i follows a two-level tree which is rooted at supply node $i \in \mathcal{N}^t$. Note that the links in the second level of these trees may be shared by different trees (see the links out of node 3 in figure 2), inducing the *bundle constraints* (26). In the next section, we present a method to handle these bundle constraints.

2.1.3 Relaxation

Throughout this section, we are only concerned with the stage t problem. Hence, we suppress the time index t without ambiguity by writing: $S = S^{t-1}$, $q = q^{t+1}$, $x_{ij} = x_{ij}^t$, $y_{jk}^i = y_{jk}^{t+1,i}$, etc. Furthermore, we let T_i be the feasible set for commodity i without the bundle constraints (26), that is:

$$T_i = \left\{ x_{ij}, y_{jk}^i \mid \text{(22) – (25) are satisfied} \right\}.$$

Essentially, the underlying structure of T_i is a tree.

For a fixed supply vector S and a given realization ξ (in stage t), assume that we relax (26) using a set of deterministic multipliers $\lambda_{jk} \ge 0$. Then, the *Lagrangian relaxation* of

problem (21) – (26) is:

$$L(S,\lambda) = \min_{\{x,y^i\}\in T_i} \sum_i \left(c^T x + q^T y^i\right) + \lambda^T \left(\sum_i y^i - \underline{1}\right) + K \tag{27}$$

where $\underline{1}$ is the vector of 1. Notice that $L(S,\lambda)$ is defined for a particular realization. Therefore, by choosing an "appropriate" vector $\lambda \in \Re_+^n$, where $n = \sum_j |\mathcal{Z}_j'|$ (the number of coupling links), and by taking expectations on both sides and rearranging the terms, we obtain an approximation of $E\tilde{Q}(S,\tilde{\xi})$ (the expected objective value of problem (21) – (26)). Let $\hat{Q}(S,\lambda)$ be such an approximation. Then,

$$\hat{Q}(S,\lambda) = E\left\{\min_{\{x,y^i\}\in T_i} \sum_i \left(c^T x + q^T y^i\right) + \lambda^T \left(\sum_i y^i - \underline{1}\right) + K\right\} \tag{28}$$

$$= \sum_{i\in\mathcal{N}^t} E\left\{\min_{\{x,y^i\}\in T_i} \left(c^T x + (q+\lambda)^T y^i\right)\right\} - \lambda^T \underline{1} + K \tag{29}$$

Now, let

$$\hat{Q}_i(S_i,\lambda) = E\left\{\min_{\{x,y^i\}\in T_i} \left(c^T x + (q+\lambda)^T y^i\right)\right\} \tag{30}$$

then, we can write $\hat{Q}(S,\lambda)$ as

$$\hat{Q}(S,\lambda) = \sum_{i\in\mathcal{N}^t} \hat{Q}_i(S_i,\lambda) - \lambda^T \underline{1} + K \tag{31}$$

which has the form of (12).

The embedded minimization problem in (30) is a directed tree with random arc capacities. Therefore, we can apply the algorithm in [13] to find $\hat{Q}_i(S_i,\lambda)$ parametrically as a function of scalar supply S_i. The slopes of these functions are used to measure the expected marginal value for the supply to stage t (recall that we have used the q_{jk}^{t+1} coefficients to approximate the expected marginal value for the supply to stage $t+1$). In the context of dynamic fleet management, these functions estimate the expected marginal contributions for having additional vehicles in city i. The following proposition shows the convexity of our approximation.

Proposition 2.1 *The function $\hat{Q}(S,\lambda)$ defined by (31) is convex on S.*

Proof: Each function $\hat{Q}_i(S_i)$ is the expected recourse function of a tree problem, which is convex (see Van Slyke and Wets[14]). Since the sum of convex functions and a constant is a convex function, the result follows. □

The bottom line of this method is to choose an appropriate vector of multipliers λ. The following propositions, which have appeared in [12] suggest one method to determine λ.

Proposition 2.2 *For any given λ,*

$$\hat{Q}(S,\lambda) \leq E\tilde{Q}(S,\tilde{\xi})$$

where $\tilde{Q}(S,\xi)$ is defined by (21) and $\hat{Q}(S,\lambda)$ is defined by (31).

Proposition 2.3 *Let*

$$\bar{y}_{jk} = \sum_{i\in\mathcal{N}^t} Ey^i_{jk}, \tag{32}$$

$$g_{jk}(S,\lambda) = \bar{y}_{jk} - 1 \qquad \forall\ j \in \mathcal{N}^{t+1}, k = 1,...,\mathcal{Z}'_j \tag{33}$$

Then, $g(S,\lambda) \in \partial_\lambda \hat{Q}(S,\lambda)$, where $\partial_\lambda \hat{Q}(S,\lambda)$ is the set of subgradients of $\hat{Q}(S,\lambda)$ with respect to λ.

Proposition 2.2 shows that $\hat{Q}(S,\lambda)$ is a lower bound of the expected objective value of modified problem $[MP_t]$ (that is, (21) – (26)). We now show that $\hat{Q}(S,\lambda)$ is also a lower bound of the original expected total cost function $EQ(S,\tilde{\xi})$.

Proposition 2.4 *For any given λ,*

$$\hat{Q}(S,\lambda) \leq EQ(S,\tilde{\xi})$$

where $Q(S,\xi)$ is the original stage t problem.

Proof: Consider again the original stage t problem:

$$[P_t] \quad \tilde{Q}_t(S^{t-1},\xi^t) = \min_{x^t,S^t}\left\{c^{t^T}x^t + E_{t+1}Q_{t+1}(S^t,\tilde{\xi}^{t+1}) \text{ subject to: (8) – (11)}\right\}. \tag{34}$$

Since we assume $\hat{Q}_{t+1}(S^t) \leq E_{t+1}Q(S^t,\tilde{\xi}^{t+1}$ (see property (P2) of the approximation), we know that for any given (x^t,S^t),

$$c^{t^T}x^t + E_{t+1}Q_{t+1}(S^t,\tilde{\xi}^{t+1}) \geq c^{t^T}x^t + \hat{Q}_{t+1}(S^t_j)$$

Let (x^{*t}, S^{*t}) be the optimal solution for problem $[P_t]$, then we have

$$c^{t^T} x^{*t} + E_{t+1} Q_{t+1}(S^{*t}, \tilde{\xi}^{t+1}) \geq c^{t^T} x^{*t} + \hat{Q}_{t+1}(S^{*t}) \tag{35}$$

Conversely, let $(\hat{x}^t, \hat{S}^t)$ be the optimal solution for problem $[MP_t]$, then we get

$$c^{t^T} x^{*t} + \hat{Q}_{t+1,j}(S_j^{*t}) \geq c^{t^T} \hat{x}^t + \hat{Q}_{t+1}(\hat{S}^t) \tag{36}$$

We now combine equations (35) and (36) and take the expectation of both sides of the resulting inequality, we get

$$E_t \tilde{Q}_t(S^{t-1}, \tilde{\xi}^t) \leq E_t Q_t(S^{t-1}, \tilde{\xi}^t)$$

By proposition 2.2, the result follows. □

Proposition 2.3 shows that $g(S, \lambda)$ defined in (33) is a valid subgradient for $\hat{Q}(S, \lambda)$ (a lower bound for both the modified and the original stage t problems). Consequently, we can find λ^* which maximizes $\hat{Q}(S, \lambda)$ using subgradient optimization. In a typical subgradient method, we iteratively move the current solution λ^l along the direction of the subgradient $g(S, \lambda)$ with a small amount. Mathematically, we have for iteration l:

$$\lambda^{l+1} = \max\left\{\lambda^l + \rho_l \cdot g(S, \lambda^l),\ 0\right\} \tag{37}$$

where ρ_l are the step sizes and $g(S, \lambda^l)$ are obtained using equation (33).

For a given S, let $\hat{Q}(S, \lambda^l)$ be the value of $\hat{Q}(S, \lambda)$ when $\lambda = \lambda^l$. A common choice of step sizes is (see, for example [8]):

$$\rho_l = \gamma_l \cdot \max\left\{\frac{Q^U(S) - \hat{Q}(S, \lambda^l)}{||g(S, \lambda^l)||^2}, 0\right\}, \tag{38}$$

where $\gamma_l \in [0, 2]$ and $Q^U(S)$ is an statistical estimate of $E\tilde{Q}(S, \xi)$ for a given supply vector S, which can be obtained via Monte Carlo methods. Note that $\bar{y}_{jk}$ in (32) represents the total expected flow on link (j, k) among all commodities. This amount is simply a byproduct while finding the expected recourse function for each individual tree using the algorithm as described in [13].

2.2 Algorithm

In summary, the pseudo-algorithm for SCAM is as follows:

ALGSCAM

Step 1 *Compute the approximations in stage N, $\hat{Q}_{N,i}(S_i), \forall i \in \mathcal{N}^N$ by solving one-level trees.*

Step 2 *Augment stage $N-1$ problem by the recourse links corresponding to $\hat{Q}_{N,i}(S_i)$.*

Step 3 *For $t = N-1$ down to 2*

- *Decompose the augmented stage t problem by origins, producing tree subproblems.*
- *Set $l = 0$.*
- *Repeat*
  1. *Set $l = l + 1$;*
  2. *Obtain $\hat{Q}_{t,i}(S_i, \lambda^l)$ for each tree.*
  3. *Compute expected link flows (using (50) in the Appendix).*
  4. *Compute subgradient $g(S, \lambda^l)$ using (33).*
  5. *Update λ using (37).*

  until required accuracy is met.
- *Augment stage $t-1$ problem by the recourse links corresponding to $\hat{Q}_{t,i}(S_i, \lambda^l)$.*

Step 4 *Solve the augmented first stage network*

To have a clearer picture of this method, we illustrate a step of it using figure 3. The $t-1$, t and $t+1$ recourse problems of a multistage dynamic network are shown in figure 3a. Assume we are given the approximations $\hat{Q}_{t+1,j}(S_j^t), j \in \mathcal{N}^{t+1}$, as shown in figure 3b. First, we represent these functions by the recourse links. Second, we add them to the original stage t recourse problem, forming a larger network, given in figure 3c. Third, we decompose the flows in this augmented stage t network by their origins such that the flow from each origin follows a two-level tree, as depicted in figure 3d. We then obtain the expected recourse function for each tree. Clearly, these trees are overlapping. If our decisions of pushing flows over the trees are made independently, then the capacity constraints for the links in the second level can be violated. Hence, we employ the stochastic relaxation method (see (29)) to decouple the bundle constraints, which results in a modification of the link costs. Finally, we apply the the algorithm

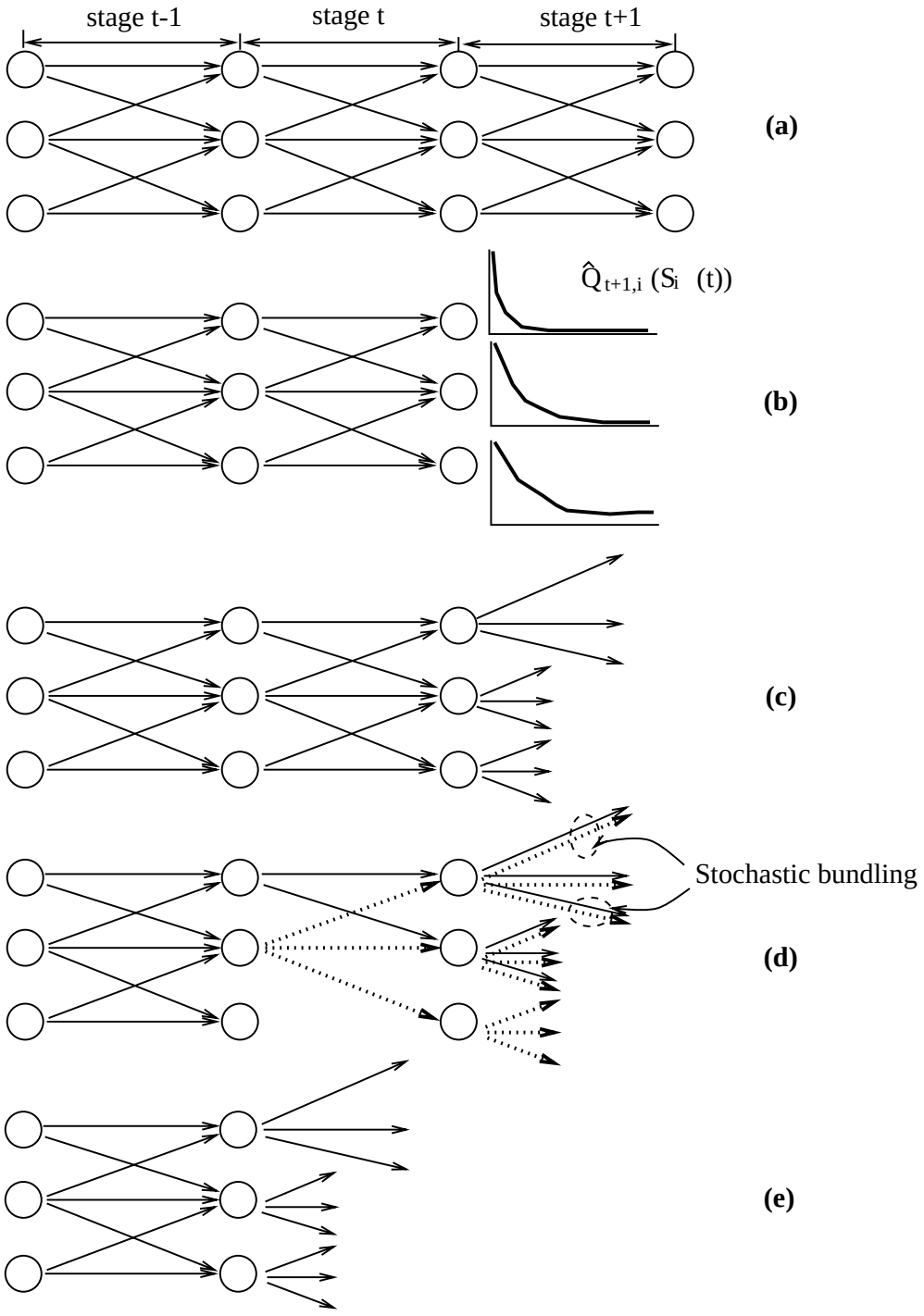

Figure 3: A step in SCAM. (a) Recourse problems in stage $t-1$, t and $t+1$. (b) Approximations for stage $t+1$ problem. (c) Augmented stage t problem. (d) Decomposition by origins, producing trees. (e) Augmented stage $t-1$ problem.

in [13] for each individual tree, producing a separable, convex approximation: $\hat{Q}_t(S^{t-1})$. Again, using the recourse links to represent these convex approximations, we obtain the augmented stage $t-1$ network shown in figure 3e.

Remarks:

1. Since $Q_{N+1}(\cdot,\cdot) \equiv 0$, the stage N problem consists of independent one-level trees. Therefore, the expected recourse function for the stage N is exact.
2. The algorithm for tree problems is not limited to one or two-level trees. Therefore, our method can also be applied to recourse problems which are transshipment networks within each stage. In these cases, the definition of commodity and the generation of trees may need to be modified (see [12]).
3. The basic idea of solving tree problems and a numerical example which illustrates a step in SCAM are given in the appendix.

3 Alternative methods

This section describes two alternative methods for solving dynamic fleet management problems, namely, the dynamic deterministic method and the successive linear approximation method. These two methods are used to compare with SCAM in section 4.

3.1 Dynamic deterministic method

The dynamic deterministic method (DYNDET) is the most widely used technique for solving fleet management problems. In this method, all the random arc capacities in the dynamic network are replaced with their means, producing the following deterministic problem:

$$\min_{x^t} \sum_{t=1}^{N} c^{tT} x^t \tag{39}$$

subject to:

$$\sum_{j \in \mathcal{N}^2} x_{ij}^1 = R_i^1 \quad \forall i \in \mathcal{N}^1, \tag{40}$$

$$\sum_{i \in \mathcal{N}^t} x_{ij}^t - \sum_{k \in \mathcal{N}^{t+2}} x_{jk}^{t+1} = 0 \quad \forall j \in \mathcal{N}^{t+1}, t = 2, ..., N-1 \tag{41}$$

$$x_{ij}^1 \leq \xi_{ij}^1 \quad \forall i \in \mathcal{N}^1, j \in \mathcal{N}^2 \tag{42}$$

$$x_{ij}^t \leq \bar{\xi}_{ij}^t \quad \forall i \in \mathcal{N}^t, j \in \mathcal{N}^{t+1} \tag{43}$$

where $\bar{\xi}_{ij}^t = E_t[\tilde{\xi}_{ij}^t]$.

Since the mean arc capacities are generally fractional, the solution of this problems would also be fractional. Therefore, in order to get an integer solution, we set

$$\hat{x}_{ij}^1 = \lfloor x_{ij}^1 + \gamma_i \rfloor$$

where $\lfloor x \rfloor$ is the largest integer less than or equal to x and $0 \leq \gamma_i \leq 1$ is a rounding factor dependent on i such that the flow conservation is maintained:

$$\sum_i \hat{x}_{ij}^1 = \sum_i x_{ij}^1.$$

3.2 Successive linear approximation procedure

Frantzeskakis and Powell[5] develops a heuristic, called the *Successive Linear Approximation Procedure* (SLAP), for solving multistage dynamic networks with random arc capacities. This stochastic method is an extension of the nodal recourse strategy introduced by Powell[11] (the nodal recourse model is a special case of the one-level tree recourse model). Similar to SCAM, SLAP is a backward recursion procedure. However, in each stage t, where $t = 3, 4, \ldots, N-1$, SLAP linearizes the convex approximations obtained in stage $t+1$, so as to calculate the recourse function in period t. To see the differences between SLAP and SCAM in a step of the recursion, we recast problem (13):

$$\tilde{Q}_t(S^{t-1}, \xi^t) = \min_{x^t, S^t} \left\{ c^{tT} x^t + \sum_{j \in \mathcal{N}^{t+1}} \hat{Q}_{t+1,j}(S_j^t) \text{ subject to: (3) - (6)} \right\}. \tag{44}$$

This problem is not separable in supply nodes $\mathcal{N}^t$ since each $\hat{Q}_{t+1,j}(S_j^t)$ is a convex function which depends on flows from different origins, that is:

$$\hat{Q}_{t+1,j}(S_j^t) = \hat{Q}_{t+1,j}\left(\sum_{i\in\mathcal{N}^{t+1}} x_{ij}^t\right)$$

In SCAM, we replace $\hat{Q}_{t+1,j}(S_j^t)$ by a set of links and then induce separability via relaxation. On the contrary, SLAP uses linearization to induce separability. Hence, the convex approximation $\hat{Q}_{t+1,j}(S_j^t)$ is further replaced with a linear function:

$$\begin{aligned}\hat{Q}_{t+1,j}(S_j^t) &= \theta_j^t \cdot S_j^t + b_j^t \\ &= \theta_j^t \cdot \sum_i x_{ij}^t + b_j^t\end{aligned}$$

resulting in the following problem:

$$\tilde{Q}_t'(S^{t-1},\xi^t) = \min_{x^t,S^t} \sum_i \sum_j \left(c_{ij}^t + \theta_j^t\right) x_{ij}^t \quad \text{subject to: (3) - (6).} \tag{45}$$

Problem (45) is separable in supply nodes $\mathcal{N}^t$. Hence, it can be decomposed into a number of entirely decoupled one-level tree subproblems. The expected recourse functions for these subproblems can be obtained easily. Finally, these functions are then used in stage $t-1$ recourse problem. Various schemes for choosing θ_j^t are discussed in [5].

In our experiment, we use two versions of SLAP, which differ in the choice of θ_j^t. In the first version, θ_j^t is the expected slope of the convex function $\hat{Q}_{t+1,j}(S_j^t)$ obtained by:

$$\theta_j^t = \sum_{k=1}^{\infty} P\left\{S_j^t = k\right\} q_{jk}^{t+1}.$$

where we have fit a distribution for the supplies S_j^t. In another version, we choose θ_j^t to be the slope of $\hat{Q}_{t+1,j}(m_j^{t+1})$ where m_j^{t+1} is the estimated mean supply to node j.

The SLAP methodology has been very successful for a class of dynamic fleet management problems (see Frantzeskakis[4]). However, this method does not allow transshipment nodes within a stage. On the other hand, one difficulty with SLAP is that it can be sensitive to the choice of θ^t. Also, the case of a linear approximation can lead to extreme solutions which give poor results.

4 Numerical experiments

In this section, we evaluate the SCAM methodology by comparing it to the methods described in section 3. Section 4.1 describes the experimental design and section 4.2 reports the numerical results.

4.1 Experimental design

We compared the three methodologies using a real problem and a set of randomly generated problems. We include the random problems for extensive comparisons of these methods under various conditions. To evaluate the solutions, we conducted the rolling horizon experiment since obtaining the "optimal" solution for large stochastic networks is numerically intractable. In the following, we first describe our test problems. Then, we outline the rolling horizon simulation and the performance measures.

Test problems

There are two classes of the test problems. The first class contains a real data set obtained from a major common carrier (the same data set used in [5]). We do not intend to describe this data set in detail, but provide a sense of the problem size. In this data set, the country is partitioned into 60 regions. On average, loaded vehicles from a region can be moved to ten other regions (that is, the network is not completely dense). Therefore, there are about 600 random variables per day representing market demands between regions. In this dataset, the average daily demand is 312 for weekdays and 32 for weekends. Consequently, the mean demand between a pair of regions is roughly 0.5. These random demands are assumed to follow truncated Poisson distributions where a Poisson variate $\tilde{\xi}$ is truncated at k if k is the first integer such that $P\{\tilde{\xi} > k\} < 0.0001$. Thus, each random arc capacities can take an average of 5 possible values. Over a 14-day planning horizon, we are dealing with several thousand independent random arc capacities. As a result, the total number of possible realizations exceeds 10^{1000} which is

effectively infinite.

The second class contains randomly generated problems, with which we can make comparisons under various conditions. The parameters for the random problem generator include the number of regions, the distances between regions, the net profit of each loaded movement, the cost of each empty movement, the cost for holding a vehicle at its current location, the initial vehicle allocation and the forecast demands.

The random problem generator creates R points which are uniformly located in a 1000 by 2000 mile rectangle. These points represent the centers of the R regions. We simply take the Euclidean distance between the centers of two regions as the corresponding travel distance. We assume that vehicles can travel 600 miles per period (one day). The net profit for each loaded movement and the cost for each empty movement are 40 cents/mile and 70 cents/mile respectively. There is no cost for holding a vehicle at its current location over time. The initial vehicle allocation is made proportional to the average outbound flow of the regions. The fleet size (denoted by F) was chosen after performing a set of calibrating runs. The market demand ξ_{ij}^t between region i and region j at time t is assumed to follow a Poisson distribution with mean m_{ij}^t which is given by:

$$m_{ij}^t = \alpha_i \cdot \beta_j \cdot \gamma_t \cdot \nu$$

where

α_i = inbound potential for region i,
β_j = outbound potential for region j,
γ_t = time variation factor,
ν = an exponential random variable.

These four factors are determined as follows.

The inbound potential and outbound potential for each region capture the region's ability to attract the inbound flows or generate the outbound flows. The inbound potential for a region i, α_i, is drawn uniformly between 0.2 and 1.8. The corresponding outbound potential β_i is obtained by $\beta_i = 2 - \alpha_i$. Therefore, these two potentials are

negatively correlated. The motivation for this setting is partly because regions with large inbound flows often have small outbound flows in real world applications. More importantly, under this setting, a myopic algorithm may produce a poor solution since a vehicle may be sent to a region with high inbound potential but with very low outbound demand. The time parameter γ_t is used to represent weekly pattern for the demands in daily operation: high demand in weekdays and low demand in weekends. Finally, to capture the randomness of demands, we also include an exponentially distributed random number ν with mean 0.5 which is the typical average daily demand between regions for the first data set.

Finally, we use a parameter d where $d \in [0, 1]$ to control the link density in the network. For a lower value of d, the future impacts of current decision are accentuated, meaning that "bad" current decisions may induce more serious consequences. With a sparse network, corrective actions are hard to make.

The rolling horizon simulation

For multistage problems with the size of practical interest, obtaining optimal solutions is generally intractable. Therefore, it is often impossible to compare the solution produced by a method with the optimal solution. One possible way is to develop some bounds on the optimal solution and to use these bounds for evaluating the quality of a solution. Again, obtaining tight bounds for large multistage problems can be extremely difficult. A much more simpler alternative is using the rolling horizon simulation.

The basic idea of the rolling horizon simulation is to generate a stream of realizations of the random demands over a period of time. We refer to this period as the *simulation horizon* or the *rolling horizon*. The effectiveness of a solution method is then evaluated over this simulation horizon during which the performance measures are accumulated.

Let

T = the length of the simulation horizon,

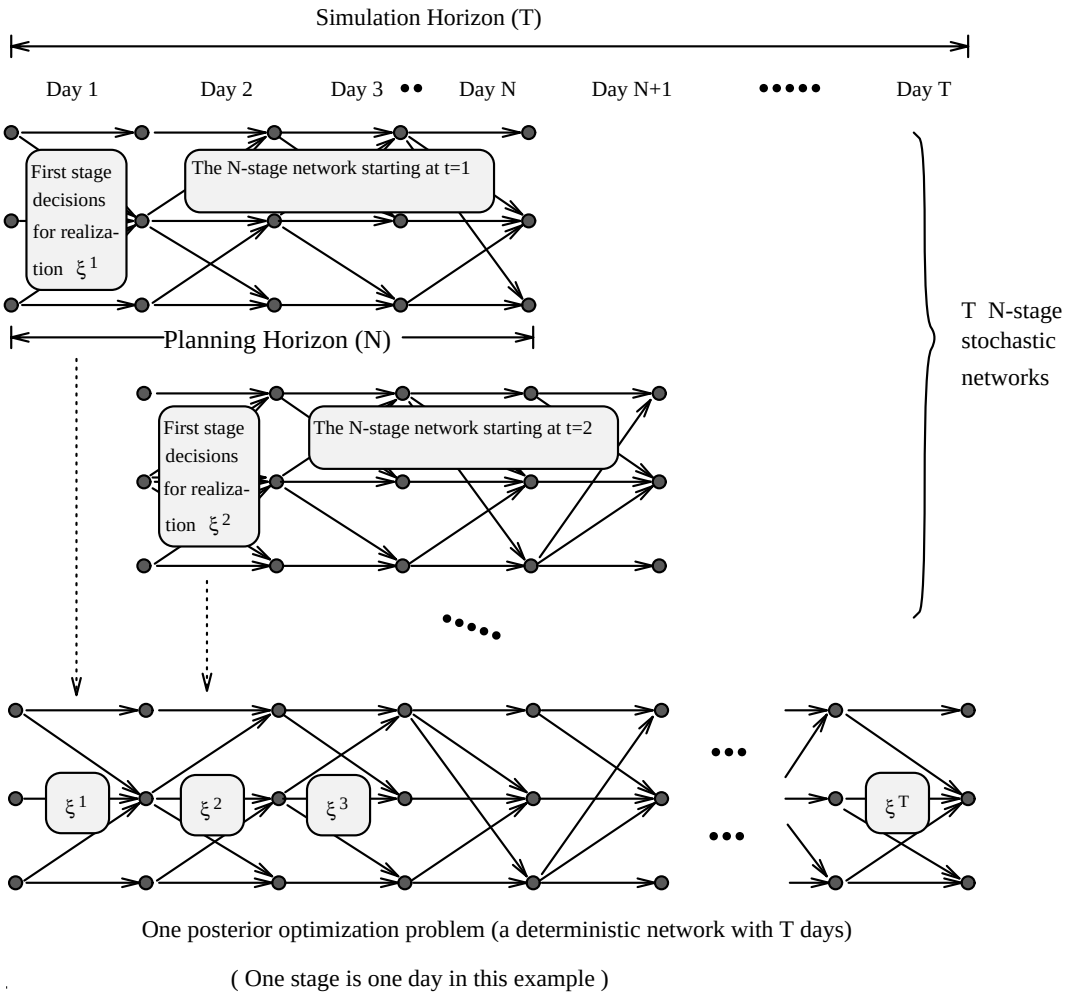

Figure 4: The rolling horizon simulation.

N = the length of the planning horizon, $N \leq T$,

$\mathcal{M}$ = a solution method,

$\mathcal{C}^t(\mathcal{M}, \xi^t)$ = total cost in stage t for using method $\mathcal{M}$ under realization ξ^t,

$\mathcal{C}(\mathcal{M}, \xi^1, \ldots, \xi^T)$ = total cost for using method $\mathcal{M}$ over the simulation horizon with realizations $\xi^1, \ldots, \xi^T$.

Figure 4 shows how the rolling horizon simulation works. First, we start at time $t = 1$. We obtain a realization, denoted by ξ^1, of the random demands at time $t = 1$. Next, we solve an N-stage dynamic network (spanning from $t = 1$ to $t = N$) by a solution

method $\mathcal{M}$. The recommended first stage decisions are then actually implemented. The corresponding cost for $t = 1$ is thus $\mathcal{C}^1(\mathcal{M}, \xi^1)$. Next, we advance the clock to $t = 2$ and obtain a realization ξ^2 of the random demands at time $t = 2$. Again, we apply method $\mathcal{M}$ to solve the N-stage network which spans $t = 2$ to $t = N+1$. Similarly, we implement the recommended first stage decisions and obtain the total stage 2 cost $\mathcal{C}^2(\mathcal{M}, \xi^2)$. We continue to advance the clock from $t = 3$ to $t = T$ and solve the corresponding N-stage networks. As a result, we are actually solving T N-stage networks and accumulating the total cost over the entire simulation horizon. In other words, we obtain the performance measure of method $\mathcal{M}$ by:

$$\mathcal{C}(\mathcal{M}, \xi^1, \xi^2, \ldots, \xi^T) = \sum_{t=1}^{T} \mathcal{C}^t(\mathcal{M}, \xi^t).$$

Through rolling horizon simulations, we can compare the relative effectiveness of different solution methods. However, little is known for the absolute effectiveness of the methods, that is, how good or how bad the solution of a method is when it is compared with the "optimal solution"? To answer this question, consider a device called the *posterior bound*. When we solve an N-stage network which starts at time t, we must make the first stage decisions without knowing the actual market demands $\tilde{\xi}^\tau$ where $\tau = t + 1, \ldots, t + N - 1$. Assume that after we know the whole stream of the demands, $\xi^1, \xi^2, \ldots, \xi^T$, instead of solving T N-stage stochastic networks, we solve a single deterministic network over the entire simulation horizon. Let

$$\mathcal{C}^p(\xi^1, \xi^2, \ldots, \xi^T) = \text{total cost of the } T\text{-period network with realizations } \xi^1, \xi^2, \ldots, \xi^T.$$

Notice that such a posterior optimization involves no uncertainty since decisions are allowed to anticipate future demands (see the bottom of figure 4). Therefore, the cost $\mathcal{C}^p(\xi^1, \xi^2, \ldots, \xi^T)$ is the lowest overall cost, that is,

$$\mathcal{C}^p(\xi^1, \xi^2, \ldots, \xi^T) \leq \mathcal{C}(\mathcal{M}, \xi^1, \xi^2, \ldots, \xi^T) \tag{46}$$

for the cost produced by any method $\mathcal{M}$. Hence, we refer to $\mathcal{C}^p(\xi^1, \xi^2, \ldots, \xi^T)$ as the *posterior bound*(PB) which is normally unreachable.

Problems	Number of regions (R)	Fleet size (F)	Length of planning horizon (N)	Link density (d)
P1	60	500	7	-
P2	60	500	14	-
P3	60	750	7	-
P4	60	750	14	-
P5	20	100	7	0.4
P6	20	100	14	0.4
P7	20	225	7	0.9
P8	20	225	14	0.9
P9	40	300	7	0.3
P10	40	300	14	0.3
P11	40	750	7	0.8
P12	40	750	14	0.8

Table 1: Characteristics of test problems

Let $\mathcal{M}^*$ be a method that can produce "optimal solution". Then, the corresponding performance measure is $\mathcal{C}(\mathcal{M}^*, \xi^1, \xi^2, \ldots, \xi^T)$. From (46), we know that

$$\mathcal{C}^p(\xi^1, \xi^2, \ldots, \xi^T) \leq \mathcal{C}(\mathcal{M}^*, \xi^1, \xi^2, \ldots, \xi^T) \tag{47}$$

Since $\mathcal{C}(\mathcal{M}^*, \xi^1, \xi^2, \ldots, \xi^T)$ is generally never known, one way to measure the absolute effectiveness of a method $\mathcal{M}$ is to compare the difference between $\mathcal{C}^p(\xi^1, \xi^2, \ldots, \xi^T)$ and $\mathcal{C}(\mathcal{M}, \xi^1, \xi^2, \ldots, \xi^T)$. The smaller is the difference, the better the method $\mathcal{M}$ is.

4.2 Numerical results

In this section, we report the experimental results. We have 12 test problems with characteristics summarized in table 1. The first four problems use the real data set and

the remaining are randomly generated. For the real problems, we varied the fleet size and the length of the planning horizon. For the random problems, we further varied the problem size and the link density. For each test problem, the actual demands over the T simulation periods are the same when applying different methodologies in each simulation run. Also, the initial vehicle allocation is held constant from one method to the next. Therefore, all methods are compared under a homogeneous environment. All problems were tested on a 14-day simulation period.

Table 2 shows the total costs and the percentage gaps for the 12 test problems for the three solution methods. For the two versions of SLAP, we choose the best one for comparison purposes. Clearly from table 2, as far as the total cost is concerned, both the stochastic methods (SCAM and SLAP) are significantly better than the deterministic method (DYNDET). Furthermore, SCAM produces better results than SLAP in all instances. In particular, SLAP cuts the average percentage gap to PB from 15.8 to 11.6 percent, which amounts to a 26.6 percent reduction over the gap for DYNDET. Meanwhile, SCAM further shortens the average percentage gap to 8.8 percent, which amounts to an additional 18.2 percent reduction over SLAP.

Finally, we are also interested in the performance of the stochastic methods with respect to the length of planning horizon. The experiments show that stochastic methods with longer planning horizon may not necessarily produce better results. One explanation is that the benefits obtaining from using a longer planning horizon cannot offset the errors resulting from a larger number of approximations. However, SCAM seems more stable with respect to changes in the planning horizon. For example, when we extend the planning horizon for P5 from 7 to 14 (resulting in P6), the percentage gap of SLAP increases from 17.8 to 24.2 percent, while that of SCAM increases from 15.2 to 16.2 percent.

These experiments show encouraging results: (1) stochastic models seems better than the deterministic model in the context of dynamic fleet management; (2) SCAM performs better than SLAP as far as the total cost and the length of planning horizon are concerned.

	Total Cost ('000)				Percentage Gap to PB		
Problems	PB	DYNDET	SLAP	SCAM	DYNDET	SLAP	SCAM
P1	-1,114.4	-1,012.1	-1,056.5	-1,058.2	10.1	5.5	5.3
P2	-1,114.4	-1,011.9	-1,037.8	-1,064.9	10.1	7.4	4.7
P3	-1,352.7	-1,232.7	-1,295.3	-1,310.0	9.7	4.4	3.2
P4	-1,352.7	-1,225.1	-1,293.3	-1,305.0	10.4	4.6	3.6
P5	-74.3	-58.7	-63.1	-64.5	26.6	17.8	15.2
P6	-74.3	-58.2	-59.8	-63.9	27.6	24.2	16.2
P7	-208.5	-181.7	-188.4	-191.7	14.7	10.6	8.7
P8	-208.5	-178.6	-179.1	-188.1	16.7	16.4	10.8
P9	-249.8	-207.9	-221.7	-224.1	20.2	12.3	11.5
P10	-249.8	-208.9	-215.7	-223.6	19.6	15.8	11.7
P11	-794.9	-712.1	-726.2	-740.9	11.6	9.5	7.3
P12	-794.9	-710.5	-719.6	-736.6	11.9	10.5	7.9
Average Gap					15.8	11.6	8.8

Table 2: Comparison of solution methods on test problems

5 Conclusion

We formulate the dynamic fleet management problem using a stochastic programming framework. Then, we develop the SCAM methodology for the resulting stochastic networks. In this method, we recursively approximate the expected total cost in each stage as a function of the supply to this stage. The resulting approximations are convex, separable functions which are obtained by structurally decomposing the underlying network into trees, whose expected recourse functions are obtained easily. We use a stochastic analog of the Lagrangian relaxation to iteratively improve our approximations. Finally, we apply SCAM to some dynamic fleet management problems. Numerical experiments show that the use of the SCAM methodology is very encouraging. However, the question of how well SCAM can perform in other contexts has yet to be addressed.

Acknowledgement

We would like to thank the referees for their comments and suggestions which helped to strengthen the presentation of this paper.

References

[1] J. Aronson. A survey of dynamic network flows. In P. Hammer, editor, *Annals of Operations Research*, pages 1–66. J.C. Baltzer AG, 1989.

[2] J. R. Birge and R. J-B Wets. Designing approximation schemes for stochastic optimization problems, in particular for stochastic programs with recourse. *Mathematical Programming Study*, 27:54–102, 1986.

[3] G. Dantzig and D. Fulkerson. Minimizing the number of tankers to meet a fixed schedule. *Naval Research Logistics Quarterly*, 1:217–222, 1954.

[4] L. Frantzeskakis. Dynamic networks with random arc capacities, with application to the stochastic dynamic vehicle allocation problem. Ph.D. dissertation, Department of Civil Engineering, Princeton University, 1990.

[5] L. F. Frantzeskakis and W. B. Powell. A successive linear approximation procedure for stochastic, dynamic vehicle allocation problems. *Transportation Science*, 24(1), Feb 1990.

[6] W. Jordan and M. Turnquist. A stochastic dynamic network model for railroad car distribution. *Transportation Science*, 17:123–145, 1983.

[7] P. Kall, A. Ruszczynski, and K. Frauendorfer. Approximation in stochastic programming. In *Numerical Techniques for Stochastic Optimization*, pages 313–351. Springer-Verlag, 1988.

[8] C. Lemaréchal. Nondifferentiable optimization. In *Handbooks in OR & MS : Optimization*, volume 1, chapter 7. Elsevier Science Publishers B.V. (North Holland), 1989.

[9] T. Magnanti and R. Simpson. Transportation network analysis and decomposition methods. Report no. dot-tsc-rspd-78-6, U.S. Department of Transportation, Washington D.C., 1978.

[10] W. B. Powell. A stochastic model of the dynamic vehicle allocation problem. *Transportation Science*, 20:117–129, 1986.

[11] 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.

[12] W. B. Powell and R. K.-M. Cheung. Network recourse decomposition method for dynamic networks with random arc capacities. *Networks*, 24:369–384, 1994.

[13] W. B. Powell and R. K.-M. Cheung. Stochastic programs over trees with random arc capacities. *Networks*, 24:161–175, 1994.

[14] R. M. Van Slyke and R. J-B Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal of Applied Mathematics*, 17(4):638–663, 1969.

[15] S. W. Wallace. Solving stochastic programs with network recourse. *Networks*, 16:295–317, 1986.

[16] W. White. Dynamic transshipment networks: An algorithm and its application to the distribution of empty containers. *Networks*, 2(3):211–236, 1972.

[17] W. White and A. Bomberault. A network algorithm for empty freight car allocation. *IBM Systems Journal*, 8(2):147–171, 1969.

Appendix

This appendix provides a numerical example for showing a step of SCAM. We begin by reviewing some key results in [13] which are used in the example.

Tree subproblem

Consider a directed tree with random arc capacities, denoted by $\mathcal{T}$, where the root node is the only supply node. Let $\tilde{\xi}$ be the vector of random arc capacities defined over a probability space $(\Omega, \mathcal{F}, P)$ with elementary outcomes $\Omega = \{\omega\}$. Assume all paths are ranked according to their cost (from least to most). Let

$$
\begin{aligned}
N_p &= \text{the number of paths in } \mathcal{T} \\
\delta_{ij}^n &= \begin{cases} 1 & \text{if link } (i,j) \text{ is on path } n \\ 0 & \text{otherwise} \end{cases} \\
\tilde{c}_n &= \text{the cost of path } n \\
\phi(k,n) &= P\{\text{the } k^{th} \text{ unit of supply entering the tree takes the } n^{th} \text{ path}\} \\
q(k) &= \text{the marginal expected contribution for the } k^{th} \text{ unit of supply} \\
Z_n &= \text{the total capacity of the first } n \text{ ranked paths} \\
Q(s,\xi) &= \text{the total tree cost with scalar supply } s \text{ for realization } \xi.
\end{aligned}
$$

Powell and Cheung[13] show that the expected value of $Q(s,\xi)$ can be found parametrically as a function of the scalar supply s using the following equations:

$$\phi(k,n) = P\{Z_n \geq k\} - P\{Z_{n-1} \geq k\}. \tag{48}$$

$$EQ(s,\tilde{\xi}) = \sum_{k=1}^{s} q(k) = \sum_{k=1}^{s} \sum_{n=1}^{N_p} \tilde{c}_n \cdot \phi(k,n). \tag{49}$$

and the expected link flow can be obtained by:

$$\bar{x}_{ij} = Ex_{ij} = \sum_{k=1}^{s} \sum_{n=1}^{N_p} \delta_{ij}^n \ \phi(k,n) \quad \forall\ (i,j) \in \mathcal{A} \tag{50}$$

Note that equation (48) says that the k^{th} unit of flow is on path n if and only if the first $n-1$ ranked paths have total capacity of less than k and the first $n-1$ ranked paths have total capacity of at least k.

Example

Now consider the network in figure 5. Some arc capacities are assumed to have truncated Poisson variates with mean indicated in figure 5 (we truncate a variable $\tilde{\xi}$ at k when k is the first integer such that $P\{\tilde{\xi} > k\} < 0.0001$).

Initial condition
$S^T = [S_1 \ \ S_2 \ \ S_3]^T = [2 \ \ 3 \ \ 2]^T$, and $\lambda^0 = 0$.

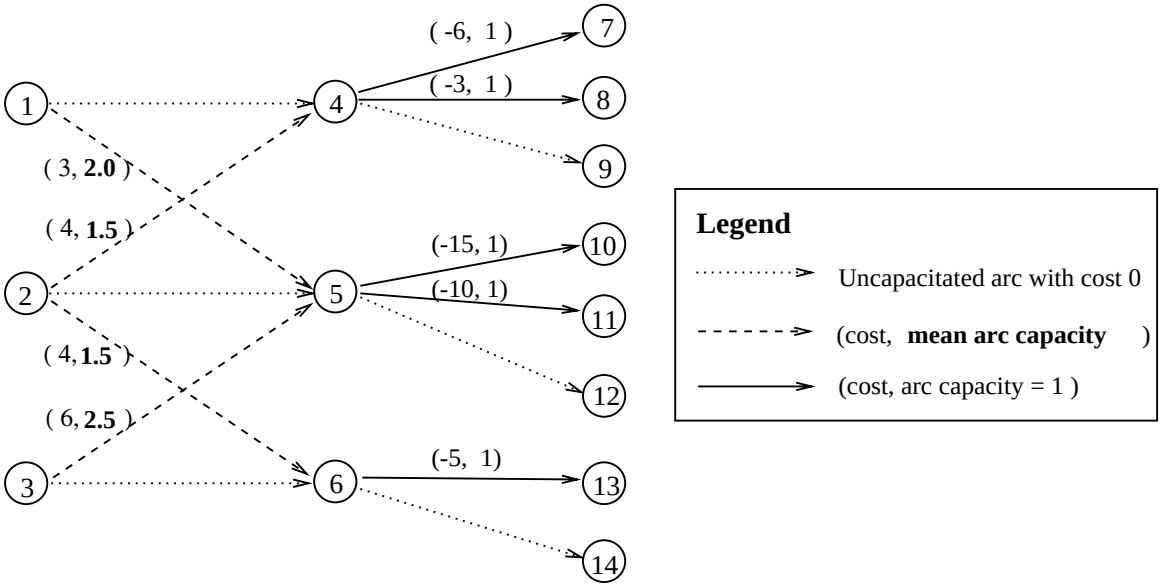

Figure 5: An example of network recourse problem.

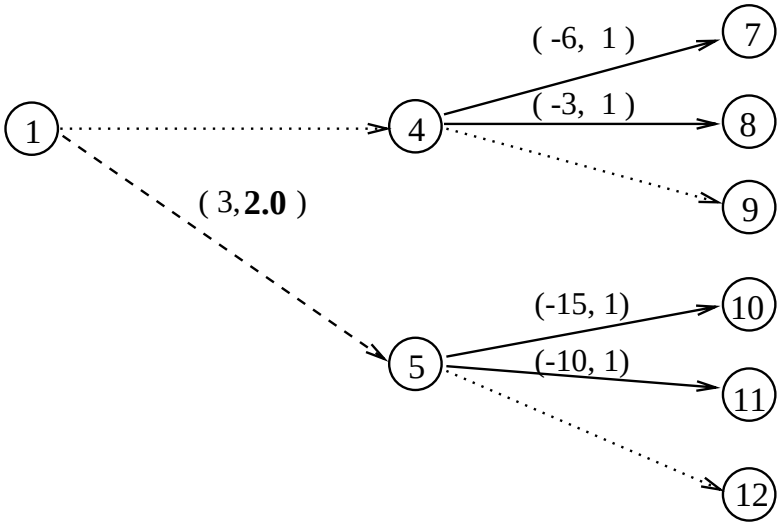

Figure 6: The two-level tree for node 1.

Find the expected recourse function for the first tree:
The tree rooted at node 1 is depicted in figure 6. We denote this two-level tree by $\mathcal{T}_1$.

1. *Rank the paths:*
   The ranked paths are $p_1 = (1 \rightarrow 5 \rightarrow 10)$ with $\bar{c}_1 = -12$, $p_2 = (1 \rightarrow 5 \rightarrow 11)$ with $\bar{c}_2 = -7$, $p_3 = (1 \rightarrow 4 \rightarrow 7)$ with $\bar{c}_3 = -6$, $p_4 = (1 \rightarrow 4 \rightarrow 8)$ with $\bar{c}_4 = -3$, $p_5 = (1 \rightarrow 4 \rightarrow 9)$ with $\bar{c}_5 = 0$.

2. *Compute $\phi(k,n)$:*
   The Z_n can be found by: $Z_1 = \min\{\xi_{14}, 1\}$, $Z_2 = \min\{\xi_{14}, 2\}$, $Z_3 = \xi_{14} + 2, \ldots$, etc. Thus, using (48) we obtain:

$\phi(k,n)$	n=1	2	3	4	5
$k = 1$	0.86	0.00	0.14	0.00	0.00
2	0.00	0.59	0.27	0.14	0.00
3	0.00	0.00	0.59	0.27	0.14
4	0.00	0.00	0.00	0.59	0.41
≥ 5	0.00	0.00	0.00	0.00	1.00

3. *Obtain $\hat{Q}_{t,1}(k, \lambda^0)$ using (49):*

$$\hat{Q}_{t,1}(k, \lambda^0) = \begin{cases} -11.19 & k = 1 \\ -17.38 & k = 2 \\ -21.75 & k = 3 \\ -23.53 & k = 4 \\ -25.14 & k \geq 5 \end{cases}$$

4. *Compute the expected link flows using equation (50)*
   For instance, the expected flow on $(4,7)$ in $\mathcal{T}_1$ is:

$$\bar{x}_{4,7} = \sum_{k=1}^{2} \sum_{n} \delta_{ij}^{n} \phi(k,n) = \phi(1,3) + \phi(2,3) = 0.41 \tag{51}$$

We proceed in the same manner for trees $\mathcal{T}_2$ and $\mathcal{T}_3$, producing the results in table 3.

In this table, we include all links that are capacitated. However, only the last five links induce bundle constraints. Columns (b), (c) and (d) shows the expected link flow on trees with roots 1, 2 and 3 respectively, whereas column (e) shows their sum.

(a) arcs	(b) $\mathcal{T}_1$	(c) $\mathcal{T}_2$	(d) $\mathcal{T}_3$	(e) Total expected flow	(f) Capacity	(g) g^1	(h) λ^1
(1,5)	1.46	-	-	1.46	2.0	-	-
(2,4)	-	0.78	-	0.78	1.5	-	-
(2,6)	-	0.17	-	0.17	1.5	-	-
(3,4)	-	-	-	0.00	2.5	-	-
(4,7)	0.41	0.78	-	1.19	1	0.19	0.5
(4,8)	0.14	-	-	0.14	1	-0.86	0.0
(5,10)	0.86	1.00	-	1.86	1	0.86	2.4
(5,11)	0.59	1.00	-	1.59	1	0.59	1.7
(6,13)	-	0.17	1.00	1.17	1	0.17	0.5

Table 3: Results after the first iteration of SCAM

Column (g) shows the subgradient which simply amounts to column (f) minus column (e). Finally, using equation (31), the lower bound is:

$$\hat{Q}(S, \lambda^0) = \sum_{i \in \mathcal{N}^t} \hat{Q}_i(S_i) - {\lambda^0}^T 1 = -54.1.$$

Assume that we get $Q^U(S) = -39.0$ through Monte Carlo simulation. When choosing $\gamma = 2.0$, we compute λ^1 using (37) and the updated λ are shown in column (h) of table 3. This completes the first iteration of SCAM.