Theorem 3.2 *Algorithm* Consolidate *finds a solution of cost at most* 8 *times the cost of an optimum solution for the metric k-median problem.*

Proof. It is not hard to see that the graph H built in step 6 is a forest and, hence, we can efficiently find the required dominating set I as follows. First, add to I all those vertices i with $\bar{y}_i = 1$ and, then, remove such vertices from H. Any isolated vertex in the resulting graph H is dominated by at least one vertex in I and, thus, these vertices are also deleted from H. After these deletions, H is a forest in which every tree contains at least 2 vertices, and the total number of vertices in H is at most $|N'_>| - |I| = 2(|N'_>| - k)$. A minimum dominating set I' of H can be easily found and it includes at most half of the vertices in each tree of H. Thus, $|I'| \leq |N'_>| - k$. We set $I \leftarrow I \cup I'$ to get a dominating set for the original graph H of size $|I| \leq 2k - |N'_>| + |N'_>| - k = k$ as desired.

Consider this solution I. Each vertex $i \in I$ with $\bar{y}_i = 1$ does not contribute to the cost of the solution since $\bar{x}_{ii} = \bar{y}_i = 1$ and $c(i,i) = 0$. For those vertices i with $\bar{y}_i = \frac{1}{2}$, either $i \in I$ or $s(i) \in I$. Hence, the contribution of i to the cost of the solution is at most $d'_i\, c(s(i), i) \leq 2d'_i\, c(s(i), i)\bar{x}_{s(i)i}$. Therefore, the cost $c(I)$ of solution I is at most 2 times the cost of the half integral solution $\bar{x}, \bar{y}$. By Lemmas 3.4 and 3.6, $c(I)$ is at most 4 times the cost of the fractional solution $\hat{x}, \hat{y}$ with demands d'.

To determine the effect of the re-allocation of demands performed in step 2 on the cost of the solution, consider a location $j \in N$ which has its demand moved to another location $j' \in N'$. For this location $c(j', j) \leq 4\hat{C}_j$. Let j' be served by center i in solution I. If we move back the demand to j and let j be served by center i, this would increase the cost of the solution by at most $d_j\, c(j', j) \leq 4d_j\hat{C}_j$. Therefore, moving all demands back to the original locations increases the cost of the solution by at most 4 times the cost of solution $\hat{x}, \hat{y}$.

By the above arguments, the cost of solution I is at most 8 times the cost of an optimum solution for the k-median problem. $\square$

By using a more complex rounding algorithm than the one described in step 6 of the algorithm, Charikar et al. [12] are able to show that the performance ratio of the algorithm can be improved to $6\frac{2}{3}$.

4 A Primal-Dual Algorithm

In this section we describe another linear programming based approximation algorithm for the k-median problem. The approach that we present now differs from the ones presented in the previous section in that it does not need to solve a linear program, but rather it finds primal and dual solutions for the problem using combinatorial methods. This results in a faster algorithm, and, as we show, the performance ratio is better than the performance ratio of the algorithm of Charikar et al. [12].

The primal-dual algorithm that we present here is due to Jain and Vazirani [22]. This algorithm is based on an interesting relationship between the k-median problem and the uncapacitated facility location problem. In order to understand the algorithm we need first to describe a primal-dual approximation algorithm for the uncapacitated facility location problem.

4.1 The Uncapacitated Facility Location Problem

The uncapacitated facility location problem differs from the k-median problem in that facilities have assigned building costs $f(j)$ and there is no bound on the number of facilities that might be selected as centers. The goal is to minimize the total cost for servicing the clients plus the cost of the facilities selected. An integer program defining the uncapacitated facility location problem is very similar to integer program (1), with the differences stated above. A linear program relaxation of this integer program is given below.

$$\text{Minimize} \quad \sum_{j\in F}\sum_{i\in D} c(i,j)x_{ij} + \sum_{j\in F} f(j)y_j \tag{6}$$

$$\begin{array}{rcll} \text{subject to} & & & \\ \sum_{j\in F} x_{ij} & \geq & 1 & \text{for all} \;\; i \in D \\ x_{ij} & \leq & y_j & \text{for all} \;\; i\in D,\; j \in F \\ 0 & \leq & x_{ij},\; y_j & \text{for all} \;\; i\in D,\; j \in F \end{array}$$

The dual linear program corresponding to this linear program is the following (for a comprehensive study of linear programming concept the reader is referred to [16, 38, 43]).

$$\text{Maximize} \sum_{i\in D} \alpha_i \tag{7}$$

subject to

$$\begin{array}{rcll} \alpha_i - \beta_{ij} & \leq & c(i,j) & \text{for all} \;\; i\in D,\; j \in F \\ \sum_{i\in D}\beta_{ij} & \leq & f(j) & \text{for all} \;\; j \in F \\ \alpha_i,\; \beta_{ij} & \geq & 0 & \text{for all} \;\; i\in D,\; j \in F \end{array} \tag{8}$$

There is a nice interpretation for the dual variables α, β. Let us think, as it happens in the real world, that the clients must pay for the service cost and for the cost of building the selected facilities. If client i is served by facility j, then the amount α_i paid by the client must be at least equal to $c(i,j)$. If $\alpha_i > c(i,j)$, the rest of the money paid by the client, i.e., $\beta_{ij} = \alpha_i - c(i,j)$, goes towards paying for the cost of building facility j. By the complementary slackness conditions, the second constraint of the dual linear program is *tight* (which means that $\sum_{i\in D}\beta_{ij} = f(j)$) if facility j is selected. By the above argument, the total amount $\sum_{i\in D}\beta_{ij}$ contributed by the clients served by j, has to be exactly equal to the building cost of facility j.

To solve the dual linear program, we must determine the price that each client must pay. From the dual solution it is easy to determine which facilities are selected. Each client is assigned to that facility with smallest service cost.

An instance of the uncapacitated facility location problem can be modeled with a graph $G = (D \cup F, E)$ having as vertices the clients and facilities, and edges connecting every facility to each client. The weight of an edge (i,j) is the service cost $c(i,j)$. In the above primal-dual context, an edge (i,j) is said to be *tight* if $\alpha_i \geq c(i,j)$ (or in other words, if $\alpha_i = c(i,j) + \beta_{ij}$ and $\beta_{ij} \geq 0$), and a facility j is said to be *paid for* if $\sum_{i\in D}\beta_{ij} = f(j)$. A client i is *marked* if there is a facility j which has been paid for, and edge (i,j) is tight. The algorithm for approximately solving the uncapacitated facility location problem is as follows.

Algorithm PrimalDual

0. Initially all clients are un-marked. All values α_i and β_{ij} are initialized to 0.
1. Repeat steps 2 and 3 as long as there are un-marked clients.
2. Simultaneously and uniformly (at the same rate) raise the values of, both, the dual variables α_i for all un-marked clients i and the variables β_{ij} for all tight edges (i,j), until either:

- $\alpha_i = c(i,j)$ for some edge (i,j), or
- $\sum_{i \in D} \beta_{ij} = f(j)$ for some facility j.

  In the first case we label edge (i,j) as tight. In the second case we label facility j as paid for.

3. Every un-marked client i with a tight edge (i,j) connecting it to a paid for facility j is marked.
4. Build the graph $T = (D \cup F, E')$ containing those edges (i,j) for which $\beta_{ij} > 0$.
5. Build the square T^2 of graph T by adding an edge between vertices u and v if there is a path of length at most 2 between u and v in T.
6. Build the subgraph H of T^2 induced by those facilities that are paid for.
7. Find a maximal independent set I of H.
8. Select I as the set of centers and let each client i be served by a center $j \in I$ for which the service cost $c(i,j)$ is minimum. If for a client i there are 2 or more centers with minimum service cost, choose any one of them for servicing i.

In the solution I produced by this algorithm, we say that a client i is *directly connected* to the facility j that serves it if $\beta_{ij} > 0$. Otherwise client i is said to be *indirectly connected.*

4.2 Performance Ratio

We interpret the value of variable α_i as the price that client i has to pay for, both, building a facility and for being serviced by that facility. Let client i be serviced by facility j. Let α_i^f be the price that client i pays for building facility j and let α_i^s be the service cost paid by the client. Clearly, $\alpha_i = \alpha_i^s + \alpha_i^f$. If client i is directly connected to j, then $\alpha_i = c(i,j) + \beta_{ij}$, and so we set $\alpha_i^s \leftarrow c(i,j)$, and $\alpha_i^f \leftarrow \beta_{ij}$. If i is indirectly connected, then we set $\alpha_i^s \leftarrow \alpha_i$ and $\alpha_i^f \leftarrow 0$.

Lemma 4.1

$$\sum_{j \in I} f(j) = \sum_{i \in D} \alpha_i^f.$$

Proof. Every facility $j \in I$ is paid for, i.e. $f(j) = \sum_{i \in D} \beta_{ij} = \sum_{i \in D_j} \alpha_i^f$, where D_j is the set of clients directly connected to j. Since sets D_j are disjoint, the claim follows. □

Lemma 4.2 *For every indirectly connected client i, $c(i,j) \le 3\alpha_i^s$, where $j \in I$ is the facility that serves i.*

Proof. Since the loop in steps 1-3 terminates when all clients are marked, then, i is marked in step 3 because these is a facility j' for which $\alpha_i \ge c(i,j')$. However, $j' \notin I$ since i is indirectly connected. As i is indirectly connected, facility j' was not selected to be in the final solution I computed in step 7. Note that since I is a maximal independent set of H, there has to be at least one facility $k \in I$ such that there is an edge from j' to k in the graph H built in step 6 (see Figure 1). Because clients are assigned to their nearest centers in step 8, then $c(i,k) \ge c(i,j)$.

Observe that k and j' are facilities and so there is no edge (k,j') in the graph $G = (D \cup F, E)$. Since edge (k,j') belongs to H, there must be a client h with edges (h,k) and (h,j') such that $\beta_{hk} > 0$ and $\beta_{hj'} > 0$. This implies that $\alpha_h > c(h,k)$ and $\alpha_h > c(h,j')$.

The algorithm does not rise the value of α_h after facility j' is paid for (and in fact it might stop raising the value of α_h when k or other neighbouring facility of h is paid for). Since, as we assumed above, i is marked when α_i takes value $c(i,i')$, then α_i is raised at least until the time

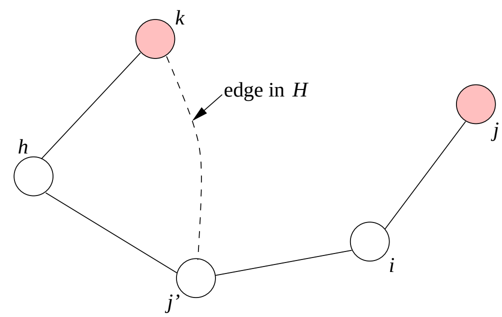

Figure 1: Service cost of client i. Centers k and j belong to I, but $j' \notin I$.

when j' is paid for. Therefore, $\alpha_i > \alpha_h$. By the triangle inequality (see Figure 1), $c(i,j) \le c(i,k) \le c(i,j') + c(h,j') + c(h,k) \le \alpha_i + 2\alpha_h \le 3\alpha_i$. □

Using these two lemmas, we can compute the performance ratio of algorithm PrimalDual.

Theorem 4.1 *The performance ratio of algorithm* PrimalDual *is* 3*.*

Proof. Let x, y and α, β be the primal and dual solutions produced by the algorithm. Since for a client i directly connected to a facility j, $\alpha_i^s = c(i,j)$, and by Lemma 4.2, for each indirectly connected client i, $c(i,j) \le 3\alpha_i^s$, then

$$\sum_{j \in F} \sum_{i \in D} c(i,j) x_{ij} \le 3 \sum_{i \in D} \alpha_i^s.$$

From this inequality and Lemma 4.1, we get

$$\sum_{j \in F} \sum_{i \in D} c(i,j) x_{ij} + 3 \sum_{j \in F} f(j) y_j \le 3 \sum_{i \in D} (\alpha_i^s + \alpha_i^f) = 3 \sum_{i \in D} \alpha_i. \tag{9}$$

Since the value of an optimum solution for the k-median problem is at least $\sum_{i \in D} \alpha_i$, the claim follows. □

4.3 Running Time

Steps 1-3 are, perhaps, the most difficult steps of the algorithm to implement and analyze, so we will give some details here as to how these steps can be efficiently implemented. Note that since the values α_i are raised uniformly, the order in which the edges will become tight is consistent with a non-decreasing ordering of the edges by cost. Hence, if we store the edges in a list L in non-decreasing order of cost, we will know the order in which the events $\alpha_i = c(i,j)$ of step 2 will take place. To keep track of the second class of events that take place in step 2, namely $\sum_{i \in D} \beta_{ij} = f(j)$ for some facility j, we need to maintain for each facility j the following 3 variables:

- a variable b_j giving the number of tight edges currently contributing to facility j,
- a variable t_j giving the time when the value of b_j last changed, and
- a variable p_j giving the total contribution made by clients to facility j up to time t_j.

To determine the event that will take place during the following iteration of steps 2-3 we first compute

$$\tau = \min\left\{c(i,j), \min_{j\in F}\left\{\frac{f(j)-p_j}{b_j}\right\}\right\},$$

where (i,j) is the first edge in L, if any. If L is empty, we simply ignore the first term $c(i,j)$. We can efficiently compute τ by using a heap h.

a. If $c(i,j)=\tau$, then edge (i,j) becomes tight in this iteration, so we discard (i,j) from L.

- If j is not yet paid for, we update $p_j \leftarrow p_j + (\tau - t_j)b_j$ and, then, increase the value of b_j by 1 since in the next iterations the value of β_{ij} will be increased. Finally, we set $t_j \leftarrow \tau$. These steps can be performed in constant time and updating the value $\frac{f(j)-p_j}{b_j}$ associated with j in the heap needs $O(\log n_f)$ time, where n_f is the number of facilities.
- If j is already paid for, then we mark i. Since from this point on the value α_i and the values $\beta_{ij'}$ for all tight edges (i,j') will not increase any more, then, we need to update the values of the variables for such facilities j'. For each facility j' such that (i,j') is tight we set $p_{j'} \leftarrow p_{j'} + b_{j'}(\tau - t_{j'})$ and, then, we decrease $b_{j'}$ by 1. Finally, we set $t_{j'} \leftarrow \tau$. The total amount of time needed to perform these steps is $O(\text{degree}(i)\log n_f)$, where degree$(i)$ is the degree of client i in the graph $G=(D\cup F, E)$.

b. On the other hand, if $\frac{f(j)-p_j}{b_j} = \tau$, then facility j will next get paid for, so we remove this value $\frac{f(j)-p_j}{b_j}$ from the heap. Furthermore, all clients i with tight edges to j will be marked, and so their α_i and β_{ij} values will stop increasing. For each one of these clients with tight edges (i,j') we need to update the values of $p_{j'}$, $t_{j'}$, and $b_{j'}$ as described above. Let S_j be the set of clients marked in this step. The total time needed to perform the above steps is $O(\sum_{i\in S_j}\text{degree}(i)\log n_f)$.

Since every client is marked only once, then the total time needed to complete steps 1-3 is $O(\sum_{i\in D}\text{degree}(i)\log n_f) = O(m\log n_f)$, where m is the number of edges in the graph $G = (D\cup F, E)$.

4.4 Algorithm for the k-Median Problem

By studying the integer program formulations for the k-median and uncapacitated facility location problems, we note that the Lagrangian relaxation of constraint (2) in the integer program (1) for the k-median problem gives an instance of the uncapacitated facility location problem in which all the facilities have the same cost z, where z is the Lagrange multiplier:

$$\begin{aligned}
&\text{Minimize} && \sum_{j\in F}\sum_{i\in C} c(i,j)x_{ij} + \sum_{j\in F} z y_j && (10)\\
&\text{subject to} && && \\
&\sum_{j\in F} x_{ij} \ge 1 && \text{for all } i\in D && \\
&x_{ij} \le y_j && \text{for all } i\in D,\ j\in F && \\
&x_{ij},\ y_j \in \{0,1\} && \text{for all } i\in D,\ j\in F &&
\end{aligned}$$

The value of the Lagrange multiplier z controls the number of facilities selected by the algorithm PrimalDual. As the value of z increases, the number of facilities selected decreases. If it happens