A Decomposition Algorithm for Local Access Telecommunications Network Expansion Planning

Anantaram Balakrishnan,
Thomas L. Magnanti
and
Richard T. Wong

WP# 3496-92-MSA September, 1992

A Decomposition Algorithm for Local Access Telecommunications Network Expansion Planning †

Anantaram Balakrishnan *§
Thomas L. Magnanti §

Sloan School of Management
M. I. T.
Cambridge, MA

Richard T. Wong

AT&T Bell Laboratories
Holmdel, NJ

Revised: September 1992

† This research was initiated through a grant from GTE Laboratories Incorporated, Waltham, Massachusetts

* Supported in part by a research award from the AT&T Foundation

§ Supported in part by a NATO Collaborative research grant CRG 900281

Abstract

Growing demand, increasing diversity of services, and advances in transmission and switching technologies are prompting telecommunication companies to rapidly expand and modernize their networks. This paper develops and tests a decomposition methodology to generate cost-effective expansion plans, with performance guarantees, for one major component of the network hierarchy—the local access network connecting customers to the local switching center. The model captures economies of scale in facility costs, and addresses the central tradeoff between installing concentrators and expanding cables to accommodate demand growth. By exploiting the special tree and routing structure of the expansion planning problem, our solution method integrates two major algorithmic strategies from mathematical programming—the use of valid inequalities, obtained by studying a problem's polyhedral structure, and dynamic programming, which can be used to solve an uncapacitated version of the local access network expansion planning problem. The computational results for three actual test networks demonstrate that this enhanced dynamic programming algorithm, when embedded in a Lagrangian relaxation scheme (with problem preprocessing and local improvement), is very effective in generating good upper and lower bounds: implemented on a personal computer, the method was able to generate solutions that are within 1.2 to 7.0% of optimality. In addition to developing a successful solution methodology for a practical problem, this paper illustrates the possibility of effectively combining decomposition methods and polyhedral approaches.

Keywords: *Integer programming decomposition, concentrator location, telecommunications planning, polyhedral methods*

1. Introduction

Advances in switching and transmission technologies combined with growing demand, increasing diversity of services, and deregulation of the telecommunications industry have prompted telephone companies to rapidly upgrade and expand their networks. Modernization of the local access networks, which connect switching centers to customers, is a particularly important priority since these networks account for over 50% of the total investment in communication facilities (according to Standard and Poor's Industry Surveys [1992], the total value of plant in 1990 exceeded 250 billion dollars in the U. S. alone); yet they are not as technologically advanced as the higher levels–the long-distance and inter-office networks–in the telecommunications network hierarchy. For instance, over 80% of the local access networks still use analog transmission over copper cables. However, recent technology and cost trends in the industry have improved the economic and technical viability of introducing electronic switching and fiber optic transmission to increase capacity in local access networks.

The new technologies such as electronic remote units (or multiplexers) introduce discrete choice decisions and spatial couplings between different parts of the network, thus vastly increasing the number of possible ways to meet growing demand. Traditional manual planning methods that consider only the option of adding more cables to expand capacity are no longer adequate. Since telecommunication investments are so expensive (total annual investments by U. S. local exchange companies is approximately $20 billion, *Telephony* [1991]), a cost effective network expansion plan can offer considerable economic value. For instance, Jack, Kai, and Shulman [1992] report savings of over $30 million per year at GTE using an interactive decision support system to assist network planners.

This paper develops and tests an optimization-based methodology to identify a minimum cost network expansion plan to meet increasing demand. We formulate an integer programming model, validated by consulting network planners in industry, that captures the essential tradeoffs between concentrator location and cable expansion, and accommodates economies of scale in investment and operating costs. Although the model approximates the expansion costs as piecewise-linear concave functions, and does not consider investment timing decisions, it is an important building block for detailed, multi-period planning systems (see, for example, Shulman and Vachani [1990]). To solve the local access network expansion model, we develop a decomposition method combining Lagrangian relaxation with a dynamic programming algorithm that exploits the problem's special tree structure and routing restrictions. Since the basic problem formulation does not provide satisfactory lower bounds, we identify several classes of valid inequalities that strengthen the model's linear programming relaxation. Unlike other cutting plane methods that use general purpose linear programming codes to solve the enhanced

formulations (e.g., Hoffman and Padberg [1985]), we modify the dynamic programming algorithm to directly incorporate the valid inequalities. Our computational tests using representative data (obtained from industry) demonstrate that the method generates good upper and lower bounds. Thus, this paper not only develops an effective solution method for the important practical problem of local access network design, but also adds to the growing literature demonstrating the usefulness of polyhedral methods for solving difficult, large-scale optimization problems.

The rest of this paper is organized as follows: Section 2 presents a formal definition of the local access network planning problem, reviews our modeling assumptions, and describes a basic mixed-integer programming formulation. Section 3 describes an efficient dynamic programming algorithm to solve the uncapacitated version of the local access network planning problem, develops a Lagrangian relaxation scheme that uses the dynamic program to solve an uncapacitated subproblem, and outlines a Lagrangian-based heuristic procedure. In Section 4 we describe two algorithmic enhancements—a problem preprocessing procedure to eliminate variables, and a coefficient reduction method to strengthen the problem formulation. Section 5 presents three classes of valid inequalities, and shows how to modify the dynamic program to incorporate them. Section 6 describes our implementation, and presents computational results for three networks provided to us by a major telephone company. We illustrate how the valid inequalities dramatically improve the lower bounds (by about 80%) relative to the basic model, and we study the robustness of the method to changes in demand and cost parameters. Our results show that the combination of Lagrangian relaxation, dynamic programming, and polyhedral methods permits us, using a personal computer, to efficiently find solutions that are within 1.2 to 7.0% of optimality. Section 7 identifies directions for further work.

2. The Basic Local Access Network Expansion Model

2.1 Problem description

The local access network (also called the feeder loop, central office network, outside plant, or customer access network) connects customer nodes (control or distribution points, in telecommunication parlance) to the switching center (also called the central office). Each customer node is a collection point for individual customers (possibly hundreds) connected via a subsidiary distribution network. Balakrishnan et al. [1991] and Jack et al. [1992] describe the technologies and characteristics of local access networks in greater detail. Most current local access networks have a tree structure, rooted at the switching center (Shulman and Vachani [1990]). Edges of the tree correspond to physical sections of underground or overhead cables.

All communications to and from each customer node flow through the assigned switching center. Each node has a **demand**, measured by the required number of **circuits** from that node to the switching center. In conventional copper networks, each circuit requires a dedicated twisted copper pair which we will call a **cable**. A node's demand depends on the number and type (e.g., residential or commercial) of individual customers connected to it. The local access network can satisfy this demand in two ways: either provide a dedicated cable (from the customer node to the switching center) for each required circuit, or route the circuits through a traffic compression device called a **concentrator**. Concentrators are electronic devices that combine incoming signals (e.g., analog signals) on several lines into a single composite signal (e.g., high frequency digital or optical signal) that requires only one outgoing line. In practice, a variety of devices such as electronic multiplexers, remote switches, and fiber optic terminals can perform traffic compression (see Balakrishnan et al. [1991]). We collectively refer to all these different technologies as concentrators. Our planning model distinguishes between different technologies through their installation and operating cost functions.

As customer demand increases (for example, due to new construction, customer movement, or new services), the existing cables and concentrators can no longer accommodate the required number of circuits from each node. In the expansion planning problem, we wish to locate new concentrators, selectively expand cable capacities, and reroute traffic from customer nodes via concentrators in order to satisfy the projected demand using the minimum possible total network expansion cost. By routing traffic through concentrators, we reduce the downstream (or central office side) cable requirements. This tradeoff between installing concentrators and expanding cable capacities is central to the local access network expansion problem. We next introduce some notation and formally describe the expansion planning model and its assumptions.

Notation and problem parameters

Let **T** denote the given (undirected) rooted tree over which the local access network expansion problem is defined. The nodes of this network represent customer nodes and/or potential concentrator locations, and its edges correspond to cable sections. We index the set of nodes **N** from **0** to **n**, with the **root node 0** representing the switching center. Let $\mathbf{d_i}$ denote the projected **demand** at each customer node i. The **capacity** $\mathbf{B_{ij}}$ of edge (i,j) is the number of existing cables in the section connecting nodes i and j. For simplicity, all our subsequent discussions assume that the *existing network does not contain any concentrators*; however, our method extends easily to problems with existing concentrators.

Let $\mathbf{P_{ij}}$ denote the (unique) **path** in the tree connecting nodes i and j. To provide one circuit from node i to the switching center, we must reserve one cable on each edge of the path P_{i0}. Since the existing network does not have adequate capacity to

meet the projected demand, one or more edges of the current network must have projected *exhaust*, i.e., the number of available cables on that edge is less than the total demand for all nodes communicating through that edge to the switching center.

Cost structure

Our model minimizes the total cost of installing concentrators and adding cables to meet the projected demand. The model and solution methodology can also incorporate additional node-to-concentrator connection costs (e.g., to disconnect the current circuit and reroute it through a concentrator at a different location) which we ignore for simplicity. *Cable expansion* costs vary by edge (depending on the length and location of the cable section), and has both a fixed and variable component. On each edge (i,j), we incur a **fixed cable cost G_{ij}** (e.g., to install additional ducts or poles) and a **variable cable cost e_{ij}** that might represent, for instance, investment in cables or maintenance expenses. *Concentrator* costs also have location-dependent fixed and variable components. The **fixed concentrator cost F_j** models land acquisition and infrastructure investments at node j, while the **variable concentrator cost c_j** reflects the purchase price and operating expenses of concentrator modules. As we note later, the concentrator cost also includes the cost of the required high-speed concentrator-to-switching center connection.

Our solution procedure also applies when concentrator and cable expansion costs have the more general piecewise-linear, concave structure shown in Figure 1; each segment in this function might correspond, for instance, to a different concentrator technology or transmission medium. The true cost of concentrators (or cable expansion) is a step function of the required capacity since concentrator modules are available only in discrete units; however, our analysis of actual cost estimates and information provided by our industrial collaborator suggest that a piecewise-linear, concave function can adequately approximate this cost especially for long-term planning purposes (due to rapid technological changes, predicting the costs exactly for, say, a 5-year time horizon is often very difficult). Although we have implemented and tested the solution method for problems with concave concentrator costs, for expositional ease in describing the model formulation and solution algorithm, we will assume the simpler fixed plus variable cost structure for both cable expansion and concentrator location. (We do, however, indicate how the approach will handle concave costs.)

2.2 Modeling assumptions

To reduce the complexity of managing and maintaining the local access network, planners often impose several restrictions on the permissible expansion options and routing patterns. Discussions with planners in industry suggest that the following four modeling assumptions reflect or adequately approximate current practice.

Shulman and Vachani [1990] and Jack et al. [1992] use similar assumptions in their successful decision support system for local access network planning.

Assumption A1: Single-level concentration
Traffic originating at any node of the network is concentrated at most once before reaching the switching center.

Assumption A2: Non-bifurcated routing
A single concentrator (or the switching center) processes the entire demand of each customer node.

Assumption A3: Contiguity restriction
Every concentrator serves a contiguous region surrounding it, i.e., if a concentrator at node j serves node i, then this concentrator also serves all other nodes (including node j) on the connecting path P_{ij}.

Assumption A4: Transmission cost for concentrated traffic
Concentrated traffic (flowing from each concentrator to the switching center) either consumes a negligible amount of existing cable capacity, or uses a dedicated umbilical connection (also called a remote-to-host connection) whose cost depends only on the location and throughput of the concentrator. In the latter case, we can incorporate the cost of the umbilical connection in the concentrator cost.

Assumption A1 reflects current state-of-the-art in local access network design. Introducing multiple levels of concentration within the local access network is often uneconomical given the current costs of cables and concentrators. Assumptions A2 and A3 reflect operational convenience. For example, maintaining and repairing networks with multiple routes from each customer node or non-contiguous concentrator service regions can be burdensome. Assumption A4 greatly simplifies the model and improves solution effectiveness, while introducing only minor distortions in total cost or actual capacity usage. For instance, one digital copper pair, also called a T1 span, can accommodate up to 96 voice channels (see, for example, Jack et al. [1992]); thus a customer node that previously required, say, 1000 copper pairs for analog transmission, requires only 11 T1 spans to transmit compressed signals from that node. If the "concentrator" includes a fiber optic terminal, the fiber cable connecting the terminal to the switching center might replace an existing copper cable.

Since the local access network has a tree structure, assumptions A1 and A2 together imply that assigning a concentrator (or switching center) to each node i completely specifies the routing decisions in the network. We say that **node i homes on node j** if a concentrator located at node j processes node i's traffic. In this case, the traffic from node i is routed on d_i cables along the unique path P_{ij} to node j, where it

is concentrated and transmitted via the remote-to-host connection from node j to the switching center. Any node whose traffic is not concentrated is said to home on the switching center; equivalently, we assume that the switching center always has a concentrator. We permit ***backfeed*** or flow away from the switching center, i.e., a concentrator at node j can also serve downstream nodes that are closer than node j to the switching center. The contiguity assumption (A3) reduces the concentrator selection and assignment problem to one of decomposing (and covering) the tree into subtrees, and selecting one concentrator location within each subtree to serve its traffic requirements. We use this observation to efficiently solve an uncapacitated subproblem using dynamic programming.

To illustrate these concepts, consider the local access network shown in Figure 2. The dashed edges in this network are sections with projected exhaust. For instance, the current capacity of 1675 units on edge (26, 34) is less than the total demand (2203 units) of nodes 34 and 41. Hence, we must either expand this edge by 528 units in order to home nodes 34 and 41 on a downstream concentrator (or the switching center), or install a concentrator at either of these nodes. For instance, a concentrator at node 34 can serve both nodes 34 and 41. Since we permit backfeed, we can also home the downstream nodes, say nodes 26, 18, and 12, on this concentrator. Notice that, if we use this homing strategy, we need not expand edges (26, 34), (18, 26), and (12, 18) since their existing capacities can accommodate the required backflow. Also, if node 7 homes on the concentrator at node 34, the contiguity property specifies that node 13 (and node 19) cannot home on the switching center; this node must either home on node 34 or on a concentrator located at nodes 13 or 19.

2.3 Basic Integer Programming Formulation

In the local network expansion planning problem we consider, we are given the projected demand at each customer node, the existing cable capacity in each section, and the costs for adding cables and installing concentrators at each location. We need to decide:

- where to locate concentrators, and with what capacity;
- which edges (cable sections) to expand, and by how much;
- how to route the traffic from each node to the switching center; and,
- in a more general model, which concentrator technology to use at each node.

Our formulation uses the following decision variables:

Assignment variable	x_{ij}	=	1 if node i homes on node j, 0 otherwise;
Concentrator location variable	y_j	=	1 if we install a concentrator at node j, 0 otherwise;

Cable installation variable z_{ij} (z_{ji}) = 1 if we expand cable capacity from node i to node j (node j to node i), 0 otherwise; and,

Cable expansion variable s_{ij} (s_{ji}) = number of cables added from node i to node j (j to i).

We use directed variables to model cable installation and expansion, i.e., although edges are undirected, we distinguish between expansion in the i-to-j direction and the j-to-i direction on each edge (i,j). Using directed cable addition variables increases the formulation size but strengthens the model's linear programming relaxation, thus improving our algorithm's performance. To emphasize the direction of flow, we will consider two directed **arcs**, denoted as <i,j> and <j,i>, corresponding to each original undirected **edge** (i,j); both arcs have the same fixed and variable cable expansion costs as the original edge. We also redefine P_{ij} as the **directed path** from node i to node j in the tree. We assume, for convenience, that each customer node can home on any other node in the network. In practice, we might prohibit certain node-to-concentrator assignments (e.g., due to proximity restrictions limiting the maximum distance a concentrator can serve in order to ensure good transmission quality), in which case we can eliminate the corresponding assignment variables.

The Local Access Network Expansion Planning Problem has the following basic mixed-integer programming formulation:

Network Expansion Planning Model [LAN1]

$$\text{minimize} \quad \sum_{j \in N} F_j y_j + \sum_{i \in N} \sum_{j \in N} (d_i c_j) x_{ij} + \sum_{<i,j> \in T} G_{ij} z_{ij} + \sum_{<i,j> \in T} e_{ij} s_{ij} \qquad (2.1)$$

subject to

Assignment constraints:

$$\sum_{j \in N} x_{ij} = 1 \qquad \text{all } i \in N, \qquad (2.2)$$

Concentrator location constraints:

$$y_j = x_{jj} \qquad \text{all } j \in N, \qquad (2.3)$$

Contiguity restrictions:

$$x_{ij} \leq x_{k_{ij},j} \qquad \text{all } i,j \in N, \qquad (2.4)$$

Cable capacity constraints:

$$\sum_{k,l \in OD_{ij}} d_k x_{kl} \leq B_{ij} + s_{ij} + s_{ji} \qquad \text{all } (i,j) \in T, \qquad (2.5)$$

Cable installation-forcing constraints:

$$s_{ij} \leq M_{ij} z_{ij} \qquad \text{all } \langle i,j \rangle \in T, \qquad (2.6)$$

Arc orientation constraints:

$$z_{ij} + z_{ji} \leq 1 \qquad \text{all } (i,j) \in T, \text{ and} \qquad (2.7)$$

Integrality/Nonnegativity constraints:

$$y_j,\ x_{ij},\ z_{ij},\ z_{ji} = 0 \text{ or } 1 \qquad \text{all } j \in N,\ (i,j) \in T, \text{ and} \qquad (2.8)$$

$$s_{ij},\ s_{ji} \geq 0 \qquad \text{all } (i,j) \in T. \qquad (2.9)$$

In this formulation,

k_{ij}	is the node adjacent to node i on path P_{ij},
OD_{ij}	is the set of all node pairs k,l whose connecting path P_{kl} contains edge (i,j), and
M_{ij} (M_{ji})	is an upper bound on the maximum required cable expansion on edge (i,j) in the i-to-j (j-to-i) direction.

The objective function (2.1) minimizes the sum of the fixed and variable concentrator costs, and the cable installation and expansion costs. Constraints (2.2) ensure that each node i is assigned to exactly one concentrator (possibly at the switching center, in which case $x_{i0} = 1$). Equation (2.3) specifies that node i contains a concentrator ($y_j = 1$) if and only if this node homes on itself (i.e., $x_{jj} = 1$). Constraints (2.4) models the contiguity restriction, i.e., if node i homes on node j, then node i's immediate neighbor k_{ij} on the connecting path P_{ij} must also home on j. The left-hand side of the cable capacity constraint (2.5) expresses the total flow on edge (i,j) in terms of the node-to-concentrator assignments that use this edge; we must add cables if this flow exceeds the available capacity B_{ij}. If we add cables on arc $\langle i,j \rangle$ (i.e., if $s_{ij} > 0$), constraint (2.6) forces the cable installation variable z_{ij} to assume a value of 1, thus absorbing the fixed cable expansion cost G_{ij} in the objective function. Constraint (2.7) permits cable expansion on edge (i,j) in either the i-to-j or the j-to-i direction, but not both.

The parameter $\mathbf{M_{ij}}$ in the right-hand side of the forcing constraint (2.6a), which we call the **cable expansion bound,** represents the maximum number of additional cables that any optimal solution can possibly install on arc $\langle i,j \rangle$. For instance, if D_{ij} is the total demand for all nodes k whose path P_{kj} contains arc $\langle i,j \rangle$, we can set $M_{ij} = \text{Max } \{ 0, D_{ij} - B_{ij} \}$. In Section 4.2, we show how to strengthen the formulation by using tighter values for M_{ij}.

Let us briefly indicate how we might incorporate piecewise-linear, concave cost functions (Figure 1) instead of the simple fixed plus linear cost structure. Suppose the concentrator cost function consists of M linear segments (corresponding to M different concentrator types or technologies), with increasing fixed costs F_{jm} and decreasing variable costs c_{jm} as a function of the technology type m = 1,2,...M. To capture these costs, we replace y_j and x_{ij} with disaggregate concentrator location and assignment variables y_{jm} and x_{ijm} for each segment m = 1,2,...,M. The binary variable y_{jm} is 1 if the solution installs a type m concentrator at node j, and is 0 otherwise; similarly, x_{ijm} equals 1 if a type m concentrator at node j serves node i, and is 0 otherwise. We modify constraints (2.2) to (2.5) accordingly. Because the cost function is concave, we need not introduce explicit concentrator capacity constraints since the cost minimizing solution will automatically select the appropriate technology to process the required throughput at each concentrator location. We can similarly model piecewise-linear, concave cable expansion costs using disaggregate cable installation and expansion variables.

3. Decomposition Algorithm for the Basic Local Access Network Planning Model

Formulation [LAN1] is a large-scale mixed-integer program whose size increases quadratically with the number of nodes. Like many other network design problems, this problem is NP-complete (Balakrishnan et al. [1992]). However, the uncapacitated version of this problem without existing cable cable capacities is easy to solve using a polynomial-time dynamic programming algorithm. We, therefore, propose a Lagrangian relaxation approach that solves an uncapacitated subproblem to generate good upper and lower bounds. Our solution method consists of three components: (i) preprocessing and coefficient reduction, i.e., performing some prior analysis to reduce the problem size and strengthen the formulation; (ii) solving the Lagrangian subproblems to generate lower bounds on the optimal cost; and (iii) generating good heuristic solutions from the Lagrangian subproblem solutions. This section first discusses the dynamic programming method for solving the uncapacitated problem, and develops the Lagrangian-based lower bounding and heursitic procedures. In subsequent sections, we describe the preprocessing and coefficient reduction methods, and propose various additional formulation and algorithmic enhancements to improve the method's performance.

3.1 Solving the Uncapacitated Local Access Network Planning Problem

Given a tree network without existing cable capacities, and fixed and variable costs for installing concentrators and cables, the uncapacitated local access network planning (ULAN) problem seeks the concentrator locations, homing patterns, and cable expansion plan that meets projected demand at minimum total cost. To

distinguish the cost parameters for the uncapacitated problem from the original values, we let Φ_j and γ_j denote the fixed and variable concentrator costs at node j, and let Γ_{ij} and ε_{ij} denote the fixed and variable cable costs on arc <i,j>. Later we will indicate how, using Lagrange multipliers, we compute the values of these "uncapacitated" cable and concentrator cost parameters from the original costs.

3.1.1 Simplifying the uncapacitated problem

First, we simplify the uncapacitated problem by transforming all the fixed and variable cable and concentrator costs into equivalent node-to-concentrator **assignment costs**. This transformation exploits the non-bifurcated routing and contiguity properties, and is valid only when the network does not contain any existing cable capacities. The assignment cost a_{ij} represents the "incremental" cost of assigning node i to a concentrator at node j. We compute its value as follows:

$$\begin{aligned} a_{ij} &= \Phi_j + d_i \gamma_j && \text{if } i = j, \text{ and} \\ &= \Gamma_{ik_{ij}} + d_i \{ \sum_{<k,l> \in P_{ij}} \varepsilon_{kl} \} + d_i \gamma_j && \text{if } i \neq j. \end{aligned} \tag{3.1}$$

When i = j, equation (3.1) sets the self-assignment cost a_{jj} equal to the total cost of installing a concentrator at node j and serving this node's demand. When $i \neq j$, the assignment cost consists of three components: (a) the fixed cable cost on the arc $<i,k_{ij}>$ joining node i to its adjacent node k_{ij} on path P_{ij}; (b) the total variable cable cost on path P_{ij} to create d_i circuits from node i to node j; and (c) the total variable concentrator cost at node j for serving node i's demand. Equation (3.1) has the following rationale. Since the network does not initially contain any cables or concentrators, the assignment cost a_{ij} must include the variable cable and concentrator costs to transmit and concentrate node i's demand at node j. If concentrator costs are positive, the contiguity property implies that the optimal solution installs a concentrator at node j if and only if node j homes on itself. We, therefore, include the fixed concentrator cost in the self-assignment cost a_{jj}. Similarly, the optimal solution expands arc $<i,k_{ij}>$ if and only if node i homes on node j, $i \neq j$. Hence, we include this arc's fixed cost in the i-to-j assignment cost a_{ij}. By contiguity, if node i homes on node j, then every intermediate node on the path P_{ij} must also home on node j. Therefore, when we add the assignment costs for all nodes on path P_{ij}, we capture the total fixed cable costs for all arcs on this path. These intuitive arguments justify the following claim:

> For the uncapacitated local access network planning problem, the true total concentrator and cable costs of the optimal expansion plan equals the sum of the assignment costs incurred by that plan.

Thus, the ULAN problem has the following simple *assignment tree packing* (ATP) formulation containing only the binary assignment variables x_{ij}:

$$\text{minimize the total assignment cost} = \sum_{i \in N} \sum_{j \in N} a_{ij} x_{ij}$$

subject to

assignment constraints (2.2), contiguity constraints (2.4), and $x_{ij} \in \{0,1\}$ for all $i,j \in N$.

3.1.2 Dynamic Programming Algorithm

We can solve the ATP formulation using Barany, Edmonds, and Wolsey's [1986] $O(n^2)$ dynamic programming algorithm. This algorithm is related to Kariv and Hakimi's [1979] algorithm for solving the p-median problem on a tree. Barany et al. [1986] have also shown that the linear programming relaxation of the ATP formulation has integer extreme points.

To describe the dynamic program, let us introduce some notation and conventions. The *level* of a node i is the number of edges lying on path P_{i0}. Thus, the root node (node 0) has level 0, its immediate successors have level 1, and so on. For convenience, we index the nodes in increasing order of their levels. For any node i ($i \neq 0$) in the tree, let p_i denote its *predecessor*, and S_i the set of all its *immediate successors*. Let *T(i)* denote the *subtree rooted at node i* formed when we delete edge (i,p_i) from tree T.

Starting at the bottom of the tree, the dynamic programming procedure recursively calculates, for each node i, the optimal *total assignment cost TC(i) of serving all nodes in subtree T(i) using only homing nodes (concentrators) located within this subtree.* This tree cost TC(i) represents the optimal total network expansion cost if T(i) is a stand-alone tree. Hence, TC(0) is the optimal cost of the ULAN problem.

To calculate TC(i), we must first determine where node i should home within its subtree T(i). For any node $j \in T$ (note that j might lie outside T(i)), let *HC(i,j)* (HC stands for homing cost) denote the *total cost of covering all nodes in subtree T(i), assuming node i homes on node j*. Then,

$$TC(i) = \underset{j \in T(i)}{\text{minimum}} HC(i,j). \qquad (3.2)$$

The homing cost HC(i,j) consists of the i-to-j assignment cost a_{ij} plus the homing costs of the subtrees rooted at node i's successors. By contiguity, if node i homes on node j, then each successor $u \in S_i$ must either home on node j, or home within its subtree T(u). Therefore, the following recursive equations permit us to compute HC(i,j) for intermediate nodes i:

$$HC(i,j) = a_{ij} + \sum_{u \in S_i} \min\{HC(u,j), TC(u)\} \quad \text{if } j=i \text{ or } j \notin T(i), \text{ and} \qquad (3.3a)$$

$$HC(i,j) = a_{ij} + HC(v,j) + \sum_{u \in S_i \setminus \{v\}} \min\{HC(u,j), TC(u)\} \quad \text{if } j \in T(v),\ v \in S_i. \qquad (3.3b)$$

Equation (3.3b) applies when node i homes on an internal node j in rooted subtree T(v) for some successor node $v \in S_i$ (by contiguity, node v must also home on node j). Note that if node i is a leaf node, $S_i = \phi$, and therefore $HC(i,j) = a_{ij}$.

To ensure that the required quantities in the right-hand side of equations (3.3) are available when needed, we compute the values of HC(i,j) for nodes i in a bottom-to-top sequence. The ULAN dynamic programming algorithm consists of (n+1) stages. At stage i, for i = n, (n-1), ..., 0, we first compute HC(i,j) for every node $j \in T$. We then apply equation (3.2) to calculate TC(i), the optimal cost of serving all nodes of subtree T(i) using only internal concentrators. The final value TC(0) computed at stage 0 gives the optimal value of the ULAN problem. The usual dynamic programming backtracking procedure gives the optimal concentrator location and node assignment strategy. The following formal description of the dynamic programming algorithm summarizes its steps.

DP Algorithm for the Basic ULAN model: [DP1]

For i = n,n-1,...,0,

if i is a leaf node,

$HC(i,j) \leftarrow a_{ij}$ for all $j \in T$, and

$TC(i) \leftarrow HC(i,i) = a_{ii}$;

else,

for all $j \in T \backslash T(i)$, compute HC(i,j) using equation (3.3a);

for all $v \in S_i$ and all $j \in T(v)$, compute HC(i,j) using equation (3.3b);

compute HC(i,i) using equation (3.3a);

Set $TC(i) \leftarrow \underset{j \in T(i)}{\text{Minimum}} HC(i,j)$;

next i;

The following argument shows that this dynamic program has complexity $O(n^2)$. For each node u, which has a unique predecessor i, we compare HC(u,j) and TC(u) (in equations (3.3a) or (3.3b)) for every homing node $j \in T$. Therefore, the algorithm requires O(n) computations for each node u, requiring a total of $O(n^2)$ computations. With minor changes (by defining separate homing costs H(i,j,m) for each piecewise-linear segment or concentrator technology m at node j), this solution method can incorporate piecewise-linear, concave concentrator costs.

3.2 The Lagrangian Relaxation Scheme

To solve the original (capacitated) local access network planning problem, we use a Lagrangian relaxation scheme (see, for instance, Fisher [1981]) that dualizes the cable capacity constraints (2.5) of formulation [LAN1] using *Lagrange multipliers* μ_{ij}

for all edges (i,j) $\in$ T (for directed arcs <i,j>, $\mu_{ij} = \mu_{ji}$ by convention). The resulting Lagrangian problem is:

$$\text{minimize} \quad \sum_{i \in N} \sum_{j \in N} d_i \{c_j + \sum_{<k,l> \in P_{ij}} \mu_{kl} \} x_{ij} + \sum_{j \in N} F_j y_j + \sum_{<i,j> \in T} G_{ij} z_{ij} + \sum_{<i,j> \in T} (e_{ij} - \mu_{ij}) s_{ij} - \sum_{(i,j) \in T} \mu_{ij} B_{ij} \qquad (3.4)$$

subject to constraints (2.2) – (2.4) and (2.6) – (2.9).

This problem decomposes into two subproblems: an ULAN subproblem, and a cable expansion subproblem.

3.2.1 The ULAN subproblem

The *uncapacitated network expansion subproblem,* which we denote as ULAN1(μ), contains the x and y variables. Using our notation of Section 3.1.1, the ULAN1(μ) subproblem has the following equivalent "uncapacitated" cost parameters: $\Phi_j = F_j$, $\gamma_j = c_j$, $\Gamma_{kl} = 0$, and $\varepsilon_{kl} = \mu_{kl}$. Notice that this subproblem does not contain fixed cable costs; our later formulation enhancements will strengthen the Lagrangian relaxation by introducing arc fixed costs in the uncapacitated subproblem. For any given set of Lagrange multipliers $\{\mu_{kl}\}$, solving subproblem ULAN1(μ) using our dynamic program gives a set of concentrator locations, and node-to-concentrator assignments that satisfy the contiguity and non-bifurcated routing properties. We later use this subproblem solution to construct a feasible heuristic solution to the original problem.

3.2.2 The Cable Expansion Subproblem

The *cable expansion subproblem,* denoted [CES(μ)], determines the optimal values of the cable installation and expansion variables, z_{ij} and s_{ij}, for all arcs <i,j>:

[CES(μ)]

$$\text{minimize} \quad \sum_{<i,j> \in T} G_{ij} z_{ij} + \sum_{<i,j> \in T} (e_{ij} - \mu_{ij}) s_{ij} \qquad (3.5)$$

subject to

$$s_{ij} \le M_{ij} z_{ij} \quad \text{all } <i,j> \in T, \qquad (3.6)$$

$$z_{ij} + z_{ji} \le 1 \quad \text{all } (i,j) \in T, \text{ and} \qquad (3.7)$$

$$z_{ij} = 0 \text{ or } 1, \; s_{ij} \ge 0 \quad \text{all } <i,j> \in T. \qquad (3.8)$$

This subproblem decomposes by edge, and is easy to solve. For each arc <i,j>, we first express the optimal value of s_{ij} in terms of z_{ij} as follows:

$$s_{ij} = 0 \quad \text{if } (e_{ij} - \mu_{ij}) \ge 0, \text{ and} \qquad (3.9a)$$

$$s_{ij} = M_{ij} z_{ij} \qquad \text{if } (e_{ij} - \mu_{ij}) < 0. \tag{3.9b}$$

Substituting for s_{ij} in (3.5) gives the following cost coefficients for z_{ij}:

$$G_{ij}(\mu) \equiv G_{ij} + M_{ij} \min \{e_{ij} - \mu_{ij}, 0\}. \tag{3.10}$$

If $G_{ij}(\mu)$ and $G_{ji}(\mu)$ are both nonnegative, we set $z_{ij} = z_{ji} = 0$. Otherwise, we set $z_{ij} = 1$ and $z_{ji} = 0$ if $G_{ij}(\mu) \leq G_{ji}(\mu)$, and $z_{ij} = 0$ and $z_{ji} = 1$ if $G_{ji}(\mu) < G_{ij}(\mu)$. Again, this solution procedure extends easily to problems with piecewise-linear, concave concentrator costs.

For any nonnegative Lagrange multiplier vector μ, the sum of the optimal values of the ULAN and cable expansion subproblems minus the term $\sum_{(i,j) \in T} \mu_{ij} B_{ij}$ gives a lower bound on the optimal cost of [LAN1]. We use subgradient optimization (see, for instance, Held, Wolfe, and Crowder [1974] or Fisher [1981]) to heuristically adjust the Lagrange multipliers to maximize the Lagrangian lower bound. Since the linear programming relaxations for both of our Lagrangian subproblems have integer optimal solutions (Aghezzaf and Wolsey [1990] have shown this property for the ULAN problem with piecewise-linear, concave costs), the best possible Lagrangian lower bound cannot exceed the optimal value of the linear programming relaxation of formulation [LAN1]. We next describe a method to obtain upper bounds on the optimal value.

3.3 Lagrangian-based heuristic procedure

Our Lagrangian-based heuristic procedure first constructs a feasible starting solution using the optimal values of subproblem ULAN1(μ), and then applies a local improvement procedure to further reduce the cost of this starting solution. We construct the starting solution by "completing" the contiguous, non-bifurcated node-to-concentrator assignments chosen by subproblem ULAN1(μ), i.e., we compute the actual cable expansion required to accommodate the node-to-concentrator flows, and compute the total concentrator and cable cost of this expansion plan. We then apply a myopic improvement strategy called the *Greedy Reassignment Heuristic* that iteratively reassigns nodes to concentrators, one at a time, without violating the contiguity condition. To preserve contiguity, we need to only consider reassigning every node i to each of the ($|S_i|+1$) homing nodes of its neighbors. At each iteration, the greedy heuristic: (i) evaluates the cost impact of all feasible changes in node-to-concentrator assignments; and, (ii) performs the reassignment that gives the greatest reduction in total cost. If all feasible reassignments increase total cost, the local improvement procedure terminates.

To reduce computational time, instead of improving the Lagrangian-based starting solution after every subgradient iteration, our implementation applies the greedy method only intermittently (e.g., when the current Lagrangian starting

solution has lower cost than the previous best starting solution). We also use the greedy heuristic to generate an **initial upper bound**, before performing the subgradient procedure. We consider two different starting solutions for initial improvement—a *centralized* solution that homes all demand nodes on the root node (i.e., this solution employs only cable expansion to satisfy projected demand), and a *distributed* solution that locates a concentrator at each node. The better of the two improved solutions provides the initial upper bound.

4. Modeling and Algorithmic Enhancements I: Variable Elimination and Coefficient Reduction

Our preliminary computational experience (summarized in Section 6.2) with the Lagrangian relaxation algorithm for the basic model [LAN1] suggested that, while the heuristic method generates very good solutions, the Lagrangian lower bounds are weak. To improve the lower bounds, we developed various modeling and algorithmic enhancements. This section describes two types of improvements: problem preprocessing to eliminate certain assignment variables, and reducing the values of the cable expansion bounds M_{ij} in order to tighten the forcing constraints (2.6) in formulation [LAN1]. Section 5 describes new inequalities that further strengthen the Lagrangian relaxation.

4.1 Variable Elimination by Problem Preprocessing

To reduce the size of problem [LAN1], we perform a tradeoff analysis to identify suboptimal node-to-concentrator assignments a priori. Eliminating the corresponding assignment variables x_{ij} from the problem formulation not only reduces the problem size and computational effort, but might also improve the lower and upper bounds.

For each node pair i,j, our preprocessing method determines if node i can home on node j in an optimal expansion plan by comparing a *lower bound* L_{ij} on the incremental cost of assigning node i to node j to an upper bound U_{ii} on the cost of locating a concentrator at node i (and homing node i on this concentrator). If $L_{ij} > U_{ii}$, the i-to-j assignment is provably suboptimal, and we can eliminate the assignment variable x_{ij} from formulation [LAN1].

The lower bound L_{ij} on the incremental cost of assigning node i to node j consists of two components: an incremental *cable expansion* cost, and an incremental *concentrator* cost. To calculate the incremental cable expansion cost, consider any arc <k,*l*> on the path P_{ij} connecting node i to node j. By contiguity, if node i homes on node j, the total demand, say, D_{ik} of all nodes on the path P_{ik} (nodes i and k inclusive) must flow through arc <k,*l*>. Let $\phi_{kl} = \text{Max}\{ [D_{ik}-B_{kl}], 0\}$ denote the excess

flow on arc <k,*l*>. Node i contributes Min{ϕ_{kl}, d_i} to this excess flow. Hence, δ_{kl} = e_{kl} Min{ϕ_{kl}, d_i} represents the incremental cable cost if node i homes on node j (note that we do not include the fixed cable cost G_{kl} in this incremental cost). Adding the incremental costs δ_{kl} for all arcs <k,*l*> of path P_{ij} gives the cable expansion cost component of the lower bound L_{ij}. For the incremental concentrator cost, we use the variable concentrator cost $c_j d_i$ incurred at node j to serve node i's demand. Thus, the lower bound L_{ij} is:

$$L_{ij} = \sum_{<k,l> \in T \in P_{ij}} \delta_{kl} + c_j d_i \qquad \text{for all } i,j \in N. \tag{4.1}$$

If node i is a leaf node of T, and if the current capacity of the incident arc (i,k_{ij}) is less than node i's demand, we can improve the lower bound L_{ij} by adding the fixed cost $G_{i,k_{ij}}$ to the right-hand side of (4.1).

The upper bound U_{ii} on the incremental cost when node i homes on itself is the total concentrator cost to process node i's demand, i.e.,

$$U_{ii} = F_i + c_i d_i \qquad \text{for all } i \in N. \tag{4.2}$$

If $L_{ij} > U_{ii}$, the i-to-j assignment is provably suboptimal. For, suppose an optimal expansion plan assigns node i to a concentrator at node j. Let N(i,j) be the subset of nodes (including node i) that currently home on node j via node i. For every node $k \in N(i,j)$, canceling the k-to-j assignment saves at least L_{ij}/d_i per unit demand, while reassigning node k to a new concentrator at node i incurs a cost of at most U_{ii}/d_i per unit demand (due to concavity of concentrator costs). Therefore, if $L_{ij} > U_{ii}$, we can improve the current solution by installing a new concentrator at node i, and reassigning all nodes in N(i,j) to this concentrator (all the nodes on path P_{ij} except node i continue to home on the concentrator at node j), contradicting the optimality of the given solution.

This preprocessing technique extends easily to piecewise-linear, concave cost functions (to compute L_{ij} we use the lowest variable cable cost e_{klm} on each arc <k,*l*> $\in P_{ij}$, and the lowest variable concentrator cost c_{jm} at node j among all available technologies m). The preprocessing method not only reduces the number of variables and constraints in the integer programming formulation but also strengthens it by decreasing the maximum possible flows (and hence the cable expansion bounds M_{ij}) on certain arcs.

4.2 Tightening the Cable Forcing Constraints by Coefficient Reduction

To improve the relaxation lower bounds of formulation [LAN1], we first tighten the cable installation forcing constraints (2.6) by reducing the cable expansion bounds M_{ij}. Recall that M_{ij} represents the largest possible value of the cable expansion variable s_{ij} in any optimal solution. In Section 2.3, we computed M_{ij} as the

difference between the total demand that can enter arc <i,j> and its existing capacity. We refer to this value as the *demand-based cable expansion bound*, and denote it as M_{ij}^d. Since this demand-based bound represents the worst-case cable expansion requirements in any feasible solution, its value can be much larger than the actual flow routed on arc <i,j> in an optimal solution. Consequently, the cable installation variables z_{ij} often take small fractional values in the optimal linear programming (or Lagrangian) solution (with nonnegative costs, formulation [LAN1] has an optimal LP solution with $z_{ij} = s_{ij}/M_{ij}$).

To reduce M_{ij}, we compare the cable expansion cost on arc <i,j> with the concentrator cost at node i to determine the breakeven flow value above which locating a concentrator at node i is cheaper than routing flow on arc <i,j>. If an expansion plan routes $f_{ij} > B_{ij}$ units of traffic from i to j, it incurs an expansion cost of at least $CE_{ij}(f_{ij}) = \{G_{ij} + e_{ij}*(f_{ij}-B_{ij})\} + f_{ij}\, c_{min}$, where c_{min} is the smallest variable concentrator cost taken over all nodes at or beyond node j. Consider the alternate solution obtained by installing a concentrator at node i, and rehoming all the traffic that previously flowed through arc <i,j> on this concentrator. This solution incurs a concentrator cost of $CC_i(f_{ij}) = \{F_i + c_i*f_{ij}\}$. Clearly, if $CE_{ij}(f_{ij})$ exceeds $CC_i(f_{ij})$, then installing a concentrator at node i improves the given solution.

Let U_{ij} denote the flow value at which the cost functions $CE_{ij}(f)$ and $CC_i(f)$ intersect (assuming $G_{ij} < F_j$ and $e_{ij} > c_j$) as shown in Figure 3. Since routing more than U_{ij} units of flow on arc <i,j> is suboptimal, we can limit the value of flow on arc <i,j> to U_{ij}, giving us a *cost-based cable expansion bound* of $M_{ij}^c = (U_{ij} - B_{ij})$. We then set the right-hand side coefficient M_{ij} in the forcing constraint (2.6) equal to min $\{M_{ij}^d, M_{ij}^c\}$. Again, these cost-based upper limits extend to the case of piecewise-linear, concave costs.

5. Modeling and Algorithmic Enhancements II: Incorporating Valid Inequalities

To further improve the Lagrangian lower bounds, we add certain valid inequalities or cuts to the original problem formulation, and modify our solution method to incorporate the new constraints. These cuts reduce the feasible region for the Lagrangian (and linear programming) relaxation without eliminating the optimal integer solution, thus improving the lower bound. For a review of the underlying ideas and successful applications of this polyhedral combinatorics solution approach, see Hoffman and Padberg [1985] or Nemhauser and Wolsey [1988].

Balakrishnan et al. [1992] have identified several classes of valid inequalities for the local access network expansion problem, and showed that, under certain conditions, these inequalities are facets of the integer programming polytope. In this paper, we focus on a subset of those valid inequalities that are easy to incorporate in our dynamic programming algorithm for the Lagrangian subproblem. These constraints relate the assignment variables x_{ij} and the cable installation and expansion variables (z_{ij} and s_{ij}); adding them to the problem formulation results in a single, comprehensive Lagrangian subproblem (instead of our previous two subproblems) that simultaneously determines homing assignments, concentrator locations, and cable additions.

Sections 5.1 to 5.3 motivate and describe the three classes of valid inequalities that we implemented. Section 5.4 describes requisite modifications to the dynamic programming method needed to accommodate these three types of inequalities. Our computational experience indicates that these inequalities are very effective in reducing the gap between the Lagrangian lower and upper bounds.

Throughout this discussion, recall that T(i) is the subtree rooted at node i. Let D_i denote the total demand of all nodes in T(i); p_i is the predecessor of node i, and S_i is the set of node i's immediate successors.

5.1 Assignment-forcing Arc Installation Inequalities

Our first class of valid inequalities exploits the contiguity property to relate the assignment variables x_{ij} to the binary cable installation variables z_{ij}. Assuming positive arc expansion costs, any optimal local network expansion plan expands arc <i,k> only if node i homes on some node j via arc <i,k>. This observation motivates the following ***assignment-forcing arc installation inequalities***:

$$\sum_{j:\, <i,k> \in P_{ij}} x_{ij} \geq z_{ik} \qquad \text{for all arcs } <i,k>. \tag{5.1}$$

Balakrishnan et al. [1992] generalize these constraints to cutsets of the tree T other than a single arc <i,k>.

5.2 Bottleneck-Arc Installation and Expansion Inequalities

Our next class of inequalities relate the concentrator location decisions to the cable installation and expansion decisions. We refer to arc $<i,p_i>$ as a *bottleneck arc* and node i as a *bottleneck node* if the total demand D_i in subtree T(i) exceeds the arc's current capacity B_{ip_i}. Let I_B denote the set of *bottleneck nodes* in T. For every bottleneck node $i \in I_B$, any feasible expansion plan must either install at least one concentrator within subtree T(i) or expand arc $<i,p_i>$ (or both). Furthermore, if subtree T(i) does not contain any concentrators, the amount of capacity expansion on arc $<i,p_i>$ must be at least $(D_i - B_{ip_i})$. Hence, we can add the following valid ***bottleneck-arc installation*** and ***expansion inequalities*** to the problem formulation:

$$\sum_{k \in T(i)} y_k + z_{ip_i} \geq 1 \qquad \text{for all } i \in I_B, \text{ and} \qquad (5.2)$$

$$(D_i - B_{ip_i}) \{ \sum_{k \in T(i)} y_k \} + s_{ip_i} \geq D_i - B_{ip_i} \qquad \text{for all } i \in I_B . \qquad (5.3)$$

Note that, if arc $<i,p_i>$ is *not* a bottleneck arc, then $M^d_{ip_i} = 0$ and we can eliminate the arc installation and expansion variables z_{ip_i} and s_{ip_i} from the problem formulation. Balakrishnan et al. [1992] generalize constraints (5.2) to subtrees of T other than the rooted subtrees T(i) for i = 1,2,..., n.

5.3 Subtree-splitting Arc Installation and Expansion Inequalities

Given any feasible expansion plan, we say that subtree T(i) *completely homes* on an external node $j \notin T(i)$ if all nodes of T(i) home on node j in that expansion plan. On the other hand, T(i) *partially homes* on node $j \notin T(i)$ if node j serves only a subset of nodes in T(i) (including node i), and T(i) contains one or more concentrators that serve the remaining nodes in this subtree. If all nodes in T(i) home on concentrator(s) within T(i), we say that subtree T(i) is *self-sufficient*.

The bottleneck inequalities (5.2) and (5.3) apply when subtree T(i) completely homes on an external node j, in which case the total flow on arc $<i,p_i>$ *exactly equals* the total demand subtree D_i. On the other hand, when T(i) is self-sufficient, arc $<i,p_i>$ does not carry any flow; the assignment-forcing arc installation inequalities (5.1) prevent expanding arc $<i,p_i>$ in this case. We now consider additional valid inequalities for situations when subtree T(i) partially homes on a node $j \notin T(i)$. For the partial homing case, we do not know the exact flow on arc $<i,p_i>$, but we can compute *upper* and *lower bounds* on this flow. These bounds will vary depending on the homing patterns in the "successor" subtrees, i.e., the subtrees rooted at node i's successors. If, for a particular homing pattern, the upper bound (lower bound) is less (more) than arc $<i,p_i>$'s capacity B_{ip_i}, then any solution containing that homing pattern *must not* (*must*) expand arc $<i,p_i>$. We will refer to this class of inequalities as ***Subtree-splitting Arc Installation*** and ***Expansion Inequalities*** since they exploit the dynamic program's strategy of splitting each subtree into its constituent successor subtrees.

Let $\mathbf{dmin_h}$ denote the smallest leaf node demand in subtree T(h). If T(i) partially homes on an external node j, then at least one leaf node in T(i) must home on an internal concentrator (by contiguity); hence, the maximum possible flow on arc $<i,p_i>$ is $(D_i - dmin_i)$. Since node i homes outside T(i), the minimum flow on this arc is d_i. We might use these two bounds to decide if arc $<i,p_i>$ must necessarily be included (if $d_i > B_{ip_i}$) or excluded (if $D_i - dmin_i < B_{ip_i}$) when subtree T(i) partially homes on any external node. We can further sharpen the upper and lower bounds on arc $<i,p_i>$'s flow by separately considering different combinations of homing

patterns in the successor subtrees. In general, every successor subtree can: (i) completely home on node j, (ii) partially home on node j, or (iii) be self-sufficient. However, since subtree T(i) partially homes on node j, at least one of its successor subtrees must have a concentrator, i.e., we do not permit every successor subtree to completely home on node j. Thus, we consider $(3^{|S_i|} - 1)$ different successor homing combinations. Each combination or <u>case</u> is characterized by a partition Q = $\{S1_i, S2_i, S3_i\}$ of the successor node set S_i: $S1_i$, $S2_i$, and $S3_i$ correspond, respectively, to the subsets of node i's successors whose rooted subtrees completely home on node j, partially home on node j, or are self-sufficient in the chosen combination. Each case Q has associated upper and lower bounds on flow along arc $<i,p_i>$; we next show how to compute these bounds.

Calculating upper and lower bounds on flow on arc $<i,p_i>$:

Consider any successor node $u \in S_i$, and suppose $u \in S1_i$ for a given case Q, i.e., T(u) <u>completely homes</u> on external node j. Then, the <u>flow out of subtree</u> T(u) exactly equals the total demand D_u in that subtree. If $u \in S3_i$, then subtree T(u) is <u>self-sufficient,</u> and no flow emanates from T(u). Finally, if T(u) <u>partially homes</u> on an external node j (i.e., $u \in T2_i$) then, as we noted earlier ,the flow out of subtree T(u) cannot exceed $(D_u - dmin_u)$ units but must be at least d_u units.

Using these minimum and maximum outflows from the successor subtrees, we can compute upper and lower bounds on arc $<i,p_i>$'s flow as follows :

$$fmax_{ip_i}(Q) = d_i + \sum_{u \in S1_i} D_u + \sum_{u \in S2_i} \{D_u - dmin_u\}, \text{ and} \tag{5.4}$$

$$fmin_{ip_i}(Q) = d_i + \sum_{u \in S1_i} D_u + \sum_{u \in S2_i} d_u. \tag{5.5}$$

If $fmin_{ip_i}(Q)$ is greater than arc $<i,p_i>$'s capacity B_{ip_i}, then we can add a valid inequality specifying that arc $<i,p_i>$ must be **IN**, i.e., $z_{ip_i} = 1$ and $s_{ip_i} \geq (fmin_{ip_i}(Q) - B_{ip_i})$, whenever the solution selects the homing pattern Q. Similarly, if $fmax_{ip_i}(Q)$ is less than B_{ip_i}, we force arc $<i,p_i>$ to be **OUT**, i.e., $z_{p_i} = s_{p_i} = 0$, for homing pattern Q. Finally, if $fmin_{ip_i}(Q) \leq B_{ip_i} < fmax_{ip_i}(Q)$, then arc $<i,p_i>$ is **FREE**, i.e., we permit z_{ip_i} to be either 0 or 1, but $s_{ip_i} \leq (fmax_{ip_i}(Q_i) - B_{ip_i})$. We can formulate these logical restrictions as mathematical constraints in terms of the assignment, concentrator location, and cable addition variables. For instance, to force arc $<i,p_i>$ to be IN for a case Q (when $fmin_{ip_i}(Q) > B_{ip_i}$), we can add the following *subtree-splitting arc installation inequality*:

$$\sum_{u \in S1_i} \sum_{l \in T(u)} \sum_{k \in T(i)} x_{lk} + \sum_{u \in S2_i} \sum_{k \in T(i)} x_{uk} + z_{ip_i} \geq 1. \tag{5.6}$$

The first two terms in the left-hand side of (5.6) are both zero if the solution selects a homing pattern consistent with case Q, i.e., all nodes in subtrees T(u) for $u \in S1_i$ and all successors $u \in S2_i$ home on a node outside T(i), in which case constraint (5.7) forces cable installation on arc $<i,p_i>$. (We can tighten this subtree-splitting constraint by including in the first term only assignment variables x_{lk} corresponding to leaf nodes l in subtrees T(u) for $u \in S1_i$; by contiguity, all nodes of subtree T(u) must home on the external node j if all its leaf nodes home on node j.) The subtree-splitting constraint strengthens the original problem formulation, thus potentially improving the Lagrangian (and LP) lower bounds. Similarly, we can formulate a *subtree-splitting arc expansion* constraint to enforce cable expansion s_{ip_i} of at least $(fmin_{ip_i}(Q)-B_{ip_i})$ units for all IN arcs $<i,p_i>$ under case Q. We can also model the OUT and FREE restrictions on arc $<i,p_i>$. Since our dynamic programming approach can implicitly account for these inequalities, we do not require explicit mathematical representations such as (5.6).

To summarize, for every node i, we consider each of the $(3^{|S_i|}-1)$ cases corresponding to partial homing of subtree T(i). For every case Q, we compute the upper and lower bounds on flow on arc $<i,p_i>$ using equations (5.4) and (5.5). By comparing these bounds with the arc's capacity, we determine if we can restrict arc $<i,p_i>$ to be IN, OUT or FREE for homing pattern Q. In Section 5.4 we indicate how to incorporate this information in the dynamic program. Note that the subtree-splitting arc installation inequality (5.6) generalizes our previous bottleneck-arc installation inequality (5.2). Recall that the bottleneck inequalities apply when the subtree T(i) completely homes on an external node j. In this case, every successor of node i must also completely home on node j, i.e., the complete homing pattern corresponds to a case Q with $S1_i = S_i$, and $S2_i = S3_i = \phi$. Applying equations (5.4) and (5.5) to this case, we find $fmax_{ip_i}(Q) = fmin_{ip_i}(Q) = D_i$; hence, if $D_i > B_{ip_i}$, i.e., if arc $<i,p_i>$ is a bottleneck arc, then we must force arc $<i,p_i>$ to be IN whenever the solution completely homes subtree T(i) on an external node j. Since $S2_i$ and $S3_i$ are empty for this special case, the subtree-splitting arc installation inequality (5.6) reduces to the bottleneck-arc installation inequality (5.2). Similarly, the subtree-splitting arc expansion inequality (which imposes a lower bound on the arc expansion variable s_{ip_i}) generalizes the bottleneck-arc expansion inequality (5.3).

Our discussions thus far have focused on the case when node i homes on an external node j. Suppose T(i) is self-sufficient, i.e., node i homes on an internal concentrator at node $j \in T(v)$ for some successor $v \in S_i$. In this case, we wish to use the demand parameters to fix or restrict, if possible, the cable installation and expansion variables for arc $<i,v>$. Since i homes on $j \in T(v)$, the successor subtree T(v) must also be self-sufficient, but every other successor subtree T(u), for all $u \in S_i \backslash \{v\}$ can either completely home on j, partially home or j, or be self-sufficient.

Thus, we can separately consider $3^{|S_i|-1}$ combinations of successor homing patterns. Using our previous notation, we consider only cases Q with $v \in S1_i$. As before, we can compute upper and lower bounds on the flow on arc <i,v> for each case. The lower bound $fmin_{iv}(Q)$ has the same form as equation (5.5). However, for the upper bound $fmax_{iv}(Q)$ we must add to the right-hand side of equation (5.4) the total demand of all nodes outside subtree T(i). Again, we use these bounds to determine if arc <i,v> must be IN, OUT, or FREE for case Q, and impose the corresponding arc installation and expansion inequalities.

Our subtree-splitting inequalities consider combinations of homing patterns for the immediate successors of node i. We can further refine this partition of homing patterns and obtain sharper flow bounds (and, hence, tighter inequalities) by enumerating the homing patterns for subtrees that are two levels, three levels, and so on below node i. However, incorporating these inequalities in the dynamic program adds to the algorithmic complexity of the solution approach. Our implementation performs quite well with only the first level subtree-splitting inequalities.

5.4 Modifying the Dynamic Programming Algorithm

Adding the three classes of inequalities—the assignment-forcing, bottleneck-arc, and subtree-splitting inequalities—to the original formulation introduces additional linkages between the assignment, concentrator location, and cable addition variables. Consequently, when we dualize the cable capacity constraints (2.5) using multipliers $\{\mu_{ij}\}$, the resulting Lagrangian subproblem, which we denote as [ULAN2(μ)], combines the previous uncapacitated network expansion problem [ULAN1(μ)] and the cable expansion subproblem [CES(μ)]. This section describes how to modify the ULAN dynamic programming approach [DP1] of Section 3.1 to solve the new, integrated Lagrangian subproblem.

To incorporate the new valid inequalities, we exploit the dynamic program's ability to account for (uncapacitated) arc fixed costs Γ_{ij}. These fixed costs were zero in the uncapacitated subproblem ULAN1(μ) for the basic formulation [LAN1]. Adding the inequalities of Sections 5.1 to 5.3 effectively introduces nonzero arc fixed costs that vary depending on the homing pattern. Since our cost transformation (equation (3.1)) includes the arc fixed cost Γ_{ik} in the assignment cost a_{ij} only if $k \in P_{ij}$, we automatically satisfy the assignment-forcing cable installation inequalities (5.1), i.e., the Lagrangian subproblem solution does not install arc <i,k> if node i does not home via node k.

We now show how the bottleneck-arc inequalities and the subtree-splitting constraints determine the value of the arc fixed costs Γ_{ik}. Suppose, for some feasible

case Q, the subtree-splitting inequality for arc <i,k> specifies that this arc must be IN (since $fmin_{ik}(Q_i) > B_{ik}$), i.e., z_{ik} must be set to 1, and s_{ik} must be greater than or equal to $(fmin_{ik}(Q)-B_{ik})$ if the solution selects the homing pattern Q. Constraint (2.6) together with the upper bound $fmax_{ik}(Q)$ specify an upper limit of $\hat{M}_{ik} = \min \{M_{ik}, fmax_{ik}(Q)-B_{ik}\}$ on s_{ik}. The variables z_{ik} and s_{ik} have coefficients of G_{ik} and $(e_{ik}-\mu_{ik})$ in the objective function of subproblem ULAN2(μ); when arc <i,k> is forced to be IN, this subproblem has the following optimal solution:

$$\begin{aligned} z_{ik} &= 1, \text{ and} \\ s_{ik} &= fmin_{ik}(Q)-B_{ik} \quad \text{if } (e_{ik}-\mu_{ik}) \geq 0, \text{ and} \\ &= \hat{M}_{ik} \quad \text{if } (e_{ik}-\mu_{ik}) < 0. \end{aligned}$$

Effectively, these optimal values contribute an equivalent "uncapacitated" arc fixed cost of

$$\Gamma^{IN}_{ik}(Q) = G_{ik} + \min \{ (e_{ik}-\mu_{ik}) [fmin_{ik}(Q_i)-B_{ik}], (e_{ikj}-\mu_{ik}) \hat{M}_{ik} \}. \qquad (5.7a)$$

Similarly, if the upper and lower bounds force arc <i,k> to be OUT or FREE (i.e., if $fmax_{ik}(Q) \leq B_{ik}$ or $fmin_{ik}(Q) \leq B_{ik} < fmax_{ik}(Q)$), we have the following equivalent fixed costs corresponding to homing pattern Q:

$$\Gamma^{OUT}_{ik}(Q) \equiv 0; \text{ or} \qquad (5.7b)$$

$$\Gamma^{FREE}_{ik}(Q) \equiv \min \{0, G_{ik} + (e_{ik} - \mu_{ik}) \hat{M}_{ik}\}. \qquad (5.7c)$$

To recapitulate, for every feasible case Q, we: (i) compute the upper and lower bounds $fmax_{ik}(Q)$ and $fmin_{ik}(Q)$ for arc <i,k>, (ii) compare these bounds with the existing capacity B_{ik} to determine if arc <i,k> must be IN, OUT, or FREE, and (iii) accordingly set the uncapacitated fixed cost $\Gamma_{ik}(Q)$ for arc <i,k> corresponding case Q equal to either $\Gamma^{IN}_{ik}(Q)$, $\Gamma^{OUT}_{ik}(Q)$, or $\Gamma^{FREE}_{ik}(Q)$. Since these arc fixed costs vary by case, applying our cost transformation (3.1) gives different assignment costs $a_{ij}(Q)$ for different homing patterns Q within subtree T(i). Consequently, the dynamic program must have the ability to differentiate the homing costs for various cases.

Recall that the dynamic program already treats the self-sufficient case separately; the tree cost TC(i) denotes the optimal total (assignment) cost of covering all nodes in rooted subtree T(i) assuming T(i) is self-sufficient. To distinguish between complete and partial homing of subtree T(i), we replace our original homing cost HC(i,j) (see Section 3.1.2) with the following two *complete* and *partial* homing costs:

CHC(i,j) = Cost of serving all nodes in T(i), assuming <u>all</u> nodes of T(i) home on an external node $j \notin T(i)$; and

PHC(i,j) = Cost of serving all nodes in T(i), assuming node i homes on (external or internal) node j, and T(i) contains at least one concentrator.

When T(i) completely homes on an external node j, all its successor subtrees T(u) for all $u \in S_i$ also home completely on node j (i.e., case $Q^c = \{S_i, \phi, \phi\}$). Thus,

$$CHC(i,j) = a_{ij}(Q^c) + \sum_{u \in S_i} CHC(u,j). \quad (5.8)$$

If arc $<i,p_i>$ is a bottleneck arc, equation (5.7a) introduces a positive arc fixed cost $\Gamma_{ip_i}(Q^c)$, which the cost transformation (3.1) adds to the assignment cost $a_{ij}(Q^c)$.

We compute the partial homing cost PHC(i,j) as the minimum homing cost over all partial homing cases. If $j \notin T(i)$, the partial homing cases consist of all combinations of complete homing, partial homing, and self-sufficient successor subtrees of node i except the case Q^c in which every successor completely homes on node j. If $j \in T(v)$ for some $v \in S_i$, we consider only cases in which subtree T(v) is self-sufficient (i.e., $v \in S1_i$). Let PHC(i,j,Q) denote the partial homing cost for case Q assuming node i homes on node j. We compute node i's partial homing costs as follows:

$$PHC(i,j,Q) = a_{ij}(Q) + \sum_{u \in S1_i} CHC(u,j) + \sum_{u \in S2_i} PHC(u,j) + \sum_{u \in S3_i} TC(u), \text{ and} \quad (5.9)$$

$$PHC(i,j) = \underset{\text{all partial homing cases } Q}{\text{minimum}} PHC(i,j,Q). \quad (5.10)$$

Finally, we compute the tree cost TC(i) as:

$$TC(i) = \underset{j \in T(i)}{\text{minimum}} PHC(i,j). \quad (5.11)$$

Equations (5.8) to (5.10) are the recursive equations for the *enhanced dynamic program* to solve the uncapacitated network expansion Lagrangian subproblem ULAN2(μ). As before, we consider nodes in bottom-to-top sequence to ensure that the required quantities on the right-hand side of these recursive equations are available when needed.

To summarize, this section has described several classes of valid inequalities that strengthen the local access network planning problem formulation. We also discussed modifications to the dynamic programming algorithm needed to incorporate these cuts. The next section presents computational results comparing the performance of the decomposition method, without the cuts and with the cuts and other model enhancements, for three test networks.

6. Computational Results

As the discussions in Sections 4 and 5 suggest, we followed an iterative process of computational testing and algorithmic enhancement—testing the algorithm using three problems derived from actual networks, gaining insight about its shortcomings, and devising techniques (preprocessing, coefficient reduction, valid inequalities) to address these deficiencies. Section 6.1 describes some broad characteristics of our test networks. We then present computational results using: (i) the basic model [LAN1], and (ii) the enhanced model with reduced cable expansion bounds and valid inequalities. To obtain benchmarks for solution time and quality, we also attempted to solve the mixed-integer formulation and the linear programming relaxation for all three problems using a general purpose mathematical programming software package (LINDO). Section 6.2 reports the IP and LP results, and our initial experience with the basic [LAN1] model. Section 6.3 shows the dramatic performance improvement derived from the modeling and algorithmic enhancements. Section 6.4 reports on several computational tests with parametrically-scaled demand and cost values for the three network configurations.

6.1 Test Problems

Our computational tests employed three test problems—a 27-node network (***Problem 1***), a 25-node network (***Problem 2***), and a 41-node network (***Problem 3***). Each network represents an existing feeder route from a Central office. Our implementation assumes (for convenience, and without loss of generality) that each node of the network has at most two successors. Problems 2 and 3 had a total of three nodes with three successors each; all other nodes had 1 or 2 successors. We added a zero-demand node (with infinite concentrator cost) and a zero-cost edge to convert each three-successor node into 2 two-successor nodes.

For each network, telecommunication planners provided us with information on the projected demand at every customer node, and the current cable capacities. Figure 2 shows demand and capacity information for the 41-node problem. We examined the actual, prevailing costs for various transmission and concentration technologies to estimate the fixed and variable cost coefficients for our model. Cable expansion costs vary by (i) construction type (aerial, buried, or underground), and (ii) cable gauge (22, 24 or 26). On each edge, new cables must have the same cable type and gauge as existing cables on that section. For concentrators, we use a piecewise-linear, concave function containing three segments, each representing a different technology. For our test problems, concentrator costs do not vary by node (except for the root node which has zero concentrator cost).

6.2 Initial Computational Results

6.2.1 Lagrangian results for the Basic Model [LAN1]

We initially implemented the Lagrangian/dynamic programming algorithm for the basic model [LAN1] in FORTRAN on an IBM 3083. The implementation incorporated the problem preprocessing method (Section 4.1), and the local improvement heuristic (Section 3.3) but did not contain the coefficient reduction method (Section 4.2), or any of the valid inequalities described in Section 5. For all our tests, we initialized all Lagrange multipliers to value zero, used an initial step size multiplier (see, for example, Held et al. [1974]) of 2.0, and permitted a maximum of 100 subgradient iterations (the procedure might terminate earlier if the percentage gap between the upper and lower bounds reduces to a very small fraction).

Table 1 summarizes the computational results of this initial solution approach. We measure *preprocessing effectiveness* in terms of the proportion of assignment variables x_{ij} that the preprocessing technique eliminates (the total number of possible assignments shown in Table 1 excludes assignments to dummy nodes and from the root node). The *% gap* statistic, defined as the difference between the best upper and lower bounds as a percentage of the lower bound, measures the Lagrangian algorithm's effectiveness. The CPU times reported in Table 1 correspond to the total computational time (in seconds) on the IBM 3083, including the time required for input and initialization, preprocessing, subgradient optimization, and the local improvement heuristic.

The % gaps range from 64% to 124%, and the algorithm required approximately 8 seconds for the smaller problems (Problems 1 and 2), and 22 seconds for the 41-node problem (Problem 3). To assess the impact of preprocessing, we performed a separate set of computational experiments (not reported here) without the provisions for variable elimination. For Problem 3, preprocessing reduced the % gap from 160% to 124% gap through improvements in both the upper and lower bounds; Problems 1 and 2 had only marginal improvements. For all three test problems, the preprocessing procedure did not require more than 0.1 seconds of CPU time.

6.2.2 Linear and Integer Programming Solutions for Basic Model

To determine the underlying cause of the large % gaps in Table 1, and to generate benchmarks for the bounds and computation times, we attempted to solve the integer programming formulations as well as the linear programming relaxations for all three test problems using LINDO (a commercial mathematical programming package, running on the IBM 3083 mainframe). To reduce the formulation size we substituted the self-assignment variables x_{jjm} for the concentrator installation variables y_{jm} (the index m = 1,2,3 represents the three concentrator technologies), and used undirected cable installation and expansion variables (the directed version of these variables strengthens the LP relaxation only when we incorporate the additional inequalities of Section 5); this reformulation reduces the number of

binary variables by almost half and eliminates the concentrator location constraints (2.3) and the arc orientation constraints (2.7), without weakening the basic fomulation. We also used the preprocessing results to eliminate several additional variables and constraints.

We solved two versions of the basic model—our original formulation **with backfeed**, and a restricted version **without backfeed**. Prohibiting backfeed reduces the number of assignment variables (and constraints) since a node i can now home only on downstream nodes j lying on path P_{i0}; this restriction can also eliminate some cable expansion variables (and constraints) since fewer edges are likely to have maximum possible flow values greater than existing capacity. Consequently, the "without backfeed" problem formulation contained fewer variables and constraints than the original, unrestricted version. Since prohibiting backfeed restricts the space of feasible solutions, we expect the corresponding LP lower bound to be higher, but the optimal with-backfeed integer solution might be cheaper than the without-backfeed solution.

Table 2 contains statistics on the reduced problem dimensions (ranging from 364 variables and 320 constraints to 2908 variables and 2830 constraints) and the optimal LP and IP values, with and without backfeed, for each test network. We were able to optimally solve all 6 linear programming relaxations, and 5 out of the 6 integer programs. For Problem 3 with backfeed, the branch-and-bound procedure did not reach optimality even after more than 10 hours of (elapsed) computer time. As expected, the LP values without backfeed exceed the with-backfeed values—by 12%, 3.6%, and 8%, respectively, for the three problems. However, the optimal integer solutions were the same. For Problems 1 and 2, the optimal integer solutions coincide with the best Lagrangian-based heuristic solutions (see Table 1); for Problem 3, the Lagrangian-based solution was only 2.3% more expensive than the best IP value. Finally, the Lagrangian lower bounds were within 5% of the LP lower bounds (with backfeed). However, the computation time to solve even the LP relaxation was, on average, *three* times the CPU time required for the entire Lagrangian procedure. Solving the IP by branch and bound required considerably more time.

The results in Table 2 confirm that the Lagrangian-based heuristic generates near-optimal solutions, but the LP (and, hence, the Lagrangian) lower bounds for the basic model are weak, and hence do not provide a reliable measure of the quality of the heuristic solutions. This observation led to our study of various modeling and algorithmic enhancements described in Sections 4.2 and 5 to strengthen the relaxation and generate better lower bounds.

6.3 Improved Computational Results

We implemented the enhanced version of our Lagrangian algorithm (Section 5.4) in the C programming language on a Macintosh II computer (with a Math co-processor). This final implementation incorporated the following features:

(i) initial heuristic solution, using local improvement on the "centralized" and "distributed" starting solutions;
(ii) problem preprocessing to eliminate variables;
(iii) the Lagrangian-based heuristic with local improvement;
(iv) cost-based cable expansion bounds; and,
(v) the assignment-forcing, bottleneck-arc, and subtree-splitting inequalities.

We used the same subgradient settings described in Section 6.2. Table 3 contains summary statistics on the the initial upper bound, the (final) best upper and lower bounds, the % gap, and the total computation time (elapsed time on Mac II, including time for input/output, initialization, heuristic, and subgradient iterations) for the three test problems. Again, we have scaled all the bounds for each problem with respect to the optimal LP value.

Comparing the results of Table 1 and Table 3 highlights the dramatic reduction in the % gap due to our enhancements—from 64%, 81%, and 124% to 1.2%, 3.2%, and 7.0%, respectively, for Problems 1, 2 and 3. As before, the Lagrangian-based heuristic procedure found the optimal solutions to Problems 1 and 2. For Problem 3, the enhancements to the Lagrangian lower bounding procedure also led to a modest improvement (1.6%) over the previous heuristic solution (using the basic model); this improved Lagrangian-based heuristic solution is only 0.66% more expensive than the best integer solution found using LINDO.

The order-of-magnitude reduction in the % gaps is mainly due to the vastly improved lower bounds. The new Lagrangian lower bounds are 60 to 95% larger than the optimal LP values for the basic [LAN1] model. These results suggest that, even though the local access network planning formulation has numerous other valid inequalities, the cumulative effect of the few classes of cuts that we selected and implemented far exceeds the potential effectiveness of the remaining, more complex inequalities.

Although the computation times reported in Tables 2 and 3 are not commensurate, we estimate that general-purpose branch and bound methods might require several orders of magnitude more computation time than the composite Lagrangian relaxation algorithm. Finally, we note that, even with weak lower bounds, the Lagrangian relaxation procedure generates good starting solutions for local improvement. For Problems 2 and 3, the Lagrangian-based heuristic was 15 to 20% cheaper than the initial heuristic solution.

6.4 Computational Results with Demand and Cost Variations

To test the method's robustness as demand and cost parameters change, we applied the algorithm to problem variations created by scaling the input data, holding the topology of the three networks fixed. In particular, we tested the following scaled versions of each problem:

(i) **Demand variation**: We considered scenarios with uniformly lower or higher demand values obtained by multiplying the original demand at each node by a common scale factor. We tested three demand scale factors: 0.5, 2.0, and 5.0;

(ii) **Cable expansion cost variation**: We uniformly increased or decreased the variable cable expansion cost by a common scale factor for all edges. We tested two values of the variable cost scale factor: 0.5 (lower variable cost), and 2.0 (higher variable cost); and,

(iii) **Concentrator cost variation**: We increased or decreased the fixed concentrator cost (for every concentrator type) by a fixed amount at all locations. For our tests, we considered a concentrator fixed cost increase (or decrease) equal to approximately 50% of a type 2 concentrator's fixed cost.

Table 4 compares the computational results for the 6 variations of each problem with the base case. For 19 out of the 21 problem instances, the % gap between the upper and lower bounds is less than 8%. The trends in the best expansion strategies are consistent with our expectations. For instance, as demand increases, concentrators become more cost-effective, and hence the number of concentrators increases. Similarly, as the cable cost increases, the expansion plans tend to replace cable expansions with additional concentrators. Table 4 also illustrates that, as the demand scale factor increases, the % gap initially increases and then declines. With very high demands, the existing cable capacities are negligible compared to the nodal demands. Consequently, the total cost of an optimal "new" network, ignoring existing cable capacities (which our dynamic program can find in a single iteration), does not differ appreciably from the optimal cost of the capacitated problem. Furthermore, as demand increases, remote homing (i.e., homing node i on another node j) becomes less attractive relative to local concentration, thus enabling the preprocessing method to eliminate many assignments and strengthen the formulation (through lower cable expansion bounds).

These experiments with varying demand and cost data were intended to provide us with a broad understanding of the robustness of our solution approach. Using a formal design of experiment analysis with multiple-factor variation would undoubtedly provide further insight, and might even suggest additional improvements to our methodology.

7. Concluding Remarks

This paper has attempted to integrate three essential ingredients of contemporary applied integer programming research:

(1) an important practical application: Many changes in the telecommunications industry—technological advances, explosive growth, capital intensive facilities, and increasing competition—have opened new opportunities to apply sophisticated, optimization-based decision support techniques for telecommunication network planning;

(2) decomposition methodology: Since its first successful application for solving large-scale traveling salesman problems (Held and Karp [1970],[1971]), Lagrangian relaxation has become a pervasive technique for exploiting special structure and solving difficult, large-scale discrete optimization models; and

(3) model improvements based on polyhedral combinatorics: Motivated by their marked success for solving large-scale traveling salesman and many other problems, polyhedral techniques have become an essential building block for designing state-of-the-art algorithms for solving several classical integer programming problems.

We have demonstrated how to exploit the local access network planning problem's special tree and routing structure, and how to integrate formulation enhancements with Lagrangian relaxation and dynamic programming. Our computational results establish the effectiveness of this combined approach for solving practical problem instances. This project also serves to highlight model-building issues such as eliciting information on common practice (e.g., the contiguity property) that can significantly impact algorithmic development, and the tradeoff between model accuracy (e.g., approximating the true concentrator costs as piecewise-linear concave functions) and solution effectiveness (time and quality).

Although our model does not consider detailed investment timing decisions and approximates the cable and concentrator costs, we believe that it is an useful building block for a practical decision support system to assist local access network planning activities. Solving a monolithic multi-period planning model is likely to be much more difficult that its single-period counterpart. Instead, as Shulman and Vachani [1990] and Jack et al. [1992] suggest, we might employ an iterative framework that first selects an end-of-horizon target network configuration, and then determines the optimal evolution plan to reach this eventual target while meeting the demand in each intermediate period. Our model can generate the target network at each iteration. Comparing this iterative scheme with other multi-period planning algorithms is an important research direction.

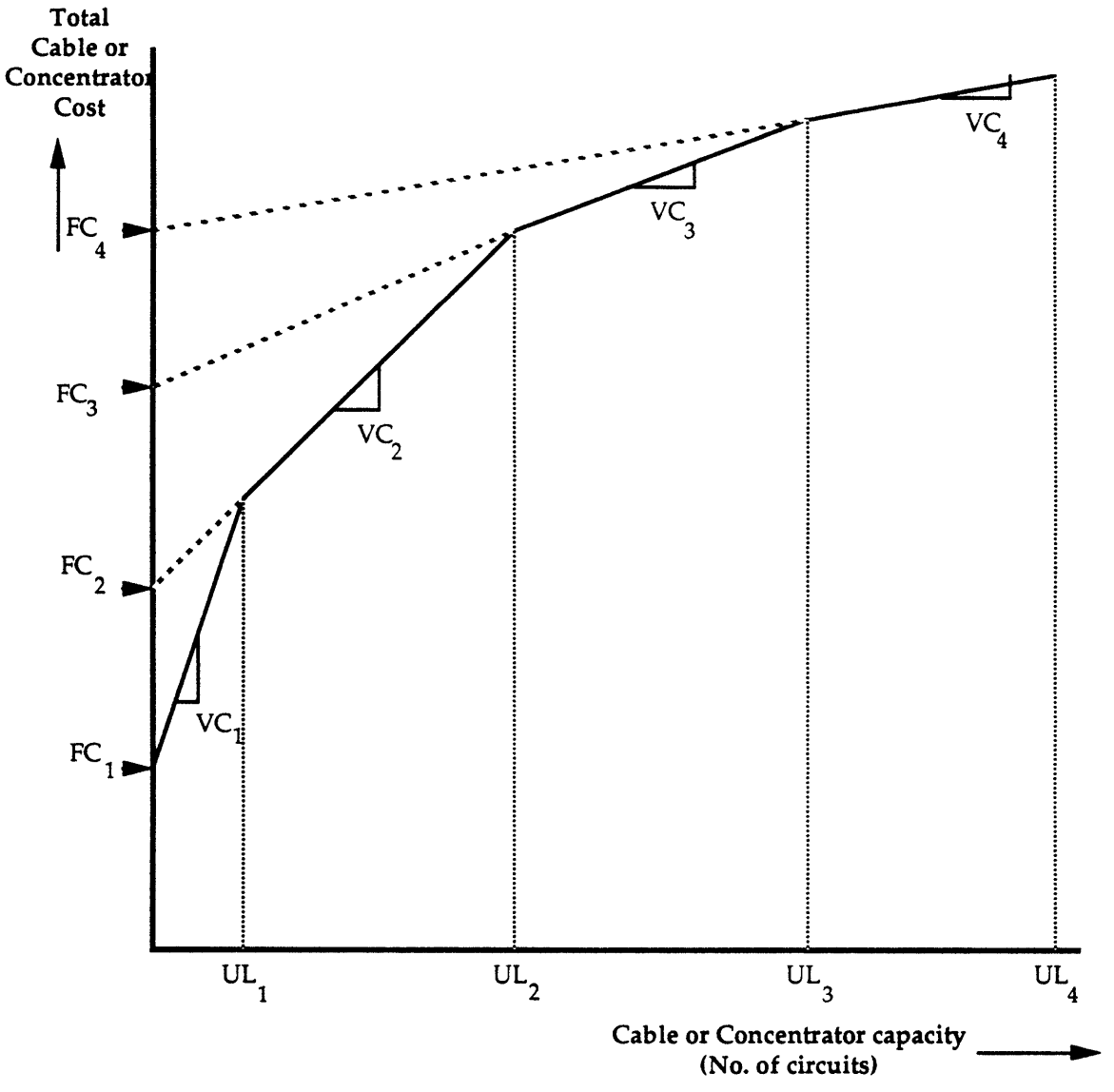
Modeling multiple services is another potentially fruitful area for investigation, especially in view of the anticipated diversification of services. Our model applies to

contexts where we can express the demands for different services (such as voice, data, and video) in commensurate units, and different services do not impose unique processing requirements or require different transmission media. With advanced technologies still under development, the structural differences between the single (aggregate) sevice model and the muliple services version is still unclear. Finally, as telephone companies transition from copper to fiber optic-based transmission in the local network, issues of reliability and connectivity might become more important. Because of the extremely high bandwith and vulnerability of fiber-optic networks, the focus of modeling efforts for the next generation of local access networks might move towards configuring reliable networks at minimum total fixed cost. Some recent research (e.g., Groetschel, Monma, and Stoer [1992]) has shed some light on these issues.

Acknowledgments: We wish to thank Marcia Helme, Jeffrey Musser, and Alexander Shulman for their valuable insights and modeling contributions, and for providing the linkage with network planners in the field. Much of Richard Wong's work was completed at Purdue University, West Lafayette, Indiana.

Figure 1

Piecewise-Linear, Concave Cable or Concentrator Cost Function



FC_m = Fixed (cable or concentrator) cost for technology m

VC_m = Variable (cable or concentrator) cost for technology m

UL_m = Upper limit on (cable or concentrator) throughput for technology m

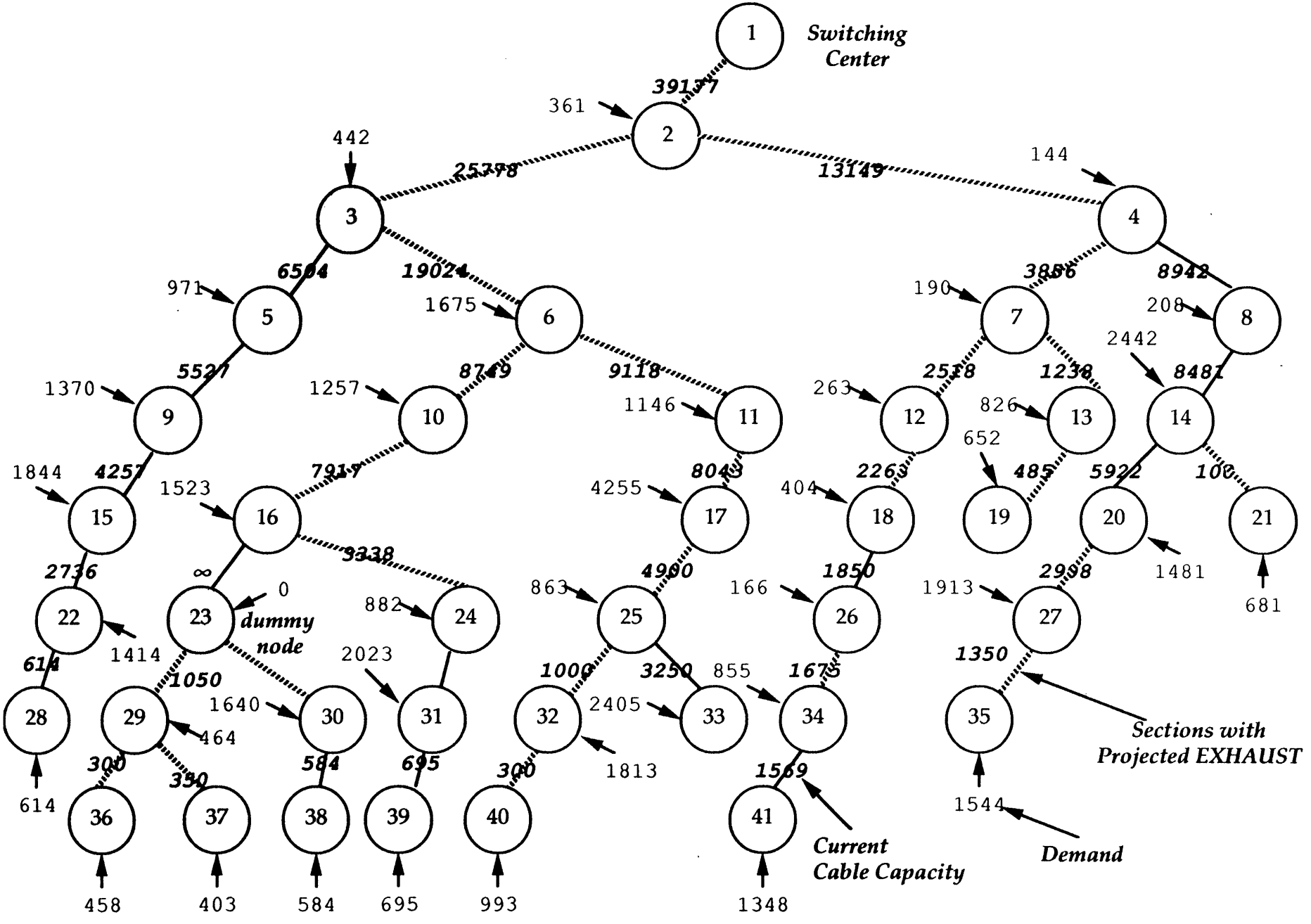
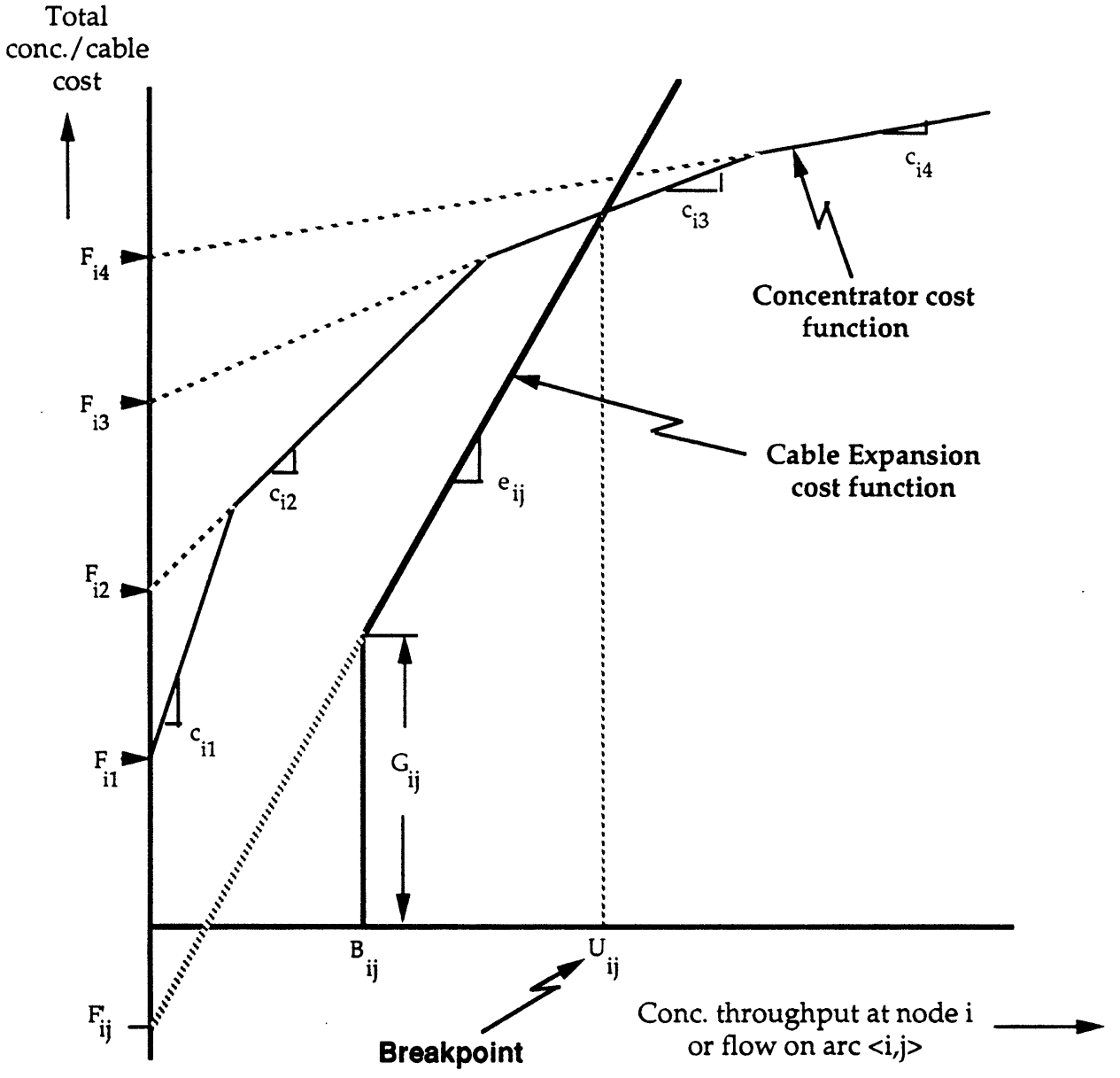


Figure 2: *Network Configuration for Problem 3*

Figure 3
Reducing the Cable Expansion Bound M_{ij}



F_{im} = Fixed cost of type m concentrator at node i

c_{im} = Variable cost of type m concentrator at node i

G_{ij} = Fixed cost of cable expansion on arc <i,j>

e_{ij} = Variable cost of cable expansion on arc <i,j>

B_{ij} = Existing capacity of arc <i,j>

Table 1

Computational Results for the Basic Lagrangian Relaxation Scheme
(with Preprocessing and the Local Improvement Heuristic)

Statistic	**Problem 1**	**Problem 2**	**Problem 3**
Number of nodes	27	25	41
No. of possible assignments*	676	441	1521
% reduction in assignment variables	25%	24%	30%
Best Upper Bound using Lagrangian-based heuristic†	162.72	173.91	212.40
Best Lagrangian Lower Bound†	99.25	96.15	94.91
% gap**	**64%**	**81%**	**124%**
CPU time (seconds on IBM 3083)§	8.4 secs	6.8 secs	22.2 secs

* % reduction = No. of assignments eliminated ÷ total no. of possible assignments (excluding assignments to dummy nodes and from root node).

† Bounds for each problem are scaled with respect to optimal value of LP relaxation (Table 2).

** % gap = (Best Upper Bound - Best Lower Bound) /(Best Lower Bound)

§ CPU time includes time for input and initialization, subgradient iterations, and the local improvement heuristic.

Table 2

Linear and Integer Programming Results using LINDO
for the Basic Model [LAN1]

	Problem 1	Problem 2	Problem 3
# Nodes	27	25	41
# Variables[§]: Integer/Continuous	814/17	911/18	2870/38
# Constraints[§]	781	889	2830
Optimal Value[*]			
• LP Relaxation	100.00	100.00	100.00
• Integer Program	162.72	173.91	207.70[†]
CPU secs (on IBM 3083)			
• LP Relaxation	14.2	24.3	72.9
• Integer Program	307.8	691.8	†
No. of *Iterations*			
• LP Relaxation (# pivots)	384	1058	1295
• Integer Prog. (# pivots/branches)	47/15898	89/56239	-
Without Backfeed			
# Variables[§]: Integer/Continuous	421/5	352/12	765/24
# Constraints[§]	376	320	711
Optimal Value[*]			
• LP Relaxation	111.98	103.63	108.02
• Integer Program	162.72	173.91	207.70
CPU secs (on IBM 3083) for LP Relaxation/Integer Program	2.8/36.1	4.2/72.4	6.2/2862.6

§ Number of variables and constraints shown in table correspond to compact formulation (Section 6.2.2) after preprocessing .

* Objective function values for each problem are scaled with respect to optimal value of LP relaxation with backfeed.

† Branch-and-bound for Problem 3 terminated, without reaching optimality, after **10 hours** of elapsed time on the IBM 3083. The table reports the value of the best incumbent at termination.

Table 3

Computational Results for the
Enhanced Lagrangian Relaxation Scheme

Statistic	**Problem 1**	**Problem 2**	**Problem 3**
Number of nodes	27	25	41
% reduction in assignment variables*	25%	24%	30%
Intial Upper Bound †	162.72	208.19	242.16
Best Upper Bound using Lagrangian-based heuristic†	162.72	173.91	209.08
Best Lagrangian Lower Bound†	160.85	168.52	195.36
% gap**	**1.2%**	**3.2%**	**7.0%**
Computation (elapsed) time on Mac II §	285 secs	223 secs	879 secs.

* % reduction = No. of assignments eliminated ÷ total no. of possible assignments (excluding assignments to dummy nodes and from root node).

† Bounds for each problem are scaled with respect to optimal value of LP relaxation with backfeed (Table 2).

** % gap = (Best Upper Bound - Best Lower Bound) /(Best Lower Bound)

§ Computation (elapsed) time includes time for input, initialization, heuristic, and subgradient iterations.

Table 4

Computational Results with Demand and Cost Variations

Parameter Variation	Problem 1			Problem 2			Problem 3		
	% Problem Reducn.	% GAP	Comput. time	% Problem Reducn.	% GAP	Comput. time	% Problem Reducn.	% GAP	Comput. time
Base Case	***25%***	***1.2%***	***285 secs***	***24%***	***3.2%***	***223 secs***	***30%***	***7.03%***	***879 secs***
Demand x 0.5	12%	0.0%	32 secs	12%	0.0%	30 secs	9%	0.0%	82 secs
Demand x 2.0	52%	11.3%	259 secs	53%	5.5%	203 secs	63%	2.71%	648 secs
Demand x 5.0	79%	1.5%	221 secs	80%	0.0%	179 secs	87%	1.1%	546 secs
Var. Cable Cost x 0.5	14%	0.0%	315 secs	14%	3.2%	318 secs	12%	7.5%	1210 secs
Var. Cable Cost x 2.0	40%	3.0%	311 secs	35%	2.8%	234 secs	46%	6.3%	850 secs
Fixed Conc. Cost - 50%†	40%	2.0%	282 secs	36%	5.6%	218 secs	43%	19.9%	836 secs
Fixed Conc. Cost + 50%†	17%	0.5%	292 secs	19%	3.0%	256 secs	20%	5.8%	965 secs

† Increase/decrease in concentrator fixed cost ≈ 50% of type 2 concentrator's fixed cost

References

Aghezzaf, E. H., and L. A. Wolsey, "Modeling Piecewise Linear Concave Costs in a Tree Partitioning Probelm", Working Paper, Universite Catholique de Louvain, Louvain-la-Neuve, Belgium (1990).

Balakrishnan, A., T. L. Magnanti, A. Shulman, and R. T. Wong, "Models for Planning Capacity Expansion in Local Access Telecommunication Networks", *Annals of Operations Research*, Vol. 33, pp. 239–284 (1991).

Balakrishnan, A., T. L. Magnanti, and R. T. Wong, "Local Access Telecommunication Network Expansion: Modeling and Polyhedral Characterization", in preparation (1992).

Barany, I., J. Edmonds, and L. A. Wolsey, "Packing and Covering a Tree by Subtrees", *Combinatorica*, Vol. 6, pp. 221-233 (1986).

Fisher, M. L., "The Lagrangian Relaxation Method for Solving Integer Programming Problems", *Management Science*, Vol. 27, pp. 1-18 (1981).

Groetschel, M., C. L. Monma, and M. Stoer, "Computational Results with a Cutting Plane Algorithm for Designing Communication Networks with Low-connectivity Requirements", *Operations Research*, Vol. 20, pp. 309-330 (1992).

Held, M., and R. M. Karp, "The Traveling Salesman Problem and Minimum Spanning Trees", *Operations Research*, Vol. 18, pp. 1138-1162 (1970).

Held, M., and R. M. Karp, "The Traveling Salesman Problem and Minimum Spanning Trees: Part II", *Mathematical Programming*, Vol. 1, pp. 6-25 (1971).

Held, M., P. Wolfe, and H. D. Crowder, "Validation of Subgradient Optimization", *Mathematical Programming*, Vol. 6, pp. 62-88 (1974).

Hoffman, K. and M. W. Padberg, "LP-based Combinatorial Problem Solving", *Annals of Operations Research*, Vol. 4, pp. 145-194 (1985).

Jack, C., S. Kai, and A. Shulman, "NETCAP–An Interactive Optimization System for GTE Telephone Network Planning", *Interfaces*, Vol. 22, pp. 72-89 (1992).

Kariv, O., and S. L. Hakimi, "An Algorithmic Approach to Network Location Problems II: The p-medians", *SIAM Journal of Applied Mathematics*, Vol. 27, pp. 539-560 (1979).

Nemhauser, G. and L. A. Wolsey, Integer and Combinatorial Optimization, John Wiley & Sons, New York (1988).

Shulman, A., and R. Vachani, "An Algorithm for Capacity Expansion of Local Access Networks", *IEEE Infocom '90*, San Francisco, California (1990).

Standard and Poor's, *Industry Surveys: Telecommunications*, January 23 (1992).

Wilson, Carol, "An Uncertain Future for LECs", *Telephony*, December 16 (1991).