Annals of Operations Research 106, 199–228, 2001
© 2002 Kluwer Academic Publishers. Manufactured in The Netherlands.

Node Placement and Sizing for Copper Broadband Access Networks *

TAMRA CARPENTER, MARTIN EIGER and DAVID SHALLCROSS
Telcordia Technologies, Inc., 445 South Street, Morristown, NJ 07960, USA

PAUL SEYMOUR
Department of Mathematics, Princeton University, Princeton, NJ 08544, USA

Abstract. We consider a node placement and sizing problem that arises in certain types of broadband access architectures, such as ADSL and FTTC. We consider three variants of the problem that become progressively more restrictive, to capture realistic planning concerns and to produce solutions that have desirable practical attributes. A distinguishing feature of the problem is a constraint that limits the distance between each customer and the placed node that is assigned to serve it. We present a dynamic programming algorithm to solve the two most practical variants of the problem, and we provide computational results for both realistic and randomly generated test problems.

1. Introduction

In network planning, the term *node placement* refers to the process of determining the locations, or *nodes*, at which to install certain types of networking equipment. In placing nodes, a planner selects the nodes to equip from a possibly larger set of candidate nodes. The term placement is used because there is no active equipment between placed nodes, so we may view the placed nodes as being logically connected by uninterrupted runs of cable. Thus, the equipped nodes become the only nodes in a logical view of the network. A simple example of node placement in a tree network is illustrated in figure 1. Often concurrent with node placement is *node sizing*, which is the process of determining the capacity of equipment to be installed at a placed node. In this paper, we consider node placement and sizing for broadband access technologies that combine copper and fiber distribution media. We use the term *copper broadband access* (CBA) as a generic term for these technologies that include digital subscriber line (DSL), fiber-to-the-curb (FTTC), and their many variants. DSL technology itself includes variants like ADSL, HDSL, and VDSL among others, which are generically referred to as xDSL. Similarly, FTTx generically refers to more fiber-rich technologies that include FTTC and its relatives: fiber-to-the-neighborhood, fiber-to-the-cabinet, fiber-to-the-home (FTTH) and others. We use the term CBA to refer to all of these technologies except FTTH, which is essentially fiber all way from the central office (CO) to the subscriber.

* Copyright 2001 Telcordia Technologies, Inc. All rights reserved.

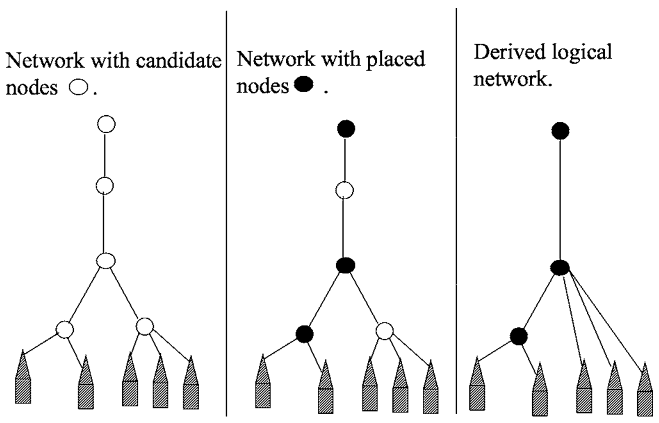
Network with candidate nodes ○.
Network with placed nodes ● .
Derived logical network.

Figure 1. Illustration of node placement in a network

Although they differ in their details, the CBA technologies are all characterized by the combined use of copper and fiber to deliver broadband and more traditional services to subscribers. These services include voice telephony, high-speed data, and may also include video. In general, subscribers may be either business or residential, but it will simplify discussion to restrict ourselves to a single type of subscriber, which we assume to be residential.

CBA technologies deploy fiber between the central office and an intermediate point in the network, and they use traditional copper into the subscribers' premises. At this intermediate node, equipment converts signals between the optical domain, used on the fiber, and the electrical domain, used on the copper. We refer to this node as the O/E node (optical/electrical), and to the equipment as O/E converters or O/E devices. This equipment also typically performs other networking functions, such as multiplexing and demultiplexing of signals. Traditional telephone networks use copper all the way from the CO to the customer, and thus no O/E equipment is used.

The use of fiber in the network allows more information, or higher bandwidth, to be delivered to subscribers, thereby enabling higher-speed services. The use of copper makes CBA technologies attractive to telephone providers who have large investments in existing copper infrastructure. Copper, however, poses some technical challenges that lead to the node placement problem that we consider in this paper. Attenuation and other impairments along the copper degrade signals and limit the bandwidth that can be delivered to subscribers. The distance that a signal can travel on copper varies, depending on the bandwidth of the signal, as well as the gauge and condition of the copper plant.

From a customer's perspective, the CBA technologies differ primarily in the amount of information that can be delivered. Each CBA technology is typically associated with a characteristic bandwidth, or line speed, that is assured by requiring that each customer be no more than a prescribed distance from the O/E node that connects it to the CO. This prescribed distance applies to every subscriber in the network and is, again, a characteristic of the particular technology. We refer to this characteristic dis-

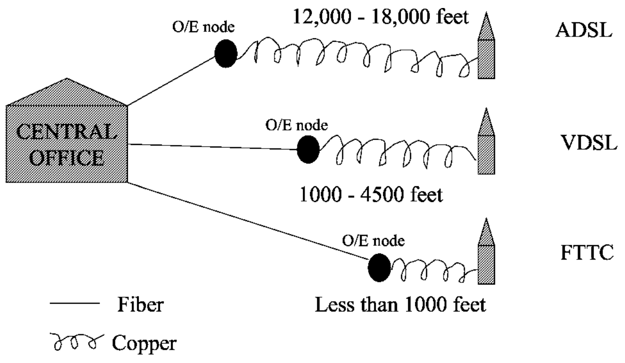

Figure 2. Typical range limits for different types of xDSL.

tance as the *range limit*, or simply *range*, for the technology. We illustrate typical range limits in figure 2, which is based on a similar figure in [1].

From a mathematical standpoint, the design problems associated with different types of CBA technology are very similar, but they have some practical differences. First, they differ in their range limits. As a consequence, they differ in the degree to which they can leverage the existing copper plant. They also differ in the type of equipment that can be placed at the O/E nodes. As a rule, the O/E converters for longer-reach variants, like ADSL, can support more customers but at lower bitrates. Finally, there may be differences in the types of locations at which converters may be placed. For shorter-reach cases, every telephone pole, roadside pedestal, or manhole may be a candidate node, while the choice might be more limited for longer-reach technologies. Although there are many practical differences between the types of CBA, mathematically they manifest themselves in differences in the parameters to an otherwise identical problem. Thus, we consider CBA network planning in this unified context.

In the next section, we provide integer programming formulations for the node placement and sizing problem. In section 3, we develop a dynamic programming approach to address the more practical of the formulations presented. In section 4, we present results from two realistic networks that we have encountered and several random ones. In section 5, we describe some variations and extensions of the basic problems described in section 2, and we show how many of these can be handled with the same core algorithm. Section 6 contains brief concluding remarks.

2. Problem formulations

The CBA node placement and sizing problem requires placing and sizing O/E nodes in a given candidate network. Regardless of the type of CBA network we are planning, the problem is essentially the same. The problem is described using a network of candidate nodes that forms a tree with the CO at its root. Each customer routes directly to a single candidate node to form what might be viewed as a larger tree network with the customers at the leaves. However, throughout the paper, we will use the terms *tree* and *network* to refer to the network of candidate nodes, and we will use the terms *augmented tree* or *augmented network* to refer to the larger tree that includes the customers.

Each link in the augmented tree has an associated length. In addition, we have O/E converters that we can place at the candidate nodes to serve the customers. There may be several different models of O/E converter that differ in their costs and capacities, where the capacity of a converter is the number of subscribers that it may serve. Specific equipment models and range limits vary among the different flavors of CBA. Given the augmented network and the equipment data, the node placement and sizing problem is to place equipment at some subset of the candidate nodes so that customers can be feasibly served at the least cost. The nodes at which equipment is placed are the solution to the node placement problem and the capacity of the equipment placed is the solution to the node sizing problem.

In our work, we have considered three variants of the node placement and sizing problem. We can view them as becoming progressively more restrictive to produce solutions that have desirable practical attributes. Thus, the precise meaning of a feasible solution remains to be defined. We will begin with the least restrictive problem statement and then impose further restrictions.

2.1. The baseline problem

This first version of the problem is originally stated in [5] and is extensively studied in [12]. We believe that this version captures the technical capability of the network equipment, but does not consider the practical issues of network construction and management. It also does not ensure effective reuse of existing copper. Thus, although it delivers a solution that is technically correct, these solutions may be difficult to implement and manage in the network. We present this version of the problem as a baseline from which to build the more constrained formulations, but we do not address solving this version of the problem in this paper.

The constraints for this version of the problem require that:

(1) Each customer is served at a node that is within the prescribed range limit.

(2) The number of customers served at a node does not exceed the capacity of the O/E converters placed at the node.

The input for our problem is the augmented tree network, a range limit, and a set of O/E converters for each candidate node. To state and later solve the problem, we perform two preprocessing operations. First, we use the augmented tree to compute distances between candidate nodes and customers. This identifies the subset of candidate nodes that are within range to serve each customer.

Next, we augment the set of O/E converters available at a candidate node to include combinations. We assume that it is possible to place converters in all possible combinations and multiples at any candidate site. This leads to a formulation with general integer variables representing the number of each type of converter to place at each candidate node. We prefer an alternate binary formulation that first requires determining the least-cost way to serve s subscribers at each location, for all values of s between 1 and the number of subscribers that reach it. The binary formulation that results is often more amenable to general integer programming [13], but it is also more consistent with the way that our dynamic programming algorithm works, which is the reason for our preference. To avoid confusion, we will refer to the non-dominated combinations of converters computed for a candidate node as the *stacks* available for that node. Each stack is characterized by a cost, a capacity, and a set of constituent O/E devices.

Computing the stacks is equivalent to solving an integer knapsack cover problem to obtain non-dominated combinations of converters [12]. When the input includes only the amount of customer demand to cover (i.e., an integer), this would not be a polynomial-time operation. However, it is polynomial in the number of houses. Also, the number of equipment models available at each candidate node is generally small, so the computation is fast in practice. The set of O/E converters available to place might be common to all candidate nodes or it might have site dependencies. In this paper, we allow site dependencies.

The objective for our problem is to minimize the cost of serving the customers. In practice, there are several components to the total cost of serving the customers. The main ones are: the cost of the O/E converters; the cost of terminating equipment at both the O/E node and the customer site; the cost of housing for equipment at placed nodes; the cost of fiber; the cost of copper; and the various installation costs. The terminating equipment is essentially a fixed cost per subscriber, so from the standpoint of our optimization, it is a constant that we can ignore. Placing and sizing the O/E nodes and then assigning customers to be served by those nodes effectively determines the remaining costs. The models that we present in this section and the algorithm described in the next explicitly consider only the costs borne at the nodes. This can include the cost of the O/E converters, the cost of their housing, and installation costs. We can consider the installation cost for an O/E device as part of the cost for an individual unit, and we can add the cost for housing to the cost of the different stacks computed by the knapsack cover. Thus, the cost of a stack can capture a variety of costs incurred at a node.

In many cases, subscribers already have telephone service, so there would already be copper in the network. Fiber, however, may need to be installed. The costs to add fiber may include the cost to add conduit in which to place the fiber and the cost of the fiber itself. Of these, conduit placement is usually the more costly. In our discussions with network specialists, the costs that they were most concerned with were those incurred at the nodes. For this reason, we present our basic models and algorithms with the objective of minimizing the costs incurred at the nodes, but note that the algorithm adapts to handle some of the link costs as well.

We provide an integer programming (IP) formulation for each version of the problem, but we do so mainly to concretely establish the problems that we will address. We do not use IP techniques to solve the problem. To state the problem, we now define the following parameters:

R: the range limit;
N: the set of candidate nodes;
S: the set of customers;
N_s: the set of candidate nodes that are within range limit R of customer s;
M_i: the set of stacks of O/E devices available at candidate node i;
c_{ij}: the cost of equipment stack j for candidate node i; and
q_{ij}: the number (quantity) of subscribers that stack j can serve at candidate node i.

The variables are:

x_{ij}: is 1 if stack j is placed at candidate node i and 0 otherwise; and
z_{si}: is 1 if subscriber s is assigned to node i for service and 0 otherwise.

Now, our first formulation is:

$$\begin{aligned} \text{[BASE]} \qquad & \text{Minimize} \sum_{i\in N}\sum_{j\in M_i} c_{ij}x_{ij} \\ & \text{s.t.} \quad \sum_{i\in N_s} z_{si} = 1, \quad \forall s \in S, \\ & \sum_{s\in S:\, i\in N_s} z_{si} \leqslant \sum_{j\in M_i} q_{ij}x_{ij}, \quad \forall i \in N, \\ & \sum_{j\in M_i} x_{ij} \leqslant 1, \quad \forall i \in N, \\ & z_{si} \in \{0,1\}, \quad \forall s \in S, i \in N_s, \\ & x_{ij} \in \{0,1\}, \quad \forall i \in N, j \in M_i. \end{aligned}$$

The first set of constraints assure that each customer is assigned to some node for service. The second set of constraints guarantee that the capacity placed at each node is sufficient to serve the assigned customers. The third set of constraints assure that at most one stack is placed at any node. This third set of constraints is not really necessary in the usual cases when the cost to serve customers at a node either increases linearly or exhibits economies of scale, but it may be useful computationally. Finally, the range limits are imposed by including the variable z_{si} only if customer s can reach candidate node i.

This problem can be thought of as a variant of the *capacitated concentrator location problem* [10], which is itself a variant of the *capacitated facility location problem* [6] that requires each demand (customer in our case) to be served entirely by one facility. Our problem, however, has a number of important differences from more general problems:

(1) all demands are for one unit;

(2) there are no costs for assigning a customer to a facility;

(3) reach constraints limit the locations at which customer demand may be served;

(4) the nodes (as opposed to the equipment) are effectively uncapacitated;

(5) the underlying network is a tree.

In his thesis, Mazur [12] considers the version of the problem that we have described above (and in [5]) when the stacks available at the candidate nodes are identical. He shows that, when distances are imposed on an underlying network that is more general than a tree, the decision version of the problem is NP-complete, and he conjectures that the tree instance is also NP-complete. If there is only a single type of converter to place at the candidate nodes, then the problem is polynomially solvable by a simple greedy algorithm proposed by Jaeger and Goldberg [8] that extends the previous work of Kariv and Hakimi [9]. A special case in which there is only one size converter whose cost is location dependent is considered by Williams [14]. The survey by Magnanti and Wolsey [11] provides an excellent discussion of a wide variety of design problems on trees.

2.2. *Restricted formulations*

In discussing solutions to the [BASE] problem with access network specialists, we found that our solutions allowed too much freedom and produced solutions that might be technically feasible but would not likely be used in practice. Our solutions allowed the ability to assign customers to service nodes independently, subject to capacity and reach limitations. Thus, there is no preference for serving neighboring customers from the same node. This is not the way network providers typically manage their networks.

Figure 3 provides two illustrations of assignments that might arise from the previous formulation that would likely not be implemented in a real network. Both examples show cases in which the copper wires for customers route through a common site en route to their assigned O/E node. In the first example, the copper for both customers routes over location 1, but neither is served there. In the second example, the copper for both customers routes to node 2, where one customer is served and the other passes by. Our colleagues felt that these types of solutions were impractical for real networks because they would be cumbersome to manage. They felt that if two customers were close enough that their copper wires met en route to their serving locations, then these

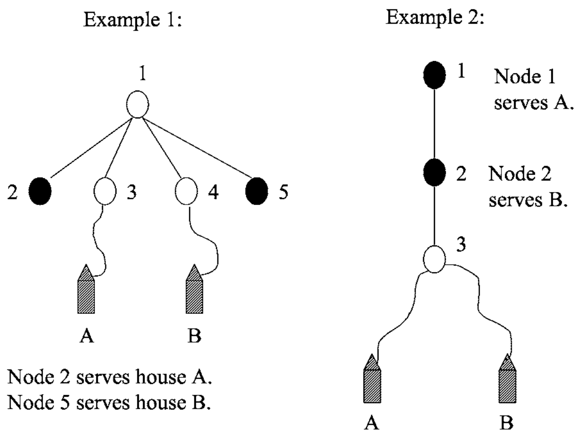

Figure 3. Examples of unrealistic service assignments.

customers should ultimately be served at the same node. For example 1, this would imply that if either customer were served by nodes placed at 1, 2, or 5, then the other must either be served at the same node or its copper wire should not reach location 1. However, assigning house A to a node at 3 or house B to a node at 4 implies no constraint on where the other house is served. In example 2, copper from both customers must go to location 3, so these two customers must always be served at the same node.

When we require that the copper wire between a customer and its O/E node can intersect that of another customer only if they are served at the same node, the solutions obtained tend to be much more orderly. Now, each placed node serves *all* customers homed within an associated subtree of the tree. This feature is exploited in the solution approach that we describe in the next section.

To enforce the proper type of solution, we need to add some constraints on copper intersection to the [BASE] problem. We will begin by letting $P(s,i)$ denote the unique path in the tree that connects customer s to candidate node i. The path $P(s,i)$ can be thought of as being associated with variable z_{si}. The variable effectively considers whether copper should route along the associated path. If we wish to make two z variables positive then we need to verify that their associated paths intersect in an allowable fashion – either not at all, or en route to a common serving node. We can do this by adding the following “wire-crossing” constraints:

$$z_{si} + z_{rj} \leqslant 1 \quad \forall s \neq r, i \neq j\colon\ P(s,i) \cap P(r,j) \neq \emptyset. \tag{1}$$

These constraints say that if two paths intersect, then at most one of the implied assignments can be made. Note that if $s = r$ the constraint is already implied in [BASE] and if $i = j$ the constraint is not required because the paths lead to the same serving node. Adding these constraints to [BASE], the new formulation becomes:

$$\begin{aligned}
\text{[WIRE]}\qquad & \text{Minimize} \sum_{i\in N}\sum_{j\in M_i} c_{ij}x_{ij} \\
& \text{s.t.} \quad \sum_{i\in N_s} z_{si} = 1, \quad \forall s \in S, \\
& \qquad \sum_{s\in S\colon\, i\in N_s} z_{si} \leqslant \sum_{j\in M_i} q_{ij}x_{ij}, \quad \forall i \in N, \\
& \qquad \sum_{j\in M_i} x_{ij} \leqslant 1, \quad \forall i \in N, \\
& \qquad z_{si} + z_{rj} \leqslant 1, \quad \forall s \neq r, i \neq j\colon\ P(s,i) \cap P(r,j) \neq \emptyset, \\
& \qquad z_{si} \in \{0,1\}, \quad \forall s \in S, i \in N_s, \\
& \qquad x_{ij} \in \{0,1\}, \quad \forall i \in N, j \in M_i.
\end{aligned}$$

Both the [BASE] and [WIRE] formulations implicitly assume that there is the flexibility to route copper to any node placed within a subscriber’s range limit. This is not necessarily true, so we also consider a slightly modified third formulation.

Particularly for the longer-reach xDSL technologies, the ability to leverage existing copper is a key consideration for network providers. In order to deploy xDSL quickly and cost-effectively, many providers do not want to install new copper. They want to overlay xDSL on their existing copper network. When this is the case, the set of candidate nodes that could serve each customer is further restricted to those that are already along the subscriber's path to the central office. These are the locations that are already traversed by copper from the CO to the subscriber, and the new network would simply replace the portion of the copper between the O/E node and the CO with fiber. Thus, by restricting the locations at which a subscriber can be served, we can assure that the solution will require no new copper. The candidate nodes in the input network would typically represent cross-connect locations or other locations at which there is convenient access to the existing copper. The formulation for this "overlay" design problem is stated as in [WIRE], but the definition of N_s must be modified as we just described. We do not state the resulting formulation but will refer to it as [OVER].

We note that in the context of the overlay formulation, the wire-crossing constraints have a simpler interpretation. In this version of the problem, the wire-crossing constraint is equivalent to saying that a copper wire will never route across a placed node.

The [WIRE] and [OVER] formulations bear some similarity to the local access network expansion problem considered by Balakrishnan, Magnanti and Wong [3]. (See also [2].) They also consider installing fiber along the path to the CO, but they do so to relieve copper exhaust. To obtain practical solutions, they impose "contiguity constraints" that are similar to our wire-crossing constraints. If we define the *homing node* for a customer to be the candidate node that the customer is incident to in the augmented network, then the contiguity constraints require that all customers with homing nodes on $P(s, i)$ be served at node i when s is served there. This is somewhat less restrictive than the wire-crossing constraint, but similar in its purpose. The problem in [3] focuses on augmenting copper link capacity and installing concentrators between the customer and CO that communicate with the CO over fiber. Thus, they consider link costs and capacities, which we do not. But, we consider placing capacitated facilities at the nodes, and we must further impose range limits.

3. The dynamic programming solution

We now describe the specifics of our algorithm for obtaining optimal solutions for the [WIRE] and [OVER] formulations. We will begin by describing the approach for the [WIRE] formulation and then describe how it can be modified to accommodate the more constrained [OVER] formulation. The input to the algorithm is:

(a) The tree network of candidate nodes, including the distances between them;

(b) The customers, including their homing nodes and the distances to them;

(c) For each candidate node, the stacks of equipment that can be placed there, given by their cost and capacity;

(d) The range limit.

Note that we can always assume that there is a feasible solution to any instance provided to the algorithm because we can scan the tree in a preprocessing step to verify that each customer can reach at least one candidate node. Thus, we assume that the algorithm begins with a feasible instance.

The solution algorithm is essentially a dynamic programming (DP) method. It begins at the leaves of the tree formed by the candidate locations and works toward the central office at its root. The algorithm essentially runs in two passes. The first pass computes the optimal cost to serve the customers, while the second backtracks to recover an optimal solution. Before describing the method, we define a few helpful terms. Throughout this discussion, we will use *node* to imply a candidate node. The *parent* of any node is the first node on the path between it and the root. The *children* of a node are the ones that have it as a parent. The deletion of any link in the tree yields two components. We will refer to the component that contains the root as the *rootward subtree* and the other component as the *leafward subtree*. Similarly, we will refer to the subtree that results from deleting the link between the node and its parent as the rootward subtree from the node. And, we refer to the other component as being the leafward subtree from the node, or simply the node's subtree.

In general, dynamic programming is a method for solving optimization problems that have a large search space but are structured in such a way that it is possible to "grow" a full solution from optimal partial solutions. A common thread in dynamic programming is solving a complex problem by solving a sequence of simple problems. In CBA network design, we begin by solving small problems at the leaves of the tree. At a particular leaf, we must eventually decide whether to serve the customers that home there with an O/E stack at that node or at some other node in the rootward subtree from that node. If we decide to serve the customers at that leaf, then we must place equipment there, so we'd choose the cheapest stack that can serve the subtending customers. If we decide to serve these customers in the rootward subtree, then the cost at the current leaf is zero, but these customers must percolate through the rootward subtree until they are served by equipment at another node. An illustration of how a basic dynamic programming algorithm proceeds on a tree is provided in [11].

Typically, solving a dynamic program involves a certain amount of bookkeeping that is handled by defining a *value function*. The value function maps (in this case) decisions into costs. A key property of the value function is that we can grow solutions by manipulating value functions. At a high level, the value function is evaluated at each node and it tells us how much it will cost to serve the customers in that node's component when they are served in each possible way. We compute the value function for a parent node directly from the value functions of its children.

For the current problem, the value function for a node A is of the form $c(A, B, n)$, where B is another node and n is a number of customers. The precise meaning of $c(A, B, n)$ varies depending upon whether B is in the subtree rootward from A or not. If B is in the subtree rootward from A, then $c(A, B, n)$ is the cost of the cheapest way to serve all of the customers in node A's subtree within A's subtree except for n of them which are served at node B. When B is in node A's subtree then $c(A, B, n)$ is the cost of the cheapest way to serve all of the customers in node A's subtree within this subtree plus n more from the rootward subtree that must all be served at B. In either case, $c(A, B, n)$ represents the total cost of equipment placed within A's subtree under the particular scenario defined by B and n.

Notice that customers in A's subtree may be served in the rootward subtree or rootward customers may be served in A's subtree, but not both. Thus, it may be that A's subtree absorbs customers from the rootward subtree or it may shunt its own customers rootward, but the wire-crossing constraint assures that customers flow only one way across A. Moreover, the constraint assures that all customers crossing A must be served at the same node.

We now describe how to compute the value functions more precisely. To do this, we use the notation defined in section 2.1 plus some additional notation that we now define:

h_s:	the homing node for customer demand s;
d_A:	the number of demands homed to node A;
N_A:	$\bigcap_{s:\, h_s=A} N_s$, which is the set of nodes that can reach every demand homed to node A;
T_A:	the subtree leafward from node A;
$m(i, d)$:	minimum $\{c_{ij}\colon\ j \in M_i,\ q_{ij} \geqslant d\}$, which is the cost of the least expensive stack at node i that can serve d demands.

The value function of the dynamic program, $c(A, B, n)$ is defined as: the cost of serving the demands in T_A within that subtree, if n is zero; the cost of serving these demands within T_A plus serving an additional n demands from outside the subtree at B, if B is inside the subtree; or the cost of serving all but n of these demands within A's subtree and the remainder at B, if B is not in T_A. In this last case we let $c(A, B, n) = \infty$ if the number of demands homed in T_A that can reach B is less than n.

The base cases of the dynamic program correspond to the leaf nodes. If A is a leaf node, there are three simple cases:

(i) If $A \neq B, n = d_A$, and $B \in N_A$, then $c(A, B, n) = 0$.

(ii) If $A = B$, or $n = 0$, then $c(A, B, n) = m(A, d_A + n)$.

(iii) Otherwise, $c(A, B, n) = \infty$.

If A is not a leaf node, then let K_A be the set of children of A. First consider the cases in which n is nonzero. Suppose $B \notin T_A$. We must route n demands from T_A to B. Since these routes must pass through A, wire-crossing constraints require that any demand homed at A must also be routed to B. Thus if $d_A > 0$ but $B \notin N_A$, then $c(A, B, n) = \infty$. Otherwise we can route all of the demand homed at A to B, plus the remaining part of n from the subtrees at the children of A. Thus, $c(A, B, n)$ is obtained by minimizing over all partitions of $n - d_A$ among the children of A. Specifically, let $n(\cdot)$ denote a partition function; then $c(A, B, n)$ is the minimum, over all partitions of $n - d_A$ among K_A, of $\sum_{k \in K_A} c(k, B, n(k))$.

Now, suppose that B is in T_A, but $B \neq A$. In this case, B must lie in T_D for some $D \in K_A$. Because n is nonzero, we must route from outside T_A to B, passing through A. Once again, any demand homed at A must also be routed to B. So, if $d_A > 0$ but $B \notin N_A$, then $c(A, B, n) = \infty$. Otherwise, we can route to B all of the demand homed at A, plus n demands from outside T_A, plus arbitrary demands homed to subtrees at the other children of A. Thus $c(A, B, n)$ is the minimum, over all nonnegative integer functions $n(\cdot)$ on $K_A - D$, of

$$c\left(D, B, n + d_A + \sum_{k \in K_A - D} n(k)\right) + \sum_{k \in K_A - D} c\big(k, B, n(k)\big).$$

We note that in the above we minimize over all attainable integer values for $n(k)$. However, the number of different values that $n(k)$ can attain is certainly bounded by the number of customers in T_k that can reach B.

Next, suppose that $B = A$. Then all of the demand homed to A must be served at A. In addition, the n demands from outside, and arbitrary demands from the rest of T_A will be served at A. Thus, $c(A, B, n)$ is the minimum, over all nonnegative integer functions $n(\cdot)$ on K_A, of

$$m\left(A, n + d_A + \sum_{k \in K_A} n(k)\right) + \sum_{k \in K_A} c\big(k, A, n(k)\big).$$

The case that remains is where n is zero, and the node B is therefore irrelevant. (We, thus, replace it with a place-holder "$\cdot$" in the value function.) To compute the function we must explore two subcases that depend upon whether or not we place equipment at A. When we place equipment at A, it will serve all demand homed at A plus arbitrary amounts of demand homed elsewhere in T_A. If we place equipment at A, $c(A, \cdot, 0)$ would be the minimum, over all nonnegative integer functions $n(\cdot)$ on K_A, of

$$m\left(A, d_A + \sum_{k \in K_A} n(k)\right) + \sum_{k \in K_A} c\big(k, A, n(k)\big).$$

If we place no equipment at A, then all demand in T_A must be served in the subtrees of A's children. In this case, $c(A, \cdot, 0)$ would be the minimum, over all children D of A, over all nodes $B \in T_D$ such that either $B \in N_A$ or $d_A = 0$, over all nonnegative integer functions $n(\cdot)$ on $K_A - D$, of

$$c\left(D, B, d_A + \sum_{k \in K_A - D} n(k)\right) + \sum_{k \in K_A - D} c\big(k, B, n(k)\big).$$

Finally, $c(A, \cdot, 0)$ is the minimum over the two subcases.

The properties we need for a successful dynamic program are:

(a) the value functions at the leaves are easily computed from input data;

(b) the value functions at non-leaves can be computed from those of their children and the input data; and

(c) the value of the optimal solution is easily extracted.

The steps (i)–(iii) above show that the value functions at the leaves are computable as required, and the discussion immediately following (i)–(iii) shows how to compute value functions at non-leaf nodes. The value of the optimal solution is provided directly by the value function at the root. The optimal value is $c(R, \cdot, 0)$, where R is the root node. Now, knowing the components that make up $c(R, \cdot, 0)$, we can work back down the tree. At each node we determine the specific components from its children that led to the optimal decision at the parent node, and we note where equipment is placed.

As a final comment, we would like to mention that the complexity is not as bad as it looks. We can bound the value of a function $n(k)$ in any of the above formulas by the number of demands from T_k that can actually reach the appropriate node. The resulting minimization problems can be solved by relatively obvious dynamic programs, with a number of stages at most the number of children of A, each stage consisting of a number of states at most the number of demands in T_A.

We now present a very small example to illustrate how we perform these functions. Suppose that we may place equipment stacks with the following costs and capacities at nodes A, B, or C:

Cost	Capacity
1000	200
1500	400
2500	600
3000	800

Now suppose that we have nodes A and B that are children of node C and that there are 300 customers homed to A, and 200 homed to B and C as shown in figure 4.

Let X denote any particular node in the rootward subtree from C that all customers homed to A, B, and C can reach. (The figure shows it as C's parent.)

For node A: We must serve either all or none of the houses that home at A at node A. Thus, we either serve all 300 at A, or we serve them in the rootward subtree at

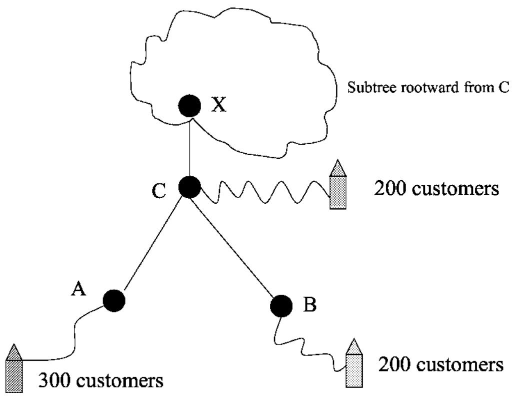

Figure 4. Sample network.

B, C, or X.

$$\begin{aligned}
&c(A, C, 300) = 0 && \text{(the cost at } A \text{ is 0 because we do not need equipment at } A\text{)},\\
&c(A, X, 300) = 0 && \text{(the cost at } A \text{ is 0 because we do not need equipment at } A\text{)},\\
&c(A, B, 300) = 0 && \text{(the cost at } A \text{ is 0 because we do not need equipment at } A\text{)},\\
&c(A, \cdot, 0) = 1500 && \text{(we serve } A\text{'s demand at } A \text{ and need equipment at } A\text{)},\\
&c(A, A, 200) = 2500 && \text{(we serve } C\text{'s demand at } A\text{)},\\
&c(A, A, 400) = 3000 && \text{(we serve all demand at } A\text{)}.
\end{aligned}$$

The case at node B is analogous:

$$\begin{aligned}
&c(B, C, 200) = 0,\\
&c(B, X, 200) = 0,\\
&c(B, A, 200) = 0,\\
&c(B, \cdot, 0) = 1000,\\
&c(B, B, 200) = 1500,\\
&c(B, B, 500) = 3000.
\end{aligned}$$

Now, the value function at node C combines information from its children.

$$\begin{aligned}
&c(C, X, 700) = 0 && \text{(the cost at } C \text{ is 0 because customers are served at } X\text{)},\\
&c(C, X, 500) = 1000 && \text{(this is } c(B, \cdot, 0) + c(A, X, 300)\text{)},\\
&c(C, X, 400) = 1500 && \text{(this is } c(A, \cdot, 0) + c(B, X, 200)\text{)},\\
&c(C, X, 200) = 2500 && \text{(this is } c(A, \cdot, 0) + c(B, \cdot, 0)\text{)},\\
&c(C, \cdot, 0) = 3000.
\end{aligned}$$

We will now describe how to compute $c(C, \cdot, 0)$ in detail. Determining $c(C, \cdot, 0)$ requires taking the minimum over two cases that correspond to whether or not we place equipment at node C. First consider the case in which we place equipment at C. If equipment is placed at C, then customers homed to A and B must be served at or below C. This yields four subcases corresponding to the different customer partitions possible when equipment is placed at C:

(a) Only customers homed at C are served at C:
$n(A) = 0$, $n(B) = 0$ and the cost is

$$m(C, 200) + c(A, C, 0) + c(B, C, 0) = 1000 + 1500 + 1000 = 3500.$$

(b) Customers homed at B and C are served at C:
$n(A) = 0$, $n(B) = 200$ and the cost is

$$m(C, 400) + c(A, C, 0) + c(B, C, 200) = 1500 + 1500 + 0 = 3000.$$

(c) Customers homed at A and C are served at C:
$n(A) = 300$, $n(B) = 0$ and the cost is

$$m(C, 500) + c(A, C, 300) + c(B, C, 0) = 2500 + 0 + 1000 = 3500.$$

(d) All customers are served at C:
$n(A) = 300$, $n(B) = 200$ and the cost is

$$m(C, 700) + c(A, C, 300) + c(B, C, 200) = 3000 + 0 + 0 = 3000.$$

Next, consider the case in which we do not place equipment at C. Therefore, we must serve C's customers at either A or at B. If we serve them at A, we get two subcases depending upon whether or not we also serve B's customers at A:

(e) B's customers are served at B:
$n(B) = 0$ and the cost is $c(A, A, 200) + c(B, A, 0) = 2500 + 1000 = 3500$.

(f) B's customers are also served at A:
$n(B) = 200$ and the cost is $c(A, A, 400) + c(B, A, 200) = 3000 + 0 = 3000$.

If we serve C's customers at B, we get two more subcases that are analogous to the previous ones:

(g) A's customers are served at A:
$n(A) = 0$ and the cost is $c(B, B, 200) + c(A, B, 0) = 1500 + 1500 = 3000$.

(h) A's customers are also served at B:
$n(A) = 300$ and the cost is $c(B, B, 500) + c(A, B, 300) = 3000 + 0 = 3000$.

We obtain $c(C, \cdot, 0)$ by minimizing over these eight subcases. If C were in fact the root, we would see that the least cost to serve the given set of customers is \$3000, which is obtained in five possible ways.

Now, to handle the [OVER] formulation, we require very little change to the method described for the [WIRE] formulation. Essentially, we need to modify the reachable set, N_s, of candidate nodes for each customer s. This can be accomplished either by manipulating the sets directly or by altering the distances so that we can continue to use distance to determine reachability. We do the latter by replacing the undirected links forming the tree with two directed links. The link from a node to its parent has the same length as the original link; whereas, the link from a parent to its child has a length that exceeds the range limit. The long length of the leafward link prevents routing in any direction except toward the root.

4. Computational comparisons

In this section, we present computational comparisons on two network models based on actual geographic areas.[1] The first is based on a network in Iowa where FTTC deployment was under study; the second is derived from a network in Colorado and is being used to demonstrate ADSL network planning [4]. In our tests, we consider placing

[1] Precise data for real networks is scarce because it is generally considered proprietary by its owners.

Table 1
Equipment models used for Colorado example (xDSL).

Capacity (# of subscribers)	Cost ($)
20	15,500
36	26,500
52	36,000
68	45,500

Table 2
Equipment models for Iowa example (FTTC).

Capacity (# of subscribers)	Cost ($)
4	796
8	1,026
16	1,441
32	2,081

smaller, cheaper O/E devices characteristic of FTTC in Iowa and larger, more expensive ones characteristic of ADSL in Colorado. The specific O/E equipment that we use is given in tables 1 and 2. We believe these data to be realistic, but not real.

In performing these tests we focus on two things:

(1) the solution time;

(2) the effect of the formulation on the cost of the solution.

We present solution times to illustrate that the algorithm seems to run quickly on relatively large problems. This is critical for use in an interactive planning process, such as described in [4]. We envision the algorithm for CBA network design being used to plan the area served by one CO (which is called a *wire center*) at a time. A typical wire center might have on the order of twenty thousand customers, but of those, a relatively small fraction would be interested in subscribing to high-speed services like xDSL. Thus, the number of subscribers in a typical wire center is likely to be on the order of 2000. The near-term demand for higher-speed access with FTTC is likely to be even less. In the networks that we consider as test cases, Colorado has 1619 customers and 393 candidate nodes, while Iowa has 609 customers and 218 candidate nodes. A schematic illustration of the Iowa network is given in figure 5. The figure shows the central office (labeled FDI in the figure) and the remaining candidate nodes but not the customers.

Since two customers with the same homing node must ultimately be served at the same placed node, the DP algorithm can effectively collapse them into one. Thus, the number of distinct homing nodes may be a better indicator of the problem size for the DP than the specific number of customers. (However, the number of customers is certainly relevant because the pre-processing to determine equipment stacks depends on the number of customers.) The Colorado data contains 233 customers with distinct homing nodes and the Iowa data contains 148. In both cases, this is considerably smaller

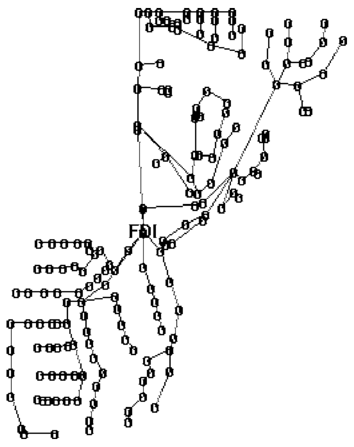

Figure 5. Iowa network.

Table 3
Results for Colorado network.

Formulation	Cost ($)	% gap	Placed nodes	Time (s)
[BASE]	1,141,500[2]	0.53	35	N/A
[WIRE]	1,166,500	2.73	29	1.1
[OVER]	1,258,500	10.83	41	0.9

Table 4
Results for Iowa network.

Formulation	Cost ($)	% gap	Placed nodes	Time (s)
[BASE]	81,064	0.0	69	N/A
[WIRE]	81,064	0.0	69	0.2
[OVER]	90,148	11.21	83	0.2

than the number of customers, which illustrates how the algorithm can use the wire-crossing constraint to its advantage.

Results for the two networks are given in tables 3 and 4. The times provided are the user times reported by the time command running on a sun4m machine. The times reported include all processing to solve the problem along with input and output. Thus, these times include both the time spent in the core DP algorithm and all preprocessing to compute equipment stacks and distances and reachability.

To solve the [BASE] version of the problem, we apply a combinatorial search algorithm that is briefly described in [5]. This algorithm constructs both a lower bound on the cost of the network design and an upper bound provided by the best feasible solution it identifies. For the Colorado problem, we allow the algorithm to run for 5000

[2] This is the best feasible solution identified, but could not be confirmed optimal.

seconds. In this time, the solution it finds is within 1% of the lower bound (\$1,135,500), but cannot be confirmed optimal.[3] The [BASE] problem for Iowa is easily solved to optimality by the combinatorial method. The percent gap provided in the tables represents the percentage by which the obtained solution differs from a lower bound on the solution cost for the [BASE] formulation. This lower bound is either the optimal cost of the [BASE] solution or a lower bound thereon. For the [BASE] solution obtained, the gap is a measure of suboptimality. For the [WIRE] and [OVER] formulations we are guaranteed to find the corresponding optimal solution by the DP algorithm, so this value provides a comparison among formulations. It can be viewed as a bound on the extra cost incurred by using the more restrictive formulations. Note that this is an entirely different interpretation of what the gap represents.

Our reason for solving the [BASE] problem is primarily to provide a comparison on solution cost. The two methods that restrict wire crossing are more constrained, so the resulting designs will cost at least as much as that of the [BASE] design. The difference in cost between the solution for the [BASE] problem and that of the [WIRE] formulation gives us a measure of how much we might need to pay to build a more maintainable network. The difference in cost between solutions for the two more restricted formulations might provide a measure of the additional cost of networking equipment that one must pay for the ability to reuse existing copper. If this is more than the cost of installing new copper, a planner may wish to reconsider the overlay restriction. For both of our test cases, the increase in cost to enforce the wire crossing restrictions imposed for easier maintenance is relatively modest. For Iowa there is no increase in cost. For Colorado, the cost is within 3% of the cost of the lower bound.

The additional equipment cost incurred to enforce copper overlay is somewhat larger for these two examples. In both cases it is approximately 11% over the lower bound on equipment cost.

In light of the above results, and since the [BASE] version of the problem appears to be difficult [12], it is compelling to consider employing an algorithm that solves [WIRE] as an approximation for [BASE]. If we do this, a natural question becomes: how bad can this approximation be? In the appendix, we demonstrate that an optimal design under the [WIRE] formulation can cost as much as $|S|$ times that of the [BASE] formulation, where $|S|$ is the number of customers and where we assume that the cost to serve k customers at a node is no more than k times the cost to serve one customer there. When costs at the candidate nodes are more arbitrary (as when there are diseconomies of scale) the comparison can be arbitrarily bad. Not surprisingly, examples that actually achieve the worst-case bounds are highly contrived.

4.1. Results for random problems

To augment our results from realistic networks, we also consider five randomly generated problems. These problems each contain 5000 customers and 1000 candidate nodes.

[3] These are tighter bounds than we were able to obtain applying the CPLEX® MIP solver.

Table 5
Results for random network 1.

Formulation	Cost ($)	% gap	Placed nodes	Time (s)
[BASE]	401,365	0.07	280	N/A
[WIRE]	470,796	17.4	297	8.4
[OVER]	499,329	24.5	324	7.3
Lower bound	401,085			

Table 6
Results for random network 2.

Formulation	Cost ($)	% gap	Placed nodes	Time (s)
[BASE]	395,050	0.06	270	N/A
[WIRE]	466,853	18.2	292	8.5
[OVER]	490,227	24.2	315	7.1
Lower bound	394,820			

Table 7
Results for random network 3.

Formulation	Cost ($)	% gap	Placed nodes	Time (s)
[BASE]	390,813	0.07	263	N/A
[WIRE]	453,561	16.1	278	7.4
[OVER]	479,043	22.7	308	7.2
Lower bound	390,533			

The problems are created as follows. The candidate nodes are assigned a unique identification number between 1 and 1000, and the tree structure is created by randomly selecting a candidate's parent from among the lower-numbered candidate nodes. Each customer's homing node is randomly selected from among the candidate nodes. (This resulted in at least 991 customers with distinct homing nodes in each example.) Distances between candidate nodes are randomly generated integers between 1 and 200. The distance between a customer and its parent is a randomly generated integer in the range [0, 900]. For these examples, we apply a maximum range limit of 1000.

We perform the tests using the equipment models shown in table 2. Once again, we run the combinatorial algorithm to obtain bounds for the [BASE] formulation, allowing it to run for up to 5000 seconds. The results for the random problems are shown in tables 5–9.

The algorithm for the [BASE] formulation is unable to confirm optimality for the first three problems but reaches optimality for the last two. We observe that the DP algorithm solves these problems quickly, but the cost for implementing a more manageable solution appears higher in the random examples. We surmise that this is likely because of the natural "clustering" that occurs in real networks is absent in the random ones.

Table 8
Results for random network 4.

Formulation	Cost ($)	% gap	Placed nodes	Time (s)
[BASE]	393,077	0.0	267	N/A
[WIRE]	456,995	16.3	285	7.6
[OVER]	487,126	23.9	315	6.2
Lower bound	393,077			

Table 9
Results for random network 5.

Formulation	Cost ($)	% gap	Placed nodes	Time (s)
[BASE]	393,077	0.0	267	N/A
[WIRE]	463,500	17.9	281	7.4
[OVER]	488,666	24.3	311	6.3
Lower bound	393,077			

4.1.1. A more detailed study on customers' reach

We now conduct a more detailed study using random network 1. In the preceding trials, the range limit is 1000 and the distance between a customer and its homing node is an independent randomly generated number in the interval $[0, 900]$. Thus, the *reach* of each customer beyond its homing node is an independent random number in the interval [100, 1000]. We now conduct a more detailed study on the effect of customer reach on both the solution time of our algorithm and on the characteristics of solutions generated. We note that when reaches beyond the homing node are short, the solutions are more constrained and more costly, and when reaches are long, solutions are less constrained and less costly. More intuitively, when reaches are long, solutions will try to take advantage of economies of scale by placing fewer nodes that serve more customers. When reaches are short, we have no choice but to place more nodes that typically serve fewer customers.

In the next set of experiments, we vary customers' reach and observe the effects on both the solution and the running time of the algorithm. We conduct a sequence of 10 tests corresponding to reaches from 100 to 1000 in steps of 100. For each test, every customer's reach past its homing node is fixed to a common value and a solution is generated. The results for the [BASE], [WIRE], and [OVER] formulations are summarized in tables 10–12. Table 10 includes data for reaches only up to 400. When each customer has a reach of 400 past its homing node, the cost of the optimal solution for the [BASE] formulation is $325,662, which is equal to the cost to serve 5000 customers when there are no range limits at all. (This is the cost of placing 156 size-32 O/E converters and a single size-8 converter, which is the cheapest way to serve 5000 customers at a single location.) Thus, the cost will not decrease further as the reach increases.

Most of the results produced by the dynamic program are consistent with intuition. As the reach increases, the cost and the number of nodes placed decrease. As the reach

Table 10
Results varying reach for BASE formulation.

Reach	Cost ($)	Cost lower bound	% gap	Placed nodes
100	675413	675413	0.0	586
200	413042	407307	1.41	266
300	343441	338948	1.33	176
400	325662	325662	0.0	118

Table 11
Results varying reach for WIRE formulation.

Reach	Cost ($)	% gap	Placed nodes	Time (s)
100	684134	1.29	591	5.9
200	452484	11.09	269	7.0
300	400050	18.03	194	8.8
400	360999	10.85	126	14.0
500	345737	6.16	88	18.2
600	337868	3.75	66	36.5
700	333087	2.28	47	69.4
800	331035	1.65	43	128.5
900	328539	0.88	30	217.1
1000	326717	0.32	17	326.8

Table 12
Results varying reach for OVER formulation.

Reach	Cost ($)	% gap	Placed nodes	Time (s)
100	702604	4.03	613	5.9
200	484521	18.96	306	5.7
300	413705	22.06	212	6.5
400	370886	13.89	139	6.8
500	351580	7.96	99	7.1
600	341781	4.95	71	9.4
700	337156	3.53	59	10.2
800	333169	2.31	48	9.6
900	329184	1.08	35	9.9
1000	327484	0.56	18	9.3

increases, so too does the number of choices available to the DP. This manifests itself in increasing solution times. With the [OVER] formulation, the solution times seem to level off, but this is not the case with the less-constrained [WIRE] formulation. While it is not surprising that the impact on solution time is greater with the [WIRE] formulation than it is with the [OVER] formulation, the magnitude of the difference when the reaches are long is somewhat surprising. This observation may prove useful in a practical setting because solutions generated by the more constrained formulation do not cost much more when the reaches are long.

5. Design problem extensions

There are a number of additional issues that lead to extensions to the basic algorithm that we have described. Some of these additional issues do not require changes to the core algorithm described in section 3, while others do. In this section, we consider four variations that have arisen during the time that we have studied CBA network design. All of these can be addressed by applying the basic DP algorithm to an altered network provided as input. We present them roughly in order of their practical motivation. Throughout this section, we focus on input adaptations for the DP algorithm to handle the modified problem; therefore, we restrict our discussion to only the [WIRE] and [OVER] formulations.

5.1. Network augmentation

One of the primary practical considerations is whether we can augment an existing network. In such a situation, we may have existing O/E converters with spare capacity already placed in the network, and we may have existing customers already assigned to O/E nodes. Expanding an existing network can be accomplished with some preprocessing but no fundamental change to the DP. The two issues we must address are how to incorporate existing spare capacity at the nodes and whether the wire-crossing constraints apply between new and existing subscribers. Handling spare capacity is accomplished as a post-processor to computing the O/E stacks. In the simplest case, we can assume that the existing spare capacity is free. If a node has spare capacity p, then we add one converter with cost 0 and capacity p and we add p to the capacity of all stacks constructed for this node.

If there are no wire-crossing constraints imposed between new and old subscribers, then we can simply ignore the existing subscribers and proceed. If the wire-crossing constraints must be enforced between all customers (both new and old), then we can modify the input to assure a correct solution. We need to make sure that new nodes are placed only in allowable regions of the network and that new customers “respect” the existing copper wires. We can view an existing solution as a set of non-overlapping connected subgraphs overlaying the network. Each subgraph contains exactly one O/E node and corresponds to the part of the network traversed by the copper wires of subscribers served by this O/E node. We call these subgraphs the *domain* of the O/E node contained within it. The first part of figure 6 illustrates an O/E node with its surrounding domain and a set of new customers that desire service.

To strictly enforce the wire-crossing constraints, we must assure that any new subscriber whose wire enters this domain is served at the existing node. We can do this by replacing the entire domain with the single O/E node. Any structure (customer or node) outside of the domain that had formerly had a link into it is now directly connected to the O/E node. To guarantee a proper solution, we need to invoke the bidirectional representation of the tree described in section 3. The length of the link to O/E node is the length of the corresponding path in the original network, and the length of any link exiting the

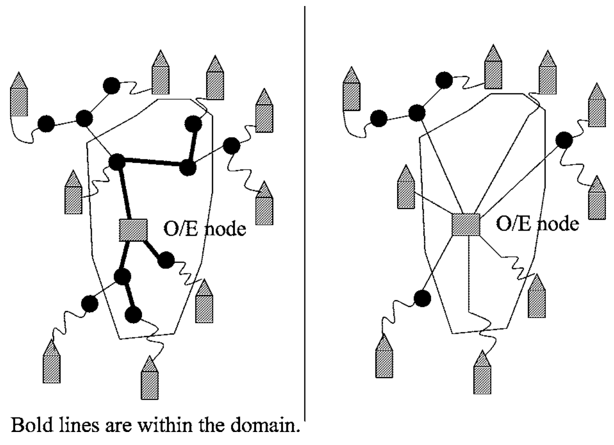

Figure 6. Example of shrinking the domain of an O/E node.

O/E node will be greater than the allowable range. This transformation is shown in the second part of figure 6.

We note that any new customer homing to a location within an existing domain is served at the domain's O/E node if it can reach it; otherwise, it cannot be served. In practice, somewhat looser wire-crossing requirements might be enforced between new and old customers. Such constraints might allow placement of new equipment to serve customers that cannot reach the domain's O/E node but might still forbid passing through a domain without being served there.

5.2. Node capacities

So far we have considered the case where the equipment is capacitated but the sites themselves are uncapacitated. We may also consider applying two different types of capacity restrictions at a node. There may be a limit on the number of customers that can be served at a node, or there may be a limit on the number of O/E converters that can be placed at a node. The latter restriction might capture space limitations in equipment housings. Site-based capacity restrictions constrain the choices when we preprocess to generate equipment stacks, and they affect problem feasibility.

From the standpoint of "stack creation", the limit on the number of customers that may be served at a node is easy to handle: we simply stop building stacks when we have optimally covered the smaller of the customer limit and the number of customers within its reach. The introduction of limits on the number of converters that may be placed at a node is a bit more complicated. The new limit makes the knapsack cover problem at the candidate node more like a two-dimensional knapsack in which there are limits on dimensions corresponding to both customers and equipment. In practice, our experience yields intuition that may help to mitigate this added complexity. We rarely observe solution where more than two or three converters are placed at a node. Thus, if the limit on the number of converters is large, then it is likely not to be violated. Alternatively, when the limit is small, stack building can probably consider the possibilities exhaustively.

With node capacities in place, it may be the case that some customers cannot be served even though they can reach a candidate node. We can address feasibility issues as part of either pre-processing or post-processing. If we handle it in post-processing then we alter the problem to assure a feasible solution but are able to identify and remove infeasibilities in the solution. We can assure a feasible problem by inserting a new "fake candidate node" between each customer and its homing node. At this new candidate node, there will be one available equipment stack. It has larger cost than any real converter and capacity equal to 1. The distances to and from the new node can be set to assure a feasible solution without altering the total distance to any real candidate node. This effectively applies a "Big-M" type approach to gain feasibility. Any customer that can be served feasibly is served at a real node and those that cannot are served at the new fake nodes.

For certain special cases, it is probably more straightforward to pre-process to remove infeasible customers prior to invoking the DP. One popular special case of node capacities is where equipment may be prohibited at a "candidate" node. Such nodes have an "empty" equipment stack provided to the DP as input, and the DP serves no customers at these locations. When candidate nodes either have infinite capacity or no capacity, we assure feasibility by simply checking that every customer can reach a candidate node where equipment is available and removing those that cannot.

5.3. Profit maximization

As telecommunications providers begin to offer services that are not regulated, they will become more interested in the profitability of their network. To this end, suppose that our objective were to maximize the profit over a given period. Assume that we can associate a common value for the expected revenue with each customer in the planning area. (This might be the case when there is a subscription service like ADSL being offered.) With another simple data pre-processing step, we can alter the input to the DP to "trick" it into maximizing profit. The trick is to insert a new, hypothetical candidate node between each potential customer and its homing node. At this new candidate node, there will be one available equipment stack. It has cost equal to the expected revenue of the incident customer and capacity equal to 1. The distance from the customer to this new node can be any positive distance smaller than the original distance to its homing node. The distance between the new node and the original home pole will be the difference between the original distance and the amount applied between the new node and the customer.

When we apply the DP algorithm to this altered problem, customers assigned to non-hypothetical nodes are profitably served with the equipment placed at those nodes. Other customers are not profitable to serve in the implied solution. However, planners may still choose to assign these customers to actual O/E nodes at additional cost.

We are currently exploring more general profit-based formulations to see how easily they can also be accommodated within the context of the DP algorithm.

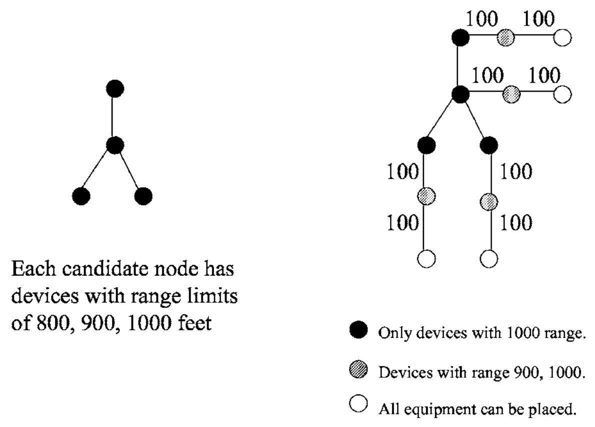
Each candidate node has devices with range limits of 800, 900, 1000 feet
100
100
100
100
100
100
100
100
Only devices with 1000 range.
Devices with range 900, 1000.
All equipment can be placed.

Figure 7. Transformation to allow devices with differing range limits.

5.4. *Equipment-based range limits*

So far, we have only considered the case in which there is a constant range limit characteristic of a particular type of CBA technology. Although we know of no specific examples, it is conceivable that different types of equipment for a particular problem instance may have different signal propagation properties that might result in equipment-specific range limits. Thus, if we have two different O/E devices for FTTC, one might be able to reach customers within 1000 feet, while the other can reach only 800 feet.

Such a situation is also easily handled by a modification to the problem input data. Consider a given problem has available O/E devices with n different range limits. For simplicity, we will illustrate the transformation assuming that the same equipment is available at all candidate nodes, and discuss the more general case afterward. First, we replace every candidate node with n candidate nodes that are connected to each other by a path. Figure 7 provides an example in which there are O/E devices available at a candidate node having reaches of 800, 900, and 1000. One of the locations in the path occupies the same position as the original candidate node in the tree, and the rest of the path is accessible only through this location. The equipment stacks available at the first candidate node include only the devices with the longest range. Now, moving along the path, each candidate node is associated with a progressively shorter range limit. The distance between two candidate nodes in the path is the difference between their associated range limits. The stack at any candidate node may use devices with the associated range limit or longer ones. To implement the [OVER] formulation we use the distances along the path as described, but we do not make distances away from the root along this path prohibitively long as we usually do.

We now apply the DP algorithm to this altered problem using the longest of the limits for its range limit parameter. Equipment placed along a path in the altered problem is placed at the associated site in the real network. By making shorter-reach equipment available only at a distance from the true candidate node, we assure that it is reachable in the real solution.

To allow a more general instance in which different candidate nodes may have equipment of different reaches, we employ the same basic approach. The range limit that the DP will use is the longest range for any piece of equipment. This will be the range associated with the closest candidate node to the root in every path. However, the associated candidate node may or may not have available equipment with this reach. If it does not, this candidate is associated with an empty stack of equipment, as described in section 5.2. Otherwise, it is as above. The remaining candidate nodes along the path are associated with the remaining allowable ranges for the equipment that can be placed at the true candidate node.

6. Concluding remarks

The version of the algorithm that is described in section 3 and tested in section 4 is implemented as the design engine for Telcordia's™ Network Planner – a prototype software tool for xDSL network planning over an existing copper network. Since we are planning over an existing network, the tool implements the [OVER] formulation. It already allows consideration of several features described in the previous section and is being enhanced to include others. Within this tool, the placement and sizing algorithm is combined with demand forecasting for a more comprehensive planning tool. An overview of this broader context in which the node placement and sizing problem is solved is given in [4].

In an idealized planning context, we can assume that distance accurately captures the signal degradation in the copper portion of the network. To apply the algorithm within Network Planner or to extend its applicability to a more realistic engineering setting, we need to model the capabilities of the copper plant with more detail. To date, we have encountered a number of "real world" issues that we have incorporated into the algorithm to varying degrees. Some of the issues that have arisen are different cable gauges, and the presence of things that interfere with xDSL delivery like DLC systems, load coils, bridged taps, poor quality copper, and interference from other wires within a binder group[4] [1,7]. (These latter issues arise in the process of loop qualification.) Several of these are handled by simple modifications to the input to the DP algorithm, but others need to be better understood within the context of our algorithm to determine whether or not we can (or should) handle them.

Variations on profit maximization pose interesting extensions. We have described the simple case in which all subscribers yield equal revenue and our objective becomes one of profit maximization. Rather than just maximizing profit, we expect that a question that network providers might ask when breaking into emerging markets like xDSL is: given that we have $\$X$ to spend, which areas should we serve? This appears to be somewhat harder than the simple profit maximization described previously.

To date, the strongest interest in this algorithm is for expanding ADSL network coverage beyond the houses that can be reached directly from the CO. Placing O/E de-

[4] A binder group is a group of wires that are bundled together in large cables.

vices deeper into the network can enable service to all houses within a CO's area. As demand for bandwidth increases, we expect that demand for more fiber-rich technologies, like FTTC or FTTH, will also increase. The algorithm that we have described remains applicable until the point that the fiber reaches the home.

Acknowledgments

The authors are grateful to Ron Menendez, Mark Seely, and Frank Effenberger for many helpful discussions on the design and planning of broadband access networks. We are also grateful to the Network Planner team – Cliff Behrens, Chris Dworkin, Yu-Yun Ho, George Seymour, Vijay Sreedhar, Gary Truax, Rochelle Wachtel, and Adrian Wrixon – for their part in providing a "home" for this work, and we are especially grateful to Hanan Luss for his helpful suggestions and interest in this work. Finally, we appreciate the thoughtful comments and suggestions for elaboration made by one of the referees.

Appendix

We now address the relationship between the [WIRE] and [BASE] formulations in more detail. In particular, we consider the worst-case performance guarantees that can be made if we use an optimal solution for [WIRE] as an approximation to the optimal solution for [BASE]. When we assume that the equipment costs at the candidate nodes are such that the cost to serve k customers a site is no more than k times the cost to serve one, we can guarantee that the optimal value of the [WIRE] version of the problem is no more than $|S|$ times the optimal value of the [BASE] problem. ($|S|$ is the number of customers.) Thus, solving the [WIRE] version of the problem provides a feasible solution for [BASE] that costs no more than $|S|$ times optimal.

To illustrate, we will demonstrate that we can take any feasible solution for the [BASE] problem and "uncross" the wires to obtain a feasible solution for the [WIRE] problem that costs no more than $|S|$ times more. So, in particular, uncrossing the optimal solution to the [BASE] problem yields a feasible solution to [WIRE] that costs no more than $|S|$ times more. Thus, the optimal solution for [WIRE] is certainly within this bound.

Assume that we have a feasible solution to the [BASE] problem that has cost C. If it is not feasible for [WIRE] then it has wires destined for different O/E nodes overlapping someplace along their routes. We will call candidate or placed nodes that are overlapped by wires terminating at different O/E nodes *crossing points*. We can remove all crossing points by rerouting each wire to go from the customer to the *closest* of the placed O/E nodes. If there are ties in distance, we can associate a unique number or name with each node and use them to break ties lexicographically.

Clearly, each customer is within range of its new O/E node. To see that this routing includes no crossing points, observe that a crossing point requires there to be some wire w_1 overlapping another wire w_2 at a location y, and also requires these wires to route to

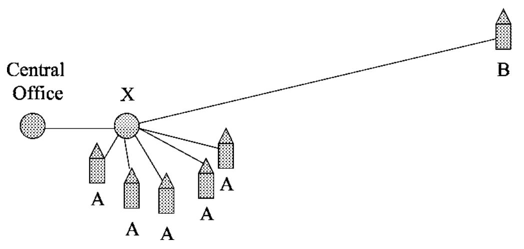

Figure 8. Worst-case example for approximating [BASE] with [WIRE].

different O/E nodes. Since the closest location to y is uniquely determined, a crossing point cannot occur because if it did, one of the wires could be “shortened” by routing to the other O/E node. Thus, such situations do not occur.

The uncrossing procedure creates no new O/E nodes. It merely shuffles the customers among previously-existing O/E nodes. To build a feasible solution from the routing obtained, we may need to add capacity at the O/E nodes. Let C_i denote the cost of the equipment at placed node i under the [BASE] solution. Now, notice that every O/E node serves at least one customer in the [BASE] solution and serves no more than $|S|$ customers in the new arrangement. Thus, our rearrangement will incur a cost of no more than $|S|C_i$ at any node, and its total cost is no more than $|S|C$. Therefore, we are assured that there exists a feasible solution to [WIRE] whose cost is no more than $|S|C$.

Although this appears to be a crude cost bound, a simple example shows that it is tight. Suppose that we have the network illustrated in figure 8. Here we have a cluster of $|S| - 1$ houses labeled A that can reach candidate locations at X and the central office. We also have a single house labeled B that can reach only X. Now, suppose that O/E converters available at the CO are free and have sufficient capacity to serve all $|S|$ customers. Further, suppose that there is one type of converter available at X and it has capacity equal to 1 and cost C. The optimal solution to the [BASE] problem serves $|S| - 1$ houses at the CO and one at X, for a total cost of C. The optimal solution to the [WIRE] problem must serve all customers at X, incurring a cost of $|S|C$ and demonstrating that the worst-case bound is tight.

If we were now to remove the assumption that the cost to serve k customers at a node is no more than k times the cost to serve one, then it is easy to see that the [WIRE] solution can cost arbitrarily more than that of [BASE]. To see this, we can use the above example but make only two stacks available at X. One has cost C and capacity equal to 1, while the other has capacity $|S|$ and an arbitrarily high cost.

Although our description of the algorithm allows different stacks of O/E devices to be available at each node, the results in section 4 are generated assuming that the same devices are available at each node for the same cost. When this is the case, the worst-case relationship between [BASE] and [WIRE] improves. If the optimal value of the [BASE] problem is C, then the value provided by [WIRE] is within $O(\sqrt{|S|})C$.

If we are given a solution to [BASE] with cost C, we can uncross it as described previously. Let S_i denote the set of customers whose wires route to a node i in the uncrossed solution. Now, let us partition the nodes into two sets:

$$I = \{i\colon |S_i| \leqslant \sqrt{|S|}\}, \qquad J = \{i\colon |S_i| > \sqrt{|S|}\}.$$

We can provide enough capacity for customers served at nodes in I, by multiplying the previous cost at each of these nodes by $|S_i|$, for a cost of no more than $\sqrt{|S|}C$. Now, we can satisfy the customers assigned to nodes in J by placing a stack of equipment that includes all of the O/E devices in the entire [BASE] solution at each node in J. There are fewer than $\sqrt{|S|}$ nodes in J, so the cost at these nodes is no more than $\sqrt{|S|}C$. In total, the cost can be no more than $2\sqrt{|S|}C$. Thus, we know that the bound is no worse than $\mathrm{O}(\sqrt{|S|})C$.

Now, consider an example that expands on the one in figure 8. Let there be K branches off of the CO that each look like the branch in figure 8, so that the network of candidate nodes is a star. Thus, the candidate nodes are the CO and the K leaves, which we denote $X_1, \ldots, X_K$. Each node X_i is the homing node for K customers, $K - 1$ of which reach the CO and one that cannot. Now, suppose that there are two O/E devices available at each candidate node, and we can use them to construct stacks that combine them as desired. One device has capacity equal to 1 and cost 1; the other has capacity $K(K-1)$ and cost K. The optimal solution for this instance of the [BASE] problem has cost $2K$. The optimal solution for the [WIRE] formulation has cost K^2. Thus, we have a ratio of $K/2$, which is $\sqrt{|S|}/2$.

References

[1] G. Abe, *Residential Broadband* (Macmillan, Indianapolis, IN, 1997).

[2] A. Balakrishnan, T. Magnanti, A. Shulman and R. Wong, Models for planning capacity expansion in local access telecommunication networks, Annals of Operations Research 33 (1991) 239–284.

[3] A. Balakrishnan, T. Magnanti and R. Wong, A decomposition algorithm for local access network expansion planning, Operations Research 43 (1995) 58–76.

[4] C. Behrens, T. Carpenter, M. Eiger, Y. Ho, H. Luss, G. Seymour, P. Seymour and G. Truax, Network planning for xDSL, in: *Proceedings of the 16th Annual National Fiber Optic Engineers Conference*, Denver, CO (2000).

[5] T. Carpenter, M. Eiger, P. Seymour and D. Shallcross, Automated design of fiber-to-the-curb and hybrid fiber-coax access networks, in: *Proceedings of the 12th Annual National Fiber Optic Engineers Conference*, Denver, CO (1996).

[6] G. Cornuejols, R. Sridharan and J.-M. Thizy, A comparison of heuristics and relaxations for the capacitated plant location problem, European Journal of Operational Research 50 (1991) 280–297.

[7] S. Cray, E. Gallo and R. Warner, Realizing the potential of DSL, Telcordia Exchange Magazine 2 (2000) 16–21.

[8] M. Jaeger and J. Goldberg, A polynomial algorithm for the equal capacity p-center problem on trees, Transportation Science 28 (1994) 167–175.

[9] O. Kariv and S. Hakimi, An algorithmic approach to network location problems. I: The p-centers, SIAM Journal of Applied Mathematics 37 (1979) 513–538.

[10] J. Klincewicz and H. Luss, A Lagrangian relaxation heuristic for capacitated facility location with single-source constraints, Journal of the Operational Research Society 37 (1986) 495–500.

[11] T. Magnanti and L. Wolsey, Optimal trees, in: *Network Routing*, eds. M. Ball, T. Magnanti, C. Monma and G. Nemhauser, Handbooks in Operations Research and Management Science, Vol. 7 (North-Holland, Amsterdam, 1995) chapter 9.

[12] D. Mazur, Integer programming approaches to a multi-facility location problem, Ph.D. thesis, Johns Hopkins University, Baltimore, MD (1999).

[13] C. ReVelle, Facility siting and integer-friendly programming, European Journal of Operational Research 65 (1993) 147–158.

[14] K. Williams, Families of subtrees and the optical network unit placement problem, B.A. thesis, Princeton University, Princeton, NJ (1997).