Algorithmica manuscript No.
(will be inserted by the editor)

Local Search for Red-Blue Median and its Generalizations

M. Hajiaghayi · R. Khandekar · G. Kortsarz

Received: date / Accepted: date

Abstract In this paper, we consider the following *red-blue median* problem which is a generalization of the well-studied k*-median* problem. The input consists of a set of *red* facilities, a set of *blue* facilities, and a set of clients in a metric space and two integers $k_r, k_b \geq 0$. The problem is to open at most k_r red facilities and at most k_b blue facilities and minimize the sum of distances of clients to their respective closest open facilities.

We show, somewhat surprisingly, that the following simple local search algorithm yields a constant factor approximation for this problem. Start by opening any k_r red and k_b blue facilities. While possible, decrease the cost of the solution by closing a pair of red and blue facilities and opening a pair of red and blue facilities.

We also show that the same algorithm yields a constant factor approximation for the *prize-collecting* version of the red-blue median problem as well.

Keywords facility location · k-median · prize-collecting · local search algorithms

1 Introduction

Consider the following natural problem called the *red-blue median problem* which generalizes the famous metric k-median problem. The input is a set of facilities $\mathcal{F}$ and a set of clients $\mathcal{C}$ in a metric space. The distance between two points in this metric space $i, j \in \mathcal{F} \cup \mathcal{C}$ is denoted by $d(i, j)$. The facilities are partitioned into two sets: *red* facilities $\mathcal{R}$ and *blue* facilities $\mathcal{B}$. The input also includes two integers $k_r, k_b > 0$. Given a subset of *open* facilities, a client j gets served by the nearest open facility. The goal

A preliminary version of this paper appeared in Proceeding of the 18th Annual European Symposium on Algorithms (ESA'10). Part of this work was done while the authors were meeting at DIMACS. We would like to thank DIMACS for hospitality. Third author was partially supported by NSF grant 0819959.

MohammadTaghi Hajiaghayi
AT&T Labs– Research & University of Maryland. E-mail: hajiagha@research.att.com

Rohit Khandekar
IBM T.J. Watson Research Center. E-mail: rohitk@us.ibm.com

Guy Kortsarz
Rutgers University–Camden. E-mail: guyk@camden.rutgers.edu

of the problem is to open a subset of red facilities $R \subseteq \mathcal{R}$ and a subset of blue facilities $B \subseteq \mathcal{B}$ such that

– $|R| \leq k_r$ and $|B| \leq k_b$,
– the total connection cost $\mathbf{cost}(R, B) := \sum_{j \in \mathcal{C}} d(j, R \cup B)$ is minimized.

Here $d(j, S) = \min_{i \in S} d(j, i)$ denotes the shortest distance from j to any point in S. The special case with $k_b = 0$ corresponds to the well-known *k-median problem.* Our problem and in general the problem with arbitrary number of facility types are motivated by applications to designing content distribution networks and other problems in telecommunications where it is vital to obtain solutions that do not violate the budget on the number of open facilities of a particular type (see e.g., [6,3]). In this paper, we present a constant factor approximation algorithm for the red-blue median problem.

In some scenarios, each client can be satisfied by a different vendor at a cost. Indeed, this cost is called the *penalty* of this client that we pay in case she is not connected to one of our deployed facilities. See [4,10,7,16,18] for work on prize collecting problems. More formally, in our problem formulations we can assume that each client $j \in \mathcal{C}$ has a *penalty* $p_j \in \mathbf{Q}_+$. The client pays the service cost, i.e., its distance to the nearest open facility, if it is at most its penalty p_j; otherwise the client remains unserved and pays the penalty p_j. The goal then is to minimize the sum of connection costs and paid penalties. In this paper we improve the best-known approximation for the prize-collecting k-median problem from 4 [10] to $3 + \epsilon$.

Another related and well-motivated problem is the *knapsack-median* problem in which given a non-negative opening cost w_i for each facility i, we want to open a set of facilities whose opening cost is within our budget W and minimize the total connection cost. This budget constraint is a Knapsack constraint and for general opening costs, we are not aware of any constant approximation algorithm that does not violate the budget W. Similarly to Knapsack, if we are allowed to violate the budget within a factor $1+\epsilon$, one can obtain a constant factor approximation algorithm using the filtering method of Lin and Vitter [26]. In fact, Charikar and Guha [8] obtain a better tradeoff between the budget violation and the approximation ratio. In addition, for polynomially bounded opening costs (for which Knapsack is solvable), we can solve the problem on trees without violating budget W (see Section 4). Turning to general graphs using probabilistic embeddings of general metrics into tree metrics [5,13]), this immediately results in an $O(\log n)$ approximation algorithm for knapsack-median in general metrics without violating budget W when opening costs are polynomially bounded. To the best of our knowledge, there is no known work on the red-blue median problem and some of its natural generalizations. In the special case of the knapsack-median problem when there are only two different facility opening costs, one can guess the number of facilities of each type in the optimal solution. Thus this special case can be reduced to the red-blue median problem. Our results therefore imply a constant factor approximation algorithm for this special case of the knapsack-median problem.

1.1 Related results

As mentioned above, an important special case of the red-blue median problem is the well-known k-median problem. The first constant factor approximation for the k-median problem was given by Charikar et al. [9], which was subsequently improved by Jain-Vazirani [21], Charikar-Guha [8], and Arya et al. [3]. The latter presents the current best approximation factor of $3 + \epsilon$ for k-median via a local search heuristic. Their analysis was recently simplified by Gupta and Tangwongsan [17]. The problem cannot be approximated within a factor strictly less than $1 + 2/e$, unless $\mathbf{NP} \subseteq \mathbf{DTIME}[n^{O(\log \log n)}]$ [20]. It is known that the integrality gap of the natural LP relaxation of the problem is at most 3, but currently there is no algorithm that achieves a 3-approximation in polynomial time [1]. An extension of k-median to the case in which we can open at most k facilities, but also have to pay their facility opening cost was studied by [12], who gave a 5-approximation. The k-median problem with penalties was also considered; the current best approximation factor for prize-collecting k-median is 4 due to Charikar et al. [10]. The problem in which the underlying metric is Euclidean, although NP-hard [28], admits a PTAS due to the results of Arora, Raghavan, and Rao [2], and then Kolliopoulos and Rao [22] (who provided an almost-linear time algorithm).

The Lagrangian relaxation approach was used by Jain and Vazirani [21] for the k-median problem. When we apply this approach to the red-blue median problem, we can get two solutions whose convex combination has cost at most a constant factor times the optimum cost. These two solutions have k_r^1 (resp. k_r^2) red and k_b^1 (resp. k_b^2) blue facilities where $k_r^1 + k_b^1 = k_r^2 + k_b^2 = k_r + k_b$. It may happen, for example, that $k_r^1 > k_r$ and $k_b^2 > k_b$, i.e., the bound on red facilities is violated in the first solution and the bound on blue facilities is violated in the second solution. Unlike the case for the k-median, both of these solutions may be infeasible. Therefore, it seems very hard to combine them to get a solution that has no violation while having cost within a constant factor of the optimum cost. The observation that the Lagrangian relaxation approach fails for the red-blue median problem was also shared and verified by Jain [19].

Subsequent to our work, Krishnaswamy et al. [24] considered a substantially more general *matroid median problem* in which there is a matroid structure on the set of facilities and one is allowed to open a set of facilities that is an independent set in the matroid. This problem not only generalizes the red-blue median problem but also the version with more than two colors and their respective budgets. They present *linear programming based* constant approximation algorithms for the matroid median problem and its prize-collecting version. They also present 16-approximation for the knapsack-median problem while violating the budget by $(1 + \epsilon)$ factor for any $\epsilon > 0$. Their algorithm runs in time $n^{O(1/\epsilon)}$.

Local search based approaches. From a practical point of view, a simple combinatorial algorithm is much more desirable than one that requires solving a linear programming relaxation. To this end, our main approach in this paper is to extend the local search technique which is a popular heuristic for hard combinatorial optimization problems. Relatively few instances of approximation guarantees via local search are known. Korupolu, Plaxton, and Rajaraman [23] gave the first approximation guarantees of this type for the facility location and k-median problems based on a simple local search heuristic proposed by Kuehn and Hamburger [25]. For the k-median problem, however, they violate the constraint on the number of open facilities by a factor $1+\epsilon$. Later Arya et al. [3] could approximate the problem without violating this constraint. Local search was later used for other facility location type problems [27,31,11,30] and recently even for maximum generalized assignment [15] and maximizing submodular functions [14]. Recently and independently of our work, Meyerson and Tagiku [29] proved that the q-swap local search algorithm yields $(3 + 2/q)$-approximation for the prize-collection version of the k-median problem as well.

1.2 Our results

The main result of this paper is a constant factor approximation algorithm for the red-blue median problem via novel analysis of a natural local search algorithm. More formally, we analyze the following local search algorithm.

```
1. Let $R \subset \mathcal{R}$ and $B \subset \mathcal{B}$ be *arbitrary* subsets with $|R| = k_r$ and $|B| = k_b$.
2. While there exist $r \in R$, $r' \in \mathcal{R}$ and $b \in B$, $b' \in \mathcal{B}$ such that
       $\mathbf{cost}(R - r + r', B - b + b') < \mathbf{cost}(R, B)$
   do: $R \leftarrow R - r + r'$ and $B \leftarrow B - b + b'$.
3. Output $R$ and $B$.
```

Here $S - s_1 + s_2$ denotes $(S \setminus \{s_1\}) \cup \{s_2\}$. Since r and r' (or b and b') may be identical, our algorithm outputs a locally optimal solution w.r.t. three local operations: (1) delete a red facility and add a red facility, (2) delete a blue facility and add blue facility, and (3) delete a red facility and a blue facility and add a red facility and a blue facility. In Section 2, we prove the following theorem.

Theorem 1 *The above local search algorithm yields a constant factor approximation to the red-blue median problem.*

It is somewhat surprising that this natural local search algorithm works. We point out why in the next section by explaining the main challenges in the analysis. We omit the standard details regarding how to make this algorithm run in polynomial time. Our techniques improve the best known result for the standard k-median problem. The previous approximation is factor 4 of the LP-rounding algorithm for prize-collecting k-median due to Charikar et al. [10]. We prove the following theorem in Section 3.

Theorem 2 *The multi-swap local search algorithm of Arya et al. [3] yields* $(3 + \epsilon)$-*approximation for the prize-collecting* k-*median problem.*

1.3 An overview of our techniques

The local search analyses [3, 17] for the k-median problem begin by considering a locally optimal solution, and show that since a *carefully chosen* set of test swaps are non-improving, one can infer some relationship between our cost and the optimal cost. Gupta and Tangwongsan [17] define this set of test swaps based on distance information about which of the optimal facilities O are close to which of our facilities S. The intuition is simple: consider each of the optimal facilities in O, and look at the closest facility to it in S. If some facility $s \in S$ is the closest to only one facility $o \in O$, then we should try swapping s with o. However, if there is some facility $s \in S$ that is the closest facility to many facilities in O, then swapping s might be bad for our solution, and hence we do not want to close this facility in any potential move.

Let us now understand why this standard local search analysis does not extend easily to the red-blue median problem. For the red-blue median problem, the choice of test swaps needed for the analysis to work may conflict with the budget constraints on the number of red and blue facilities allowed. For example, after deleting, say, a red facility, to keep the cost bounded, one may need to add a blue facility to serve the clients previously served by the dropped red facility. This happens, for example, when there is no other red facility close-by. In such a case, we are forced to delete another blue facility and possibly add another red facility in order to balance the number of red and blue facilities. As a result, bounding the cost of the solution after the swap becomes much trickier.

Our analysis begins by partitioning the solutions S and O into *blocks* (see Section 2.1) with some useful structural properties. Intuitively speaking, a block is a subset of $S \cup O$ for which the test swaps can be analyzed "independently" of other blocks, even when a test swap involves rerouting clients served by facilities from multiple blocks. These blocks are defined based on the distances and the colors of the facilities. For example, if a facility $s \in S$ is the closest facility in S for exactly one facility $o \in O$ and furthermore s and o have the same color, the pair $\{s, o\}$ defines a block. Another example of a block is as follows. Let $s_i \in S$ be the closest facility in S for exactly one facility $o_i \in O$ for $i = 1, 2$. If s_i has the same color as o_i, then $\{s_i, o_i\}$ defines a block. On the other hand if s_1 has the same color as o_2, s_2 has the same color as o_1 and the two colors are different, the set $\{s_1, s_2, o_1, o_2\}$ defines a block. In general, a block contains an equal number of red facilities and an equal number of blue facilities from the two solutions S and O such that for any facility $o \in O$ in a block, the closest facility to it in S is also in the same block. A typical block also satisfies a key property: it contains a large number of facilities in S that are not the closest facilities in S to any facility in O. It turns out that such facilities, called *very good facilities*, are compatible to be swapped with any facility in O [17] and their abundance is crucial to the overall analysis. We use a careful counting argument to show that a partitioning into blocks satisfying these properties exists.

In Section 2.2, we describe the test swaps for any single block. If s_i and o_i described above have the same color, we can consider the swap: add o_i and delete s_i. However, if $s = s_i$ is the closest one to several facilities $\{o_1, \ldots, o_l\}$ in the optimal solution, then deleting s may be bad for our solution. The previous k-median analyses, therefore, avoided swaps in which s is deleted.

Unfortunately, it turns out that we do not have the luxury of avoiding such swaps. Consider, for example, the case where $k_r = 1$ and s is the only red facility in S. Suppose that o is the unique red facility in O. To bound the cost of clients served by o in solution O, we need to consider a test swap in which o is added. Note however that if o is added, s must be deleted to satisfy the budget $k_r = 1$. Our analysis considers a test swap in which we delete s and open the facility $o_i \in \{o_1, \ldots, o_l\}$ that is closest to s. If s and o_i are of different colors, we combine this swap with another carefully chosen red-blue swap to balance the number of red and blue facilities. The cost after such a swap may potentially be significantly higher than that of the optimal solution. To "cancel" this high cost, we consider several other test swaps in which facilities $\{o_1, \ldots, o_l\}$ are added one-by-one. Using the properties of a block mentioned above, we show how to bound the overall cost for all the swaps considered.

In our opinion, these new swaps and a method to bound their costs is the main technical contribution of our paper. We encourage the reader to read the exposition in paragraphs titled 'Intuition' and 'Example in Figure 1' in Section 2.3 for further intuition behind our approach.

The prize-collecting version. We show that the multi-swap local search algorithm of the k-median problem [3] yields $(3+\epsilon)$-approximation for the prize-collecting k-median problem. The proof is based on the techniques of Arya et al. [3] or Gupta and Tangwongsan [17] applied to the clients that do not pay penalty in either solution S or O. The other clients contribute the same amount to either solutions and thus are easy to handle. Essentially the same line of argument holds for the prize-collecting version of red-blue problem.

1.4 Future work

As mentioned before, Krishnaswamy et al. [24] present a linear programming based approximation algorithm for a substantially more general matroid median problem which includes the case of multiple types of facilities. It will be interesting to see if the local search techniques generalize to multiple types of facilities.

By combining the techniques of Theorems 1 and 2, we believe that local search based algorithms could be obtained for the prize-collecting version of the red-blue median problem as well.

2 Proof of Theorem 1

We begin with some notation and preliminaries. We call the local search operations that our algorithm tests as *valid swaps*. A test swap may or may not improve the cost of the current solution. Let $O = R^* \cup B^*$ denote the optimal solution where $R^* \subset \mathcal{R}$ and $B^* \subset \mathcal{B}$ and let $S = R \cup B$ denote the locally optimal (also called local) solution. For a facility $o \in O$, let $N^*(o)$ denote the clients that are served by o in solution O, i.e., these clients have o as the closest facility among facilities in O. Similarly, for $s \in S$, let $N(s)$ denote the clients that are served by s in solution S. For $A \subset O$, let $N^*(A) = \cup_{o \in A} N^*(o)$ and for $A \subset S$, let $N(A) = \cup_{s \in A} N(s)$. For a client $j \in \mathcal{C}$, let $O_j = d(j, O)$ and $S_j = d(j, S)$ be its contribution to the optimal and local solutions respectively.

Roadmap. To bound the cost of a local solution S relative to that of optimal solution O, we consider several *test swaps*, which we know are non-improving. To bound the contribution S_j of a client j, we would ideally like to delete the facility s that serves j in S, add a facility o that serves j in O and reroute j from s to o. The increase in the contribution of client j after this swap is $O_j - S_j$ and the overall increase in the cost is non-negative. The $-S_j$ terms can then be used to bound $\sum_j S_j$ in terms of $\sum_j O_j$. There are several hurdles however. First of all s and o may have different colors and thus the above swap is not even valid. To make it valid we have to delete another facility from S and add another facility from O to balance the colors. Furthermore, there are clients that are not always served by the added facilities in the solution O. The rerouting of such clients may incur additional $+S_j$ terms. In such cases, we may have to consider more test swaps to obtain $-S_j$ terms to cancel the $+S_j$ terms. The overall set of swaps needs to be carefully chosen so that we obtain at least one $-S_j$ term and at most a constant number of $+O_j$ terms for each client j. Below, we define the notion of *very good*, *good* and *bad* facilities depending on their ability to obtain $-S_j$ terms for various clients. But to do so, we first define functions η and μ capturing the proximity of the facilities in S and O.

Definition 1 (functions η and μ) Define a function $\eta : O \to S$ as follows. For $o \in O$, let $\eta(o)$ be the facility in S that it closest to o, where ties are broken arbitrarily. Thus we have $d(o, \eta(o)) = d(o, S)$.

For a facility $s \in S$ with $\eta^{-1}(s) \neq \emptyset$, define $\mu(s)$ to be the facility in $\eta^{-1}(s)$ that it closest to s where ties are broken arbitrarily, i.e., we have $d(s, \mu(s)) = d(s, \eta^{-1}(s))$.

See Figure 1 for an example. Note that if $o \in O \cap S$, then we have $\eta(o) = o$. The definition of function η is motivated by the paper of Gupta and Tangwongsan [17] who offer a simplified proof of the k-median local search algorithm of Arya et al. [3].

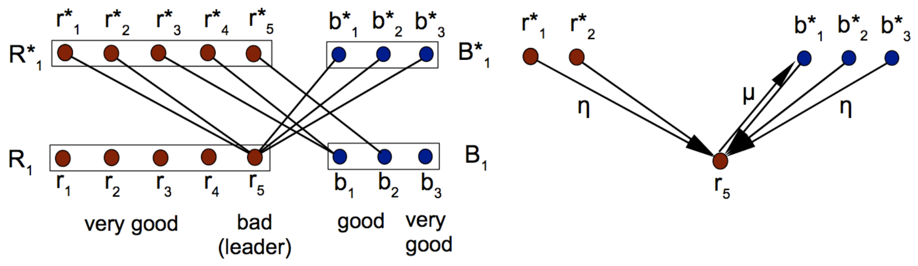


Fig. 1: On the left is an example of `block`-1: the facilities $R_1^*, B_1^* \subset O$ are shown at the top while $R_1, B_1 \subset S$ are shown at the bottom. We draw an edge between each $o \in R_1^* \cup B_1^*$ and $\eta(o) \in R_1 \cup B_1$. A single bad facility in $R_1 \cup B_1$, called leader, is r_5. The facilities $r_1, \ldots, r_4, b_3$ are very good while the facilities b_1, b_2 are good. Here case 4(a) holds. On the right is an example of functions η and μ. We have $\mu(r_5) = b_1^* \in \eta^{-1}(r_5)$.

Definition 2 (very good, good, and bad facilities) We call a facility $s \in S$ *very good*, if $\eta^{-1}(s) = \emptyset$; *good*, if $\eta^{-1}(s) \neq \emptyset$ and no facility in $\eta^{-1}(s)$ has the same color as s; and *bad*, if some facility in $\eta^{-1}(s)$ has the same color as s.

2.1 The blocks

We now present a procedure (see Figure 2) to partition the set R^* into $R_1^*, \ldots, R_t^*$, the set B^* into $B_1^*, \ldots, B_t^*$, the set R into $R_1, \ldots, R_t$, and the set B into $B_1, \ldots, B_t$ for some integer t. The parts R_i^*, B_i^*, R_i, B_i are said to form `block`-i for $i = 1, \ldots, t$. Note that this procedure is used only for the sake of analysis. It shows how to first compute `block`-1 and then recursively compute `block`-i for $i = 2, \ldots, t$.

Lemma 1 *The partitions of R^*, B^*, R, and B computed in Figure 2 satisfy the following properties. For all $i = 1, \ldots, t$, we have*

1. $|R_i^*| = |R_i|$ *and* $|B_i^*| = |B_i|$.
2. *For each $o \in R_i^* \cup B_i^*$, we have $\eta(o) \in R_i \cup B_i$.*
3. *At most one facility in $R_i \cup B_i$ is bad. We call such a facility* leader.
4. *If there is a leader in $R_i \cup B_i$, we have*
   (a) *either all facilities in R_i, except the leader, are very good,*
   (b) *or all facilities in B_i, except the leader, are very good.*

```
Compute block-$i$, i.e., $R^*_i$, $B^*_i$, $R_i$, and $B_i$ for $i \geq 1$.
 0. Let $i = 1$.
 1. Start with $R^*_i = B^*_i = R_i = B_i = \emptyset$.
 2. If there is a bad facility $r \in R$ such that $|\eta^{-1}(r)| = 1$, then let $R_i = \{r\}$, $B_i = \emptyset$,
    $R^*_i = \eta^{-1}(r)$, $B^*_i = \emptyset$, and go to step 9. If there is a bad facility $b \in B$ such that
    $|\eta^{-1}(b)| = 1$, then let $R_i = \emptyset$, $B_i = \{b\}$, $R^*_i = \emptyset$, $B^*_i = \eta^{-1}(b)$, and go to step 9.
 3. If there are good facilities $r \in R$ and $b \in B$ such that $|\eta^{-1}(r)| = |\eta^{-1}(b)| = 1$, then
    let $R_i = \{r\}$, $B_i = \{b\}$, $R^*_i = \eta^{-1}(b)$, $B^*_i = \eta^{-1}(r)$, and go to step 9.
 4. If there is no bad facility in $S$, let $R^*_i = R^*$, $B^*_i = B^*$, $R_i = R$, $B_i = B$, and go to
    step 9. Otherwise let $s \in S$ be a bad facility such that $|\eta^{-1}(s)|$ is maximum.
 5. Add $s$ to either $R_i$ or $B_i$ according to whether it is a red or a blue facility. Add
    facilities in $\eta^{-1}(s)$ to $R^*_i$ and $B^*_i$ according to their colors.
 6. If $|R^*_i| > |R_i|$, then add $|R^*_i| - |R_i|$ very good or good red facilities from $R$ to $R_i$.
    While doing so, give a preference to very good red facilities. For each facility $s$ thus
    added to $R_i$, add facilities in $\eta^{-1}(s)$ to $B^*_i$.
 7. If $|B^*_i| > |B_i|$, then add $|B^*_i| - |B_i|$ very good or good blue facilities from $B$ to $B_i$.
    While doing so, give a preference to very good blue facilities. For each facility $s$ thus
    added to $B_i$, add facilities in $\eta^{-1}(s)$ to $R^*_i$.
 8. If $|R^*_i| \neq |R_i|$ or $|B^*_i| \neq |B_i|$, go to step 6.
 9. $R^* \leftarrow R^* \setminus R^*_i$, $B^* \leftarrow B^* \setminus B^*_i$, $R \leftarrow R \setminus R_i$, $B \leftarrow B \setminus B_i$.
10. If $R \neq \emptyset$, let $i \leftarrow i + 1$ and go to step 1.
```

Fig. 2: A procedure to compute partitions of R^*, B^*, R, and B

Proof We prove this lemma only for $i = 1$; a similar argument holds for $i > 1$. It is easy to argue by induction that during the procedure in Figure 2, we maintain the following invariants: $|R^*_1| \geq |R_1|$ and $|B^*_1| \geq |B_1|$. Note that property 2 above holds since we add a facility o to $R^*_1 \cup B^*_1$ only if $\eta(o) \in R_1 \cup B_1$. Also property 3 hold since we add at most one bad facility in the beginning of the procedure.

We argue that in steps 6 or 7, we do not get stuck, i.e., there always exist a desired number of very good or good facilities to add. Suppose that at some point in the procedure, the condition $p := |R^*_1| - |R_1| > 0$ for step 6 holds. Since $|R^*| = |R|$, we have $|R \setminus R_1| - |R^* \setminus R^*_1| = p$. Now for each $o \in R^*_1$, we have $\eta(o) \in R_1 \cup B_1$, i.e., $\eta(o) \notin R \setminus R_1$. Therefore by a simple counting argument, there must exist at least $p = |R^*_1| - |R_1|$ very good or good facilities in $R \setminus R_1$, as desired. A similar argument holds also for step 7. Now since the procedure terminates, it is easy to see that property 1 holds from the termination condition and the fact that $|R^*| = |R|$ and $|B^*| = |B|$.

Now in order to prove property 4, we assume that both 4(a) and 4(b) do not hold and get a contradiction. Note that in steps 6 and 7, while adding new facilities, we give preference to very good facilities. Recall also that the procedure always maintains the invariant $|R^*_1| \geq |R_1|$ and $|B^*_1| \geq |B_1|$. Now assume that both 4(a) and 4(b) do not hold at some point. Assume without loss of generality that the first good facility added to R_1 was added before the first good facility was added to B_1. Consider the time just before the first good facility was added to B_1. Since we are trying to add a facility to B_1, it must hold that $|R^*_1 \cup B^*_1| > |R_1 \cup B_1|$, i.e., $|(R^* \cup B^*) \setminus (R^*_1 \cup B^*_1)| < |(R \cup B) \setminus (R_1 \cup B_1)|$. Furthermore, there are no very good facilities left in $R \setminus R_1$ or $B \setminus B_1$. Note now that each facility $s \in (R \cup B) \setminus (R_1 \cup B_1)$, being either good or bad, has $|\eta^{-1}(s)| \geq 1$ and $\eta^{-1}(s) \subset (R^* \cup B^*) \setminus (R^*_1 \cup B^*_1)$ implying that $|(R^* \cup B^*) \setminus (R^*_1 \cup B^*_1)| \geq |(R \cup B) \setminus (R_1 \cup B_1)|$. This is a contradiction. □

2.2 The swaps

Since $S = R \cup B$ is a local solution, any swap of a red facility and a blue facility does not decrease the cost of the solution, i.e., $\mathbf{cost}(R - r^- + r^+, B - b^- + b^+) \geq \mathbf{cost}(R, B)$. We use $\mathtt{swap}(r^-, r^+ \mid b^-, b^+)$ to denote this swap. When $r^- = r^+$, we also use $\mathtt{swap}(b^-, b^+)$ to denote this swap. Similarly, when $b^- = b^+$, we also use $\mathtt{swap}(r^-, r^+)$ to denote this swap. We now consider several inequalities of this type and add them to get the desired result. For each such swap considered below, we upper bound $\mathbf{cost}(R - r^- + r^+, B - b^- + b^+) - \mathbf{cost}(R, B)$ by giving a feasible assignment of clients to facilities.

Recall the definition of valid swaps; we call a swap valid if it does not change the number of red and blue facilities in the solution.

Lemma 2 *For all $i = 1, \ldots, t$, the following holds.*

1. *Let $s \in R_i$ (resp. $s \in B_i$) be a very good facility and $o \in R_i^*$ (resp. $o \in B_i^*$) be any facility. Then*
$$\sum_{j \in N^*(o)} (O_j - S_j) + \sum_{j \in N(s) \setminus N^*(o)} 2O_j \geq 0. \tag{1}$$
2. *Let $s \in R_i$ (resp. $s \in B_i$) be either good or bad facility with $o = \mu(s) \in R_i^*$ (resp. $o \in B_i^*$). Then*
$$\sum_{j \in N^*(o)} (O_j - S_j) + \sum_{j \in N(s) \cap N^*(\eta^{-1}(s) \setminus \{o\})} (O_j + S_j) + \sum_{j \in N(s) \setminus N^*(\eta^{-1}(s))} 2O_j \geq 0. \tag{2}$$
3. *Let $s_1 \in R_i \cup B_i$ be either good or bad facility, $s_2 \in R_i \cup B_i$ be a very good facility, and $o_2 \in R_i^* \cup B_i^*$ be any facility such that deleting s_1, s_2 and adding $o_1 = \mu(s_1), o_2$ is a valid swap. Then*
$$\sum_{\substack{j \in N^*(o_1) \\ \cup N^*(o_2)}} (O_j - S_j) + \sum_{\substack{j \in [N(s_1) \cup N(s_2)] \cap \\ [N^*(\eta^{-1}(s_1) \setminus \{o_1, o_2\})]}} (3O_j + S_j) + \sum_{\substack{j \in [N(s_1) \cup N(s_2)] \setminus \\ [N^*(\eta^{-1}(s_1) \cup \{o_2\})]}} 2O_j \geq 0. \tag{3}$$
4. *Let $s_1, s_2 \in R_i \cup B_i$ be either good or bad facilities such that deleting s_1, s_2 and adding $o_1 = \mu(s_1), o_2 = \mu(s_2)$ is a valid swap. Then*
$$\sum_{\substack{j \in N^*(o_1) \\ \cup N^*(o_2)}} (O_j - S_j) + \sum_{\substack{j \in [N(s_1) \cup N(s_2)] \cap \\ [N^*(\eta^{-1}(s_1) \setminus \{o_1\}) \cup N^*(\eta^{-1}(s_2) \setminus \{o_2\})]}} (3O_j + S_j) + \sum_{\substack{j \in [N(s_1) \cup N(s_2)] \setminus \\ [N^*(\eta^{-1}(s_1)) \cup N^*(\eta^{-1}(s_2))]}} 2O_j \geq 0. \tag{4}$$

Proof For a client j, let $s(j)$ denote the facility that serves j in solution S and let $o(j)$ denote the facility that serves j in solution O.

For item 1, consider $\mathtt{swap}(s, o)$. We reroute clients as follows. A client $j \in N^*(o)$ is rerouted to o and thus the increase in its service cost is $O_j - S_j$. A client $j \in N(s) \setminus N^*(o)$ is rerouted to $\eta(o(j))$. Note that $\eta(o(j)) \neq s$ since s is very good. The increase in its service cost is thus $d(j, \eta(o(j))) - S_j \leq O_j + d(o(j), \eta(o(j))) - S_j \leq O_j + d(o(j), s(j)) - S_j \leq O_j + O_j + S_j - S_j = 2O_j$. This sequence of inequalities follows from repeated use of triangle inequality. The clients not in $N^*(o) \cup N(s)$ are not rerouted. This proves item 1.

For item 2, consider $\mathtt{swap}(s, o)$. A client $j \in N^*(o)$ is rerouted to o and thus the increase in its service cost is $O_j - S_j$. Consider a client $j \in N(s) \setminus N^*(\eta^{-1}(s))$. Since $o(j) \notin \eta^{-1}(s)$, we have $\eta(o(j)) \neq s$. Such a client is therefore rerouted to $\eta(o(j))$ and thus the increase in its service cost is $d(j, \eta(o(j))) - S_j \le 2O_j$ as shown in item 1. A client $j \in N(s) \cap N^*(\eta^{-1}(s) \setminus \{o\})$ is rerouted to o and thus the increase in its service cost is $d(j, o) - S_j \le d(j, s(j)) + d(s(j), o) - S_j \le S_j + d(s(j), o(j)) - S_j \le O_j + S_j$. Here $d(s(j), o) \le d(s(j), o(j))$ follows from $o(j) \in \eta^{-1}(s)$, $o = \mu(s)$, and the definition of μ. The clients not in $N^*(o) \cup N(s)$ are not rerouted. This proves item 2.

The proofs of items 3 and 4 are very similar. Therefore we prove item 4 and omit the proof of item 3. For item 4, consider the swap: delete s_1, s_2 and add o_1, o_2. In this swap, we reroute the clients as shown in Figure 3 and described follows. A client $j \in N^*(o_1)$ is rerouted to o_1 and a client $j \in N^*(o_2)$ is rerouted to o_2. Clearly the increase in service cost of clients $j \in N^*(o_1) \cup N^*(o_2)$ is $O_j - S_j$.

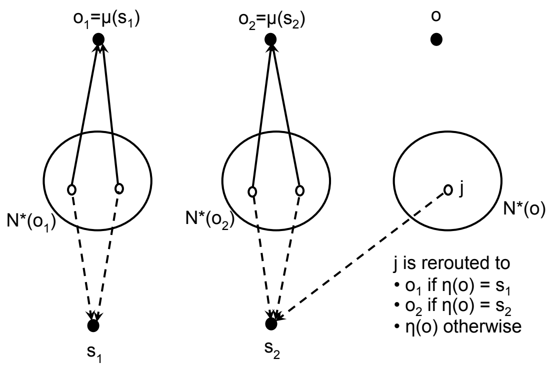


Fig. 3: Rerouting of clients after the swap: delete s_1, s_2 and add $o_1 = \mu(s_1), o_2 = \mu(s_2)$. The dotted and solid lines represent the assignment of clients before and after the swap respectively. A client $j \in N^*(o) \cap [N(s_1) \cup N(s_2)]$ is routed as mentioned depending on the value of $\eta(o)$.

Now consider a client $j \in [N(s_1) \cup N(s_2)] \setminus [N^*(o_1) \cup N^*(o_2)]$. Assume without loss of generality that $j \in N(s_1)$; a similar argument also holds for the case $j \in N(s_2)$. Let $o(j)$ be the facility that serves j in O. If $\eta(o(j)) = s_1$, then j is rerouted to o_1 and the increase in service cost is $d(j, o_1) - d(j, s_1) \le d(s_1, o_1) \le d(s_1, o(j)) \le S_j + O_j$. This sequence of inequalities follows from repeated use of triangle inequality and from the fact $o_1 = \mu(s_1)$. If $\eta(o(j)) = s_2$, then it is rerouted to o_2 and the increase in service cost is $d(j, o_2) - S_j \le d(j, o(j)) + d(o(j), s_2) + d(s_2, o_2) - S_j \le d(j, o(j)) + d(o(j), s_2) + d(s_2, o(j)) - S_j \le d(j, o(j)) + d(o(j), s_1) + d(s_1, o(j)) - S_j \le O_j + 2(O_j + S_j) - S_j = 3O_j + S_j$. This sequence of inequalities follows from repeated use of triangle inequality and from the fact $o_2 = \mu(s_2)$ and $\eta(o(j)) = s_2$. Now consider the case that $\eta(o(j))$ is neither s_1 or s_2. Let $s(j)$ denote the facility that serves j in S. We reroute j to $\eta(o(j))$ and the increase in service cost is thus $d(j, \eta(o(j))) - S_j \le O_j + d(o(j), \eta(o(j))) - S_j \le O_j + d(o(j), s(j)) - S_j \le O_j + O_j + S_j - S_j = 2O_j$. This proves item 4. □

2.3 Putting everything together

Intuition. Note that inequality (1) has "$-S_j$" terms for some clients and "$+O_j$" terms for some clients. The analysis of Arya et al. [3] or Gupta and Tangwongsan [17] is based on adding several inequalities of this type so that the "$-S_j$" term is included for each client j once and "$+O_j$" term is included for each client j at most 5 times. Thus overall, they get $-\sum_j S_j + 5\sum_j O_j \geq 0$. This directly gives a 5-approximation. Unfortunately, such an analysis does not work in our setting. We also have to add several inequalities (2)-(4), thus incurring "$+S_j$" terms for some clients. We then use inequality (1) repeatedly to "cancel" the "$+S_j$" terms in order to prove a constant approximation. All the swaps to be considered are contained in a block. For `block`-i, we prove the following inequality:

$$\sum_{j \in N^*(R_i^* \cup B_i^*)} S_j \leq O(1) \cdot \left[\sum_{j \in N^*(R_i^* \cup B_i^*)} O_j + \sum_{j \in N(R_i \cup B_i)} O_j \right]. \tag{5}$$

Adding these inequalities over all the blocks, we get a constant approximation.

$$\begin{aligned} \mathbf{cost}(S) &= \sum_{i=1}^{t} \sum_{j \in N^*(R_i^* \cup B_i^*)} S_j \leq O(1) \cdot \sum_{i=1}^{t} \left[\sum_{j \in N^*(R_i^* \cup B_i^*)} O_j + \sum_{j \in N(R_i \cup B_i)} O_j \right] \\ &\leq O(1) \cdot 2 \cdot \mathbf{cost}(O). \end{aligned}$$

We start by an example to illustrate how to prove it using the example in Figure 1.

We start with some notation. If R_1 has at least one good or very good facility, we fix a function $\mathtt{g} : R_1^* \to R_1$ such that each facility in $\mathtt{g}(R_1^*)$ is either good or very good and $|\mathtt{g}^{-1}(r)| \leq 2$ for all $r \in R_1$. It is easy to see that such a function exists. Similarly, if B_1 has at least one good or very good facility, we fix a function $\mathtt{g} : B_1^* \to B_1$ such that each facility in $\mathtt{g}(B_1^*)$ is either good or very good and $|\mathtt{g}^{-1}(b)| \leq 2$ for all $b \in B_1$. These functions $\mathtt{g}$ are used to decide which facilities in $R_1^* \cup B_1^*$ are swapped with which facilities in $R_1 \cup B_1$. A facility $r^* \in R_1^*$ will be swapped with $\mathtt{g}(r^*)$ and a facility $b^* \in B_1^*$ will be swapped with $\mathtt{g}(b^*)$.

Example in Figure 1. To convey our intuition, we prove inequality (5) for the example of `block`-1 in Figure 1. For concreteness, assume that the function μ is given by $r_5 \mapsto b_1^*, b_1 \mapsto r_4^*, b_2 \mapsto r_5^*$. Also assume that $\mathtt{g}$ is given by $r_1^* \mapsto r_1, r_2^* \mapsto r_2, r_3^* \mapsto r_3, r_4^* \mapsto r_4, r_5^* \mapsto r_4, b_1^* \mapsto b_1, b_2^* \mapsto b_2, b_3^* \mapsto b_3$. To obtain "$-S_j$" terms for clients in $N^*(B_1^*)$, we consider the following swaps and the corresponding inequalities:

- $\mathtt{swap}(\mathtt{g}(\mu(\mathtt{g}(b_1^*))), \mu(\mathtt{g}(b_1^*)) \mid \mathtt{g}(b_1^*), b_1^*)$ which is same as $\mathtt{swap}(r_4, r_4^* \mid b_1, b_1^*)$ (consider inequality (3)),
- $\mathtt{swap}(\mathtt{g}(\mu(\mathtt{g}(b_2^*))), \mu(\mathtt{g}(b_2^*)) \mid \mathtt{g}(b_2^*), b_2^*)$ which is same as $\mathtt{swap}(r_4, r_5^* \mid b_2, b_2^*)$ (consider inequality (3)),
- $\mathtt{swap}(\mathtt{g}(b_3^*), b_3^*)$ which is same as $\mathtt{swap}(b_3, b_3^*)$ (consider inequality (1)).

If we add these three inequalities, we get

$$\sum_{j \in N^*(\{r_4^*, b_1^*, r_5^*, b_2^*, b_3^*\})} (O_j - S_j) + \sum_{j \in N^*(r_3^*)} (3O_j + S_j) + \sum_{j \in N(\{b_3, r_4, b_1\})} 2O_j + \sum_{j \in N(\{r_4, b_2\})} 2O_j \geq 0. \tag{6}$$

We next consider the following swaps and the corresponding inequalities:

- $\mathtt{swap}(\mathtt{g}(r_1^*), r_1^*)$ which is same as $\mathtt{swap}(r_1, r_1^*)$ (consider inequality (1)),
- $\mathtt{swap}(\mathtt{g}(r_2^*), r_2^*)$ which is same as $\mathtt{swap}(r_2, r_2^*)$ (consider inequality (1)),

– $\texttt{swap}(\texttt{g}(r_3^*), r_3^*)$ which is same as $\texttt{swap}(r_3, r_3^*)$ (consider inequality (1)). We in fact multiply this inequality by factor 2 in order to cancel the "$+S_j$" term in the second term of (6) above.

Note that the leader r_5 is not involved in any of the considered swaps. Adding these three inequalities, we get

$$\sum_{j\in N^*(\{r_1^*,r_2^*\})}(O_j - S_j) + 2\sum_{j\in N^*(r_3^*)}(O_j - S_j) + \sum_{j\in N(\{r_1,r_2\})}2O_j + 2\sum_{j\in N(r_3)}2O_j \geq 0. \quad (7)$$

Adding (6) and (7), we get our desired inequality

$$\sum_{j\in N^*(R_1^*\cup B_1^*)} S_j \leq 5\sum_{j\in N^*(R_1^*\cup B_1^*)} O_j + 4\sum_{j\in N(R_1\cup B_1)} O_j.$$

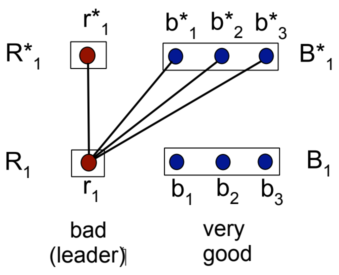
Proof of Theorem 1. We now discuss the general case and prove Theorem 1 for $\texttt{block}$-1. Similar swaps are defined for all the other blocks. We handle the following four cases separately.

1. $|R_1| \leq 1$ and $|B_1| \leq 1$.
2. ($|R_1| \geq 2$ or $|B_1| \geq 2$) and there is a red leader $\ell \in R_1$ and condition 4(a) in Lemma 1 holds (or there is a blue leader $l \in B_1$ and condition 4(b) in Lemma 1 holds).
3. ($|R_1| \geq 2$ or $|B_1| \geq 2$) and there is a blue leader $\ell \in B_1$ and condition 4(a) in Lemma 1 holds (or there is a red leader $l \in R_1$ and condition 4(b) in Lemma 1 holds).
4. ($|R_1| \geq 2$ or $|B_1| \geq 2$) and there is no leader in $R_1 \cup B_1$.

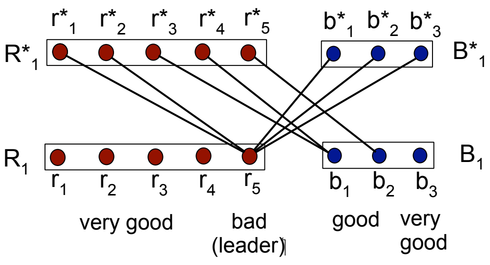
Case 1. $|R_1| \leq 1$ and $|B_1| \leq 1$. If $|R_1 \cup B_1| = 1$, we consider $\texttt{swap}(s, o)$ where $s \in R_1 \cup B_1$ and $o \in R_1^* \cup B_1^*$ and consider inequality (2). Since $\eta^{-1}(s)\setminus\{o\} = \emptyset$, we do not incur $+S_j$ term for any client j. Thus we have $\sum_{j\in N^*(o)}(O_j - S_j) + \sum_{j\in N(s)\setminus N^*(o)} 2O_j \geq 0$. If $|R_1 \cup B_1| = 2$, we consider $\texttt{swap}(r, \mu(b) \mid b, \mu(r))$ where $r \in R_1$ and $b \in B_1$ and consider inequality (4). Again we do not incur $+S_j$ term for any client j. Thus we have $\sum_{j\in N^*(\{\mu(r),\mu(b)\})}(O_j - S_j) + \sum_{j\in N(\{r,b\})\setminus N^*(\{\mu(r),\mu(b)\})} 2O_j \geq 0$.

Case 2. There is a red leader $\ell \in R_1$ and condition 4(a) in Lemma 1 holds. We consider two subcases: case (a) $|R_1^*| = |R_1| = 1$, and case (b) $|R_1^*| = |R_1| > 1$. See Figures 4(a) and 4(b) for examples of subcases (a) and (b) respectively.

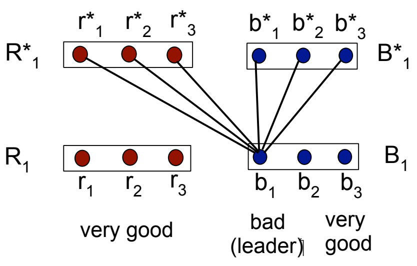
In case (a), we have $|B_1^*| = |B_1| \geq 2$. In this case, all the facilities in B_1 are very good. Indeed if we had included a good blue facility in B_1, then $|R_1|$ would have been more than 1. If ℓ and $\mu(\ell)$ have the same color (say, red), we consider swap $\texttt{swap}(\ell, \mu(\ell))$ and add inequality (2). Thus we get one $-S_j$ term for clients $j \in N^*(\mu(\ell))$ and one $+S_j$ term for clients $j \in N(\ell) \cap N^*(B_1^*)$. We also get at most two $+O_j$ terms for clients $j \in N(\ell)$. If ℓ and $\mu(\ell)$ have different colors, however, we consider $\texttt{swap}(\ell, r^* \mid \texttt{g}(\mu(\ell)), \mu(\ell))$ where $r^* \in R_1^*$. Note that $\texttt{g}(\mu(\ell))$ is very good. We can therefore pick $r^* \in R_1^*$ arbitrarily and add the corresponding inequality (3). Thus we get one $-S_j$ term for clients $j \in N^*(\{r^*, \mu(\ell)\})$ and one $+S_j$ term for clients $j \in N(\ell) \cap N^*(B_1^* \setminus \{\mu(\ell)\})$. We also get at most three $+O_j$ terms for clients $j \in N(\{\ell, b\})$. We next consider $\texttt{swap}(\texttt{g}(b^*), b^*)$ for all $b^* \in B_1^*$, and add corresponding inequalities (1), each *multiplied by 2*. Thus we get two $-S_j$ terms for clients $j \in N^*(B_1^*)$ and at most constant number of $+O_j$ terms for clients $j \in N(B_1)$. Overall, by adding



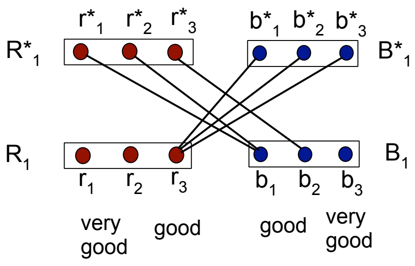
(a) Case 2(a): there is a red leader $r_1 \in R_1$, condition 4(a) in Lemma 1 holds and $|R_1^*| = |R_1| = 1$.



(b) Case 2(b): there is a red leader $r_5 \in R_1$, condition 4(a) in Lemma 1 holds and $|R_1^*| = |R_1| > 1$.



(c) Case 3: there is a blue leader $b_1 \in B_1$ and condition 4(a) in Lemma 1 holds.



(d) Case 4: there is no bad leader in $R_1 \cup B_1$ and $|R_1| \geq 2$ or $|B_1| \geq 2$.

Fig. 4: Examples of different cases considered in the Proof of Theorem 1.

these inequalities, we get $-S_j$ terms for all $j \in N^*(R_1^* \cup B_1^*)$ and constant number of terms $+O_j$ for all $j \in N^*(R_1^* \cup B_1^*) \cup N(R_1 \cup B_1)$.

Now consider case (b). Similar to case (a) above, if ℓ and $\mu(\ell)$ have the same color (say, red), we consider $\texttt{swap}(\ell, \mu(\ell))$ and add the corresponding inequality (2). If ℓ and $\mu(\ell)$ have different colors, however, we consider $\texttt{swap}(\ell, r^* \mid \texttt{g}(\mu(\ell)), \mu(\ell))$ where $r^* \in R_1^*$. If $\texttt{g}(\mu(\ell))$ is good, we let $r^* = \mu(\texttt{g}(\mu(\ell)))$ and add inequality (4), else we pick $r^* \in R_1^*$ arbitrarily and add the corresponding inequality (3). For each $b^* \in B_1^*$, we consider the following. If $\texttt{g}(b^*)$ is very good, we consider $\texttt{swap}(\texttt{g}(b^*), b^*)$ and add corresponding inequality (1). If $\texttt{g}(b^*)$ is good, we consider $\texttt{swap}(\texttt{g}(\mu(\texttt{g}(b^*))), \mu(\texttt{g}(b^*)) \mid \texttt{g}(b^*), b^*)$ and add corresponding inequality (3). For each $r^* \in R_1^*$, since $\texttt{g}(r^*)$ is very good, we consider $\texttt{swap}(\texttt{g}(r^*), r^*)$ and add corresponding inequalities (1), each *multiplied by 2*. Overall, by adding these inequalities, we get $-S_j$ terms for all $j \in N^*(R_1^* \cup B_1^*)$ and constant number of terms $+O_j$ for all $j \in N^*(R_1^* \cup B_1^*) \cup N(R_1 \cup B_1)$.

The case when there is a blue leader $\ell \in B_1$ and condition 4(b) in Lemma 1 holds is similar (by interchanging the roles of red and blue).

Case 3. There is a blue leader $\ell \in B_1$ and condition 4(a) in Lemma 1 holds. See Figure 4(c) for an example this case. Similar to case 2(a) above, depending on whether ℓ and $\mu(\ell)$ have the same or different colors, we consider swaps involving ℓ and $\mu(\ell)$ and add the corresponding inequalities. If $|B_1^*| = |B_1| > 1$, we also consider the following swaps. For each $b^* \in B_1^*$, we consider the following. If $\texttt{g}(b^*)$ is very good, we consider $\texttt{swap}(\texttt{g}(b^*), b^*)$ and add corresponding inequality (1). If $\texttt{g}(b^*)$ is good, we consider $\texttt{swap}(\texttt{g}(\mu(\texttt{g}(b^*))), \mu(\texttt{g}(b^*)) \mid \texttt{g}(b^*), b^*)$ and add corresponding inequalities (3). For each

$r^* \in R_1^*$, since $\mathtt{g}(r^*)$ is very good, we consider $\mathtt{swap}(\mathtt{g}(r^*), r^*)$ and add corresponding inequalities (1), each *multiplied by 2.* Overall, by adding these inequalities, we get $-S_j$ term for all $j \in N^*(R_1^* \cup B_1^*)$ and constant number of terms $+O_j$ for all $j \in N^*(R_1^* \cup B_1^*) \cup N(R_1 \cup B_1)$.

The case when there is a red leader $\ell \in R_1$ and condition 4(b) in Lemma 1 holds is similar (by interchanging the roles of red and blue).

Case 4. There is no bad leader in $R_1 \cup B_1$ and $|R_1| \geq 2$ or $|B_1| \geq 2$. See Figure 4(d) for an example this case. We first argue that there cannot exist good facilities $r \in R_1$ and $b \in B_1$ such that $|\eta^{-1}(r)| = |\eta^{-1}(b)| = 1$. If this condition was true, this block would have formed in step 3 of the Procedure in Figure 2 and this case would not arise. Without loss of generality we assume that there is no good facility $r \in R_1$ such that $|\eta^{-1}(r)| = 1$. We now prove the following claim that there is a significant fraction of very good facilities in either R_1 or B_1.

Claim Either the number of very good facilities in R_1 is at least $|R_1|/4$ or the number of very good facilities in B_1 is at least $|B_1|/3$.

Proof Since each good facility $r \in R_1$ has $\eta^{-1}(r) \subseteq B_1^*$ and $|\eta^{-1}(r)| \geq 2$, the number of good facilities in R_1 is at most $|B_1^*|/2 = |B_1|/2$. Now if the number of very good facilities in R_1 is less than $|R_1|/4$, then the number of good facilities in R_1 is more than $3|R_1|/4$. Thus we get $|B_1|/2 > 3|R_1|/4$, which implies $|B_1| > 3|R_1|/2$. Since each good facility $s \in B_1$ has $\eta^{-1}(s) \subseteq R_1^*$ and $|\eta^{-1}(s)| \geq 1$, the maximum number of good facilities in B_1 is at most $|R_1^*| = |R_1| < 2|B_1|/3$. Therefore the number of very good facilities in B_1 is at least $|B_1|/3$. $\square$

Let us consider the case when the number of very good facilities in R_1 is at least $|R_1|/4$. The argument for the other case is very similar and is omitted. We now modify the function $\mathtt{g} : R_1^* \to R_1$ so that each facility in $\mathtt{g}(R_1^*)$ is *very good* and $|\mathtt{g}^{-1}(r)| \leq 4$ for any $r \in R_1$. There exists such a function since there are enough number of very good facilities in R_1. Now the swaps considered for this case are very similar to those in case 2(b).

For each good facility $r \in R_1$, we consider $\mathtt{swap}(r, r^* \mid \mathtt{g}(\mu(r)), \mu(r))$ where $r^* \in R_1^*$. If $\mathtt{g}(\mu(r))$ is good, we let $r^* = \mu(\mathtt{g}(\mu(r)))$ and add inequality (4), else we pick $r^* \in R_1^*$ arbitrarily and add the corresponding inequality (3). For each $b^* \in B_1^*$, we consider the following. If $\mathtt{g}(b^*)$ is very good, we consider $\mathtt{swap}(\mathtt{g}(b^*), b^*)$ and add corresponding inequality (1). If $\mathtt{g}(b^*)$ is good, we consider $\mathtt{swap}(\mathtt{g}(\mu(\mathtt{g}(b^*))), \mu(\mathtt{g}(b^*)) \mid \mathtt{g}(b^*), b^*)$ and add corresponding inequality (3). For each $r^* \in R_1^*$, since $\mathtt{g}(r^*)$ is very good from our modification to the function $\mathtt{g}$, we consider $\mathtt{swap}(\mathtt{g}(r^*), r^*)$ and add corresponding inequalities (1). Note that since $|\mathtt{g}^{-1}(r)| \leq 4$ for all $r \in R_1$, we incur a constant number $+O_j$ terms for $j \in N(r)$ for $r \in R_1$. Overall, by adding these inequalities, with appropriate $O(1)$ multipliers, we get $-S_j$ term for all $j \in N^*(R_1^* \cup B_1^*)$ and constant number of terms $+O_j$ for all $j \in N^*(R_1^* \cup B_1^*) \cup N(R_1 \cup B_1)$.

This ends the proof of Theorem 1.

3 Proof of Theorem 2

In this section, we outline the proof of Theorem 2. We consider the multi-swap local search algorithm of Arya et al. [3]: start with any k facilities in the solution S and output a local optimal solution w.r.t. the following q-swap operation: delete q facilities from S and add q facilities in $\mathcal{F} \setminus S$ to S. We use a notation similar to the previous section. In addition, let $P \subseteq \mathcal{C}$ denote the set of clients that pay penalty in the locally optimal solution S and let $P^* \subseteq \mathcal{C}$ denote the set of clients that pay penalty in the optimal solution O. We prove the following theorem which implies that S is a $(3+2/q)$-approximation.

Theorem 3

$$\sum_{j \notin P} S_j + \sum_{j \in P} p_j \leq \left(3 + \frac{2}{q}\right) \sum_{j \notin P^*} O_j + \left(1 + \frac{1}{q}\right) \sum_{j \in P^*} p_j.$$

Our proof is based on the simplified proof, given by Gupta and Tangwongsan [17], of the multi-swap k-median algorithm. We partition the set S (resp. O) into S_i (resp. O_i) for $i = 1, \ldots, t$ as follows. Let $s_1 \in S$ any facility with $\eta^{-1}(s_1) \neq \emptyset$. Let $O_1 = \eta^{-1}(s_1)$ and let S_1 be s_1 along with arbitrary $|\eta^{-1}(s_1)| - 1$ facilities $s \in S$ such that $\eta^{-1}(s) = \emptyset$. Recurse on $S \setminus S_1$ and $O \setminus O_1$ for $i > 1$. Note that even if the multiplier of $\sum_{j \in P^*} p_j$ on the right is $(1 + 1/q)$ instead of 1, one may use the above result, as a subroutine, in the algorithm for the robust k-median problem [10]. This is a version of the k-median problem in which at most l clients may be left unserved. We obtain a solution which has number of outliers at most $l(1+\epsilon)(1+\gamma)$ and has cost at most $(3+\epsilon)(1+1/\gamma)$ for any fixed $\epsilon, \gamma > 0$.

We now prove Theorem 3 using the following two lemmas.

Lemma 3 *Fix i such that $|S_i| = |O_i| \leq q$. We have*

$$\sum_{j \in N^*(O_i) \setminus P} (O_j - S_j) + \sum_{j \in N^*(O_i) \cap P} (O_j - p_j) + \sum_{j \in N(S_i) \cap P^*} (p_j - S_j) + \sum_{j \in N(S_i) \setminus P^*} 2O_j \geq 0.$$

Proof Since $|S_i| = |O_i| \leq q$, we can consider the swap: delete facilities in S_i and add facilities in O_i. Since S is a locally optimal solution, we have $\mathbf{cost}((S \setminus S_i) \cup O_i) - \mathbf{cost}(S) \geq 0$. We now given an upper bound on the LHS of this inequality by giving a rerouting of the clients. A client $j \in N^*(O_i)$ is rerouted to the facility that serves j in solution O. The increase in its cost contribution is $(O_j - S_j)$ if $j \notin P$ or $(O_j - p_j)$ if $j \in P$. A client $j \in N(S_i) \cap P^*$ now pays penalty. Thus the increase in its cost contribution is $(p_j - S_j)$. A client $j \in N(S_i) \setminus P^*$ is rerouted to $\eta(o(j))$ where $o(j)$ denotes the facility that serves j in solution O. Thus the increase in its cost contribution is $d(j, \eta(o(j))) - S_j \leq O_j + d(o(j), \eta(o(j))) - S_j \leq O_j + d(o(j), s(j)) - S_j \leq O_j + O_j + S_j - S_j = 2O_j$. The remaining clients are not rerouted. This completes the proof. □

Lemma 4 *Fix i such that $|S_i| = |O_i| > q$. We have*

$$\sum_{j \in N^*(O_i) \setminus P} (O_j - S_j) + \sum_{j \in N^*(O_i) \cap P} (O_j - p_j) \\ + \left(1 + \frac{1}{q}\right) \left[\sum_{j \in N(S_i) \cap P^*} (p_j - S_j) + \sum_{j \in N(S_i) \setminus P^*} 2O_j \right] \geq 0.$$

Proof The proof is similar to that of Lemma 3. Let $S'_i = S_i \setminus \{s_i\}$. Note that $|S'_i| = |O_i| - 1 \geq q$. For all $A \subseteq S'_i$ and all $B \subseteq O_i$ such that $|A| = |B| = q$, we consider swap: delete A and add B and associate a weight of

$$\left(\binom{|O_i|-1}{q-1} \cdot \binom{|O_i|-1}{q}\right)^{-1} \tag{8}$$

to this swap. Now for each of these swaps, the inequality

$$\sum_{j \in N^*(B)\setminus P} (O_j - S_j) + \sum_{j \in N^*(B)\cap P} (O_j - p_j) + \sum_{j \in N(A)\cap P^*} (p_j - S_j) + \sum_{j \in N(A)\setminus P^*} 2O_j \geq 0 \tag{9}$$

holds. The proof of this is identical to that of Lemma 3.

We next argue that if we add the inequalities (9) multiplied by their weights, given by (8), over all such swaps, we exactly get the desired inequality. We first make two observations. For any fixed $o \in O_i$, the number of swaps (delete A, add B) such that $A \subseteq S'_i$, $|A| = q$, $B \subseteq O_i$, $|B| = q$ and $o \in B$ is

$$\binom{|O_i|-1}{q} \cdot \binom{|O_i|-1}{q-1}.$$

Therefore the total weight, given by (8), of these swaps is 1. Similarly, for any fixed $s \in S'_i$, the number of swaps (delete A, add B) such that $A \subseteq S'_i$, $|A| = q$, $s \in A$, $B \subseteq O_i$ and $|B| = q$ is

$$\binom{|O_i|-2}{q-1} \cdot \binom{|O_i|}{q}.$$

Therefore the total weight, given by (8), of these swaps is $\frac{|O_i|}{|O_i|-1}$.

Now consider all the swaps (delete A, add B) such that $A \subseteq S'_i$, $|A| = q$, $B \subseteq O_i$ and $|B| = q$. From the two observations made above, we can conclude the following. If we sum the inequalities (9) of these swaps, each multiplied by its weight (8), we get

$$\sum_{j \in N^*(O_i)\setminus P} (O_j - S_j) + \sum_{j \in N^*(O_i)\cap P} (O_j - p_j) \\ + \left(\frac{|O_i|}{|O_i|-1}\right)\left[\sum_{j \in N(S_i)\cap P^*} (p_j - S_j) + \sum_{j \in N(S_i)\setminus P^*} 2O_j\right] \geq 0.$$

Noting that $\frac{|O_i|}{|O_i|-1} \leq 1 + \frac{1}{q}$ and that $p_j \geq S_j$ for all $j \in N(S_i) \cap P^*$, we get the desired inequality in Lemma 4. □

By adding inequalities in Lemmas 3 and 4 for all $i = 1, \ldots, t$ and again using the fact that $p_j \geq S_j$ for all $j \in N(S)$, we get

$$\sum_{j \in N^*(O)\setminus P} (O_j - S_j) + \sum_{j \in N^*(O)\cap P} (O_j - p_j) \\ + \left(1 + \frac{1}{q}\right)\left[\sum_{j \in N(S)\cap P^*} (p_j - S_j) + \sum_{j \in N(S)\setminus P^*} 2O_j\right] \geq 0.$$

Simplifying the above inequality, we get

$$\sum_{\substack{j\in(N^*(O)\setminus P)\\ \cup(N(S)\cap P^*)}} S_j + \sum_{j\in N^*(O)\cap P} p_j$$
$$\leq \sum_{\substack{j\in(N^*(O)\setminus P)\\ \cup(N^*(O)\cap P)}} O_j + \left(1+\frac{1}{q}\right)\left[\sum_{j\in N(S)\cap P^*} p_j + \sum_{j\in N(S)\setminus P^*} 2O_j\right].$$

Adding $\sum_{j\in P\cap P^*} p_j$ to both the sides and simplifying, we get the desired inequality in Theorem 3.

4 The knapsack-median on trees and general graphs

In this section, we sketch how we can solve knapsack-median on trees when opening costs of facilities are polynomially bounded. Without loss of generality by adding dummy nodes and dummy edges of connection costs zero, we can assume that our tree is a binary tree rooted at r, there is no facility or client at r and there is no facility at any leaves. We denote by T_t the subtree of T rooted at node t. Now for each node t, we fill in a table $\mathcal{T}_t[W', f, D']$ which keeps the minimum total cost of serving all clients in T_t except D' of them that we bring up to the root t (and thus we pay the partial connection costs to t for these clients) while the total opening cost of facilities in T_t is W' and the nearest opened facility to t in T_t is f ($f = \perp$ if there is no open facility in T_t). First it is clear that as long as $W \geq W'$ is polynomially bounded the size of table $\mathcal{T}$ and thus our running time is polynomial. The final solution is $\min_{W'\leq W, f\in V(T)} \mathcal{T}_r[W', f, 0]$. For a leaf ℓ, table $\mathcal{T}_\ell$ is filled in with ∞, except when ℓ has D clients in which case $\mathcal{T}_\ell[0, \perp, D] = 0$ and the rest of the table is filled in with ∞. We compute entries in $\mathcal{T}_t$ from those in $\mathcal{T}_{t'}$ and $\mathcal{T}_{t''}$ where t' and t'' are the children of t as follows. First we choose the number of clients in T_t that we want to serve later by bringing them up to t (and thus compute this cost in our final cost). The rest of the clients in $T_{t'}$ (resp., $T_{t''}$) should be satisfied either in $T_{t'}$ (resp., $T_{t''}$), or by the facility in $T_{t''}$ (resp., $T_{t'}$) nearest to t'' (resp., t''), or the facility at t if there exists one. Finally, when we open a facility of opening cost w at t, the total opening cost of facilities in T_t' and $T_{t''}$ should be w less than the budget. We pick the best solution among all possible ways of doing this.

Acknowledgments

We thank anonymous referees for valuable comments that improved the presentation.

References

1. A. Archer, R. Rajagopalan, and D. B. Shmoys. Lagrangian relaxation for the k-median problem: new insights and continuity properties. In *Proceedings of the European Symposium on Algorithms (ESA)*, volume 2832 of *Lecture Notes in Comput. Sci.*, pages 31–42, 2003.
2. S. Arora, P. Raghavan, and S. Rao. Approximation schemes for Euclidean k-medians and related problems. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 106–113, 1998.

3. V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit. Local search heuristics for k-median and facility location problems. *SIAM J. Comput.*, 33(3):544–562, 2004.
4. E. Balas. The prize collecting traveling salesman problem. *Networks*, 19(6):621–636, 1989.
5. Y. Bartal. On approximating arbitrary metrics by tree metrics. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 161–168, 1998.
6. M. Bateni and M. Hajiaghayi. Assignment problem in content distribution networks: unsplittable hard-capacitated facility location. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 805–814, 2009.
7. D. Bienstock, M. X. Goemans, D. Simchi-Levi, and D. Williamson. A note on the prize collecting traveling salesman problem. *Math. Programming*, 59(3, Ser. A):413–420, 1993.
8. M. Charikar and S. Guha. Improved combinatorial algorithms for facility location problems. *SIAM J. Comput.*, 34(4):803–824, 2005.
9. M. Charikar, S. Guha, É. Tardos, and D. Shmoys. A constant-factor approximation algorithm for the k-median problem. *J. Comp. Sys. Sci.*, 65(1):129–149, 2002.
10. M. Charikar, S. Khuller, D. M. Mount, and G. Narasimhan. Algorithms for facility location problems with outliers. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 642–651, 2001.
11. F. A. Chudak and D. P. Williamson. Improved approximation algorithms for capacitated facility location problems. *Math. Program.*, 102(2, Ser. A):207–222, 2005.
12. N. R. Devanur, N. Garg, R. Khandekar, V. Pandit, A. Saberi, and V. V. Vazirani. Price of anarchy, locality gap, and a network service provider game. In *Proceedings of the International Workshop on Internet and Network Economics (WINE)*, pages 1046–1055, 2005.
13. J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. System Sci.*, 69(3):485–497, 2004.
14. U. Feige, V. S. Mirrokni, and J. Vondrak. Maximizing non-monotone submodular functions. In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 461–471, 2007.
15. L. Fleischer, M. X. Goemans, V. S. Mirrokni, and M. Sviridenko. Tight approximation algorithms for maximum general assignment problems. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 611–620, 2006.
16. M. X. Goemans and D. P. Williamson. A general approximation technique for constrained forest problems. *SIAM J. Comput.*, 24(2):296–317, 1995.
17. A. Gupta and K. Tangwongsan. Simpler analyses of local search algorithms for facility location. *ArXiv e-prints*, arXiv:0809.2554, 2008.
18. M. T. Hajiaghayi and K. Jain. The prize-collecting generalized Steiner tree problem via a new approach of primal-dual schema. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 631–640, 2006.
19. K. Jain. Private communication, 2009.
20. K. Jain, M. Mahdian, and A. Saberi. A new greedy approach for facility location problems. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 731–740, 2002.
21. K. Jain and V. V. Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation. *J. ACM*, 48(2):274–296, 2001.
22. S. G. Kolliopoulos and S. Rao. A nearly linear-time approximation scheme for the Euclidean k-median problem. *SIAM J. Comput.*, 37(3):757–782, 2007.
23. M. R. Korupolu, C. G. Plaxton, and R. Rajaraman. Analysis of a local search heuristic for facility location problems. *J. Algorithms*, 37(1):146–188, 2000.
24. R. Krishnaswamy, A. Kumar, V. Nagarajan, Y. Sabharwal, and B. Saha. The matroid median problem. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2011.
25. A. Kuehn and M. Hamburger. A heuristic program for locating warehouses. *Management Science*, 9:643–666, 1963.
26. J.-H. Lin and J. S. Vitter. Approximation algorithms for geometric median problems. *Inf. Process. Lett.*, 44(5):245–249, 1992.
27. M. Mahdian and M. Pál. Universal facility location. In *Proceedings of the European Symposium on Algorithms (ESA)*, volume 2832 of *Lecture Notes in Comput. Sci.*, pages 409–421, 2003.

28. N. Megiddo and K. J. Supowit. On the complexity of some common geometric location problems. *SIAM J. Comput.*, 13(1):182–196, 1984.
29. A. Meyerson and B. Tagiku. Minimizing average shortest path distances via shortcut edge addition. In *Proceedings of the International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, pages 272–285, 2009.
30. M. Pál, É. Tardos, and T. Wexler. Facility location with nonuniform hard capacities. In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 329–338, 2001.
31. J. Zhang, B. Chen, and Y. Ye. A multi-exchange local search algorithm for the capacitated facility location problem. In *Proceedings of the International Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 219–233, 2004.