

A Stochastic Integer Programming Approach to Solving a Synchronous Optical Network Ring Design Problem

J. Cole Smith

Department of Systems and Industrial Engineering, P.O. Box 210020, University of Arizona, Tucson, Arizona 85721-0020

Andrew J. Schaefer

Department of Industrial Engineering, University of Pittsburgh, Pittsburgh, Pennsylvania 15261

Joyce W. Yen

University of Washington, Seattle, Washington 98195-2180

We develop stochastic integer programming techniques tailored toward solving a Synchronous Optical Network (SONET) ring design problem with uncertain demands. Our approach is based on an L-shaped algorithm, whose (integer) master program prescribes a candidate network design, and whose (continuous) subproblems relay information regarding potential shortage penalty costs to the ring design decisions. This naive implementation performs very poorly due to two major problems: (1) the weakness of the master problem relaxations, and (2) the limited information passed to the master problem by the optimality cuts. Accordingly, we enforce certain necessary conditions regarding shortage penalty contributions to the objective function within the master problem, along with a corresponding set of valid inequalities that improves the solvability of the master problem. We also show how a nonlinear reformulation of the model can be used to capture an exponential number of optimality cuts generated by the linear model. We augment these techniques with a powerful upper-bounding heuristic to further accelerate the convergence of the algorithm, and demonstrate the effectiveness of our methodologies on a test bed of randomly generated stochastic SONET instances. © 2004 Wiley Periodicals, Inc. *NETWORKS*, Vol. 44(1), 12–26 2004

Keywords: stochastic integer programming; L-shaped method; decomposition; SONET; network design

1. INTRODUCTION

The last decade has witnessed a proliferation in the research and development of optical telecommunication fiber. Modern optical fibers typically permit the simultaneous transmission of 2.4 Gbps (gigabits per second) of data over each strand of fiber. With the advent of this technology, the failure of a single optical fiber may result in a substantial loss of customer service. To alleviate this problem, self-healing rings (SHR) are often utilized to connect client nodes by a ring of fibers. These rings automatically reroute telecommunication traffic in the case of equipment failure, providing essential survivability to high-bandwidth networks.

The Synchronous Optical Network (SONET) is a standard of transmission technology used in optical fiber networks designed under SHR topologies. Within a SONET ring, voice and data streams are *multiplexed* onto a fiber-optic strand. The equipment responsible for identifying traffic that should be added and dropped from the flow of ring traffic is called an *add-drop multiplexer* (ADM). (Wu [34] and Wu and Burrowes [35] provide a more in-depth description of the SONET ring architecture.) Our study is motivated by the relatively low cost of ADMs compared with digital crossconnect systems (DCS), which transmit data between two nodes not placed on the same ring by acting as a hub, or central connection, among the rings. As a result, we forbid interring traffic, and focus instead on designing minimum-cost networks in which all demands are satisfied via intraring routing.

Two deterministic variants of the intraring problem considered in this article have been investigated. When demands between nodes must be fully placed on one ring and not split between two or more rings (nonsplit demand), Lee

Received November 2002; accepted January 2004

Correspondence to: J. C. Smith; e-mail: cole@sie.arizona.edu

Contract grant sponsor: Defense Advanced Research Projects Agency (to J.C.S.); contract grant number: N66001-01-1-8925

Contract grant sponsor: National Science Foundation (to A.S.); contract grant number: DMI-0217190

DOI 10.1002/net.20009

Published online in Wiley InterScience (www.interscience.wiley.com).

© 2004 Wiley Periodicals, Inc.

et al. [16] present a branch-and-cut algorithm, and Sutter et al. [29] utilize column generation to minimize the required number of ADMs that must be installed. When demands between nodes may be placed across several rings (split demand), Sherali et al. [26] provide a preprocessing/cutting-plane technique for obtaining optimal solutions within reasonable computational limits. Other related studies have been undertaken by Goldschmidt et al. [9], who provide an approximation algorithm for the problem, and by Belvaux et al. [1], who investigate a dynamic version of a similar design problem. Laguna [13] develops a mixed-integer programming model to solve a SONET design problem that permits interring traffic (e.g., by DCS), subject to various ring capacity restrictions. Goldschmidt et al. [10], who also consider interring traffic, minimize the number of rings necessary to satisfy a given set of demands.

In contrast to the foregoing studies, we observe that forecast telecommunication demands among client nodes are actually quite stochastic in practice. Moreover, it is often preferable for a network provider to partially satisfy certain demands at the cost of a given shortage penalty, rather than expanding the telecommunication infrastructure and incurring additional capital equipment expenditures to satisfy some marginal extra demand. We accommodate both the stochastic demand and shortage penalty features within our design model by incorporating new modeling and algorithmic strategies within contemporary stochastic integer programming methods.

The importance of considering stochastic elements within network design models has been a theme of recent optimization studies. In the telecommunications field, Dye et al. [7] provide an approximation algorithm for a stochastic network design problem. Related problems arise in stochastic transportation network design problems, which have recently been studied by numerous researchers. For example, Waller and Ziliaskopoulos [31] examine a freeway corridor network design problem with random demands and model the problem with chance constraints and a two-stage linear program. Rosenberger et al. [20] consider the robust fleet assignment problem, which may be viewed as the design of an airline's timeline network under uncertainty. They demonstrate that the solution to the stochastic model performs better in a random environment than a fleet assignment obtained by a purely deterministic approach. Finally, Soroush and Mirchandani [28] describe efforts in solving stochastic multicommodity flow problems, while Malashenko and Novikova [18] analyze the feasibility of certain network designs under uncertain demand and capacity values.

The remainder of this article is organized as follows. In Section 2, we provide a mixed-integer programming formulation for the deterministic variant of our problem. In Section 3, we introduce and formulate the stochastic version of this problem, develop a decomposition approach for its solution, and provide various modeling and algorithmic procedures to improve the efficiency of our algorithm. In Section 4, we develop a heuristic for quickly obtaining tight

upper bounds to the problem, and show how the heuristic may be integrated as an upper bounding procedure within the exact solution algorithm. In Section 5, we demonstrate the efficacy of the proposed approach on a set of realistic test instances. Finally, we conclude our discussion in Section 6 by summarizing our research and providing directions for future studies.

2. DETERMINISTIC PROBLEM STATEMENT

Consider a set N of client nodes, indexed by $i \in N \equiv \{1, \dots, |N|\}$. Let d_{ij} be the number of channels required to carry the traffic between node $i \in N$ and node $j (> i) \in N$. Accordingly, define an (undirected) edge set $A = \{(i, j) : i < j, d_{ij} > 0\}$ comprised of such demand pairs. A demand shortage penalty ϕ_{ij} is assessed for every unsatisfied unit of demand between nodes i and j , $\forall (i, j) \in A$. The values of these penalties are specified as a percentage of the cost of an ADM.

A set M of Unidirectional Path Switched Rings (UPSR) exists on which this demand may be routed. Note that to route traffic between two nodes assigned to a ring, we must have an ADM installed at both nodes. Each UPSR $k \in M$ is capacitated by both the number of ADMs $R_k \geq 2$ and the total amount of demand b_k it can support. Although previous SONET ring design studies assume a common value $R = R_k$ and $b = b_k \forall k \in M$, we allow differences in these capacities to address possible emerging technologies. One assumption of this study is that demands can be split among multiple rings, which is not always possible in practice depending on the nature of the rings. It is also worth noting that research has been conducted on the use of optimizing SONET networks with Bidirectional Line Switched Rings (BLSR). However, the optimization of BLSR capacity planning problems is itself an NP-hard problem under certain assumptions (see the seminal work by Cosares and Saniee [5], along with a host of optimization studies by Dell'Amico et al. [6], Karunanithi and Carpenter [12], and Schrijver et al. [21] for more details). Hence, we concentrate on UPSR-based systems, and leave an analogous study utilizing BLSRs for future research.

The (deterministic) optimization problem seeks to construct a set of SONET rings that can satisfy all demands subject to the foregoing restrictions. We choose to minimize the total number of ADMs that must be installed in the network, as this specialized equipment is the driving cost in the system. Note that we do not address node arrangement within a particular ring, but rather focus on the assignment of a set of nodes to each ring. The problem of arranging nodes on a ring has been studied by Wasem et al. [32], and by Lee et al. [15], for example.

To model the deterministic version of our SONET ring design optimization problem, we define a set of binary decision variables

$$x_{ik} = \begin{cases} 1 & \text{if node } i \text{ is assigned to ring } k \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in N, k \in M.$$

We also define a set of continuous decision variables $f_{\rho k}$ $\forall \rho = (i, j) \in A$ and $\forall k \in M$, representing the fraction of demand between the node pair (i, j) that will be satisfied by ring k , and w_{ij} $\forall (i, j) \in A$, representing the fraction of demand between nodes i and j that is not satisfied by the network. For notation convenience, we define the set of arcs incident to node i as

$$S_i = \{\rho \in A : \rho = (i, j) \text{ or } \rho = (j, i) \text{ for some } j\}.$$

The deterministic ring design problem (DRD) can now be stated as follows.

$$\text{DRD: Minimize } \sum_{i \in N} \sum_{k \in M} x_{ik} + \sum_{\rho \in A} \phi_{\rho} d_{\rho} w_{\rho} \quad (1a)$$

subject to

$$\sum_{k \in M} f_{\rho k} + w_{\rho} = 1 \quad \forall \rho \in A \quad (1b)$$

$$\sum_{\rho \in A} d_{\rho} f_{\rho k} \leq b_k \quad \forall k \in M \quad (1c)$$

$$\sum_{i \in N} x_{ik} \leq R_k \quad \forall k \in M \quad (1d)$$

$$0 \leq f_{\rho k} \leq x_{ik} \quad \forall i \in N, k \in M, \rho \in S_i \quad (1e)$$

$$x_{ik} \in \{0, 1\} \quad \forall i \in N, k \in M \quad (1f)$$

$$w_{\rho} \geq 0 \quad \forall \rho \in A. \quad (1g)$$

The objective function (1a) minimizes the sum of the total number of node-to-ring assignments (ADM installations), plus the total demand shortages weighted by their corresponding penalties. Constraints (1b) define the fraction of demand between each node pair that is not satisfied in the network, and (1c,d) impose the demand and ADM capacity restrictions on each ring. Finally, (1e) states that demand between nodes i and j , $(i, j) \in A$, may be satisfied on ring $k \in M$ only if both nodes i and j are placed on ring k , while (1f,g) represent logical variable restrictions.

3. SONET RING DESIGN WITH STOCHASTIC DEMANDS

In practice, customer demand may depend on several unknown factors, such as the state of competing technology or economic conditions, and is not known with certainty when the network is designed. In such cases, a network design that considers this uncertainty may perform better than one that does not. Laguna [14] considers the expansion of a single location under uncertainty while Gutierrez et al. [11] consider the problem of designing an entire network

given uncertainty in the input data. Both articles take a robust optimization approach. (See Mulvey et al. [19] for an overview of robust optimization.) We present a stochastic programming model designed to accommodate uncertain demands in designing the network, and then explore several techniques for improving the computational efficiency of the algorithm.

3.1. An L-Shaped Approach

We will formulate this problem as a two-stage stochastic program. (See [4] for an overview of stochastic programming.) Such programs are characterized by an initial *first-stage* decision, after which the true values of the random events are realized, upon which a *second-stage* or *recourse* decision is made. The network design problem serves as the first-stage problem, after which the second-stage problem determines the optimal allocation of unmet demands. The first-stage decision is an integer program, while the recourse decision can be solved as a network flow problem.

Let ξ be a discretely distributed random vector with finite support Ξ . In this problem, a scenario ξ represents a realization of the demand requests across the arcs in the network. Index the scenarios by $\kappa = 1, \dots, |\Xi|$. For all $\kappa = 1, \dots, |\Xi|$, define the parameters p^{κ} as the probability of realizing the κ th scenario, and d_{ρ}^{κ} as the number of channels requested for demand pair ρ in scenario κ , $\forall \rho \in A$. The decision variables w_{ρ}^{κ} are now defined to be the fraction of demand pair ρ left unsatisfied in scenario ξ_{κ} , $\forall \rho \in A$ and $\forall \kappa = 1, \dots, |\Xi|$.

The *extensive form* of the resulting two-stage stochastic program can be formulated as

$$\text{SEF: Minimize } \sum_{i \in N} \sum_{k \in M} x_{ik} + \sum_{\kappa=1}^{|\Xi|} p^{\kappa} \sum_{\rho \in A} \phi_{\rho} d_{\rho}^{\kappa} w_{\rho}^{\kappa} \quad (2a)$$

subject to

$$\sum_{k \in M} f_{\rho k}^{\kappa} + w_{\rho}^{\kappa} = 1 \quad \forall \rho \in A, \kappa = 1, \dots, |\Xi| \quad (2b)$$

$$\sum_{\rho \in A} d_{\rho}^{\kappa} f_{\rho k}^{\kappa} \leq b_k \quad \forall k \in M, \kappa = 1, \dots, |\Xi| \quad (2c)$$

$$\sum_{i \in N} x_{ik} \leq R_k \quad \forall k \in M \quad (2d)$$

$$0 \leq f_{\rho k}^{\kappa} \leq x_{ik} \quad \forall i \in N, k \in M, \rho \in S_i, \kappa = 1, \dots, |\Xi| \quad (2e)$$

$$x_{ik} \in \{0, 1\} \quad \forall i \in N, k \in M \quad (2f)$$

$$w_{\rho}^{\kappa} \geq 0 \quad \forall \rho \in A, \kappa = 1, \dots, |\Xi|. \quad (2g)$$

As a prelude to the decomposition procedure that we will use to solve this problem, let us provide the *deterministic equivalent* formulation of (2), given by

$$\text{SRD: Minimize } \sum_{i \in N} \sum_{k \in M} x_{ik} + \mathcal{Q}(x) \quad (3a)$$

subject to

$$\sum_{i \in N} x_{ik} \leq R_k \quad \forall k \in M \quad (3b)$$

$$x_{ik} \in \{0, 1\} \quad \forall i \in N, k \in M, \quad (3c)$$

where $\mathcal{Q}(x)$, the *expected recourse function*, is given by $\mathcal{Q}(x) = \sum_{\kappa=1}^{|\Xi|} p^\kappa \mathcal{Q}(x, \xi_\kappa)$, and

$$\mathcal{Q}(x, \xi_\kappa) = \text{Minimize } \sum_{\rho \in A} \phi_\rho d_\rho^\kappa w_\rho^\kappa \quad (4a)$$

subject to

$$\sum_{k \in M} f_{\rho k}^\kappa + w_\rho^\kappa = 1 \quad \forall \rho \in A \quad (4b)$$

$$\sum_{\rho \in A} d_\rho^\kappa f_{\rho k}^\kappa \leq b_k \quad \forall k \in M \quad (4c)$$

$$0 \leq f_{\rho k}^\kappa \leq x_{ik} \quad \forall i \in N, k \in M, \rho \in S_i \quad (4d)$$

$$w_\rho^\kappa \geq 0, \quad \forall \rho \in A. \quad (4e)$$

We will solve problem SEF using the L-shaped method with integer first-stage variables [30, 33]. The L-shaped method is an iterative procedure closely related to Benders' decomposition [2] that divides the extensive form given by (2) into $|\Xi| + 1$ problems: a first-stage master program, and one second-stage *subproblem* for each scenario. In particular, we employ the multicut L-shaped variation developed by Birge and Louveaux [3]. This version differs slightly from the standard L-shaped method in that we introduce a separate variable Θ_κ for each scenario $\xi_\kappa \in \Xi$, which represents the second-stage (recourse) cost corresponding to scenario ξ_κ .

The L-shaped method establishes a piecewise linear function of Θ_κ in terms of the first-stage variables (in this case, the x -variables) for each $\kappa = 1, \dots, |\Xi|$. We use duality information from (4) to find this function. Associate dual variables λ with equations (4b), $-\mu$ with Equations (4c), and $-\pi$ with the upper bounding equations of (4d). For a given realization ξ_κ and a first-stage solution \hat{x} , the dual to (4) is given by

$D(\hat{x}, \xi_\kappa)$:

$$\text{Maximize } \sum_{\rho \in A} \lambda_\rho - \sum_{k \in M} b_k \mu_k - \sum_{i \in N} \sum_{k \in M} \sum_{\rho \in S_i} \hat{x}_{ik} \pi_{ik\rho} \quad (5a)$$

subject to

$$\lambda_\rho - d_\rho^\kappa \mu_k - \pi_{ik\rho} - \pi_{jk\rho} \leq 0 \quad \forall k \in M, \rho = (i, j) \in A \quad (5b)$$

$$\lambda_\rho \leq \phi_\rho d_\rho^\kappa \quad \forall \rho \in A \quad (5c)$$

$$\lambda \text{ unrestricted}, \mu, \pi \geq 0. \quad (5d)$$

Note that because (4) has a finite optimal solution for any binary choice of \hat{x} , (5) will have a finite optimal (and thus extreme point) solution as well. (This property, in which a feasible solution exists to every subproblem provided that the first-stage decision variables are feasible to the master problem, is called *relatively complete recourse*.) The following constraints ensure that for each $\kappa = 1, \dots, |\Xi|$, the second-stage recourse cost variable Θ_κ must be at least as large as the objective function value at any dual extreme point:

$$\Theta_\kappa \geq \sum_{\rho \in A} \lambda_\rho^\ell - \sum_{k \in M} b_k \mu_k^\ell - \sum_{i \in N} \sum_{k \in M} \left(\sum_{\rho \in S_i} \pi_{ik\rho}^\ell \right) x_{ik} \quad \forall \ell \in L(\kappa), \quad (6)$$

where $L(\kappa)$ is the set of all extreme points to the polyhedron defined by (5b–d) for a given $\kappa \in \{1, \dots, |\Xi|\}$. (Note that this polyhedron is identical for any choice of \hat{x} .) Because constraints (6) define Θ_κ as a convex piecewise linear function of x , these constraints are termed *optimality cuts*. Rather than generating and adding all optimality cuts, we will add them only as necessary. The *restricted master problem* (or just “master problem”) is given by considering only a subset $L'(\kappa)$ of the set $L(\kappa)$ of all extreme points of the dual problem (5) corresponding to scenario ξ_κ . At an arbitrary iteration the master problem is given by

$$\text{Minimize } \sum_{i \in N} \sum_{k \in M} x_{ik} + \sum_{\kappa=1}^{|\Xi|} p^\kappa \Theta_\kappa \quad (7a)$$

subject to

$$\sum_{i \in N} x_{ik} \leq R_k \quad \forall k \in M \quad (7b)$$

$$\Theta_\kappa + \sum_{i \in N} \sum_{k \in M} \left(\sum_{\rho \in S_i} \pi_{ik\rho}^\ell \right) x_{ik} \geq \sum_{\rho \in A} \lambda_\rho^\ell - \sum_{k \in M} b_k \mu_k^\ell \quad \forall \ell \in L'(\kappa), \kappa = 1, \dots, |\Xi| \quad (7c)$$

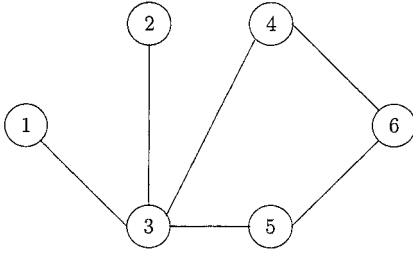


FIG. 1. Minimum cost flow demand graph.

$$x_{ik} \in \{0, 1\} \quad \forall i \in N, k \in M, \\ \Theta_\kappa \geq 0 \quad \forall \kappa = 1, \dots, |\Xi|, \quad (7d)$$

where

$$\Theta_\kappa + \sum_{i \in N} \sum_{k \in M} \left(\sum_{\rho \in S_i} \pi_{ik\rho}^\ell \right) x_{ik} \geq \sum_{\rho \in A} \lambda_\rho^\ell - \sum_{k \in M} b_k \mu_k^\ell \quad \forall \ell \in L'(\kappa)$$

represents the ℓ th optimality cut generated for scenario $\kappa = 1, \dots, |\Xi|$.

The L-shaped method then proceeds as follows. Let \hat{x} be an optimal solution to (7). For each scenario $\xi_\kappa \in \Xi$, we solve the subproblem $D(\hat{x}, \xi_\kappa)$ given by (5) and obtain the optimal dual variables $\hat{\lambda}$, $\hat{\mu}$, and $\hat{\pi}$. Based on these duals, an optimality cut

$$\Theta_\kappa + \sum_{i \in N} \sum_{k \in M} \left(\sum_{\rho \in S_i} \hat{\pi}_{ik\rho} \right) x_{ik} \geq \sum_{\rho \in A} \hat{\lambda}_\rho - \sum_{k \in M} b_k \hat{\mu}_k \quad (8)$$

is created. If (8) is violated by the current solution $(\hat{\Theta}_\kappa, \hat{x})$, we add this optimality cut to $L'(\kappa)$. If $(\hat{\Theta}_\kappa, \hat{x})$ satisfy the optimality cuts for each $\kappa = 1, \dots, |\Xi|$, then \hat{x} is an optimal solution. Otherwise, the master problem is resolved, and another iteration is performed.

3.2. Subproblem Solution Procedure

We may garner some nominal computational improvements along with an insight into the nature of the optimality cuts (8) by converting (4) to a network flow problem and utilizing specialized network flow algorithms to obtain its solution. Furthermore, the solution of these subproblems is the bottleneck operation in an upper bounding heuristic developed in Section 4, and thus improving the speed of solving the subproblems provides an improvement in the efficiency of that algorithm. For the following discussion, we examine (4) given some binary network design solution vector \hat{x} , under scenario $\kappa \in \{1, \dots, |\Xi|\}$ (i.e., the set of node-to-ring assignments has already been determined).

Smith [27] addresses the transformation of (4) to a minimum cost flow problem, which we recap here for complete-

ness. Define \mathcal{P}_ρ as the set of all rings on which demand pair ρ may be satisfied, that is, for $\rho = (i, j) \in A$, $\mathcal{P}_\rho = \{k \in M : \hat{x}_{ik} = \hat{x}_{jk} = 1\}$. Similarly, define \mathcal{D}_k as the set of demand pairs that may be satisfied on ring $k \forall k \in M$. The minimum cost flow problem is then constructed with $|M| + |A| + 1$ nodes. Nodes q_ρ correspond to each demand pair $\rho \in A$ and have a surplus of d_ρ^κ units, while nodes t_k correspond to each ring $k \in M$ and have a demand of b_k units. For each $\rho \in A$, an arc with cost 0 exists from q_ρ to all nodes t_k such that $k \in \mathcal{P}_\rho$. The flow on the arc from node q_ρ to node t_k represents the amount of demand pair ρ satisfied on ring k . Define node u as an intermediate node with a surplus/demand of $\sum_{k \in M} b_k - \sum_{\rho \in A} d_\rho^\kappa$ units. Node u is required to collect unused capacity from the rings and unsatisfied demand from each demand pair. An arc with zero cost exists from u to node $t_k \forall k \in M$ to represent unused ring capacities. Finally, an arc exists from q_ρ to u with a cost of ϕ_ρ for each $\rho \in A$, which accordingly penalizes each unit of unsatisfied demand. Solving this network flow problem, we can compute the optimal f values based on the arc flows from the q nodes to the t nodes, and the optimal w values based on the arcs from the q nodes to node u . To illustrate, consider the demand network depicted by Figure 1, and suppose that nodes 1, 2, 3, and 4 have been assigned to ring 1 and nodes 3, 4, 5, and 6 have been assigned to ring 2. The minimum cost flow network is shown in Figure 2, where the flow costs are displayed alongside the arcs and all demand and surplus values are displayed alongside their respective nodes.

The general formulation for this network flow problem is given as follows, where $\tilde{f}_{\rho k}$ is the number of channels for demand pair $\rho \in A$ satisfied on ring $k \in M$, and where \tilde{w}_ρ is the number of channels for demand pair $\rho \in A$ that remain unsatisfied.

$$\text{Minimize } \sum_{\rho \in A} \phi_\rho \tilde{w}_\rho \quad (9a)$$

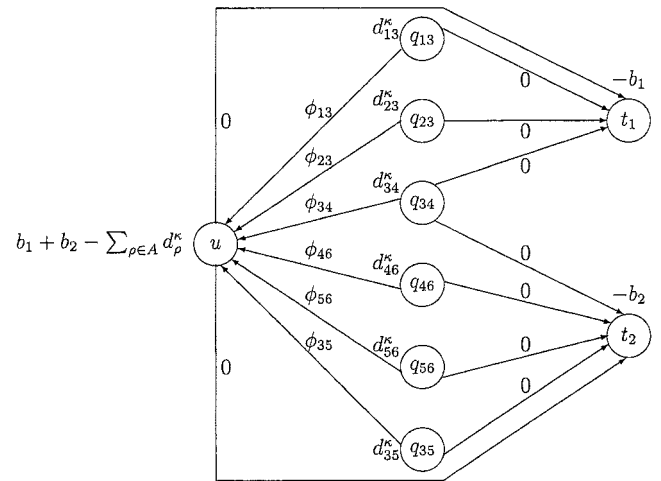


FIG. 2. Minimum cost flow graph for the demand network depicted in Figure 1.

subject to

$$\sum_{k \in \mathcal{P}_\rho} \tilde{f}_{\rho k} + \tilde{w}_\rho = d_\rho^\kappa \quad \forall \rho \in A \quad (9b)$$

$$\sum_{\rho \in \mathcal{Q}_k} \tilde{f}_{\rho k} \leq b_k \quad \forall k \in M \quad (9c)$$

$$\tilde{f}_{\rho k} \geq 0 \quad \forall \rho \in A, k \in M, \quad \tilde{w}_\rho \geq 0 \quad \forall \rho \in A. \quad (9d)$$

The dual formulation to (9) can be stated as follows, where dual variables α_ρ are associated with (9b), and variables $-\beta_k$ are associated with (9c).

$$\text{Maximize } \sum_{\rho \in A} d_\rho^\kappa \alpha_\rho - \sum_{k \in M} b_k \beta_k \quad (10a)$$

subject to

$$\alpha_\rho - \beta_k \leq 0 \quad \forall \rho \in A, k \in \mathcal{P}_\rho \quad (10b)$$

$$\alpha_\rho \leq \phi_\rho \quad \forall \rho \in A \quad (10c)$$

$$\alpha_\rho \text{ unrestricted} \quad \forall \rho \in A, \quad \beta_k \geq 0 \quad \forall k \in M. \quad (10d)$$

It is not obvious how one can recover the optimal π variables to (5) from the solution of (10), because the primal constraints (4d) have been implicitly defined by the construction of the minimum cost flow problem. The following proposition prescribes a fast method for determining the optimal solution to (5) given an optimal dual basic feasible solution to (10).

Proposition 1. Consider a scenario $\kappa \in \{1, \dots, |\Xi|\}$ and a binary network design vector \hat{x} , and define (α^*, β^*) to be an optimal basic feasible solution to (10) for κ and \hat{x} . Then there exists an optimal solution to (5) in which $\lambda_\rho^* = d_\rho^\kappa \alpha_\rho^*$ $\forall \rho \in A$ and $\mu_k^* = \beta_k^* \forall k \in M$, and in which the optimal values for $(\pi_{ik\rho}^*, \pi_{jk\rho}^*)$ for $\rho = (i, j) \in A$ are determined as

$$(\pi_{ik\rho}^*, \pi_{jk\rho}^*) = \begin{cases} (0, 0) & \text{if } \hat{x}_{ik} = 1 \text{ and } \hat{x}_{jk} = 1, \text{ or if } \lambda_\rho^* - d_\rho^\kappa \mu_k^* \leq 0 \\ (\lambda_\rho^* - d_\rho^\kappa \mu_k^*, 0) & \text{if } \hat{x}_{ik} = 0, \hat{x}_{jk} = 1, \text{ and } \lambda_\rho^* - d_\rho^\kappa \mu_k^* > 0 \\ (0, \lambda_\rho^* - d_\rho^\kappa \mu_k^*) & \text{if } \hat{x}_{ik} = 1, \hat{x}_{jk} = 0, \text{ and } \lambda_\rho^* - d_\rho^\kappa \mu_k^* > 0 \\ \text{Any solution to } \pi_{ik\rho}^* + \pi_{jk\rho}^* = \lambda_\rho^* - d_\rho^\kappa \mu_k^*, \pi_{ik\rho}^*, \pi_{jk\rho}^* \geq 0 & \text{if } \hat{x}_{ik} = 0, \hat{x}_{jk} = 0, \text{ and } \lambda_\rho^* - d_\rho^\kappa \mu_k^* > 0 \end{cases} \quad (11)$$

Proof. Because models (4) and (9) solve the same recourse subproblem, their optimal objective function values will be equal, and hence, so will the optimal objectives of (5) and (10). Given an optimal solution (α^*, β^*) to (10), we need to show that the objective function value (5a) given by the solution $(\lambda^*, \mu^*, \pi^*)$ is the same as that given by (10a) for (α^*, β^*) , and that $(\lambda^*, \mu^*, \pi^*)$ is feasible to constraints (5b–d). The equivalence of the objective function values is clear, noting that

$$\sum_{i \in N} \sum_{k \in M} \sum_{\rho \in \mathcal{S}_i} \hat{x}_{ik} \pi_{ik\rho}^* = 0$$

due to the construction of π^* values according to (11). For constraints (5b) corresponding to $\rho \in A$ and $k \in \mathcal{P}_\rho$ (i.e., $\hat{x}_{ik} = \hat{x}_{jk} = 1$), feasibility of $(\lambda^*, \mu^*, \pi^*)$ is assured due to (10b). For (5b) corresponding to $\rho = (i, j) \in A$ and $k \notin \mathcal{P}_\rho$, we must have by definition of \mathcal{P}_ρ that at least one of \hat{x}_{ik} or \hat{x}_{jk} is 0. Hence, $(\pi_{ik\rho}^*, \pi_{jk\rho}^*)$ is defined according to one of the latter three cases of (11) to ensure feasibility to (5b). Finally, constraints (5c) and (5d) hold by (10c), (10d), and the nonnegativity of the π -variables prescribed by (11). This completes the proof. ■

The dual value recovery process reveals that when $\lambda_\rho - d_\rho^\kappa \mu_k > 0$ for some demand pair $\rho = (i, j) \in A$ such that $\hat{x}_{ik} = \hat{x}_{jk} = 0$, we must choose between setting $\pi_{ik\rho}$ or $\pi_{jk\rho}$, or some convex combination thereof, to equal this dif-

ference. The outcome of this decision has a direct impact on the optimality cut being passed back to the first-stage problem: if $\pi_{ik\rho}$ is set equal to $\lambda_\rho - d_\rho^\kappa \mu_k$, it will encourage node i to be added to ring k in the next master problem iteration, and vice versa if $\pi_{jk\rho}$ is chosen to be positive. Magnanti and Wong [17] underscore the importance of properly selecting dual solutions by investigating methods by which dual solutions leading to *nondominated* Benders' cuts may be identified. (A cut is nondominated if no alternative optimal dual solution yields a stronger cut.) Unfortunately, each choice of dual values gives rise to a nondominated cut in the context of our problem. Because there are 2^h nondominated optimality cuts that can be formulated, where h is the number of “choices” as mentioned above, enumerating each possible cut is computationally prohibitive. We conducted a brief computational experiment to investigate the effectiveness of various rules for choosing among setting $\pi_{ik\rho}$ and $\pi_{jk\rho}$ such that $\pi_{ik\rho} + \pi_{jk\rho} = \lambda_\rho - d_\rho^\kappa \mu_k$. The results of this experiment indicate that setting either $\pi_{ik\rho}$ or $\pi_{jk\rho}$ equal to $\lambda_\rho - d_\rho^\kappa \mu_k$ when given a choice is the best strategy. We arbitrarily select $\pi_{ik\rho} = \lambda_\rho - d_\rho^\kappa \mu_k$ and $\pi_{jk\rho} = 0$ for the last case of (11) in our computational study in Section 5.

An alternative for resolving the choice in π -values is to consider the following problem reformulation. Define variable $v_{\rho k} = x_{ik}x_{jk}$ for $\rho = (i, j) \in A$ and $k \in M$. We may resolve this nonlinearity by enforcing the restrictions

$$v_{\rho k} \leq x_{ik}, v_{\rho k} \leq x_{jk}, \text{ and } v_{\rho k} \geq 0 \quad \forall \rho \in A, k \in M \quad (12)$$

in (3) [and thus in (7) as well]. Note that this technique is a subset of the linearization strategy employed in the Reformulation-Linearization Technique (RLT) of Sherali and Adams [22, 23], and also, by Sherali and Fraticelli [24] in the context of stochastic programming problems with *integer* recourse. We omit the lower bounding constraint $v_{\rho k} \geq x_{ik} + x_{jk} - 1$, because it will be implied at optimality. The second-stage problem may now be modified by replacing (4d) with

$$0 \leq f_{\rho k}^{\kappa} \leq v_{\rho k} \quad \forall \rho \in A, \\ \forall k \in M, \text{ given scenario } \kappa \in \{1, \dots, |\Xi|\}. \quad (13)$$

The dual problem (5) would now be modified as follows, noting that only \hat{v} values need to be passed to the subproblem:

$$D(\hat{v}, \xi_{\kappa}): \text{ Maximize } \sum_{\rho \in A} \lambda_{\rho} - \sum_{k \in M} b_k \mu_k - \sum_{\rho \in A} \sum_{k \in M} \hat{v}_{\rho k} \pi_{\rho k} \quad (14a)$$

subject to

$$\lambda_{\rho} - d_{\rho}^{\kappa} \mu_k - \pi_{\rho k} \leq 0 \quad \forall k \in M, \rho \in A \quad (14b)$$

$$\lambda_{\rho} \leq d_{\rho}^{\kappa} \phi_{\rho} \quad \forall \rho \in A \quad (14c)$$

$$\lambda \text{ unrestricted}, \mu, \pi \geq 0. \quad (14d)$$

Using this higher dimensional formulation, we would compute λ and μ values as prescribed by Proposition 1. The π values will now be set according to

$$\pi_{\rho k} = \max\{(\lambda_{\rho} - d_{\rho}^{\kappa} \mu_k), 0\} \quad \forall \rho \in A, k \in M. \quad (15)$$

Given optimal $\hat{\lambda}$, $\hat{\mu}$, and $\hat{\pi}$ values to (14), the optimality cuts derived from (14) now take the form

$$\Theta_{\kappa} + \sum_{\rho \in A} \sum_{k \in M} \hat{\pi}_{\rho k} v_{\rho k} \geq \sum_{\rho \in A} \hat{\lambda}_{\rho} - \sum_{k \in M} b_k \hat{\mu}_k. \quad (16)$$

Because no choice of optimality cuts is available in this case, a unique dual solution exists to every nondegenerate primal problem. Moreover, it can be shown that this higher dimensional representation captures *all* of the exponentially many cuts that may be identified from the original model. The proof, omitted here for brevity, constructs any optimality cut (8) that could be obtained in the original model by aggregating (16) with negative multiples of the variable upper bounding constraints in (12).

3.3. Incorporating Subproblem Data within the Master Problem

One weakness of the L-shaped method is that several master problems must be solved before optimality cuts begin to relate ample information about demand shortage penalties under various proposed node-to-ring assignments. To combat this difficulty, we offer a modification of the master problem that captures a minimum shortage penalty to be incurred for each demand pair given a proposed network design. First, let us define *baseline* demands, \underline{d} , as follows:

$$\underline{d}_{\rho} = \min_{\kappa \in \{1, \dots, |\Xi|\}} d_{\rho}^{\kappa} \quad \forall \rho \in A.$$

Proposition 2. *Given some binary vector \hat{x} , suppose (f^*, w^*) is an optimal solution to the deterministic problem (1) with x fixed at \hat{x} and demands given by \underline{d} . Then for all demand pairs $\rho \in A : w_{\rho}^* > 0$, an optimal solution to each subproblem $\mathcal{Q}(\hat{x}, \xi_{\kappa}) \quad \forall \kappa = 1, \dots, |\Xi|$ exists in which the shortage for pair ρ is at least $\underline{d}_{\rho} w_{\rho}^* + (d_{\rho}^{\kappa} - \underline{d}_{\rho})$.*

Proof. We prove this proposition by induction, given any scenario $\kappa \in \{1, \dots, |\Xi|\}$ and binary vector \hat{x} . Clearly, this proposition holds if $d^{\kappa} = \underline{d}$. By induction, assume that the result is true for a set of demands \bar{d} , where $\underline{d} \leq \bar{d} \leq d^{\kappa}$. Given the node-to-ring assignments \hat{x} of our network, we may compute optimal f and w values for any demand vector via the minimum cost flow problem stated in Section 3.2. Define (f^*, w^*) to be this optimal solution to the minimum cost flow subproblem given \hat{x} and \bar{d} . Recall that the arcs corresponding to basic solution variables will form a tree with respect to the minimum cost flow graph.

Suppose that we seek a solution to the minimum cost flow problem in which the demands have changed to $\hat{d} = \bar{d} + e_{\hat{\rho}}$, where $e_{\hat{\rho}}$ is a vector of length $|A|$ containing a 1 in the element corresponding to some $\hat{\rho} \in A$ and zeros elsewhere, for some $\hat{\rho} \in A$ such that $\bar{d}_{\hat{\rho}} < d_{\hat{\rho}}^{\kappa}$. With respect to the minimum cost flow subproblem, this change is akin to increasing the surplus of $q_{\hat{\rho}}$ by 1 and decreasing the surplus/demand of u by 1. Note that because the basic arcs of (f^*, w^*) form a tree, exactly one path (in the undirected sense) exists from $q_{\hat{\rho}}$ to u . This path consists of a (possibly null) series of arcs between the q -nodes and t -nodes, followed either by an arc from some q -node to u (call this a path of Type I), or an arc from some t -node to u (path of Type II).

Let us first consider the case in which all of the prior flow values along the arcs of the path from $q_{\hat{\rho}}$ to u oriented in the opposite direction of the path are positive. We retain an optimal basic feasible solution with respect to demands \hat{d} by increasing the flows on arcs oriented in the same direction of the path by 1, and decreasing the flows of arcs oriented in the opposite direction of the path by one. In particular, if $w_{\hat{\rho}}^* > 0$, then an arc exists directly from $q_{\hat{\rho}}$ to u , and flow is increased by one along this arc. However, if the path from $q_{\hat{\rho}}$ to u is another path of Type I, the f -values in the new

solution are modified, and the value of some other w -variable increases by 1. If a Type II path exists, we will modify the values of the f -variables, and will decrease the amount of unused capacity for some ring by decreasing the flow from node u to some t -node. In any of these cases, the previous solution continues to give rise to the same complementary slack primal and dual feasible bases. Because the w -values are nondecreasing, and because $w_{\hat{\rho}}$ increases by 1 if $w_{\hat{\rho}}^*$ was a positive value, we have that Proposition 2 holds for this case.

We next discuss the case in which some arc oriented in the opposite direction of the path from $q_{\hat{\rho}}$ to u has a flow of zero. Note that since $w_{\hat{\rho}}^* = 0$ for this case, we must simply show that no w -values will decrease in the new solution. Suppose we follow the path from $q_{\hat{\rho}}$ to u , and identify the first degenerate basic arc oriented in the opposite direction of the path. Removing this arc from the basis, we now have two components of the minimum cost flow graph with respect to the basic arcs. Note that the component containing node $q_{\hat{\rho}}$ does not contain node u , and thus a basic set of arcs is reconstituted by adding an arc from the component containing $q_{\hat{\rho}}$ to the component containing u . This connecting arc must either be from a q -node to a t -node, or from a q -node to u . A finite number of such degenerate pivots may need to be performed before a nondegenerate iteration takes place, at which time the situation reduces to the one addressed previously.

Note that we may iteratively construct any set of demands d^{κ} for $\kappa \in \{1, \dots, |\Xi|\}$ by independently incrementing the baseline demands, with Proposition 2 holding after each such incremental step. This completes the proof. ■

Suppose the deterministic ring design problem (1) is solved using the baseline demands, with one optimal solution having a set of shortages given by \hat{w} . The result of Proposition 2 implies that if $\hat{w}_{\rho} > 0$ for some $\rho \in A$, then there will exist an optimal recourse decision in which the shortage for demand pair ρ in scenario $\kappa \in \{1, \dots, |\Xi|\}$ is at least $\underline{d}_{\rho}\hat{w}_{\rho} + (d_{\rho}^{\kappa} - \underline{d}_{\rho})$ units. The following *baseline strategy* explicitly solves the baseline demand problem in the first stage, and provides a minimum penalty measure that will be incurred in the second-stage problems to the first-stage variables. First, we include the following constraints within the master problem (7):

$$\sum_{k \in M} f_{\rho k}^0 + w_{\rho}^0 = 1 \quad \forall \rho \in A \quad (17a)$$

$$\sum_{\rho \in A} \underline{d}_{\rho} f_{\rho k}^0 \leq b_k \quad \forall k \in M \quad (17b)$$

$$0 \leq f_{\rho k}^0 \leq x_{ik} \quad \forall i \in N, k \in M, \rho \in S_i, \quad \text{and} \quad w_{\rho}^0 \geq 0 \quad \forall \rho \in A, \quad (17c)$$

where $w_{\rho}^0 \forall \rho \in A$ and $f_{\rho k}^0 \forall \rho \in A, k \in M$ serve as master

problem decision variables for the solution of the baseline scenario. To fully exploit the strength of Proposition 2, we introduce a new set of binary variables to the problem. Define y_{ρ} to equal 1 if a shortage for demand pair $\rho \in A$ exists in the solution of the baseline demand, and 0 otherwise. We impose the following additional constraints in (7):

$$y_{\rho} \geq w_{\rho}^0 \text{ and } y_{\rho} \in \{0, 1\} \quad \forall \rho \in A \quad (17d)$$

$$\Theta_{\kappa} \geq \sum_{\rho \in A} \phi_{\rho} [y_{\rho}(d_{\rho}^{\kappa} - \underline{d}_{\rho}) + \underline{d}_{\rho} w_{\rho}^0] \quad \forall \kappa = 1, \dots, |\Xi|. \quad (17e)$$

Constraints (17d) state the lower bounds on the binary variables y , while constraints (17e) state the lower limits for shortage penalties given the baseline scenario solution. We will empirically examine the tradeoff between increasing the difficulty of the master problem by employing the baseline strategy embodied by (17) and the decrease in the number of problem iterations that must be executed.

3.4. Valid Inequality Generation

An important consideration in constructing a decomposition algorithm with integer first-stage variables is the tightness of the master problem. If the master problem formulation is weak, excessive computational effort must be expended in finding an optimal solution to be passed to the second-stage subproblems. We thus develop classes of valid inequalities to reduce both the effort required to solve the master problem and the number of master problems that must be solved before the algorithm terminates.

First, recall that no node will be assigned to a ring in an optimal solution unless an adjacent node in A is also assigned to the ring. (Otherwise, the node could be deleted without affecting the amount of demand that could be satisfied.) We thus incorporate the following *connectivity* constraints within the master problem.

$$x_{ik} \leq \sum_{j \in S_i} x_{jk} \quad \forall i \in N, k \in M. \quad (18)$$

Note that these constraints also implicitly impose the restriction that each ring either contains zero or at least two ADMs.

Next, observe that two SONET rings \bar{k} and \hat{k} are identical if $b_{\bar{k}} = b_{\hat{k}}$ and $R_{\bar{k}} = R_{\hat{k}}$. A solution in which some \hat{m} identical rings are utilized to route demand has as many as $(\hat{m}! - 1)$ alternative optimal solutions that may be obtained by simply reindexing the rings. Sherali and Smith [25] and Sherali et al. [26] have shown such symmetry unduly burdens a branch-and-bound approach to solving these problems by requiring the enumeration of several symmetrical branches. We state here two effective hierarchies for defeating this symmetry. For simplicity, suppose the rings are sorted in nonincreasing order of node capacities, breaking

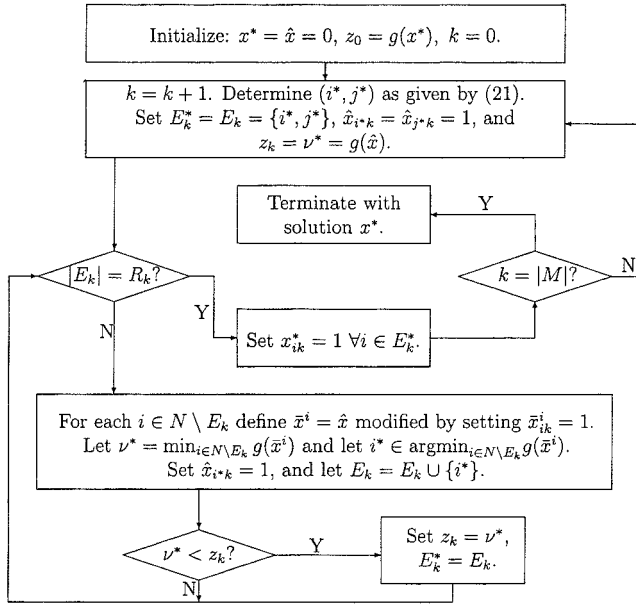


FIG. 3. Heuristic for generating node-to-ring assignments (before post-processing).

ties in nonincreasing order of demand capacity. (That is, all identical rings are indexed consecutively.)

The first hierarchy requires that the number of nodes placed on ring k is no less than the number of nodes placed on ring $k + 1$ if k and $k + 1$ are identical rings, for $k = 1, \dots, |M| - 1$:

$$\sum_{i \in N} x_{ik} \geq \sum_{i \in N} x_{i,(k+1)} \quad \forall k = 1, \dots, |M| - 1$$

such that rings k and $k + 1$ are identical. (19)

Note that although the hierarchy embodied by (19) is unlikely to completely break the symmetry within the model, it contains a constraint structure with nonzeros equal to 1 or -1 , which tends to encourage integrality of solutions (see [25], for example). Alternatively, we may impose the following hierarchy, which requires that the sum of the node indices placed on ring k is no less than the sum of node indices placed on ring $k + 1$ if k and $k + 1$ are identical rings, for $k = 1, \dots, |M| - 1$:

$$\sum_{i \in N} ix_{ik} \geq \sum_{i \in N} ix_{i,(k+1)} \quad \forall k = 1, \dots, |M| - 1$$

such that rings k and $k + 1$ are identical. (20)

This hierarchy tends to eliminate all symmetric solutions, but constraints (20) tend to intersect the linear relaxation of the master problem at fractional points. The introduction of these fractional extreme points complicates the branch-and-bound process and negates the advantages afforded by eliminating symmetry.

4. HEURISTIC PROCEDURE

Because finding an optimal solution to Problem SRD may be too difficult to achieve within imposed computational limits, we may wish to quickly determine a heuristic solution to SRD. This solution would also benefit the multicut L-shaped algorithm of Section 3 by providing an upper bound on the stochastic program, and by prescribing a solution from which an initial set of optimality cuts may be obtained.

We describe in this section the Greedy Augmenting Heuristic (GAH), which builds one ring at a time according to a sequential node addition rule. Additionally, we perform two elementary hill-climbing steps to improve the quality of solutions obtained by the heuristic. The first of these steps states that nodes should continue to be added to a ring being constructed even if the last node added does not satisfy ample additional demands to justify the cost of an ADM. The second step states that we continue to construct rings even if the previously generated ring is not cost effective. These hill-climbing steps are accompanied by a simple ring/network revision procedure to recover the best solution identified during the hill-climbing process. We justify each of these policies in the following detailed discussion of GAH.

The logic for GAH is depicted as a flowchart in Figure 3, accompanied by a summary of the key notation used in the flowchart in Table 1. We first introduce a metric that measures the benefit of adding a node to a particular ring. Given a set of node-to-ring assignments, we define the *utility* of a node $i \in N$ on a ring $k \in M$ as the additional amount of weighted demand that could be satisfied on the collection of SONET rings due to the addition of node i to ring k . For a binary vector \hat{x} representing the node-to-ring assignments that have already been made, we define $g(\hat{x}) = \sum_{i \in N} \sum_{k \in M} \hat{x}_{ik} + \mathcal{Q}(\hat{x})$, that is, $g(\hat{x})$ is the optimal solution to SRD given \hat{x} . (Recall that $\mathcal{Q}(x)$ is evaluated by solving $|\Xi|$ network flow problems as described in Section 3.2.) To determine the utility of adding some node $i \in N$ to a ring $k \in M$ with $\hat{x}_{ik} = 0$, we evaluate $g(\bar{x}) - g(\hat{x})$, where $\bar{x} = \hat{x}$ with the modification that $\bar{x}_{ik} = 1$.

We initialize the algorithm by setting the solution vector $\hat{x}_{ik} = 0 \forall i \in N$ and $k \in M$. Note that when the current ring has no nodes attached to it, the utility of each node must

TABLE 1. Notation used in the Greedy Augmenting Heuristic.

E_k	set of nodes currently assigned to ring k
E_k^*	best (nonempty) allocation of nodes to ring k found
k	current ring index
ν^*	best objective function value found
x^*	best node-to-ring assignment vector found
\hat{x}	current node-to-ring assignment vector
\bar{x}^i	node-to-ring assignment vector equal to \hat{x} , but with $\bar{x}_{ik}^i = 1$ for current ring k
\bar{x}^{ij}	node-to-ring assignment vector equal to \hat{x} , but with $\bar{x}_{ik}^{ij} = \bar{x}_{jk}^{ij} = 1$ for current ring k
z_k	current objective function value given assignments to the first k rings

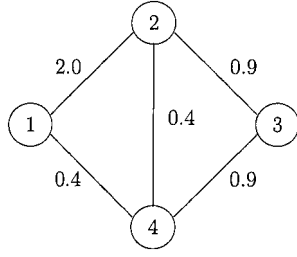


FIG. 4. Illustration of property 1.

equal zero. To prevent an arbitrary assignment of the initial node for an empty ring, we determine two initial nodes by identifying the “best” demand pair that could be added to a new ring. Define \bar{x}^{ij} as \hat{x} modified by setting $\bar{x}_{ik}^{ij} = \bar{x}_{jk}^{ij} = 1$, where k is the new ring being constructed. We initialize ring k with nodes i^* and j^* satisfying

$$(i^*, j^*) \in \underset{(i,j) \in A}{\operatorname{argmin}} \{g(\bar{x}^{ij})\}. \quad (21)$$

Given a nonempty ring, we may now meaningfully compute the utility of each node $i \in N \setminus \{i^*, j^*\}$ as described above. Note that the demand capacity restriction of a ring is implicitly enforced at each step of the algorithm by the $\mathcal{Q}(x)$ component of $g(x)$. The following property justifies why GAH continues to add nodes to the current ring until the node cardinality limit is reached, rather than stopping once the cost of an ADM outweighs the additional demands being satisfied on the network.

Property 1. Consider the construction of ring k by GAH, and let $g(x_k^a)$ be the objective value obtained by the heuristic in constructing ring k , when ring k contained a nodes. Then $g(x_k^a)$ is not necessarily quasiconvex in terms of a .

Example. Consider the demand graph given by Figure 4 as a one-scenario (deterministic) example whose penalties are given by the values alongside the arcs, and whose demands are given by $d_\rho^1 = 1$ for all $\rho \in A$. Suppose that for each ring, the demand capacity restriction is set to $b_k = 5$ channels and that the node cardinality restriction is set to $R_k = 4$ nodes. Table 2 describes the construction of the first ring by GAH for this example. In this example, a two-node ring is less expensive than a three-node ring, but more expensive than the (optimal) four-node ring.

It is thus necessary to track the best objective value z_k

TABLE 2. GAH solution of Figure 4 problem.

Iteration	Nodes assigned	Shortage penalty	Number of ADMs	Ring cost (v^*)
Initialize	1,2	2.6	2	4.6
1	1,2,3	1.7	3	4.7
2	1,2,3,4	0	4	4

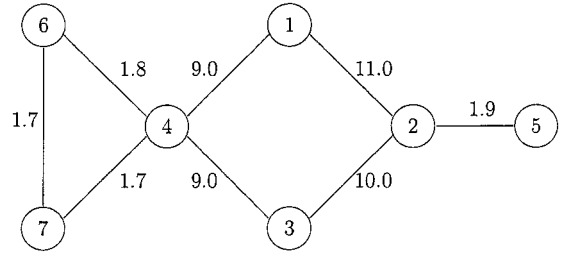


FIG. 5. Illustration of Property 2.

achieved in the construction of the ring, the set of nodes E_k^* assigned to ring k leading to that solution, and the current set of nodes E_k assigned to ring k . At the end of the construction of this ring, we set $x_{ik}^* = 1 \forall i \in E_k^*$.

Property 2 illustrates a similar behavior in the overall construction of the rings, wherein a ring may be constructed that temporarily worsens the objective function value, but ultimately leads to the construction of another ring that results in an improved solution.

Property 2. The sequence of objective function values z_k , $k = 1, \dots, |M|$, is not necessarily quasiconvex in terms of k .

Example. We again consider a one-scenario instance whose demand network is depicted in Figure 5, where penalties are displayed alongside the arcs and $d_\rho^1 = 1$ for all $\rho \in A$, and instance parameters are defined as $|M| = 3$, $b_k = 6 \forall k \in M$, and $R_k = 4 \forall k \in M$. Table 3 summarizes the GAH steps for this example. Note that $z_1 < z_2$ but $z_1 > z_3$ for this example, which justifies the construction of new rings even if previously constructed rings are not cost efficient.

Observe that deleting ring 2 in the previous example would result in a better solution with objective function 8.9 (7 ADMs plus a penalty of 1.9 for not satisfying demand between nodes 2 and 5). In general, consider a solution \hat{x} with objective function value \hat{z} , where $\mathcal{F}_k(\hat{x})$ denotes the expected weighted amount of demand satisfied on ring k for $k \in M$, and $|E_k^*|$ represents the number of nodes attached to ring k for $k \in M$. If there exists a \bar{k} such that $|E_{\bar{k}}^*| - \mathcal{F}_{\bar{k}}(\hat{x}) > 0$, then removing ring \bar{k} from the current solution will decrease \hat{z} by at least $|E_{\bar{k}}^*| - \mathcal{F}_{\bar{k}}(\hat{x})$.

Based on this fact, we develop the following postprocessing procedure. Given a current solution \hat{x} , identify ring $\bar{k} \in \operatorname{argmax}_{k \in M} \{|E_k^*| - \mathcal{F}_k(\hat{x})\}$. If $|E_{\bar{k}}^*| - \mathcal{F}_{\bar{k}}(\hat{x}) > 0$,

TABLE 3. GAH solution of Figure 5 problem.

Number of rings (k)	E_k^*	Shortage	Total number of ADMs	Total cost (z_k)
0		46.1	0	46.1
1	1,2,3,4	7.1	4	11.1
2	2,5	5.2	6	11.2
3	4,6,7	0	9	9

delete ring \bar{k} , modify \hat{x} by setting $\hat{x}_{i\bar{k}} = 0 \forall i \in E_k^*$, determine the optimal f -variables given the remaining node-to-ring assignments, and repeat. The heuristic is complete when $|E_k^*| - \mathcal{J}_k(\hat{x}) \leq 0 \forall k \in M$. Note that we do not simultaneously delete all rings meeting this criterion, as the deletion of one ring may change the demand-to-ring assignments to make the inclusion of each other ring cost effective.

5. COMPUTATIONAL RESULTS

We begin the empirical investigation of our algorithms in Section 5.1 by describing how two groups of test instances were generated for this study, and then demonstrate the impact of our various strategies using this test data in Section 5.2. All computations were executed on a SUN-Ultra 10 Workstation with 256 MB of RAM using the CPLEX 7.0 callable library to solve integer programming and network flow problems.

5.1. Test Instance Generation

To test the computational efficacy of the procedures developed in this article, we generated two groups of test data: one generated randomly according to graph generation policies for SONET problems given in the literature, and one adapted from the study of Goldschmidt et al. [10]. The first group contains three problem sets (Sets 1, 2, and 3), each containing 10 stochastic SONET instances, resulting in 30 total demand graphs. Each instance in Set 1 contains 10 scenarios, while Set 2 instances contain 20 scenarios, and Set 3 instances contain 30 scenarios. For each instance, $|M| = 4$ SONET rings may be constructed, b_k is set to 48 $\forall k \in M$, and R_k is set to 5 $\forall k \in M$. The policy of creating instances with identical rings is in line with prior SONET studies, and makes the solution of these problems more challenging due to the symmetry that arises in such instances. The 30 demand graphs are independently created by randomly generating 10-node graphs with 40% density, leading to the random construction of 18 edges. For each demand pair $\rho \in A$, the shortage penalty ϕ_ρ is randomly generated according to a uniform distribution on the interval $(0, 1]$, and an expected demand d_ρ^e is randomly generated as an integer according to a uniform distribution on the interval $[4, 10]$. Finally, the demands for each scenario $\kappa = 1, \dots, |\Xi|$ are set equal to $d_\rho^\kappa = d_\rho^e$ with probability 0.3, $d_\rho^\kappa = d_\rho^e + 1$ or $d_\rho^\kappa = d_\rho^e - 1$, each with probability 0.2, $d_\rho^\kappa = d_\rho^e \pm 2$ with probability 0.1, and $d_\rho^\kappa = d_\rho^e \pm 3$ with probability 0.05.

The second group of problems, GOLDGL, was adapted from a class of 10 test instances used by Goldschmidt et al. [10]. Each of these instances contains 15 nodes, and has an edge density of approximately 45%, with the exact number of edges varying from 34 to 62 in this set. The demands and capacities for this set are given in terms of megabits per second (Mbps). The average edge demand is 7.5 Mbps, and varies between 4.5 and 10.5 Mbps. The ring capacity is

given by 155 Mbps. For more details on these instances, the reader is referred to the original work, particularly regarding the discussion of 15-node geometric low-density instances. To adapt these instances to the stochastic problem discussed in this article, we permitted the use of five rings, and varied the common value of $R (=R_k \forall k \in M)$ between 5 and 10. (For simplicity, we will refer to R rather than R_k for the computational discussion, because $R_{k1} = R_{k2}$ for $k1, k2 \in M$ in all test instances.) Ten scenarios are generated for each instance as done for Set 1 above: the stochastic demands are varied from the expected demand by 0, ± 1 , ± 2 , or ± 3 Mbps according to the same distribution as before. The shortage penalties were randomly generated from a uniform distribution on the interval $(0, 0.15]$, rather than $(0, 1]$ as before. This new interval was chosen to ensure that there did not exist an overriding bias towards requiring that all demands be satisfied due to large penalties, or a bias towards satisfying no demands due to small penalties. In the former case, we do not recommend this stochastic decomposition procedure, but instead suggest an alternative procedure that would require the satisfaction of these high-penalty demands rather than penalizing them if they are not satisfied. (In fact, initial computational results indicate that the extensive form approach outperforms any combination of our decomposition strategies when the penalties are scaled to be very large.)

5.2. Empirical Experimentation

In this section, we first demonstrate the impact of our various strategies on Sets 1, 2, and 3 (our randomly generated test instances). Having determined which combination of strategies seems to result in the most efficient overall algorithm, we validate our conclusions by testing them on the GOLDGL data set adapted from Goldschmidt et al. [10].

We begin by briefly examining the quality of GAH before focusing on the performance of our exact procedures. Table 4 compares the average heuristic objective function value ("Heur Soln") with the average optimal solution over 10 instances in each of the three test sets, and provides the average CPU time in seconds required to complete the heuristic. Although GAH executes quickly and does manage to identify the optimal solution in 3 of the 30 instances, the average optimality gap is still large enough to warrant the use of an exact algorithm. For the remaining implementations of the L-shaped algorithm, we use GAH to obtain an

TABLE 4. Average performance of the Greedy Augmenting Heuristic.

Instance set			#		
	Heur soln	Optimum	Opt	Opt gap	CPU (secs)
Set 1	17.49	15.24	0	14.87%	1.56
Set 2	16.60	15.47	2	7.27%	3.12
Set 3	17.20	15.31	1	12.47%	4.82

Opt: number of times the heuristic identifies the optimal solution.
Opt gap: average $(UB - \text{Optimal Value})/(\text{Optimal Value}) \times 100\%$.

TABLE 5. Algorithmic performance without baseline demands after 3 CPU hours.

Instance set	Iter	LB	UB	Relative gap	Opt. gap
Set 1	43.6	12.73	17.46	26.96%	14.65%
Set 2	34.6	12.28	16.60	25.48%	7.27%
Set 3	31.1	12.09	16.97	28.44%	10.94%

Iter: average number of iterations performed within the time limit.
 LB: average lower bound obtained after 3 CPU hours.
 UB: average best solution found within 3 CPU hours.
 Relative gap: average $(UB - LB)/UB \times 100\%$.
 Opt. gap: average $(UB - \text{Optimal Value})/(\text{Optimal Value}) \times 100\%$.

initial upper bound as well as an initial set of optimality cuts based on this heuristic solution. Observe that given any feasible solution \hat{x} , each subproblem (10) can be solved to obtain valid optimality cuts for the master problem. Because these initial optimality cuts based on the GAH solution are readily obtainable and do not worsen any of the implementations investigated, they are used in all of the following L-shaped implementations.

We now analyze the impact of the following implementation options for the L-shaped method: including a baseline scenario within the master problem, incorporating certain valid inequalities to further tighten the master problem, and reformulating the master and subproblems with quadratic variables (which lead to stronger optimality cuts). For each of the following tests, we execute GAH to provide an initial upper bound for the algorithm, and use this solution to obtain a set of initial optimality cuts.

We first demonstrate the ineffectiveness of the L-shaped method when the baseline demand scenario is *not* incorporated within the master problem. Even when using (18) and (19) in the master problem along with the initial optimality cuts provided by the heuristic solution obtained from GAH, none of the 30 test instances could be solved to optimality within 3 hours. [The performance of this algorithm does not improve when the valid inequalities (18) and (19) are omitted from the model.] In fact, Table 5 demonstrates that the average final gap between the lower and upper bounds identified by the algorithm is considerable. The strategy of using the best-known solution after the 3-hour mark would be only slightly better than using the heuristic solution; in just 3 out of the 30 instances is the solution found at this point better than the heuristic solution.

Having established that the baseline demand scenario is critical to our algorithm, we analyze the effectiveness of incorporating the connectivity valid inequalities given by (18) and the symmetry-breaking inequalities given by (19) within the master problem. [A preliminary set of tests demonstrated that the node-based hierarchy (19) is more effective for this class of problems than the demand-based hierarchy (20).] Table 6 illustrates the average running times in terms of CPU minutes and seconds (reported as mm:ss) required by our algorithm using combinations of the connectivity and symmetry-breaking constraints. All runs were performed with the baseline demand scenario incorporated

in the master problem. The symmetry-breaking constraints are clearly vital to the efficiency of this algorithm, while the connectivity constraints provide a slight improvement in the average running time. Without the symmetry-breaking constraints (19), one instance in each of Set 2 and Set 3 could not be solved within the imposed time limit of 3 hours. Hence, the averages listed for Sets 2 and 3 are only over the nine instances that all implementations could solve within 3 hours. The true benefit of incorporating (19) within the master problem is thus dampened somewhat in Table 6 by disregarding the two instances in which these cuts are most vital. The average CPU times over *all* Set 2 and Set 3 instances using both (18) and (19) are 1:31 and 2:09, respectively. When only inequalities (19) are used, the average CPU times for all Set 2 and Set 3 instances are 2:27 and 2:12, respectively. We thus incorporate both of these sets of valid inequalities within the master problem in all future computational tests.

Our last set of computational tests on Sets 1, 2, and 3 compares in detail four implementation options of our overall L-shaped algorithm versus simply solving the extensive form of the stochastic program. We denote the large integer programming approach [as formulated in (2)] as “Extensive Form” in Table 7. The “Multicut” approach uses the multicut L-shaped algorithm augmented by the heuristic, valid inequality, and baseline enhancements mentioned thus far. The “Single Cut” option is performed in the same manner, with the following modification to the multicut L-shaped approach. After solving each master problem, instead of adding an optimality cut to the master problem for each subproblem violating (8), we aggregate all generated optimality cuts at that iteration into one single optimality cut. This single cut is formed by multiplying each generated optimality cut by the probability of realizing its corresponding scenario and summing them together. Note that, whereas this strategy would normally allow Θ to be considered as one variable instead of $|\Xi|$ separate variables, we retain the individual Θ_κ variables for $\kappa = 1, \dots, |\Xi|$ to garner the effectiveness of the baseline demand subproblem. The “Quadratic” option refers to the use of the reformulated dual subproblem (14) requiring the addition of quadratic variables $v_{\rho k}$ for $\rho \in A$ and $k \in M$, developed in

TABLE 6. Effectiveness of connectivity (18) and symmetry-breaking (19) valid inequalities.

Instance set	(18) and (19)		(19) only		(18) only		No valid inequalities	
	CPU	Iter	CPU	Iter	CPU	Iter	CPU	Iter
Set 1	0:34	2.5	0:33	2.1	2:17	10.0	2:27	9.6
Set 2*	0:37	2	0:46	2.22	8:02	16.11	8:03	15.11
Set 3*	1:08	2.89	1:14	3	13:55	30.22	18:00	30.56

CPU time reported in minutes and seconds (mm:ss).

Iter: average number of iterations required.

* One instance could not be solved within 3 hours for combinations not using (19).

TABLE 7. Effectiveness of various L-shaped implementations and the extensive form.

Instance	Multicut		Quadratic		Single cut		Quad + single		Extensive form
	CPU	Iter	CPU	Iter	CPU	Iter	CPU	Iter	CPU
(a) Set 1 instances									
1	0:14	1	0:21	1	0:14	1	0:16	1	7:19
2	0:09	1	0:08	1	0:06	1	0:08	1	2:39
3	0:48	6	0:37	2	1:00	4	1:01	5	3:43
4	0:07	1	0:10	1	0:07	1	0:07	1	1:57
5	0:07	1	0:09	1	0:07	1	0:10	1	1:51
6	0:06	1	0:10	1	0:07	1	0:06	1	2:06
7	0:21	2	0:08	1	0:20	2	0:06	1	1:44
8	1:43	4	2:18	4	1:03	4	1:41	4	3:27
9	0:37	4	1:15	5	0:29	4	0:34	4	2:38
10	0:88	4	0:52	2	0:58	3	0:44	3	4:46
Avg	0:34	2.5	0:37	1.9	0:27	2.2	0:29	2.2	3:13
(b) Set 2 instances									
1	0:08	1	0:11	1	0:09	1	0:10	1	5:55
2	0:15	1	0:17	1	0:12	1	0:13	1	19:14
3	9:37	7	8:31	4	6:40	7	5:02	5	51:35
4	1:45	3	4:16	3	2:28	3	2:26	3	25:04
5	0:08	1	0:08	1	0:09	1	0:07	1	6:20
6	0:32	2	1:05	2	0:15	1	0:12	1	8:31
7	1:11	3	2:09	3	0:55	3	1:34	4	11:36
8	0:30	2	0:56	2	0:38	2	0:40	2	8:19
9	0:25	2	0:36	2	0:26	2	0:37	2	6:41
10	0:32	3	0:44	3	0:30	3	0:46	3	8:27
Avg	1:31	2.5	1:54	2.2	1:14	2.4	1:11	2.3	15:12
(c) Set 3 instances									
1	11:18	11	9:04	5	6:03	11	10:21	9	52:57
2	0:17	1	0:42	2	0:13	1	0:14	1	20:54
3	0:28	2	0:35	2	0:24	2	0:26	2	11:48
4	1:21	4	2:35	4	1:11	4	1:13	4	12:31
5	1:36	4	1:57	4	1:11	4	1:26	4	14:54
6	0:51	2	1:42	2	0:34	2	0:47	2	18:30
7	1:35	2	1:43	2	1:21	2	1:09	2	38:56
8	1:50	5	3:07	5	1:52	6	1:40	5	13:52
9	1:18	2	1:41	2	1:08	2	0:54	2	22:01
10	0:57	4	0:44	2	0:39	3	0:35	3	11:08
Avg	2:09	3.7	2:23	3.0	1:34	3.7	1:53	3.4	21:45

Section 3.2. Finally, “Quad + Single” denotes the implementation of the quadratic model in conjunction with the single cut strategy.

The results in Table 7 indicate that the single cut implementations outperform their disaggregated counterparts for these runs. Surprisingly, even though the disaggregated (multicut) version provides stronger cuts for the master problem, our aggregation strategy for the linear model resulted in slightly *fewer* overall iterations for the L-shaped algorithm. The aggregation of optimality cuts thus becomes an attractive computational strategy, as it reduces the size and difficulty of each master problem while requiring roughly the same number of iterations to solve each instance. As expected, the quadratic model option was effective in reducing the number of overall iterations. Unfortunately, this reduction in iterations was not always sufficient to make up for the increase in difficulty of solving the master problem. Due to the inconsistency of the running times when using the quadratic model (even when used in conjunction with the single cut strategy), we do not recommend its use within the L-shaped algorithm. However, the effectiveness of this option on some of the test instances (e.g., on Set 2 instances) implies that it might indeed be useful on certain classes of problems for which the quadratic representation provides a substantial reduction in the number of iterations executed. Finally, these results demonstrate that the extensive form option is clearly inferior to any of the four L-shaped implementations for this set of test data.

To validate these conclusions, we conducted a set of summary experiments on the GOLDGL test set as R varies from 5 to 10. As in the previous tests, we found that including the baseline scenario in the master problem is necessary to solve these instances, and that the inclusion of valid inequalities (18) and (19) improves the performance of each implementation option model. Using these model improvements, along with the optimality cuts based on the initial heuristic solution, we tested the same five strategies on GOLDGL as we did for Sets 1, 2, and 3. A summary table giving the average performance of each algorithm over the 10 GOLDGL test instances, with R ranging between 5 and 10, is given in Table 8, where the “Best” column is the number of instances for which that implementation optimized the problem in the least amount of CPU time.

These results demonstrate that the master problem enhanced by the baseline scenario and valid inequalities is

TABLE 8. Average algorithmic performance on the GOLDGL set.

R	Multicut			Quadratic			Single cut			Quad + single			Ext. form	
	CPU	Iter	Best	CPU	Iter	Best	CPU	Iter	Best	CPU	Iter	Best	CPU	Best
5	1:31	1	3	2:22	1	0	1:40	1	5	1:50	1	2	44:45	0
6	0:43	1	1	0:38	1	1	0:34	1	7	0:37	1	1	8:10	0
7	1:00	1.2	1	1:17	1.2	1	1:22	1.2	7	1:38	1.2	1	5:33	0
8	4:01	2.6	1	3:52	2.5	0	4:04	2.9	6	6:09	2.8	1	4:21	2
9	7:14	3.9	1	7:18	3.5	0	7:45	4.6	1	8:25	4.1	3	4:41	5
10	12:14	4.4	1	13:32	3.8	2	13:14	5.1	0	14:16	4.4	2	7:55	5

Best: number of times implementation used the least CPU time out of all implementations.

strong enough to solve each of the 10 instances in the first iteration for $R = 5$ and 6. As R increases, the number of iterations required for the L-shaped options tends to increase. Moreover, for $R \geq 9$, solving the extensive form integer program is preferable to any L-shaped approach on the average. Interestingly, the extensive form option performs very poorly for problems with $R = 5$, but tends to improve as R increases to 8, and then only slightly worsens as R increases to 9 and 10. As for the individual L-shaped implementations, note that, once again, the quadratic formulation tends to reduce the number of iterations required. The cut aggregation strategy increases the number of iterations on the average for both the linear and quadratic models, but despite this increase, the single cut implementation is usually the most efficient one when $R \leq 8$. (The average performance of the single cut option is worse than the average multicut performance due to one instance for which the multicut implementation performs significantly better than the single cut implementation for $R = 5, 7$, and 8.) Generally speaking, the single cut L-shaped method with the baseline scenario and valid inequalities still tends to be the best algorithm on this set of test data for $R \leq 8$. However, for all of these tests, the performance of the multicut implementation is less variable, and serves as a viable alternative to the single cut implementation. Also, for large values of R relative to the number of nodes in the problem, we recommend a different methodology (perhaps even the solution of the extensive form integer program) instead of any of the L-shaped implementations investigated in this article.

6. SUMMARY AND CONCLUSIONS

We consider a stochastic intraring SONET design problem that generalizes the deterministic SONET ring design problem. We begin by providing a basic decomposition framework based on the L-shaped method, whose subproblems may be solved by a transformation to a minimum cost flow problem. Using concepts derived from a dual recovery algorithm for these subproblems, we develop an alternative formulation capable of generating strong optimality cuts for the master program. Following this, we identify several classes of valid inequalities along with a novel technique for including subproblem data into the master problem. These techniques for obtaining strong lower bounds to the problem are used in conjunction with an effective upper bounding heuristic executed after the solution of each master problem. We then examine the effectiveness of our methodologies on a suite of SONET test instances to demonstrate the benefit of our overall prescribed algorithm, versus simply executing a rudimentary L-shaped implementation or solving the extensive form program.

Several opportunities for future research exist. One extension would consider the case in which demands cannot be split among multiple rings. Another might include the presence of bidirectional rings (BLSR) rather than the unidirectional type studied herein. The additional algorithmic

challenge would be substantial, because in either of these cases, the subproblem investigated would be an integer program rather than a linear program. Because duality information cannot directly be obtained for these subproblems, a different approach must be taken in this case. We will research methods of capturing cutting planes from integer subproblems in this context in our future research.

Another interesting extension of this problem might consider the addition of ADMs to the network in a future decision-making stage. Such an extension would require the repeated solution of mixed-integer second-stage problems. Although the current state of the art in stochastic integer programming may not permit the direct solution of such a problem, the heuristic developed in this article could readily be modified to account for these second-stage network design decisions.

One final avenue of study regards the single cut versus multicut approaches for the integer L-shaped method. Works by Birge and Louveaux [3] and Gassmann [8] conduct experiments on stochastic *linear* programs, suggesting that the multicut approach is usually preferable when the number of realizations $|\Xi|$ is not significantly larger than the number of first-stage constraints. We are unaware of similar experiments on stochastic programs with first-stage integer variables. Our findings indicate that the single-cut version is preferable even though the number of first-stage constraints greatly exceeds the number of scenarios. One reason for this behavior might be that continuous relaxations of the master problem must be solved many times within a branch-and-bound framework, making additional constraints more burdensome than when the master problem is a linear program.

Acknowledgments

The authors are grateful to Julie Higle, Nan Kong, two anonymous referees, the editor, and an associate editor for insightful comments that enhanced this article.

REFERENCES

- [1] G. Belvaux, N. Boissin, A. Sutter, and L.A. Wolsey, Optimal placement of add/drop multiplexers: static and dynamic models, *Eur J Operat Res* 108 (1998), 26–35.
- [2] J.F. Benders, Partitioning procedures for solving mixed variables programming problems, *Numer Math* 4 (1962), 238–252.
- [3] J.R. Birge and F.V. Louveaux, A multicut algorithm for two-stage stochastic linear programs, *Eur J Operat Res* 34 (1988), 384–392.
- [4] J.R. Birge and F.V. Louveaux, *Introduction to stochastic programming*, Springer, New York, 1997.
- [5] S. Cosares and I. Saniee, An optimization problem related to balancing loads on SONET rings, *Telecommun Syst* 3 (1994), 165–181.
- [6] M. Dell'Amico, M. Labbé, and F. Maffioli, Exact solution

- of the SONET ring loading problem, *Operat Res Lett* 25 (1999), 119–129.
- [7] S. Dye, L. Stougie, and A. Tomasgard, The stochastic single node service provision problem, *Naval Research Logistics* 50 (2003), 869–887.
 - [8] H.I. Gassmann, MSLiP: a computer code for the multistage stochastic linear programming problem, *Math Program* 47 (1990), 237–254.
 - [9] O. Goldschmidt, D.S. Hochbaum, A. Levin, and E.V. Olinick, The SONET edge-partition problem, *Networks* 41 (2003), 13–23.
 - [10] O. Goldschmidt, A. Laugier, and E.V. Olinick, SONET/SDH ring assignment with capacity constraints, *Discrete Appl Math* 129 (2003), 99–128.
 - [11] G.J. Gutierrez, P. Kouvelis, and A.A. Kurawarwala, A robustness approach to uncapacitated network design problems, *Eur J Operat Res* 94 (1996), 362–376.
 - [12] N. Karunanithi and T. Carpenter, SONET ring sizing with genetic algorithms, *Comput Operat Res* 24 (1997), 581–591.
 - [13] M. Laguna, Clustering for the design of SONET rings in interoffice telecommunications, *Manage Sci* 40 (1994), 1533–1541.
 - [14] M. Laguna, Applying robust optimization to capacity expansion of one location in telecommunications with demand uncertainty, *Manage Sci* 44 (1998), S101–S110.
 - [15] Y. Lee, J. Han, and S. Kim, A physical ring design problem of synchronous optical networks for mass market multimedia telecommunication services, *INFORMS Fall Conference*, Philadelphia, PA, November 7–10, 1999.
 - [16] Y. Lee, H.D. Sherali, J. Han, and S. Kim, A branch-and-cut algorithm for solving an intra-ring synchronous optical network design problem, *Networks* 35 (2000), 223–232.
 - [17] T.L. Magnanti and R.T. Wong, Accelerating Benders decomposition: algorithmic enhancement and model selection criteria, *Operat Res* 29 (1981), 464–484.
 - [18] Y. Malashenko and N.M. Novikova, The generalized problem of analyzing a multicommodity network, *Soviet J Comput Syst Sci* 26 (1990), 154–167.
 - [19] J.M. Mulvey, R.J. Vanderbei, and S.A. Zenios, Robust optimization of large scale systems, *Operat Res* 43 (1995), 264–281.
 - [20] J. Rosenberger, E.L. Johnson, and G.L. Nemhauser, A robust fleet assignment model with hub isolation, *Transport Sci* (to appear).
 - [21] A. Schrijver, P. Seymour, and P. Winkler, The ring loading problem, *SIAM J Discrete Math* 11 (1999), 1–14.
 - [22] H.D. Sherali and W.P. Adams, A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems, *SIAM J Discrete Math* 3 (1990), 411–430.
 - [23] H.D. Sherali and W.P. Adams, A hierarchy of relaxations and convex hull characterizations for mixed-integer zero-one programming problems, *Discrete Appl Math* 52 (1994), 83–106.
 - [24] H.D. Sherali and B.M.P. Fraticelli, A modification of Benders’ decomposition algorithm for discrete subproblems: an approach for stochastic programs with integer recourse, *J Global Optimizat* 22 (2002), 319–342.
 - [25] H.D. Sherali and J.C. Smith, Improving discrete model representations via symmetry considerations, *Manage Sci* 47 (2001), 1396–1407.
 - [26] H.D. Sherali, J.C. Smith, and Y. Lee, Enhanced model representations for an intraring synchronous optical network design problem allowing demand splitting, *INFORMS J Comput* 12 (2000), 284–298.
 - [27] J.C. Smith, Algorithms for distributing telecommunication traffic on a multiple-ring SONET-based network, *Eur J Operat Res* 154 (2004), 659–672.
 - [28] H. Soroush and P.B. Mirchandani, The stochastic multicommodity flow problem, *Networks* 20 (1990), 121–155.
 - [29] A. Sutter, F. Vanderbeck, and L.A. Wolsey, Optimal placement of add/drop multiplexers: heuristic and exact algorithms, *Operat Res* 46 (1998), 719–728.
 - [30] 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 (1969), 638–663.
 - [31] S.T. Waller and A.K. Ziliaskopoulos, Stochastic dynamic network design problem, *Transport Res Rec* 1771 (2001), 106–113.
 - [32] O. Wasem, T. Wu, and R. Cardwell, Survivable SONET networks-design methodology, *IEEE J Selected Areas Commun* 12 (1994), 205–212.
 - [33] R. Wollmer, Two-stage linear programming under uncertainty with 0-1 integer first-stage variables, *Math Program* 19 (1980), 279–288.
 - [34] T. Wu, *Fiber network service survivability*, Artech House, Boston, MA, 1992.
 - [35] T. Wu and M.E. Burrowes, Feasibility study of a high-speed SONET self-healing ring architecture in future interoffice networks, *IEEE Commun Mag* 28 (1990), 33–43.