Primal-Dual Algorithms for Connected Facility Location Problems*

Chaitanya Swamy† Amit Kumar‡

Abstract

We consider the *Connected Facility Location* problem. We are given a graph $G = (V, E)$ with costs $\{c_e\}$ on the edges, a set of facilities $\mathcal{F} \subseteq V$, and a set of clients $\mathcal{D} \subseteq V$. Facility i has a facility opening cost f_i and client j has d_j units of demand. We are also given a parameter $M \geq 1$. A solution opens some facilities, say F, assigns each client j to an open facility $i(j)$, and connects the open facilities by a Steiner tree T. The total cost incurred is $\sum_{i \in F} f_i + \sum_{j \in \mathcal{D}} d_j c_{i(j)j} + M \sum_{e \in T} c_e$. We want a solution of minimum cost.

A special case of this problem is when all opening costs are 0 and facilities may be opened anywhere, i.e., $\mathcal{F} = V$. If we *know* a facility v that is open, then the problem becomes a special case of the *single-sink buy-at-bulk* problem with two cable types, also known as the *rent-or-buy* problem.

We give the first primal-dual algorithms for these problems and achieve the best known approximation guarantees. We give a 8.55-approximation algorithm for the connected facility location problem and a 4.55-approximation algorithm for the rent-or-buy problem. Previously the best approximation factors for these problems were 10.66 and 9.001 respectively [8]. Further, these results were not combinatorial — they were obtained by solving an exponential size linear programming relaxation. Our algorithm integrates the primal-dual approaches for the facility location problem [11] and the Steiner tree problem [1, 3]. We also consider the connected k-median problem and give a constant-factor approximation by using our primal-dual algorithm for connected facility location. We generalize our results to an edge capacitated variant of these problems and give a constant-factor approximation for these variants.

Keywords : Approximation algorithms, Primal-dual algorithms, Facility location, Connected facility location, Steiner trees.

1 Introduction

Facility location problems have been widely studied in the Operations Research community (see for e.g. [20]). These problems can be described as follows: we are given a graph $G = (V, E)$, a set of facilities $\mathcal{F} \subseteq V$, and a set of clients $\mathcal{D} \subseteq V$. We want to *open* some facilities from the set $\mathcal{F}$ and assign each demand to one of these open facilities. Facilities may have *opening costs*. This class of problems has a wide range of applications. For example, a company might want to open its warehouses at some locations so that its total cost of opening warehouses and servicing customers is minimized.

Many modern day applications occur in settings where the open facilities also want to communicate with each other. Here one desires a two-layer solution where the demand points are first clustered around hubs (facilities) and the hubs are then interconnected to allow them to communicate with one another. One such example is telecommunication network design [2, 21]. A common model of a telecommunication network consists of a *central core* and a set of *endnodes*. The core consists of a set of interconnected core nodes

*A preliminary version of this paper [24] appeared in the Proceedings of the 5th International Workshop on Approximation Algorithms for Combinatorial Optimization, 2002.

†Department of Computer Science, Cornell University, Ithaca, NY 14853. Research partially supported by NSF grant CCR-9912422. Email : swamy@cs.cornell.edu.

‡600 Mountain Ave., Bell Laboratories, Murray Hill, NJ 07974. Work done while the author was at Cornell University supported by NSF grant CCR-9820951 and NSF ITR/IM grant IIS-0081334. Email : amitk@research.bell-labs.com.

which have switching capability. Each core node also incurs some switch cost. Designing the network involves selecting a subset of core nodes, connecting the core nodes to each other and routing traffic from the endnodes to the selected core nodes. Here the clients are the endnodes of the network, and the facilities are the core nodes. The opening cost of a facility corresponds to the switch cost of the corresponding core node.

We capture such a setting by requiring that the open facilities be connected to each other by a *Steiner tree*, i.e., a tree which connects all the open facilities but may also include other non-facility nodes. A Steiner tree is less restrictive than a spanning tree and is appealing because of the simplicity and scalability of the tree architecture. This is the *Connected Facility Location* (ConFL) problem. We are given a graph $G = (V, E)$ with costs $\{c_e\}$ on the edges, a set of facilities $\mathcal{F} \subseteq V$ and a set of demand nodes or clients $\mathcal{D} \subseteq V$. Client j has d_j units of demand and facility i has an opening cost of f_i. We are also given a parameter $M \geq 1$. A solution to ConFL *opens* some facilities, say F, and assigns each demand to an open facility. Let c_{ij} denote the shortest distance between nodes i and j in G (with respect to the costs c_e). If client j gets assigned to facility $i(j)$, we incur an assignment cost proportional to the demand d_j and the distance $c_{i(j)j}$. Further, the solution must connect the open facilities by a Steiner tree T. The cost of connecting facilities is simply the cost of the Steiner tree T scaled by a factor of M. In a telecommunication network, the parameter M reflects the more expensive cost of interconnecting the core nodes with high bandwidth links. The total cost of the solution is the sum of the cost of opening the facilities in F, the assignment costs of demands, and the cost of connecting the open facilities. More precisely, the cost of this solution is

$$\sum_{i \in F} f_i + \sum_{j \in \mathcal{D}} d_j c_{i(j)j} + M \sum_{e \in T} c_e.$$

Our objective is to find a solution of minimum cost. This problem has recently attracted the interest of both the operations research community [14, 18, 19] and the computer science community [8, 9, 12, 13].

The Rent-or-Buy Problem. A special case of this problem is when all opening costs are 0 and facilities may be opened anywhere, i.e., $\mathcal{F} = V$. This problem has many interesting applications.

Suppose we *know* that a facility v is opened by the optimal solution. Then the problem becomes a special case of the *single-sink buy-at-bulk* problem with two cable types, also known as the *rent-or-buy* problem. Here the clients want to send traffic to a special sink vertex v. We need to construct a tree which connects the clients to v and install sufficient capacity on the tree edges to route this traffic. We can either *rent* capacity on an edge, the renting cost being proportional to the amount of capacity rented, or we can pay a one-time expense of M per unit length and *buy* unlimited capacity. The objective is to find a tree with minimum cost.

Applications and Previous Work. Connected Facility Location arises as a natural problem in various important applications. Krick et al. [15] consider a data management/caching problem. Here we have some users issuing read and write requests for data objects. Each object has to be stored in a memory module by paying a certain storage cost — an object may be replicated and stored in multiple locations. Given a placement of objects, a read request for an object issued at node j is served by the nearest location, $i(j)$, that has a copy of the object; a write request however needs to update *all* copies of the object. Krick et al. [15] show that with a small loss in performance, this can be modeled by a single multicast tree connecting all locations that hold a copy of the object. A write request at j first sends a message to $i(j)$ which then initiates the update of all copies via the multicast tree. The goal is to find a placement of objects to memory modules that minimizes the sum of the storage, read and write request costs. This is exactly in the framework of connected facility location. The facilities are the memory modules and the clients are the nodes issuing read/write requests. The facility cost is the storage cost associated with the memory module. The demand of a client is the number of requests issued by the node. Here the connectivity requirement is imposed by

the need to maintain consistency of data. The scaling parameter M corresponds to the total number of write requests for an object.

The rent-or-buy problem is a non-trivial special case of ConFL that arises in diverse scenarios. It abstracts a setting in which demand points need to be clustered around centers and the centers have to be connected further up. Karger & Minkoff [12] introduced the *maybecast* problem which is a probabilistic version of the Steiner tree problem. Each demand point j is activated independently with probability p_j. Given a fixed Steiner tree on $\mathcal{D} \cup \{v\}$, when demand j is activated, all edges on its path to v (the root) become active. The goal is to find a Steiner tree that minimizes the expected cost of the active edges. Gupta et. al. [8] arrived at the rent-or-buy problem by considering the problem of provisioning a virtual private network (VPN) where each VPN endpoint specifies only an upper bound on the amount of incoming and outgoing traffic. In both cases it is shown that there is an optimal or near-optimal solution in which the demand points are first clustered around hubs using shortest-length paths, and the hubs are then connected to the root by a Steiner tree. Thus both these problems reduce to the rent-or-buy problem.

Ravi & Selman [21] gave a constant-factor approximation algorithm for a close variant of this problem where the open facilities have to be connected by a tour. Their algorithm is based on solving an exponential size linear program using the ellipsoid method and rounding the LP solution, which makes the algorithm very inefficient in practice. Karger & Minkoff [12] gave a combinatorial algorithm, but the constant guarantee is much larger. Independently Krick et al. [15] also gave a combinatorial algorithm with an even larger constant. The rent-or-buy problem is a special case of the single-sink buy-at-bulk problem for which Guha, Meyerson and Munagala [6] gave a constant-factor approximation algorithm. The constant was improved by Talwar [25]. Gupta et al. [8] gave an algorithm with an approximation guarantee of 10.66 for ConFL and 9.001 for the rent-or-buy problem. This is also based on rounding an exponential size LP as in [21], and suffers from the same drawbacks. Previously these were the best known guarantees. Kumar et al. [17] implemented a heuristic for the rent-or-buy problem and used it to construct VPN trees. They report that the algorithm outperforms standard heuristics over a wide range of parameter values. However they could not give any worst case performance guarantee.

Our Results. We give a primal-dual 8.55-approximation algorithm for the connected facility location problem and a 4.55-approximation algorithm for the rent-or-buy problem. Thus we give a combinatorial algorithm and also achieve the current best approximation ratios. Our algorithm is conceptually simple and can be easily implemented. We feel that the algorithm will also perform well in practice and justify its theoretical merit.

In many settings there is an additional requirement that at most k facilities can be opened. We call this variant of ConFL the Connected k-Median problem. We use our primal-dual algorithm to get a 15.55-approximation algorithm for this problem. To the best of our knowledge, this is the first time anyone has considered this problem, though the connected k-center problem has been considered earlier [5]. We generalize our results to an edge capacitated version of these problems. These differ from the uncapacitated versions in the facility location aspect. We now require clients to be connected to facilities via cables which have a fixed cost of σ per unit length and a capacity of u. Multiple cables may be laid along an edge. The cost of connecting facilities is still M times the cost of the tree T. We give a constant-factor approximation for these capacitated variants.

Our Techniques. Connected Facility Location has elements of both the facility location and the Steiner tree problem. Without the connectivity requirement, the problem is simply the uncapacitated facility location problem. Once we know which facilities to open, we can assign each demand to the closest open facility and connect the open facilities by a Steiner tree.

However, a simple strategy that first decides which facilities to open by running a facility location

algorithm, and then connects the open facilities by a Steiner tree, performs poorly. For example, in the rent-or-buy problem, this would just open a facility at each demand point, but connecting all the open facilities might incur a huge cost. Thus there is an implicit cost imposed on each facility by the connectivity requirement. To ensure that it is economically viable to open a facility we need to cluster enough demand at the facility. Previously [7, 12] the clustering was achieved by solving a Load Balanced Facility Location (LBFL) instance, where we want each open facility to serve at least M clients. The disadvantage of this approach is that by reducing to LBFL we throw away problem specific information. The LBFL instance is solved using a black box and makes no use of the fact that the need to cluster demands is imposed by the connectivity requirement of ConFL. Further we only know a bicriteria approximation for LBFL, so the demand lower bound on a facility is only approximately satisfied which increases the approximation ratio.

Our algorithm is based on a novel application of the primal-dual schema. The basic idea is to consider an integer programming formulation of the problem and the dual of its linear programming relaxation, and construct simultaneously an integer primal solution and a dual solution. The dual linear program can be interpreted as comprising two parts; a part resembling the dual of the facility location problem and a part corresponding to the dual of the Steiner tree problem. The algorithm is in two phases. The first phase is a *facility location* phase where we decide which facilities to open, connect demands to facilities and cluster the demands at each facility. At the end of this phase, we obtain a primal facility location solution and a feasible dual solution. In the primal solution the demands are clustered so that each open facility serves *at least* M demand points, satisfying the demand lower bound. We do this by *charging some of the cost incurred to the Steiner tree portion of the dual solution*, thereby exploiting the fact that any ConFL solution also needs to connect the facilities it opens. Despite the added clustering requirement, our algorithm has a fairly simple description. Each demand j keeps raising its dual variable, α_j, till it gets connected to a facility and is 'near' a point at which M demands are clustered. All other variables simply respond to this change trying to maintain feasibility or complementary slackness. Phase 2 is a *Steiner* phase where we connect the open facilities by a Steiner tree. The dual solution constructed in this phase is not feasible, but we show that the infeasibility is bounded by a small *additive* factor.

Previous Work on Primal-Dual Algorithms. Our work reinforces the belief that the primal-dual schema is extremely versatile. The first truly primal-dual approximation algorithm was given by Bar-Yehuda & Even (see [4]) for the vertex cover problem. Subsequently, primal-dual algorithms have especially flourished in the area of network-design problems. One of the first such algorithms was by Agrawal, Klein & Ravi [1] for the generalized Steiner problem on networks. Goemans & Williamson [3] further refined the primal-dual method and highlighted its usefulness by extending it to a large class of network-design problems; see [4, 26] for a survey of this and earlier work. The basic mechanism involves raising the dual variables and setting primal variables till an integral primal solution is found satisfying the primal complementary slackness conditions. Next a *reverse delete* step is used to remove any redundancies in the primal solution. This relaxes the dual slackness conditions. The approximation ratio of the algorithm is this relaxation factor.

Jain & Vazirani [11] gave an elegant primal-dual algorithm for various facility location problems which could not be solved by the earlier schema. They remove redundancies while relaxing the primal slackness conditions. They also show that their algorithm can be used to solve other facility location variants, most notably the k-median problem using a Lagrangian relaxation.

We need to integrate both the above approaches. As observed earlier simply running a facility location algorithm and then a Steiner tree algorithm performs poorly. The added requirement is that each open facility should serve a significant amount of demand. Our algorithm inherits the versatility of the algorithm by [11] and we use it to solve the Connected k-Median problem using similar ideas.

Subsequent Work. Since the publication of an extended abstract of this paper [24], two results of interest have been obtained.

Kumar, Gupta and Roughgarden [16] considered the multicommodity rent-or-buy problem and gave a constant-factor approximation algorithm. Here there are multiple source-sink pairs and each source wants to send traffic to its sink. We need to build a network connecting each source-sink pair and install enough capacity on the edges to route this traffic. We may either rent capacity by paying a cost proportional to the capacity rented or buy unlimited capacity paying a fixed cost of M per unit length. Their algorithm is however much more involved and they get a *much* worse approximation factor. It remains a very interesting open problem to see whether the techniques used here and in [16] can be extended or adapted to solve the multiple source-sink buy-at-bulk problem with multiple types of cables.

Gupta, Kumar and Roughgarden [9] very recently gave a randomized approximation algorithm for the rent-or-buy problem with a ratio of $2 + \rho_{ST}$. This improves upon our result but it is not clear if their algorithm can be derandomized. For connected facility location they give a 10.1-approximation algorithm.

2 A Linear Programming Relaxation

In what follows, i will be used to index facilities, j to index the clients and e to index the edges in G. We will use the terms client and demand point interchangeably.

ConFL can be formulated naturally as an Integer Program. We assume that we know a facility v that is opened and hence belongs to the Steiner tree constructed by the optimal solution. We can make this assumption because we can try all $|\mathcal{F}|$ different possibilities for v.

We can now write an integer program (IP) for ConFL as follows:

$$\begin{aligned}
\min \quad & \sum_i f_i y_i + \sum_j d_j \sum_i c_{ij} x_{ij} + M \sum_e c_e z_e && && \text{(IP)}\\
\text{s.t.} \quad & \sum_i x_{ij} \geq 1 && \text{for all } j\\
& x_{ij} \leq y_i && \text{for all } i, j\\
& y_v = 1\\
& \sum_{i \in S} x_{ij} \leq \sum_{e \in \delta(S)} z_e && \text{for all } S \subseteq V, v \notin S,\ j\\
& x_{ij}, y_i, z_e \in \{0,1\} && && (1)
\end{aligned}$$

Here y_i indicates if facility i is open, x_{ij} indicates if client j is connected to facility i and z_e indicates if edge e is included in the Steiner tree. Relaxing the integrality constraints (1) to $x_{ij}, y_i, z_e \geq 0$ gives us a linear program (LP).

3 A Primal-Dual Approximation Algorithm

We now show that the integrality gap of (LP) is at most 9 by giving a primal-dual algorithm for this problem. For simplicity, we assume that all d_j are equal to 1. We show how to get rid of this assumption in Section 3.3.

3.1 The Rent-or-Buy Problem

We first consider the case where all opening costs are 0 and $\mathcal{F} = V$, i.e., a facility can be opened at any vertex of V. The linear program (LP) now simplifies to:

$$\begin{aligned}
\min \quad & \sum_j \sum_i c_{ij} x_{ij} + M \sum_e c_e z_e && && \text{(P1)}\\
\text{s.t.} \quad & \sum_i x_{ij} \geq 1 && \text{for all } j\\
& \sum_{i \in S} x_{ij} \leq \sum_{e \in \delta(S)} z_e && \text{for all } S \subseteq V, v \notin S,\ j\\
& x_{ij}, z_e \geq 0
\end{aligned}$$

The dual of this linear program is:

$$\begin{aligned}
\max \quad & \sum_j \alpha_j && && \text{(D1)}\\
\text{s.t.} \quad & \alpha_j \leq c_{ij} + \sum_{S \subseteq V : i \in S, v \notin S} \theta_{S,j} && \text{for all } i \neq v, j && \text{(2)}\\
& \alpha_j \leq c_{vj} && \text{for all } j && \text{(3)}\\
\sum_j & \sum_{S \subseteq V : e \in \delta(S), v \notin S} \theta_{S,j} \leq M c_e && \text{for all } e && \text{(4)}\\
& \alpha_j, \theta_{S,j} \geq 0
\end{aligned}$$

Intuitively, α_j is the *payment* that demand j is willing to make towards constructing a feasible primal solution. Constraint (2) says that a part of the payment α_j goes towards assigning j to a facility i. The remaining part goes towards constructing the part of the Steiner tree that joins i to v.

Algorithm Description

We begin with a simplifying assumption. We assume that a facility can be opened *anywhere along an edge*. We collectively refer to vertices in V and internal points on an edge as *locations*. We reserve the term facility for a vertex in $\mathcal{F}$. We may assume that for any edge $e = (u, w)$, c_e is equal to c_{uw}, the shortest path distance from u to w. The metric c is extended to a metric on locations by considering e to be composed of infinitely many edges of infinitesimal length. So for points p on e the distance c_{up} varies continuously and monotonically from 0 to c_e as we go from u to w, and $c_{wp} = c_e - c_{up}$. For any other vertex $r \neq u, w$, we set $c_{rp} = \min(c_{ru} + c_{up}, c_{rw} + c_{wp})$. Finally for any two points p, q on edges $e_1 = (u, w), e_2$ respectively, $c_{pq} = \min(c_{uq} + c_{up}, c_{wq} + c_{wp})$.

The intuition behind our algorithm is as follows. Suppose each client had at least M units of demand. Then, the optimal solution would locate a facility at each of these clients and connect them by a Steiner tree. So, our algorithm first *clusters* the demands in groups of M and then builds a Steiner tree joining these clusters.

Initially, all the dual variables are 0. The algorithm runs in two phases. In the first phase, we *cluster* the demands in groups of M. Once we have this, we run the second phase where we build the Steiner tree.

Phase 1. We raise the dual variables α_j for all demands in this phase. We have a notion of time, t. Initially $t = 0$. As time increases, we raise the dual variables α_j at unit rate. We shall also *tentatively open* some locations. At $t = 0$, v is tentatively open and all other locations are closed.

At some point of time, we say that demand j is *tight* with a location i if $\alpha_j \geq c_{ij}$. Let S_j be the set of vertices with which j is tight at some point of time. When we raise α_j, we also raise $\theta_{S_j,j}$ at the same rate. This will ensure feasibility of constraints (2). So, it is enough to describe how to raise the dual variables α_j.

Demands can be in two states: *frozen* or *unfrozen*. When a demand j gets frozen, we stop raising its dual variable α_j. So if demand j is unfrozen at time t, $\alpha_j = t$. After j is frozen, it does not become tight with any new location, i.e., a location not in S_j. Initially, all demands are unfrozen.

We start raising the α_j of all demands at unit rate until one of the following events happens (if several events happen, consider them in any order):

1. j becomes tight with a tentatively open location i: j becomes frozen.

2. There is a closed location i with which at least M demands are tight: tentatively open i. All of the demand points tight with i become frozen.

We now raise the α_j of unfrozen demands only. We continue this process until all demands become frozen. Figure 3.1 shows a sample run of the algorithm with $M = 2$ and 5 demand points. Note that although there is a continuum of points along an edge, to implement the above process we only need to know the time when the next event will take place. This can be obtained by keeping track of, for every edge and every demand j, the portion of the edge that is tight with j.

Now we decide which locations to open. Let L be the set of tentatively open locations. We say that $i, i' \in L$ are dependent if there is demand j which is tight with both these locations. We say that a set of locations is *independent* if no two locations in this set are dependent. We find a maximal independent set L' of locations in L as follows: arrange the locations in L in the order they were tentatively opened. Consider the locations in this order and add a location to L' if no dependent location is already present in L'. We open the locations in L'. Observe that $v \in L'$.

We assign a demand j to an open location as follows. If j is tight with some $i \in L'$, assign j to i. Otherwise let i be the location in L that caused j to become frozen. So j is tight with i. There must be some previously opened location $i' \in L'$ such that i and i' are dependent. We assign j to i'. Let $\sigma(j)$ denote the location to which j is assigned.

We now have to build a Steiner tree on L'. First we augment the graph G to include edges incident on open non-vertex locations. Let $\{i_1, \ldots, i_k\}$ be the open locations on edge $e = (u, w)$ ordered by increasing distance from u, with $i_1 \neq u, i_k \neq w$. We add edges $(u, i_1), (i_1, i_2), \ldots, (i_{k-1}, i_k), (i_k, w)$ to G.

Phase 2. For a location $i \in L'$, let D_i be the set of demands tight with i. Let $D' = \bigcup_{i \in L'-\{v\}} D_i$. First, we set $\alpha_j = 0$ for all j. We raise the α_j value of demands in D' only, and simulate the primal-dual algorithm for the (rooted) Steiner tree problem.

Initially, the minimal violated sets (MVS) are the singleton sets $\{i\}$ for $i \in L' - \{v\}$. For a set S, define $D_S = \bigcup_{i \in S \cap L'} D_i$. The tree T that we shall construct is empty to begin with. For each MVS S, $j \in D_S$, we raise α_j at rate $1/|D_S|$. We also raise $\theta_{S,j}$, at the same rate. This ensures that $\sum_j \theta_{S,j}$ grows at rate 1 for any MVS S. Note that we are not ensuring feasibility of constraints (2), (3).

We say that an edge goes tight if (4) holds with equality for that edge. We raise the dual variables till an edge e goes tight. We add e to T and update the minimal violated sets. This process continues till there is no violated set, i.e., we have only one component (so v is in this component). Now we consider edges of T in the reverse order they were added and remove any redundant edges. This is our final solution.

Analysis

Let $\left(\alpha^{(1)}, \theta^{(1)}\right)$, $\left(\alpha^{(2)}, \theta^{(2)}\right)$ be the value of the dual variables at the end of Phases 1 and 2 respectively.

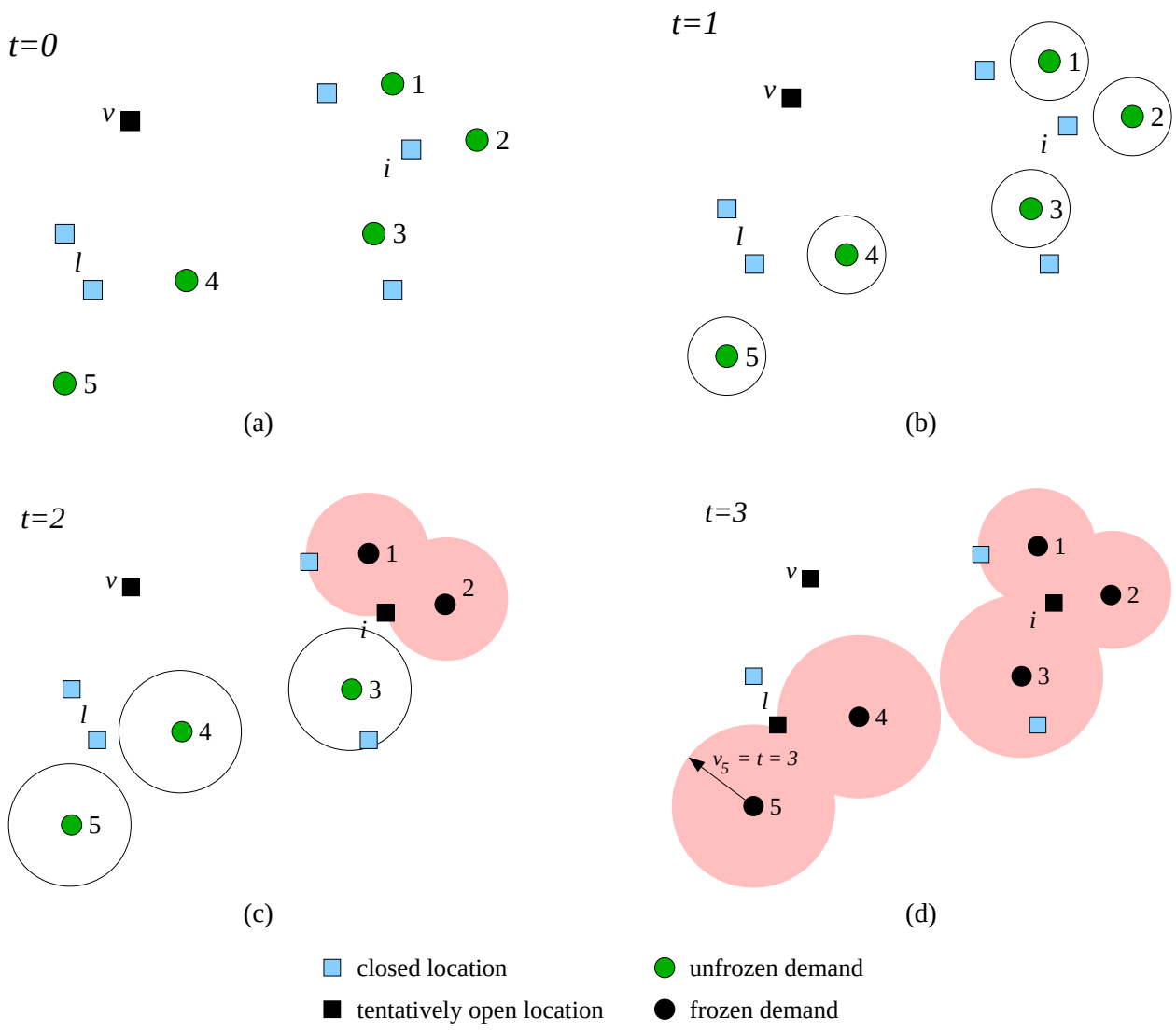

Figure 3.1: A sample run with $M = 2$. (a) The initial state, (b) $t = 1$, (c) i becomes tight with demands 1 and 2; i is tentatively opened and 1, 2 become frozen, (d) The final solution. Demand 3 reaches i and gets frozen; l becomes tight with demands 4 and 5 and is tentatively opened, causing demands 4 and 5 to freeze.

Lemma 3.1 *The dual solution* $\left(\alpha^{(1)}, \theta^{(1)}\right)$ *is feasible.*

Proof : It is easy to see that (2) is satisfied. Indeed, once j gets tight with i, α_j and $\sum_{S:i\in S, v\notin S} \theta_{S,j}$ are raised at the same rate. Similarly, (3) is satisfied.

Now consider an edge $e = (u, w)$. Let $l(j)$ be the contribution of j to the left hand side of (4) for this edge, i.e., $l(j) = \sum_{S:e\in\delta(S), v\notin S} \theta^{(1)}_{S,j}$. Suppose $c_{ju} \leq c_{jw}$. So, j becomes tight with u before it gets tight with w. Consider a point p on the edge (u, w) at distance x from u. If p were the last point on this edge with which j became tight with (before it became frozen), then $l(j) \leq x$. Define $f(j, x)$ as 1 if j is tight with p and j was not frozen at the time at which it became tight with p, otherwise $f(j, x)$ is 0. So, we can write

$l(j) \leq \int_0^{c_e} f(j,x)dx$. Interchanging the summation and the integral in (4), we get

$$\sum_j \sum_{S \subseteq V : e \in \delta(S), v \notin S} \theta^{(1)}_{S,j} \leq \sum_j \int_0^{c_e} f(j,x)dx = \int_0^{c_e} \sum_j f(j,x)dx$$

Now for any x, $\left(\sum_j f(j,x)\right) \leq M$. Otherwise, we have more than M demands that are tight with a point such that none of these demands are frozen — a contradiction. So $\int_0^{c_e} \sum_j f(j,x)dx$ is at most Mc_e which proves the lemma. ■

Lemma 3.2 *At the end of Phase 1, the assignment cost of any demand j is at most $3\alpha^{(1)}_j$.*

Proof : This clearly holds if j is tight with a location in L'. Otherwise let j be assigned to i. Let i' be the tentatively open facility that caused j to become frozen. It must be the case that i and i' are dependent. So there is a demand k which is tight with both i and i'. Let $t_{i'}$ be the time at which i' was tentatively opened. Define t_i similarly. It is clear that $\alpha^{(1)}_j \geq t_{i'}$.

Now, $c_{ij} \leq c_{ik} + c_{ki'} + c_{i'j} \leq 2\alpha^{(1)}_k + \alpha^{(1)}_j$. Also, $\alpha^{(1)}_k \leq t_{i'}$. Otherwise, at time $t = \alpha^{(1)}_k$, k is tight with both i and i'. Suppose it becomes tight with i first (the other case is similar). If i is tentatively open at this time, then k will freeze and so it will never become tight with i'. Therefore, i can not be tentatively open at this time. But then, k must freeze by the time i becomes tentatively open, i.e., $\alpha^{(1)}_k \leq t_i \leq t_{i'}$. So, $\alpha^{(1)}_k \leq t_{i'} \leq \alpha^{(1)}_j$. This implies that $c_{ij} \leq 3\alpha^{(1)}_j$. ■

Lemma 3.3 *If i is an open location and $j \in D_i$, $c_{\sigma(j)j} \leq \alpha^{(1)}_j$.*

We now bound the cost of the tree T. Recall that $D' = \bigcup_{i \in L' - \{v\}} D_i$.

Lemma 3.4 $cost(T) \leq 2 \cdot \sum_{j \in D'} \alpha^{(2)}_j$.

Proof : Consider Phase 2. At any point of time, define the variable θ_S, where S is a minimal violated set, as $\sum_j \theta_{S,j}$. We observed that θ_S grows at rate 1. Thus, Phase 2 simulates the primal-dual algorithm for the rooted Steiner tree problem with v as the root. So, the cost of the tree is bounded by $2 \cdot \sum_S \theta_S$ [4, 1, 26], where the sum is over all subsets of vertices S. But $\sum_S \theta_S = \sum_{j \in D'} \alpha^{(2)}_j$. ■

Lemma 3.5 *Consider a demand j. If $i \neq v$, then $\alpha^{(2)}_j \leq c_{\sigma(j)j} + c_{ij} + \sum_{S \subseteq V : i \in S, v \notin S} \theta^{(2)}_{S,j}$. Further, $\alpha^{(2)}_j \leq c_{\sigma(j)j} + c_{vj}$.*

Proof : If $j \notin D'$, $\alpha^{(2)}_j = \theta^{(2)}_{S,j} = 0$ and the inequalities above hold. So fix a demand $j \in D'$ and facility i, $i \neq v$. During the execution of Phase 2, let S_t be the component to which j contributes at time t. Consider the earliest time t' for which $i \in S_{t'}$. After this time, both the left hand side and right hand side of (2) increase at the same rate, so we only need to bound the increase in α_j by time t'. Let $l = \sigma(j)$. Since we are raising α_j, it must be the case that $j \in D_l$ and so, $c_{lj} \leq \alpha^{(1)}_j$. We claim that $t' \leq Mc_{li}$. This is true since S_t always contains l, and by time $t = Mc_{li}$ all of the edges along the shortest path between l and i would have grown tight. Since α_j increases at a rate of at most $1/M$, the increase in α_j by time t' is at most $\frac{Mc_{li}}{M} \leq c_{lj} + c_{ij}$. This proves the first inequality. The second inequality is proved similarly. ■

It is clear that the $\theta^{(2)}_{S,j}$ values satisfy (4), so we have shown that $\left(\alpha', \theta^{(2)}\right)$ is a feasible dual solution, where $\alpha'_j = \max(\alpha^{(2)}_j - c_{\sigma(j)j}, 0)$. We can now prove the main theorem. Let OPT be the cost of the optimal solution.

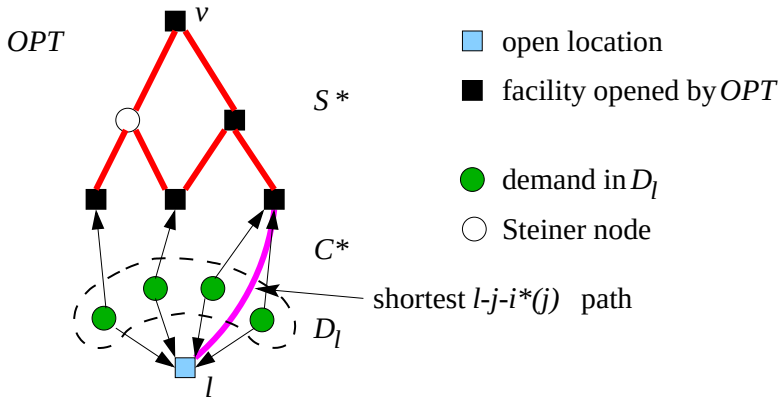

Figure 3.2: Extending the tree in OPT to a Steiner tree on the open locations.

Theorem 3.6 *The above algorithm produces a solution of cost at most* $5 \cdot OPT$.

Proof : Note that $\alpha_j^{(2)} \leq \alpha'_j + c_{\sigma(j)j}$. By Lemma 3.4, $cost(T) \leq 2\sum_j \alpha'_j + 2\sum_{j \in D'} c_{\sigma(j)j} \leq 2 \cdot OPT + 2\sum_{j \in D'} \alpha_j^{(1)}$. By Lemmas 3.3 and 3.2, the assignment cost of j is at most $\alpha_j^{(1)}$ if $j \in D'$, and at most $3\alpha_j^{(1)}$ otherwise. Adding all terms, we see that the cost of our solution is at most $5 \cdot OPT$. ■

In Phase 2 we can use any ρ_{ST}-approximation algorithm to build the Steiner tree on the open locations and get an approximation ratio of $3 + \rho_{ST}$. We assume that $\rho_{ST} \leq 2$. Let C^*, S^* denote the assignment and Steiner tree costs of an optimal solution, OPT. The tree in OPT yields a Steiner tree on the open locations if we connect each $l \in L'$ to the tree in OPT via the shortest $l - j - i^*(j)$ path for $j \in D_l$, where $i^*(j)$ is the facility to which j is assigned in OPT (see Fig. 3.2). For any $l \in L'$ the cost of adding the connecting edges is M(length of the shortest $l - j - i^*(j)$ path for $j \in D_l$) $\leq \sum_{j \in D_l}(c_{i^*(j)j} + c_{lj})$ since $|D_l| \geq M$. Summing over all $l \in L'$, the cost of the tree obtained is at most $S^* + C^* + \sum_{j \in D'} c_{\sigma(j)j}$. So we can bound the total cost of our solution by $\sum_j c_{\sigma(j)j} + \rho_{ST}\left(S^* + C^* + \sum_{j \in D'} c_{\sigma(j)j}\right) \leq 3\sum_j \alpha_j^{(1)} + \rho_{ST} \cdot OPT \leq (3 + \rho_{ST}) \cdot OPT$ using Lemmas 3.3 and 3.2. Taking $\rho_{ST} = 1.55$ [23] we get the following.

Theorem 3.7 *There is a* $4.55 \cdot OPT$*-approximation algorithm for the rent-or-buy problem.*

Our solution may be infeasible since a non-vertex location may be opened as a facility. Let $e = (u, w)$ be an edge and suppose we open locations on the internal points of e. Let D_e be the set of demands which are assigned to such locations. Let $D_u \subseteq D_e$ be the set of demands that reach their assigned location on e via u, i.e., $c_{\sigma(j)j} = c_{uj} + c_{\sigma(j)u}$ for $j \in D_u$. D_w is defined similarly. The Steiner tree T must contain at least one of u or w. If both $u, w \in T$, we assign clients in D_u to u and clients in D_w to w without increasing the cost. Suppose $u \in T, w \notin T$. Let l be the open location which is farthest from u on e. We assign all demands in D_u to u. If $|D_w| < M$, we assign clients in D_w to u and remove edges in T that lie along e; otherwise we reassign all clients in D_w to w and add all of e to T. Note that T is still a Steiner tree on the open locations. It is easy to see that the total cost only decreases. Thus, we can shift all open locations to a vertex of G without increasing the total cost.

Let C, S denote the assignment and Steiner tree costs of our solution. We now bound the quantity $2C + S$. We will use this result in Section 5 where we consider a generalization to edge capacities.

Lemma 3.8 *The solution obtained satisfies* $2C + S \leq 7.55 \cdot OPT$.

Proof : $2C + S \leq (C + S) + 3\sum_j \alpha_j^{(1)} \leq 7.55 \cdot OPT$. ■

3.2 The General Case

We now consider the case where $\mathcal{F}$, need not be V and facility i has an opening cost $f_i \geq 0$. Since facilities may only be opened at specific locations, it is possible that an edge is used both to route demand from a client to a facility, and also as an edge in the Steiner tree to connect facilities. We call the former type of edge a *facility location edge* and the latter a *Steiner edge*. For convenience we assume that $f_v = 0$. Clearly, this does not affect the approximation ratio of the algorithm. Recall that i indexes the facilities in $\mathcal{F}$. The primal and dual LPs are:

$$\begin{aligned}
\min \quad & \sum_{i \neq v} f_i y_i + \sum_j \sum_i c_{ij} x_{ij} + M \sum_e c_e z_e && && \text{(P2)} \\
\text{s.t.} \quad & \sum_i x_{ij} \geq 1 && \text{for all } j \\
& x_{ij} \leq y_i && \text{for all } i \neq v, j \\
& x_{vj} \leq 1 \\
& \sum_{i \in S} x_{ij} \leq \sum_{e \in \delta(S)} z_e && \text{for all } S \subseteq V, v \notin S,\ j \\
& x_{ij}, y_i, z_e \geq 0
\end{aligned}$$

$$\begin{aligned}
\max \quad & \sum_j \alpha_j - \sum_j \beta_{vj} && && \text{(D2)} \\
\text{s.t.} \quad & \alpha_j \leq c_{ij} + \beta_{ij} + \sum_{S \subseteq V : i \in S, v \notin S} \theta_{S,j} && \text{for all } i \neq v, j && \text{(5)} \\
& \alpha_j \leq c_{vj} + \beta_{vj} && \text{for all } j \\
& \sum_j \beta_{ij} \leq f_i && \text{for all } i \neq v && \text{(6)} \\
& \sum_j \sum_{S \subseteq V : e \in \delta(S), v \notin S} \theta_{S,j} \leq M c_e && \text{for all } e \\
& \alpha_j, \beta_{ij}, \theta_{S,j} \geq 0
\end{aligned}$$

3.2.1 Overview of the algorithm

The basic idea is similar to the algorithm in the previous section. We still want to gather at least M demands at every facility that we open so that the cost of connecting this facility to other open facilities by Steiner edges can be amortized against the gathered demand. However, while earlier where we could tentatively open any location with which M demands are tight, we cannot do that here since the set of candidate facility locations $\mathcal{F}$ may be a very small subset of V. Also, we need to pay a facility opening cost before we can open a facility.

We will not quite be able to meet the demand requirement of M at every facility we open, but we will ensure that for every open facility, there are M demands gathered at a point "near" the facility (see Fig. 3.3). In Phase 1 of the algorithm, we will open facilities and assign each client to an open facility. Additionally, for each open facility we will connect it to the point near it at which M demands are gathered using Steiner edges, and we will argue that we can pay for the cost of buying this path by the combined dual of the gathered demands. These components act as the terminals upon which the Steiner tree is constructed in Phase 2, whose cost we bound as in the previous section.

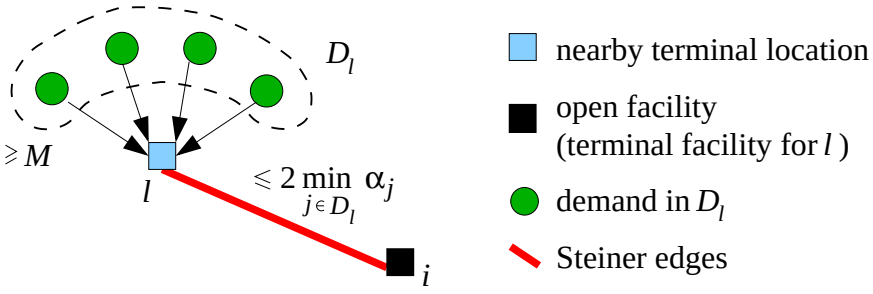

Figure 3.3: Steiner edges connecting an open facility to the point "nearby" where M demands are gathered.

3.2.2 Details of the algorithm

Phase 1. Most of the changes are in this phase. A location still refers to a vertex in V or a point along an edge. We will only open facilities at locations in $\mathcal{F} \subseteq V$. Initially all dual variables are 0 and only facility v is tentatively open. We also declare location v to be a *terminal location*. Recall that demand j is said to be tight with location i if $\alpha_j \geq c_{ij}$. As in the previous section, we will grow each dual variable α_j till j becomes tight with a location, referred to as a terminal location, with which at least M demands are tight. Once this happens however, we do not freeze j yet. Since we have to assign client j to an open facility and also have to pay for opening facilities, we continue to increase α_j till j becomes tight with a tentatively open facility. While doing so, if j becomes tight with a facility that is not yet open, then it starts contributing toward the facility opening cost of this facility.

To describe the primal-dual process in detail we define a few additional concepts. As before, a demand can be frozen or unfrozen. Further, a demand j could either be *free* or be a *slave*. At $t = 0$, each demand j is free and unfrozen. We say that demand j is *bound* to a location l if j is tight with l *and was free when it became tight with* l. Define the weight of a location l as the number of demands that are *bound* to l. We say that a facility i has been *paid* for if $\sum_j \beta_{ij} = f_i$.

At any point of time, define S_j to be the set of vertices with which demand j is tight. When j becomes tight with a facility i, we have two options — we can raise β_{ij} or we can raise $\theta_{S_j,j}$. We raise $\theta_{S_j,j}$[1] at the same rate and continue this till j becomes tight with a terminal location, that is, a location that has at least M demands *bound* to it. At this point we say that j becomes a *slave* — it is no longer free. Similarly, when j becomes tight with a location l that is *not* a facility, we may or may not raise $\theta_{S_j,j}$ (we have this option since constraint (5) only applies to facilities i). We first increase $\theta_{S_j,j}$ till j becomes tight with a terminal location and is declared to be a slave. After this point we start raising β_{ij} for each facility $i \in S_j$ and do not raise $\theta_{S_j,j}$ any more. More precisely, we raise the α_j of *every* unfrozen demand, be it free or a slave, at unit rate until one of the following events happens:

1. The weight of some location l becomes at least M: declare l to be a terminal location. If j is free and tight with l, it now becomes a *slave*. From this point on we raise only β_{ij} for facilities i in S_j (there may be none if the current $\alpha_j < \min_i c_{ij}$) as described above.

2. A free j becomes tight with a terminal location l: j becomes a slave. If $l = v$, connect j to l and freeze j. Otherwise we stop raising $\theta_{S_j,j}$ and raise β_{ij} for facilities i in S_j.

3. A facility i gets paid for, i.e., $\sum_j \beta_{ij} = f_i$: tentatively open i. If an (unfrozen) slave demand j is tight with i, connect it to i and freeze j.

[1]The reverse —raising β_{ij} first until j gets connected to a facility and then increasing $\theta_{S_j,j}$ also works —but we raise the dual variables in this fashion in order to prove a guarantee for the connected k-median problem.

4. A slave demand j becomes tight with a tentatively open facility i: connect j to i, freeze j.

We continue this process until all j become frozen. Frozen demands do not participate in any new events. Note that every demand j starts out as free and unfrozen, then becomes a slave by becoming tight with a terminal location, and finally gets frozen by getting connected to exactly one tentatively open facility. Let $\left(\alpha^{(1)}, \beta^{(1)}, \theta^{(1)}\right)$ be the dual solution obtained. Clearly $\beta^{(1)}_{vj} = 0$ for all j.

Let L be the set of all terminal locations. Let t_l be the time at which l was declared a terminal location. Let D_l be the set of demands *bound to* l. We associate a *terminal facility* with each $l \in L$. Consider the demand in D_l with smallest $\alpha^{(1)}_j$ and let i be the tentatively open facility to which it is connected. We call this demand the *representative demand* of location l, and denote i as the terminal facility corresponding to l. Let the terminal facility corresponding to v be v itself. Let F be the set of all terminal facilities. We will only open facilities from the set F.

We will pick a subset of terminal locations and open the terminal facilities corresponding to these locations. For each location l that we pick, we will connect l to its terminal facility i by buying Steiner edges along a shortest $l - i$ path (see Fig. 3.3). We choose the subset of terminal locations carefully so as to ensure that a demand j does not pay for opening or connecting more than one facility. Say that two facilities i, i' are dependent if either (1) there is a demand j with both $\beta^{(1)}_{ij}, \beta^{(1)}_{i'j} > 0$, or (2) there is a location $l \in L$ and a demand j such that i is the terminal facility corresponding to l, j is in D_l, and $\beta^{(1)}_{i'j} > 0$. The second condition is added to ensure that j does not pay for both opening i' and for connecting i to l via Steiner edges. We also have a notion of dependence between *locations* in L. We say that *locations* l and l' in L are dependent if either there is a demand that is bound to both l and l', or the terminal facilities corresponding to l and l' are dependent. Now we greedily select a maximal independent set of locations by looking at locations in a particular order. With each $l \in L$ we associate a value ϕ_l. Let j be the representative demand of l. Define $\phi_l = \max(\alpha^{(1)}_j, t_l)$, set $\phi_v = 0$. We look at the locations in L in increasing order of ϕ_l, and select a maximal independent subset L' of L as before. Let F' be the set of terminal facilities corresponding to locations in L'. We open all the facilities in F'. Note that $v \in F'$.

We associate a terminal location $\sigma(j)$ with each demand j. If $j \in D_l$ where $l \in L'$, set $\sigma(j) = l$. Note that $\sigma(j)$ is well defined due to our independent set construction. Otherwise let l be the location in L that caused j to become a slave. There is a previously selected location $l' \in L'$ such that l and l' are dependent. Set $\sigma(j) = l'$. Demand j is assigned to a facility $i(j) \in F'$ as follows: if there is a facility $i \in F'$ such that $\beta^{(1)}_{ij} > 0$, assign j to i. Otherwise assign j to the terminal facility corresponding to $\sigma(j)$.

Let $D' = \bigcup_{l \in L' - \{v\}} D_l$. We now form some components by adding edges connecting each l in L' to its terminal facility via a shortest path. Break any cycles by deleting edges. Let T' be the set of edges added.

Phase 2. This phase is similar to that of the previous section. G is augmented as before to include edges incident on locations $l \in L'$. We initialize our minimal violated sets to the components of T'. All dual variables are initially 0. We do not raise any β_{ij} in this phase. We shall raise the α_j value of demands in D' only. For a set S, define D_S to be $\bigcup_{l \in S \cap L'} D_l$. The rest of the procedure is identical to Phase 2 of the previous section. This yields a tree T connecting all the open facilities. Let $\left(\alpha^{(2)}, 0, \theta^{(2)}\right)$ be the dual solution constructed by this phase.

Remark. It is possible that T contains an edge which has a non-vertex location as an end-point — this will happen if such a location is a leaf of the tree T. We simply delete such edges to get a new tree that only uses edges of the original graph.

Analysis

The proof of the following lemma is very similar to the proof of Lemma 3.1.

Lemma 3.9 *$\left(\alpha^{(1)}, \beta^{(1)}, \theta^{(1)}\right)$ is a feasible dual solution.*

Lemma 3.10 *Let l be a terminal location and i be its corresponding terminal facility. Then $c_{il} \leq \min_{j \in D_l} 2(\alpha_j^{(1)} - \beta_{ij}^{(1)}) \leq 2\phi_l$.*

Proof : Let j be any demand in D_l and k be the representative demand of l, so k is connected to i. Then, $c_{il} \leq 2\alpha_k^{(1)} \leq 2\phi_l$. So if $\beta_{ij}^{(1)} = 0, c_{il} \leq 2(\alpha_j^{(1)} - \beta_{ij}^{(1)})$. Otherwise, let t_j be the time at which j became a slave. Note that $\alpha_j^{(1)} = \max(t_j, c_{ij}) + \beta_{ij}^{(1)}$ and $c_{lj} \leq t_j$, so $c_{il} \leq 2(\alpha_j^{(1)} - \beta_{ij}^{(1)})$. ■

Lemma 3.11 *Let l and l' be dependent terminal locations with $\phi_l \leq \phi_{l'}$. If i is the terminal facility corresponding to l, $c_{il'} \leq 6\phi_{l'}$.*

Proof : Let k be the representative demand of location l. Let i' be the terminal facility for l' and k' be the representative demand of l'. By Lemma 3.10, $c_{il} \leq 2\phi_l$ and $c_{i'l'} \leq 2\phi_{l'}$. Let t_i and $t_{i'}$ be the times at which i and i' got tentatively opened respectively. There are four cases to consider depending on why l and l' are dependent.

1. $\exists j \in D_l \cap D_{l'}$. Since j was free when it became tight with l and l', $c_{lj}, c_{l'j} \leq \max(t_l, t_{l'}) \leq \max(\phi_l, \phi_{l'}) = \phi_{l'}$. Combined with Lemma 3.10 this gives $c_{il'} \leq c_{il} + c_{ll'} \leq 4\phi_{l'}$.

2. $\exists j$ such that $\beta_{ij}^{(1)}, \beta_{i'j}^{(1)} > 0$. This implies that $c_{i'j}, c_{ij} \leq \alpha_j^{(1)} \leq t_{i'}, t_i$. So $c_{ii'} \leq 2t_{i'} \leq 2\alpha_{k'}^{(1)} \leq 2\phi_{l'}$, and $c_{il'} \leq 4\phi_{l'}$.

3. There is a terminal location r (could be l), demand $j \in D_r$ such that i is the terminal facility for r and $\beta_{i'j}^{(1)} > 0$. By the above argument, $c_{i'j} \leq \alpha_j^{(1)} \leq t_{i'} \leq \phi_{l'}$, and $c_{ij} \leq c_{ir} + c_{jr} \leq 3\alpha_j^{(1)}$ using Lemma 3.10. So $c_{ii'} \leq 4\phi_{l'} \implies c_{il'} \leq 6\phi_{l'}$.

4. There is a terminal location r (could be l'), demand $j \in D_r$ such that i' is the terminal facility for r and $\beta_{ij}^{(1)} > 0$. As above, $c_{ii'} \leq 4\phi_l \implies c_{il'} \leq 6\phi_{l'}$.

■

For an open facility i, define C_i as the set of demands j for which $\beta_{ij}^{(1)} > 0$. Let $C_{F'} = \cup_{i \in F'} C_i$. Note that the sets C_i are disjoint, and all demands in C_i are assigned to i. Recall that T' is the set of Steiner edges added in Phase 1.

Lemma 3.12 $cost(T') \leq 2\sum_{j \in D'} \alpha_j^{(1)} - 2\sum_{j \in D' \cap C_{F'}} \beta_{i(j)j}^{(1)}$.

Proof : $cost(T') \leq \sum_{l \in L'} M c_{i_l l}$ where i_l is the terminal facility corresponding to l. Consider any terminal location $l \in L'$ with terminal facility i. By Lemma 3.10, $c_{il} \leq 2(\alpha_j^{(1)} - \beta_{ij}^{(1)})$ for any $j \in D_l$. Since $|D_l| \geq M$, $Mc_{il} \leq \sum_{j \in D_l} 2(\alpha_j^{(1)} - \beta_{ij}^{(1)}) = 2\sum_{j \in D_l} \alpha_j^{(1)} - 2\sum_{j \in D_l \cap C_{F'}} \beta_{i(j)j}^{(1)}$ since $\beta_{ij}^{(1)} > 0 \implies j \in C_{F'}$ and $i(j) = i$ for $j \in D_l$ by our independent set construction. Summing over all $l \in L'$ proves the lemma. ■

Lemma 3.13 *The solution obtained satisfies,* $7\sum_{i \in F'} f_i + \sum_j c_{i(j)j} + cost(T') + 2\sum_{j \in D'} c_{\sigma(j)j} \leq 7\sum_j \alpha_j^{(1)}$.

Proof : We will charge each j an amount $charge(j)$ such that

$$7\sum_{i\in F'} f_i + \sum_j c_{i(j)j} + cost(T') + 2\sum_{j\in D'} c_{\sigma(j)j} \le \sum_j charge(j) \le 7\sum_j \alpha_j^{(1)}. \tag{7}$$

$$\text{Set } charge(j) = \begin{cases} c_{i(j)j} + 7\beta^{(1)}_{i(j)j} & \text{if } j \in C_{F'} - D' \\ c_{i(j)j} + 7\beta^{(1)}_{i(j)j} + 2(\alpha_j^{(1)} - \beta^{(1)}_{i(j)j}) + 2c_{\sigma(j)j} & \text{if } j \in C_{F'} \cap D' \\ c_{i(j)j} + 2\alpha_j^{(1)} + 2c_{\sigma(j)j} & \text{if } j \in D' - C_{F'} \\ c_{i(j)j} & \text{if } j \notin D' \cup C_{F'} \end{cases}.$$

The first inequality in (7) follows from Lemma 3.12 and the fact that for each $i \in F'$, all j in C_i are assigned to i and $\sum_{j\in C_i} \beta_{ij}^{(1)} = f_i$. To prove the second inequality, note that if $j \in C_{F'}$ then $c_{i(j)j} + \beta^{(1)}_{i(j)j} \le \alpha_j^{(1)}$. If $j \in D'$ then $c_{\sigma(j)j} \le t_{\sigma(j)j} \le \alpha_j^{(1)} - \beta^{(1)}_{i(j)j}$ as argued in Lemma 3.10. Also if $j \in D' - C_{F'}$ then $c_{i(j)j} \le 3\alpha_j^{(1)}$. So if $j \in D' \cup C_{F'}$, $charge(j) \le 7\alpha_j^{(1)}$.

Consider $j \notin D' \cup C_{F'}$. We show that $c_{i(j)j} \le 7\alpha_j^{(1)}$. Let $l' \in L - L'$ be the location that caused j to become a slave and let $\sigma(j) = l \in L'$. Clearly $\alpha_j^{(1)} \ge t_{l'}$ and since $j \in D_{l'}, \alpha_j^{(1)} \ge \phi_{l'}$. Since $\sigma(j) = l$, l and l' are dependent with $\phi_l \le \phi_{l'}$, and $i(j)$ is the terminal facility corresponding to l. So by Lemma 3.11, $c_{i(j)l'} \le 6\phi_{l'}$. This implies that $c_{i(j)j} \le 7\alpha_j^{(1)}$. ■

Theorem 3.14 *The above algorithm produces a solution of total cost at most* $9 \cdot OPT$.

Proof : Let T'' be the set of Steiner edges added in Phase 2. By Lemma 3.5, $\left(\alpha', 0, \theta^{(2)}\right)$ is a feasible dual solution where $\alpha'_j = \max(\alpha_j^{(2)} - c_{\sigma(j)j}, 0)$. So $cost(T'') \le 2 \cdot OPT + 2\sum_{j\in D'} c_{\sigma(j)j}$ and the cost of tree T is at most $2 \cdot OPT + 2\sum_{j\in D'} c_{\sigma(j)j} + cost(T')$. Adding this to $\sum_{i\in F'} f_i + \sum_j c_{i(j)j}$ and using Lemma 3.13, the total cost is at most $2 \cdot OPT + 7\sum_j \alpha_j^{(1)} \le 9 \cdot OPT$. ■

As in the previous section we can get a ratio of $(7 + \rho_{ST})$ by using a ρ_{ST}-approximation algorithm ($\rho_{ST} \le 2$) in Phase 2 to build the Steiner tree on the components of T'. If C^*, S^* denote the assignment and Steiner tree costs of an optimal solution, we can obtain a Steiner tree on the components of T' by connecting each $l \in L'$ to the tree in OPT via the shortest $l - j - i^*(j)$ path for $j \in D_l$, where $i^*(j)$ is the facility to which j is assigned in OPT. This tree has cost at most $S^* + C^* + \sum_{j\in D'} c_{\sigma(j)j}$. So the total cost is at most $\sum_{i\in F'} f_i + \sum_j c_{i(j)j} + cost(T') + \rho_{ST}\sum_{j\in D'} c_{\sigma(j)j} + \rho_{ST}(S^* + C^*) \le (7 + \rho_{ST}) \cdot OPT$ by Lemma 3.13.

Theorem 3.15 *Taking* $\rho_{ST} = 1.55$, *the algorithm above produces a solution of cost at most* $8.55 \cdot OPT$.

The following results are used in Sections 4 and 5. Let F, C, S be the facility, assignment and Steiner costs of the solution obtained.

Lemma 3.16 *(i)* $14F + 2C + cost(T') + 2\sum_{j\in D'} c_{\sigma(j)j} \le 14\sum_j \alpha_j^{(1)}$, *(ii)* $F + 2C + S \le 15.55 \cdot OPT$.

Proof : Part (ii) follows from (i). Let $charge(j)$ be as defined in Lemma 3.13. The expression in (i) is at most $\sum_j charge(j) + \sum_{j\in C_{F'}}(c_{i(j)j} + 7\beta_j^{(1)}) + \sum_{j\notin C_{F'}} c_{i(j)j}$. For $j \in C_{F'}$, $c_{i(j)j} + 7\beta_j^{(1)} \le 7\alpha_j^{(1)}$, and for $j \notin C_{F'}$ as argued in Lemma 3.13, $c_{i(j)j} \le 7\alpha_j^{(1)}$. The lemma follows. ■

3.3 Extensions and Refinements

Arbitrary Demands. Suppose instead of unit demands each client j has a demand of $d_j \geq 0$. All our results extend to this case. A simple way to handle this is to make d_j copies of client j. But this only works if the demands are integer or rational, and gives a pseudo-polynomial time algorithm. We can however simulate this reduction. In phase 1, we raise each α_j at a rate of d_j. The variables $\beta_{ij}, \theta_{S,j}$ responding to the increase in α_j, also increase at rate d_j. We modify the definitions of reachability, weight of a location, ϕ_l for terminal location l, and terminal facility for l, to reflect this — we replace α_j by α_j/d_j. So j has reached i if $\alpha_j/d_j \geq c_{ij}$, $weight(i) = \sum_{j:\alpha_j \geq d_j c_{ij}} d_j$ and in the general case we again consider only demands j bound to i. In the general case, for a terminal location l, let k be the demand bound to l with smallest $\alpha_j^{(1)}/d_j$ value. We set $\phi_l = \max\left(\alpha_k^{(1)}/d_k, t_l\right)$ and the *terminal facility* for location l is the tentatively open facility to which k is connected. In phase 2, when we raise α_j and $\theta_{S,j}$, we raise them at a rate of $\frac{d_j}{\sum_{j \in D_S} d_j} \leq \frac{d_j}{M}$ so that θ_S increases at a rate of 1. The analogues of lemmas proved in sections 3.1 and 3.2 are easily shown to be true and we get the same approximation ratios.

The Case $M = 1$. We can get significantly better results for this case. In phase 1, we run the Jain-Vazirani facility location algorithm [11]. For completeness, we very briefly describe their algorithm.

We grow each dual variable α_j uniformly at rate 1. Once α_j becomes equal to c_{ij} for some facility i, we start increasing β_{ij} and start paying toward the facility opening cost of i. When the total contribution to i from the various clients equals f_i, we declare i to be tentatively open, assign all the unassigned clients tight with i to i, and freeze all these clients. The primal-dual process ends when all clients are frozen, so every client is assigned to a tentatively open facility. So at the end of this process, $\theta_{S,j}^{(1)} = 0$ for all j, S and we get a feasible dual solution $\left(\alpha^{(1)}, \beta^{(1)}, 0\right)$. At this point a client could be contributing towards multiple tentatively open facilities. We call a set of facilities *independent* if for each client j, there is at most one facility i in the set such that $\beta_{ij}^{(1)} > 0$. We select a *maximal independent subset* F' of tentatively open facilities and open all these facilities.

Let $C_{F'}$ be the set of demands j such that $\beta_{ij}^{(1)} > 0$ for some open facility i. We assign each client j in $C_{F'}$ to the unique facility $i \in F'$ such that $\beta_{ij}^{(1)} > 0$, and every other client $j \notin C_{F'}$ to the nearest facility in F'. Let $i(j)$ denote the facility to which j is assigned. For any i in F', $f_i = \sum_{j \in C_{F'}: i(j)=i} \beta_{ij}^{(1)}$; if $j \in C_{F'}$ we have $c_{i(j)j} + \beta_{i(j)j}^{(1)} = \alpha_j^{(1)}$ and the analysis in [11] shows that for $j \notin C_{F'}$, $c_{i(j)j} \leq 3\alpha_j^{(1)}$. For each $i \in F'$ we identify a client j connected to i such that $\beta_{ij}^{(1)} > 0$. Call this the *primary demand point* for i. We add edges on the path from i to j to the Steiner tree and contract these edges to form a supernode w_i. Also make v a supernode, if it is not already included in some supernode. In phase 2 a Steiner tree is built on the supernodes. Only the primary demand points pay for the Steiner tree by increasing their α_js. Let $D' \subseteq C_{F'}$ be the set of primary demand points.

Theorem 3.17 *The cost of the solution produced is at most* $4 \cdot OPT$.

Proof : By the arguing as in Lemma 3.5, we get that $\left(\alpha^{(2)}, 0, \theta^{(2)}\right)$ is now a *feasible* dual solution. The total cost is bounded by $\sum_{i \in F'} f_i + \sum_j c_{i(j)j} + \sum_{j \in D'} (c_{i(j)j} + 2\alpha_j^{(2)}) \leq \sum_{j \in D'} (2\alpha_j^{(1)} + 2\alpha_j^{(2)}) + \sum_{j \notin D'} c_{i(j)j}$. For $j \in D'$ and any i, $2\alpha_j^{(1)} + 2\alpha_j^{(2)} \leq 4c_{ij} + 2\beta_{ij}^{(1)} + 2\sum_{S \subseteq V: i \in S, v \notin S} \theta_{S,j}^{(2)}$, and for $j \notin D'$, $c_{i(j)j} \leq 3\alpha_j^{(1)} \leq 3c_{ij} + 3\beta_{ij}^{(1)}$ for any i. So the cost is at most 4 times the value of a dual feasible solution, hence at most $4 \cdot OPT$. ∎

The following results are used in Sections 4 and 5. Let (F, C, S) denote the cost of the solution obtained and T' be the set of Steiner edges added in Phase 1.

Lemma 3.18 *(i)* $3F + C + cost(T') \leq 3\sum_j \alpha_j^{(1)}$, *(ii)* $F + 2C + S \leq 6 \cdot OPT$.

Proof : $3F + C + cost(T') \leq \sum_{j \in C_{F'}} (2c_{i(j)j} + 3\beta_{i(j)j}^{(1)}) + \sum_{j \notin C_{F'}} c_{i(j)j} \leq 3\sum_j \alpha_j^{(1)}$ proving (i). The expression in part (ii) is at most $\big(F + C + cost(T')\big) + \sum_j charge(j)$ where,

$$charge(j) = \begin{cases} c_{i(j)j} + 2\alpha_j^{(2)}; & \text{if } j \in D' \\ c_{i(j)j}; & \text{if } j \notin D' \end{cases} \leq \begin{cases} 3c_{ij} + \beta_{ij}^{(1)} + 2\sum_{S \subseteq V : i \in S, v \notin S} \theta_{S,j}^{(2)} & \text{for all } i, \quad \text{if } j \in D' \\ 3c_{ij} + 3\beta_{ij}^{(1)} & \text{for all } i, \quad \text{if } j \notin D' \end{cases}.$$

So $\sum_j charge(j) \leq 3 \cdot OPT$ and from part (i), $F + C + cost(T') \leq 3 \cdot OPT$. This proves (ii). ■

4 The Connected k-Median Problem

The Connected k-Median problem is the same as ConFL with the additional constraint that at most k facilities can be be opened. Since v is already open, this extra constraint adds the following inequality to the linear program (P2) for ConFL: $\sum_{i \neq v} y_i \leq k - 1$. This changes the objective function of the dual (D2) to $\max \sum_j \alpha_j - \sum_j \beta_{vj} - k'\lambda$, where $k' = k - 1$. Constraint (6) in the dual LP gets replaced by $\sum_j \beta_{ij} \leq f_i + \lambda$. We use Phase 1 of the ConFL algorithm as a black box to get a 15.55-approximation for this problem using the technique of *Lagrangian relaxation*. This is similar in spirit to the algorithm for the k-median problem given by Jain & Vazirani [11].

Let (F^*, C^*, S^*) be the cost of an optimal connected k-median solution. Suppose we fix λ, modify the facility opening costs to $f_i + \lambda$ for all $i \neq v$, and run *only Phase 1* of the ConFL algorithm to get a (partial) primal solution (x, y, z), and a dual solution $\big(\alpha^{(1)}, \beta^{(1)}, \theta^{(1)}\big)$. Let (F, C, T') be the cost of the primal solution where T' denotes both the partial Steiner tree on F and its cost. Here $F = \sum_i f_i y_i$ is the unmodified facility cost. By a now familiar argument, the tree S^* can be extended to yield a Steiner tree on the components of T' of cost of at most $S^* + C^* + \sum_{j \in D'} c_{\sigma(j)j}$. So the total cost of building an approximate Steiner tree on the open facilities is at most,

$$T' + 1.55\big(S^* + C^* + \sum_{j \in D'} c_{\sigma(j)j}\big). \tag{8}$$

Suppose that the algorithm opens exactly k' facilities, i.e., $\sum_{i \neq v} y_i = k'$. Then, we claim that we obtain a solution of cost at most $8.55 \cdot OPT$. To see this, note that $\big(\alpha^{(1)}, \beta^{(1)}, \theta^{(1)}, \lambda\big)$ is a feasible solution to the dual of the connected k-median LP and by Lemma 3.13, $7(F + k'\lambda) + C + T' + 2\sum_{j \in D'} c_{\sigma(j)j} \leq 7\alpha_j^{(1)} \implies 7F + C + T' + 2\sum_{j \in D'} c_{\sigma(j)j} \leq 7\big(\sum_j \alpha_j^{(1)} - k'\lambda\big) \leq 7 \cdot OPT$. Combining this with (8), we can bound the total cost by $7F + C + T' + 2\sum_{j \in D'} c_{\sigma(j)j} + 1.55(S^* + C^*) \leq 7 \cdot OPT + 1.55 \cdot OPT = 8.55 \cdot OPT$. The trick then is to *guess* the right value of λ so that when the facility cost is updated to $f_i + \lambda$, we end up opening k facilities. This idea was first used in [11].

Suppose the algorithm opens at most k' facilities when $\lambda = 0$. Then, since $\big(\alpha^{(1)}, \beta^{(1)}, \theta^{(1)}, 0\big)$ is a feasible connected k-median dual solution, we get a solution of cost at most $8.55 \cdot OPT$. So assume that at $\lambda = 0$ the algorithm opens greater than k' facilities. When λ is large, say, $|\mathcal{D}| \max_j c_{vj}$, the algorithm will connect all demands to v and not open any other facility. We can show that there is a value $\lambda = \lambda_0$ such that depending on how we break ties between events, we get two primal solutions — one opening $k_1 < k'$ facilities and the other opening $k_2 > k'$ facilities, and a single dual solution. These two solutions can be found in polynomial time by performing a bisection search in the range $\big[0, |\mathcal{D}| \max_j c_{vj}\big]$ and terminating the search when the length of the search interval becomes less than $2^{-(\text{poly}(n)+L)}$ where L is the number of

bits to represent the longest edge[2]. The proof of this is very similar to the proof in the conference version of [11] (see Section 3.2), so we only sketch the proof briefly at the end of this Section.

Let (x_1, y_1, z_1) and (x_2, y_2, z_2) be the two solutions obtained at $\lambda = \lambda_0$, and $\left(\alpha^{(1)}, \beta^{(1)}, \theta^{(1)}\right)$ be the common dual solution. It is important to note that the values $\alpha^{(1)}, \beta^{(1)}, \theta^{(1)}, \lambda_0$ *are used only in the analysis, we do not need them to specify the algorithm.* Let (F_1, C_1, T_1') and (F_2, C_2, T_2') denote the cost of the solutions (x_1, y_1, z_1) and (x_2, y_2, z_2) respectively. A *convex combination* of the two solutions yields a fractional solution (x, y, z) that opens exactly k' facilities. Let $ak_1 + bk_2 = k', a + b = 1$. To avoid cumbersome notation, let A denote the quantity $2\sum_{j \in D'} c_{\sigma(j)j}$ in the solution (x_1, y_1, z_1) and B denote the corresponding quantity in (x_2, y_2, z_2). Then,

$$7(aF_1 + bF_2) + (aC_1 + bC_2) + (aT_1' + bT_2') + aA + bB \leq 7\Big(\sum_j \alpha_j^{(1)} - k'\lambda\Big) \leq 7 \cdot OPT. \qquad (9)$$

We now round (x, y, z) to get a solution that opens at most k facilities (including v) losing a factor of 2.

If $a \geq \frac{1}{2}$ we take the solution (x_1, y_1, z_1) and from (9) we get that $F_1 + C_1 + T_1' + A \leq 14 \cdot OPT$. Otherwise we open a subset of the facilities opened by y_2 as in [11] to get a solution of assignment cost at most $2(aC_1 + bC_2)$. For each facility $i \in y_1$ (i.e., opened in y_1) we look at the facility in y_2 closest to it. Let N be this set of facilities. If $|N| < k_1$ we arbitrarily add facilities from y_2 to N till $|N| = k_1$. We open all the facilities in N. We also randomly pick a set of $k' - k_1$ facilities opened by y_2 but not in N, and open these. Note that each such facility is opened with probability $(k' - k_1)/(k_2 - k_1) = b$. We also add edges of T_2' corresponding to the open facilities.

For a demand j, let i_1, i_2 be the facilities to which it is assigned in y_1, y_2 respectively. Let i_3 be the facility nearest to i_1 in y_2. Note that i_3 is always opened. We assign j to i_2 if it is open and to i_3 otherwise. Since $c_{i_3 j} \leq c_{i_1 j} + c_{i_1 i_3} \leq c_{i_1 j} + c_{i_1 i_2} \leq 2c_{i_1 j} + c_{i_2 j}$ and $a < b$, the expected assignment cost is at most, $bc_{i_2 j} + ac_{i_3 j} \leq 2(ac_{i_1 j} + bc_{i_2 j})$. So the total assignment cost is at most $2(aC_1 + bC_2)$. From (9), $F_2 + 2(aC_1 + bC_2) + T_2' + B \leq 14 \cdot OPT$.

Completing the Steiner tree on the open facilities costs an additional $1.55(S^* + C^*) \leq 1.55 \cdot OPT$, so the total cost is at most $15.55 \cdot OPT$. Also if (F, C, S) denotes the cost of the solution returned, then using Lemma 3.16 and arguing as above we get that $F + 2C + S \leq 29.55 \cdot OPT$.

Theorem 4.1 *There is a 15.55-approximation algorithm for the Connected k-Median problem. Further, the solution returned of cost (F, C, S) satisfies $F + 2C + S \leq 29.55 \cdot OPT$.*

When $M = 1$ if we use the ConFL algorithm for $M = 1$ in the above procedure, we first get a partial solution of cost (F, C, T') such that $F + C + T' \leq 6 \cdot OPT$ using Lemma 3.18. Building an approximate Steiner tree on the components of T' costs at most $1.55(S^* + C^*)$ since an optimal Steiner tree on the components of T' has cost at most $S^* + C^*$.

Theorem 4.2 *When $M = 1$, we get a solution of cost (F, C, S) such that $F + C + S \leq 7.55 \cdot OPT$ and $F + 2C + S \leq 13.55 \cdot OPT$.*

4.1 Obtaining the solutions (x_1, y_1, z_1) and (x_2, y_2, z_2)

For a given value of λ, let the sequence for λ denote the list of events occurring in Phase 1 of the primal-dual algorithm arranged in non-decreasing order of the time at which they take place, with ties broken in an arbitrary, but fixed way. We say that λ is a *critical point* if an infinitesimal change in λ results in a change in the sequence. One can show that if λ_0 is a critical point then both the sequence for λ_0 and the sequence for $\lambda_0 \pm \epsilon$ can be obtained *at $\lambda = \lambda_0$ depending on how we break ties between events*. Furthermore, two critical

[2]Note that it only takes a polynomial number of bits to represent this length, so the bisection search runs in polynomial time.

points are separated by at least $2^{-(\text{poly}(n)+L)}$, where L is the number of bits to represent the longest edge. Suppose we terminate the bisection search when the search interval $[\lambda_2, \lambda_1]$ satisfies $\lambda_1 - \lambda_2 < 2^{-(\text{poly}(n)+L)}$ and $(x_1, y_1, z_1), (x_2, y_2, z_2)$ are the (partial) primal solutions at λ_1, λ_2 respectively that open $k_1 < k'$ and $k_2 > k'$ facilities respectively. We know that there is a single critical point $\lambda_0 \in [\lambda_1, \lambda_2]$, so by the above property there is a way of breaking ties between events so that we get both (x_1, y_1, z_1) and (x_2, y_2, z_2) as solutions at $\lambda = \lambda_0$. These two solutions, which can be found in polynomial time, satisfy all the required properties. Note that we do not explicitly need to find the value of λ_0 or the dual solution $\left(\alpha^{(1)}, \beta^{(1)}, \theta^{(1)}\right)$ at $\lambda = \lambda_0$.

5 Generalization to Edge Capacities

We can extend our results to a capacitated generalization of connected facility location where edges have capacities. Each edge has a *length* c_e. We are given two kinds of cables; one having a cost of σ per unit length and capacity u, the other has a cost of M per unit length and infinite capacity. We wish to open facilities and lay a network of cables so that clients are connected to open facilities using the first kind of cable. Further we want the facilities to be connected to each other by a Steiner tree using cables of the second type. We may install multiple copies of a cable along an edge, if necessary, to handle the total demand through the edge. So routing d units of demand through edge e now costs $\sigma\lceil \frac{d}{u} \rceil c_e$ while earlier the cost was simply $d \cdot c_e$. Assuming integer demands, the uncapacitated problem considered earlier is a special case obtained by setting $u = 1$ and scaling edge costs by σ, M by $\frac{1}{\sigma}$. The facility location aspect of this problem where we only have cables of the first type and do not require that facilities be interconnected was considered in [22].

The rent-or-buy case with $\mathcal{F} = V, f_i = 0$ for all i now corresponds to a rent-or-buy problem where we can either buy unlimited capacity on an edge paying a large fixed cost of M per unit length, or rent capacity in steps of u units, paying a cost of σ per unit length for every u units installed.

Let us first consider unit demands. We assume $\sigma \leq M$ (otherwise the optimal solution is just a Steiner tree connecting the clients to v). We will use a Theorem proved by Hassin et al. [10] (see also [22]) in a slightly different form.

Theorem 5.1 *Let Z be a Steiner tree on a set of terminals D rooted at v where each edge has capacity u. Let w_j be a weight associated with terminal $j \in D$. We can clump the terminals into subtrees $Z_1, \ldots, Z_k$ so that,*

(i) Each subtree except possibly Z_k has exactly u terminals and Z_k has at most u terminals.

(ii) If we route flow along edges of Z from the $u-1$ terminals in Z_i to the terminal in Z_i with minimum weight for each $i < k$, and route flow from the terminals in Z_k to v, then we get a flow that respects edge capacities.

If terminal j has a demand $d_j < u$, we can clump the terminals so that each subtree Z_i has demand between u and $2u$ for $i < k$, Z_k has at most u units of demand, and (ii) still holds.

We can get a $(\rho_{ConFL}+\rho_{ST})$-approximation algorithm for this problem by using a ρ_{ConFL}-approximation algorithm for ConFL and a ρ_{ST}-approximation algorithm for the Steiner tree problem.

1. Obtain a ConFL instance by setting the edge costs to $c'_e = \frac{\sigma c_e}{u}$ and $M' = \frac{Mu}{\sigma}$. A solution to the original instance gives a solution to the ConFL instance of no greater cost — the Steiner edges cost the same and the cost of routing d units of demand through a facility location edge is $d \cdot \frac{\sigma c_e}{u} \leq \sigma\lceil \frac{d}{u} \rceil c_e$. We solve this relaxation approximately using the ρ_{ConFL}-approximation algorithm. Let $i(j)$ be the facility to which j is assigned and T be the Steiner tree on the open facilities.

Connected Facility Location

		$u=1$	$u>1$
Unit demands	general case	8.55	10.1
	rent-or-buy	4.55	6.1
	$M=1$	4	5.55
Arbitrary demands	general case	8.55	17.1
	rent-or-buy	4.55	9.1
	$M=1$	4	7.55

Connected k-Median Problem

		$u=1$	$u>1$
Unit demands	general case	15.55	17.1
	$M=1$	7.55	9.1
Arbitrary demands	general case	15.55	31.1
	$M=1$	7.55	15.1

Table 5.1: Summary of the results of Sections 3, 4 and 5.

2. Obtain a Steiner tree instance by setting the edge costs to σc_e with the terminals being the demand points and vertex v. This is a relaxation, since a solution to the original instance connects all demand points to open facilities and all open facilities to v with each edge costing at least σc_e, be it a facility location edge or a Steiner tree edge (since $M \geq \sigma$). We solve this Steiner tree instance approximately. Let Z be the resulting tree.

3. Now we combine the two approximate solutions to get a feasible solution of cost no greater than the sum of the costs of the two solutions. We use the above Theorem with the tree Z and $w_j = c'_{i(j)j}$ for demand j. Let $Z_1, \ldots, Z_k$ be the subtrees obtained. We first route demand in each subtree along edges of Z as specified in the Theorem. For each subtree $Z_i, i < k$ the u units of demand collected at the client $j \in Z_i$ for which $c'_{i(j)j}$ is minimum is then sent to facility $i(j)$ along the path from j to $i(j)$.

The cost of routing demand along Z is at most the cost of Z in the Steiner tree instance since each edge of Z carries at most u units of demand. Routing demand along the path from $j \in Z_i$ to $i(j)$ costs $\sigma c_{i(j)j} \leq \sum_{k \in Z_i} \frac{\sigma c_{i(k)k}}{u} = \sum_{k \in Z_i} c'_{i(k)k}$. The only facilities we use are v and the facilities opened in the ConFL solution and these are connected by the tree T that costs the same in both the original instance and the ConFL instance. So we get a feasible solution of cost at most $(\rho_{ConFL} + \rho_{ST}) \cdot OPT$. Taking $\rho_{ST} = 1.55$ [23] and using Theorems 3.7, 3.15 and 3.17 we have,

Theorem 5.2 *There is a 10.1-approximation algorithm for Connected Facility Location with edge capacities and unit demands. For the case $\mathcal{F} = V$ and $f_i = 0$ for all i, there is a 6.1-approximation algorithm. If $M = 1$, we get a ratio of 5.55 in both cases.*

For arbitrary demands, the algorithm above works with the same guarantee if demands may be split across facilities. For the unsplittable case we get a somewhat worse guarantee. We approximately solve the ConFL and Steiner tree instances as above. Let (F, C, S) denote the cost of the ConFL solution. Clients with demand at least u send their demand directly to the facility serving it in the ConFL instance. The cost incurred is at most twice the connection cost incurred by the ConFL instance. Again using Theorem 5.1 and routing demand as above we get a feasible solution of cost at most $F + 2C + S + \rho_{ST} \cdot OPT$. Using Lemmas 3.8, 3.16 and 3.18 we get a bound on $F + 2C + S$ that is better than the naive bound of $2\rho_{ConFL} \cdot OPT$, and obtain the following theorem.

Theorem 5.3 *There is a 17.1-approximation algorithm for Connected Facility Location with edge capacities and arbitrary demands. When $\mathcal{F} = V$ and $f_i = 0$ for all i, we get an approximation ratio of 9.1. If $M = 1$ the ratio improves to 7.55 in both cases.*

We can use the algorithm for the connected k-median problem from the previous section as a black box to get a constant-factor approximation-algorithm for the Connected k-Median problem with edge capacities. Using Theorem 4.1 we get a 17.1- and 31.1-approximation algorithm for unit demands and arbitrary

demands respectively. For $M = 1$, the approximation ratios improve to 9.1 and 15.1 respectively using Theorem 4.2. Table 5.1 summarizes all the results.

Acknowledgments

We thank David Shmoys and Jon Kleinberg for useful discussions, reading through drafts of this paper and pointing out suggestions. We also thank the anonymous referee for useful suggestions.

References

[1] A. Agrawal, P. Klein, and R. Ravi. When trees collide: An approximation algorithm for the generalized Steiner problem on networks. *SIAM Journal on Computing*, 24(3):440–456, 1995. Preliminary version in *STOC '91*.

[2] M. Andrews and L. Zhang. The access network design problem. In *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 40–49, 1998.

[3] M. X. Goemans and D. P. Williamson. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24:296–317, 1995. Preliminary version in *SODA '94*.

[4] M. X. Goemans and D. P. Williamson. The primal-dual method for approximation algorithms and its application to network design problems. In D. S. Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*, chapter 4, pages 144–191. PWS Publishing Company, 1997.

[5] S. Guha and S. Khuller. Connected facility location problems. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 40:179–190, 1997.

[6] S. Guha, A. Meyerson, and K. Munagala. A constant factor approximation for the single sink edge installation problems. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 383–388, 2001.

[7] S. Guha, A. Meyerson, and K. Munagala. Hierarchical placement and network design problems. In *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 603–612, 2000.

[8] A. Gupta, J. Kleinberg, A. Kumar, R. Rastogi, and B. Yener. Provisioning a virtual private network: A network design problem for multicommodity flow. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 389–398, 2001.

[9] A. Gupta, A. Kumar, and T. Roughgarden. Simple and better approximation algorithms for network design. To appear in *Proceedings of the 35th Annual ACM Symposium on Theory of Computing (STOC)*, 2003.

[10] R. Hassin, R. Ravi, and F. S. Selman. Approximation algorithms for a capacitated network design problem. In *Proceedings of the 3rd International Workshop on Approximation Algorithms for Combinatorial Opimization (APPROX)*, pages 167–176, 2000.

[11] K. Jain and V. V. Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation. *Journal of the ACM*, 48(2):274–296, 2001. Preliminary version in *FOCS '99*.

[12] D. R. Karger and M. Minkoff. Building Steiner trees with incomplete global knowledge. In *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 613–623, 2000.

[13] S. Khuller and A. Zhu. The general Steiner tree-star problem. *Information Processing Letters*, 2002. To appear.

[14] Tae Ung Kim, Timothy J. Lowe, Arie Tamir, and James E. Ward. On the location of a tree-shaped facility. *Networks*, 28(3):167–175, 1996.

[15] C. Krick, H. Räcke, and M. Westermann. Approximation algorithms for data management in networks. In *Proceedings of the 13th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 237–246, 2001.

[16] A. Kumar, A. Gupta, and T. Roughgarden. A constant-factor approximation algorithm for the multicommodity rent-or-buy problem. In *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 333–342, 2002.

[17] A. Kumar, R. Rastogi, A. Silberschatz, and B. Yener. Algorithms for provisioning virtual private networks in the hose model. In *Proceedings of the Annual ACM Conference of the Special Interest Group on Data Communication (SIGCOMM)*, pages 135–146, 2001.

[18] M. Labbé, G. Laporte, I. Rodrígues Martin, and J. J. Salazar González. The median cycle problem. Technical Report 2001/12, Department of Operations Research and Multicriteria Decision Aid at Université Libre de Bruxelles, 2001.

[19] Y. Lee, S. Y. Chiu, and J. Ryan. A branch and cut algorithm for a Steiner tree-star problem. *INFORMS Journal on Computing*, 8(3):194–201, 1996.

[20] P. Mirchandani and R. Francis, eds. *Discrete Location Theory*. John Wiley and Sons, Inc., New York, 1990.

[21] R. Ravi and F. S. Selman. Approximation algorithms for the traveling purchaser problem and its variants in network design. In *Proceedings of the 7th Annual European Symposium on Algorithms (ESA)*, pages 29–40, 1999.

[22] R. Ravi and A. Sinha. Integrated logistics : Approximation algorithms combining facility location and network design. In *Proceedings of the 9th Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 212–229, 2002.

[23] G. Robins and A. Zelikovsky. Improved steiner tree approximation in graphs. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 770–779, 2000.

[24] C. Swamy and A. Kumar. Primal-dual algorithms for connected facility location problems. In *Proceedings of the 5th International Worksohp on Approximation Algorithms for Combinatorial Opimization (APPROX)*, pages 256–269, 2002.

[25] K. Talwar. The single-sink buy-at-bulk LP has constant integrality gap. In *Proceedings of the 9th Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 475–486, 2002.

[26] D. P. Williamson. The primal-dual method for approximation algorithms. *Mathematical Programming, Series B*, 91(3):447–478, 2002.