Local Search Heuristics for k-median and Facility Location Problems

Vijay Arya[*]
Naveen Garg [†]
Rohit Khandekar[†]
Dept of Comp Sc. & Engg.
Indian Institute of Technology
New Delhi - 110016

Adam Meyerson[‡]
Kamesh Munagala[§]
Dept of Computer Sc.
Stanford University CA 94305

Vinayaka Pandit
IBM India Research Lab
Indian Institute of Technology
New Delhi - 110016

ABSTRACT

In this paper, we analyze local search heuristics for the k-median and facility location problems. We define the *locality gap* of a local search procedure 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 exactly 5. When we permit p facilities to be swapped simultaneously then the locality gap of the local search procedure is exactly $3 + 2/p$. This is the first analysis of 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 exactly 3. This improves the 5 bound of Korupolu et al. 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 operation which opens one or more copies of a facility and drops zero or more facilities. We prove that local search which permits this new operation has a locality gap between 3 and 4.

[*]Partially supported by a fellowship from IBM India Research Lab.

[†]Partially supported by a fellowship from Infosys Technologies Ltd.,Bangalore.

[‡]Supported by ARO DAAG-55-97-1-0221.

[§]Supported by ONRN00014-98-1-0589.

STOC'01, July 6-8, 2001, Hersonissos, Crete, Greece.

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 distances 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 classical facility location problem. Note that in this setting the facility costs need not be 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 [7] 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. One setting where local search is amenable to a worst-case analysis is facility location.

For an instance I, let $\texttt{global}(I)$ denote the global optimum and $\texttt{local}(I)$ be the locally optimum solution provided by a certain local search heuristic. We call the supremum of the ratio $\texttt{global}(I)/\texttt{local}(I)$, the *locality gap* of this local search procedure. For 1-CFL with uniform capacities, Korupolu et al. [8, 9] 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 [4] to yield a locality gap of at most 6. For the uncapacitated version, Korupolu et al. [8, 9] provide 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 [1] 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 et al. [8, 9] gave a local search procedure which permitted adding, deleting and swapping facilities and gave a solution with $k(1+\epsilon)$ 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 [10] and Charikar, Guha, Shmoys and Tardos [11]. They formulated the problems as linear programs and rounded the optimum fractional solution to obtain a $6\frac{2}{3}$ approximation for k-median and a 3 approximation for the uncapacitated facility location problem. Jain and Vazirani [6] gave an alternate 3 approximation algorithm for the uncapacitated facility location problem (UFL) using the primal-dual schema. They also observed that k-median can be viewed as a Lagrange-relaxation of UFL and utilized this to give a 6 approximation algorithm for k-median. Charikar and Guha [1] improved this to a 4 approximation. Guha and Khuller [5] employed randomization to improve the approximation guarantee of UFL to 2.408. This was further improved to $(1+2/e)$ by Chudak [2] and finally to 1.728 by Charikar and Guha [1]. Similar ideas were used by Chudak and Shmoys [3] to obtain a 3 approximation algorithm for ∞-CFL when the capacities are uniform. Jain and Vazirani [6] obtained a 4 approximation algorithm for ∞-CFL when the capacities were non-uniform by solving a related UFL problem using their primal-dual algorithm.

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

1. For k-median, we show that local search with single swaps has a locality gap of 5. This is the first analysis of 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 [1]. 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 $(3+2/p)$ times that of the global optimum.

2. For UFL, we show that local search, which permits adding, dropping and swapping a facility, has a locality gap of 3. This improves the 5 bound of Korupolu et al. [8]. Our analysis of the algorithm is tight, that is, we show a locally optimal solution which has cost 3 times that of the optimum solution. Using standard scaling techniques [1] our algorithm can be improved to achieve a $1+\sqrt{2}\approx 2.414$ approximation.

3. For ∞-CFL, we consider the setting when the capacities are non-uniform 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. As for UFL, we give a polynomial algorithm that uses $\texttt{Knapsack}$ as a subroutine to search a subspace of adjacent solutions. We also show an instance where the polynomial time algorithm cannot find an adjacent solution of lower cost and which has cost 3 times the optimum. Again using scaling techniques [1] the algorithm can be improved to obtain a $2+\sqrt{3}\approx 3.732$ approximation.

The paper is organized as follows. Section 2 introduces some notation. In Section 3, we prove a locality gap of 5 for the k-median problem when only single swaps are permitted; in Section 3.3, we show how the above analysis can be extended to argue a locality gap of $3+2/p$ when up to p facilities can be swapped simultaneously. Section 4 and Section 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*. There is a specified *distance* $c_{ij}\geq 0$ between every pair $i,j\in F\cup C$. In these problems, the goal is to identify a subset of facilities $S\subseteq F$ and to serve the clients in C by the facilities in S, such that some cost function is minimized. The facilities in S are said to be *open*. The metric versions of these problems assume that the distances c_{ij} are symmetric and satisfy the triangle inequality.

The algorithm that we analyze is described in Figure 1. Here $\epsilon>0$ is a constant, $n=|F|$ is the number of facilities, $m=|C|$ is the number of clients and $p(n,m)$ is a polynomial in n and m. The cost function $cost(S)$ and the operation $\texttt{op}(S)$ will be defined differently for different problems.

Algorithm `Local Search.`

1. $S\leftarrow$ an arbitrary feasible solution.
2. While $\exists$ an operation $\texttt{op}$ such that, $cost(\texttt{op}(S))\leq(1-\frac{\epsilon}{p(n,m)})cost(S)$, do $S\leftarrow\texttt{op}(S)$.
3. return S.

Figure 1: A generic local search algorithm for k-median and facility location problems

An operation op is called *admissible* for S if $cost(\mathsf{op}(S)) \leq (1-\epsilon/p(n,m))cost(S)$. At any execution of the step 2 of the algorithm, there will be at most a polynomial number of ops to be checked for admissibility. Also during each admissible op, the cost of the current solution decreases by a factor of at least $\epsilon/p(n,m)$. If S^* denotes an optimum solution and S_0 denotes the initial solution, then the number of ops that the algorithm does is at most $\log(cost(S_0)/cost(S^*))/\log\frac{1}{1-\epsilon/p(n,m)}$ where $p(n,m)$ is some polynomial in $n=|F|$ and $m=|C|$. As $\log(cost(S_0))$ is polynomial in the input size and performing each op takes a polynomial time, this algorithm terminates in polynomial time.

When there are no admissible operations, we know that every operation op reduces the cost by factor of at most $\epsilon/p(n,m)$, that is,

$$cost(\mathsf{op}(S)) \;\geq\; (1-\frac{\epsilon}{p(n,m)})\;cost(S)$$

To simplify the exposition, we work with the simplifying assumption that no operation improves the cost of the solution, that is, for every operation op,

$$cost(\mathsf{op}(S)) \;\geq\; cost(S)$$

We will add at most $p(n,m)$ of such inequalities to conclude that $cost(S) \leq \alpha \cdot cost(S^*)$ for some $\alpha \geq 1$. Adding the corresponding original inequalities implies that $cost(S) \leq \alpha(1+\epsilon)cost(S^*)$. Thus our proof that a certain local search procedure has locality gap α translates into a $\alpha(1+\epsilon)$ approximation algorithm.

We use the following notations. Let S denote the output of the algorithm and S^* denote an optimum solution. The cost of serving a client j, or the *service cost* of j, is the distance between j and the facility that serves it. Let s_j and o_j denote the service costs of j in the solutions S and S^* respectively. Let $N_S(s)$ denote the set of clients in C that are served by a facility $s \in S$ in the solution S. Similarly, let $N_{S^*}(o)$ denote the set of clients in C that are served by a facility $o \in S^*$ in the solution S^*. For a subset $A \subseteq S$ and $B \subseteq S^*$, let $N_S(A) = \bigcup_{s\in A} N_S(s)$ and $N_{S^*}(B) = \bigcup_{o\in B} N_{S^*}(o)$.

3. THE K-MEDIAN PROBLEM

In the k-median problem, we are given an input parameter k, $0 < k \leq |F|$. The problem is to identify a subset $S \subseteq F$ of at most k facilities and to serve the clients in C by the facilities in S such that the total service cost is minimized. Thus, if a client $j \in C$ is served by a facility $\sigma(j) \in S$, then we want to minimize $cost(S) = \sum_{j\in C} c_{\sigma(j)j}$. For a fixed S, serving each client by the nearest facility in S, minimizes this cost.

3.1 Local search with swaps

The only operation permitted in the local search procedure is a *swap*. A swap is effected by closing a facility $s \in S$ and opening a facility $s' \notin S$. We start with an arbitrary set of k facilities and keep improving our solution with such swaps till it is possible to do so. The algorithm is described in Figure 1. The operation op is defined as,

$$\mathsf{op}(S) := S - s + s' \quad \text{for } s \in S \text{ and } s' \notin S$$

This swap will be denoted by $\langle s, s'\rangle$.

3.2 The analysis

We now show that the local search procedure as defined above has a locality gap of 5. From the local optimality of S, we know that any swap $\langle s, o\rangle$ for $s \in S$ and $o \in S^*$,

$$cost(S - s + o) \geq cost(S) \quad \text{for all } s \in S, o \in S^* \qquad (1)$$

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

Consider a facility $o \in S^*$. We partition $N_{S^*}(o)$ into subsets $p_s = N_{S^*}(o) \cap N_S(s)$ for $s \in S$. Consider a 1-1 and onto mapping $\pi : N_{S^*}(o) \to N_{S^*}(o)$ with the following property.

PROPERTY 3.1. *For all $s \in S$ such that, $|p_s| \leq \frac{1}{2}|N_{S^*}(o)|$, we have, $\pi(p_s) \cap p_s = \emptyset$.*

It is easy to see that such a mapping π exists.

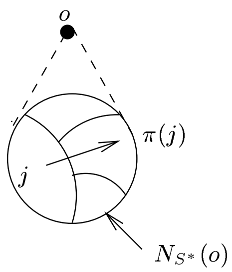

Figure 2: A matching π on $N_{S^*}(o)$

We say that a facility $o \in S^*$ is *captured* by a facility $s \in S$ if s serves more than half the clients served by o, that is, $|N_S(s) \cap N_{S^*}(o)| > \frac{1}{2}|N_{S^*}(o)|$. Note that a facility $o \in S^*$ is captured by at most one $s \in S$. We call a facility $s \in S$, *bad* if it captures some facility in S^* and *good* otherwise.

We now consider k swaps, one for each facility in S^*. If some bad facility $s \in S$ captures exactly one facility $o \in S^*$ then we consider the swap $\langle s, o\rangle$. Suppose l facilities in S (and hence l facilities in S^*) are not considered in such swaps. These l facilities in S are either good or bad, and the bad facilities capture at least two facilities in S^*. Hence there are at least $l/2$ good facilities in S. Now, consider l swaps in which the remaining l facilities in S^* get swapped with the good facilities in S such that each good facility is swapped-out at most twice.

It is easy to verify that the swaps considered above satisfy the following properties.

1. Each $o \in S^*$ is swapped-in exactly once.
2. Each $s \in S$ is swapped-out at most twice. This is because a facility in S that captures more than one

facility in S^* is never swapped-out and a facility that captures exactly one facility in S^* is swapped only with the facility that it captures.

3. If a swap $\langle s, o\rangle$ is considered, the facility s does not capture any facility $o' \neq o$.

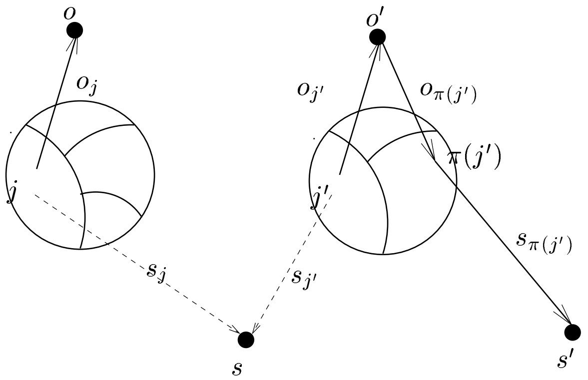

Figure 3: Reassigning the clients in $N_S(s) \cup N_{S^*}(o)$.

We now analyze these swaps by considering an arbitrary swap $\langle s, o\rangle$. We place an upper bound on the increase in cost due to this swap by reassigning the clients in $N_S(s) \cup N_{S^*}(o)$ to the facilities in $S - s + o$ as follows. Refer to Figure 3. The clients $j \in N_{S^*}(o)$ are now assigned to o. Consider a client $j' \in N_S(s) \cap N_{S^*}(o')$, for $o' \neq o$. As s does not capture o', we have $|N_S(s) \cap N_{S^*}(o')| \leq \frac{1}{2}|N_{S^*}(o')|$ and hence by the property 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 $c_{j's'}$. Also from triangle inequality, $c_{j's'} \leq c_{j'o} + c_{o\pi(j')} + c_{\pi(j')s'} = o_{j'} + o_{\pi(j')} + s_{\pi(j')}$. The remaining clients continue to be assigned to the old facilities. From inequality (1) we have,

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

Therefore,

$$\sum_{j \in N_{S^*}(o)} (o_j - s_j) + \sum_{\substack{j \in N_S(s),\\ j \notin N_{S^*}(o)}} (o_j + o_{\pi(j)} + s_{\pi(j)} - s_j) \geq 0 \qquad (2)$$

As each facility $o \in S^*$ is swapped-in exactly once, the first term of the inequality (2) added over all the k swaps gives exactly, $cost(S^*) - cost(S)$. For the second term, we use the fact that each s is swapped-out at most twice. Also for any $j \in C$, as s_j is the shortest distance from j to a facility in S, we get, using triangle inequality, $o_j + o_{\pi(j)} + s_{\pi(j)} \geq s_j$. Thus the second term of the inequality (2) added over all the k swaps is not greater than $2\sum_{j\in C}(o_j + o_{\pi(j)} + s_{\pi(j)} - s_j)$. But as π is 1-1 and onto mapping, $\sum_{j\in C} o_j = \sum_{j\in C} o_{\pi(j)} = cost(S^*)$ and $\sum_{j\in C}(s_{\pi(j)} - s_j) = 0$. Thus, $2\sum_{j\in C}(o_j + o_{\pi(j)} + s_{\pi(j)} - s_j) = 4 \cdot cost(S^*)$. Combining the two terms we get, $cost(S^*) - cost(S) + 4 \cdot cost(S^*) \geq 0$. Thus we have the following theorem.

THEOREM 3.1. *A local search procedure for the metric k-median problem with operations defined as,* $\mathtt{op}(S) := S - s + s'$ *for* $s \in S$ *and* $s' \notin S$*, has a locality gap 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 multi-swaps

In this section, we generalize the algorithm in Section 3 to consider multi-swaps in which up to p facilities could be swapped simultaneously. The operation $\mathtt{op}$ is now defined as,

$$\mathtt{op}(S) := (S \setminus A) \cup B \quad \text{for } A \subseteq S \text{ and } B \subseteq F \setminus S \text{ such that } |A| = |B| \leq p$$

This swap will be denoted by $\langle A, B\rangle$, and 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,

$$capture(A) = \{o \in S^* : |N_S(A) \cap N_{S^*}(o)| > |N_{S^*}(o)|/2\}$$

It is easy to observe the following properties.

CLAIM 3.1. *1. If $X, Y \subseteq S$ are disjoint then $capture(X)$ and $capture(Y)$ are disjoint.*

2. If $X \subset Y$ then $capture(X) \subseteq capture(Y)$.

We now partition S into sets $A_1, A_2, \ldots, A_r$ and S^* into sets $B_1, B_2, \ldots, B_r$ such that for all $i, 1 \leq i \leq r-1$, $|A_i| = |B_i|$ and $B_i = capture(A_i)$. As before, we call a facility in S *bad* if it captures at least one facility in S^*, and *good* otherwise. Our partition of S would have the property that every $A_i, 1 \leq i \leq r-1$ would have exactly one bad facility; thus $r - 1$ equals the number of bad facilities. The set A_r contains only good facilities and it follows from the preceding discussion that $|A_r| = |B_r|$. Our procedure to define these partitions is described in Figure 4.

CLAIM 3.2. *The procedure defined in Figure 4 terminates with a partition of S, S^*, satisfying the properties listed above.*

PROOF. Note that at the start of each iteration of the for-loop, $|S| = |S^*|$. In each iteration we remove exactly one bad facility from S and hence in step 1 we will always be able to find a bad facility. Note that at step 3.1 our procedure maintains the invariant that $|A_i| < |B_i|$. Since $|S| = |S^*|$ and each bad facility in $S \setminus A_i$ captures at least one facility in $S^* \setminus B_i$, it follows that there is a good facility in $S \setminus A_i$. This same argument ensures that the while-loop terminates. $\square$

```
procedure Partition;
    for i = 1 to r − 1 do
        {iteration i }
        1. A_i ← {b} where b ∈ S be any bad facility;
        2. B_i ← capture(A_i);
        3. while |A_i| ≠ |B_i| do
            3.1. A_i ← A_i ∪ {g} where g ∈ S \ A_i be any good facility;
            3.2. B_i ← capture(A_i);
        4. S ← S \ A_i;
           S* ← S* \ B_i;
    A_r ← S;
    B_r ← S*;
end.
```

Figure 4: A procedure to define the partitions

Now we define the swaps as follows. If for some i, we have, $|A_i| = |B_i| \leq p$ then we consider the swap $\langle A_i, B_i\rangle$. From the local optimality of S we have the following inequality.

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

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

If on the other hand, for some i, we have, $|A_i| = |B_i| = q > p$, we swap each facility $o \in B_i$ with each of the $q-1$ *good* facilities $s \in A_i$. 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 q good facilities in A_r. For each such swap $\langle s, o\rangle$, we have,

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

We add such $q(q-1)$ inequalities and multiply them by a factor $1/(q-1)$. Thus, each good facility in A_i is considered in at most $q/(q-1) \leq (p+1)/p$ swaps.

For each facility $o \in S^*$, $N_{S^*}(o)$ is partitioned as follows.

1. Let i, $1 \leq i \leq r$, be such that $|A_i| \leq p$, so that the swap $\langle A_i, B_i\rangle$ was considered above. We consider the part, $p_{A_i} = N_S(A_i) \cap N_{S^*}(o)$.
2. Let i, $1 \leq i \leq r$, be such that $|A_i| > p$. We consider the parts $p_s = N_S(s) \cap N_{S^*}(o)$ for each $s \in A_i$.

Now, for each facility $o \in S^*$, we consider a 1-1 and onto mapping $\pi : N_{S^*}(o) \to N_{S^*}(o)$ with the following property.

Property 3.2. *For all parts $p = p_{A_i}$ or p_s defined above, such that $|p| \leq \frac{1}{2}|N_{S^*}(o)|$, we have, $\pi(p) \cap p = \emptyset$.*

As this condition is imposed only on the parts that have at most half the number of clients in $N_{S^*}(o)$, such a mapping π exists. While doing a swap $\langle A_i, B_i\rangle$ (resp. $\langle s, o'\rangle$), we would be able to reassign clients $j \in N_S(A_i) \cap N_{S^*}(o)$ (resp. $N_S(s) \cap N_{S^*}(o)$) to the facility $s' \notin A_i$ (resp. $s' \neq s$) that serves $\pi(j)$ in S.

The swaps defined above together satisfy the following properties:

1. Each facility in S^* is swapped-in to extent exactly 1.
2. Each facility in S is swapped-out to extent at most $(p+1)/p$.
3. If a swap $\langle A, B\rangle$ is considered, $capture(A) \subseteq B$.

Recall that in the single swap analysis, as each facility in S was getting swapped-out at most twice, we got a $(1+2\times 2)$ approximation result. Here $(p+1)/p$ replaces 2 and the same argument gives a $(1 + 2 \times (p+1)/p) = 3 + 2/p$ approximation result.

3.5 Tight example

In Figure 5, we show an instance where a locally optimum solution, with respect to the p-swap heuristic, has cost $(3+2/p)$ times the cost of the global optimum. The locally optimum solution is given by $\{s_1, s_2, \ldots, s_k\}$, and the optimum solution is given by $\{o_1, o_2, \ldots, o_k\}$. It is easy to verify that we can not decrease the cost by performing any p-swaps. The cost of our solution is $3k - 2 - \frac{k-1}{p+1}$, and the cost of the optimal solution is $k - \frac{k-1}{p+1}$. This ratio approaches $(3 + 2/p)$ as k tends to infinity. Hence our analysis of the locality gap is tight.

4. UNCAPACITATED FACILITY LOCATION

In facility location problems, we are given costs $f_i \geq 0$ for opening the facilities $i \in F$. The uncapacitated facility location problem is to identify a subset $S \subseteq F$ and to serve the clients in C by the facilities in S such that the total facility cost plus the total service cost is minimized. That is, if a client $j \in C$ is assigned to a facility $\sigma(j) \in S$ then we want to minimize $cost(S) = \sum_{i\in S} f_i + \sum_{j\in C} c_{\sigma(j)j}$. As in k-median, 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 uncapacitated facility location problem with a locality

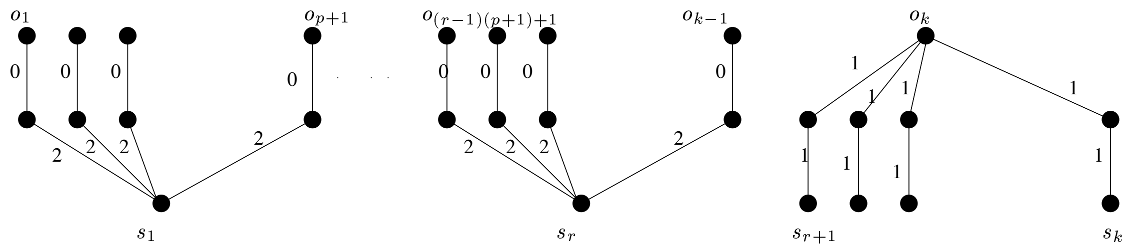

Figure 5: Tight example for the p-swaps

gap of exactly 3. The operation op is now defined as,

$$\mathsf{op}(S) := \begin{cases} S + s', & \text{for } s' \notin S; \\ S - s, & \text{for } s \in S; \\ S - s + s', & \text{for } s' \notin S \text{ and } s \in S \end{cases} \quad (3)$$

These operations correspond to adding, dropping and swapping the facilities. Again, as there are polynomially many ops to be checked for admissibility, the algorithm terminates in polynomial time.

Charikar and Guha [1] 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 et.al. [8] considered the operations of adding, deleting and swapping a facility but could only prove a locality gap of 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'.

LEMMA 4.1 (SERVICE COST).

$$cost_s(S) \ \leq \ cost_f(S^*) + cost_s(S^*)$$

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

Now, we analyze the facility cost $cost_f(S)$. As before, we assume that π is a 1-1 and onto mapping satisfying the property 3.1. In addition, we assume that if $|N_S(s) \cap N_{S^*}(o)| > \frac{1}{2}|N_{S^*}(o)|$ then for all $j \in N_S(s) \cap N_{S^*}(o)$ for which $\pi(j) \in N_S(s)$, we have that $\pi(\pi(j)) = j$. It is easy to see that such a mapping exists. Recall that a facility $s \in S$ is called good if s does not capture any o, that is, for all $o \in S^*$, $|N_S(s) \cap N_{S^*}(o)| \leq \frac{1}{2}|N_{S^*}(o)|$. The facility cost of good facilities can be bounded easily as follows. 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 $o \in S^*$, we have that $s' \neq s$. If we assign j to s', we get, for a good facility $s \in S$,

$$-f_s + \sum_{j \in N_S(s)} (o_j + o_{\pi(j)} + s_{\pi(j)} - s_j) \ \geq \ 0 \quad (4)$$

For bounding the facility cost of a bad facility $s \in S$ we proceed as follows. Suppose a bad facility s captures the facilities $O \subseteq S^*$. Let $o \in O$ 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 $\pi(j) \in N_S(s')$ where $s' \neq s$. Then, j is assigned to s'. Let $j \in N_{S^*}(o')$. We have, $c_{js'} \leq c_{jo'} + c_{o'\pi(j)} + c_{\pi(j)s'} = o_j + o_{\pi(j)} + s_{\pi(j)}$.

2. Suppose $\pi(j) \in N_S(s)$. Let $j \in N_{S^*}(o')$. Then, by the property of the mapping π, the facility s captures the facility o' and hence $o' \in O$. The client j is now assigned to the facility o. From triangle inequality, $c_{jo} \leq c_{js} + c_{so}$. Since o is nearer to s than o' is, $c_{so} \leq c_{so'} \leq c_{js} + c_{jo'}$. Therefore, $c_{jo} \leq c_{js} + c_{js} + c_{jo'} = s_j + s_j + o_j$.

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

$$f_o - f_s + \sum_{\substack{j \in N_{S^*}(o), \\ \pi(j) \in N_S(s)}} (o_j - s_j) + \sum_{\substack{j \notin N_{S^*}(o), \\ \pi(j) \in N_S(s)}} (s_j + s_j + o_j - s_j) + \sum_{\pi(j) \notin N_S(s)} (o_j + o_{\pi(j)} + s_{\pi(j)} - s_j) \ \geq \ 0 \quad (5)$$

Now consider an operation in which a facility $o' \in O - o$ is added. The clients $j \in N_{S^*}(o')$ for which $\pi(j) \in N_S(s)$, are now assigned to the facility o' and this yields the following inequality.

$$f_{o'} + \sum_{\substack{\pi(j) \in N_S(s), \\ j \in N_{S^*}(o')}} (o_j - s_j) \ \geq \ 0 \qquad \text{for each } o' \in O - o \quad (6)$$

Adding inequality (5) with inequalities (6) one for each $o' \in O - o$, we get, for a bad facility $s \in S$,

$$\sum_{o' \in O} f_{o'} - f_s + 2 \sum_{\substack{j \in N_S(s), \\ \pi(j) \in N_S(s)}} o_j$$
$$+ \sum_{\substack{j \in N_S(s), \\ \pi(j) \notin N_S(s)}} (o_j + o_{\pi(j)} + s_{\pi(j)} - s_j) \geq 0 \quad (7)$$

Now, if we add the inequalities (4) for all good facilities $s \in S$ together with the inequalities (7) for all bad facilities s, we get, $cost_f(S^*) - cost_f(S) + 2 \cdot cost_s(S^*) \geq 0$. This proves the following lemma.

LEMMA 4.2 (FACILITY COST).

$$cost_f(S) \leq cost_f(S^*) + 2 \cdot cost_s(S^*)$$

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

THEOREM 4.3. *The local search procedure for the metric uncapacitated facility location problem where operation* op *is defined as*

$$\texttt{op}(S) := \begin{cases} S + s', & \text{for } s' \notin S; \\ S - s, & \text{for } s \in S; \\ S - s + s', & \text{for } s' \notin S \text{ and } s \in S, \end{cases}$$

has a locality gap 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. Using standard scaling techniques [1] our algorithm can be improved to achieve a $1 + \sqrt{2} \approx 2.414$ approximation.

4.3 Tight example

In Figure 6, we show an instance where a local optimum has cost 3 times the cost of the global optimum. The locally optimum solution consists of a single facility s. The optimum solution consists of $\{o_0, o_1, \ldots, o_k\}$. Clearly, we cannot delete the facility s. It is easy to verify that we can not decrease the cost of our solution by either adding any facility from the optimum, or by any swap which involves bringing in a facility from the optimum and deleting s. The cost of current solution is $3k+1$, while the cost of the optimum solution is $k+1$. Hence our analysis of the algorithm is tight.

5. THE CAPACITATED FACILITY LOCATION PROBLEM

In the capacitated facility location problem, along with the facility costs $f_i \geq 0$, we are given 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 multi-set S of facilities and to serve the clients in C by the facilities in S such that the capacity constraints are satisfied and the total facility cost plus the total service cost is minimized. If a client $j \in C$ is assigned to a facility $\sigma(j) \in S$ then we want to minimize $cost(S) = \sum_{i \in S} f_i + \sum_{j \in C} c_{\sigma(j)j}$. Now, for a fixed S, in order to minimize the service cost, we solve a mincost flow problem. The clients $j \in C$ send unit amount of flow to the facilities in S such that the capacity constraints are satisfied. Such a mincost flow can be computed efficiently.

In the remainder of this section we let S and S^* be the multi-sets of the facilities opened in the output and optimum solutions 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 capacitated facility location problem described above. The operation op is now defined as follows.

$$\texttt{op}(S) := \begin{cases} S + s', & \text{for } s' \in F; \\ S - T + l \cdot s', & \text{for } s' \in F,\ T \subseteq S \\ & \text{and } l \in \mathbf{Z}^+ \end{cases} \quad (8)$$

The operation $S - T + l \cdot s'$ stands for dropping the facilities in T and opening l new copies of s' where l is sufficiently large so that the clients $j \in N_S(T)$ can be served by these new copies of s', that is, $l \cdot u_{s'} \geq |N_S(T)|$. As in the case of the uncapacitated facility location, we restrict this operation so that all clients in $N_S(T)$ are served by the facility s'. The cost of the new solution is now given by

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

Given a facility $s' \in F$, we use the Procedure T-Hunt described in Figure 7 to find a subset $T \subseteq S$ of facilities. Here $m = |C|$ is the upper bound on the number of new copies of s' that we need to open. Dropping a facility $s \in T$ gives an extra $|N_S(s)|$ clients to be served by the new facility s'. A client $j \in N_S(s)$ where $s \in T$ now travels a extra distance of at most $(c_{s'j} - c_{sj})$. Thus, dropping a facility $s \in T$ gives a *saving* of $f_s - \sum_{j \in N_S(s)} (c_{s'j} - c_{sj})$. Due to the 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 $weight(s) = |N_S(s)|$ and $profit(s) = f_s - \sum_{j \in N_S(s)} (c_{s'j} - c_{sj})$. The oracle Knapsack$(W)$ returns a multi-set $T \subseteq S$ such that $\sum_{s \in T} weight(s) \leq W$ and $profit(T) = \sum_{s \in T} profit(s)$ is maximized.

It is interesting to note that since we are permitting any subset of facilities, T, from our current solution, S, to be dropped, the number of operations are 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 admissible operations. It might be case that T-hunt is not able to identify any admissible operations, while there are operations, as defined by op, which are admissible. However, our analysis will work only with the assumption that T-hunt could not find admissible operations.

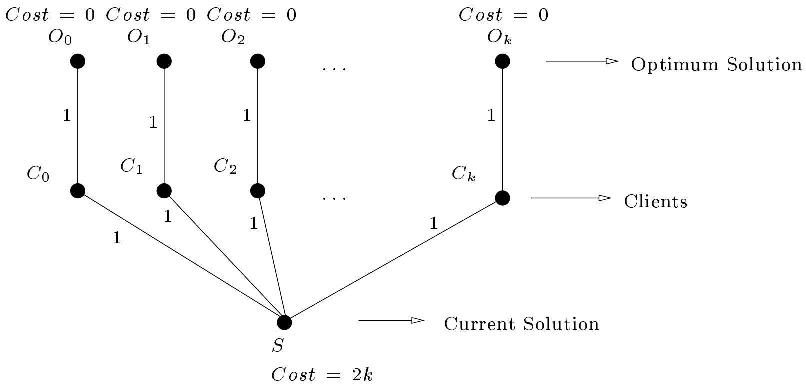

Figure 6: Tight example for uncapacitated facility location algorithm.

Procedure `T-Hunt`.

1. For $l = 1$ to m do,
2. $\quad T \leftarrow$ `Knapsack`$(l \cdot u_{s'})$.
3. $\quad$ If $cost(S) + l \cdot f_{s'} - profit(T) \leq (1 - \frac{\epsilon}{p(n,m)})cost(S)$, then return T.
4. return "no admissible swap".

Figure 7: A procedure to find a subset $T \subseteq S$ of facilities

5.2 The analysis

LEMMA 5.1. *For any $T \subseteq S$ and any $s' \in F$, we have,*

$$\lceil |N_S(T)|/u_{s'} \rceil \cdot f_{s'} + \sum_{s \in T} |N_S(s)| \cdot c_{ss'} \;\geq\; \sum_{s \in T} f_s$$

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 "no admissible swap". Hence,

$$\begin{aligned} & l \cdot f_{s'} - profit(T) \\ = \;\; & l \cdot f_{s'} - \sum_{s \in T} \left(f_s - \sum_{j \in N_S(s)} (c_{s'j} - c_{sj}) \right) \;<\; 0 \end{aligned}$$

But, for a client $j \in N_S(s)$, we have, $c_{s'j} - c_{sj} \geq c_{ss'}$. Therefore we have the lemma. $\square$

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

LEMMA 5.2 (SERVICE COST).

$$cost_s(S) \;\leq\; cost_f(S^*) + cost_s(S^*)$$

Now, we bound the facility cost of S. Consider a directed graph $G = (V, E)$ with lengths on edges, where,

$$V = \{v_s \mid s \in S\} \bigcup \{w_o \mid o \in S^*\} \bigcup \{sink\},$$

$$E = \{(v_s, w_o) \mid s \in S, o \in S^*\} \bigcup \{(w_o, sink) \mid o \in S^*\}$$

The lengths of (v_s, w_o) and $(w_o, sink)$ are c_{so} and f_o/u_o respectively. The cost of routing unit amount of 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 $sink$.

LEMMA 5.3. *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 $cost_s(S) + cost_s(S^*) + cost_f(S^*)$.*

PROOF. Consider the clients $j \in C$. If $j \in N_S(s) \cap N_{S^*}(o)$ then route one unit of flow along the path $v_s \rightarrow w_o \rightarrow sink$. Triangle inequality implies, $c_{so} \leq s_j + o_j$. Also, for a facility $o \in S^*$, the routing cost on the edge $(w_o, sink)$ is $|N_{S^*}(o)| \cdot f_o/u_o \leq \lceil |N_{S^*}(o)|/u_o \rceil \cdot f_o$, which in turn is the contribution of o to $cost_f(S^*)$. Thus, the routing cost of this flow is at most $cost_s(S) + cost_s(S^*) + cost_f(S^*)$. $\square$

In the flow with the minimum routing cost, for each v_s, the flow of $|N_S(s)|$ units is routed along the shortest path from v_s to the sink. That is, along $v_s \rightarrow w_o \rightarrow sink$, where o is such that $c_{so} + f_o/u_o$ is minimized, ties being broken arbitrarily. For each $o \in S^*$, let $T_o \subseteq S$ denote the set of facilities s that route their flow via w_o. As this gives a minimum cost flow, from Lemma 5.3, we have,

$$\begin{aligned} & cost_s(S) + cost_s(S^*) + cost_f(S^*) \\ \geq \;\; & \sum_{o \in S^*} \sum_{s \in T_o} |N_S(s)|(c_{so} + f_o/u_o) \end{aligned} \qquad (9)$$

Now, applying Lemma 5.1 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 $o \in S^*$, we get,

$$\sum_{o \in S^*} f_o + \sum_{o \in S^*} \sum_{s \in T_o} |N_S(s)|(c_{so} + f_o/u_o) \geq \sum_{o \in S^*} \sum_{s \in T_o} f_s \;=\; cost_f(S) \qquad (10)$$

The inequalities (9) and (10) together imply

$$cost_f(S) \leq 2 \cdot cost_f(S^*) + cost_s(S^*) + cost_s(S)$$

This inequality together with Lemma 5.2 gives the following lemma.

LEMMA 5.4 (FACILITY COST).

$$cost_f(S) \;\leq\; 3 \cdot cost_f(S^*) + 2 \cdot cost_s(S^*)$$

Combining Lemmas 5.2 and 5.4, we obtain the following result.

THEOREM 5.5. *The local search procedure for the metric capacitated facility location problem with the operation* op *defined as in (8) has a locality gap of 4.*

Again using scaling techniques [1] the algorithm can be improved to obtain a $2 + \sqrt{3} \approx 3.732$ approximation. The tight example given in Section 4.3 for the uncapacitated facility location problem shows that a locally optimum solution for this problem can have cost 3 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 uncapacitated facility location problems. Our sharper analysis leads to a $3 + 2/p$-approximation algorithm for the k-median with a running time of $O(n^p)$. For capacitated facility location, 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.

7. ACKNOWLEDGEMENTS

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.

8. REFERENCES

[1] 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*, October 1999.

[2] F. Chudak. Improved approximation algorithms for uncapacitated facility location problem. In *Proceedings of the 6th Conference on Integer Programming and Combinatorial Optimization*, June 1998.

[3] F. Chudak and D. Shmoys. Improved approximation algorithms for capacitated facility location problem. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, January 1999.

[4] 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*, June 1999.

[5] S. Guha and S. Khuller. Greedy strikes back: Improved facility location algorithms. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, January 1998.

[6] K. Jain and V. Vazirani. Primal-dual approximation algorithms for metric facility location and k-median problems. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, October 1999.

[7] S. Lin and B. W. Kernighan. An effective heuristic algorithm for the traveling salesman problem. *Operations Research*, 21, 1973.

[8] M. Korupolu and C. Plaxton and R. Rajaraman. Analysis of a local search heuristic for facility location problems. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, January 1998.

[9] M. Korupolu and C. Plaxton and R. Rajaraman. Analysis of a local search heuristic for facility location problems. Technical Report 98-30, DIMACS, June 1998.

[10] D. Shmoys and E. Tardos and K. Aardal. Approximation algorithms for facility location problems. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, May 1997.

[11] M. Charikar and S. Guha and E. Tardos and D. Shmoys. A constant-factor approximation algorithm for the k-median problem. In *Proceedings of the 31th Annual ACM Symposium on Theory of Computing*, May 1999.