Primal-Dual Algorithms for Connected Facility Location Problems

Chaitanya Swamy* and Amit Kumar**

Department of Computer Science, Cornell University, Ithaca, NY 14853, USA
{swamy, amitk}@cs.cornell.edu

Abstract. We consider the *Connected Facility Location* problem. We are given a graph $G = (V, E)$ with cost c_e on edge e, a set of facilities $\mathcal{F} \subseteq V$, and a set of demands $\mathcal{D} \subseteq V$. We are also given a parameter $M \geq 1$. A solution opens some facilities, say F, assigns each demand j to an open facility $i(j)$, and connects the open facilities by a Steiner tree T. The 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 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 this problem reduces to 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 9-approximation algorithm for connected facility location and a 5-approximation for the rent-or-buy problem. Our algorithm integrates the primal-dual approaches for facility location [7] and Steiner trees [1, 2]. 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 version of these problems.

1 Introduction

Facility location problems have been widely studied in the Operations Research community(see for e.g. [14]). 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 demands $\mathcal{D} \subseteq V$. Facilities may have *opening costs*. We want to *open* some facilities from the set $\mathcal{F}$ and assign each demand to one of these open facilities. We consider a setting where besides opening the facilities, we also want to connect them by a Steiner tree. This will allow the facilities to communicate easily with each other. For example, the facilities could be caches or file servers which need to communicate with each other to maintain consistent data, and the clients could be users or processes requesting data items. Another example is telecommunication network design. Designing the network involves selecting a subset of core nodes, connecting the selected core nodes, and routing traffic from the endnodes to the

* Research partially supported by NSF grant CCR-9912422.
** Research supported by NSF grant CCR-9820951 and NSF ITR/IM grant IIS-0081334.

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.

The problems mentioned above are instances of the *Connected Facility Location* problem (ConFL). We are given a graph $G = (V, E)$ with cost c_e on edge e, a set of facilities $\mathcal{F} \subseteq V$ and a set of demand nodes or clients $\mathcal{D} \subseteq V$. Let c_{ij} be the shortest path distance between i and j (with respect to the costs c_e). We are also given a parameter $M \geq 1$. Client j has d_j units of demand and facility i has an opening cost of f_i. A solution has to *open* a set of facilities F, assign each demand j to an open facility $i(j)$, and further has to 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. The total cost of this solution is $\sum_{i \in F} f_i + \sum_{j \in \mathcal{D}} d_j c_{i(j)j} + M \sum_{e \in T} c_e$. We want to find a solution of minimum cost. This problem has attracted the interest of both the operations research community [10, 12, 13] and the computer science community [5, 8, 9].

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., $\mathcal{F} = V$. Suppose we *know* that facility v is opened by an 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 we want to route traffic in a minimum-cost way from the clients to the sink v by installing capacity on edges. We can either *rent* capacity on an edge paying a cost proportional to the capacity rented, or *buy* unlimited capacity on an edge by paying a large fixed cost of M.

This problem arises in various scenarios. Karger & Minkoff [8] reduced the *maybecast* problem to this special case of ConFL. Gupta et. al. [5] arrived at this problem by considering the problem of provisioning a virtual private network.

Our Results. We give a primal-dual 9-approximation algorithm for the connected facility location problem and a 5-approximation algorithm for the rent-or-buy problem. Previously the best known approximation guarantees for these problems were 10.66 and 9.001 respectively [5]. But these results were obtained by solving an exponential size linear program using the ellipsoid method, making the algorithm very inefficient in practice. Karger & Minkoff [8] gave a combinatorial algorithm, but the constant guarantee was much larger.

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 20-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 [4]. 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 facility location and the Steiner tree problem. Without the connectivity requirement, the problem is simply the uncapacitated facility location problem. If we know which facilities are open we only need to connect them by a Steiner tree. However, simply running a facility location algorithm and then a Steiner tree algorithm does not work, since we are ignoring the connectivity requirement. 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. The connectivity requirement implicitly imposes a facility opening cost so that it is only profitable to open a facility if it serves a significant demand. Previously [8] the clustering of demands around facilities 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 with this approach is that (1) we only know a bicriteria approximation for LBFL, so the demand lower bound on a facility is only approximately satisfied, and (2) the LBFL instance is solved using a black box, so we do not use anything specific to the ConFL instance. In particular we make no use of the fact that the need to cluster demands is imposed by the connectivity requirement of ConFL.

Our algorithm is based on a novel application of the primal-dual schema. The algorithm is in two phases. First, we decide which facilities to open, connect demands to facilities, and cluster demands at each open facility. At the end of this phase, we obtain a feasible dual solution and a primal facility location solution where 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 open facilities. 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 the infeasibility is bounded by a small *additive* factor.

Very recently, Kumar et al. [11] have subsequently obtained a constant-factor approximation for a multicommodity rent-or-buy problem. Their algorithm is however much more involved and they get a much worse approximation factor. A very interesting open problem is to see whether the techniques used here and in [11] can be extended to solve the multiple source-sink buy-at-bulk problem with multiple cable types.

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 [3]) 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 [2] further refined the primal-dual method and extended it to a large class of network-design problems; see [3, 17] 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 [7] 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.

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, and connection cost and assignment cost interchangeably.

ConFL can be formulated naturally as an Integer Program. Suppose we *know* that a facility v 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 :

$$\min \quad \sum_i f_i y_i + \sum_j d_j \sum_i c_{ij} x_{ij} + M \sum_e c_e z_e \tag{IP}$$

$$\text{s.t.} \quad \sum_i x_{ij} \geq 1 \qquad \text{for all } j \tag{1}$$

$$x_{ij} \leq y_i \qquad \text{for all } i, j \tag{2}$$

$$y_v = 1 \tag{3}$$

$$\sum_{i \in S} x_{ij} \leq \sum_{e \in \delta(S)} z_e \qquad \text{for all } S \subseteq V, v \notin S,\ j \tag{4}$$

$$x_{ij}, y_i, z_e \in \{0, 1\} \tag{5}$$

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 (5) 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 5.

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 :

$$\begin{aligned} \max \quad & \sum_j \alpha_j && && \text{(D1)} \\ \text{s.t.} \quad & \alpha_j \leq c_{ij} + \sum_{S \subseteq V : i \in S, v \notin S} \theta_{S,j} && \text{for all } i \neq v, j && (6) \\ & \alpha_j \leq c_{vj} && \text{for all } j && (7) \\ \sum_j \sum_{S \subseteq V : e \in \delta(S), v \notin S} & \theta_{S,j} \leq M c_e && \text{for all } e && (8) \\ & \alpha_j, \theta_{S,j} \geq 0 \end{aligned}$$

Intuitively, α_j is the *payment* that demand j is willing to make towards constructing a feasible primal solution. Constraint (6) 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 which 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 vertices in $\mathcal{F}$. Clearly the metric c can be extended to a metric on locations.

The intuition behind our algorithm is as follows. Suppose all demands were of size at least M. Then, the optimal solution would locate a facility at each of these demands 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 $t = 0$. 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 which j is tight with 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 (6). So, it is enough to describe how to raise the dual variables α_j.

Initially, all locations are closed. We shall *tentatively open* some locations. Initially v is tentatively open. Demands can be in two states : *frozen* or *unfrozen*. When a demand j gets frozen, we stop raising its dual variable α_j. 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 the same rate until one of the following events happen (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 till all demands become frozen. 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 j, the portion of the edge that is tight with j.

Now we decide which locations to open. Let F' be the set of tentatively open locations. We say that $i, i' \in F'$ 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 F of locations in F' as follows : arrange the locations in F' in the order they were tentatively opened. Consider the locations in this order and add a location to F if no dependent location is already present in F. We open the locations in F. Observe that $v \in F$.

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

We still have to build a Steiner tree on F. 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 $i \in F$, let D_i be the set of demands tight with i. Let $D' = \bigcup_{i \in F - \{v\}} D_i$. Initially, 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 $i \in F - \{v\}$. For a set S, define $D_S = \bigcup_{i \in S \cap F} D_i$. 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 (6), (7).

We say that an edge goes tight if (8) 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 perform a reverse delete step to remove any redundant edges from T. This is our final solution.

Analysis

Let (α^1, θ^1), (α^2, θ^2) be the value of the dual variables at the end of Phases 1 and 2 respectively.

Lemma 3.1. *The dual solution (α^1, θ^1) is feasible.*

Proof. It is easy to see that (6) 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, (7) is satisfied.

Now consider an edge $e = (u, w)$. Let $l(j)$ be the contribution of j to the left hand side of (8) for this edge, i.e., $l(j) = \sum_{S:e\in\delta(S), v\notin S} \theta_{S,j}$. 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 (8), we get

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

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

Lemma 3.2. *At the end of Phase 1, demand j is assigned to an open location i such that $c_{ij} \leq 3\alpha_j^1$.*

Proof. This clearly holds if j is tight with a location in F. 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 $\alpha_j \geq t_{i'}$.

Now, $c_{ij} \leq c_{ik} + c_{ki'} + c_{i'j} \leq 2\alpha_k^1 + \alpha_j^1$. Also, $\alpha_k^1 \leq t_{i'}$. Otherwise, at time $t = \alpha_k^1$, 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., $\alpha_k^1 \leq t_i \leq t_{i'}$. So, $\alpha_k^1 \leq t_{i'} \leq \alpha_j$. This implies that $c_{ij} \leq 3\alpha_j^1$. □

Lemma 3.3. *Let i be an open location. If j is tight with i, then the assignment cost of j is at most α_j^1.*

We now bound the cost of the tree T. Define D_V as $\cup_{i \in F} D_i$.

Lemma 3.4. $cost(T) \leq 2 \cdot \sum_{j \in D_V} \alpha_j^2$.

Proof. Consider Phase 2. At any point in 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 $2 \cdot \sum_S \theta_S^2$ [3, 1, 17], where the sum is over all subsets of vertices S. But $\sum_S \theta_S^2 = \sum_{j \in D_V} \alpha_j^2$. □

Lemma 3.5. *Consider a demand j. If $i \neq v$, then $\alpha_j^2 \leq \alpha_j^1 + c_{ij} + \sum_{S \subseteq V : i \in S, v \notin S} \theta_{S,j}^2$. Further, $\alpha_j^2 \leq \alpha_j^1 + c_{vj}$.*

Proof. Fix a demand j 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 (6) increase at the same rate, so we only need to bound the increase in α_j by time t'. Let l be the location that j is assigned to in Phase 1. Since we are raising α_j, it must be the case that $j \in D_l$ and so, $c_{lj} \leq \alpha_j^1$. We claim that $t' \leq M c_{li}$. This is true since S_t always contains l, and by time $t = M c_{li}$ all of the edges along the shortest path between l and i would have grown tight.

Note that α_j rises at a rate of at most $1/M$. Indeed, initially, $\left|D_{\{i\}}\right| \geq M$ for any open location i, and as new components S form, $|D_S|$ can only increase. So, the increase in α_j by time t' can then be bounded by $\frac{M c_{li}}{M} \leq c_{lj} + c_{ij} \leq \alpha_j^1 + c_{ij}$. This proves the first inequality. The second inequality is proved similarly. □

It is clear that the $\theta_{S,j}^2$ values satisfy (8), so we have shown that $\left(\alpha', \theta^2\right)$ is a feasible dual solution, where $\alpha_j' = \max(\alpha_j^2 - \alpha_j^1, 0)$. We can now prove the main theorem. Let OPT be the cost of the optimal solution.

Theorem 3.1. *The above algorithm produces a solution of cost at most $5 \cdot OPT$.*

Proof. Note that $\alpha_j^2 \leq \alpha_j' + \alpha_j^1$. So, Lemma 3.4 implies that the cost of T is at most $2 \sum_j \alpha_j' + 2 \sum_{j \in D_V} \alpha_j^1 \leq 2 \cdot OPT + 2 \sum_{j \in D_V} \alpha_j^1$.

If $j \in D_V$, Lemma 3.3 implies that its assignment cost is at most α_j^1. Otherwise by Lemma 3.2, its assignment cost is at most $3\alpha_j^1$. Adding all terms, we see that the cost of our solution is at most $5 \cdot OPT$. □

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_u be the set of demands that reach their assigned location on e via u, i.e., $c_{i(j)j} = c_{uj} + c_{i(j)u}$ for $j \in D_u$. D_w is defined similarly. 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 $u \in T, w \notin T$. 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. It is easy to see that the total cost only decreases.

3.2 The General Case

We now consider the case where $\mathcal{F}$, need not be V and facility i has an opening cost $f_i \geq 0$. For convenience we assume that $f_v = 0$. Clearly, this does not affect the approximation ratio of the algorithm. The primal and dual LPs are :

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

$$\begin{aligned}
\max \quad & \sum_j \alpha_j - \sum_j \beta_{vj} && && \text{(D2)}\\
\text{s.t.} \quad & \alpha_j \leq c_{ij} + \beta_{ij} + \sum_{S \subseteq V: i \in S, v \notin S} \theta_{S,j} && \text{for all } i \neq v, j && (9)\\
& \alpha_j \leq c_{vj} + \beta_{vj} && \text{for all } j && (10)\\
& \sum_j \beta_{ij} \leq f_i && \text{for all } i \neq v && (11)\\
& \sum_j \sum_{S \subseteq V: e \in \delta(S), v \notin S} \theta_{S,j} \leq M c_e && \text{for all } e && (12)\\
& \alpha_j, \beta_{ij}, \theta_{S,j} \geq 0 && &&
\end{aligned}$$

Phase 1. Most of the changes are in this phase. We now also have to pay for opening facilities. Besides opening facilities and connecting clients to facilities, we will also form some components. These will act as the terminals for the Steiner tree constructed in Phase 2. 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. As before, a demand can be frozen or unfrozen. Further, a demand may be *connected* or *unconnected*. Initially, all demands are unfrozen and unconnected. As before, we say that a demand j gets tight with a location i if $\alpha_j \geq c_{ij}$. We say that a facility i has been *paid* for if $\sum_j \beta_{ij} = f_i$. The *weight* of a location l is defined as the number of *connected* demands j which are tight with l.

The basic idea is similar to the algorithm in the previous section. Earlier we tentatively opened any location with which M demands became tight, but we cannot do that here because of two reasons — (1) we cannot open *any* location; the set of candidate facillity locations, $\mathcal{F}$, may be a very small subset of V, (2) we need to pay a facility opening cost before we can open a facility.

At any point of time, define S_j to be the set of facilities that a demand j is tight with. When j becomes tight with a facility i, we have two options – we can raise β_{ij} or we can raise $\theta_{S_j,j}$. If none of the facilities in S_j have been paid for, we raise β_{ij} for all $i \in S_j$ at the same rate. If there is a facility $i \in S_j$ which has been paid for, then we raise $\theta_{S_j,j}$ and do not raise β_{ij} for any $i \in S_j$. Thus, it is enough to describe how the α_js get raised.

We now describe the algorithm in more detail. We raise the α_j of all unfrozen demands uniformly till one of the following events happen :

1. An unconnected demand j becomes tight with a tentatively open facility i : j becomes connected to i. If $i = v$, freeze j. Otherwise, as described above, we raise $\theta_{S_j,j}$ at the same rate as α_j. Further we do not raise any variable $\beta_{i'j}$ for any facility i' from now on.
2. A facility i gets paid for, i.e., $\sum_j \beta_{ij} = f_i$: tentatively open i. If an unconnected demand j is tight with i, connect j to i. From this point on we only raise $\theta_{S_j,j}$ as described above.
3. The weight of some location l becomes at least M : declare l to be a *terminal location*. Freeze all unfrozen demands which are tight with l.
4. A connected demand j becomes tight with a terminal location l : freeze j.

We continue this process until all j become frozen. Let $(\alpha^1, \beta^1, \theta^1)$ be the dual solution obtained. Note that β^1_{vj} is 0 for all j.

Let L be the set of all terminal locations. As in the previous section, we greedily select an independent set of terminal locations from L and assign j to a terminal location $\sigma(j)$. We say that locations l and l' in L are dependent if there is a demand tight with both these locations. We look at the locations in L in the order they were declared to be a terminal location, and greedily select a maximal independent subset L' of L. If demand j is tight with a location $l' \in L'$, set $\sigma(j) = l'$. Otherwise let l be a location in L that caused j to get frozen, and $l' \in L'$ be some location such that l and l' are dependent. Set $\sigma(j) = l'$. Note that if j is tight with v, $\sigma(j) = v$.

Now, consider a location $l \in L'$. l may not be in the set $\mathcal{F}$ of candidate facilities. So, we need to locate a facility $i \in \mathcal{F}$ *near* l and open it. Let j be the demand tight with l having the smallest value of α^1_j. j is connected to a tentatively open facility i. Call i a *terminal facility*. We say that i is the terminal facility corresponding to the terminal location l. Let F be the set of all terminal facilities. Add v to F. Again, we have a notion of dependence among facilities in F. We say that two facilities i, i' are dependent if there is a demand j with both $\beta^1_{ij}, \beta^1_{i'j} > 0$. We select a maximal independent set from F – call it F'. Note that $v \in F'$ because $\beta^1_{vj} = 0$ for all j. We open all the facilities in F'.

A demand j is assigned to a facility in F' as follows : if there is a facility $i \in F'$ such that $\beta^1_{ij} > 0$, assign j to i. If $\sigma(j) = v$, assign j to v. Otherwise, let i be the be terminal facility corresponding to the terminal location $\sigma(j) \in L'$. If $i \in F'$, assign j to i. Otherwise, there is a facility $i' \in F'$ such that i and i' are dependent. We assign j to i'. Let $i(j)$ be the facility that j is assigned to.

Let L_1 be the terminal locations in L' such that the terminal facilities corresponding to them are in F'. We now add some Steiner edges. We initialize the Steiner tree T to the empty set. For each terminal location $l \in L_1$ with corresponding terminal facility $i \in F'$, we add all edges along a shortest path between l and i to the set T. Break any cycles by deleting edges.

Phase 2. This phase is very similar to that of the previous section. For any $l \in L$, let D_l be the set of demands which are tight with l. Define $D_{L_1} = \bigcup_{l \in L_1} D_l$. G is augmented as before to include edges incident on locations $l \in L_1$. 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_{L_1} only. For a set S, define D_S to be $\bigcup_{l \in S \cap L_1} D_l$. The rest of the procedure is identical to Phase 2 of the previous section. This yields the tree T connecting all the open facilities. Let (α^2, θ^2) be the dual solution constructed by this phase.

Analysis

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

Lemma 3.6. $\left(\alpha^1, \beta^1, \theta^1\right)$ *is a feasible dual solution.*

Lemma 3.7. *Consider a demand j with $\sigma(j) = l$. Let i be the terminal facility corresponding to l. Then, $c_{ij} \leq 5\alpha_j^1$. If $j \in D_l$ then $c_{ij} \leq 3\alpha_j^1$.*

Proof. Let k be the demand with smallest α_k which is tight with l. k is connected to i. So, $c_{ij} \leq c_{lj} + 2\alpha_k^1$. If j is tight with l, then $c_{lj} \leq \alpha_j^1$, otherwise by Lemma 3.2, $c_{lj} \leq 3\alpha_j^1$. Further, $\alpha_k^1 \leq \alpha_j^1$ (this is true if if $j \in D_l$, otherwise we can argue as in Lemma 3.2). So $c_{ij} \leq 3\alpha_j^1$ if $j \in D_l$ and $c_{lj} \leq 5\alpha_j^1$ otherwise. □

Lemma 3.8. *The cost of opening facilities and connecting demands to facilities is at most $3\sum_{j \in D_{L_1}} \alpha_j^1 + 7\sum_{j \notin D_{L_1}} \alpha_j^1$.*

Proof. For an open facility i, define C_i as the set of demands j for which $\beta_{ij}^1 > 0$. Note that the sets C_i are disjoint, and all demands in C_i are assigned to i. We charge the cost of opening a facility at i to the demands in C_i. Each $j \in C_i$ is charged β_{ij}^1. Let $C_{F'} = \cup_{i \in F'} C_i$. So, the cost of opening facilities and connecting demands in $C_{F'}$ to facilities is at most $\sum_{j \in C_{F'}} \left(c_{i(j)j} + \beta_{i(j)j}^1\right) \leq \sum_{j \in C_{F'}} \alpha_j^1$.

If $j \in D_{L_1}$, we know by the previous lemma that $c_{i(j)j} \leq 3\alpha_j^1$. So, assume $j \notin D_{L_1} \cup C_{F'}$. By the previous lemma, we know that there is a terminal location l such that the terminal facility i corresponding to l is at most $5\alpha_j^1$ from j. If i is open, we are done. Otherwise, there is a facility i' and a demand j' such that i' is open and $\beta_{ij'}, \beta_{i'j'} > 0$.

Since i is the terminal facility corresponding to l, there is a demand k such that α_k is smallest among all the demands tight with l and k is connected to i. Let t_k and $t_{j'}$ be the times at which k and j' get connected respectively. Let t_i and $t_{i'}$ be the times at which i and i' become tentatively open. Since both $\beta_{ij'}, \beta_{i'j'} > 0$, we have $c_{ij'}, c_{i'j'} \leq t_{j'}$ and $t_{j'} \leq t_i, t_{i'}$. Since k is connected to i', $t_{i'} \leq t_k \leq \alpha_k^1$. Further, $\alpha_k^1 \leq \alpha_j^1$. So, $c_{ij} \leq c_{li} + c_{ij'} + c_{i'j'} \leq 5\alpha_j^1 + 2t_{i'} \leq 7\alpha_j^1$. □

Lemma 3.9. *The total cost of the Steiner edges added to the set T in Phase 1 is at most $2\sum_{j\in D_{L_1}} \alpha_j^1$.*

Proof. Consider a terminal location $l \in L_1$ with terminal facility i. Let k be the demand in D_l with smallest α_k So, k is connected to i and $c_{li} \leq 2\alpha_k^1$. Note that $|D_l| \geq M$. Further if $j \in D_l$, then $\alpha_j^1 \geq \alpha_k^1$. So, $2\sum_{j\in D_l} \alpha_j^1 \geq Mc_{li}$. □

Theorem 3.2. *The above algorithm produces a solution of total cost at most 9·OPT and is thus a 9-approximation algorithm for ConFL.*

Proof. The cost of opening facilities and connecting clients to facilities in Phase 1 is bounded by $3\sum_{j\in D_{L_1}} \alpha_j^1 + 7\sum_{j\notin D_{L_1}} \alpha_j^1$ (Lemma 3.8). Let T' be the set of edges added to T in Phase 2. The cost of tree T is at most $cost(T')+2\sum_{j\in D_{L_1}} \alpha_j^1$ by Lemma 3.9. Finally $cost(T') \leq 2 \cdot OPT + 2\sum_{j\in D_{L_1}} \alpha_j^1$ since $\left(\alpha', 0, \theta^2\right)$ is a feasible dual solution where $\alpha'_j = \max(\alpha_j^2 - \alpha_j^1, 0)$ (Lemma 3.5). Adding, the total cost is at most $2 \cdot OPT + 7\sum_j \alpha_j^1 \leq 9 \cdot OPT$[1]. □

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 (11) in the dual LP gets replaced by $\sum_j \beta_{ij} \leq f_i + \lambda$.

Let (F^*, C^*, S^*) be the facility, assignment and Steiner tree cost respectively of an optimal ConFL solution. Phase 1 generates a partial primal solution (x, y) and a feasible dual solution $(\alpha^1, \beta^1, \theta^1)$ satisfying $\beta_{vj}^1 = 0$ for all j. Building a Steiner tree on the open facilities costs at most $2S^* + 2(C^* + \sum_{j\in D_{L_1},i} c_{ij}x_{ij})$. Suppose we modify the primal-dual algorithm for ConFL so that after Phase 1,

$$9\sum_i f_i y_i + 3\sum_{j\in D_{L_1},i} c_{ij}x_{ij} + \sum_{j\notin D_{L_1},i} c_{ij}x_{ij} \leq 9\sum_j \alpha_j^1. \tag{13}$$

Now fix λ. We modify the facility opening cost to $f_i + \lambda$ for all $i \neq v$, and run Phase 1 of the algorithm to get primal and dual solutions (x, y) and $(\alpha^1, \beta^1, \theta^1)$. Let z be denote the Steiner tree on the open facilities. Suppose it so happens that $\sum_{i\neq v} y_i = k'$. Then, (x, y, z) and $(\alpha, \beta, \theta, \lambda)$ are feasible solutions to the primal and dual programs respectively for the connected k-median problem. Further from (13) we get that $9\sum_i f_i y_i + \sum_{j,i} c_{ij}x_{ij} + M\sum_e c_e z_e \leq 9\left(\sum_j \alpha_j^1 - k'\lambda\right) + 2(S^* + C^*) \leq 11 \cdot OPT$. The trick then 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 by Jain & Vazirani [7].

[1] In Phase 2, if we use the 1.55-approximation algorithm [16] for the Steiner tree problem we get a slightly better guarantee of 8.55 (4.55 for the rent-or-buy problem).

We can show that there is a value of λ such that depending on how we break ties, we get two ConFL solutions after Phase 1 — one opening $k_1 < k'$ facilities and the other opening $k_2 > k'$ facilities. These two solutions can be found in polynomial time. A *convex combination* of these two solutions yields a fractional solution (x, y, z) that opens k' facilities and satisfies $9\sum_i f_i y_i + 3\sum_{j \in D_{L_1}, i} c_{ij}x_{ij} + \sum_{j \notin D_{L_1}, i} c_{ij}x_{ij} \leq 9 \cdot OPT$. We can round this solution (as in [7]) to get a solution which opens k facilities (including v) losing a factor of 2. Finally we build a Steiner tree on the open facilities. We can show the following.

Theorem 4.1. *There is a 20-approximation algorithm for the Connected k-Median problem.*

To satisfy (13), we do not add any edges to the set T in Phase 1. Instead the Steiner tree in Phase 2 is built with the terminals being the open facilities.

5 Extensions and Refinements

Arbitrary Demands. All our results generalize to the case where instead of unit demands, client j may have a demand $d_j \geq 0$. We can reduce this to the unit demand case by making d_j copies of client j, but this makes the algorithm run in pseudo-polynomial time. But we can easily simulate this reduction by raising α_j at a rate proportional to d_j wherever necessary. All d_j units of demand at j behave *identically*. The analogues of lemmas proved in section 3 are easily shown to be true and consequently we get the same approximation ratios.

Generalization to Edge Capacities. We can extend our algorithm to the following more general problem. We have two types of cables — the first type has a fixed cost of σ per unit length and a *capacity* of u units. The second cable has a fixed cost of M per unit length but unlimited 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, and facilities are connected by a Steiner tree using cables of type 2. This differs from ConFL only in the specification of the first cable type. Assuming integer demands, setting $\sigma = u = 1$ reduces this to ConFL. Ravi & Sinha [15] gave an algorithm for the case when we only have cables of type 1, and want to open facilities and connect clients to open facilities.

We get a constant-factor approximation for this problem by solving a relaxed ConFL instance and a relaxed Steiner tree instance and combining the two solutions. The approximation ratios we get are 7.55, 15.55 and 31.1 for the capacitated versions of the rent-or-buy problem, ConFL, and the connected k-median problem respectively. We get better guarantees if all demands are 1.

The Case $M = 1$. We can get significantly better results for this case. In Phase 1, we run the facility location algorithm of Jain & Vazirani [7]. For each open facility i we identify a client j that is tight with i, and add edges connecting i and j to the set T. In Phase 2 a Steiner tree is built joining the components of T. We show that this is a 4-approximation algorithm. This gives a 8-approximation for the the k-median version. This also gives better guarantees for the capacitated versions of these problems.

Acknowledgments. We thank Jon Kleinberg and David Shmoys for useful discussions, reading through drafts of this paper and pointing out 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.

[2] M. X. Goemans and D. P. Williamson. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24:296–317, 1995.

[3] 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.

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

[5] 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.

[6] R. Hassin, R. Ravi, and F. S. Selman. Approximation algorithms for a capacitated network design problem. In *Proceedings of 3rd APPROX*, pages 167–176, 2000.

[7] 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.

[8] 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.

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

[10] 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.

[11] 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)*, 2002. To appear.

[12] 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.

[13] 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.

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

[15] R. Ravi and A. Sinha. Integrated logistics : Approximation algorithms combining facility location and network design. In *Proceedings of 9th IPCO*, pages 212–229, 2002.

[16] 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.

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