

# Price of Anarchy, Locality Gap, and a Network Service Provider Game

Nikhil R. Devanur, Naveen Garg, Rohit Khandekar,  
Vinayaka Pandit, Amin Saberi, and Vijay V. Vazirani

College of Computing, Georgia Tech, Atlanta, GA.  
Indian Institute of Technology, Delhi.

## 1 Introduction

A crucial difference between the Internet, which is undoubtedly one of the biggest human artifacts built in the last few years, and large networks built in the past, such as the highways and telephone and electric networks, is that whereas the latter were centrally planned, the former has been built essentially in an anarchic manner by a large multitude of agents with varying interests and varying end-goals in mind. It is for this reason that methodology from game theory is most appropriate for analyzing problems and issues arising with the Internet. A very useful concept in this respect has been the *price of anarchy* – attempts to put a bound on the cost incurred by selfish agents in accomplishing a goal in relation to the optimal cost (incurred by a centralized authority). This notion was introduced by Koutsoupias and Papadimitriou [2] and was named so by Papadimitriou [3]. Roughgarden and Tardos [4–6] have used this notion to compare latency incurred by traffic in a network in the two situations: anarchic and centrally planned. The anarchic can be formally captured via the notion of a Nash equilibrium, a situation in which none of the selfish agents has an incentive to switch strategies.

In this paper, we define a network service provider game that is natural in the context of the Internet and we raise the question of upper bounding the price of anarchy for this game. Our answer is obtained in two steps. In the first step, we show that the price of anarchy of this game is precisely the locality gap of the  $k$ -facility location problem under a certain neighborhood structure; the *locality gap* of a problem is the maximum ratio of a local optimal solution and a global optimal solution for this problem. Our second step is to show that the locality gap of the  $k$  facility location problem is at most 5. The latter result is of independent interest. Additionally, the technique introduced in the first step may help reduce the question of upper bounding the price of anarchy for

other games to the question of upper bounding the locality gap for the corresponding optimization question.

Our service provider game is as follows: we are given a network with a distinguished node called the root. The remaining nodes are partitioned into two sets: nodes on which clients reside,  $C$ , and nodes that can be occupied by service providers,  $F$ . The distances between the nodes in the network satisfy the metric property. Assume there are  $k$  service providers. Once a subset of them occupy nodes in  $F$ , each client is served by the closest service provider. A client, say  $j$ , pays to the closest service provider, say  $i$ , the VCG cost, i.e., the cost of connecting to the second closest service provider. This defines the total revenue accrued by  $i$ . On the other hand, the cost incurred by  $i$  is the total connecting cost to each of its clients and the cost to connect from  $i$  to the root node. The difference of the revenue and the cost is the net profit of  $i$ .

Each service provider is allowed three kinds of moves:

- Deletion: A service provider that is occupying a node in  $F$  may decide to not participate in the network (e.g., if its profit is negative).
- Addition: A service provider that is not participating in the network may decide to occupy a vacant node in  $F$ .
- Swap: A service provider may move from one node of  $F$  to another vacant node.

A configuration of service providers is said to be *Nash equilibrium* if none of the service providers can improve their profit using the three moves stated above. The cost of this Nash equilibrium is the total cost incurred by the service providers. The optimal cost is the minimum cost incurred by at most  $k$  service providers in serving all clients. The price of anarchy is supremum, over all Nash equilibria, of the ratio of the cost of a Nash equilibrium and the optimal cost.

The *k-facility location* problem is the following: we are given a set of clients,  $C$ , and a set of facilities,  $F$ . For each facility in  $F$ , we are specified the cost for opening that facility. We are also given a bound  $k$  on the number of facilities. The problem is to determine which facilities to open so as to minimize the sum of opening costs and the costs of connecting each city to its closest facility.

The *k-facility location* problem inherits features of both, the *k*-median and the uncapacitated facility location problems. In the *k* median problem we are permitted to open at most  $k$  facilities but there is no cost associated with opening a facility. A simple local search algorithm which swaps facilities as long as the cost of the solution reduces, has a locality gap of

5 [1]. The uncapacitated facility location problem permits the opening of any number of facilities, but now there is a cost associated with opening a facility. Once again, a local search algorithm which at each step can add, delete or swap facilities has a locality gap of 3 [1]. We extend the ideas in these two results to show that for the  $k$  facility location problem, a local search algorithm which adds, deletes or swaps facilities at each step has a locality gap of 5. Note that we can add a facility only if the number of facilities is less than  $k$ . It is for this reason that the analysis for the uncapacitated facility location problem does not directly apply to this setting. Further, since the  $k$  median problem is a special case of this problem — when facility costs are zero — the bound of 5 on the locality gap is the best possible.

## 2 Formal Setting

### 2.1 The $k$ -facility location problem

In the  $k$ -facility location problem, we are given a set of facility locations  $F$  and a set of clients  $C$ . For a facility  $i \in F$ , its *facility cost*  $f_i \geq 0$  is the cost of *opening* that facility. We are also given the distances  $c_{ij}$  between  $i, j \in F \cup C$  that satisfy metric properties:  $c_{ij} \geq 0$ ,  $c_{ij} = c_{ji}$ ,  $c_{ij} \leq c_{il} + c_{lj}$  for all  $i, j, l \in F \cup C$ . The distance  $c_{ij}$  between  $i \in F$  and  $j \in C$  is the cost of serving the client  $j$  by the facility  $i$ . We are given an upper bound  $k > 0$  on the number of facilities to be opened. The objective is to open at most  $k$  facilities in  $F$  such that the sum of the facility costs of the open facilities and the cost of serving each client by the nearest open facility is minimized.

For a subset  $S \subseteq F$  of at most  $k$  facilities, let  $fac(S) = \sum_{i \in S} f_i$  denote its facility cost, let  $serv(S) = \sum_{j \in C} \min_{i \in S} c_{ij}$  denote its service cost, and let  $cost(S) = fac(S) + serv(S)$  denote its total cost. We define  $cost(\emptyset) = \infty$ . Consider the following local search algorithm for the  $k$ -facility location problem. We start with opening any subset  $S \subseteq F$  of at most  $k$  facilities. We then try to reduce  $cost(S)$  repeatedly by doing the following local search operations. For  $i \in F$ , let us use  $S - i$  and  $S + i$  to denote  $S \setminus \{i\}$  and  $S \cup \{i\}$  respectively.

- *delete* a facility: if there is  $i \in S$  such that  $cost(S - i) < cost(S)$ , then  $S := S - i$ .
- *add* a facility: if  $|S| < k$  and there is  $i \in F \setminus S$  such that  $cost(S + i) < cost(S)$ , then  $S := S + i$ .
- *swap* facilities: if there is  $i \in S$  and  $i' \in F \setminus S$  such that  $cost((S - i) + i') < cost(S)$ , then  $S := (S - i) + i'$ .

We call this a *delete-add-swap* local search algorithm. We call  $S \subseteq F$  with  $|S| \leq k$  a *local optimum* solution if  $cost(S)$  cannot be reduced by doing any of the above operations. The maximum ratio of the cost of a local optimum solution to the cost of the global optimum solution is called the *locality gap* of the local search algorithm. In Section 4, we prove the following theorem.

**Theorem 1.** *The locality gap of the above local search algorithm is at most 5.*

## 2.2 Service provider game

In the service provider game, we are given a network with a distinguished node  $r$  called *root*. The set of remaining nodes in the network is partitioned into a set of clients  $C$  and a set of service locations  $F$ . We are also given the distances  $c_{ij}$  between  $i, j \in F \cup C \cup \{r\}$  that satisfy metric properties. Suppose that there are  $k$  service providers. Each service provider is allowed to occupy at most one service location. Two service providers cannot occupy a single service location. The assignment of service providers to the service locations, say  $S \subseteq F$ , defines a *configuration*. Note that not all providers may get assigned to locations. We abuse the notation and use  $i \in S$  to indicate both the service provider and the service location that it occupies, with the context clarifying the intention.

Consider a configuration  $S \subseteq F$  with  $|S| \leq k$ . For a service provider  $i \in S$ , let  $N_S(i)$  be the neighborhood of  $i$ , i.e., the set of clients for which  $i$  is the closest among the providers in the solution  $S$ . To service all the clients in  $N_S(i)$  and to connect to the root  $r$ , the provider  $i$  incurs a total cost of  $cost_S(i) = c_{ir} + \sum_{j \in N_S(i)} c_{ij}$ . Let the total cost of all the providers in a configuration  $S$  be denoted by  $cost(S) = \sum_{i \in S} cost_S(i)$ . For a client  $j \in C$  and a configuration  $S$ , let  $S_j = \min_{i \in S} c_{ij}$  be the distance of  $j$  to the closest provider in  $S$ .

Each client connects to the service provider closest to it, but is charged the VCG payment, which is the distance to the second closest service provider. The *revenue* of a service provider  $i \in S$  is the total payment it receives from all the clients it serves, i.e.,  $revenues_S(i) = \sum_{j \in N_S(i)} T_j$  where  $T = S - i$ . The profit of a service provider  $i$  is  $profits_S(i) = revenues_S(i) - cost_S(i)$ . Note that both  $revenues_S(i)$  and  $profits_S(i)$  are with respect to a particular configuration  $S$ .

Each service provider behaves selfishly and wishes to maximize his own profit. This defines a game among the service providers: We allow only pure strategies, i.e., a strategy of a service provider can be either to

occupy a particular location, or to occupy none at all. We assume that no two service providers occupy the same location. The strategies of all the service providers defines a configuration. The payoff of a particular service provider is equal to his profit in that configuration. A *Nash equilibrium* is a configuration so that no service provider can unilaterally change his strategy and get a higher profit. The *price of anarchy* is the supremum, over all Nash equilibria, of the ratio between the cost of a Nash equilibrium and the optimum cost.

### 3 Connection between Price of Anarchy and Locality Gap

There is a natural correspondence between an instance of the service provider game and the  $k$ -facility location problem. The set of service locations in the game corresponds to the set of facilities in the  $k$ -facility location problem. The cost  $c_{ir}$  that a provider  $i$  incurs in connecting to the root corresponds to the facility cost  $f_i$ . The total cost of a configuration  $S$  corresponds to the cost of facilities  $S$  in the  $k$ -facility location problem.

In this section, we show that a Nash equilibrium in the service provider game corresponds to a local optimum solution in the  $k$ -facility location instance with respect to the delete-add-swap local search algorithm.

**Lemma 1.** *The profit of a provider  $s$  in the configuration  $S$  is given by  $profit_S(s) = cost(S - s) - cost(S)$ .*

*Proof.* Let  $T = S - s$ .

$$\begin{aligned}
cost(T) - cost(S) &= \left( \sum_{i \in T} f_i + \sum_{j \in C} T_j \right) - \left( \sum_{i \in S} f_i + \sum_{j \in C} S_j \right) \\
&= \sum_{j \in C} (T_j - S_j) - f_s \\
&= \sum_{j \in N_S(s)} T_j - \left( \sum_{j \in N_S(s)} c_{sj} + f_s \right) \\
&= revenue_S(s) - cost_S(s) \\
&= profit_S(s).
\end{aligned}$$

**Theorem 2.** *A configuration  $S \subseteq F$  is a Nash equilibrium of an instance of the service provider game if and only if  $S$  is a local optimum solution of the corresponding instance of the  $k$ -facility location problem with respect to the delete-add-swap local search.*

*Proof.* Let  $S \subseteq F$  be a Nash equilibrium. From the definition,  $profits_S(s) = cost(S - s) - cost(S) \geq 0$  for all  $s \in S$ . Therefore  $cost(S)$  cannot be reduced by deleting a facility  $s \in S$ . Let  $S' = S + s$  for some  $s \notin S$ . Since any provider not in  $S$  did not occupy the location  $s$ , we have  $profits_{S'}(s) \leq 0$ . Therefore, from Lemma 1, we have  $cost(S' - s) - cost(S') \leq 0$ . Thus  $cost(S) \leq cost(S + s)$ . Therefore  $cost(S)$  cannot be reduced by adding a facility  $s \notin S$ . Now let  $S' = S - s + s'$  for some  $s \in S$  and  $s' \notin S$ . Since the provider  $s$  does not move from location  $s$  to  $s'$ , we have  $profits_{S'}(s') \leq profits_S(s)$ . Let  $T = S - s = S' - s'$ . Then we have

$$\begin{aligned} cost(S') - cost(S) &= (cost(T) - cost(S)) - (cost(T) - cost(S')) \\ &= profits_S(s) - profits_{S'}(s') \\ &\geq 0. \end{aligned}$$

Therefore  $cost(S)$  cannot be reduced by swapping a pair of facilities. Thus the solution  $S \subseteq F$  is indeed a local optimum solution with respect to the delete-add-swap local search.

Similarly, we can show that if  $S$  is a local optimum solution, then it is a Nash equilibrium in the service provider game.

**Theorem 3.** *The price of anarchy for the service provider game is at most 5.*

*Proof.* The proof follows from Theorems 1 and 2.

## 4 Proof of Theorem 1

### 4.1 Notation and preliminaries

Let  $S$  and  $O$  denote a local optimum and a global optimum solutions respectively. In this section, we use  $s$  to denote a facility in  $S$  and  $o$  to denote a facility in  $O$ . We use the notation introduced in the service provider game. For a client  $j \in C$ , let  $S_j = \min_{s \in S} c_{sj}$  and  $O_j = \min_{o \in O} c_{oj}$  denote its service costs in  $S$  and  $O$  respectively. For a facility  $s \in S$ , let  $N_S(s)$  denote the set of clients served by  $s$  in  $S$  and for a facility  $o \in O$ , let  $N_O(o)$  denote the set of clients served by  $o$  in  $O$ . Let  $N_s^o = N_S(s) \cap N_O(o)$ . A facility  $s \in S$  is said to “capture” a facility  $o \in O$  if  $|N_s^o| > |N_O(o)|/2$ . A facility  $s \in S$  is called “good” if it does not capture any facility in  $O$ . It is called “1-bad”, if it captures exactly one facility in  $O$ . It is called “2+bad” if it captures at least 2 facilities in  $O$ . These notions were introduced by Arya et al. [1].

We now consider a one-to-one onto function  $\pi : C \rightarrow C$  such that the following properties are satisfied.

1. for each facility  $o \in O$ , we have  $\pi(N_O(o)) = N_O(o)$ ,
2. if  $s \in S$  does not capture  $o \in O$ , then  $\pi(N_s^o) \cap N_s^o = \emptyset$ ,
3. if  $s \in S$  captures  $o \in O$  and if  $j \in N_s^o$  is such that  $\pi(j) \in N_s^o$ , then  $\pi(j) = j$ ,
4. if  $s \in S$  captures  $o \in O$ , then for all  $s' \in S$  such that  $s' \neq s$ , we have  $\pi(N_{s'}^o) \subset N_s^o$ .

The first three properties were used in the analysis of the uncapacitated facility location problem by Arya et al. [1]. We need the additional fourth property. The function  $\pi$  can be obtained in a manner similar to that in Arya et al. [1].

## 4.2 Deletes, Adds, and Swaps considered

Since  $S$  is a local optimum solution, its cost cannot be reduced by doing any of the three operations. For any  $s \in S$ , we have  $\text{cost}(S - s) \geq \text{cost}(S)$ . If  $|S| < k$ , then for any  $o \in O$ , we have  $\text{cost}(S + o) \geq \text{cost}(S)$ . For any  $s \in S$  and  $o \in O$ , we have  $\text{cost}(S - s + o) \geq \text{cost}(S)$ . We now carefully consider some delete, add, and swap operations. If we delete  $s \in S$ , we reroute the clients in  $N_S(s)$  to other facilities in  $S - s$ . If we add  $o \in O$ , we reroute the clients in  $N_O(o)$ . If we swap  $s \in S$  and  $o \in O$ , we reroute the clients in  $N_S(s) \cup N_O(o)$  to other facilities in  $S - s + o$ . The assignment of the other clients is not changed. For each of the operations considered, we obtain an upper bound on  $\text{cost}(S') - \text{cost}(S) \geq 0$  where  $S'$  is the solution obtained after the operation. We then add these inequalities to prove Theorem 1.

We consider the following operations.

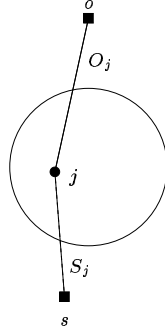
1. Each 1-bad facility  $s \in S$  is swapped with the facility  $o \in O$  that it captures. The clients  $j \in N_O(o)$  are rerouted to  $o$ . The clients  $j \in N_S(s) \setminus N_O(o)$  are rerouted to  $s' \in S$  that serves  $\pi(j)$  in  $S$ . Since  $s$  does not capture any  $o' \neq o$ , from property 2 of the function  $\pi$ , we have  $s' \neq s$  and the rerouting is feasible. We call these swaps as “Type 1” operations.
2. Each 2+bad facility  $s \in S$  is swapped with the nearest facility  $o \in O$  that it captures. All the clients in  $N_O(o)$  are rerouted to  $o$ . Consider a facility  $o' \neq o$  captured by  $s$ . Such a facility  $o'$  is called a *far* facility. The clients  $j \in N_{s'}^{o'}$  such that  $\pi(j) = j$  are rerouted to  $o$ . The remaining clients  $j \in N_S(s)$  are rerouted to  $s' \in S$  that serves  $\pi(j)$  in  $S$ . From properties 2 and 3 of  $\pi$ , such rerouting is feasible. We call these swaps as “Type 2” operations.

3. The facilities in  $S$  that are not considered in Type 1 or Type 2 swaps are good facilities. Let  $l_S$  and  $l_O$  be the number of facilities in  $S$  and  $O$  respectively that are not considered in Type 1 or Type 2 swaps. We consider some operations with these facilities. Let  $l = \min\{l_S, l_O\}$ . Any  $l$  of the remaining facilities  $o \in O$  are swapped with any  $l$  of the good facilities  $s \in S$ . All the clients in  $N_O(o)$  are rerouted to  $o$ . The clients in  $N_S(s) \setminus N_O(o)$  are rerouted to  $s' \in S$  that serves  $\pi(j)$  in  $S$ . From property 2 of  $\pi$ , we have  $s' \neq s$ . In the following cases, we consider some more operations.
  - Case  $l_S < l_O$ . Note that  $|S| < k$ . The remaining  $l_O - l$  facilities  $o \in O$  are added one by one. The clients in  $N_O(o)$  are rerouted to  $o$ .
  - Case  $l_S > l_O$ . Note that  $|S| > 1$ . The remaining  $l_S - l$  facilities  $s \in S$  are deleted one by one. The clients in  $N_S(s)$  are rerouted to  $s' \in S$  that serves  $\pi(j)$  in  $S$ . From property 2 of  $\pi$ , we have  $s' \neq s$ .
 We call these operations as “Type 3” operations.
4. Let  $n_S$  be the number of good facilities in  $S$  and let  $n_O$  be the number of far facilities in  $O$ . Let  $n = \min\{n_S, n_O\}$ . Any  $n$  of the far facilities  $o \in O$  are swapped with any  $n$  of the good facilities  $s \in S$ . All the clients in  $N_O(o)$  are rerouted to  $o$ . The clients in  $N_S(s) \setminus N_O(o)$  are rerouted to  $s' \in S$  that serves  $\pi(j)$  in  $S$ . From property 2 of  $\pi$ , we have  $s' \neq s$ . In the following cases, we consider some more operations. If  $n_S < n_O$  then the remaining  $n_O - n$  far facilities  $o \in O$  are added one by one. The clients in  $N_O(o)$  are rerouted to  $o$ . We call these operations as “Type 4” operations.

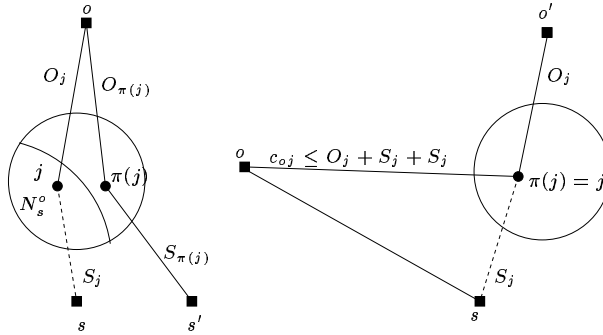
In the sections to follow, we estimate the change in the service cost of the clients and the total facility cost due to all the 4 types of operations together.

**Counting the change in the service cost** In this section, we upper bound the change in the service cost of the clients due to the operations considered using the mentioned rerouting. We partition the clients in two classes as follows.

- Consider a client  $j \in C$  such that  $\pi(j) = j$  and  $j \in N_O(o')$  where  $o'$  is a far facility. We call such clients *white*. A white client is not rerouted in Type 1 operations. In Type 2 operations, this client is rerouted to the facility  $o$  captured by  $s$  (right side of Figure 2). Note that  $s$  also captures  $o'$ . The change in the service cost of  $j$  is at most  $c_{jo} - c_{js}$ . Since  $c_{jo} \leq c_{js} + c_{so} \leq c_{js} + c_{so'} \leq c_{js} + c_{js} + s_{jo'}$ , the change is at most



**Fig. 1.** Change in service cost of  $j \in N_O(o)$  when  $o$  is brought in



**Fig. 2.** Change in service cost of  $j \in N_s^o$  when  $s$  is taken out and  $o$  is not brought in

$c_{js} + c_{jo'} = S_j + O_j$ . In Type 3 and 4 operations, the change in the service cost of  $j$  is  $O_j - S_j$  each (Figure 1). Thus the sum of the changes in all types of operations is at most  $(O_j + S_j) + 2(O_j - S_j) = 3O_j - S_j$ . Thus the change over all white clients is at most

$$\sum_{j:\text{white}} (3O_j - S_j). \quad (1)$$

- The remaining clients are called *colored*. We first estimate the change in service cost of these clients due to operations of Type 1, 2, and 3. In Type 1, 2, and 3 operations, each colored client is rerouted in exactly two operations: once when the facility  $o$  serving it in  $O$  is brought in, and next when the facility  $s$  serving it in  $S$  is taken out. In the first case,  $j$  is rerouted to  $o$  and the change in its service cost is  $O_j - S_j$  (Figure 1). In the second case,  $j$  is rerouted to  $s'$

that serves  $\pi(j)$  in  $S$  and the change in the service cost is at most  $c_{js'} - c_{js} \leq c_{jo} + c_{o\pi(j)} + c_{\pi(j)s'} - c_{js} = O_j + O_{\pi(j)} + S_{\pi(j)} - S_j$  (left side of Figure 2). If we sum these changes over all colored clients, we get

$$\sum_{j:\text{colored}} (O_j - S_j + O_j + O_{\pi(j)} + S_{\pi(j)} - S_j) = \sum_{j:\text{colored}} (3O_j - S_j). \quad (2)$$

The above equality holds since

$$\sum_{j:\text{colored}} O_j = \sum_{j:\text{colored}} O_{\pi(j)} \text{ and } \sum_{j:\text{colored}} S_j = \sum_{j:\text{colored}} S_{\pi(j)}.$$

Now consider a Type 4 operation in which a far facility  $o'$  is brought in and a good facility  $s$  is taken out or a far facility  $o'$  is added. Let  $s'$  be the 2+bad facility that captures  $o'$ . A colored client  $j \in N_{s'}^{o'}$  is rerouted to  $o'$ , thus the change in its service cost is  $O_j - S_j$  (Figure 1). We call such clients as *red*. Consider a colored client  $j \in N_{s''}^{o'}$  served by a good facility  $s'' \neq s'$  in  $S$ . We call such a client *blue*. A blue client is rerouted to  $o'$ , thus the change in its service cost in this operation is  $O_j - S_j$  (Figure 1). However, a blue client gets rerouted if  $s''$  is taken out with some far facility  $o''$ . In this case, it contributes at most  $O_j + O_{\pi(j)} + S_{\pi(j)} - S_j$  to the change in the service cost (left side of Figure 2). Note that for any client  $j \in C$ , from triangle inequality we have  $O_j + O_{\pi(j)} + S_{\pi(j)} - S_j \geq 0$ . The overall change in the service cost of colored clients due to the Type 4 operations is at most

$$\sum_{j:\text{red}} (O_j - S_j) + \sum_{j:\text{blue}} (O_j - S_j) + \sum_{j:\text{blue}} (O_j + O_{\pi(j)} + S_{\pi(j)} - S_j). \quad (3)$$

Now from the property 4 of the mapping  $\pi$ , a red client is mapped to a blue client and vice versa. Thus

$$\sum_{j:\text{red}} O_j = \sum_{j:\text{blue}} O_{\pi(j)} \text{ and } \sum_{j:\text{red}} S_j = \sum_{j:\text{blue}} S_{\pi(j)}.$$

Substituting this in (3), we get that the change in the service cost of colored clients due to the Type 4 operations is at most

$$\sum_{j:\text{red}} 2O_j + \sum_{j:\text{blue}} (2O_j - 2S_j) \leq \sum_{j:\text{colored}} 2O_j. \quad (4)$$

Adding (2) and (4), the overall change in the service cost of the colored clients dues to all 4 types of operations is at most

$$\sum_{j:\text{colored}} (5O_j - S_j). \quad (5)$$

**Counting the change in the facility cost** In Type 1, 2, and 3 operations, each facility in  $O$  is brought in exactly once and each facility in  $S$  is taken out exactly once. Thus in these operations the change in the facility cost is exactly  $fac(O) - fac(S)$ . In Type 4 operations, each far facility is brought in exactly once and some good facilities are taken out. Thus the change in facility cost in these operations is at most  $\sum_{o:\text{far}} f_o \leq fac(O)$ . Thus the overall change in the facility cost is at most

$$2 \cdot fac(O) - fac(S). \tag{6}$$

**Combining the changes in the service and facility costs** Since the sum of the changes in the total cost in all the operations is at least zero, putting together (1), (5), and (6) we have,

$$\sum_{j:\text{white}} (3O_j - S_j) + \sum_{j:\text{colored}} (5O_j - S_j) + 2 \cdot fac(O) - fac(S) \geq 0.$$

This implies  $5 \cdot cost(O) \geq cost(S)$ , thus proving Theorem 1.

The  $k$ -median problem is a special case of the  $k$ -facility location problem in which all facility costs are zero. For the  $k$ -median problem, there is a family of instances in which the ratio of the costs of a local optimum and a global optimum is arbitrarily close to 5 [1]. Thus our analysis of the locality gap for the  $k$ -facility location problem is tight. However, the special case in which the facility costs are distances to a fixed node may have lower locality gap.

## References

1. V. Arya, N. Garg, R. Khandekar, V. Pandit, K. Munagala, and A. Meyerson. "Local Search Heuristics for  $k$ -median and Facility Location Problems." In *Proceedings, 33rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 21–29, 2001.
2. E. Koutsoupias, C. H. Papadimitriou, "Worst-case equilibria." In *16th Annual Symposium on Theoretical Aspects of Computer Science*, pages 404–413, 1999.
3. C. Papadimitriou, "Algorithms, games, and the Internet". In *Proceedings of the 33rd Annual ACM Symposium on the Theory of Computing*, pages 749–753, 2001.
4. T. Roughgarden and E. Tardos. "How bad is selfish routing?" *Journal of the ACM*, 49(2):236–259, 2002. Preliminary version in FOCS '00.
5. T. Roughgarden. "The price of anarchy is independent of the network topology." In *Proceedings of the 34th Annual ACM Symposium on the Theory of Computing*, pages 428–437, 2002.
6. T. Roughgarden, "Selfish Routing." *PhD thesis*, Cornell University, May 2002.