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

Nikhil Devanur[1] and Naveen Garg[2] and Rohit Khandekar[2] and Vinayaka Pandit[3] and Amin Saberi[4] and Vijay Vazirani[1]

[1] College of Computing, Georgia Institute of Technology, USA. email: {nikhil, vazirani}@cc.gatech.edu
[2] Department of Computer Science, Indian Institute of Technology, Delhi, India. email: naveen@cse.iitd.ernet.in, rkhandekar@gmail.com
[3] IBM India Research Laboratory, New Delhi, India. email:pvinayak@in.ibm.com
[4] Stanford University, USA. email:saberi@stanford.edu

Abstract. In this paper, we define a network service provider game. We show that the price of anarchy of the defined game can be bounded by analyzing a local search heuristic for a related facility location problem called the k-facility location problem. As a result, we show that the k-facility location problem has a locality gap of 5. This result is of interest on its own. Our result gives evidence to the belief that the price of anarchy of certain games are related to analysis of local search heuristics.

1 Introduction

It is important to analyse the outcome of games involving multiple, selfish, non-cooperative agents which have a corresponding social cost. Koutsoupias and Papadimitriou [5] formulated the problem in terms of the *Nash equilibrium* attained by a set of independent, non-cooperative agents with rational behavior. In an environment in which each agent is aware of all the alternatives facing all the other agents, Nash equilibrium is a combination of choices (deterministic or randomized), one for each agent, in which, no agent has an incentive to unilaterally move away. The Nash equilibrium is known to deviate from overall optimum in many optimization scenarios. They defined *worst case equilibria* as the maximum value that the ratio of the overall optimum to the cost of a Nash Equilibrium can take over the set of all Nash equilibriums. Papadimitriou [6] called it as the *price of anarchy*. Since then, this notion has been used to analyse the efficiency of many games [7–9].

We define a network service provider game and consider the problem of bounding its price of anarchy. We first show that the price of anarchy of the game considered is the same as *locality gap* [1] of the k-facility location problem [4]. We then show that the k-facility location has a locality gap of 5. This result is of independent interest on its own. Our result gives evidence to the belief that price of anarchy and local search analysis are interrelated.

We consider a game on a network with a distinguished node called the *root*. Some of the nodes in the network are called *clients*, denoted by C. The clients

seek to obtain the root's service via a set of k service providers(SP) who may occupy one of the remaining nodes in the network, denoted by F. In a *configuration* each SP either decided to not occupy any of the nodes in F or occupies a node in F which is not occupied by any other SP. In a given configuration, each client is served by the closest SP. A client j pays to its closest SP i, an amount equal to its distance from the second closest SP (VCG cost). The revenue of an SP i is total amount paid to it by all the clients connected to i. The cost incurred by i to provide the service to its clients is the cost of connecting i to the root. The difference of the revenue and the cost incurred is the net profit of i. In order to maximize individual profits, the SPs may adopt strategies which are specified when we define the game later in the paper. A configuration of SPs is said to be in *Nash equilibrium* if none of the SPs can improve their profit using the strategies available to them. The cost of this Nash equilibrium is the total cost incurred by the SPs. The optimal cost is the minimum cost incurred by at most k SPs in serving all clients. The price of anarchy is supremum, over all Nash equilibria, the ratio of the cost of a Nash equilibrium to the optimal cost.

In the *k-facility location* problem, the input consists of a set of clients, C, and a set of facilities, F. Associated with each facility in F is a cost of opening a facility there. An integer k specifies the maximum number of open facilities. The goal is to determine the set of facilities to be opened such that the sum of opening costs and the costs of connecting each city to its closest facility is minimized. Charikar et.al. [3] gave 9.8-approximation for this problem and it was later improved to 6 in [4]. The k-facility location problem inherits features of both, the k-median and the uncapacitated facility location (UFL) 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 first constant factor approximation algorithm for the k-median problem was a $(6 + \frac{2}{3})$-approximation and was given by Charikar et.al. [3]. This ratio was improved to 6 in [4] and then to 4 in [2].

The UFL problem is a special case of the k-facility location problem when no limit is placed on the number of open facilities. Once again, a local search algorithm which at each step can add, delete or swap facilities has a locality gap of 3 [1, 2]. We show that local search with the same operations has a locality gap of 5 for the k-facility location problem. In Section 5, we exaplain why the analyses for the UFL and the k-median problem presented in [1] can not be directly adapted for the k-facility location problem.

Vetta [9] considers the social utility value of Nash Equilibrium of similar games. In this setting, the social utility value of the outcome of a game is given by a non-decreasing, submodular function and the decisions are controlled by a set of non-cooperative agents who seek to maximize their own private utility. He showed that the social utility value of a Nash Equilibrium is atleast half of the optimal social utility value. No such multiplicative bound is possible when the utility function is only submodular.

2 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 distances c_{ij} between $i, j \in F \cup C$ that satisfy metric properties. The cost of connecting the client $j \in C$ to a facility $i \in F$ is given by c_{ij}. An integer k specifies the limit on the number of open facilities. 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 $\texttt{fac}(S) = \sum_{i \in S} f_i$ denote its facility cost, let $\texttt{serv}(S) = \sum_{j \in C} \min_{i \in S} c_{ij}$ denote its service cost, and let $\texttt{cost}(S) = \texttt{fac}(S) + \texttt{serv}(S)$ denote its total cost. We define $\texttt{cost}(\emptyset) = \infty$. Consider the following local search algorithm for the k-facility location problem. We start by opening any subset $S \subseteq F$ of at most k facilities. We then try to reduce $\texttt{cost}(S)$ iteratively by employing three local operations. For $i \in F$, we use $S-i$ and $S+i$ to denote $S \setminus \{i\}$ and $S \cup \{i\}$ respectively. The three local operations are, *delete* a facility: if there is an $i \in S$ such that $\texttt{cost}(S-i) < \texttt{cost}(S)$, then $S \leftarrow S - i$; *add* a facility: if $|S| < k$ and there is an $i \in F \setminus S$ such that $\texttt{cost}(S+i) < \texttt{cost}(S)$, then $S \leftarrow S + i$; *swap* facilities: if there is an $i \in S$ and an $i' \in F \setminus S$ such that $\texttt{cost}((S-i)+i') < \texttt{cost}(S)$, then $S \leftarrow (S-i)+i'$.

We call this a *delete-add-swap* local search algorithm. We call $S \subseteq F$ with $|S| \leq k$ a *local optimum* solution if $\texttt{cost}(S)$ 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 a global optimum solution is called the *locality gap* of the local search algorithm. In Section 5, we prove the following theorem.

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

3 Service provider game

We are given a network with a distinguished node r called *root*. Remaining nodes in the network are partitioned into a set of clients C, and a set of service locations F. The distances c_{ij} between $i, j \in F \cup C \cup \{r\}$ satisfy metric properties. Suppose that there are k SPs. Each SP is allowed to occupy at most one service location. Two SPs cannot occupy the same service location. The assignment of SPs to the service locations, say $S \subseteq F$, defines a *configuration*. It is not necessary that every provider is assigned to a location.

Consider a configuration $S \subseteq F$ with $|S| \leq k$. For an SP $i \in S$, let $N_S(i)$ be the set of clients for which i is the closest of all the SPs in S. To serve the clients in $N_S(i)$ and connect to the root r, provider i incurs an expense of $\texttt{expense}_S(i) = c_{ir} + \sum_{j \in N_S(i)} c_{ij}$. Let $\texttt{expense}(S) = \sum_{i \in S} \texttt{expense}_S(i)$ be the total expense of all the SPs in S. For a client $j \in C$, let $S_j = \min_{i \in S} c_{ij}$ be the distance of j to the closest SP in S. Then $\texttt{expense}(S) = \sum_{i \in S} c_{ir} + \sum_{j \in C} S_j$.

Each client connects to the SP closest to it, but is charged the VCG payment, which is the distance to the second closest SP. The *revenue* of an SP $i \in S$ is

the total payment it receives from all the clients it serves, i.e., $\texttt{revenue}_S(i) = \sum_{j \in N_S(i)} T_j$ where $T = S - i$. The profit of an SP i is $\texttt{profit}_S(i) = \texttt{revenue}_S(i) - \texttt{expense}_S(i)$. Note that $\texttt{expense}_S(i), \texttt{revenue}_S(i)$ and $\texttt{profit}_S(i)$ are with respect to a particular configuration S.

Lemma 1. *The profit of a provider s in the configuration S is given by* $\texttt{profit}_S(s) = \texttt{expense}(S - s) - \texttt{expense}(S)$.

Proof. Let $T = S - s$.

$$\begin{aligned}
\texttt{expense}(T) - \texttt{expense}(S) &= \left(\sum_{i \in T} c_{ir} + \sum_{j \in C} T_j \right) - \left(\sum_{i \in S} c_{ir} + \sum_{j \in C} S_j \right) \\
&= \sum_{j \in C} (T_j - S_j) - c_{sr} \\
&= \sum_{j \in N_S(s)} T_j - \left(\sum_{j \in N_S(s)} S_j + c_{sr} \right) \\
&= \texttt{revenue}_S(s) - \texttt{expense}_S(s) \quad = \quad \texttt{profit}_S(s). \blacksquare
\end{aligned}$$

Each SP behaves selfishly and tries to maximize individual profit. This defines a game in which we allow only pure strategies and no two SPs can occupy the same location. The strategies of all the SPs defines a configuration. The payoff of a particular SP is equal to his profit in that configuration. A *Nash equilibrium* is a configuration so that no SP 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.

4 Connection between Price of Anarchy and Locality Gap

There is a natural correspondence between instances 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 expense c_{ir} that an SP i incurs in connecting to the root corresponds to the facility cost f_i. The expense c_{ij} incurred in connecting to the client j is just the cost of servicing client j by facility i. So, $\texttt{expense}(S)$ which is the total expense incurred by the SPs in a configuration S is the same as $\texttt{cost}(S)$ which is the cost of solution S in the k-facility location problem. We now show that a Nash equilibrium in the service provider game corresponds to a local optimum solution in the k-facility location instance with the delete-add-swap operations.

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, $\texttt{profit}_S(s) = \texttt{cost}(S - s) - \texttt{cost}(S) \geq 0$ for all $s \in S$. Therefore $\texttt{cost}(S)$ cannot be reduced by deleting a facility $s \in S$. Let $|S| < k$ and $S' = S + s$ for some $s \notin S$. Since any provider not in S did not occupy the location s, we have $\texttt{profit}_{S'}(s) \leq 0$. Therefore, from Lemma 1, we have $\texttt{cost}(S' - s) - \texttt{cost}(S') \leq 0$. Thus $\texttt{cost}(S) \leq \texttt{cost}(S + s)$. Therefore $\texttt{cost}(S)$ 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 $\texttt{profit}_{S'}(s') \leq \texttt{profit}_S(s)$. Let $T = S - s = S' - s'$. Then we have

$$\begin{aligned} \texttt{cost}(S') - \texttt{cost}(S) &= (\texttt{cost}(T) - \texttt{cost}(S)) - (\texttt{cost}(T) - \texttt{cost}(S')) \\ &= \texttt{profit}_S(s) - \texttt{profit}_{S'}(s') \quad \geq \quad 0. \end{aligned}$$

Therefore $\texttt{cost}(S)$ 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 a local optimum solution is a Nash equilibrium in our 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. ■

5 Proof of Theorem 1

The analysis uses some ideas from the analyses of local search for the k-median and the UFL problems by Arya et al. [1]. They proved that the 1-swap local search has a locality gap of 5 for the k-median problem and the delete-add-swap local search has a locality gap of 3 for the UFL problem. The k-median proof crucially uses the fact that the global optimum solution has exactly k facilities. However, for the k-facility location instance derived from the network service provider game, the global optimum may have much less than k facilities. The analysis for the UFL considers add operations irrespective of the number of facilities in the current solutions. If the k-facility location solution has exactly k facilities, we cannot add a facility to reduce its cost. Due to these reasons we cannot adapt the analyses in [1] to the k-facility location problem.

Let S denote a local optimum solution to the k-facility location problem and let O denote a global optimum solution. If $|S| < k$, then S is local optimum with respect to the addition operation as well. So the analysis for the UFL in [1] is directly applicable and we have the following lemma.

Lemma 2. *If* $|S| < k$, *then* $\texttt{cost}(S) \leq 3 \cdot \texttt{cost}(O)$.

So, in the rest of the analysis we assume that S, the local optimum has exactly k facilities implying that we can only consider swap and delete operations on S.

5.1 Notation and preliminaries

We use s to denote a facility in S and o to denote a facility in O. The other notation is as introduced in Section 2. 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 solutions S and O respectively. For a facility $s \in S$, let $N_S(s)$ denote the set of clients served by s in the solution S and for a facility $o \in O$, let $N_O(o)$ denote the set of clients served by o in the solution 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 in [1].

We now define a 1-1 onto function $\pi : N_O(o) \rightarrow N_O(o)$ for each $o \in O$ as follows. First consider a facility $o \in O$ that is not captured by any facility in S. Let $M = |N_O(o)|$. Order the clients in $N_O(o)$ as $c_0, \ldots, c_{M-1}$ such that for every $s \in S$, the clients in N_s^o are consecutive, that is, there exists p, q, $0 \leq p \leq q \leq M$ such that $N_s^o = \{c_p, \ldots, c_{q-1}\}$. Now, define $\pi(c_i) = c_j$ where $j = (i + \lfloor M/2 \rfloor)$ modulo M.

Lemma 3. *For each $s \in S$, we have $\pi(N_s^o) \cap N_s^o = \emptyset$.*

Proof. Suppose both $c_i, \pi(c_i) = c_j \in N_s^o$ for some $s \in S$. As s does not capture o, we have $|N_s^o| \leq M/2$. If $j = i + \lfloor M/2 \rfloor$, then $|N_s^o| \geq j - i + 1 = \lfloor M/2 \rfloor + 1 > M/2$. If $j = i + \lfloor M/2 \rfloor - M$, then $|N_s^o| \geq i - j + 1 = M - \lfloor M/2 \rfloor + 1 > M/2$. In both cases we have a contradiction. ∎

Now consider a facility $o \in O$ that is captured by a facility $s \in S$. Note that $|N_s^o| > |N_O(o) \setminus N_s^o|$. We pair each client $j \in N_O(o) \setminus N_s^o$ with a unique client $j' \in N_s^o$ and define $\pi(j) = j'$ and $\pi(j') = j$. Let $N \subseteq N_s^o$ be the subset of clients which are not paired in the above step; note that $|N| = 2|N_s^o| - |N_O(o)|$. For each $j \in N$, we define $\pi(j) = j$.

The function $\pi : N_O(o) \rightarrow N_O(o)$ satisfies the following properties for all $o \in O$.

P1. If $s \in S$ does not capture $o \in O$, then $\pi(N_s^o) \cap N_s^o = \emptyset$.
P2. If $s \in S$ captures $o \in O$ and if $((j \in N_s^o) \wedge (\pi(j) \in N_s^o))$, then $\pi(j) = j$.
P3. We have $\{j \in N_O(o) \mid \pi(j) \neq j\} = \{\pi(j) \in N_O(o) \mid \pi(j) \neq j\}$, $\forall o \in O$.

5.2 Deletes and Swaps considered

Since S is a local optimum solution, its cost cannot be reduced by doing any deletions and swaps. Since $|S| = k$, we cannot add a facility to reduce the cost. For any $s \in S$, we have $\texttt{cost}(S - s) \geq \texttt{cost}(S)$. For any $s \in S$ and $o \in O$, we have $\texttt{cost}(S - s + o) \geq \texttt{cost}(S)$. We now carefully consider some delete 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 swap $s \in S$ and $o \in O$, we reroute the clients in $N_S(s)$ and a subset of clients in $N_O(o)$ to the 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 $\texttt{cost}(S') - \texttt{cost}(S) \geq 0$ 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 $o' \neq o$, P1 implies that $s' \neq s$ and the rerouting is feasible. We call these swaps, **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 P1 and P2, such rerouting is feasible. We call these swaps, **Type 2** operations.
3. Let $G \subseteq S$ be the subset of facilities in S that are not swapped out in the Type 1 or Type 2 operations. Note that G is precisely the set of good facilities. Let $R \subseteq O$ be the subset of facilities in O that are not swapped-in in the Type 1 or Type 2 operations. Since $|S| = k \geq |O|$ and $|S \setminus G| = |O \setminus R|$, we have $|G| \geq |R|$. Let $G = \{s_1, \ldots, s_{|G|}\}$ and $R = \{o_1, \ldots, o_{|R|}\}$. We swap s_i with o_i for $1 \leq i \leq |R|$. For each of these swaps, we reroute the clients in $N_S(s_i) \cup N_O(o_i)$ as follows. All the clients in $N_O(o_i)$ are rerouted to o_i. The clients $j \in N_S(s_i) \setminus N_O(o_i)$ are rerouted to $s' \in S$ that serves $\pi(j)$ in S. Since s_i is a good facility, P1 implies $s' \neq s_i$ and hence this rerouting is feasible.

   We consider $|G| - |R|$ more operations as follows. For each i such that $|R| + 1 \leq i \leq |G|$, we delete s_i. After such a deletion, we reroute the clients $j \in N_S(s_i)$ to $s' \in S$ that serves $\pi(j)$ in S. Again from P1, we have $s' \neq s$ and hence this rerouting is feasible.

   We call these $|R|$ swaps and $|G| - |R|$ deletions, **Type 3** operations.
4. Let $R' \subseteq O$ be the subset of far facilities in O. Since no far facility is swapped-in in Type 1 or 2 operations, we have $R' \subseteq R$. It is no loss of generality to assume $R' = \{o_1, \ldots, o_{|R'|}\}$. Recall that $G = \{s_1, \ldots, s_{|G|}\}$ is the set of good facilities. We consider $|R'|$ swaps as follows. For each i such that $1 \leq i \leq |R'|$, we swap s_i with o_i. The clients $j \in N_O(o_i)$ such that $\pi(j) = j$ are rerouted to o_i. The clients $j \in N_S(s_i)$ are rerouted to $s' \in S$ that serves $\pi(j)$ in S. The remaining clients are not rerouted. We call these $|R'|$ swaps, **Type 4** operations.

Since S is a local optimum solution, the increase in the facility and service costs after each operation considered above is at least zero. In the sections to follow, we bound this increase due to all the 4 types of operations together. At this point, we remark that, the Type 4 operations are crucial for being able to bound the change in service cost of those clients $j \in C$ which are served by a far facility in O and $\pi(j) = j$.

Fig. 1. Rerouting a client $j \in N_O(o)$ when o is brought in

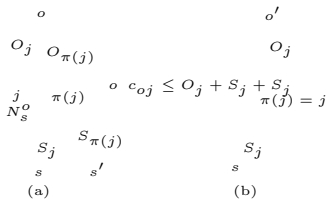

Fig. 2. Rerouting a client $j \in N_s^o$ when s is taken out and o is not brought in

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

$$2 \cdot \texttt{fac}(O) - \texttt{fac}(S). \tag{1}$$

5.4 Bounding the increase in the service cost

We call a facility $o \in O$ a *near* facility if it is captured by some $s \in S$ and o is not a far facility.

1. A client $j \in C$ served by a near facility in O is called "white" if $\pi(j) = j$.
2. A client $j \in C$ served by a far facility in O is called "gray" if $\pi(j) = j$.
3. A client $j \in C$ is called "black" if $\pi(j) \neq j$.

Recall that a client $j \in N_s^o$ is rerouted only when either o is brought in and/or s is taken out.

Lemma 4. *Increase in the service cost of a white client $j \in C$ over all operations is atmost $O_j - S_j$.*

Proof. Let $j \in N_s^o$ where $s \in S$ is a 1-bad or 2+bad facility that captures the near facility $o \in O$. Note that we swap s with o once - either as a Type 1 operation

or as a Type 2 operation. The increase in the service cost of j in this swap is $O_j - S_j$ (Figure 1). Since s or o are not considered in Type 3 or 4 operations, the client j is not rerouted in these operations. ■

Lemma 5. *Increase in the service cost of a gray client $j \in C$ over all operations is at most $3O_j - S_j$.*

Proof. Let j be served by a far facility $o' \in O$. Let s be the 2+bad facility in S that captures o'. Since $\pi(j) = j$, from P1 we have $j \in N_s^{o'}$. Let o be the closest facility in O that s captures. Note that $c_{so} \leq c_{so'}$. A gray client is not rerouted in Type 1 operations. In Type 2 operations, j is rerouted to o (Figure 2(b)) and increase in 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 increase is at most $c_{js} + c_{jo'} = S_j + O_j$. In Type 3 and 4 operations, increase in the service cost of j is $O_j - S_j$ each (Figure 1). So, the increase in service cost of j over all the operations is at most $(O_j + S_j) + 2(O_j - S_j) = 3O_j - S_j$. ■

Lemma 6. *The total increase in the service cost of a black client $j \in C$ in all the 4 types of operations is at most $2D_j + O_j - S_j$ where $D_j = O_j + O_{\pi(j)} + S_{\pi(j)} - S_j$.*

Proof. Let $j \in N_s^o$ for facilities $s \in S$ and $o \in O$. We first bound the increase in the service cost of a black client j due to operations of Type 1, 2, and 3. Amongst these operations, there is exactly one swap in which o is brought in. This swap contributes $O_j - S_j$ to the increase in service cost of j. The facility s may be either deleted once or considered in a swap with $o' \in O$. If it is considered in a swap with $o' \in O$ such that $o' = o$, then the increase in service cost of client j has already been accounted for in $O_j - S_j$. If s is deleted or considered in a swap with $o' \in O$ such that $o' \neq o$, j is rerouted to s' that serves $\pi(j)$ in S and the increase in its service cost is at most $c_{js'} - c_{js} \leq c_{jo} + c_{o\pi(j)} + c_{\pi(j)s'} - c_{js} = O_j + O_{\pi(j)} + S_{\pi(j)} - S_j = D_j$ (Figure 2(a)). Thus the total increase in service cost of j due to Type 1,2, and 3 operations is at most $D_j + O_j - S_j$.

In Type 4 operations, since $\pi(j) \neq j$, the client j is rerouted only when s is (possibly) taken out. In such a case, it is rerouted to s' that serves $\pi(j)$ in S (Figure2(a)) and the increase in its service cost is again at most $c_{js'} - c_{js} \leq c_{jo} + c_{o\pi(j)} + c_{\pi(j)s'} - c_{js} = O_j + O_{\pi(j)} + S_{\pi(j)} - S_j = D_j$. Thus the total increase in the service cost of j in all 4 types of operations is $2D_j + O_j - S_j$. ■

Lemma 7. *Increase in the service cost over all the operations is atmost $5 \cdot \texttt{serv}(O) - \texttt{serv}(S)$.*

Proof. Lemmas 4,5, and 6, imply that the the total increase in the service cost of all the clients in all the 4 types of operations is at most

$$\sum_{j:\text{white}} (O_j - S_j) + \sum_{j:\text{gray}} (3O_j - S_j) + \sum_{j:\text{black}} (2D_j + O_j - S_j).$$

Now from property P3 of the function π, we have $\sum_{j:\text{black}} S_j = \sum_{j:\text{black}} S_{\pi(j)}$ and $\sum_{j:\text{black}} O_j = \sum_{j:\text{black}} O_{\pi(j)}$.

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

Hence the total increase in the service cost of all the clients is at most $\sum_{j \in C}(5 \cdot O_j - S_j) = 5 \cdot \texttt{serv}(O) - \texttt{serv}(S)$. ■

5.5 Bounding the increase in the total cost

From (1) and Lemma 7, it follows that the total increase in cost due to all the operations is atmost $2 \cdot \texttt{fac}(O) - \texttt{fac}(S) + 5 \cdot \texttt{serv}(O) - \texttt{serv}(S)$. As S is a local optimum, in each operation the total cost increases by a non-negative amount. Together, these imply that $cost(S) \leq 5 \cdot cost(O)$, thus proving Theorem 1. The k-median problem has a family of instances in which the ratio of the costs of local optimum to the global optimum is asymptotically equal to 5 [1]. So, our analysis of locality gap is tight.

6 Conclusions

This is the first analysis of local search for the k-facility location problem. Other approximation algorithms for the problem [3, 2, 4] can not be used to bound price of anarchy. It would be interesting to see if price of anarchy of many more games can be analysed via local search analysis.

7 Acknowledgements

Vijay V. Vazirani was supported in part by NSF Grants 0311541, 0220343 and 0515186.

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." *Siam Journal of Computing"*, 33(2):544–562, 2004.
2. M. Charikar, and S. Guha. "Improved Combinatorial Algorithms for the Facility Location and k-Median Problems." In *Proceedings, IEEE Symposium on Foundations of Computer Science*, pages 378–388, 1999.
3. M. Charikar, S. Guha, E. Tardos, and D. B. Shmoys. "A constant-factor approximation algorithm for the k-median problem." In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing*, pages 1–10, 1999.
4. K. Jain, V. 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*, pages 2–13, 1999.
5. E. Koutsoupias, C. H. Papadimitriou, "Worst-case equilibria." In *16th Annual Symposium on Theoretical Aspects of Computer Science*, pages 404–413, 1999.
6. C. Papadimitriou, "Algorithms, games, and the Internet". In *Proceedings of the 33rd Annual ACM Symposium on the Theory of Computing*, pages 749–753, 2001.
7. T. Roughgarden and E. Tardos. "How bad is selfish routing?" *Journal of the ACM*, 49(2):236–259, 2002. Preliminary version in FOCS '00.

8. 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.
9. A. Vetta, “Nash Equilibria in Competitive Societies, with Applications to Facility Location, Traffic Routing and Auctions.” *In Proceedings of the 43rd Symposium on Foundations of Computer Science, 2002.*