SIAM J. COMPUT. © 2004 Society for Industrial and Applied Mathematics
Vol. 33, No. 3, pp. 544–562

LOCAL SEARCH HEURISTICS FOR k-MEDIAN AND FACILITY LOCATION PROBLEMS[*]

VIJAY ARYA[†], NAVEEN GARG[†], ROHIT KHANDEKAR[†], ADAM MEYERSON[‡], KAMESH MUNAGALA[‡], AND VINAYAKA PANDIT[§]

Abstract. We analyze local search heuristics for the metric k-median and facility location problems. We define the *locality gap* of a local search procedure for a minimization problem as the maximum ratio of a locally optimum solution (obtained using this procedure) to the global optimum. For k-median, we show that local search with swaps has a locality gap of 5. Furthermore, if we permit up to p facilities to be swapped simultaneously, then the locality gap is $3 + 2/p$. This is the first analysis of a local search for k-median that provides a bounded performance guarantee with only k medians. This also improves the previous known 4 approximation for this problem. For uncapacitated facility location, we show that local search, which permits adding, dropping, and swapping a facility, has a locality gap of 3. This improves the bound of 5 given by M. Korupolu, C. Plaxton, and R. Rajaraman [*Analysis of a Local Search Heuristic for Facility Location Problems*, Technical Report 98-30, DIMACS, 1998]. We also consider a capacitated facility location problem where each facility has a capacity and we are allowed to open multiple copies of a facility. For this problem we introduce a new local search operation which opens one or more copies of a facility and drops zero or more facilities. We prove that this local search has a locality gap between 3 and 4.

Key words. local search, approximation algorithm, facility location

AMS subject classifications. 68W25, 90B80, 90C59

DOI. 10.1137/S0097539702416402

1. Introduction.

The problem of locating facilities in a manner so that they can effectively serve a set of clients has been the subject of much research. While one could consider fairly general measures of effectiveness of a set of locations in serving the clients, one measure that is typically used is the distance between the client and the facility that is serving it. Since by opening a lot of facilities, we can be near every client, it also makes sense to take into account the number of facilities opened in judging the quality of a solution. These two measures, typically referred to as the service cost and the facility cost, can be combined in many ways to obtain interesting variants to the general facility location problem. For instance, in *k-median* we require that at most k facilities be opened and the total service cost, measured as the sum of the distance of each client to the nearest open facility, be minimum. Instead of setting a limit on the total number of facilities that could be opened, we sometimes associate with every facility a cost of opening that facility. The facility cost of a solution is then the sum of the costs of the facilities that are opened, and the quality of the solution is measured by the sum of the facility and service costs. This, in fact, is the

[*]Received by the editors October 21, 2002; accepted for publication (in revised form) October 30, 2003; published electronically March 23, 2004. A preliminary version of the paper appeared in *Proceedings of the 33rd ACM Symposium on Theory of Computing*, Crete, Greece, 2001.

http://www.siam.org/journals/sicomp/33-3/41640.html

[†]Department of Computer Science and Engineering, Indian Institute of Technology, New Delhi 110016, India (vijayarya@cse.iitd.ernet.in, naveen@cse.iitd.ernet.in, rohitk@cse.iitd.ernet.in). The first author was partially supported by an IBM Fellowship. The third author was partially supported by an Infosys Fellowship.

[‡]Department of Computer Science, Stanford University, Stanford, CA 94305 (awm@cs.ucla.edu, kamesh@cs.stanford.edu). The fourth author was supported by ARO DAAG-55-97-1-0221. The fifth author was supported by ONRN00014-98-1-0589.

[§]IBM India Research Lab, Block I, Indian Institute of Technology, New Delhi 110016, India (pvinayak@in.ibm.com).

classical *facility location problem*. Note that in this setting the facility costs need not be the same and would, in general, depend on the location at which the facility is being opened. A generalization of the classical facility location problem arises when we associate a capacity with each facility, which measures the maximum number of clients that the facility can serve. Further variants of this *capacitated facility location* (CFL) problem arise when we bound the number of facilities that can be opened at a certain location. Thus in k-CFL, we can open at most k facilities at any location.

Local search techniques have been very popular as heuristics for hard combinatorial optimization problems. The 1-exchange heuristic by Lin and Kernighan [15] for the metric-TSP remains the method of choice for practitioners. However, most of these heuristics have poor worst-case guarantees, and very few approximation algorithms that rely on local search are known. Könemann and Ravi [13] used local search algorithms for degree-bounded minimum spanning trees. Chandra, Karloff, and Tovey [3] show an approximation factor of $4\sqrt{n}$ for the 2-exchange local search heuristic for the Euclidean traveling salesman problem. Khuller, Bhatia, and Pless [12] give a local search approximation algorithm for finding a feedback edge-set incident upon the minimum number of vertices. Local search has also been used for set packing problems by Arkin and Hassin [2]. Here, we provide worst-case analysis of local search algorithms for facility location problems.

For an instance I of a minimization problem, let $\texttt{global}(I)$ denote the cost of the global optimum and $\texttt{local}(I)$ be the cost of a locally optimum solution provided by a certain local search heuristic. We call the supremum of the ratio $\texttt{local}(I)/\texttt{global}(I)$ the *locality gap* of this local search procedure. For 1-CFL with uniform capacities, Korupolu, Plaxton, and Rajaraman [14] argued that any procedure that permits adding, dropping, or swapping a facility has a locality gap of at most 8. Their analysis was subsequently refined and tightened by Chudak and Williamson [8] to yield a locality gap of at most 6. Pál, Tardos, and Wexler [19] present a local search algorithm for 1-CFL with nonuniform capacities which has a locality gap of 9. Mahdian and Pál [16] considered the universal facility location problem where the cost of opening a facility is any arbitrary but monotone function of the demand that the facility serves; note that this problem generalizes k-CFL. Mahdian and Pál extended the algorithm of [19] to obtain a local search algorithm with a locality gap of 8. For uncapacitated facility location (UFL), Korupolu, Plaxton, and Rajaraman [14] provided a bound of 5 on the locality gap when the only operations permitted are those of adding, dropping, or swapping a facility. Charikar and Guha [4] introduced an operation which permits adding a facility and dropping many and showed that this local search procedure has a locality gap of exactly 3. For k-median, Korupolu, Plaxton, and Rajaraman [14] gave a local search procedure which permitted adding, deleting, and swapping facilities and gave a solution with $k(1+\epsilon)$ facilities having a service cost at most $3+5/\epsilon$ times the optimum k-median solution.

A different approach to facility location was employed by Shmoys, Tardos, and Aardal [20] and Charikar et al. [5]. They formulated the problems as linear programs and rounded the optimum fractional solution to obtain a 3 approximation for the UFL problem and a $6\frac{2}{3}$ approximation for k-median. Jain and Vazirani [11] gave an alternate 3 approximation algorithm for UFL using the primal-dual schema. They also observed that UFL can be viewed as a Lagrange-relaxation of k-median and utilized this to give a 6 approximation algorithm for k-median. Later, Charikar and Guha [4] improved this to a 4 approximation. Recently, Archer, Rajagopalan, and Shmoys [1] showed that the algorithm due to Jain and Vazirani can be made to satisfy the "continuity" property and established an integrality gap of at most 3 for the most natural LP relaxation for the k-median problem. However, their proof gives only an exponential time algorithm. Guha and Khuller [9] employed randomization to improve the approximation guarantee of UFL to 2.408. This was further improved to $(1+2/e)$ by Chudak [6] and finally to 1.728 by Charikar and Guha [4]. Similar ideas were used by Chudak and Shmoys [7] to obtain a 3 approximation algorithm for ∞-CFL when the capacities are uniform. Jain et al. [10] used the method of dual fitting and factor revealing LP to design two greedy algorithms for the UFL problem with approximation guarantees of 1.861, 1.61 and running times of $O(m \log m)$, $O(n^3)$, respectively, where n is the number of vertices and m is the number of edges in the underlying graph. Mahdian, Ye, and Zhang [17] combined the ideas in [10] with the idea of cost scaling to obtain an approximation factor of 1.52 for UFL, which is also the best known. Jain and Vazirani [11] obtained a 4 approximation algorithm for ∞-CFL when the capacities were nonuniform by solving a related UFL problem using their primal-dual algorithm. Recently, Mahdian, Ye, and Zhang [18] gave a 2 approximation for the ∞-CFL with nonuniform capacities by reducing it to a linear-cost facility location problem.

Our results. In this paper, we analyze local search heuristics for three problems.

1. For metric k-median, we show that the local search with single swaps has a locality gap of 5. This is the first analysis of a local search for k-median that provides a bounded performance guarantee with only k medians. We also show that doing multiple swaps, that is, dropping at most p facilities and opening the same number of new facilities, yields a locality gap of $3 + 2/p$. This improves on the 4 approximation algorithm for k-median by Charikar and Guha [4]. Our analysis of the locality gap is tight; that is, for an infinite family of instances there is a locally optimum solution whose service cost is nearly $(3 + 2/p)$ times that of the global optimum.
2. For metric UFL, we show that local search, which permits adding, dropping, and swapping a facility, has a locality gap of 3. This improves the bound of 5 given by Korupolu, Plaxton, and Rajaraman [14]. Again, our analysis of the algorithm is tight. We show how our algorithm can be improved to achieve a $1 + \sqrt{2} + \epsilon \approx 2.414 + \epsilon$ approximation using ideas from [4].
3. For metric ∞-CFL, we consider the setting when the capacities may be nonuniform and argue that local search, where the only operation permitted is to add multiple copies of a facility and drop zero or more facilities, has a locality gap of at most 4. We give a polynomial time algorithm that uses `Knapsack` as a subroutine to search for a lower cost solution in the neighborhood. We also show that there is a locally optimum solution with cost 3 times the optimum. We show how our algorithm can be improved to achieve a $1 + \sqrt{3} + \epsilon \approx 3.732 + \epsilon$ approximation using ideas from [4].

The paper is organized as follows. Section 2 introduces some notation and defines the problems addressed in this paper formally. In section 3, we first prove a locality gap of 5 for the k-median problem when only single swaps are permitted and then extend the analysis to argue a locality gap of $3 + 2/p$ when up to p facilities can be swapped simultaneously. Sections 4 and 5 discuss the algorithms for UFL and ∞-CFL, respectively. Section 6 concludes with some open problems.

2. Notation and preliminaries.

In the k-median and facility location problems, we are given two sets: F, the set of *facilities*, and C, the set of *clients*. Let $c_{ij} \geq 0$ denote the cost of serving client $i \in C$ by a facility $j \in F$; we will think of this as the distance between client i and facility j. The goal in these problems is to identify a subset of facilities $S \subseteq F$ and to serve all clients by facilities in S such that some objective function is minimized. The facilities in S are said to be *open*. The metric versions of these problems assume that distances c_{ij} are symmetric and satisfy the triangle inequality. The problems considered in this paper are defined as follows.

1. The metric k-median problem. Given integer k, identify a set $S \subseteq F$ of at most k facilities to open such that the total cost of serving all clients by open facilities is minimized.
2. The metric UFL problem. For each facility $i \in F$, we are given a cost $f_i \geq 0$ of opening the facility i. The goal is to identify a set of facilities $S \subseteq F$ such that the total cost of opening the facilities in S and serving all the clients by open facilities is minimized.
3. The metric ∞-CFL problem. For each facility $i \in F$, we are given a cost $f_i \geq 0$ of opening a copy of facility i and an integer capacity $u_i > 0$, which is the maximum number of clients that a single copy of the facility i can serve. The output is a set of facilities $S \subseteq F$ and the number of copies of each facility in S to be opened. The goal is to serve each client by a copy of a facility in S such that the number of clients served by a copy of a facility is at most its capacity. The objective is to minimize the total facility cost and the cost of serving all the clients.

Thus for all the problems we consider, a solution can be specified by giving the set of open facilities together with their multiplicities. In the rest of this paper we will think of a solution as a multiset of facilities.

```
    Algorithm Local Search.

1.  $S \leftarrow$ an arbitrary feasible solution in $\mathcal{S}$.
2.  While $\exists S' \in \mathcal{B}(S)$ such that $cost(S') < cost(S)$,
        do $S \leftarrow S'$.
3.  return $S$.
```

FIG. 1. *A generic local search algorithm.*

A generic local search algorithm (Figure 1) can be described by a set $\mathcal{S}$ of all feasible solutions, a cost function $cost : \mathcal{S} \rightarrow \mathbb{R}$, a *neighborhood* structure $\mathcal{B} : \mathcal{S} \rightarrow 2^{\mathcal{S}}$, and an oracle that, given any solution S, finds (if possible) a solution $S' \in \mathcal{B}(S)$ such that $cost(S') < cost(S)$. A solution $S \in \mathcal{S}$ is called *locally optimum* if $cost(S) \leq cost(S')$ for all $S' \in \mathcal{B}(S)$; the algorithm in Figure 1 always returns one such solution. The cost function and the neighborhood structure $\mathcal{B}$ will be different for different problems and algorithms.

If S is a locally optimum solution, then for all $S' \in \mathcal{B}(S)$,

$$cost(S') - cost(S) \geq 0.$$

Our proof of the locality gap proceeds by considering a suitable, polynomially large (in the input size) subset $\mathcal{Q} \subseteq \mathcal{B}(S)$ of neighboring solutions and arguing that

$$\sum_{S' \in \mathcal{Q}} (cost(S') - cost(S)) \leq \alpha \cdot cost(O) - cost(S),$$

where O is an optimum solution and $\alpha > 1$ a suitable constant. This implies that $cost(S) \leq \alpha \cdot cost(O)$, which gives a bound of α on the locality gap.

To translate the proof of the locality gap into an approximation algorithm with polynomial running time, we modify step 2 of the algorithm as follows.

```
2.  While  $\exists S' \in \mathcal{B}(S)$ such that $cost(S') \le (1-\epsilon/Q)\, cost(S)$,
        do  $S \leftarrow S'$.
```

Here $\epsilon > 0$ is a constant and $Q = |\mathcal{Q}|$. Thus, in each local step, the cost of the current solution decreases by a factor of at least ϵ/Q. If O denotes an optimum solution and S_0 denotes the initial solution, then the number of steps in the algorithm is at most $\log(cost(S_0)/cost(O))/\log\frac{1}{1-\epsilon/Q}$. As $Q, \log(cost(S_0))$, and $\log(cost(O))$ are polynomial in the input size, the algorithm terminates after polynomially many local search steps.

To analyze the quality of this locally optimum solution S, we note that for all $S' \in \mathcal{Q}$, $cost(S') > (1-\epsilon/Q)cost(S)$. Hence

$$\alpha \cdot cost(O) - cost(S) \ge \sum_{S' \in \mathcal{Q}} (cost(S') - cost(S)) > -\epsilon \cdot cost(S),$$

which implies that $cost(S) \le \frac{\alpha}{(1-\epsilon)} cost(O)$. Thus our proof that a certain local search procedure has a locality gap of at most α translates into an $\alpha/(1-\epsilon)$ approximation algorithm with a running time that is polynomial in the input size and $1/\epsilon$.

We use the following notation. Given a solution A, let A_j denote the *service cost* of client j, which is the distance between j and the facility in A which serves it. For every facility $a \in A$, we use $N_A(a)$ to denote the set of clients that a serves (Figure 2). For a subset of facilities, $T \subseteq A$, let $N_A(T) = \bigcup_{a \in T} N_A(a)$.

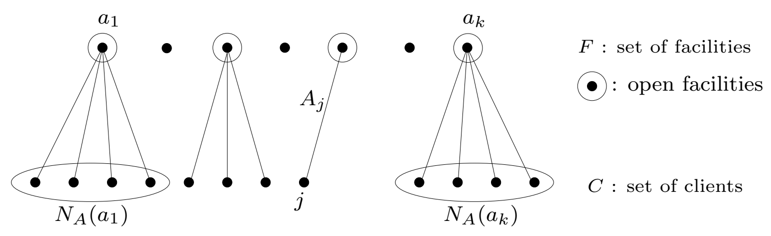

FIG. 2. *Illustration of neighborhood and service costs.*

3. The k-median problem.

The k-median problem is to open a subset $S \subseteq F$ of at most k facilities so that the total service cost is minimized. Thus, if a client j is served by a facility $\sigma(j) \in S$, then we wish to minimize $cost(S) = \sum_{j \in C} c_{j\sigma(j)}$. For a fixed S, serving each client by the nearest facility in S minimizes this cost.

3.1. Local search with single swaps.

In this section, we consider a local search using single swaps. A swap is effected by closing a facility $s \in S$ and opening a facility $s' \notin S$ and is denoted by $\langle s, s' \rangle$; hence $\mathcal{B}(S) = \{S - \{s\} + \{s'\} \mid s \in S\}$. We start with an arbitrary set of k facilities and keep improving our solution with such swaps until we reach a locally optimum solution. The algorithm is described in Figure 1. We use $S - s + s'$ to denote $S - \{s\} + \{s'\}$.

3.2. The analysis.

We now show that this local search procedure has a locality gap of 5. Let S be the solution returned by the local search procedure and let O be an optimum solution. From the local optimality of S, we know that

$$cost(S - s + o) \ge cost(S) \quad \text{for all } s \in S, o \in O. \tag{1}$$

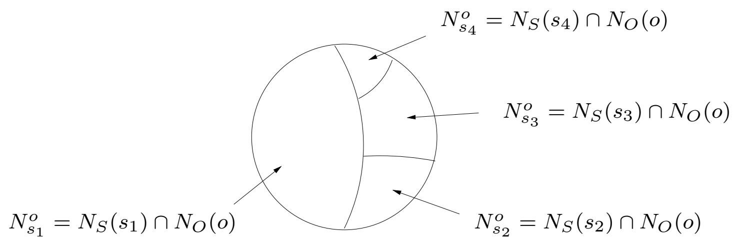
$N^o_{s_4} = N_S(s_4) \cap N_O(o)$
$N^o_{s_3} = N_S(s_3) \cap N_O(o)$
$N^o_{s_1} = N_S(s_1) \cap N_O(o)$
$N^o_{s_2} = N_S(s_2) \cap N_O(o)$

FIG. 3. *Partitioning $N_O(o)$.*

Note that even if $S \cap O \neq \emptyset$, the above inequalities hold. We combine these inequalities to show that $cost(S) \leq 5 \cdot cost(O)$.

Consider a facility $o \in O$. We partition $N_O(o)$ into subsets $N^o_s = N_O(o) \cap N_S(s)$ as shown in Figure 3.

DEFINITION 3.1. *We say that a facility $s \in S$* captures *a facility $o \in O$ if s serves more than half the clients served by o, that is, $|N^o_s| > \frac{1}{2}|N_O(o)|$.*

It is easy to see that a facility $o \in O$ is captured by at most one facility in S. We call a facility $s \in S$ *bad*, if it captures some facility $o \in O$, and *good* otherwise. Fix a facility $o \in O$ and consider a 1-1 and onto function $\pi : N_O(o) \to N_O(o)$ satisfying the following property (Figure 4).

PROPERTY 3.1. *If s does not capture o, that is, $|N^o_s| \leq \frac{1}{2}|N_O(o)|$, then $\pi(N^o_s) \cap N^o_s = \emptyset$.*

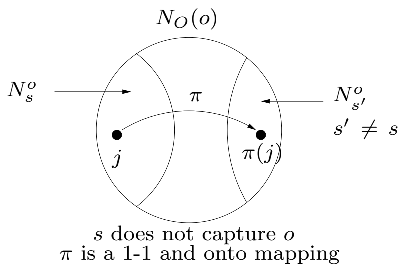

FIG. 4. *The mapping π on $N_O(o)$.*

We outline how to obtain one such mapping π. Let $D = |N_O(o)|$. Order the clients in $N_O(o)$ as $c_0, \ldots, c_{D-1}$ such that for every $s \in S$ with a nonempty N^o_s, the clients in N^o_s are consecutive; that is, there exists p, q, $0 \leq p \leq q \leq D-1$, such that $N^o_s = \{c_p, \ldots, c_q\}$. Now, define $\pi(c_i) = c_j$, where $j = (i + \lfloor D/2 \rfloor)$ modulo D. For contradiction assume that both $c_i, \pi(c_i) = c_j \in N^o_s$ for some s, where $|N^o_s| \leq D/2$. If $j = i + \lfloor D/2 \rfloor$, then $|N^o_s| \geq j - i + 1 = \lfloor D/2 \rfloor + 1 > D/2$. If $j = i + \lfloor D/2 \rfloor - D$, then $|N^o_s| \geq i - j + 1 = D - \lfloor D/2 \rfloor + 1 > D/2$. In both cases we have a contradiction, and hence function π satisfies property 3.1.

The notion of *capture* can be used to construct a bipartite graph $H = (S, O, E)$ (Figure 5). For each facility in S we have a vertex on the S-side, and for each facility in O we have a vertex on the O-side. We add an edge between $s \in S$ and $o \in O$ if s captures o. It is easy to see that each vertex on the O-side has degree at most one, while vertices on the S-side can have degree up to k. We call H the *capture graph*.

We now consider k swaps, one for each facility in O. If some bad facility $s \in S$

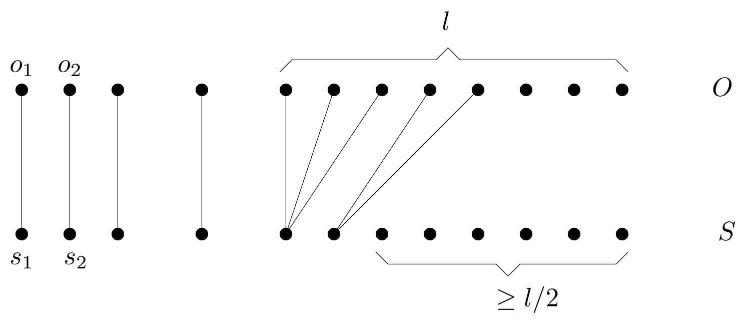

FIG. 5. *Capture graph $H = (S, O, E)$.*

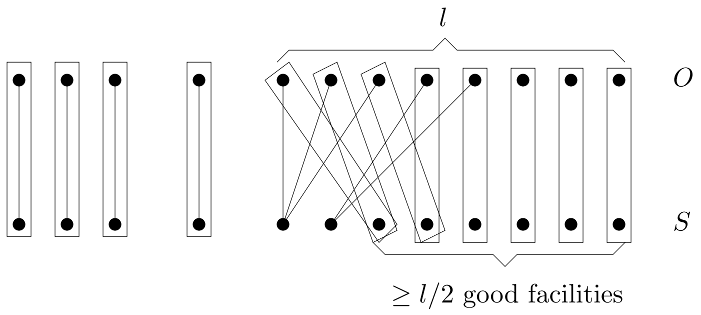

FIG. 6. *k swaps considered in the analysis.*

captures exactly one facility $o \in O$, then we consider the swap $\langle s, o\rangle$. Suppose l facilities in S (and hence l facilities in O) are not considered in such swaps. Each facility out of these l facilities in S is either good or captures at least two facilities in O. Hence there are at least $l/2$ good facilities in S. Now, consider l swaps in which the remaining l facilities in O get swapped with the good facilities in S such that each good facility is considered in at most two swaps (Figure 6). The bad facilities which capture at least two facilities in O are not considered in any swaps. The swaps considered above satisfy the following properties.

1. Each $o \in O$ is considered in exactly one swap.
2. A facility $s \in S$ which captures more than one facility in O is not considered in any swap.
3. Each good facility $s \in S$ is considered in at most two swaps.
4. If swap $\langle s, o\rangle$ is considered, then facility s does not capture any facility $o' \neq o$.

We now analyze these swaps one by one. Consider a swap $\langle s, o\rangle$. We place an upper bound on the increase in the cost due to this swap by reassigning the clients in $N_S(s) \cup N_O(o)$ to the facilities in $S - s + o$ as follows (Figure 7). The clients $j \in N_O(o)$ are now assigned to o. Consider a client $j' \in N_s^{o'}$ for $o' \neq o$. As s does not capture o', by Property 3.1 of π we have that $\pi(j') \notin N_S(s)$. Let $\pi(j') \in N_S(s')$. Note that the distance that the client j' travels to the nearest facility in $S - s + o$ is at most $c_{j's'}$. From triangle inequality, $c_{j's'} \leq c_{j'o'} + c_{\pi(j')o'} + c_{\pi(j')s'} = O_{j'} + O_{\pi(j')} + S_{\pi(j')}$. The clients which do not belong to $N_S(s) \cup N_O(o)$ continue to be served by the same facility. From inequality (1) we have

$$cost(S - s + o) - cost(S) \geq 0.$$

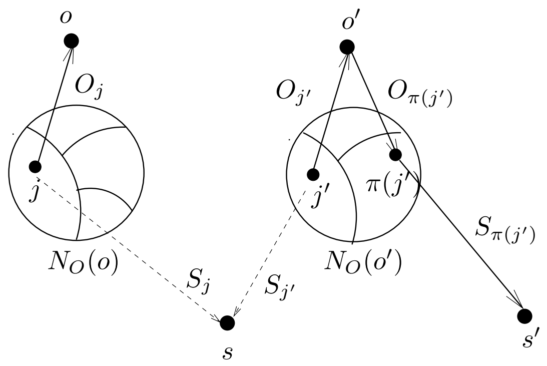

FIG. 7. *Reassigning the clients in* $N_S(s) \cup N_O(o)$.

Therefore,

$$\sum_{j \in N_O(o)} (O_j - S_j) + \sum_{\substack{j \in N_S(s),\\ j \notin N_O(o)}} (O_j + O_{\pi(j)} + S_{\pi(j)} - S_j) \geq 0. \tag{2}$$

As each facility $o \in O$ is considered in exactly one swap, the first term of inequality (2) added over all k swaps gives exactly $cost(O) - cost(S)$. For the second term, we will use the fact that each $s \in S$ is considered in at most two swaps. Since S_j is the shortest distance from client j to a facility in S, using triangle inequality we get $O_j + O_{\pi(j)} + S_{\pi(j)} \geq S_j$. Thus the second term of inequality (2) added over all k swaps is no greater than $2\sum_{j \in C}(O_j + O_{\pi(j)} + S_{\pi(j)} - S_j)$. But since π is a 1-1 and onto mapping, $\sum_{j \in C} O_j = \sum_{j \in C} O_{\pi(j)} = cost(O)$ and $\sum_{j \in C}(S_{\pi(j)} - S_j) = 0$. Thus, $2\sum_{j \in C}(O_j + O_{\pi(j)} + S_{\pi(j)} - S_j) = 4 \cdot cost(O)$. Combining the two terms, we get $cost(O) - cost(S) + 4 \cdot cost(O) \geq 0$. Thus we have the following theorem.

THEOREM 3.2. *A local search procedure for the metric k-median problem with the local neighborhood structure defined by $\mathcal{B}(S) = \{S - \{s\} + \{s'\} \mid s \in S\}$ has a locality gap of at most* 5.

The above algorithm and analysis extend very simply to the case when the clients $j \in C$ have arbitrary demands $d_j \geq 0$ to be served.

3.3. Local search with multiswaps. In this section, we generalize the algorithm in section 3 to consider multiswaps in which up to $p > 1$ facilities could be swapped simultaneously. The neighborhood structure is now defined by

$$\mathcal{B}(S) = \{(S \setminus A) \cup B \mid A \subseteq S,\ B \subseteq F,\ \text{and } |A| = |B| \leq p\}. \tag{3}$$

The neighborhood captures the set of solutions obtainable by deleting a set of at most p facilities A and adding a set of facilities B where $|B| = |A|$; this swap will be denoted by $\langle A, B\rangle$. We prove that the locality gap of the k-median problem with respect to this operation is exactly $(3 + 2/p)$.

3.4. Analysis. We extend the notion of capture as follows. For a subset $A \subseteq S$, we define

$$capture(A) = \{o \in O \mid |N_S(A) \cap N_O(o)| > |N_O(o)|/2\}.$$

It is easy to observe that if $X, Y \subseteq S$ are disjoint, then $capture(X)$ and $capture(Y)$ are disjoint and if $X \subset Y$, then $capture(X) \subseteq capture(Y)$. We now partition S into sets $A_1, \ldots, A_r$ and O into sets $B_1, \ldots, B_r$ such that

```
procedure Partition;
        $i = 0$
        while $\exists$ a bad facility in $S$ do
            1. $i = i + 1$                                        {iteration $i$}
            2. $A_i \leftarrow \{b\}$ where $b \in S$ is any bad facility
            3. $B_i \leftarrow capture(A_i)$
            4. while $|A_i| \neq |B_i|$ do
                4.1. $A_i \leftarrow A_i \cup \{g\}$ where $g \in S \setminus A_i$ is any good facility
                4.2. $B_i \leftarrow capture(A_i)$
            5. $S \leftarrow S \setminus A_i$
               $O \leftarrow O \setminus B_i$
        $A_r \leftarrow S$
        $B_r \leftarrow O$
end.
```

FIG. 8. *A procedure to define the partitions.*

1. for $1 \leq i \leq r-1$, we have $|A_i| = |B_i|$ and $B_i = capture(A_i)$; since $|S| = |O|$, it follows that $|A_r| = |B_r|$;
2. for $1 \leq i \leq r-1$, the set A_i has exactly one bad facility;
3. the set A_r contains only good facilities.

A procedure to obtain such a partition is given in Figure 8.

CLAIM 3.1. *The procedure defined in Figure* 8 *terminates with partitions of S and O, satisfying the properties listed above.*

Proof. The condition in the while loop in step 4 and the assignment in step 5 of the procedure maintain the invariant that $|S| = |O|$. Steps 3 and 4.2 of the procedure ensure that for $1 \leq i \leq r-1$, we have $B_i = capture(A_i)$, and steps 2 and 4.1 ensure that each for $1 \leq i \leq r-1$, the set A_i has exactly one bad facility. Now before each execution of step 4.1, we have $|A_i| < |B_i|$. This together with the invariant that $|S| = |O|$ implies that in step 4.1, we can always find a good facility in $S \setminus A_i$. Since with each execution of the while loop in step 4 the size of A_i increases, the loop terminates. The condition in step 4 then ensures that for $1 \leq i \leq r-1$, we have $|A_i| = |B_i|$. Since there are no bad facilities left when the procedure comes out of the outer while loop, we have that the set A_r contains only good facilities. ▯

We now use this partition of S and O to define the swaps we would consider for our analysis. We also associate a positive real weight with each such swap.

1. If $|A_i| = |B_i| \leq p$ for some $1 \leq i \leq r$, then we consider the swap $\langle A_i, B_i \rangle$ with weight 1. From the local optimality of S we have

$$cost((S \setminus A_i) \cup B_i) - cost(S) \geq 0.$$

   Note that even if $A_i \cap B_i \neq \emptyset$ or $S \cap B_i \neq \emptyset$, the above inequality continues to hold.

2. If $|A_i| = |B_i| = q > p$, we consider all possible swaps $\langle s, o \rangle$ where $s \in A_i$ is a good facility and $o \in B_i$. Note that if $i \neq r$, there are exactly $q-1$ good facilities in A_i, and for $i = r$ we select any $q-1$ out of the q good facilities in A_r. We associate a weight of $1/(q-1)$ with each of these $q(q-1)$ swaps. For each such swap $\langle s, o \rangle$, we have

$$cost(S - s + o) - cost(S) \geq 0.$$

Note that any good facility in A_i is considered in swaps of total weight at most $q/(q-1) \leq (p+1)/p$. The swaps we have considered and the weights we assigned to them satisfy the following properties:

1. For every facility $o \in O$, the sum of weights of the swaps $\langle A, B\rangle$ with $o \in B$ is exactly one.
2. For every facility $s \in S$, the sum of weights of the swaps $\langle A, B\rangle$ with $s \in A$ is at most $(p+1)/p$.
3. If a swap $\langle A, B\rangle$ is considered, then $capture(A) \subseteq B$.

For each facility $o \in O$, we partition $N_O(o)$ as follows:

1. For $|A_i| \leq p, 1 \leq i \leq r$, let $N^o_{A_i} = N_S(A_i) \cap N_O(o)$ be a set in the partition.
2. For $|A_i| > p, 1 \leq i \leq r$, and all $s \in A_i$, let $N^o_s = N_S(s) \cap N_O(o)$ be a set in the partition.

As before, for each facility $o \in O$, we consider a 1-1 and onto mapping $\pi : N_O(o) \to N_O(o)$ with the following property.

Property 3.2. *For all sets P, in the partition of $N_O(o)$ for which $|P| \leq \frac{1}{2}|N_O(o)|$, we have $\pi(P) \cap P = \emptyset$.* Such a mapping π can be defined in a manner identical to the one described in section 3.2. The analysis is similar to the one presented for the single-swap heuristic. For each of the swaps defined above, we upper-bound the increase in the cost by reassigning the clients. Property 3.2 ensures that the function π can be used to do the reassignment as described in section 3.2. We take a weighted sum of the inequalities corresponding to each of the swaps considered above. Recall that in the single-swap analysis, we used the fact that each facility in S was considered in at most two swaps and upper-bounded the second term of (2) by $2\sum_{j\in C}(O_j + O_{\pi(j)} + S_{\pi(j)} - S_j) = 4 \cdot cost(O)$. Similarly, we can now make use of the fact that each facility in S is considered in swaps with total weight at most $(p+1)/p$ and upper-bound the second term by $(p+1)/p \cdot \sum_{j\in C}(O_j + O_{\pi(j)} + S_{\pi(j)} - S_j) = 2(p+1)/p \cdot cost(O)$. This gives us a locality gap of $1 + 2(p+1)/p = 3 + 2/p$.

3.5. Tight example. In Figure 9, we show an instance of the k-median problem in which a solution that is locally optimum for the 2-swap heuristic ($p = 2$) has cost at least $4 - o(1)$ times the cost of the global optimum. Since $3 + 2/p = 3 + 2/2 = 4$ is also the locality gap proved, it shows that the analysis of the 2-swap heuristic is tight. This tight example can be generalized for p-swaps for any $p \geq 1$. In Figure 9, the black squares are the facilities $\{o_1, o_2, \dots, o_k\}$ opened by a solution O, the gray squares are the facilities $\{s_1, s_2, \dots, s_k\}$ opened by a locally optimum solution S, and the circles are the clients. In the graph in Figure 9 each edge has a label which is its length. The cost of serving a client j by a facility i is length of the shortest path between client j and facility i in the graph; the cost is infinite if there is no path.

Note that $cost(S) = \frac{8k-10}{3}, cost(O) = \frac{2k+2}{3}$, and hence the ratio $cost(S)/cost(O)$ approaches 4 as k approaches ∞. We now show that S is a locally optimum solution; that is, if we swap $\{o_l, o_m\}$ for $\{s_i, s_j\}$, then the cost does not decrease. To show this we consider various cases:

1. $i, j \leq r$. Then o_l, o_m will have to lie in the connected components containing s_i, s_j. But in this case the cost would increase by 4.
2. $i \leq r < j$. At least one of o_l, o_m would have to lie in the connected component containing s_i; let this be o_l. If o_m also lies in this component, then the cost remains unchanged. If o_m is in a different component and $m \leq k - 2$, then the cost increases by 2. If $m > k - 2$, then the cost of the solution increases by 3.

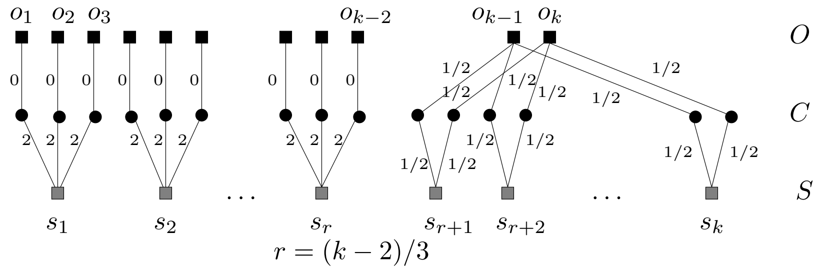

FIG. 9. *Tight example for 2-swap. The same example can be generalized to p-swap.*

3. $i, j > r$. If both l, m are at most $k-2$, then the cost of the solution remains unchanged. The cost remains unchanged even if $l \leq k-2 < m$. If both l, m are larger than $k-2$, then, once again, the cost of the solution remains unchanged.

4. Uncapacitated facility location.

In facility location problems, we can open any number of facilities, but each facility $i \in F$ has a cost $f_i \geq 0$ of opening it. The UFL problem is to identify a subset $S \subseteq F$ and to serve the clients in C by the facilities in S such that the sum of facility costs and service costs is minimized. That is, if a client $j \in C$ is assigned to a facility $\sigma(j) \in S$, then we want to minimize $cost(S) = \sum_{i \in S} f_i + \sum_{j \in C} c_{\sigma(j)j}$. Note that for a fixed S, serving each client by the nearest facility in S minimizes the service cost.

4.1. A local search procedure.

We present a local search procedure for the metric UFL problem with a locality gap of 3. The operations allowed in a local search step are adding a facility, deleting a facility, and swapping facilities. Hence the neighborhood $\mathcal{B}$ is defined by

$$\mathcal{B}(S) = \{S + \{s'\}\} \cup \{S - \{s\} \mid s \in S\} \cup \{S - \{s\} + \{s'\} \mid s \in S\}. \tag{4}$$

As the number of neighbors to be checked at each local search step is polynomial, the algorithm can be run in polynomial time as described earlier.

Charikar and Guha [4] proved a locality gap of 3 for a local search procedure where the operation was of adding a facility and dropping zero or more facilities. Korupolu, Plaxton, and Rajaraman [14] considered the operations of adding, deleting, and swapping a facility but could only prove a locality gap of at most 5.

4.2. The analysis.

For any set of facilities $S' \subseteq F$, let $cost_f(S') = \sum_{i \in S'} f_i$ denote the facility cost of the solution S'. Also, let $cost_s(S')$ be the total cost of serving the clients in C by the nearest facilities in S'. The total cost of a solution S' is denoted by $cost(S')$. We use S to denote a locally optimum solution. The following bound on the service cost of S has earlier been proved by Korupolu, Plaxton, and Rajaraman [14].

LEMMA 4.1 (service cost).

$$cost_s(S) \leq cost_f(O) + cost_s(O).$$

Proof. Consider an operation in which a facility $o \in O$ is added. Assign all the clients $N_O(o)$ to o. From the local optimality of S we get $f_o + \sum_{j \in N_O(o)} (O_j - S_j) \geq 0$. Note that even if $o \in S$, this inequality continues to hold. If we add such inequalities for every $o \in O$, we get the desired inequality. ∎

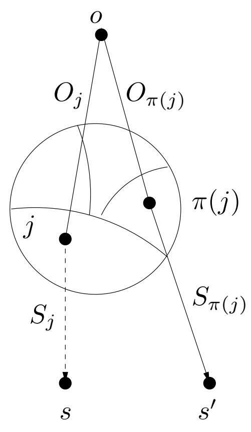

FIG. 10. *Bounding the facility cost of a good facility s.*

The following lemma gives a bound on the facility cost of S.

LEMMA 4.2 (facility cost).

$$cost_f(S) \le cost_f(O) + 2 \cdot cost_s(O).$$

Proof. As before, we assume that for a fixed $o \in O$, the mapping $\pi : N_O(o) \to N_O(o)$ is 1-1 and onto and satisfies Property 3.1. In addition, we assume that if $|N_s^o| > \frac{1}{2}|N_O(o)|$, then for all $j \in N_s^o$ for which $\pi(j) \in N_s^o$, we have that $\pi(j) = j$. Here we give an outline of how to define such a function π. Let $|N_s^o| > \frac{1}{2}|N_O(o)|$. We pick any $|N_s^o| - |N_O(o) \setminus N_s^o|$ clients j from N_s^o and set $\pi(j) = j$. On the remaining clients in $N_O(o)$, the function π is defined in the same manner as in section 3.2.

Recall that a facility $s \in S$ is *good* if it does not capture any o, that is, for all $o \in O$, $|N_s^o| \le \frac{1}{2}|N_O(o)|$. The facility cost of good facilities can be bounded easily as follows (see Figure 10). Consider an operation in which a good facility $s \in S$ is dropped. Let $j \in N_S(s)$ and $\pi(j) \in N_S(s')$. As s does not capture any facility $o \in O$, we have that $s' \ne s$. If we assign j to s', then we have $-f_s + \sum_{j \in N_S(s)}(O_j + O_{\pi(j)} + S_{\pi(j)} - S_j) \ge 0$. Since for all $j \in N_S(s), \pi(j) \ne j$, the term $\sum_{\substack{j \in N_S(s)\\ \pi(j)=j}} O_j$ is trivially zero and hence we can rewrite the above inequality as

$$-f_s + \sum_{\substack{j \in N_S(s)\\ \pi(j)\ne j}} (O_j + O_{\pi(j)} + S_{\pi(j)} - S_j) + 2 \sum_{\substack{j \in N_S(s)\\ \pi(j)=j}} O_j \ge 0. \tag{5}$$

For bounding the facility cost of a bad facility $s \in S$ we proceed as follows. Fix a bad facility $s \in S$. Suppose s captures the facilities $P \subseteq O$. Let $o \in P$ be the facility nearest to s. We consider the swap $\langle s, o\rangle$. The clients $j \in N_S(s)$ are now assigned to the facilities in $S - s + o$ as follows:

1. Suppose $\pi(j) \in N_S(s')$ for $s' \ne s$. Then j is assigned to s'. Let $j \in N_O(o')$. We have, $c_{js'} \le c_{jo'} + c_{\pi(j)o'} + c_{\pi(j)s'} = O_j + O_{\pi(j)} + S_{\pi(j)}$ (Figure 11).
2. Suppose $\pi(j) = j \in N_S(s)$ and $j \in N_O(o)$. Then j is assigned to o.
3. Suppose $\pi(j) = j \in N_S(s)$ and $j \in N_O(o')$ for $o' \ne o$. By Property 3.1 of the mapping π, facility s captures facility o' and hence $o' \in P$. The client j is now assigned to facility o. From triangle inequality, $c_{jo} \le c_{js} + c_{so}$. Since $o \in P$ is the closest facility to s, we have $c_{so} \le c_{so'} \le c_{js} + c_{jo'}$. Therefore, $c_{jo} \le c_{js} + c_{js} + c_{jo'} = S_j + S_j + O_j$ (Figure 11).

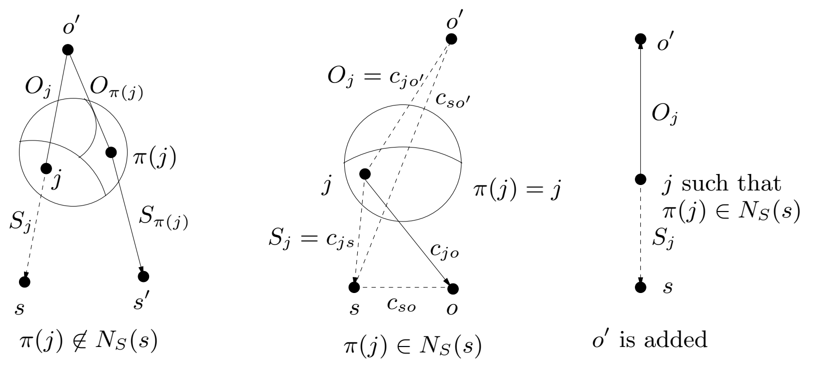

FIG. 11. *Bounding the facility cost of a bad facility s. The figures on the left and in the middle show reassignment when s is dropped, and the figure on the right shows reassignment when o' is added.*

Thus for the swap $\langle s, o\rangle$ we get the following inequality:

$$\begin{aligned} f_o - f_s + \sum_{\substack{j \in N_S(s)\\ \pi(j)\neq j}} (O_j + O_{\pi(j)} + S_{\pi(j)} - S_j) \\ + \sum_{\substack{j\in N_O(o),\\ \pi(j)=j\in N_S(s)}} (O_j - S_j) + \sum_{\substack{j\notin N_O(o),\\ \pi(j)=j\in N_S(s)}} (S_j + S_j + O_j - S_j) \geq 0. \end{aligned} \tag{6}$$

Now consider an operation in which a facility $o' \in P - o$ is added (Figure 11). The clients $j \in N_O(o')$ for which $\pi(j) = j \in N_S(s)$ are now assigned to the facility o', and this yields the following inequality.

$$f_{o'} + \sum_{\substack{j\in N_O(o')\\ \pi(j)=j\in N_S(s)}} (O_j - S_j) \geq 0 \qquad \text{for each } o' \in P - o. \tag{7}$$

Adding inequality (6) to inequalities (7) we get, for a bad facility $s \in S$,

$$\sum_{o'\in P} f_{o'} - f_s + \sum_{\substack{j\in N_S(s),\\ \pi(j)\neq j}} (O_j + O_{\pi(j)} + S_{\pi(j)} - S_j) + 2 \sum_{\substack{j\in N_S(s),\\ \pi(j)=j}} O_j \geq 0. \tag{8}$$

The last term on the left is an upper bound on the sum of the last two terms on the left of inequality (6) and the last term on the left of the inequality (7) added for all $o' \in P - o$.

Now we add inequalities (5) for all good facilities $s \in S$, inequalities (8) for all bad facilities $s \in S$, and inequalities $f_o \geq 0$ for all $o \in O$, which are not captured by any $s \in S$, to obtain

$$\sum_{o\in O} f_o - \sum_{s\in S} f_s + \sum_{\pi(j)\neq j} (O_j + O_{\pi(j)} + S_{\pi(j)} - S_j) + 2 \sum_{\pi(j)=j} O_j \geq 0.$$

Note that $\sum_{j:\pi(j)\neq j} O_j = \sum_{j:\pi(j)\neq j} O_{\pi(j)}$ and $\sum_{j:\pi(j)\neq j} S_j = \sum_{j:\pi(j)\neq j} S_{\pi(j)}$. Therefore we have $\sum_{j:\pi(j)\neq j}(O_j + O_{\pi(j)} + S_{\pi(j)} - S_j) = 2\sum_{j:\pi(j)\neq j} O_j$ and hence $cost_f(O) - cost_f(S) + 2 \cdot cost_s(O) \geq 0$. This proves the desired lemma. $\square$

Combining Lemmas 4.1 and 4.2, we get the following result.

THEOREM 4.3. *The local search procedure for the metric UFL problem with the neighborhood structure $\mathcal{B}$ given by $\mathcal{B}(S) = \{S + \{s'\}\} \cup \{S - \{s\} \mid s \in S\} \cup \{S - \{s\} + \{s'\} \mid s \in S\}$ has a locality gap of at most* 3.

The algorithm described above extends very simply to the case when the clients $j \in C$ have arbitrary demands $d_j \geq 0$ to be served. We now show how to use a technique from [4] to obtain $1 + \sqrt{2} + \epsilon \approx 2.414 + \epsilon$ approximation to the UFL. The main idea is to exploit the asymmetry in the service and facility cost guarantees.

Note that Lemmas 4.1 and 4.2 hold for any solution O and not just the optimal solution. We multiply the facility costs by a suitable factor $\alpha > 0$ and solve the new instance using local search.

THEOREM 4.4. *The metric UFL problem can be approximated to factor* $1+\sqrt{2}+\epsilon$ *using a local search procedure.*

Proof. As before, we denote the facility cost and the service cost of an optimum solution O by $cost_f(O)$ and $cost_s(O)$, respectively. Let $cost'_f(A)$ and $cost'_s(A)$ denote the facility and service costs of a solution A in the scaled instance and let S be a locally optimum solution. Then

$$\begin{aligned} cost_f(S) + cost_s(S) &= \frac{cost'_f(S)}{\alpha} + cost'_s(S) \\ &\leq \frac{cost'_f(O) + 2cost'_s(O)}{\alpha} + cost'_f(O) + cost'_s(O) \\ &= (1+\alpha)cost_f(O) + \left(1 + \frac{2}{\alpha}\right) cost_s(O). \end{aligned}$$

The inequality follows from Lemmas 4.1 and 4.2. Now, by setting $\alpha = \sqrt{2}$, we get $cost(S) \leq (1 + \sqrt{2})cost(O)$. Thus local search can be used to obtain a $1 + \sqrt{2} + \epsilon$ approximation. ▯

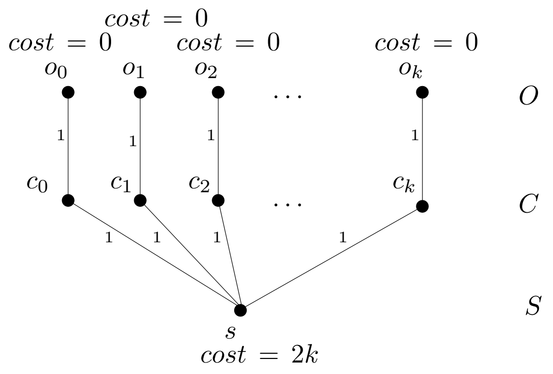

FIG. 12. *Tight example for the locality gap of UFL.*

4.3. Tight example. In Figure 12, we show an instance where a local optimum has cost at least $3 - o(1)$ times the cost of the global optimum. The locally optimum solution S consists of a single facility s while the optimum solution O consists of facilities $\{o_0, o_1, \ldots, o_k\}$. All edges shown have unit lengths, and the cost of serving client j by facility f is the length of the shortest path between client j and facility f in the graph. The cost of opening facility s is $2k$, while that of opening any other facility is zero. To argue that the solution S is locally optimum, note that we cannot delete facility s. It is also easy to verify that we cannot decrease the cost of our solution by either the addition of any facility from O or by any swap which involves bringing in a facility from O and deleting s. Thus S is locally optimum and has cost $3k+1$, while the cost of O is $k+1$. Since the ratio $cost(S)/cost(O)$ tends to 3 as k tends to ∞, our analysis of the local search algorithm is tight.

5. The capacitated facility location problem.

In the CFL problem, along with the facility costs $f_i \geq 0$, we are given integer capacities $u_i > 0$ for each $i \in F$. We can open multiple copies of a facility i. Each copy incurs a cost f_i and is capable of serving at most u_i clients. Note that the capacities u_i may be *different* for different facilities i. The problem is to identify a multiset S of facilities and to serve the clients in C by the facilities in S such that the capacity constraints are satisfied and the sum of the facility costs and the service costs is minimized. Since the clients have unit demands and the facilities have integer capacities, every client will get assigned to a single facility. If a client $j \in C$ is assigned to a facility $\sigma(j) \in S$, then we want to minimize $cost(S) = \sum_{i\in S} f_i + \sum_{j\in C} c_{j\sigma(j)}$. Given an S, the service cost can be minimized by solving a min-cost assignment problem.

In the remainder of this section we let S and O be the multisets of the facilities opened in the locally optimum solution and an optimum solution, respectively.

5.1. A local search algorithm.

In this section, we prove a locality gap of at most 4 on a local search procedure for the CFL problem. The operations allowed at each local search step are adding a single copy of a facility $s' \in F$ or adding $l \geq 1$ copies of a facility $s' \in F$ and dropping a subset of the open facilities, $T \subseteq S$. For the second operation l should be sufficiently large so that the clients $j \in N_S(T)$ can be served by these new copies of s', that is, $l \cdot u_{s'} \geq |N_S(T)|$. So the neighborhood structure $\mathcal{B}$ is defined by

$$\mathcal{B}(S) = \{S + s' \mid s' \in F\} \cup \{S - T + l \cdot \{s'\} \mid s' \in F, T \subseteq S, l \cdot u_{s'} \geq |N_S(T)|\}, \tag{9}$$

where $l \cdot \{s'\}$ represents l new copies of s'. If we service all clients in $N_S(T)$ by the new copies of facility s', the cost of the new solution is at most

$$cost(S) + l \cdot f_{s'} + \sum_{s\in T} \left(-f_s + \sum_{j\in N_S(s)} (c_{js'} - c_{js}) \right).$$

Given a facility $s' \in F$, we use the procedure `T-hunt` described in Figure 13 to find a subset, $T \subseteq S$, of facilities to close. Here $m = |C|$ is an upper bound on the number of new copies of s' that we need to open. Closing a facility $s \in S$ gives an extra $|N_S(s)|$ clients to be served by the new facility s'. A client $j \in N_S(s)$ now travels an extra distance of at most $(c_{s'j} - c_{sj})$. Thus, closing facility s gives a *savings* of $f_s - \sum_{j\in N_S(s)}(c_{s'j} - c_{sj})$. Due to capacity constraints, a copy of s' can serve at most $u_{s'}$ clients. This motivates us to define the following `Knapsack` problem. For a facility $s \in S$, define $\texttt{weight}(s) = |N_S(s)|$ and $\texttt{profit}(s) = f_s - \sum_{j\in N_S(s)}(c_{s'j} - c_{sj})$. The oracle $\texttt{Knapsack}(W)$ returns a multiset $T \subseteq S$ such that $\sum_{s\in T} \texttt{weight}(s) \leq W$ and $\texttt{profit}(T) = \sum_{s\in T} \texttt{profit}(s)$ is maximized.

It is interesting to note that since we permit any subset of facilities T from our current solution S to be dropped, the number of operations is exponential in $|S|$. However, by counting the change in cost due to each such operation in a specific way, we are able to give a polynomial time procedure (the procedure `T-hunt`) to identify a local operation which improves the cost. It might be the case that `T-hunt` is not able

```
    Procedure T-Hunt.

1.  For $l = 1$ to $m$ do,
2.      $T \leftarrow \texttt{Knapsack}(l \cdot u_{s'})$.
3.      If $cost(S) + l \cdot f_{s'} - \texttt{profit}(T) < cost(S)$,
            then return $T$.
4.  return "could not find a solution that reduces the cost."
```

FIG. 13. *A procedure to find a subset $T \subseteq S$ of facilities.*

to identify a local operation which improves the cost even though such operations exist. However, our analysis will work only with the assumption that `T-hunt` could not find a solution which improves the cost.

5.2. The analysis. As the output S is locally optimum with respect to additions, Lemma 4.1 continues to bound the service cost of S. We restate Lemma 4.1 here.

LEMMA 5.1 (service cost).

$$cost_s(S) \leq cost_f(O) + cost_s(O).$$

LEMMA 5.2. *For any $U \subseteq S$ and any $s' \in F$, we have*

$$\lceil |N_S(U)|/u_{s'} \rceil \cdot f_{s'} + \sum_{s \in U} |N_S(s)| \cdot c_{ss'} \geq \sum_{s \in U} f_s.$$

Proof. The algorithm terminated with the output S. Hence for the solution S and for the facility s', the procedure `T-hunt` must have returned "could not find a solution that reduces the cost." Consider the run of the for-loop for $l = \lceil |N_S(U)|/u_{s'} \rceil$. Since $\sum_{s \in U} \texttt{weight}(s) = N_S(U) \leq l \cdot u_{s'}$, the solution T returned by the knapsack oracle has profit at least as large as `profit`(U). Hence,

$$0 \leq l \cdot f_{s'} - \texttt{profit}(T) \leq l \cdot f_{s'} - \texttt{profit}(U) = l \cdot f_{s'} - \sum_{s \in U} \left(f_s - \sum_{j \in N_S(U)} (c_{js'} - c_{js}) \right).$$

However, by triangle inequality we have $c_{js'} - c_{js} \leq c_{ss'}$. Therefore we have proved the lemma. □

We are now ready to bound the facility cost of S.

LEMMA 5.3 (facility cost).

$$cost_f(S) \leq 3 \cdot cost_f(O) + 2 \cdot cost_s(O).$$

To prove the above lemma, we consider a directed graph $G = (V, E)$ with lengths on edges, where

$$V = \{v_s \mid s \in S\} \cup \{w_o \mid o \in O\} \cup \{sink\},$$

$$E = \{(v_s, w_o) \mid s \in S, o \in O\} \cup \{(w_o, sink) \mid o \in O\}.$$

The lengths of (v_s, w_o) and $(w_o, sink)$ are c_{so} and f_o/u_o, respectively (see Figure 14). The cost of routing unit flow along any edge is equal to the length of that edge. We want to simultaneously route $|N_S(s)|$ units of flow from each v_s to the *sink*.

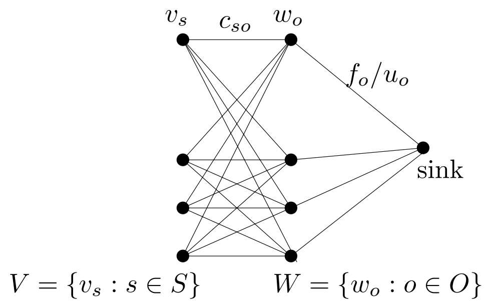

FIG. 14. *The flow graph.*

LEMMA 5.4. *We can simultaneously route $|N_S(s)|$ units of flow from each v_s to the sink such that the total routing cost is at most $cost_s(S) + cost_s(O) + cost_f(O)$.*

Proof. Consider a client $j \in C$. If $j \in N_s^o$, then route one unit of flow along the path $v_s \to w_o \to sink$. Triangle inequality implies $c_{so} \leq S_j + O_j$. If for each client we route a unit flow in this manner, then the edge $(w_o, sink)$ carries $N_O(o)$ units of flow at cost $|N_O(o)| \cdot f_o/u_o \leq \lceil |N_O(o)|/u_o \rceil \cdot f_o$, which is the contribution of o to $cost_f(O)$. Thus, the routing cost of this flow is at most $cost_s(S) + cost_s(O) + cost_f(O)$. □

Since there are no capacities on the edges of graph G, any minimum cost flow must route all $N_S(s)$ units of flow from v_s to the sink along the shortest path. This would be a path (v_s, w_o, sink), where o is such that $c_{so} + f_o/u_o$ is minimized with ties broken arbitrarily. For each $o \in O$, let $T_o \subseteq S$ denote the set of facilities s that route their flow via w_o in this minimum cost flow. From Lemma 5.4, we have

$$cost_s(S) + cost_s(O) + cost_f(O) \geq \sum_{o \in O} \sum_{s \in T_o} |N_S(s)|(c_{so} + f_o/u_o). \tag{10}$$

Now, applying Lemma 5.2 to T_o and o, we get

$$\lceil |N_S(T_o)|/u_o \rceil \cdot f_o + \sum_{s \in T_o} |N_S(s)| \cdot c_{so} \geq \sum_{s \in T_o} f_s.$$

Hence,

$$f_o + |N_S(T_o)|/u_o \cdot f_o + \sum_{s \in T_o} |N_S(s)| \cdot c_{so} \geq \sum_{s \in T_o} f_s.$$

Adding these inequalities for all $o \in O$, we get

$$\sum_{o \in O} f_o + \sum_{o \in O} \sum_{s \in T_o} |N_S(s)|(c_{so} + f_o/u_o) \geq \sum_{o \in O} \sum_{s \in T_o} f_s = cost_f(S). \tag{11}$$

The inequalities (10) and (11) together imply

$$cost_f(S) \leq 2 \cdot cost_f(O) + cost_s(O) + cost_s(S).$$

This inequality, together with Lemma 5.1, gives Lemma 5.3. Combining Lemmas 5.1 and 5.3, we obtain the following result.

THEOREM 5.5. *A local search procedure for the metric CFL problem where in each step we can either add a facility or delete a subset of facilities and add multiple copies of a facility has a locality gap of at most* 4.

Using an argument similar to the one in Theorem 4.4 with $\alpha = \sqrt{3} - 1$ we obtain a $2 + \sqrt{3} + \epsilon \approx 3.732 + \epsilon$ approximation. The tight example given in section 4.3 for the UFL problem shows that a locally optimum solution for this problem can have cost three times the cost of the global optimum.

6. Conclusions and open problems.

In this paper, we provided tighter analysis of local search procedures for the k-median and UFL problems. Our sharper analysis leads to a $3 + 2/p + \epsilon$ approximation algorithm for the k-median in which there are polynomially many local search steps, each of which can be performed in time $n^{O(p)}$. For CFL, when multiple copies of a facility can be opened, we introduce a new operation and show how a weaker version of this operation can be performed in polynomial time. This leads to a local search procedure with a locality gap of at most 4. We leave open the problem of obtaining tight bounds on the locality gap of this procedure. It would be interesting to identify such operations for other variants of facility location problems.

Acknowledgments. Arya, Garg, Khandekar, and Pandit first published a preliminary version of this paper which did not include the results in sections 3.3 and 3.4. After this version was distributed, Meyerson–Munagala and Arya–Garg–Khandekar–Pandit independently obtained the results in sections 3.3 and 3.4. Garg and Khandekar would like to thank R. Ravi, Amitabh Sinha, and Goran Konjevod for useful discussions.

REFERENCES

[1] A. Archer, R. Rajagopalan, and D. B. Shmoys, *Lagrangian relaxation for the k-median problem: New insights and continuity properties*, in Proceedings of the 11th Annual European Symposium on Algorithms, Springer-Verlag, New York, 2003, pp. 31–42.

[2] E. Arkin and R. Hassin, *On local search for weighted k-set packing*, Math. Oper. Res., 23 (1998), pp. 640–648.

[3] B. Chandra, H. Karloff, and C. Tovey, *New results on the old k-opt algorithm for the traveling salesman problem*, SIAM J. Comput., 28 (1999), pp. 1998–2029.

[4] M. Charikar and S. Guha, *Improved combinatorial algorithms for the facility location and k-median problems*, in Proceedings of the 40th Annual Symposium on Foundations of Computer Science, IEEE Computer Society Press, Los Alamitos, CA, 1999, pp. 378–388.

[5] M. Charikar, S. Guha, E. Tardos, and D. Shmoys, *A constant-factor approximation algorithm for the k-median problem*, in Proceedings of the 31st Annual ACM Symposium on Theory of Computing, ACM, New York, 1999, pp. 1–10.

[6] F. Chudak, *Improved approximation algorithms for uncapacitated facility location problem*, in Proceedings of the 6th Conference on Integer Programming and Combinatorial Optimization, 1998, pp. 182–194.

[7] F. A. Chudak and D. B. Shmoys, *Improved approximation algorithms for a capacitated facility location problem*, in Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms, ACM, New York, SIAM, Philadelphia, 1999, pp. 875–876.

[8] F. Chudak and D. Williamson, *Improved approximation algorithms for capacitated facility location problems*, in Proceedings of the 7th Conference on Integer Programming and Combinatorial Optimization, 1999, pp. 99–113.

[9] S. Guha and S. Khuller, *Greedy strikes back: Improved facility location algorithms*, in Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms, ACM, New York, SIAM, Philadelphia, 1998, pp. 649–657.

[10] K. Jain, M. Mahdian, E. Markakis, A. Saberi, and V. Vazirani, *Greedy facility location algorithms analyzed using dual fitting factor revealing LP*, J. ACM, 50 (2003), pp. 795–824.

[11] K. Jain and V. Vazirani, *Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation*, J. ACM, 48 (2001), pp. 274–296.

[12] S. Khuller, R. Bhatia, and R. Pless, *On local search and placement of meters in networks*, SIAM J. Comput., 32 (2003), pp. 470–487.

[13] J. Könemann and R. Ravi, *A matter of degree: Improved approximation algorithms for degree-bounded minimum spanning trees*, in Proceedings of the 32nd Annual ACM Symposium on Theory of Computing, ACM, New York, 2000, pp. 537–546.

[14] M. Korupolu, C. Plaxton, and R. Rajaraman, *Analysis of a Local Search Heuristic for Facility Location Problems*, Technical Report 98-30, DIMACS, Rutgers University, Piscataway, NJ, 1998.

[15] S. Lin and B. Kernighan, *An effective heuristic algorithm for the travelling salesman problem*, Oper. Res., 21 (1973), pp. 498–516.

[16] M. Mahdian and M. Pál, *Universal facility location*, in Proceedings of 11th Annual European Symposium on Algorithms, Springer-Verlag, New York, 2003, pp. 409–422.

[17] M. Mahdian, Y. Ye, and J. Zhang, *Improved approximation algorithms for metric facility location problems*, in Proceedings of the 5th International Workshop on Approximation Algorithms for Combinatorial Optimization, 2002, pp. 229–242.

[18] M. Mahdian, Y. Ye, and J. Zhang, *A 2-approximation algorithm for the soft-capacitated facility location problem*, in Proceedings of the 6th International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX), 2003.

[19] M. Pál, E. Tardos, and T. Wexler, *Facility location with nonuniform hard capacities*, in Proceedings of the 42nd Annual Symposium on Foundations of Computer Science, IEEE Computer Society Press, Los Alamitos, CA, 2001, pp. 329–338.

[20] D. Shmoys, E. Tardos, and K. Aardal, *Approximation algorithms for facility location problems*, in Proceedings of the 29th Annual ACM Symposium on Theory of Computing, ACM, New York, 1997, pp. 265–274.