An Improved Approximation Algorithm for the Prize-Collecting Red-Blue Median Problem

Zhen Zhang(✉), Yutian Guo, and Junyu Huang

School of Computer Science and Engineering, Central South University, Changsha 410083, People's Republic of China
csuzz@foxmail.com

Abstract. The *red-blue median* problem considers a set of *red* facilities, a set of *blue* facilities, and a set of clients located in some metric space. The goal is to open k_r red facilities and k_b blue facilities such that the sum of the distance from each client to its nearest opened facility is minimized, where k_r, $k_b \geq 0$ are two given integers. Designing approximation algorithms for this problem remains an active area of research due to its applications in various fields. However, in many applications, the existence of noisy data poses a big challenge for the problem. In this paper, we consider the *prize-collecting red-blue median* problem, where the noisy data can be removed by paying a penalty cost. The current best approximation for the problem is a ratio of 24, which was obtained by LP-rounding. We deal with this problem using a local search algorithm. We construct a layered structure of the swap pairs, which yields a $(9 + \epsilon)$-approximation for the prize-collecting red-blue median problem. Our techniques generalize to a more general *prize-collecting τ-color median* problem, where the facilities have τ different types, and give a $(4\tau + 1 + \epsilon)$-approximation for the problem for the case where τ is a constant.

Keywords: Clustering · Approximation · Local search

1 Introduction

k-median is a widely studied clustering problem and finds applications in many fields related to unsupervised learning. Given a set $\mathcal{D}$ of clients and a set $\mathcal{F}$ of facilities in a metric space, the k-median problem is to open k facilities such that the sum of the distance from each client to its nearest opened facility is minimized.

In many applications, the clustering problem has different types of facilities and upper bound on the number of the opened facilities of each type. One such

This work was supported by National Natural Science Foundation of China (61672536, 61872450, 61828205, and 61802441), Hunan Provincial Key Lab on Bioinformatics, and Hunan Provincial Science and Technology Program (2018WK4001).

J. Chen et al. (Eds.): TAMC 2020, LNCS 12337, pp. 94–106, 2020.
https://doi.org/10.1007/978-3-030-59267-7_9

example is in the design of Content Distribution Networks [2], where a set of clients need to be connected to a set of servers with a few different types, and there is a budget constraint on the number of the arranged severs of each type. Motivated by such applications, Hajiaghayi *et al.* [9] introduced the *red-blue median* problem which involves two facility-types. They showed that local search yields a constant factor approximation for the problem. The current best approximation for the red-blue median problem is a ratio of $5+\epsilon$ due to Friggstad and Zhang [8]. Inspired by the work on the red-blue median problem, Krishnaswamy *et al.* [12] introduced a more general *matroid median* problem, where the set of facilities has a matroid structure, and the set of the opened facilities should be an independent set in the matroid. The matroid median problem does not only generalize the red-blue median problem, but also the *τ-color median* problem where more than two facility-types are considered. Krishnaswamy *et al.* [12] gave a 16-approximation for the matroid median problem by an LP-rounding technique. The approximation guarantee was later improved by a series of work [4,14] to the current best ratio of $7.081+\epsilon$ [13].

Although red-blue median and its related clustering problems have been extensively studied, algorithms developed for these problems could significantly deteriorate their performance when applied to real-world data. One reason is that these problems implicitly assume that all clients can be clustered into several distinct groups. However, real-world data are often contaminated with various types of noises, which need to be excluded from the solution [3,6,13]. To deal with such noisy data, Charikar *et al.* [3] introduced the problem of *prize-collecting clustering*. The problem is the same as the standard clustering problem, except that we can remove a set of distant clients and pay their penalty costs instead. By discarding the distant clients, one could significantly reduce the clustering cost and thus improve the quality of solution.

Hajiaghayi *et al.* [9] gave an $O(1)$-approximation for the prize-collecting red-blue median problem by local search technique. The approximation ratio is implicit but can be easily shown not better than 30. Krishnaswamy *et al.* [12] later gave a 360-approximation algorithm for the prize-collecting red-blue median problem based on an LP-rounding technique. They showed that the guarantee of 360-approximation generalizes to the prize-collecting matroid median problem. The approximation ratio was recently improved to 24 by a novel rounding procedure [14]. This is also the current best approximation guarantee for both the problems of prize-collecting red-blue median and prize-collecting τ-color median. The noisy data appear frequently in the clustering problems, and the prize-collecting versions of other clustering problems have also been extensively studied [5,7,11,15].

We curtly remark on the commonly used approaches for clustering to show the obstacles in obtaining better ratios than 24 for the problems of prize-collecting red-blue median and prize-collecting τ-color median.

1). Hajiaghayi *et al.* [9] gave a constant factor approximation for the prize-collecting red-blue median problem by local search. Their analysis is based on a technique for dividing the facilities into *blocks* with certain properties.

This provides a clear way to get the approximation guarantee for the local search algorithm. However, getting such well structured blocks induces a large approximation ratio for the problem. It seems quite difficult to apply the technique given in [9] to beat the 24-approximation. Moreover, the method for constructing blocks relies heavily on the fact that the facilities have no more than two different types, which cannot be applied to the τ-color median problem. Indeed, it is still an open problem that whether local search works for the τ-color median problem for the case where τ is a constant, see discussions in [8,10].

2). LP-rounding has been shown to be an effective technique for the problems of red-blue median and τ-color median [4,12–14]. However, it was known that the existence of the penalized clients has a strong impact on the performance of the algorithms based on LP-rounding [12,14]. For instance, the standard LP relaxation of the red-blue median problem has a variable x_{ij} associated with each facility i and client j, which indicates that whether j is connected to i. A constraint $\sum_i x_{ij} = 1$ is given for each client j to ensure that j is connected to a facility. Unfortunately, in the prize-collecting red-blue median problem, the sum $\sum_i x_{ij}$ is not guaranteed to be an integer since not all the clients should be connected. It is unclear whether the LP-rounding approach for clustering with outliers given in [13] can be adapted to clustering with penalties and beat the 24 approximation ratio.

3). The technique of primal-dual has been widely applied for the problem of prize-collecting clustering [3,7,11]. However, it is difficult to use this technique to deal with the red-blue median problem, as discussed in [9,10]. This is further compounded in the prize-collecting red-blue median problem since there is an additional task of identifying the penalized clients.

1.1 Our Results

We use a local search algorithm to deal with the prize-collecting τ-color median problem. Starting with an arbitrary feasible solution, the algorithm tries to swap no more than $O(\tau)$ facilities of each type. It terminates if no such swap yields an improved solution. Otherwise, it iterates with the improved solution. The solution given by the algorithm is called *local optimum*.

Theorem 1. *The local optimum for the prize-collecting τ-color median problem is a $(4\tau + 1)$-approximation solution.*

On the basis of standard techniques [1] (which is curtly described in Sect. 2), the runtime of the local search algorithm can be polynomially bounded for the case where τ is a constant, which induces an arbitrarily small loss in the approximation ratio.

Theorem 2. *For any $\epsilon > 0$, there is a $(4\tau + 1 + \epsilon)$-approximation algorithm for the prize-collecting τ-color median problem that runs in polynomial time for the case where τ is a constant.*

Theorem 2 implies a $(9+\epsilon)$-approximation for the prize-collecting red-blue median problem, which improves the previous best approximation ratio of 24 given by Swamy [14]. Note that for the prize-collecting τ-color median problem, we only obtain improved approximation guarantee for the case where $\tau \leq 5$, and the ratio of 24 given in [14] is still the best guarantee for the general prize-collecting τ-color median problem. Indeed, Krishnaswamy *et al.* [12] showed that local search cannot yield constant factor approximation for the τ-color median problem with polynomial time. However, this negative result does not rule out the possibility of obtaining a local search-based $O(1)$-approximation for the problem in polynomial time for the case where τ is a constant. Theorem 2 shows that local search actually yields an $O(1)$-approximation for this special case.

1.2 Our Techniques

The local search algorithms are commonly analyzed by considering a set of swap pairs where some facilities in the local optimum are swapped with some facilities from an optimal solution. The desired approximation guarantee is obtained by the fact that no such swap pair can improve the local optimum. However, in the prize-collecting τ-color median problem, the swap pairs may violate the constraint on the number of the opened facilities of each type. For instance, after closing a red facility and opening a blue facility, we are forced to swapping another pair of facilities to balance the number of the facilities of each type. This makes the analysis of the cost induced by the local optimum much more complex.

For each to-be-clustered client j, let O_j and S_j be the costs of j induced by the local optimum and optimal solution, respectively. Our analysis starts with carefully constructing a set of *feasible swap pairs* with some special properties (see Sect. 3.1). By estimating the increased cost induced by the constructed swap pairs, we obtain a set of inequalities that involve some "$+O_j$"terms, some "$-S_j$" terms, and some "$+S_j$" terms for each client j (see Sect. 3.2). We want to add these inequalities together to get $O(1)\sum_j O_j - O(1)\sum_j S_j \geq 0$, based on which the desired approximation ratio can be obtained. The challenge is how to eliminate each "$+S_j$" term. To overcome this challenge, we prove the existence of a layered structure of our constructed swap pairs (see Sect. 3.3). It is shown that for each swap pair, the "$+S_j$" terms induced by it can be counteracted by repeatedly using the swap pairs in the lower layers. These ideas lead to the proof of the $(4\tau+1+\epsilon)$-approximation ratio.

2 Preliminaries

The prize-collecting τ-color median problem can be defined as follows.

Definition 1 (prize-collecting τ-color median). *Given a set $\mathcal{C}$ of clients and τ disjoint sets $\mathcal{F}^1, \ldots, \mathcal{F}^\tau$ of facilities in a metric space, τ positive integers*

$k_1, \ldots, k_\tau$, and a penalty function p defined over the clients in $\mathcal{C}$, where $p(j) \geq 0$ for each $j \in \mathcal{C}$, the goal is to identify a subset $\mathcal{S}^t \subseteq \mathcal{F}^t$ of no more than k_t facilities for each $t \in \{1, \ldots, \tau\}$, such that the objective function

$$\sum_{j \in \mathcal{C}} \min\{d(j, \bigcup_{1 \leq t \leq \tau} \mathcal{S}^t), p(j)\}$$

is minimized, where $d(j, \bigcup_{1 \leq t \leq \tau} \mathcal{S}^t)$ denotes the distance from j to its nearest facility in $\bigcup_{1 \leq t \leq \tau} \mathcal{S}^t$.

The special case of $\tau = 2$ corresponds to the prize-collecting red-blue median problem. Given τ sets $\mathcal{S}^1, \ldots, \mathcal{S}^\tau$ of facilities, where $\mathcal{S}^t \subseteq \mathcal{F}^t$ for each $1 \leq t \leq \tau$, we call $\mathbb{S} = (\mathcal{S}^1, \ldots, \mathcal{S}^\tau)$ a feasible solution if $\left|\mathcal{S}^t\right| = k_t$ for each $1 \leq t \leq \tau$, and let $\Phi(\mathbb{S}) = \sum_{j \in \mathcal{C}} \min\{d(j, \bigcup_{1 \leq t \leq \tau} \mathcal{S}^t), p(j)\}$ denote its cost. Let OPT denote the cost of an optimal solution. The local search algorithm for the problem is described in Algorithm 1.

Algorithm 1: Local search for the prize-collecting τ-color median problem

Input: An instance $(\mathcal{C}, \mathcal{F}^1, \ldots, \mathcal{F}^\tau, k_1, \ldots, k_\tau, p)$ of the prize-collecting τ-color median problem;
Output: A local optimum $\mathbb{S} = (\mathcal{S}^1, \ldots, \mathcal{S}^\tau)$;
1 Let $\mathbb{S} = (\mathcal{S}^1, \ldots, \mathcal{S}^\tau)$ be an arbitrary feasible solution;
2 **while** *there exists a feasible solution* $\widetilde{\mathbb{S}} = (\widetilde{\mathcal{S}}^1, \ldots, \widetilde{\mathcal{S}}^\tau)$ *such that* $\left|\widetilde{\mathcal{S}}^t - \mathcal{S}^t\right| \leq 2\tau$ *for each* $1 \leq t \leq \tau$ *and* $\Phi(\widetilde{\mathbb{S}}) < \Phi(\mathbb{S})$ **do**
3 $\quad \mathbb{S} \Leftarrow \widetilde{\mathbb{S}}$;
4 **return** $\mathbb{S}$.

Each iteration of Algorithm 1 takes $O(\left|\mathcal{C}\right| \prod_{1 \leq t \leq \tau} (\left|\mathcal{F}^t\right| k_t)^{2\tau})$ time, which is polynomial in the input size for the case where τ is a constant. However, it may be the case that the number of the iterations exponentially depends on the input size. We can use a well-known trick to ensure that the algorithm terminates in a polynomial number of steps. The idea is to execute a swap only if $\Phi(\widetilde{\mathbb{S}}) \leq (1 - \frac{\epsilon}{\Delta})\Phi(\mathbb{S})$, where the value of Δ is polynomial in the input size. Our analysis is compatible with this trick: it can be verified that the total weight of all the inequalities we consider is polynomially bounded. See [1] for details of the trick.

3 Analysis

We introduce some notations to help analyze Algorithm 1. Let $\mathcal{F} = \bigcup_{1 \leq t \leq \tau} \mathcal{F}^t$. Let $d(i, j)$ denote the distance from i to j for each $i, j \in \mathcal{C} \cup \mathcal{F}$. Let $\mathbb{S} = (\mathcal{S}^1, \ldots, \mathcal{S}^\tau)$ denote the local optimum and $\mathbb{O} = (\mathcal{O}^1, \ldots, \mathcal{O}^\tau)$ be an optimal solution. Define $\mathcal{S} = \bigcup_{1 \leq t \leq \tau} \mathcal{S}^t$ and $\mathcal{O} = \bigcup_{1 \leq t \leq \tau} \mathcal{O}^t$. Let $\mathcal{P}$ and $\mathcal{P}^*$ denote the

sets of the clients that are penalized when opening the facilities from $\mathcal{S}$ and $\mathcal{O}$, respectively. For each $j \in \mathcal{C}\backslash\mathcal{P}$, let s_j denote the nearest facility to j in $\mathcal{S}$, and define $S_j = d(j, s_j)$. Similarly, for each $j \in \mathcal{C}\backslash\mathcal{P}^*$, let o_j be the nearest facility to j in $\mathcal{O}$, and define $O_j = d(j, o_j)$. For each $i \in \mathcal{O}$ and $i' \in \mathcal{S}$, define $\mathcal{N}^*(i) = \{j \in \mathcal{C}\backslash\mathcal{P}^* : o_j = i\}$ and $\mathcal{N}(i') = \{j \in \mathcal{C}\backslash\mathcal{P} : s_j = i'\}$. For each $\mathcal{O}' \subseteq \mathcal{O}$ and $\mathcal{S}' \subseteq \mathcal{S}$, let $\mathcal{N}^*(\mathcal{O}') = \bigcup_{i\in\mathcal{O}'} \mathcal{N}^*(i)$ and $\mathcal{N}(\mathcal{S}') = \bigcup_{i\in\mathcal{S}'} \mathcal{N}(i)$. Given an integer $1 \leq t \leq \tau$ and a facility $i \in \mathcal{F}^t$, define $T(i) = t$ as its type. Given two integers t_1 and t_2, if $t_1 = t_2$, then let $\delta(t_1, t_2) = 1$. Otherwise, let $\delta(t_1, t_2) = 0$.

3.1 A Set of Swap Pairs

Algorithm 1 closes a set $\mathcal{A}_{out}$ of facilities and opens a set $\mathcal{A}_{in}$ of facilities in each iteration, let $\mathcal{A} = (\mathcal{A}_{out} \mid \mathcal{A}_{in})$ denote this swap pair. We call $\mathcal{A}$ a *feasible swap pair* if $|\mathcal{A}_{out}| = |\mathcal{A}_{in}| \neq 0$, and $|\mathcal{A}_{out} \cap \mathcal{F}^t| = |\mathcal{A}_{in} \cap \mathcal{F}^t|$ holds for each $1 \leq t \leq \tau$. It is easy to show that after performing a feasible swap pair, a feasible solution to the problem is still feasible. We also use a notation of *almost-feasible pairs*. Given a swap pair $\mathcal{B} = (\mathcal{B}_{out} \mid \mathcal{B}_{in})$ that is not feasible, if there exist a facility $i_1 \in \mathcal{B}_{out}$ and a facility $i_2 \in \mathcal{B}_{in}$, such that either $\mathcal{B}_{out}\backslash\{i_1\} = \mathcal{B}_{in}\backslash\{i_2\} = \emptyset$, or $(\mathcal{B}_{out}\backslash\{i_1\} \mid \mathcal{B}_{in}\backslash\{i_2\})$ is a feasible swap pair, then we call $\mathcal{B}$ an almost-feasible pair, and define $T(\mathcal{B}_{out}) = T(i_1)$ and $T(\mathcal{B}_{in}) = T(i_2)$. The following proposition follows directly from the definition of the almost-feasible pairs.

Proposition 1. *Given an almost-feasible pair $\mathcal{B}$ and two facilities $i_1 \in \mathcal{S}$, $i_2 \in \mathcal{O}$, swap pair $(\mathcal{B}_{out} \cup \{i_1\} \mid \mathcal{B}_{in} \cup \{i_2\})$ is a feasible swap pair iff $T(\mathcal{B}_{out}) = T(i_2)$ and $T(\mathcal{B}_{in}) = T(i_1)$.*

It can be seen that no feasible swap pair $\mathcal{A}$ with $|\mathcal{A}_{out} \cap \mathcal{F}^t| = |\mathcal{A}_{in} \cap \mathcal{F}^t| \leq 2\tau$ for each $t \in \{1, \ldots, \tau\}$ can be performed to reduce the cost of the local optimum. We consider a set of such swap pairs to show that the local optimum has small cost. These swap pairs close some facilities from $\mathcal{S}$ and open some facilities from $\mathcal{O}$. The swap pairs are selected based on the following mapping relationships.

Definition 2 ($\varphi(*)$, $\eta(*)$)**.** *For each $i \in \mathcal{O}$, let $\varphi(i)$ denote the nearest facility to i in $\mathcal{S}$. For each $i' \in \mathcal{S}$ with $\varphi^{-1}(i') \neq \emptyset$, let $\eta(i')$ denote the nearest facility to i' in $\varphi^{-1}(i')$. Given a set $\mathcal{S}' \subseteq \mathcal{S}$, define $\varphi^{-1}(\mathcal{S}') = \bigcup_{i\in\mathcal{S}'} \varphi^{-1}(i)$.*

Define $\mathcal{S}_1 = \{i \in \mathcal{S} : \varphi^{-1}(i) \neq \emptyset\}$, $\mathcal{O}_1 = \{\eta(i) : i \in \mathcal{S}_1\}$, $\mathcal{S}_2 = \mathcal{S}\backslash\mathcal{S}_1$, and $\mathcal{O}_2 = \mathcal{O}\backslash\mathcal{O}_1$. The procedure for selecting the swap pairs is given in Algorithm 2. Note that this procedure is only used in the analysis. The algorithm yields a set $\mathbb{A}$ of feasible swap pairs, which is empty initially. In the process of the algorithm, we say that a facility is unpaired if it is not yet involved in a swap pair from $\mathbb{A}$. Each facility is involved in at most one swap pair in $\mathbb{A}$.

In the first loop (steps 2 and 3), Algorithm 2 considers each subset $\mathcal{S}' \subseteq \mathcal{S}_1$ of size no more than τ, and adds $(\mathcal{S}' \mid \bigcup_{i\in\mathcal{S}'}\{\eta(i)\})$ to $\mathbb{A}$ if it is a feasible swap pair. The algorithm then constructs a set $\mathbb{B}$ of almost-feasible pairs. By the termination condition of the first loop, for each unpaired facility $i \in \mathcal{S}_1$, $(\{i\} \mid \{\eta(i)\})$ is not a feasible swap pair and can be viewed as an almost-feasible

Algorithm 2: Selecting a set of swap pairs

Input: The local optimum $(\mathcal{S}^1, \ldots, \mathcal{S}^\tau)$ and an optimal solution $(\mathcal{O}^1, \ldots, \mathcal{O}^\tau)$;
Output: A set $\mathbb{A}$ of swap pairs;

1 $\mathbb{A} \Leftarrow \emptyset$, $\mathbb{B} \Leftarrow \emptyset$, $\mathcal{S}'_1 \Leftarrow \{i \in \bigcup_{1 \le t \le \tau} \mathcal{S}^t : \varphi^{-1}(i) \neq \emptyset\}$, $\mathcal{S}'_2 \Leftarrow (\bigcup_{1 \le t \le \tau} \mathcal{S}^t) \backslash \mathcal{S}'_1$, $\mathcal{O}'_1 \Leftarrow \{\eta(i) : i \in \mathcal{S}'_1\}$, $\mathcal{O}'_2 \Leftarrow (\bigcup_{1 \le t \le \tau} \mathcal{O}^t) \backslash \mathcal{O}'_1$;

2 **while** $\exists \mathcal{S}' \subseteq \mathcal{S}'_1$ *with* $1 \le |\mathcal{S}'| \le \tau$, *such that* $\mathcal{A} = (\mathcal{S}' \mid \bigcup_{i \in \mathcal{S}'} \{\eta(i)\})$ *is a feasible swap pair* **do**

3 $\quad \mathbb{A} \Leftarrow \mathbb{A} \cup \{\mathcal{A}\}$, $\mathcal{S}'_1 \Leftarrow \mathcal{S}'_1 \backslash \mathcal{S}'$, $\mathcal{O}'_1 \Leftarrow \mathcal{O}'_1 \backslash \bigcup_{i \in \mathcal{S}'} \{\eta(i)\}$;

4 **for** *each* $i \in \mathcal{S}'_1$ **do**

5 $\quad \mathbb{B} \Leftarrow \mathbb{B} \cup \{(i \mid \eta(i))\}$;

6 **while** $\exists \mathcal{B}^1, \mathcal{B}^2 \in \mathbb{B}$ *such that* $\mathcal{B} = (\mathcal{B}^1_{out} \cup \mathcal{B}^2_{out} \mid \mathcal{B}^1_{in} \cup \mathcal{B}^2_{in})$ *is an almost-feasible pair* **do**

7 $\quad \mathbb{B} \Leftarrow \mathbb{B} \cup \{\mathcal{B}\} \backslash \{\mathcal{B}^1, \mathcal{B}^2\}$;

8 **while** $\exists\ \mathbb{B}' \subseteq \mathbb{B}$, $\mathcal{H}_1 \subseteq \mathcal{S}'_2$, *and* $\mathcal{H}_2 \subseteq \varphi^{-1}(\bigcup_{\mathcal{B} \in \mathbb{B}'} \mathcal{B}_{out}) \cap \mathcal{O}'_2$, *such that* $1 \le |\mathbb{B}'| = |\mathcal{H}_1| = |\mathcal{H}_2| \le \tau$ *and* $\mathcal{A} = (\bigcup_{\mathcal{B} \in \mathbb{B}'} \mathcal{B}_{out} \cup \mathcal{H}_1 \mid \bigcup_{\mathcal{B} \in \mathbb{B}'} \mathcal{B}_{in} \cup \mathcal{H}_2)$ *is a feasible swap pair* **do**

9 $\quad \mathbb{A} \Leftarrow \mathbb{A} \cup \{\mathcal{A}\}$, $\mathbb{B} \Leftarrow \mathbb{B} \backslash \mathbb{B}'$, $\mathcal{S}'_2 \Leftarrow \mathcal{S}'_2 \backslash \mathcal{H}_1$, $\mathcal{O}'_2 \Leftarrow \mathcal{O}'_2 \backslash \mathcal{H}_2$;

10 **while** $\exists \mathcal{B} \in \mathbb{B}, i_1 \in \mathcal{S}'_2$, *and* $i_2 \in \mathcal{O}'_2$, *such that* $\mathcal{A} = (\mathcal{B}_{out} \cup \{i_1\} \mid \mathcal{B}_{in} \cup \{i_2\})$ *is a feasible swap pair* **do**

11 $\quad \mathbb{A} \Leftarrow \mathbb{A} \cup \{\mathcal{A}\}$, $\mathbb{B} \Leftarrow \mathbb{B} \backslash \{\mathcal{B}\}$, $\mathcal{S}'_2 \Leftarrow \mathcal{S}'_2 \backslash \{i_1\}$, $\mathcal{O}'_2 \Leftarrow \mathcal{O}'_2 \backslash \{i_2\}$;

12 **while** $\exists i_1 \in \mathcal{S}'_2$ *and* $i_2 \in \mathcal{O}'_2$ *such that* $\mathcal{A} = (\{i_1\} \mid \{i_2\})$ *is a feasible swap pair* **do**

13 $\quad \mathbb{A} \Leftarrow \mathbb{A} \cup \{\mathcal{A}\}$, $\mathcal{S}'_2 \Leftarrow \mathcal{S}'_2 \backslash \{i_1\}$, $\mathcal{O}'_2 \Leftarrow \mathcal{O}'_2 \backslash \{i_2\}$;

14 **return** $\mathbb{A}$.

pair. The algorithm adds all such almost-feasible pairs to $\mathbb{B}$ in the second loop (steps 4 and 5). In the third loop (steps 6 and 7), it combines two almost-feasible pairs from $\mathbb{B}$ if this yields a new almost-feasible pair. The combination is performed iteratively until no two pairs in $\mathbb{B}$ can form an almost-feasible pair. After that, the algorithm combines the almost-feasible pairs in $\mathbb{B}$ with some facilities from $\mathcal{S}_2$ and $\mathcal{O}_2$ to obtain a set of feasible swap pairs in the fourth and fifth loops (steps 8, 9, 10, and 11). In the fourth loop (steps 8 and 9), the algorithm iteratively determines whether there exist g almost-feasible pairs $\mathcal{B}^1, \ldots, \mathcal{B}^g$ in $\mathbb{B}$, g unpaired facilities in $\varphi^{-1}(\bigcup_{1 \le t \le g} \mathcal{B}^t_{out}) \cap \mathcal{O}'_2$, and g unpaired facilities in $\mathcal{S}_2$ that can form a feasible swap pair, where $1 \le g \le \tau$. If such a feasible swap pair exists, then the algorithm adds the swap pair to $\mathbb{A}$ and deletes $\mathcal{B}^1, \ldots, \mathcal{B}^g$ from $\mathbb{B}$. In the fifth loop (steps 10 and 11), for each remained almost-feasible pair $\mathcal{B}$ in $\mathbb{B}$, the algorithm finds two unpaired facilities $i_1 \in \mathcal{S}_2$ and $i_2 \in \mathcal{O}_2$, such that $(\mathcal{B}_{out} \cup \{i_1\} \mid \mathcal{B}_{in} \cup \{i_2\})$ is a feasible swap pair and can be added to $\mathbb{A}$. Finally, the remained unpaired facilities in $\mathcal{S}_2$ and $\mathcal{O}_2$ are assigned to single-swap pairs and added to $\mathbb{A}$ in the last loop (steps 12 and 13).

See Figs. 1 and 2 for an example. By the definitions of $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{O}_1$, and $\mathcal{O}_2$, we have $\{r_1, r_2, r_3, r_4, b_1\} = \mathcal{S}_1$, $\{b_2, b_3, b_4\} = \mathcal{S}_2$, $\{r^*_1, r^*_2, b^*_1, b^*_2, b^*_3\} = \mathcal{O}_1$, and

$\{r_3^*, r_4^*, b_4^*\} = \mathcal{O}_2$. It can be seen that $\mathcal{A}_1 = (\{r_1\} \mid \{\eta(r_1)\})$ and $\mathcal{A}_2 = (\{r_2, b_1\} \mid \{\eta(r_2), \eta(b_1)\})$ are two feasible swap pairs and should be added to $\mathbb{A}$ in the first loop of Algorithm 2 (steps 2 and 3). The algorithm then considers two almost-feasible pairs $\mathcal{B}_1 = (\{r_3\} \mid \{\eta(r_3)\})$ and $\mathcal{B}_2 = (\{r_4\} \mid \{\eta(r_4)\})$. In the fourth loop (steps 8 and 9), it combines $\mathcal{B}_1$ with two facilities $r_3^* \in \varphi^{-1}(r_3) \cap \mathcal{O}_2$ and $b_2 \in \mathcal{S}_2$ to obtain a feasible swap pair $\mathcal{A}_3 = (\{r_3, b_2\} \mid \{r_3^*, \eta(r_3)\})$. In the fifth loop (steps 10 and 11), a feasible swap pair $\mathcal{A}_4 = (\{r_4, b_3\} \mid \{r_4^*, \eta(r_4)\})$ is obtained by combining $\mathcal{B}_2$ with two facilities $r_4^* \in \mathcal{O}_2$ and $b_3 \in \mathcal{S}_2$. Finally, the remained unpaired facilities b_4 and b_4^* are combined into a feasible swap pair in the last loop (steps 12 and 13). The constructed swap pairs are shown in Fig. 2.

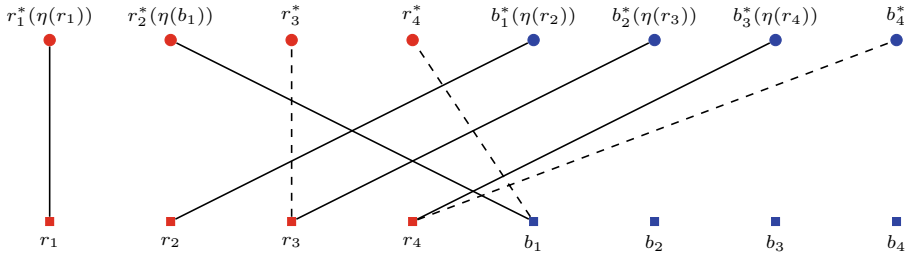

Fig. 1. $\mathcal{O} = \{r_1^*, r_2^*, r_3^*, r_4^*, b_1^*, b_2^*, b_3^*, b_4^*\}$ and $\mathcal{S} = \{r_1, r_2, r_3, r_4, b_1, b_2, b_3, b_4\}$ are the sets of the facilities opened in the optimal solution and local optimum respectively, where $\{r_1^*, r_2^*, r_3^*, r_4^*, r_1, r_2, r_3, r_4\} \subseteq \mathcal{F}^1$ and $\{b_1^*, b_2^*, b_3^*, b_4^*, b_1, b_2, b_3, b_4\} \subseteq \mathcal{F}^2$. For each $i \in \mathcal{S}$ and $i^* \in \varphi^{-1}(i) \backslash \{\eta(i)\}$, we joint i and i^* with a dashed line. For each $i \in \mathcal{S}$ with $\varphi^{-1}(i) \neq \emptyset$, we joint i and $\eta(i)$ with a solid line.

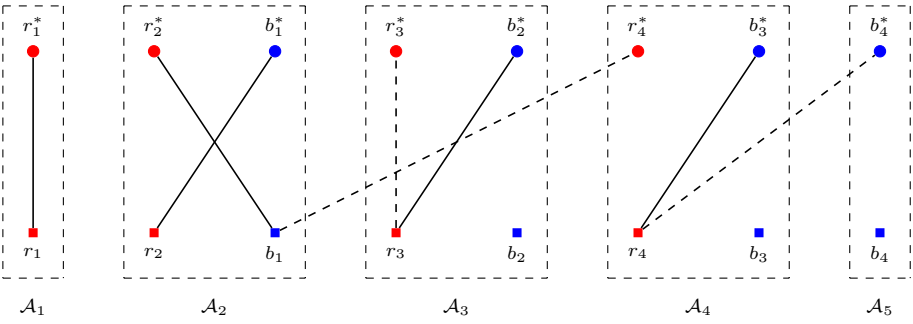

Fig. 2. The constructed swap pairs.

Let $\mathbb{A}$ denote the set of the swap pairs given by Algorithm 2. Let $\mathbb{B}$ be the set of the almost-feasible pairs obtained after the third loop of the algorithm (steps 6 and 7). We now give some useful properties of Algorithm 2.

Proposition 2. *We have* $\sum_{\mathcal{B}\in\mathbb{B}} \delta(T(\mathcal{B}_{out}), t) \cdot \sum_{\mathcal{B}\in\mathbb{B}} \delta(T(\mathcal{B}_{in}), t) = 0$ *for each* $t \in \{1, \ldots, \tau\}$.

Proposition 3. *For each* $\mathcal{A} \in \mathbb{A}$ *and* $i \in \mathcal{A}_{out}$ *with* $\varphi^{-1}(i) \neq \emptyset$, $\eta(i) \in \mathcal{A}_{in}$.

Proposition 4. *For each* $\mathcal{A} \in \mathbb{A}$ *and* $t \in \{1, \ldots, \tau\}$, $\left|\mathcal{A}_{out} \cap \mathcal{S}^t\right| = \left|\mathcal{A}_{in} \cap \mathcal{O}^t\right| \leq 2\tau$.

Proposition 5. *Each facility* $i \in \mathcal{S} \cup \mathcal{O}$ *appears exactly one time in the swap pairs from* $\mathbb{A}$.

3.2 An Upper Bound on the Cost Increase

In this section, we present a strategy for reconnecting clients after performing the swap pairs from $\mathbb{A}$ on the local optimum. This gives an upper bound on the increased cost induced by a swap pair. Consider a swap pair $\mathcal{A} \in \mathbb{A}$, we close the facilities in $\mathcal{A}_{out}$ and open the facilities in $\mathcal{A}_{in}$. We reconnect the clients from $\mathcal{N}(\mathcal{A}_{out}) \cup \mathcal{N}^*(\mathcal{A}_{in})$. Each $j \in \mathcal{N}^*(\mathcal{A}_{in})$ is reconnected to o_j (o_j is guaranteed to be opened by the definition of $\mathcal{N}^*(*)$). We pay the penalty costs of the clients from $\mathcal{N}(\mathcal{A}_{out}) \cap \mathcal{P}^*$. The clients from $\mathcal{N}(\mathcal{A}_{out}) \backslash \mathcal{P}^*$ should be reconnected to a nearby opened facility. By Proposition 3, $\eta(i)$ is opened for each $i \in \mathcal{A}_{out}$. This motivates the following strategy for reconnecting each $j \in \mathcal{N}(\mathcal{A}_{out}) \backslash \mathcal{P}^*$. See Fig. 3 for an example of the strategy.

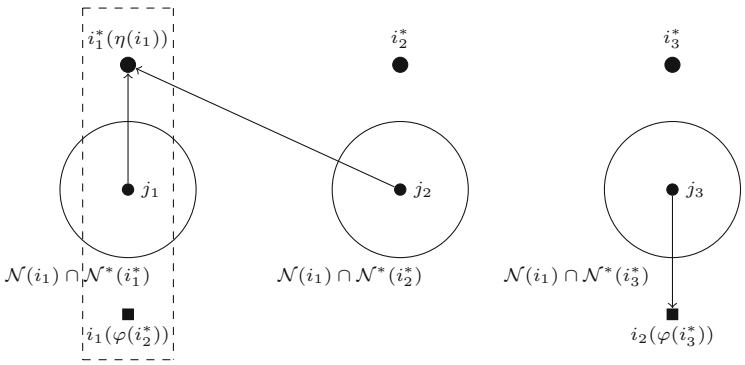

Fig. 3. Reconnection of the clients after performing the swap pair $\mathcal{A} = (\{i_1\} \mid \{i_1^*\})$. The solid lines indicate the connection of the clients after the swap. For each $j \in \mathcal{N}(i_1) \backslash \mathcal{P}^*$, j is reconnected to o_j, if $j \in \mathcal{N}(i_1) \cap \mathcal{N}^*(i_1^*)$; $\varphi(o_j)$, if $j \in \mathcal{N}(i_1) \cap \mathcal{N}^*(i_3^*)$; and $\eta(\varphi(o_j))$, if $j \in \mathcal{N}(i_1) \cap \mathcal{N}^*(i_2^*)$.

- If $o_j \in \mathcal{A}_{in}$, then j is reconnected to o_j.
- If $o_j \notin \mathcal{A}_{in}$ and $\varphi(o_j) \notin \mathcal{A}_{out}$, then j is reconnected to $\varphi(o_j)$.
- If $o_j \notin \mathcal{A}_{in}$ and $\varphi(o_j) \in \mathcal{A}_{out}$, then j is reconnected to $\eta(\varphi(o_j))$.

The following lemma shows that the reconnection cost of each client $j \in \mathcal{N}(\mathcal{A}_{out})\backslash\mathcal{P}^*$ can be bounded by a combination of O_j and S_j.

Lemma 1. *For each $j \in \mathcal{C}\backslash(\mathcal{P}^* \cup \mathcal{P})$, we have $d(j, \varphi(o_j)) \leq 2O_j + S_j$ and $d(j, \eta(\varphi(o_j))) \leq 3O_j + 2S_j$.*

The following lemma follows from an upper bound on the increased cost induced by performing a swap pair from $\mathbb{A}$.

Lemma 2. *For each swap pair $\mathcal{A} \in \mathbb{A}$, we have*

$$\begin{aligned} 0 \leq & \sum_{j \in \mathcal{N}^*(\mathcal{A}_{in}) \cap \mathcal{P}} (O_j - p(j)) + \sum_{j \in \mathcal{N}^*(\mathcal{A}_{in}) \backslash \mathcal{P}} (O_j - S_j) \\ & + \sum_{j \in \mathcal{N}(\mathcal{A}_{out}) \cap \mathcal{P}^*} (p(j) - S_j) + \sum_{j \in [\mathcal{N}(\mathcal{A}_{out}) \backslash \mathcal{P}^*] \backslash \mathcal{N}^*(\varphi^{-1}(\mathcal{A}_{out}) \cup \mathcal{A}_{in})} 2O_j \\ & + \sum_{j \in \mathcal{N}^*(\varphi^{-1}(\mathcal{A}_{out}) \backslash \mathcal{A}_{in}) \cap \mathcal{N}(\mathcal{A}_{out})} (3O_j + S_j). \end{aligned} \tag{1}$$

3.3 A Layered Structure of the Swap Pairs

Observe that inequality (1) contains "$+O_j$" terms for some clients from $\mathcal{C}\backslash\mathcal{P}^*$, "$-S_j$" terms for some clients from $\mathcal{C}\backslash\mathcal{P}$, "$+p(j)$" terms for some clients from $\mathcal{P}^*$, and "$-p(j)$" terms for some clients from $\mathcal{P}$. Our idea for obtaining the approximation guarantee for the local optimum is to add together some inequalities of this type such that we can get $O(1)(\sum_{j \in \mathcal{C}\backslash\mathcal{P}^*} O_j + \sum_{j \in \mathcal{P}^*} p(j)) - O(1)(\sum_{j \in \mathcal{C}\backslash\mathcal{P}} S_j + \sum_{j \in \mathcal{P}} p(j)) \geq 0$, which directly implies the desired approximation ratio. The challenge is that inequality (1) also involves a "$+S_j$" term for a client from $\mathcal{N}^*(\varphi^{-1}(\mathcal{A}_{out})\backslash\mathcal{A}_{in}) \cap \mathcal{N}(\mathcal{A}_{out})$. In this case, we have to repeatedly use another inequality which contains a "$-S_j$" term to counteract the "$+S_j$" term. To obtain a "$-S_j$" term for each $j \in \mathcal{C}\backslash\mathcal{P}$, we prove the existence of a layered structure of the swap pairs from $\mathbb{A}$. Note that this structure is only used in the analysis.

Lemma 3. *$\mathbb{A}$ can be partitioned into f disjoint sets $\mathbb{A}_1, \ldots, \mathbb{A}_f$ satisfying the following properties.*

- $3 \leq f \leq \tau + 1$.
- $\bigcup_{\mathcal{A} \in \mathbb{A}_t} \varphi^{-1}(\mathcal{A}_{out})\backslash\mathcal{A}_{in} \subseteq \bigcup_{\mathcal{A} \in \mathbb{A}_t^-} \mathcal{A}_{in}$ *for each* $t \in \{1, \ldots, f-1\}$, *where* $\mathbb{A}_t^- = \bigcup_{t < t' \leq f} \mathbb{A}_{t'}$.
- $\bigcup_{\mathcal{A} \in \mathbb{A}_f} \varphi^{-1}(\mathcal{A}_{out}) = \emptyset$.

Before proving Lemma 3, we first show its implication. Given a swap pair $\mathcal{A} \in \mathbb{A}$, inequality (1) involves a "$+S_j$" term for each $j \in \mathcal{N}^*(\varphi^{-1}(\mathcal{A}_{out})\backslash\mathcal{A}_{in}) \cap \mathcal{N}(\mathcal{A}_{out})$ and a "$-S_j$" term for each $j \in \mathcal{N}^*(\mathcal{A}_{in})$. Using Lemma 3, we know that $\bigcup_{\mathcal{A} \in \mathbb{A}_1} \varphi^{-1}(\mathcal{A}_{out})\backslash\mathcal{A}_{in} \subseteq \bigcup_{\mathcal{A} \in \mathbb{A}_1^-} \mathcal{A}_{in}$ and $\bigcup_{\mathcal{A} \in \mathbb{A}_2} \varphi^{-1}(\mathcal{A}_{out})\backslash\mathcal{A}_{in} \subseteq \bigcup_{\mathcal{A} \in \mathbb{A}_2^-} \mathcal{A}_{in}$. Thus, we can multiply inequality (1) by factor 2 for each $\mathcal{A} \in \mathbb{A}_1^-$ to counteract the "$+S_j$" terms induced by the swap pairs from $\mathbb{A}_1$. Now each swap pair

from $\mathbb{A}_2$ induces some "$+2S_j$" terms, which can be counteracted by multiplying inequality (1) by factor 3 instead of 2 for each $\mathcal{A} \in \mathbb{A}_2^-$. By a similar argument, we can counteract all the "$+S_j$" terms through using the swap pairs from $\mathbb{A}_t$ for t times, for each $t \in \{1, \ldots, f\}$. We will later show that this yields an $O(f)$-approximation guarantee for the prize-collecting τ-color median problem.

Proof [**of Lemma** 3]. Let $\mathbb{A}'$ denote the set of the swap pairs added to $\mathbb{A}$ in the fifth loop of Algorithm 2 (steps 10 and 11). It can be seen that each $\mathcal{A} \in \mathbb{A}'$ is a combination of an almost-feasible pair and two facilities from $\mathcal{S}_2 \cup \mathcal{O}_2$. For each $\mathcal{A} \in \mathbb{A}'$, define $T(\mathcal{A}) = T(\mathcal{B}'_{out})$, where $\mathcal{B}'$ denotes the almost-feasible pair involved in $\mathcal{A}$. For each $t \in \{1, \ldots, \tau\}$, define $\mathbb{A}'_t = \{\mathcal{A} \in \mathbb{A}' : T(\mathcal{A}) = t\}$. We construct a directed graph G as follows: A vertex v_t is constructed for each $t \in \{1, \ldots, \tau\}$ with $\mathbb{A}'_t \neq \emptyset$; For any two vertices v_{t^1}, v_{t^2} of G, there is a directed edge (simply called arc) from v_{t^1} to v_{t^2} if there exist a swap pair $\mathcal{A} \in \mathbb{A}'_{t^1}$ and a facility $i \in \varphi^{-1}(\mathcal{A}_{out}) \backslash \mathcal{A}_{in}$ such that $T(i) = t^2$. We have the following claim.

Claim 1. *G is a directed acyclic graph, whose vertices are no more than $\tau - 1$.*

Define $\mathcal{V}$ as the vertex set of G. Given two vertices $v, v' \in \mathcal{V}$, if there exists a path from v to v' in G, then let $L(v, v')$ denote the number of the vertices in a longest path from v to v'. Otherwise, let $L(v, v') = 2$. Let $\mathcal{V}_0$ denote the set of the vertices in G whose in-degrees are 0. For each $v \in \mathcal{V}_0$, define $L(v) = 2$. For each $v \in \mathcal{V} \backslash \mathcal{V}_0$, define $L(v) = \max_{v' \in \mathcal{V}_0} L(v', v) + 1$. We have $2 \leq L(v) \leq \left|\mathcal{V}\right| + 1$ for each $v \in \mathcal{V}$. Let $f = \max_{v \in \mathcal{V}} L(v) + 1$. We have $3 \leq f \leq \left|\mathcal{V}\right| + 2 \leq \tau + 1$ by Claim 1. We partition $\mathbb{A}$ into f disjoint sets $\mathbb{A}_1, \ldots, \mathbb{A}_f$ as follows.

- Let $\mathbb{A}_1$ be the set of the swap pairs added to $\mathbb{A}$ in the first and fourth loops of Algorithm 2 (steps 2, 3, 8, and 9).
- For each integer $1 < g < f$, let $\mathbb{A}_g = \bigcup_{L(v_t)=g} \mathbb{A}'_t$.
- Let $\mathbb{A}_f$ be the set of the swap pairs added to $\mathbb{A}$ in the last loop of Algorithm 2 (steps 12 and 13).

Recall that $\mathbb{A}_t^- = \bigcup_{t<t' \leq f} \mathbb{A}_{t'}$ for each $t \in \{1, \ldots, f-1\}$. Based on Claim 1 and the properties of the swap pairs from $\mathbb{A}$, we have the following result.

Claim 2. *For each $t \in \{1, \ldots, f-1\}$, $\bigcup_{\mathcal{A} \in \mathbb{A}_t} \varphi^{-1}(\mathcal{A}_{out}) \backslash \mathcal{A}_{in} \subseteq \bigcup_{\mathcal{A} \in \mathbb{A}_t^-} \mathcal{A}_{in}$.*

For each swap pair $\mathcal{A}$ added to $\mathbb{A}$ in the last loop of Algorithm 2, we have $\mathcal{A}_{out} \subseteq \mathcal{S}_2$. The definition of $\mathcal{S}_2$ implies that $\varphi^{-1}(\mathcal{S}_2) = \emptyset$, which in turn implies that $\varphi^{-1}(\mathcal{A}_{out}) = \emptyset$ for each $\mathcal{A} \in \mathbb{A}_f$. By the fact that $3 \leq f \leq \tau + 1$ and Claim 2, we complete the proof of Lemma 3. □

3.4 Bound the Cost of the Local Optimum

We are now ready to bound the cost of the local optimum. Adding together several inequalities (1), we obtain the following result.

Lemma 4. $\sum_{j\in\mathcal{C}\setminus\mathcal{P}} S_j + \sum_{j\in\mathcal{P}\setminus\mathcal{P}^*} p(j) \leq (4\tau+1)\sum_{j\in\mathcal{C}\setminus\mathcal{P}^*} O_j + (\tau+1)\sum_{j\in\mathcal{P}^*\setminus\mathcal{P}} p(j)$.

Adding $\sum_{j\in\mathcal{P}^*\cap\mathcal{P}} p(j)$ to both sides of the inequality in Lemma 4 and simplifying, we have

$$\sum_{j\in\mathcal{C}\setminus\mathcal{P}} S_j + \sum_{j\in\mathcal{P}} p(j) \leq (4\tau+1)\sum_{j\in\mathcal{C}\setminus\mathcal{P}^*} O_j + (\tau+1)\sum_{j\in\mathcal{P}^*} p(j) \leq (4\tau+1)OPT,$$

which implies that the local optimum is a $(4\tau+1)$-approximation solution to the prize-collecting τ-color median problem.

References

1. Arya, V., Garg, N., Khandekar, R., Meyerson, A., Munagala, K., Pandit, V.: Local search heuristics for k-median and facility location problems. SIAM J. Comput. **33**(3), 544–562 (2004)
2. Bateni, M., Hajiaghayi, M.: Assignment problem in content distribution networks: unsplittable hard-capacitated facility location. ACM Trans. Algorithms **8**(3), 20:1–20:19 (2012)
3. Charikar, M., Khuller, S., Mount, D.M., Narasimhan, G.: Algorithms for facility location problems with outliers. In: Proceedings of the 12th ACM-SIAM Symposium on Discrete Algorithms, pp. 642–651 (2001)
4. Charikar, M., Li, S.: A dependent LP-rounding approach for the k-median problem. In: Czumaj, A., Mehlhorn, K., Pitts, A., Wattenhofer, R. (eds.) ICALP 2012. LNCS, vol. 7391, pp. 194–205. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31594-7_17
5. Cohen-Addad, V., Feldmann, A.E., Saulpic, D.: Near-linear time approximation schemes for clustering in doubling metrics. In: Proceedings of the 60th IEEE Symposium on Foundations of Computer Science, pp. 540–559 (2019)
6. Feng, Q., Zhang, Z., Huang, Z., Xu, J., Wang, J.: Improved algorithms for clustering with outliers. In: Proceedings of the 30th International Symposium on Algorithms and Computation, pp. 61:1–61:12 (2019)
7. Feng, Q., Zhang, Z., Shi, F., Wang, J.: An improved approximation algorithm for the k-means problem with penalties. In: Chen, Y., Deng, X., Lu, M. (eds.) FAW 2019. LNCS, vol. 11458, pp. 170–181. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-18126-0_15
8. Friggstad, Z., Zhang, Y.: Tight analysis of a multiple-swap heurstic for budgeted red-blue median. In: Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming, pp. 75:1–75:13 (2016)
9. Hajiaghayi, M.T., Khandekar, R., Kortsarz, G.: Budgeted red-blue median and its generalizations. In: de Berg, M., Meyer, U. (eds.) ESA 2010. LNCS, vol. 6346, pp. 314–325. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15775-2_27
10. Hajiaghayi, M., Khandekar, R., Kortsarz, G.: Local search algorithms for the red-blue median problem. Algorithmica **63**(4), 795–814 (2012). https://doi.org/10.1007/s00453-011-9547-9
11. Jain, K., Mahdian, M., Markakis, E., Saberi, A., Vazirani, V.V.: Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP. J. ACM **50**(6), 795–824 (2003)

12. Krishnaswamy, R., Kumar, A., Nagarajan, V., Sabharwal, Y., Saha, B.: The matroid median problem. In: Proceedings of 22nd Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1117–1130 (2011)
13. Krishnaswamy, R., Li, S., Sandeep, S.: Constant approximation for k-median and k-means with outliers via iterative rounding. In: Proceedings of the 50th ACM Symposium on Theory of Computing, pp. 646–659 (2018)
14. Swamy, C.: Improved approximation algorithms for matroid and knapsack median problems and applications. ACM Trans. Algorithms **12**(4), 49:1–49:22 (2016)
15. Zhang, D., Hao, C., Wu, C., Xu, D., Zhang, Z.: Local search approximation algorithms for the k-means problem with penalties. J. Comb. Optim. **37**(2), 439–453 (2019)