Ordozgoiti, Bruno; Gionis, Aristides

Reconciliation k-median

Published in:

DOI:
10.1145/3308558.3313475

Published: 13/05/2019

Please cite the original version:
Reconciliation \( k \)-median: Clustering with Non-polarized Representatives

Bruno Ordozgoiti∗
Department of Computer Systems
Universidad Politécnica de Madrid
bruno.ordozgoiti@upm.es

Aristides Gionis
Department of Computer Science
Aalto University
aristides.gionis@aalto.fi

ABSTRACT
We propose a new variant of the \( k \)-median problem, where the objective function models not only the cost of assigning data points to cluster representatives, but also a penalty term for disagreement among the representatives. We motivate this novel problem by applications where we are interested in clustering data while avoiding selecting representatives that are too far from each other. For example, we may want to summarize a set of news sources, but avoid selecting ideologically-extreme articles in order to reduce polarization.

To solve the proposed \( k \)-median formulation we adopt the local-search algorithm of Arya et al. [2]. We show that the algorithm provides a provable approximation guarantee, which becomes constant under a mild assumption on the minimum number of points for each cluster. We experimentally evaluate our problem formulation and proposed algorithm on datasets inspired by the motivating applications. In particular, we experiment with data extracted from Twitter, the US Congress voting records, and popular news sources. The results show that our objective can lead to choosing less polarized groups of representatives without significant loss in representation fidelity.

CCS CONCEPTS
• Theory of computation → Facility location and clustering: Discrete optimization; • Applied computing → Sociology.

KEYWORDS
Clustering; \( k \)-median; polarization; committee selection; data mining; approximation algorithms.

ACM Reference Format:

1 INTRODUCTION
Consider the problem of summarizing a set of news articles on a given topic. A standard approach to this problem is clustering:

\[ \text{cost}(S) = \sum_{c \in C} \min_{s \in S} d(c, s) + \frac{\lambda}{2} \sum_{s_j \in S} \sum_{s_i \in S} d(s_i, s_j). \]  

The first term is the same as in the standard \( k \)-median, and expresses the cost of serving each client by its closest selected facility. The second term is the one introduced in this paper and expresses

---

∗The work described in this paper was done while this author was a visitor at Aalto University.
disagreement between cluster representatives. The parameter $\lambda$ determines the relative importance of the two terms. Despite clustering being one of the most well-studied problems in statistical data analysis, and the numerous formulations and problem variants that have been studied in the literature, to our knowledge, the problem defined above has not been considered before.

As expected, the problem defined by optimizing Equation (1) is NP-hard; in fact optimizing each of the two terms separately is an NP-hard problem. Given the hardness of the problem, it is compelling to consider algorithms with provable approximation guarantees. For the $k$-median algorithm several approximation algorithms exist [11, 22]. A local-search algorithm, which is simple to implement and scalable to large datasets, has been proposed by Arya et al. [2]. The algorithm starts with an arbitrary solution and considers a swap of $p$ selected facilities with $p$ non-selected facilities; the swap is materialized if the cost improves, and the process continues until the cost cannot be improved. Arya et al. show that this algorithm achieves an approximation guarantee equal to $3 + 2/p$, and the running time is $O(n^p)$. In particular, for $p = 1$, the algorithm gives an approximation guarantee equal to 5, while the running time is linear.

In this paper we show how to adapt the local-search algorithm of Arya et al. [2] for the problem we consider and obtain an approximation guarantee $O(k)$ in the general case. The proposed algorithm considers 1-facility swaps, i.e., $p = 1$. Furthermore, when the clusters of the obtained solution have size $\Omega(\lambda k)$, the approximation factor becomes 11, i.e., constant. We complete the analysis of the proposed problem by deriving bounds on the objective function.

Our contributions in this paper are summarized as follows.

- We introduce the reconciliation $k$-median problem, a novel clustering problem formulation where we aim to optimize the data representation cost plus a term for agreement between the cluster representatives.
- We adapt the local-search algorithm of Arya et al. [2] and obtain provable approximation guarantees for the proposed clustering problem.
- We run experiments on datasets extracted from the Twitter social network, US Congress voting records, and popular news sources. The results show that the proposed objective can lead to the choice of less polarized groups of representatives, as measured by a well-known method for ideology estimation [4] and an objective estimate of the political leaning of news sources.

The rest of the paper is structured as follows. In Section 2 we present a brief overview to the literature that is most related to our work. In Section 3 we formally define the reconciliation $k$-median problem. In Section 4 we present the local-search algorithm and state its approximability properties. In Section 5 we present our experimental evaluation, while Section 6 is a short conclusion. To improve readability, the hardness proof of the reconciliation $k$-median problem and the proof of the approximation guarantee of the local-search algorithm are presented in the Appendix.

2 RELATED WORK

Data clustering is one of the most well-studied problems in data analysis, with applications in a wide range of areas [17]. Among the numerous formulations that have been proposed, in this paper we focus on the $k$-median problem setting, which has been studied extensively in the theoretical computer-science literature. Charikar et al. [11] gave the first constant-factor approximation algorithm for the $k$-median problem, followed by improvements that relied on both combinatorial and LP-based algorithms [10, 18, 19]. In this paper we build upon the local-search algorithm of Byrka et al. [8], which optimizes a part of the algorithm of Li and Svensson [22]. However, the algorithm is not practical. Variants of the $k$-median problem have also been considered, including the Euclidean $k$-median [1], capacitated $k$-median [7], ordered $k$-median [9], and more. To our knowledge, however, this is the first work to study the problem of $k$-median clustering with a penalty on the disagreement of the cluster representatives. Instead, researchers have studied the problem of selecting $k$ points to maximize the sum of pairwise distances, i.e., the dispersion of the selected point set. Several constant-factor approximation algorithms have been proposed for the maximum-dispersion problem [12, 16]. However, the maximization makes the problem different and it is not clear how to adapt those algorithms in our setting.

One of our motivating applications is summarization of social media content with the aim of reducing polarization and balancing the information content delivered to users. This is a relatively new research area that is receiving increasing interest [4, 13, 14, 23, 24]. However, to the best of our knowledge, none of the proposed approaches uses a clustering formulation.

The second motivating application is election of committees and representatives. In some cases, election questions can also be formulated as voting problems. Voting in general has been studied extensively in social sciences and economics literature. From the algorithmic perspective, researchers have studied questions about voting in social networks and concepts such as liquid and viscous democracy [6, 26]. In addition to being not directly related to our paper, this line of work does not directly model agreement between elected representatives.

3 PROBLEM FORMULATION

3.1 Preliminaries

We formulate our problem in the general setting of metric facility location [18]. We consider a metric space $(X, d)$ and two subsets $F, C \subseteq X$, not necessarily disjoint. The set $F$ represents facilities, and the set $C$ represents clients. A special case of interest is when clients and facilities are defined over the same set, i.e., $F = C$. In our discussion we consider the more general case that the sets of clients and facilities are disjoint. The function $d : X \times X \to \mathbb{R}$ is a distance measure between pairs of points in $X$. When $c \in C$ and $s \in F$, the distance $d(c, s)$ represents the cost of serving client $c$ with facility $s$. The number of facilities is denoted by $m = |F|$ and the number of clients by $n = |C|$.

The goal is to open $k$ facilities — i.e., choose $k$ points in $F$ such that the cost of serving each client in $C$ with the nearest selected facility is minimized. Given a set of facilities $S \subseteq F$, with $|S| = k$, and $s \in S$, we use $N_S(s)$ to denote the set of clients served by facility $s$ in
3.2 Reconciling cluster representatives

The problems described above, facility location and $k$-median, are commonly used to find cluster representatives without regard to the relative position of the representatives themselves. As discussed in the introduction, our goal is to modify the problem definition so as to find solutions in which the cluster representatives are close to each other. To achieve our goal we propose the following clustering variant, which we name reconciliation $k$-median.

**Problem 1 (recon-$k$-median).** Given a metric space $(X, d)$, two sets $F, C \subseteq X$, and a real number $\lambda > 0$, find a set $S \subseteq F$ with $|S| = k$, so as to minimize the cost function

$$\text{cost}(S) = \sum_{c \in S} \sum_{x \in C} d(c, x) + \frac{\lambda}{2} \sum_{s_i \in S} \sum_{s_j \in S} d(s_i, s_j).$$

In order to characterize the hardness of this problem, we analyze the two terms of the objective in isolation. The first term, which results from setting $\lambda = 0$, is equivalent to the classic metric $k$-median problem, shown to be NP-hard by Papadimitriou [25].

The goal is to find a binary vector $x$ of dimension $n$ that has exactly $k$ entries equal to 1 and minimizes the sum of pairwise distances in a metric space, i.e., minimum pairwise distances (MPD).

**Problem 2 (MPD).** Given a metric space $(X, d)$, a subset of objects in the metric space $F = \{x_1, \ldots, x_k\} \subseteq X$, and a number $k \in \mathbb{N}$, with $k < n$, define a matrix $A \in \mathbb{R}^{n \times k}$ as $A_{ij} = d(x_i, x_j)$ for all $x_i, x_j \in F$. The goal is to find a binary vector $x$ of dimension $n$ that has exactly $k$ coordinates equal to 1 and minimizes the form $x^T Ax$. In other words, we want to find

$$\min \quad x^T Ax,$$

subject to $x \in \{0, 1\}^n$ and $x^T 1 = k$.

The following lemma establishes that there exists no polynomial-time algorithm to find the exact solution to the MPD problem, unless P = NP. The proof is given in the Appendix.

**Lemma 1.** Problem MPD is NP-hard.

Lemma 1 establishes that optimizing separately the second term of the objective function (2) is an NP-hard problem. Note, however, that the hardness of the two terms in Problem 1 does not immediately imply the hardness of the overall problem. Consider, for instance, an objective of the form $\min_x \{f(x) + (c - f(x))\}$. Even though optimizing $f$ can be an arbitrary NP-hard problem, the overall problem has a constant value, and thus, it is trivial to optimize — there is nothing to be done. We now show that Problem 1 is indeed NP-hard.

**Lemma 2.** Problem recon-$k$-median is NP-hard.

**Proof.** Consider an instance of the MPD problem, for a given set $F$ and a number $k$, and form an instance $(F, C, k, \lambda)$ of the recon-$k$-median problem, where $C$ is an arbitrary set of clients with $F \cap C = \emptyset$. Set $d(s, c) = \frac{1}{\lambda} \max_{s_j \in F} d(s, s_j)$, for all $c \in C$ and $s \in F$. Note that the distance function $d$ is still a valid metric. We have that $\sum_{c \in C} \min_{s \in S} d(c, s)$ is constant for any potential solution set $S \subseteq F$, which implies that optimizing recon-$k$-median is equivalent to optimizing MPD. Thus, from Lemma 1 we obtain that recon-$k$-median is NP-hard. □

3.3 Bounds on the objective

To complete the analysis of the recon-$k$-median problem we offer bounds on the objective, which can be used to evaluate the quality of the solution obtained by any algorithm for the problem at a given instance. For this to be useful, the bounds need to be non-trivial and as close as possible to the optimal solution. We will first show how to obtain a lower bound on the second term of the objective, that is, $\sum_{s_i, s_j \in S} d(s_i, s_j)$. We first introduce the following definition.

**Definition 1.** Given two sequences of real numbers $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ and $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_m$, with $m < n$, we say that the second sequence interleaves the first if $\lambda_i \geq \mu_i \geq \lambda_{n-m+i}$ for $i = 1, \ldots, m$.

We will employ the following result from Haemers [15].

**Lemma 3.** Let $A$ be a symmetric $n \times n$ matrix, and let $B$ be a principal submatrix of $A$. Then the eigenvalues of matrix $B$ interlace those of matrix $A$.

We now state the following result.

**Theorem 1.** Let $D$ be the pairwise distance matrix of facilities of an instance of recon-$k$-median problem. Define matrix $\tilde{D}$ so that $\tilde{D}_{ij} = \sqrt{D_{ij}}$, that is, a matrix whose entries are the square roots of the entries of $D$. Let $\lambda_1(\tilde{D})$ denote the $i$-th absolutely largest eigenvalue of $\tilde{D}$. Then

$$k \lambda_1(D) \geq \sum_{i, j \in S} d(s_i, s_j) \geq \sum_{i=n-k+1}^{n} \lambda_i^2(D).$$

**Proof.** Observe that for any real symmetric matrix $D$, $\|D\|^2_F = \sum_i \sigma_i^2(D) = \sum_i \lambda_i^2(D)$. It is easily seen that if $x$ is a binary vector with exactly $k$ entries equal to 1, $x^T D x$ is equal to the sum of the entries of a $k \times k$ principal submatrix of $D$, which is in turn equal to the squared Frobenius norm of the corresponding submatrix of $\tilde{D}$. Combined with Theorem 3, this proves the lower bound.

The upper bound follows immediately from the variational characterization of the eigenvalues [see 20, chap. 9.2].

A lower and upper bound on the first term of the objective can simply be given by $\sum_{c \in C} \min_{s \in F} d(c, s)$ and $\sum_{c \in C} \max_{s \in F} d(c, s)$, respectively. This is useful only for problem instances where the number of facilities is relatively small compared to the number of clients.
Algorithm 1 Local search

1: procedure LocalSearch$(F, C, k, \lambda)$
2: $S \leftarrow$ random subset of $F$ of cardinality $k$
3: converged $\leftarrow$ false
4: while not converged do
5: if there exist $s \in S$ and $t \in F \setminus S$ such that $\text{cost}(S \setminus \{s\} \cup \{t\}) < \text{cost}(S)$ then
6: $S \leftarrow S \setminus \{s\} \cup \{t\}$
7: else converged $\leftarrow$ true
8: end if
9: end while
10: return $S$
11: end procedure

4 THE LOCAL-SEARCH ALGORITHM

In this section we present the proposed algorithm for the recon-$k$-median problem and state its properties. The algorithm uses the local-search strategy, proposed by Arya et al. [2], but adapted for the objective function of recon-$k$-median. The algorithm starts with an arbitrary solution consisting of $k$ selected facilities. It then proceeds in iterations. In each iteration it considers whether it is possible to swap a selected facility with a non-selected facility and obtain an improvement in the objective score. If such an improvement is possible, the corresponding swap is performed. The algorithm terminates when no such swap is possible. At each point during its execution, the algorithm maintains a set of $k$ clusters over the set of clients, and a selected facility for each cluster, defined by assigning each client to its closest selected facility. Pseudocode of this local-search procedure is given in Algorithm 1.

For the analysis we denote by $S = \{s_1, \ldots, s_k\} \subseteq F$ the solution returned by LocalSearch, and $O = \{o_1, \ldots, o_k\} \subseteq F$ the optimal solution. As mentioned before we use the notation $N_S(s)$ to denote the set of clients that are assigned to facility $s$ in the solution $S$, and $N_O(o)$ to denote the set of clients that are assigned to facility $o$ in the optimal solution $O$.

To analyze the performance of LocalSearch we follow the ideas of Arya et al. [2]. The proofs are included in the Appendix.

As a result, in the most general case, the LocalSearch algorithm yields a $O(\lambda k)$-factor approximation guarantee on the quality of the solution achieved.

**Theorem 2.** Let $(F, C, k, \lambda)$ be an instance of the recon-$k$-median problem. Let $S$ be a solution returned by the LocalSearch algorithm, and let $O$ be an optimal solution. Then

\[ \text{cost}(S) \leq (\lambda k + 5)\text{cost}(O). \]  

(3)

Furthermore, we are able to improve the analysis and obtain an approximation guarantee that does not depend on the number of facilities $k$. For the improved result we need to make the mild assumption that the number of clients in any cluster of the optimal solution and the solution returned by the algorithm is $\Omega(\lambda k)$. In particular, we have.

**Theorem 3.** Let $(F, C, k, \lambda)$ be an instance of the recon-$k$-median problem. Let $S$ be a solution returned by the LocalSearch algorithm, and let $O$ be an optimal solution. Assume that $|N_S(s)| \geq [2\lambda]k$ for all $s \in S$ and $|N_O(o)| \geq [2\lambda]k$ for all $o \in O$. Then

\[ \text{cost}(S) \leq \max\{11, 4\lambda\} \text{cost}(O). \]

The running time of the LocalSearch algorithm is $O(nmk)$ per iteration. For most applications $k$ is considered to be a constant. When the number of facilities $m$ is of the same order of magnitude with the number of clients $n$, e.g., in the important special case $m = C$, the algorithm has quadratic complexity per iteration. However, in many applications the number of facilities is significantly smaller than the number of clients. Thus, we expect that the algorithm is very efficient in practice for those cases.

We also note that Arya et al. [2] show how to perform swaps of $p$ facilities simultaneously and obtain an improved performance guarantee at the expense of increased running time. In our case, the penalty term in our objective, which captures the disagreement among the cluster representatives, makes the analysis significantly more complex and it is not clear how to use simultaneous swaps in order to achieve a similar quality-performance trade-off. Thus, this direction is left for future work.

5 EXPERIMENTAL EVALUATION

We perform experiments to assess the proposed concept of clustering with reconciliation of representatives, as well as the performance of the proposed LocalSearch algorithm. Our objective is to evaluate by some objective measure whether the proposed problem formulation, as well as natural variations, can lead to a choice of representatives or sources that are more moderate, less polarizing and more likely to reach consensus.

To enrich the experimental setting and produce a more interesting set of empirical observations, we relax some of the requisites of our theoretical results. Namely, observe that in order to prove the approximation guarantees of the LocalSearch algorithm, we require that the distance function satisfy the properties of a metric, and that it be the same for facilities and clients. However, we believe that in practical scenarios, one might benefit from considering a wider set of options, especially if we consider the exploratory nature of clustering algorithms. Therefore, we measure dissimilarity between objects using functions that do not necessarily qualify as metrics, and we consider different ones for facilities and clients.

At a high level, our experimental methodology is as follows: We start with a dataset for which clients and facilities model a natural clustering problem and for which a distance function $d$ is available. For the facilities of the dataset we seek to obtain a polarity score $\pi$, which is independent of the distance function $d$: facilities with similar polarity scores $\pi$ are more likely to agree. In addition, facilities with scores closer to the middle of the spectrum are less likely to disseminate extremist ideologies. We then apply our clustering algorithm with varying values of the parameters $k$ and $\lambda$. We are interested in answering the following questions:

**Q1.** How does the agreement between selected representatives or the polarization of information sources (measured by the independent polarity score $\pi$) change as a function of $\lambda$? In other words, can we get more reconciled representatives or less extreme sources by increasing the weight of

---

Our implementation of the algorithm is available at https://github.com/brunez/recon-kmedian-ls
the disagreement penalty term (second term of the objective function (2))? 

Q2. How does the $k$-median score change as a function of $\lambda$? In other words, can we find solutions with more reconciled representatives but without significant loss in representation fidelity (i.e., first term of the objective function (2))? 

Q3. What is the impact of the parameter $k$ on both polarity score and $k$-median score? 

For our experimental evaluation we use the following datasets.

Twitter: The dataset, obtained by Lahoti et al. [21], consists of a set of politically active Twitter accounts. We remove stubs — i.e., accounts that follow only one account and have no followers — resulting in 3,302,362 accounts. Out of those we consider 500 popular ones — with at least 50,000 followers — as candidate facilities, that is, representatives. As remarked in the beginning of section 5, we can extend the proposed framework by considering different metrics for the two terms of the objective function. This corresponds to a practical setting where the agreeability of the selected representatives is measured differently than their affinity to their respective constituents. Specifically, for this dataset we consider the following distance functions.

1. **Facility-Client**: We compute distances between facilities and clients as the length of the shortest path between two Twitter accounts in the undirected follower Twitter graph. 

2. **Facility-Facility**: To compute distances between facilities we use shortest-path distances, as before. We also use Euclidean distances on the spectral embedding with $k$ components, as described by Belkin et al. [5]. We scale the resulting distance matrix so that the average of all entries is equal to that of the shortest-path distance matrix. This way we ensure that the magnitude of $\lambda$ has an equivalent effect using the different metrics.

Congress: We collect roll call voting records from the present US Congress using the public domain congress API. We build a dataset where each row corresponds to a Congress representative and each column is a binary variable representing the issue being voted. Missing values are imputed using class-conditional means, where the classes we consider are the two parties: democrats and republicans. “Present” and “Not voting” votes are considered to be missing. We omit votes where all representatives are missing. We also omit representatives for whom we could not obtain an ideology estimate using the approach described below, or who missed too many votes. For this dataset, we use the Euclidean distance between the vectors corresponding to each of the representatives. To make this experiment closer to a plausible practical scenario, we restrict half of the facilities to be democrats and the other half to be republicans. In addition, clients are served by the closest facility of the same party.

Domains: We combined the domain-related data described in the work of Bakshy et al. [3] with the Twitter dataset. The set of facilities consisted of 469 domains hosting the news sources most often shared on the Facebook social network. The client set is comprised of 6,104 of the most politically active Twitter users. We consider two alternatives for computing the distances.

1. **Mentions**: Given a facility $f$ and a client $c$, let $n_{cf}$ be the number of times a tweet by $c$ contains a link to $f$. Then $d(f,c) = (n_{cf} + 1)^{-1}$. To compute the pairwise distances between facilities we do the following. Consider two facilities, $f$ and $g$. Let $S_f$ (respectively $S_g$) be the set of clients that have tweeted a link to $f$ (respectively $g$) at least once. We define $W = \sum_{c \in S_f \cup S_g} \log n_{cf} I(c \in S_f) + \log n_{cg} I(c \notin S_f)$, where $I$ is the indicator function. Then

$$d(f,g) = 1 - \frac{\sum_{c \in S_f \cap S_g} \log n_{cf}}{W}.$$ 

We define $\log 0 = 0$. Since the objective function of recon-$k$-median (Equation (1)) sums over all ordered pairs of facilities, this distance function is in effect symmetric when applied to our problem. Note that this is akin to the Jaccard index for set similarity, but each element is weighted with a measure of its relevance.

2. **Latent**: We construct a matrix $A$ such that $A_{ij}$ is the number of times a tweet by user $j$ contains a link to domain $i$. We compute the singular value decomposition $A = U \Sigma V^T$ and extract the latent representation for both domains and users in the first $9$ components (which account for $50\%$ of the total Frobenius norm of $A$). If $k = 9$, domain $i$ is represented as $U_{i,k}$ and user $j$ as $V_{i,k}$. To compute both facility-facility and facility-client distances we take the Euclidean distances between the corresponding latent representations.

The characteristics of the datasets are summarized in Table 1. For all datasets, in order to compute the objective of recon-$k$-median we take averages instead of the sums of distances. Note that this amounts to scaling both sums, so it is equivalent to setting $\lambda$ to a particular value. The advantage of taking averages is that $\lambda$ has an impact at small values, i.e., at “small” factors of $1$.

**Ground truth polarity scores ($\pi$)**. To measure the polarity scores of the facilities we employ different methods depending on the dataset.

In the case of Twitter and Congress we use the approach described by Barberâ [4], which estimates the ideological leaning of a Twitter account as a real value. For Twitter, we use polarity scores collected at the same time as the follower graph (July 2017). For Congress, we collected the ideological estimates in May 2018. Using this method and the proposed datasets, all the elicited polarity scores are between -3 and 3. We measure the polarity of the chosen representatives as follows. Given a solution $S = \{s_1, \ldots, s_k\}$, let $\pi(s_i)$ denote the estimated polarity of facility $s_i$. We define the polarity of solution $S$ as the sample standard deviation of the set $\{\pi(s_1), \ldots, \pi(s_k)\}$.

For Domains, we use the ideological leaning score associated to each domain as described in the work of Lahoti et al. [21]. These scores were computed roughly as the fraction of interactions (visits or shares) by conservative users, out of total interactions. We translated the scores so that they fall between -0.5 (left) and 0.5 (right). In this scenario, we are interested in choosing less polarized sources. We therefore measure the polarity of the chosen set as the $\ell^2$ norm of the vector $(\pi(s_1), \ldots, \pi(s_k))$. 

---

[The datasets are available at https://doi.org/10.5281/zenodo.2573954]  
[The follower graph corresponds to a snapshot taken in July 2017.]  
[https://github.com/unitedstates/congress/wiki]
Table 1: Summary of the datasets.

<table>
<thead>
<tr>
<th>Name</th>
<th>Number of clients</th>
<th>Number of facilities</th>
<th>Distance functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Twitter</td>
<td>3 302 362</td>
<td>500</td>
<td>Shortest path, Spectral embedding + Euclidean</td>
</tr>
<tr>
<td>Congress</td>
<td>420</td>
<td>420</td>
<td>Euclidean</td>
</tr>
<tr>
<td>Domains</td>
<td>6 104</td>
<td>469</td>
<td>Weighted Jaccard + Mentions, Latent space + Euclidean</td>
</tr>
</tbody>
</table>

Results on Twitter dataset. We run the algorithm on the Twitter dataset setting the number of selected facilities to \( k = 2, 4, 8 \) and \( \lambda = 0.2^i / 10 \) for \( i = 1, \ldots, 6 \). For computing the pairwise distances between facilities we use either the shortest path metric or the spectral embedding with \( \gamma = k, 10, 100 \). Figure 1 illustrates the results. We depict, as a function of \( \lambda \), the polarity of the chosen representatives, measured as described above, along with the \( k \)-median cost of the solution — i.e., the first term of the objective function (2). We run the algorithm 40 times for each setting and report the average and standard deviation bands. We can see that increasing the value of \( \lambda \) leads to significantly less polarized representatives in various cases. The effect is most noticeable for larger values of \( k \), and particularly consistent using the spectral embedding with 100 components. An interesting result on this dataset is that we can achieve significant decreases in polarity without incurring much additional \( k \)-median cost. That is, it is possible to elect a much more agreeable committee — with respect to the chosen polarity measure — without notable loss in representation fidelity.
We run the algorithm on the Congress dataset, using the same configurations as for Twitter. Figure 2 illustrates the results. Here, the decrease in polarity is only clear in the case $k = 2$, for values of $\lambda$ at least 0.8. It should be noted, however, that the voting data and the polarity scores come from completely different sources. It would therefore be interesting to carry out further experiments with these data.

**Results on Domains dataset.** We run the algorithm on the Domains dataset, using the same configurations as for Twitter but considering $k = 16$ as well, as in the case of news sources it is practical to consider larger sets. Figure 3 shows the results, using the distance function defined in Equation (4) ($w$-jaccard) and the latent representation ($\text{latent}$). We run each configuration 80 times and report average results and standard deviation bands. The reduction in polarity is noticeable, in particular using the latent representation with larger values of $k$. For very small sets of news sources (e.g., $k = 2$) the method does not exhibit a reduction in polarity. In order to gain further insight on the impact of the penalty term, we report an example of the news sources that appear more frequently as $\lambda$ increases. Specifically, we take a case where decrease in polarity is noticeable ($\text{latent}, k = 8, \lambda = 0.8$). We then collect the 16 sources that appear the most, and do the same for $\lambda = 0$. The results are shown in Table 2. For each domain, we report the frequency (i.e., the fraction of times it was part of the solution out of the 80 runs), the ideological score and the number of times it was mentioned in the collected tweets.

**Number of iterations.** Even though the time complexity of the local-search algorithm per iteration is not too high, a legitimate concern to be raised is the possibility that it might require a large
We have considered the problem clustering data so as to optimize well as less ideologically-extreme sets of news sources. This work as measured by a well-known method for ideology estimation, as can lead to the choice of less polarized groups of representatives, as well as to attempt to find tight examples to know the possible extent of said improvement. Second, it would be interesting to perform further experiments on similar and other datasets. It is particularly compelling to improve our understanding of how different metrics can interact with known methods for estimating polarization.

6 CONCLUSIONS

We have considered the problem clustering data so as to optimize the total representation cost plus an additive term to penalize disagreement among the chosen representatives. The proposed problem, which we name reconciliation $k$-median, has applications in summarizing data with non-polarized representatives, as well as in electing $k$-member committees that are more likely to reach consensus. We have shown the proposed problem to be $\text{NP}$-hard optimization problem which we employ in our reduction.

Here we prove Lemma 1, which is a key ingredient of the proof of hardness for $\text{recon-}k\text{-median}$. Before we proceed, we provide a definition of Densest $k$-subgraph, a well-known $\text{NP}$-hard optimization problem which we employ in our reduction.

Problem 3. (densest $k$-subgraph — $\text{dks}$) Given a simple graph $G = (V, E)$ with $|V| = n$ and adjacency matrix $A$, and a number $k \in \mathbb{N}$ with $k < n$, find

$$
\min_{x} \quad x^T A x,
$$

subject to $x \in \{0, 1\}^n$ and $x^T 1 = k$.

We can now prove the aforementioned lemma.

Proof of Lemma 1. We proceed by reduction from Densest $k$-subgraph (dks).

Table 2: Top-16 news sources for different values of $\lambda$, using latent representations.

<table>
<thead>
<tr>
<th>Domain</th>
<th>$\lambda = 0$</th>
<th>$\lambda = 0.8$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Frequency</td>
<td>Ideology</td>
</tr>
<tr>
<td>nydailynews.com</td>
<td>1.000</td>
<td>-0.114</td>
</tr>
<tr>
<td>politico.com</td>
<td>1.000</td>
<td>-0.073</td>
</tr>
<tr>
<td>slate.com</td>
<td>1.000</td>
<td>-0.341</td>
</tr>
<tr>
<td>cbsnews.com</td>
<td>0.687</td>
<td>-0.057</td>
</tr>
<tr>
<td>buzzfeed.com</td>
<td>0.687</td>
<td>-0.262</td>
</tr>
<tr>
<td>twitchy.com</td>
<td>0.537</td>
<td>0.469</td>
</tr>
<tr>
<td>westernjournalism.com</td>
<td>0.462</td>
<td>0.450</td>
</tr>
<tr>
<td>9news.com</td>
<td>0.462</td>
<td>-0.016</td>
</tr>
<tr>
<td>politifact.com</td>
<td>0.350</td>
<td>-0.240</td>
</tr>
<tr>
<td>cbsloc.al</td>
<td>0.337</td>
<td>-0.081</td>
</tr>
<tr>
<td>christianpost.com</td>
<td>0.337</td>
<td>0.337</td>
</tr>
<tr>
<td>theatlantic.com</td>
<td>0.312</td>
<td>-0.176</td>
</tr>
<tr>
<td>newrepublic.com</td>
<td>0.312</td>
<td>-0.335</td>
</tr>
<tr>
<td>lifenews.com</td>
<td>0.200</td>
<td>0.483</td>
</tr>
<tr>
<td>6abc.com</td>
<td>0.200</td>
<td>-0.252</td>
</tr>
<tr>
<td>usatoday.com</td>
<td>0.112</td>
<td>-0.064</td>
</tr>
</tbody>
</table>

Table 3: Number of iterations on Twitter dataset (avg/max).

<table>
<thead>
<tr>
<th>Metric</th>
<th>$k = 2$</th>
<th>$k = 4$</th>
<th>$k = 8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shortest path</td>
<td>2.5/4</td>
<td>2.8/5</td>
<td>3.097/7</td>
</tr>
<tr>
<td>Spectral, $\gamma = 2$</td>
<td>2.18/3</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Spectral, $\gamma = 4$</td>
<td>-</td>
<td>2.73/5</td>
<td>-</td>
</tr>
<tr>
<td>Spectral, $\gamma = 8$</td>
<td>-</td>
<td>-</td>
<td>3.13/6</td>
</tr>
<tr>
<td>Spectral, $\gamma = 10$</td>
<td>2.32/4</td>
<td>2.8/6</td>
<td>3.32/7</td>
</tr>
<tr>
<td>Spectral, $\gamma = 100$</td>
<td>2.45/4</td>
<td>2.75/5</td>
<td>2.58/4</td>
</tr>
</tbody>
</table>

number of iterations to converge. Our observations, however, suggest that in practice a small number of iterations — where by iteration we understand the inspection of all candidate changes — are necessary. Table 3 shows the average and maximum number of iterations for the Twitter dataset.

ACKNOWLEDGMENTS

This work was supported by three Academy of Finland projects (286211, 313927, and 317085), and the EC H2020 RIA project "So-BigData" (654024).

APPENDIX

6.1 Hardness results

Here we prove Lemma 1, which is a key ingredient of the proof of hardness for recon-$k$-median. Before we proceed, we provide a definition of Densest $k$-subgraph, a well-known $\text{NP}$-hard optimization problem which we employ in our reduction.

Proof of Lemma 1. We proceed by reduction from Densest $k$-subgraph (dks).
We employ the following notation:

\[ \tilde{A}_{ij} = \begin{cases} A_{ij} & (i,j) \in E \\ 0 & i = j \\ 2 & \text{otherwise.} \end{cases} \]

Notice that this matrix is symmetric, the diagonal (and nothing else) is zero, and since \( \min_{ij} \tilde{A}_{ij} = 1 \) and \( \max_{ij} \tilde{A}_{ij} = 2 \), for all \( i,j \), and \( \ell \)

\[ \tilde{A}_{ij} \leq \tilde{A}_{i,\ell} + \tilde{A}_{\ell,j}. \]

We want to show that if

\[ x = \arg\min_{x \in \{0,1\}^n} x^T \tilde{A} x, \]

then

\[ x = \arg\max_{x \in \{0,1\}^n} x^T A x. \]

We can write

\[ x^T \tilde{A} x = \sum_i \sum_j \tilde{A}_{ij} x_i x_j = 1 \]

whence we have defined

\[ M = |\{(i,j) | x_i x_j = 1 \} \cap E| \]

\[ N = |\{(i,j) | x_i x_j = 1 \} \cap \tilde{E}| \]

That is, \( M \) is the number of pairs in \( x \) with a corresponding edge in \( G \), and \( N \) is the number of pairs in \( x \) without a corresponding edge in \( G \). Note that \( M + N = 2\left(\binom{k}{2}\right) = k^2 - k \).

Similarly, \( x^T A x = \sum_i \sum_j A_{ij} x_i x_j = 1 \), where \( M = k^2 - k - N \).

Suppose \( x \) minimizes MDP but it does not maximize DKS. Then for some \( x' \),

\[ x'^T A x' > x^T A x, \]

which implies

\[ M' > M, \]

or

\[ x'^T A x' = M' + 2N' < M + 2N = x^T \tilde{A} x. \]

This contradicts the initial assumptions, hence if \( x \) minimizes MDP, \( x \) maximizes DKS.

### 6.2 Approximation guarantees

We employ the following notation:

- \( f(S) = \sum_{s \in C} d(s, s(x)) \), where \( s(x) \) is the facility assigned to client \( x \);
- \( g(S) = \frac{1}{2} \sum_{s \in S} \sum_{y \in S} d(x, y) \);
- \( N_S(x) \) is the set of clients served by facility \( s \) in solution \( S \).

**Proof of theorem 3.** From Arya et al. [2], we know there exists a set of \( k \) pairs \( S = \{(o_i, s_j) | i, j \in [k]\} \subset O \times S \) satisfying the following properties:

1. Every \( o \in O \) is considered in exactly one pair.
2. Every \( s \in S \) is considered in two pairs at most.

From the local optimality of \( S \) we have

\[ f(S - s_j + o_i) - f(S) + g(S - s_j + o_i) - g(S) \geq 0, \]

for any \( i,j \in [k] \). Again from Arya et al. [2], we know we can choose the set of pairs \( \Sigma \) such that the following inequality holds:

\[ 5f(O) - f(S) \geq \sum_{(o_i, s_j) \in \Sigma} \left( f(S - s_j + o_i) - f(S) \right). \quad (5) \]

Furthermore, since \( S \) and \( S - s_j + o_i \) differ in one centroid only, we have

\[ g(S - s_i + o_j) - g(S) = \sum_{x \in S - s_j + o_i} d(x, y) - \sum_{x \in S} d(x, y) = \sum_{s \in S} d(o_i, s) - \sum_{s \in S} d(s, s_j). \]

Summing this difference over all \( k \) pairs we get

\[ \sum_{(o_i, s_j) \in \Sigma} (g(S - o_i + s_j) - g(S)) = \sum_{(o_i, s_j) \in \Sigma} \left( \sum_{s \in S} d(o_i, s) - \sum_{s \in S} d(s, s_j) \right). \]

Therefore, we have

\[ 5f(O) + \sum_{(o_i, s_j) \in \Sigma} \sum_{s \in S} d(o_i, s) \geq f(S) + \sum_{(o_i, s_j) \in \Sigma} \sum_{s \in S} d(s, s_j). \]

We consider this inequality, and modify it so that it becomes dependent only on factors of \( f(S), g(S) \), \( f(S) \). To accomplish this, we will consider the set of pairs \( \Sigma \).

First, let us examine the following quantity:

\[ \sum_{(o_i, s_j) \in \Sigma} \sum_{s \in S} d(o_i, s). \]

By the triangle optimality, we have that the sum corresponding to each pair \( (o_i, s_j) \in \Sigma \) is bounded as follows:

\[ \sum_{s \in S} d(o_i, s) \leq \sum_{s \in S} d(o_i, o_h) + d(o_h, s) \leq d(o_i, x) + d(x, s), \]

where \( o_h \) is such that \( N_0(o_h) \cap N_S(s) \neq \emptyset \) and \( x \in N_0(o_h) \cap N_S(s) \). If \( N_S(s) \subseteq N_0(o_i) \), we consider \( d(o_i, s) \leq d(o_i, o_h) + d(o_h, x) + d(x, s) \), choosing \( o_h \) as described below.

For notational convenience, we define

\[ \sigma(o_i) = \sum_{s \in S} d(o_i, o_h) + d(o_h, x) + d(x, s), \]

for each \( (o_i, s_j) \in \Sigma \) — note that each \( o_i \) appears exactly once in the pairs in the set \( \Sigma \), so the pair \( (o_i, s_j) \) is uniquely determined by \( o_i \).

We want to choose the entries of inequality (6) such that

- every entry of the form \( d(o_i, o_h) \) appears twice;
- every entry of the form \( d(o_h, x) \) appears once;
- every entry of the form \( d(x, s) \) appears once.
Algorithm 2 Reverse procedure to choose the replacements

1: for $i = 1, \ldots, k$ do
2: \hspace{1em} $B_i \leftarrow \emptyset$
3: end for
4: for $i = 1, \ldots, k$ do
5: \hspace{1em} $r \leftarrow$ random ordering of $1, \ldots, k$
6: \hspace{2em} for every $j = 1, \ldots, k$
7: \hspace{3em} $B_j \leftarrow B_j \cup \{r(j)\}$
8: end for

To achieve this, we need a set of replacements such that for each $\sigma(o_i)$, the $k - 1$ corresponding replacements contain $k - 1$ distinct entries of the forms $d(o_i, o_h)$, $d(o_h, x)$, $d(x, s)$.

We define a function $\mu$ that maps each $x \in C$ to a pair $(o_i, s_j)$ such that $x \in N_O(s_j) \cap N_O(o_i)$. Since $|N_O(s_j)| \geq k$ and $|N_O(o_i)| \geq k$ by assumption, we can choose a subset $\hat{C} \subseteq C$ of $k^2$ points such that each $s \in S$, as well as each $o \in O$, appears in $k$ of the $k^2$ pairs associated to the chosen points. We now define $k$ multisets $B_j = \{i \mid \forall x \in \hat{C}$ such that $\mu(x) = (o_i, s_j)\}$. That is, each $s_j \in S$ corresponds to a multiset $B_j$, which contains the index of each $o_i \in O$ with which $s_j$ shares a client in $C$, including repetitions.

For $i = 1, \ldots, k$ we define the multisets $K_i = \{i, i, \ldots, i\}$, $|K_i| = k$ and $M = \bigcup_i K_i$. Consider the procedure outlined by Algorithm 2. It is clear that such a procedure can produce any partition of the multiset $M$ into $k$ sets containing $k$ elements each. Consider the orderings such that said procedure results in the partition $B_1, \ldots, B_k$. If we reverse the procedure using those orderings, we obtain the desired sequence of replacements. Note that for any entry $d(o_i, s)$ in $\sigma(o_i)$ such that $N_O(s) \subseteq N_O(o_i)$, we can make the rest of the replacements first and then choose $o_h$ from the remaining elements of $O$. Additionally, note that each sum $\sigma(o_i)$ requires us to make only $k - 1$ replacements. Since the above method provides $k$ satisfactory replacements for each sum, we can simply pick $k - 1$ of those. We can now conclude that

- every element of the form $d(o_i, o_j)$ can only appear either in $\sigma(o_i)$ or in $\sigma(o_j)$. Since for all $\sigma(o_i)$, the entry $d(o_i, o_j)$ is unique, each $d(o_i, o_j)$ appears most twice;
- for any element of the form $d(o, x)$, $o \in O$, $x$ is unique;
- for any element of the form $d(s, x)$, $s \in S$, $x$ is unique.

We have established the following inequality

$$\sum_{(o_i, s_j) \in \Sigma} \sum_{s \in S, s \neq s_j} d(o_i, s) \leq 2g(O) + f(O) + f(S). \quad (7)$$

We now examine the quantity

$$\sum_{(o_i, s_j) \in \Sigma} \sum_{s \in S} d(s, j). \quad (8)$$

For every two facilities $s_i, s_j \in S$, either

- $d(s_i, s_j)$ appears in the sum;
- $d(s_i, s_j)$ does not appear in the sum. This implies 1) $s_i, s_j$ are not part of any pair in $\Sigma$ and 2) some other $s_k$ appears in two pairs (remember that there are exactly $k$ pairs and each $s \in S$ takes part in at most two). Since the summation $\sum_{s \in S} d(s, s)$ is computed over all elements of $S$, we have two appearances of $d(s_k, s_j)$ and other two of $d(s_k, s_j)$ (one for each summation corresponding to the two pairs involving $s_k$). Since $d(s_k, s_j) + d(s_k, s_j) \geq d(s_i, s_j)$, we can replace $d(s_k, s_j) + d(s_k, s_j)$ with $d(s_i, s_j)$ in the sum (8) and the inequality still holds.

This implies the following:

$$\sum_{(o_i, s_j) \in \Sigma} \sum_{s \in S} d(s, j) \geq g(S).$$

Combining this with inequality (5) we can state

$$10f(O) + 2 \sum_{(o_i, s_j) \in \Sigma} g(S - s + o_i) - 2f(S) - 2g(S) \geq 0. \quad (9)$$

Considering each cluster in both $S$ and $O$ to have at least $\lceil 2\ell \rceil k$ points, we can extend the result to account for $\lambda$. To do this, we apply the replacements leading to inequality (7) $\lceil 2\ell \rceil$ times to the sum $2\ell \sum_{(o_i, s_j) \in \Sigma} \sum_{s \neq s_j} d(o_i, s)$ and obtain

$$2\ell \sum_{(o_i, s_j) \in \Sigma} \sum_{s \neq s_j} d(o_i, s) \leq 4\ell g(O) + f(O) + f(S)$$

To see this, note that for every entry of the form $d(o_i, x)$ or $d(s_j, x)$, $x$ is still unique. Combining this with inequality (9), we have

$$10f(O) + 4\ell g(O) + f(O) + f(S) \geq 2f(S) + 2g(S),$$

which implies

$$11f(O) + 4\ell g(O) \geq f(S) + g(S).$$

Therefore,

$$\max\{11, 4\ell\} \cdot \text{cost}(O) \geq \text{cost}(S). \quad \Box$$

Theorem 4. Let $(F, C, k, \lambda)$ be an instance of problem 1. Let $S = \{s_1, \ldots, s_k\}$ be a solution output by algorithm 1 and let $O = \{o_1, \ldots, o_k\}$ be an optimal solution. Then

$$\text{cost}(S) \leq (\lambda k + 5) \cdot \text{cost}(O) \quad (10)$$

The proof, which we omit due to space constraints, is similar to that of theorem 3.
REFERENCES


