Revision Letter

Turn Waste into Wealth: On Efficient Clustering and Cleaning over Dirty Data

Paper ID: TKDE-2024-07-1462

We sincerely appreciate all the important and constructive suggestions from the Associate Editor and all the reviewers. We address all the concerns as follows. To better distinguish the response to different reviewers and the according revision, the following color scheme will be used hereinafter: violet for Associate Editor, magenta for Reviewer #1, blue for Reviewer #2 and red for Reviewer #3.

Associate Editor

Reviewer 1 highlights the strong motivation and efficiency of the GDORC method but raises concerns about notation complexity, parameter sensitivity across datasets, and incomplete experimental details for the LDORC method. A typographical error after Definition 6 was also noted.

Reply: (1) We simplify the complex notations in Algorithms 1 and 2, with detailed reply in Reviewer 1's W1 section. (2) Next, we address parameter sensitivity across datasets using a combination of Poisson distribution and the posterior Silhouette Score. Detailed response can be found in Reviewer 1's W2. (3) In Reviewer 1's W3, we also include the missing experimental details for LDORC, as shown in Figures 11-14 in Section VI. (4) Finally, we correct the typographical error in Definition 6 and thoroughly proofread the main text. The corrections are mentioned in Reviewer 1's D1.

Reviewer 2 commends the novelty and validation of GDORC but suggests adding more real-world applications, sampling details, and expanding the experimental analysis to explain its performance advantages.

Reply: (1) We add real-world examples of dirty data causing issues in downstream tasks like navigation, urban planning, and Points of Interest [6], [22] , detailed in our response to Reviewer 2's W1. (2) In the reply to Reviewer 2's W2, we provide sampling device's details and explain the rationale for choosing GPS data. (3) We expand the experimental analysis by explaining the root cause behind GDORC's performance in the reply to Reviewer 2's W3.

Reviewer 3 finds the manuscript meaningful and well-organized but recommends strengthening the motivation with concrete examples, reorganizing the related work section, and adding visualizations to clarify the theory-heavy Section III.

Reply: (1) In the reply to Reviewer 3's W1, we enhance the motivation section by discussing the impacts of dirty data in Section I-A. (2) In the reply to Reviewer 3's W2, we reorganize the paper structure, placing the related work section immediately after the introduction. (3) Lastly, in Reviewer 3's W3, we add an illustration Figure 3 to clarify Theorem 1.

Reviewer 1

W1: *The notations used for different variables in GDORC are concerning. The method involves numerous variables related to different types of cells and points, making it difficult to follow, particularly when reading the pseudo-code.*

Reply: Thank you for the constructive and valuable suggestion. To handle the notation issue, We have redesigned the notation used in GDORC, especially in Algorithm 1 and Algorithm 2, along with the necessary adjustments in the text, in Section V-C2, Page 7. For instance, the core points set is altered from $\mathcal{C}$ to $\mathcal{P}_c$. Besides, the noise cells set is altered from $\mathcal{N}_u$ to $\mathcal{U}_N$. With these changes, the notation for different types of the point set is simplified from $\mathcal{C}$ and $\mathcal{P}$ to $\mathcal{P}$, and the notation for different types of cells is simplified from $\mathcal{N}_u$ and $\mathcal{U}$ to $\mathcal{U}$.

W2: *GDORC's two different parameters, eta and epsilon, have limitations when implementing GDORC on different datasets. As a clustering-based method, GDORC requires two parameters, epsilon and eta, for its application. Figures 8-11 in the experimental section demonstrate that these two parameters can significantly influence the repair accuracy. However, these parameters are closely correlated with the distribution of the datasets, implying that they may vary significantly across different datasets. Inappropriate parameter settings might misclassify actual noise points as normal data points or result in an incorrect number of clusters. Without prior knowledge of the dataset distribution, users might need to spend considerable time determining the appropriate parameter settings.*

Reply: We add two new paragraphs in Section VI-D, Page 9 to discuss a way to handle varying parameters across datasets and select appropriate parameter settings.

GDORC shares the parameters η and ε with DBSCAN [8], which may affect clustering, especially with dirty data in Figure 8 (b). Misclassifications and errors can occur if these parameters are mis-tuned, leading to inefficiencies during tuning across different datasets. To mitigate this, we first use the cumulative density function of Poisson distribution [32] to estimate a coarse range of candidates η and ε. Then, we select the parameter combination among the candidates which achieve the highest Silhouette Score. The Silhouette Score [29] effectively evaluates clustering separation, density and compactness. A higher Silhouette Score means a better clustering quality.

To validate the parameter selection process, Figures 9 and 10 show the Silhouette Score on the GPS and Foursquare datasets, with the selected parameter combinations marked by the red line. For example, in Figure 9, the selected parameters for the GPS dataset is $\varepsilon = 2.2 \times 10^{-5}$ and $\eta = 18$, determined by the highest Silhouette Score. These selected parameters are further evaluated in Figures 11-14, in terms of clustering quality and repairing accuracy. For example, in Figures 11(a) and 11(b), when ε approaches the selected setting $\varepsilon = 2.2 \times 10^{-5}$, Purity and NMI can achieve a high score more than 0.95 and 0.8, respectively. Moreover, as shown in Figure 11(c), the repair error decreases to 0.6×10^{-4} when $\varepsilon = 2.2 \times 10^{-5}$. In contrast, for other parameters, e.g. $\varepsilon = 0.6 \times 10^{-5}$, the repair error may be as large as 1.2×10^{-4}. It demonstrates that GDORC can indeed achieve good performances with such selected parameters.

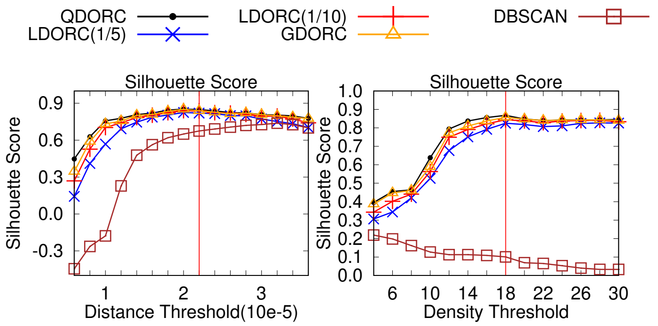


Fig. 9: Silhouette Score on GPS dataset

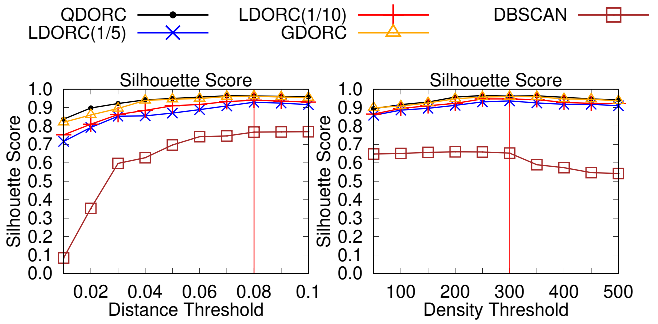


Fig. 10: Silhouette Score on FourSquare dataset

W3: *The experimental setup for the LDORC method is incomplete. In the conference version, it is noted that LDORC includes an additional threshold parameter, tau, which can impact repair accuracy. However, the current paper lacks a detailed explanation of this experimental setting, leaving out crucial information that could affect the interpretation of the results.*

Reply: We include the exact same setting for LDORC as the conference version and detailed explanation of the setting with interpretation of results in Section VI-C in Page 8.

Indeed, LDORC contains an additional parameter τ, which measures the maximum distance between a leader point and a follower point. We use the default setting for τ in LDORC, which is $\varepsilon/5$. To demonstrate the influence of τ on the accuracy of LDORC, we further evaluate LDORC with $\tau = \varepsilon/10$, exactly the same as the LDORC settings in the conference version [32]. As shown in Figures 11 and 12, reducing τ from $\varepsilon/5$ to $\varepsilon/10$ improves the purity and NMI scores while reducing the repair error. This is because the smaller distance between leaders and their followers tightens the approximation. However, the time cost of LDORC increases as τ decreases, since more leaders are selected, expanding the neighboring search space.

D1: *In the paragraph after definition 6, it appears to be a typographical error regarding the point type of p_j. p_j should be classified as a border point rather than a noise point.*

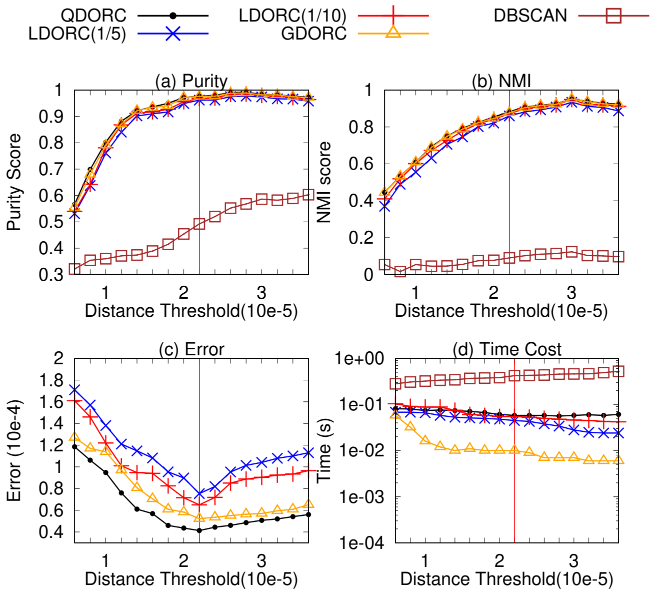


Fig. 11: Varying ε on GPS data with η=10

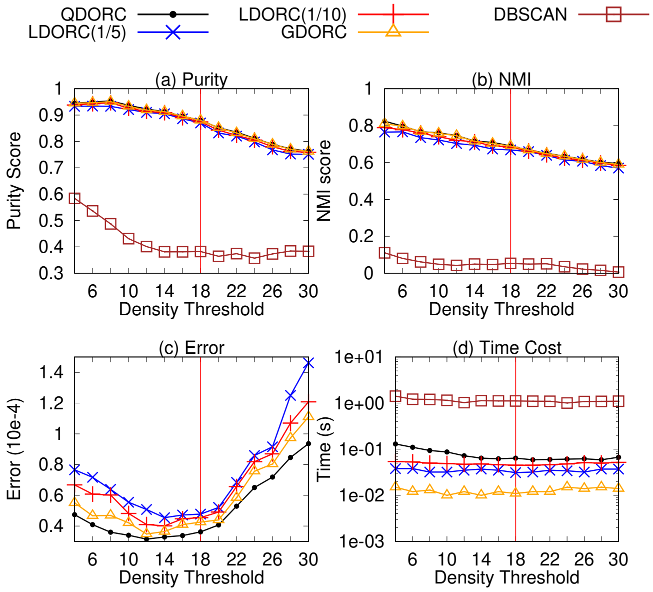


Fig. 12: Varying η on GPS data with $\varepsilon = 1.4e^{-5}$

Reply: Thank you for pointing out the careless mistake. We fix the typographical error in the paragraph after Definition 6, Page 6. The revise sentence is "In the grid-based method, a border point p_j is repaired ..." We also carefully check other sentences in the entire manuscript, especially the ones explaining important concepts or providing explanations for illustrations and examples.

Reviewer 2

W1: *The background introduction would be better if more details are included. For instance, more real-world applications and relevant literature citations can be mentioned to make it more practical.*

Reply: Thanks for the constructive suggestions. We include more details in the background introduction. Specifically, we

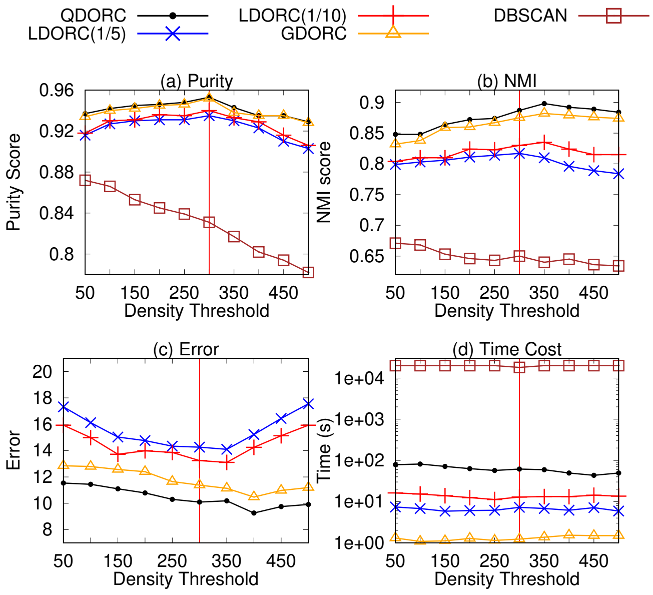


Fig. 13: Varying η on FourSquare data with $\varepsilon = 0.1$

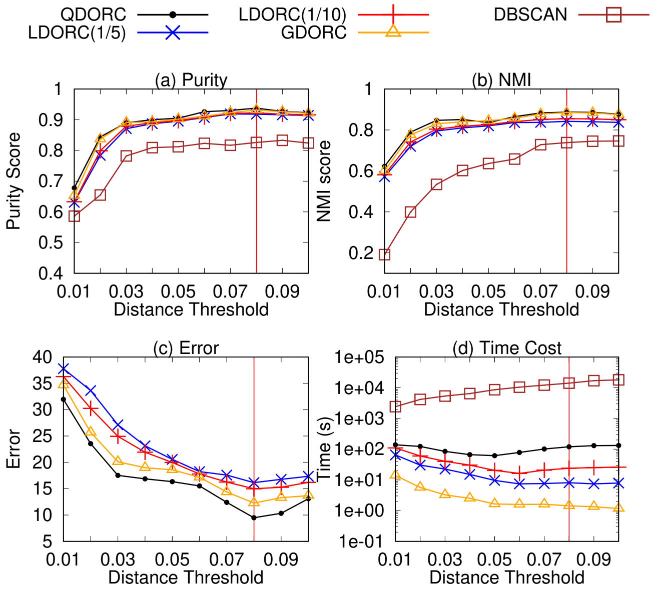


Fig. 14: Varying ε on FourSquare data with $\eta = 300$

mention more real-world examples and relevant literature citations in Section I, Page 2, to make it more practical.

Clustering and cleaning data provide significant benefits in various real-world applications. For example, in personal attribute analysis [30], cleaning GPS data improves clustering accuracy, enhancing the interpretation of regional studies. In healthcare [24], cleaning GPS data enhances the clustering of lifelog data, ensuring better accuracy in identifying individuals' activity patterns. In location-based services [35], cleaning GPS data ensures correct coordinates, which is crucial for accurate clustering in tasks such as attendance tracking and urban planning. As shown in Figures 11-12 of application study, cleaning enhances the accuracy and reliability of location-based services like attendance tracking.

W2: *The collected data will be more convincing if the details of the sampling devices and the sampling scenarios are provided. The reason why the GPS data is selected should be claimed and it's better to explain the reason with concrete applications.*

Reply: To make the collected data more convincing, we further provide more details of the sampling devices and sampling scenarios, and explain the reason why GPS data is selected with concrete applications in Section VI-G, Page 10.

In terms of sampling scenarios, the GPS dataset is collected from various mobile phones in a real-world student attendance tracking context. In this scenario, students use their mobile phones to obtain their locations and then submit the locations via QR codes in classrooms. If a student's location is near the classroom building, the student is considered to be attending the class. Besides, in terms of sampling devices, we gather data points from various models of smartphones and tablets.

The reason why GPS data is selected is because its innate spatial characteristics and relatively low collecting quality. On the one hand, GPS data reflect the physical locations in the real world, which is ideal for distance-based clustering. For example, the locations of the students who attend the class form a cluster, since these students are all in the classroom. On the other hand, GPS data are often collected with low precision from mobile phones, and are often accompanied by noise and location shifting errors. For example, due to network fluctuations and the low locating precision of mobile phones, the collected locations may shift far from the classroom locations. Therefore, GPS data is useful for evaluating our proposed repairing methods.

Figure 8 provides a concrete example in the aforementioned scenario. Students are located in 3 different classrooms as in Figure 8(a). Due to network fluctuations and low locating precision, some locations shift, leading to poor clustering results in Figure 8(b). Fortunately, our proposal can repair these errors and obtain better clustering performance in Figure 8(c). We utilize an check-in accuracy score to evaluate the impact of erroneous location points and the effectiveness of the data repair process. The check-in accuracy measures the amount of students with correct attendance checks. As demonstrated in Figure 17, with a higher check-in accuracy score, GDORC can reduce the misidentification due to dirty data through repairing.

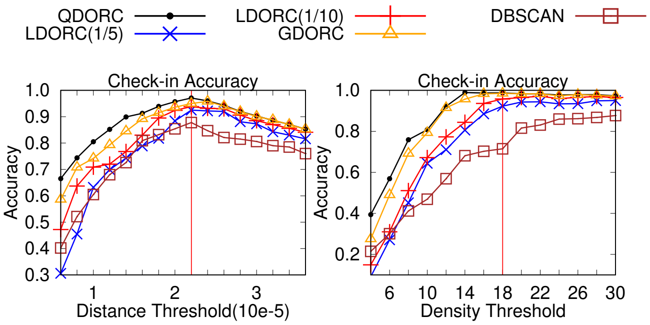


Fig. 17: Check-in application accuracy on GPS dataset

W3: *The analysis of the experiments could be more detailed and briefly explain the root cause of why the proposed method performs better than the baselines in some aspects.*

Reply: As suggested, we provide more detailed analysis of the experiments, such as for varying clustering parameters in Section VI-D. We also explain the root cause of why the proposed method performs better than the baselines, in Section VI-D, Page 9.

The root cause of GDORC's superior performance compared to QDORC and DBSCAN lies in its use of a grid structure, which transforms point-level neighboring search into cell-level search. By utilizing the grid, GDORC prunes points far from the target point, as explained in Equation 7 and the grid-based LP solution in Section V-B, avoiding unnecessary distance calculation between every pair of points. The grid also serves as a natural index for spatial proximity, accelerating the neighbor search process without sacrificing clustering quality, as detailed in Section V-C1 and Algorithm 1.

Compared to LDORC, GDORC outperforms in efficiency by eliminating the need for distance calculations between each point and the leaders, a step required by LDORC. While LDORC uses leaders for approximation, it still computes distances between data points and their assigned leaders, which can be computationally expensive. In contrast, GDORC utilizes the grid for pruning and indexing, significantly improving efficiency without compromising clustering quality.

Reviewer 3

Q1: *I remain somewhat unconvinced how one can confidently quantify better data quality after the repair process — How can one possibly know the repaired data point is (and therefore the resulting clusters are) more correct? — but this seems to be beyond the scope of this manuscript.*

Reply: We sincerely appreciate all the important and constructive suggestions. As suggested, we discuss how to quantify better data quality and clustering results in Section VI-B, Page 8, for three different cases of datasets.

For clean datasets, such as the Foursquare dataset, we introduce random errors to create dirty data. To assess repair correctness, we compare the repaired data with the ground truth, calculating the repair error (RMSE distance) between the repaired and true data, as shown in Figures 11-16. For clustering correctness, we consider the clustering results on the clean data as the ground truth and use NMI and Purity to compare the clustering accuracy after repair. Figures 11-14 display these scores, with GDORC outperforming DBSCAN.

For datasets like GPS with naturally embedded dirty data, we manually adjust inaccurate points to their correct positions. Similar to the aforesaid clean datasets with ground truth, we again evaluate repair correctness by RMSE and clustering performance with NMI and Purity.

For datasets where labeling the ground truth is challenging, we use the Silhouette Score as a measure of clustering accuracy. The Silhouette Score [29] evaluates cluster cohesion and separation by comparing the similarity of each point to its own cluster versus other clusters. Figures 9 and 10 demonstrate the superior performance of GDORC over DBSCAN using the Silhouette Score, validating that the clusters produced by the repair methods are more accurate.

Q2: *Why is there such a long time gap (nearly 10 years) between the publication of the preliminary version of DORC and the current submission?*

Reply: We defer the current submission in order to implement it in Apache IoTDB [34], an open-source, commodity database widely used in IIoT (Industrial Internet of Things) for efficient data storage and management. We add new Section VI-A, namely "System Deployment", in Page 8, to explain the reason for implementing our methods into Apache IoTDB.

Apache IoTDB [34] is an open-source, commodity time-series database widely used in IIoT (Industrial Internet of Things) for efficient data storage and management. GPS readings considered in the motivation example and experiments can be naturally stored in the database, and expected to be cleaned and clustered. We have implemented all three versions of DORC methods into Apache IoTDB. An example SQL instruction for calling GDORC in IoTDB is as follows:

```
SELECT DORC('method'='G','eta'='300',
'eps'='8e-5','dim'='2')
FROM root.device.s0, root.device.s1;
```

where η is set to be 300, and ε is set to be 8e-5. Additional constraints such as time interval can be added with other instructions in SQL instructions such as "where". With DORC methods, industrial partners can utilize our methods for efficient repairing and clustering cater to their needs.

W1: *Introduction: It helps that you use the real-world example of GPS data to ground an otherwise very theoretical paper in a practical context when your algorithm would be used. This motivation and contextualization could have been stronger by, e.g. in the first paragraph indicating what are the (potentially grave) consequences of such dirty data.*

Reply: We explain the consequences of the dirty data in the first paragraph of Section I, Page 1.

The consequences of such dirty data are significant and far-reaching. It can lead to incorrect clustering results, misidentifying patterns and relationships in the data. This has serious implications in real-world applications. For example, in location-based services such as attendance tracking (see Section VI-G), misidentifications due to uncleaned data can prevent proper check-ins when clustering. In other applications like bike-sharing location optimization [3] and traffic anomaly detection [35], accurate location data is crucial for analysis and planning. A large portion of uncleaned inaccurate data undermines the reliability of the analysis and leads to wasted resources, incorrect decisions, and potential system failures.

repeated

too complicated sentence, check the whol revision

again, focus on cleaning and clusterin ¿why

W2: *Organization: The subsections are not too long. Colors are used consistently across related figures. Perhaps the context provided by Section VI "Related Work" would be more helpful near the beginning instead.*

Reply: Following the suggestion, we move the "Related Work" section closer to the beginning, as Section II in Page 2, right after after the Introduction.

W3: *Readability: Just as it is nice to have examples and illustrating figures in Sections I and II, the proof of Theorem 1 in Section III is hard to follow without a visualization. This* *entire section is very heavy on theory, which can be harder to follow for any reader who is not in the practice of reading or writing proofs.*

Reply: In Section IV (the original Section III), page 3, we provide a proof sketch of Theorem 1 in Page 3 and a visualization in Figure 3, to make it easy to follow. Besides, we also move other heavy theoretical proofs in Section IV to Appendix A online for a better comprehension.

online

The reduction for the DORC problem considers the NP-complete Vertex Cover problem, as shown in Figure 3 (a). Each edge (u_i, u_j) in graph $G(U, E)$ is mapped to two points v_i and v_j with distance ε between them. For each vertex u_i, we introduce $\eta - l - 2$ points at $r_i \in \mathcal{P}$, one at $q_i \in \mathcal{P}$, and η points at $o_i \in \mathcal{P}$, where l is the number of edges connected to u_i and $l < \eta < n$. The distances between edges, shown in Figure 3 (b), are set to ε, with all other pairs having distances $\delta(*, *) \gg \varepsilon$. This transformation is polynomial, upon which we prove NP-hardness of DORC.

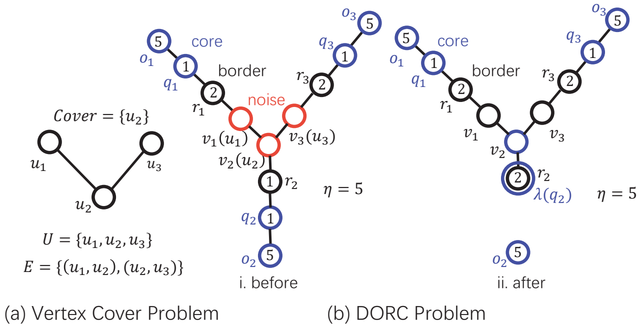


Fig. 3: Transformation for DORC problem

Turn Waste into Wealth: On Efficient Clustering and Cleaning over Dirty Data

Kenny Ye Liang, Yunxiang Su, Shaoxu Song, *Member, IEEE,* and Chunping Li

***Abstract*—Dirty data commonly exist. Simply discarding a large number of inaccurate points (as noises) could greatly affect clustering results. We argue that dirty data can be repaired and utilized as strong supports in clustering. To this end, we study a novel problem of clustering and repairing over dirty data at the same time. Referring to the minimum change principle in data repairing, the objective is to find a minimum modification of inaccurate points such that the large amount of dirty data can enhance clustering. We show that the problem is NP-hard and can be formulated as an integer linear programming (ILP) problem. A constant factor approximation algorithm GDORC is devised based on grid, with high efficiency. In experiments, GDORC has great repairing and clustering results with low time consumption. Empirical results demonstrate that *both the clustering and cleaning accuracies* can be improved by our approach of repairing and utilizing the dirty data in clustering.**

***Index Terms*—Data repairing, Density-based clustering**

I. Introduction

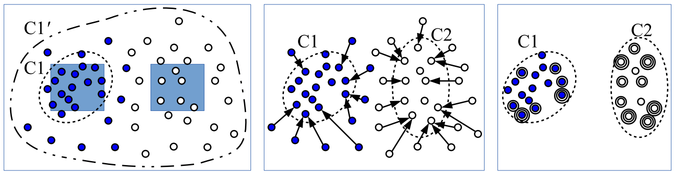
Density-based clustering can successfully identify noises (see a survey in [21]). However, rather than a small proportion of noise points, real data are often dirty with a large number of inaccurate points [14]. For instance, a (very) large portion of GPS data are inaccurate, especially in the indoor environment with weak signals. According to our experiments (in Section VI), 1000 out of 3706 (about 27%) GPS readings are inaccurate. The consequences of such dirty data are significant and far-reaching. It can lead to incorrect clustering results, misidentifying patterns and relationships in the data. This has serious implications in real-world applications. For example, in location-based services such as attendance tracking (see Section VI-G), misidentifications due to uncleaned data can prevent proper check-ins when clustering. In other applications like bike-sharing location optimization [3] and traffic anomaly detection [35], accurate location data is crucial for analysis and planning. A large portion of uncleaned inaccurate data undermines the reliability of the analysis and leads to wasted resources, incorrect decisions, and potential system failures.

W1

The large amount of noise points are simply discarded, if we directly apply the existing density-based clustering approaches, e.g., the well-known DBSCAN [8]. With too much information loss, clustering results could be dramatically affected (see examples below).

A. Motivation

Instead of discarding dirty data, we argue that dirty data can be repaired and utilized as strong supports in clustering. Study [33] shows that performing data cleaning and clustering at the same time leads to better performance.



(a) clusters in dirty data (b) repairing and clustering (c) repaired data

Fig. 1: Clustering with repairing over dirty data

We propose repairing dirty data based on *density* information, inspired by its effectiveness in identifying noisy data in density-based clustering. The approach simultaneously repairs data according to its density during clustering, allowing *both tasks to benefit* (as shown in Section VI).

Following the minimum change principle in data repairing [37]—where changes made during repairs are minimized—the objective of simultaneous repairing and clustering is to achieve the smallest possible data repair, enabling effective utilization of all data for clustering. The rationale is based on the practical intent to minimize errors, as dirty data, such as inaccurate GPS readings , are typically close to their true values. Guided by this objective, we define the problem as an optimization problem, namely *Density-based Optimal Repairing and Clustering* (DORC).

Example 1. We illustrate an example of GPS data in Figure 1, showing readings from two nearby buildings marked as blue and white points. Cluster C1 represents high-density, precise points from a building with strong GPS signal, while C2 shows inaccurate points from a building with weak GPS signal requiring focused repair. Density-based clustering methods like DBSCAN either generate cluster C1 (with high-density parameters) or C1′ (with low-density parameters). The white points from the second building fail to form a separate cluster, either being treated as noise in C1 or merged with blue points in C1′. Points identified as noises by DBSCAN in Figure 1(a) strongly suggest the need to focus on refining C2. (See more examples in Figure 8 in Section VI)

In this study, we propose to repair the inaccurate data points during clustering. For example, as shown in Figure 1(b), an arrow $(a \to b)$ denotes that a point is repaired from location a to location b. Since points are concentrated after repairing, two clusters are formed in Figure 1(c).

The problem of clustering with repairing, however, is non-trivial. Simply repairing noise points to the closest clusters

Corresponding author: Dr. S. Song, Associate Professor, Tsinghua University, Beijing, China. https://sxsong.github.io/ E-mail: sxsong@tsinghua.edu.cn
Y. Liang, Y. Su, and C. Li are with Tsinghua University, Beijing, China.

is not sufficient, e.g., repairing all the noise points to C1 in Figure 1 does not help in identifying the second cluster C2. It should be considered that dirty points may possibly form clusters with repairing (i.e., C2). □

Clustering and cleaning data provide significant benefits in various real-world applications. For example, in personal attribute analysis [30], cleaning GPS data improves clustering accuracy, enhancing the interpretation of regional studies. In healthcare [24], cleaning GPS data enhances the clustering of lifelog data, ensuring better accuracy in identifying individuals' activity patterns. In location-based services [35], cleaning GPS data ensures correct coordinates, which is crucial for accurate clustering in tasks such as attendance tracking and urban planning. As shown in Figures 11-12 of application study, cleaning enhances the accuracy and reliability of location-based services like attendance tracking.

W1

B. Contribution

Our proposed DORC techniques complement existing repair methods. Compared to these methods (details in Section II), DORC has two major advantages: (1) it does not rely on external knowledge of integrity constraints or rules, and (2) it utilizes the density information embedded within the data, which many methods overlook.

This paper focuses on the grid-based GDORC algorithm, which simultaneously repairs and clusters with a constant factor approximation. Compared to QDORC and LDORC [33], GDORC achieves comparable accuracy but with reduced time, as it searches data cells instead of individual points.

Our major contributions in this paper are summarized as:

(1) We formalize the DORC problem of simultaneous clustering and repairing (presented in Section III). In particular, *no additional parameters* are introduced for DORC besides the density and distance thresholds η and ε for clustering.

(2) We provide the hardness analysis and tractable special case for the DORC problem, and formulate DORC as an ILP problem (detailed in Section IV).

(3) We devise a *constant factor* approximation method, GDORC, based on a grid that decomposes the data space into cells (described in Section V). The correctness and the complexity of the algorithm are analyzed in Proposition 6 and Proposition 7, respectively. Furthermore, the approximation ratio of GDORC is examined in Proposition 8.

(4) We conduct extensive experiments on real datasets (illustrated in Section VI). The results demonstrate that our proposed GDORC approach has both high clustering and repairing performances, while keeping time costs low.

II. Related Work

W2

A. Clustering

Density-based clustering is widely used (see [21]). Given a distance threshold ε and a minimum neighborhood size η (MinPts), DBSCAN [8] categorizes points into cores, borders, and noise. In this study, we adhere to these established settings. Numerous algorithms elaborate on the concept of DBSCAN. Various versions of DBSCAN have been developed, including the incremental version [7], the distributed version NG-DBSCAN [25] and DBSCAN-MS [38], and the dynamic version [10]. DBSCAN++ [16] aims to reduce computational cost by estimating densities for subsets of points. DBSVEC [36] integrates support vectors to reduce unnecessary range queries, enhancing efficiency. ANYDBC [26] compresses data into subsets and labels objects based on connected components to reduce time cost. DBSCAN-DIST [1] combines the concepts of density and minimax distance to formalize density-based clustering by a loss function.

Several methods focus on stream data. DISC [20] introduces a density-based incremental strategy with elaborations on points' types, including ex-cores and neo-cores. DenForest [19] presents a concept of ”nostalgic core” and uses spanning trees for better cluster quality and efficiency.

All these studies focus primarily on identifying noise/non-noise points, while the large amount of dirty data (identified as noises) are still not employed to form clusters. In contrast, our proposal considers cleaning and clustering with dirty data. While outlier detection (see [15] for a survey) identifies dirty points, data repairing further modifies these points for correction. Indeed, our proposal incorporates density-based DBSCAN. In this sense, data repairing and outlier detection are complementary.

B. Repairing

Besides the minimum modification model [37], which we also adopt, the deletion model [4] is commonly used to identify the minimum removal of dirty data. In this context, DBSCAN can be seen as a deletion-based technique, eliminating dirty data. However, simply discarding large amounts of dirty data as noise using existing clustering methods is not ideal and can significantly affect clustering results.

Recent data repair methods, such as HOLOCLEAN [5], IMR [39], RSR [23], and MISC [32], address data errors in various ways. HOLOCLEAN integrates multiple data sources to detect and repair anomalies, but relies on external data and integrity constraints. IMR is an incremental regression method for time-series data, which is less effective for static data. RSR uses regular expressions to repair sequence and token values, but requires predefined rules.

Our study, as mentioned in the introduction, focuses on the density information within the data, without relying on external knowledge or constraints. Our approach complements these techniques when additional information, such as master data or rules, is available. MISC repairs data errors on selected attributes but is less suited for datasets like GPS data, where errors impact both longitude and latitude simultaneously. Therefore, we perform full-feature repairs directly on GPS data, where MISC's advantages are less applicable.

III. Problem Statement

A. Clustering

Consider a set of data points $\mathcal{P}$. Let $\delta : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_0^+$ be a distance function, satisfying nonnegativity $\delta(p_i, p_j) \geq$

0, identity of indiscernibles $\delta(p_i, p_j) = 0$ iff $p_i = p_j$, and symmetry $\delta(p_i, p_j) = \delta(p_j, p_i)$, where $p_i, p_j \in \mathbb{R}^d$.

Two data points $p_i, p_j \in \mathcal{P}$ are said to be in ε-neighborhood if $\delta(p_i, p_j) \le \varepsilon$. We denote $C(p_i) = \{p_j \in \mathcal{P} \mid \delta(p_i, p_j) \le \varepsilon\}$ the set of ε-neighbors of p_i, where $p_i \in C(p_i)$.

Definition 1 (Core points, Border points, Noise points)**.** *Given a distance threshold ε (Eps) and a density threshold η (called MinPts), a point p_i with $|C(p_i)| \ge \eta$ is considered as a* core point*. A* border point *has $|C(p_i)| < \eta$ but is in ε-neighborhood of some core point. The other points, which are neither core points nor border points are* noise points.

B. Repairing

A *repair* over a set of points is a mapping $\lambda : \mathcal{P} \to \mathcal{P}$. We denote $\lambda(p_i)$ the location of point p_i after repairing. The ε-neighbors of $\lambda(p_i)$ after repairing is $C_\lambda(p_i) = \{p_j \in \mathcal{P} \mid \delta(\lambda(p_i), \lambda(p_j)) \le \varepsilon\}$

Following the minimum change principle in data cleaning that we prefer a repair close to the input [37], the repairing cost $\Delta(\lambda)$ is defined as

$$\Delta(\lambda) = \sum_{i=1}^{n} w(p_i, \lambda(p_i)), \tag{1}$$

where $w(p_i, \lambda(p_i))$ is the cost of repairing a point p_i to the new location $\lambda(p_i)$. In this study, we consider the distance of point locations before and after repairing [2] as the cost function, with $w(p_i, \lambda(p_i)) = \delta(p_i, \lambda(p_i))$.

C. DORC Problem

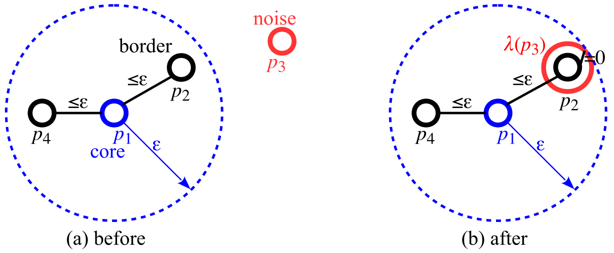
As mentioned in the introduction, by simply relaxing parameters in DBSCAN, the diffusion of nearby clusters may force them to combined (in Figure 1(a), C2's points are either ignored or merged). We propose to utilize the dirty (noise) points for clustering by repairing. That is, the noise points are repaired and clustered as either core points or border points. In other words, for each repaired $\lambda(p_i)$, either itself or one of its ε-neighbors has a ε-neighborhood size greater than MinPts η. Since a cluster is uniquely determined by its core points [8], the repairing process with identification of core points (in the repair results) outputs the clustering results as well.

Problem 1. *Given data points $\mathcal{P}$, distance threshold ε and density threshold η, the Density-based Optimal Repairing and Clustering (*DORC*) problem is to find a repair λ such that (1) the repairing cost $\Delta(\lambda)$ is minimized, and (2) for each repaired $\lambda(p_i)$, either it is a core point $|C_\lambda(p_i)| \ge \eta$, or it has $|C_\lambda(p_j)| \ge \eta$ for some p_j with $\delta(\lambda(p_i), \lambda(p_j)) \le \varepsilon$.*

It is worth noting that multiple points may be repaired to the same "physical" location, having $\lambda(p_i) = \lambda(p_j)$.

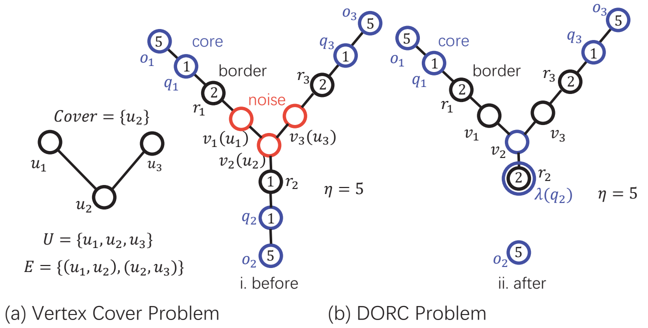
Example 2. Consider a clustering density requirement $\eta = 3$. As shown in Figure 2(a), point p_1, whose $|C(p_1)| = |\{p_1, p_2, p_4\}| = 3$, is a core point. Points p_2 and p_4 in ε-neighborhood of p_1 are border points. Point p_3, not in ε-neighborhood of any point, is considered as a noise point.

Figure 2(b) shows a possible repair λ, where p_3 is moved to the location of p_2, i.e., $\lambda(p_3) = p_2$. Point p_2 with $\lambda(p_2) = p_2$ remains unchanged (and similarly for p_1, p_4). There are two points in the location of p_2 after repairing (red and black concentric circles). We have $C_\lambda(p_2) = C_\lambda(p_3) = \{p_1, p_2, p_3\}$. That is, p_2 and p_3 upgrade to core points by repairing. □



(a) before
(b) after

Fig. 2: Example of repairing



(a) Vertex Cover Problem
(b) DORC Problem

Fig. 3: Transformation for DORC problem

IV. Problem Transformation

In this section, we first provide hardness analysis and a tractable special case of the DORC problem. Then, we illustrate how to formulate the DORC problem as an ILP problem, and thus existing ILP solvers can be directly applied.

A. Hardness and Tractable Special Case

As demonstrated in Example 1, simply assigning noise points to the closest clusters does not work. The DORC problem present significant challenges.

Theorem 1. *The* DORC *problem is* NP*-hard.*

Proof Sketch: The reduction for the DORC problem considers the NP-complete Vertex Cover problem, as shown in Figure 3 (a). Each edge (u_i, u_j) in graph $G(U, E)$ is mapped to two points v_i and v_j with distance ε between them. For each vertex u_i, we introduce $\eta - l - 2$ points at $r_i \in \mathcal{P}$, one at $q_i \in \mathcal{P}$, and η points at $o_i \in \mathcal{P}$, where l is the number of edges connected to u_i and $l < \eta < n$. The distances between edges, shown in Figure 3 (b), are set to ε, with all other pairs having distances $\delta(*, *) \gg \varepsilon$. This transformation is polynomial, upon which we prove NP-hardness of DORC. □

R3W3

When the density threshold is lowest [1], i.e., MinPts $\eta = 2$, the DORC problem can be efficiently solved. This case is meaningful and common, as each cluster is a connected

[1] MinPts $\eta = 1$ is meaningless where each point is trivially a core point.

component of core points within ε-neighborhoods. Specifically, $\eta = 2$ requires each core point to have at least one other ε-neighbor, making every point in a core point's ε-neighborhood also a core point. Consequently, clusters are connected components of core points, and noise points are singleton components.

Proposition 2. *For $\eta = 2$, there is a* PTIME *algorithm for solving the* DORC *problem.*

B. ILP Formulation

Consider variable $x_{ij}, 0 \leq x_{ij} \leq 1$. Let $x_{ij} = 1$ denote that point p_i is repaired to location p_j after repairing, i.e., $\lambda(p_i) = p_j$; otherwise, $x_{ij} = 0$. Obviously, a point can only be repaired to one location, having

$$\sum_{j=1}^{n} x_{ij} = 1. \tag{2}$$

The weight w_{ij} for x_{ij} is defined as the corresponding cost of repairing p_i to p_j, $w_{ij} = w(p_i, p_j)$.

After repairing, there may exist multiple points p_i being repaired to the location of a point p_j. The new ε-neighborhood count of location p_j is

$$c_j = |\{p_i \in \mathcal{P} \mid \delta(\lambda(p_i), p_j) \leq \varepsilon\}| = \sum_{i=1}^{n} x_{ij} + \sum_{k=1}^{n}\sum_{i=1}^{n} h_{jk} x_{ik}, \tag{3}$$

where $h_{jk} = 1$ denotes that locations p_j and p_k are in ε-neighborhood; otherwise, $h_{jk} = 0$. That is, c_j counts the total number of points located in p_j and all of its ε-neighbors p_k, after repairing.

Let $y_j = 1$ denote that p_j has ε-neighbor count no less than η, i.e., core location; otherwise, $y_j = 0$. It follows

$$\frac{c_j}{\eta} \geq y_j \geq \frac{c_j - \eta + 1}{n}, \tag{4}$$

where $n = |\mathcal{P}|$ is the total number of points. It specifies that $y_j = 1$ iff $c_j \geq \eta$; and $y_j = 0$ iff $c_j < \eta$.

The repairing should ensure eliminating all noise points. In other words, a point is either a core point or a border point (which is a neighbor of a core point). More precisely, for any location p_j with at least one point retained after repairing, i.e., $x_{kj} = 1$ for some k, it is required that either this point or one of its neighbors belongs to core points (with ε-neighborhood size no less than η). We have

$$y_j + \sum_{i=1}^{n} y_i h_{ij} \geq \frac{1}{n}\sum_{k=1}^{n} x_{kj}. \tag{5}$$

In other words, for any j with $x_{kj} = 1$ for some k, it requires either $y_j = 1$ or some other $y_i = 1$ such that p_i is in ε-neighborhood with p_j ($h_{ij} = 1$).

Given the constraints in Formulas 2, 4 and 5, the DORC problem is formulated as the following ILP problem.

$$\begin{aligned}
\text{minimize} \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} x_{ij} & & \\
\text{subject to} \quad & \sum_{j=1}^{n} x_{ij} = 1, & & 1 \leq i \leq n \\
& c_j - \eta y_j \geq 0, & & 1 \leq j \leq n \\
& y_j n - c_j \geq 1 - \eta, & & 1 \leq j \leq n \\
& y_j + \sum_{i=1}^{n} y_i h_{ij} - \frac{1}{n}\sum_{k=1}^{n} x_{kj} \geq 0, & & 1 \leq j \leq n \\
& x_{ij}, y_j \in \{0,1\} \qquad 1 \leq i \leq n, & & 1 \leq j \leq n
\end{aligned} \tag{6}$$

Existing ILP solvers can be directly applied to compute the optimal solutions. It returns not only a repair x_{ij} but also a set of core points $\lambda(p_i) = p_j$ with $y_j = 1$ after repairing.

Proposition 3. *The optimal solution $\mathbf{x}^{\text{ILP}}, \mathbf{y}^{\text{ILP}}$ of* ILP *forms an optimal repair λ^{ILP} with the minimum repairing cost*

$$\Delta(\lambda^{\text{ILP}}) = \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} x_{ij}^{\text{ILP}},$$

where $\lambda^{\text{ILP}}(p_i) = p_j$ iff $x_{ij}^{\text{ILP}} = 1, 1 \leq i \leq n, 1 \leq j \leq n$.

Example 3 (Example 2 continued)**.** Consider again the example of 4 points in Figure 2 with clustering density requirement $\eta = 3$. We show how the repair λ, with $\lambda(p_3) = p_2$ in Figure 2(b), corresponds to the feasible solution $\mathbf{x}$ to the ILP, where $x_{11} = x_{22} = x_{44} = 1$ and $x_{32} = 1$.

For the location of p_2, we have $c_2 = 2 + 1 = 3$ by Formula 3. It follows $y_2 = 1$ according to Formula 4. In other words, all the points in the location of p_2 after repairing are core points, i.e., p_2 and p_3 as indicated in Example 2.

For the location of p_3, since there is no point retained after repairing, having $\sum_{k=1}^{4} x_{k3} = 0$, Formula 5 is satisfied. The solution corresponding to the repair λ satisfies all the constraints in Formula 6 and is a feasible solution of ILP. □

We can show that the DORC problem is always solvable.

Proposition 4. *For $\eta < n$, a feasible solution to the* ILP *problem always exists.*

Finding the optimal solution is non-trivial, as shown by the hardness analysis in Theorem 1. While existing QDORC and LDORC from [33] provide approximate solutions in quadratic and linear time, respectively, QDORC becomes time-consuming as data size grows, and LDORC yields less accurate results with high repair costs due to its reliance on leaders as representatives. To address this, we propose an efficient grid-based approximation approach with guaranteed ratios. We also evaluate and compare QDORC, LDORC, and our GDORC method in the experiments.

V. Approximation with Grids

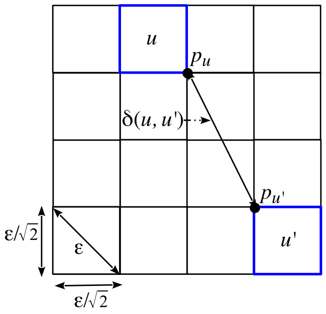


Fig. 4: Distance between Cells u and u'

Departing from the ILP formulation in Section IV, we present a grid-based method GDORC for clustering and cleaning in this section. The key idea behind GDORC is to update point types by cells for high efficiency. To clearly explain our grid-based method, we first provide preliminary knowledge about grid-based method in Section V-A. We propose a new grid-based LP solution in Section V-B. We introduce the algorithm GDORC about clustering and repairing in Section V-C. We further give approximation analysis in Section V-D.

A. Preliminary of Grid-based Method

In this subsection, we first introduce four definitions about grid-based method. Grid-based method [9] is widely used in density-based clustering for approximation and accelerating computation speed. In our approximated method GDORC, a grid $\mathcal{G}$ is imposed on the data space with cells, where grid and cell are defined as below.

Definition 2 ([9], [11], [13], [31] Grid $\mathcal{G}$ and Cell u)**.** *A grid $\mathcal{G}$ is a set of close-packed identical cells on the data space $\mathbb{R}^d$. A cell u is a d-dimensional hypercube with diagonal length ε. And the side length of a cell is $\frac{\varepsilon}{\sqrt{d}}$ for d-dimension.*

Figure 4 presents a data space $\mathbb{R}^2$ with points p_u and $p_{u'}$. We introduce grid $\mathcal{G}$ with side length $\frac{\varepsilon}{\sqrt{2}}$ on this data space.

Definition 3 (Noise Cell and Set of Noise Cells $\mathcal{U}_{\mathcal{N}}$)**.** *Cell u is a noise cell if u contains at least one noise point. $\mathcal{U}_{\mathcal{N}}$ denotes the set of noise cells.*

Figure 5 presents a grid with five points p_1, p_2, p_3, p_4, p_5. Since p_5 has less than η neighbors, cell u_2 with noise point p_5 is a noise cell and the noise cell set is $\mathcal{U}_{\mathcal{N}} = \{u_2\}$. If a non-empty cell u contains at least η points, then all points in cell u are classified as cores since each point has more than η neighbors. However, if a non-empty cell u contains fewer than η points, determining whether a point p in u is core point requires further calculation.

To take advantage of the proximity of cells, as mentioned in [9], we further define the distance between two cells in Definition 4 and the ε-neighbors of a cell in Definition 5.

Definition 4 (Minimum Distance of cells u and u', $\delta(u, u')$)**.** *Minimum distance of cells u and u' is defined as $\delta(u, u') = \inf_{x\in u, y\in u'} \delta(x, y)$, where x, y symbol for geometric points on the data space $\mathbb{R}^d$.*

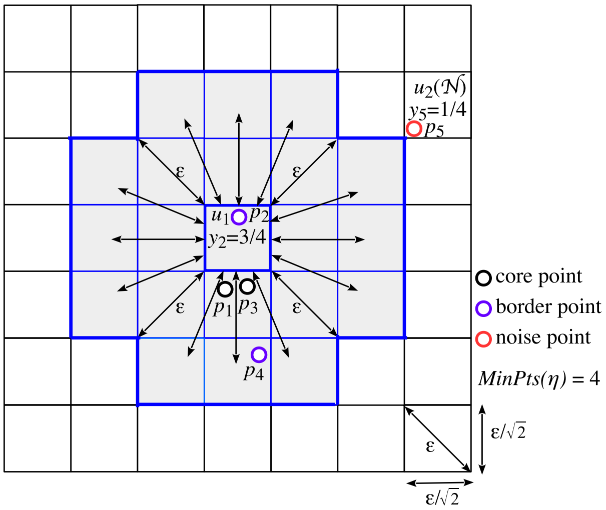


Fig. 5: ε-neighbor Cells of u_1 and Noise Cell $u_2(\mathcal{N})$

Take the 2-dimensional data space in Figure 4 as an example for better readability. We choose two points $p_u, p_{u'}$, from cells u and u' respectively, with minimum distance. Then we compute the distance of $\delta(p_u, p_{u'})$ as the minimum distance between cells u and u', i.e., $\delta(u, u') = \delta(p_u, p_{u'})$.

Definition 5 (ε-neighbor Cells of u)**.** *u and u' are ε-neighbors with each other, if $\delta(u, u') \le \varepsilon$. We denote $C(u) = \{u' \in \mathcal{U} | \delta(u', u) \le \varepsilon, u' \ne u\}$ the set of ε-neighbors of cell u.*

In Figure 5, each arrow with length ε means the distance between u_1 and other gray cells. We can find that the minimum distance between u_1 and all these gray cells are less than ε. Thus, the gray cells are ε-neighbor cells of u_1.

B. Grid-Based Method with LP Solution

Building on the preliminary discussion of grid-based method in Section V-A, we are now prepared to utilize this method to solve the DORC problem. The DORC problem is modeled as an ILP problem in Section IV, and can be relaxed to a LP problem by changing the integer constraints in Formula 6 to $0 \le x_{ij} \le 1, 0 \le y_j \le 1$, as presented in the conference version [33].

Specifically, we define boolean valued y_j to continuous valued $y_j = \min(\frac{\sum_{k=1}^{n} h_{jk}}{\eta}, 1)$ in range $[0, 1]$, symbolizing the ratio of $|C(p_j)|$ to η where $C(p_j)$ denotes the set of ε-neighbors of p_j. Note that $y_j = 1$ iff point p_j is a core point. When $y_j < 1$, at least $\eta(1 - y_j)$ additional ε-neighbor points are needed to make p_j a core point.

$$\begin{aligned} & x^{\text{LP}}_{ii} = 1, \quad i = 1, \dots, n \\ & x^{\text{LP}}_{ij} = 0, \quad i = 1, \dots, n, j = 1, \dots, n \\ & y^{\text{LP}}_{j} = \begin{cases} 1 & \text{if } \sum_{k=1}^{n} h_{jk} \ge \eta \\ \frac{\sum_{k=1}^{n} h_{jk}}{\eta} & \text{if } \sum_{k=1}^{n} h_{jk} < \eta \end{cases} \quad j = 1, \dots, n. \end{aligned} \tag{7}$$

Different than the quadratic time and the linear time methods, the grid-based method can search neighbors by grid for high efficiency in the process of solving the LP problem.

Compared to QDORC and LDORC, grid-based method does not need to traverse all points when searching neighborhood $C(p_i)$ for point p_i, and thus reduce the time cost. Detailed introduction and analysis is carried out in Section V-C and Section V-D. Here is an example of utilizing grid-based method to provide LP solution.

Example 4. For the border point p_2 in Figure 5, we first locate the cell u_1 containing p_2 and search ε-neighbor cells of u_1, i.e., the gray cells marked in Figure 5. For each point in $C(u_1)$, i.e., p_1, p_2, p_3, p_4, the distance from p_i to p_2 is calculated and compared with ε. If $\delta(p_2, p_i) \leq \varepsilon, i = 1, 2, 3, 4$, then $h_{2i} = 1$; otherwise, $h_{2i} = 0$. Since each arrow in Figure 5 represents ε distance, we can simply get the proximity information of p_2, i.e., $h_{21} = h_{22} = h_{23} = 1, h_{24} = 0$. We can obtain $y_2 = 3/4$ referring to Formula 7. Note that we do not concern the points outside $C(u_1)$, i.e., p_5 in this process. □

When repairing, we notice that the grid-based method can skip unnecessary repairs using the relaxed y_j defined in Formula 7. To clearly define cases where repairs can be unnecessary, we first introduce the noise set and the noise neighbors, and then specify the conditions under which repairs are not required.

Definition 6 (Set of Noise $\mathcal{N}$ and Set of Noise ε-neighbors for p_j, $\mathcal{N}(p_j)$)**.** *$\mathcal{N}$ denotes the set of noises. For a point p_j that is classified as noise or border, $\mathcal{N}(p_j)$ represents the set of points p_i that are ε-neighbor of p_j, i.e., $\mathcal{N}(p_j) = \{p_i \in \mathcal{N} | \delta(p_j, p_i) \leq \varepsilon\}$.*

In the grid-based method, a border point p_j is repaired, by means of moving other noise points p_i into location p_j. There is a possible situation where p_i and p_j are in ε-neighborhood. In this case, p_i does not need to be moved into location p_j, and p_j should be repaired by moving other noise points (except p_i). The possible case is formalized by the following proposition.

Proposition 5. *When selecting noise point p_i to repair p_j,i.e.,$|\mathcal{N} \setminus \mathcal{N}(p_j)| \geq (1 - y_j)\eta$, noise point p_i does not need to be repaired from location p_i to location p_j if the distance between them is less than ε, i.e., $\delta(p_i, p_j) \leq \varepsilon$.*

C. Algorithm GDORC

Based on the preliminary in Section V-A and the grid-based LP solution in Section V-B, we are now ready to solve the DORC problem by our proposed method GDORC. In this section, we present two components, i.e., Algorithm 1 and Algorithm 2, of the proposed GDORC in Section V-C1 and Section V-C2 respectively. We analyze the correctness and time complexity of GDORC in Section V-C3.

1) INITIALIZATION*:* Algorithm 1 presents the pseudocode for obtaining all noise points $\mathcal{N}$, set of core points $\mathcal{P}_c$, set of noise cells $\mathcal{U}_\mathcal{N}$, set of noise points $\mathcal{N}(u)$ in cell u and noise ε-neighbor points $\mathcal{N}(p)$ for each non-core point.

Algorithm 1 determines core points and non-core points, and calculates the initial y values of points as defined in Formula 7 on cell-level from Lines 1-15. In Lines 1-2, Algorithm 1 traverses all cells and calculates the number of points in each cell. Then, it traverses every point in each cell in Line 3 to calculate the y value for each point. Here, points in cells containing more than η points no longer need to find neighbors, since they can already be identified as core points, having $y = 1$. For points in cells containing less than η points, Algorithm 1 looks at their neighboring cells in Line 8, and traverses all points in neighboring cells to calculate the number of neighbor points in Line 9. The corresponding y values are updated in Lines 10-15 until all possible neighboring points are traversed or $y = 1$. At this point, for point p_i having $y_i < 1$, it can either be a border point or a noise point.

Algorithm 1: INITIALIZATION($\mathcal{P}$,$\mathcal{U}$,ε,η)

```
Data: Set of data points $\mathcal{P}$, set of non-empty cells $\mathcal{U}$,
      distance threshold $\varepsilon$ and density threshold $\eta$
Result: Set of data points $\mathcal{P}$ with initialized x and y,
        set of noise points $\mathcal{N}$, set of core points $\mathcal{P}_c$,
        set of noise points $\mathcal{N}(u)$ in cell $u$, set of noise
        cells $\mathcal{U}_\mathcal{N}$ and set of noise $\varepsilon$-neighbors $\mathcal{N}(p)$
        for point $p$
1  for each $u_i \in \mathcal{U}$ do            // initialize on cell level
2      let $|u_i|$ be number of points $p$ in cell $u_i$;
3      for each $p_i \in u_i$ do
4          if $|u_i| \geq \eta$ then
5              $y_i := 1$, $\mathcal{P}_c := \mathcal{P}_c \cup \{p_i\}$;
6          else
7              $y_i := |u_i|/\eta$;
8              for each $u_j \in C(u_i)$ do
9                  for each $p_j \in u_j$ s.t. $\delta(p_j, p_i) \leq \varepsilon$ do
10                     $y_i := y_i + \frac{1}{\eta}$;
11                     if $y_i = 1$ then
12                         $\mathcal{P}_c := \mathcal{P}_c \cup \{p_i\}$
13                         break;
14         if $y_i < 1$ then
15             $\mathcal{N} := \mathcal{N} \cup \{p_i\}$;
16 for each $p_i \in \mathcal{P}$, $y_i < 1$ do  // initialize on point level
17     let $u_i \in \mathcal{U}$ be the cell containing $p_i$;
18     for each $p_k \in u_k$, $\delta(p_i, p_k) \leq \varepsilon, u_k \in C(u_i)$ do
19         if $p_k \in \mathcal{N}$ then
20             $\mathcal{N}(p_i) := \mathcal{N}(p_i) \cup \{p_k\}$;
21         if $p_k \in \mathcal{P}_c$ then
22             $\mathcal{N} := \mathcal{N} \setminus \{p_i\}$
23 for each $p_i \in \mathcal{P}$ do           // initialize supporting sets
24     if $p_i \in \mathcal{N}$ then
25         let $u_i \in \mathcal{U}$ be the cell containing $p_i$;
26         $\mathcal{U}_\mathcal{N} := \mathcal{U}_\mathcal{N} \cup \{u_i\}$
27         $\mathcal{N}(u_i) := \mathcal{N}(u_i) \cup \{p_i\}$;
28 return x, y, $\mathcal{N}$, $\mathcal{N}(u)$, $\mathcal{U}_\mathcal{N}$, $\mathcal{N}(p)$, $\mathcal{P}_c$
```

Then, Algorithm 1 traverses all points again to determine the actual point type of non-core points and maintain the noise neighbor set $\mathcal{N}(p)$ for each point p in Lines 16-22. In Line 18, all neighboring points in neighboring cells for each point is traversed. Noise neighbor sets are generated in Line 20. Noise points are identified in Line 22.

Lastly, Algorithm 1 initializes the set of noise cells $\mathcal{U}_\mathcal{N}$ and the set of noise points $\mathcal{N}(u)$ in cell u in Lines 23-27 after all points' point types are determined.

```
Algorithm 2: GDORC Repair($\mathcal{P}, \mathcal{U}, \varepsilon, \eta$)
Data: Set of data points $\mathcal{P}$ with initialized x and y, set
      of noise locations $\mathcal{N}$, set of noise locations
      $\mathcal{N}(u)$ in cell $u$, set of noise cells $\mathcal{U}_\mathcal{N}$ and set of
      noise $\varepsilon$-neighbors $\mathcal{N}(p)$ for point $p$, set of core
      points $\mathcal{P}_c$
Result: A set of core and border locations with $y_j$ in
        y and the corresponding repairing $x_{ij}$ in x
1  while $\mathcal{N} \neq \emptyset$ do      // repair while noise points exist
2      let $p_j \in \mathcal{P}$ be the point with maximum $y_j$ and
         $y_j < 1$, contained in cell $u_j$ ;
3      if $|\mathcal{N} \setminus \mathcal{N}(p_j)| \geq (1 - y_j)\eta$ then
4          repeat
5              let $u_i \in \mathcal{U}_\mathcal{N}$ be with minimum $\delta(u_j, u_i)$,
                 containing noise point $p_i$ ;
6              if $\delta(p_i, p_j) > \varepsilon$ then
7                  $x_{ii} := 0, x_{ij} := 1, y_i := 1$;
8                  $\mathcal{P}_c := \mathcal{P}_c \cup \{p_i\}$;
9              $\mathcal{N} := \mathcal{N} \setminus \{p_i\}$, $\mathcal{N}(u_i) := \mathcal{N}(u_i) \setminus \{p_i\}$;
10             if $\mathcal{N}(u_i) = \emptyset$ then
11                 $\mathcal{U}_\mathcal{N} := \mathcal{U}_\mathcal{N} \setminus \{u_i\}$;
12         until $(1 - y_j)\eta + |\mathcal{N}(p_j)|$ times;
13         $y_j := 1, \mathcal{P}_c := \mathcal{P}_c \cup \{p_j\}$;
14         $\mathcal{N} := \mathcal{N} \setminus \{p_j\}$, $\mathcal{N}(u_j) := \mathcal{N}(u_j) \setminus \{p_j\}$;
15         if $\mathcal{N}(u_j) = \emptyset$ then
16             $\mathcal{U}_\mathcal{N} : \mathcal{U}_\mathcal{N} \setminus \{u_j\}$;
17     else                          // no sufficient noises retain
18         for $p_i \in \mathcal{N}$ , contained in cell $u_i$ do
19             let $u_k \in \mathcal{U} \setminus \mathcal{U}_\mathcal{N}$ be with minimum
                 $\delta(u_k, u_i)$, containing $p_k \in \mathcal{P}_c$;
20             $x_{ii} := 0, x_{ik} := 1, \mathcal{N} := \mathcal{N} \setminus \{p_i\}$;
21 return y, x
```

W1

2) GDORC Repair: Algorithm 2 presents the pseudocode for repairing all the noise points in the data space. For each repair, Algorithm 2 first consider the point $p_j \in \mathcal{P}$ with the largest y_j in Line 2, since the point with largest y_j has the most potential to become a core point. Then, to repair p_j into a core point, at least another $(1-y_j)\eta+|\mathcal{N}(p_j)|$ noise points are needed to support p_j referring to Formula 7 and Proposition 5. If there are sufficient noise points (Line 3), Algorithm 2 look at the nearest noise cell u_i containing at least one noise point p_i. If $\delta(p_i, p_j) \leq \varepsilon$, there is no need to repair p_i according to Proposition 5. Otherwise, we repair p_i to p_j in Line 7, i.e., assign $x_{ii} = 0$ and $x_{ij} = 1$. After repairing p_i into p_j, set of noise points $\mathcal{N}$, set of core points $\mathcal{P}_c$, set of noise cells $\mathcal{U}_\mathcal{N}$, and noise neighbors $\mathcal{N}(p_j)$ are updated in Lines 8-11. When p_j becomes a core point after repairing, status are also updated in Lines 13-16.

When there are no sufficient noise points left for repairing p_j into a core point, i.e., $|\mathcal{N} \setminus \mathcal{N}(p_j)| < (1 - y_j)\eta$, all the remaining noise points p_i will be repaired into their nearest core points in Lines 17-20.

Example 5. Figure 5 shows 5 points with density threshold $\eta = 4$ separated into different cells. From searching neighbor cells, we can easily find that point p_5 in cell u_2 is a noise point since it has less than 4 neighbors in the ε-neighborhood. As a result, $\mathcal{N} = \{p_5\}$ and $\mathcal{U}(\mathcal{N}) = \{u_2\}$. By Formula 7 with searching neighbor cells, we can obtain that $y_1 = 1, y_2 = 3/4, y_3 = 1, y_4 = 3/4, y_5 = 1/4$. Then we choose point p_2 with the maximum $y_2 = 3/4 < 1$ and find the cell u_1 containing p_2, as mentioned in Lines 2 of Algorithm 2. In order to repair p_2 into a core point, we should make sure that $y_2 = 1$. At least one additional noise point should be repaired to the location of p_2. Thus, we select a noise cell u_2 from $\mathcal{U}(\mathcal{N})$ which is closest to u_1, and get the noise point p_5.Then p_5 is repaired into the location of p_2 with $x_{52} = 1$ and $x_{55} = 0$.

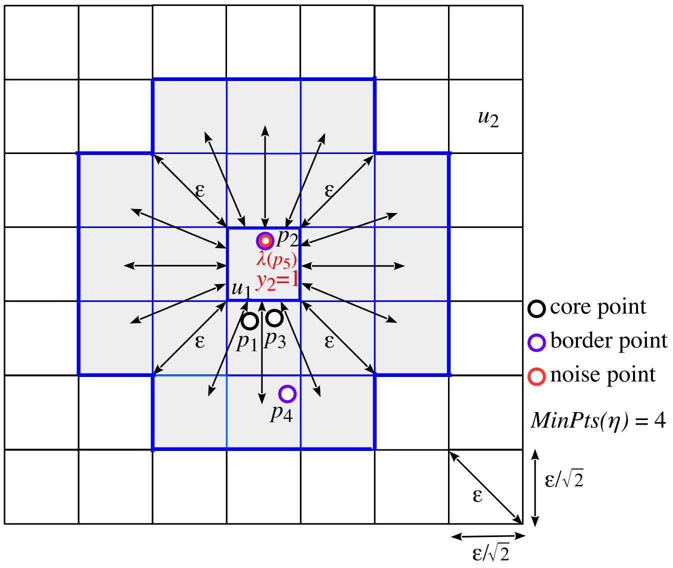


Fig. 6: Example of repairing with LP solution based on Grid-method

Consequently, in Figure 6, the final solution is $x_{11} = x_{22} = x_{33} = x_{44} = 1, x_{52} = 1, y_1 = y_2 = y_3 = 1, y_4 = 3/4$, and the others are 0. The repair λ is $\lambda(p_5) = p_2$ (other 4 points are unchanged) and the core points after repair are $\lambda(p_1)$, $\lambda(p_2)$, $\lambda(p_3)$ and $\lambda(p_5)$ (located in the core locations of p_2). In addition, cell u_2 is empty after repair and $u_2 \notin \mathcal{U}$. □

3) Algorithm Analysis: As stated in Algorithm 2, GDORC repairs a p_i into another location p_j each time. We show the correctness of each repair that no other new noises will be introduced during each repair by the following proposition.

Proposition 6. *Consider repairing point p_i into another point p_j, i.e., $\lambda(p_i) = p_j$. Let $\mathcal{N}_0$ denote the noise set before repair, and let $\mathcal{N}$ denote the noise set after repair. We have $\mathcal{N} \subset \mathcal{N}_0$.*

For a grid $\mathcal{G}$ on the data space $\mathbb{R}^d$ with n points, we denote N_m as the maximum number of points in a cell, N_c as the number of core cells, N_s as the number of noise cells, N as the number of cells, and $|\mathcal{N}|$ as the number of noise points. The complexity of GDORC can be obtained as follows.

Proposition 7. *The time complexity of the grid-based* GDORC *method is* $\mathrm{O}(nN_m + N\eta n)$*, with* $\mathrm{O}(nN_m)$ *being the time complexity of* Algorithm 1 (Initialization) *and* $\mathrm{O}(N\eta n)$

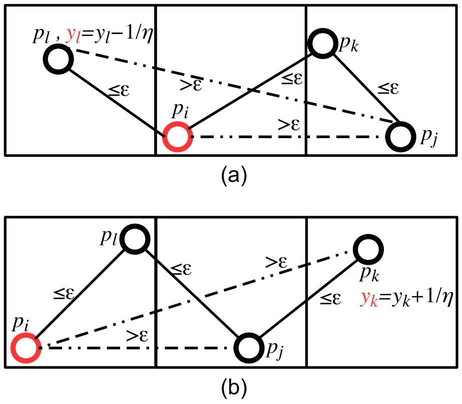


Fig. 7: Cases for updating y_l and y_k for points p_l and p_k

being the time complexity of Algorithm 2 (GDORC Repair).

D. Approximation Performance

Let $\delta_{\max}, \delta_{\min}$ respectively denote the maximum and minimum distances between any two data points in set $\mathcal{P}$. The approximation performance of the grid-based method can be obtained as follows.

Proposition 8. Algorithm 2 (GDORC Repair) *returns a feasible solution to the* ILP *problem, and is a factor-$\alpha\eta$ approximation with $\alpha = \frac{\delta_{\max}}{\delta_{\min}}$.*

Special Case of Connected Components: For the special case of $\eta = 2$ in Proposition 2, the proposition is as follows.

Proposition 9. *For $\eta = 2$, Algorithm 2 (*GDORC*) is a factor-α approximation with $\alpha = \frac{\delta_{\max}}{\delta_{\min}}$.*

VI. Experiments

Experimental evaluation answers the following questions: (1) *By handling various dirty data, can it form more accurate clusters?* (2) *How does the approach scale?*

A. System Deployment

Apache IoTDB [34] is an open-source, commodity time-series database widely used in IIoT (Industrial Internet of Things) for efficient data storage and management. GPS readings considered in the motivation example and experiments can be naturally stored in the database, and expected to be cleaned and clustered. We have implemented all three versions of DORC methods into Apache IoTDB. An example SQL instruction for calling GDORC in IoTDB is as follows:

```
SELECT DORC('method'='G','eta'='300',
'eps'='8e-5','dim'='2')
FROM root.device.s0, root.device.s1;
```

where η is set to be 300, and ε is set to be 8e-5. Additional constraints such as time interval can be added with other instructions in SQL instructions such as "where". With DORC methods, industrial partners can utilize our methods for efficient repairing and clustering cater to their needs.

Q2

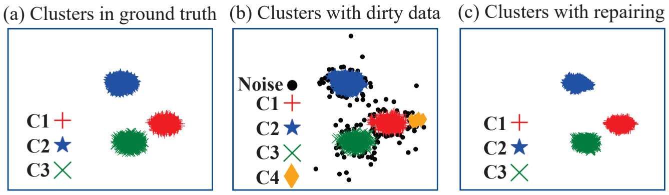


Fig. 8: Examples of clusters in (a) manually labeled GPS clean data, (b) GPS dirty data, (c) GPS data repaired

B. Experimental Settings

Our programs are implemented in Java and all the experiments run on a machine with Intel(R) Core(TM) i7-10750H CPU(2.60GHz), 16 GB RAM.

We have two real-world datasets to evaluate our proposed GDORC: GPS and Foursquare. The GPS dataset is a manually collected dataset containing 3706 real-world GPS data points, where dirty data are naturally embedded. Figure 8(b) visualizes the original GPS dataset. Foursquare dataset is a clean public dataset containing up to 400,000 check-in data points. Note that during the experiment, we find that DBSCAN is incapable of finishing in 6 hours when the data size is over 200,000.

For clean datasets, such as the Foursquare dataset, we introduce random errors to create dirty data. To assess repair correctness, we compare the repaired data with the ground truth, calculating the repair error (RMSE distance) [17] between the repaired and true data, as shown in Figures 11-16. For clustering correctness, we consider the clustering results on the clean data as the ground truth and use NMI [28] and Purity [27] to compare the clustering accuracy after repair. Figures 11-14 display these scores, with GDORC outperforming DBSCAN.

For datasets like GPS with naturally embedded dirty data, we manually adjust inaccurate points to their correct positions. Similar to the aforesaid clean datasets with ground truth, we again evaluate repair correctness by RMSE and clustering performance with NMI and Purity.

For datasets where labeling the ground truth is challenging, we use the Silhouette Score as a measure of clustering accuracy. The Silhouette Score [29] evaluates cluster cohesion and separation by comparing the similarity of each point to its own cluster versus other clusters. Figures 9 and 10 demonstrate the superior performance of GDORC over DBSCAN using the Silhouette Score, validating that the clusters produced by the repair methods are more accurate.

R3Q1

C. Comparison with Baseline

We compare our proposed GDORC with other baselines, including QDORC, LDORC, and DBSCAN. LDORC contains an additional parameter τ, which measures the maximum distance between a leader point and a follower point. We use the default setting for τ in LDORC, which is $\varepsilon/5$. To demonstrate the influence of τ on the accuracy of LDORC, we further evaluate LDORC with $\tau = \varepsilon/10$, exactly the same as the LDORC settings in the conference version [32]. As shown in Figures 11 and 12, reducing τ from $\varepsilon/5$ to $\varepsilon/10$

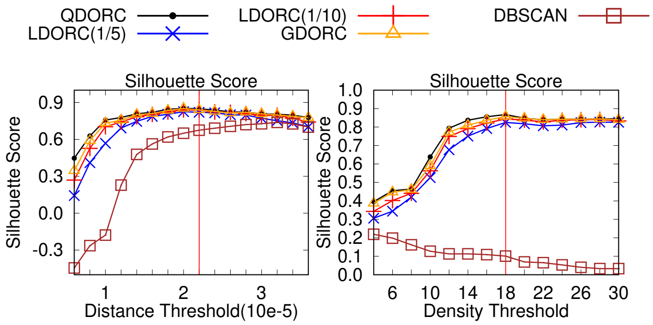


Fig. 9: Silhouette Score on GPS dataset

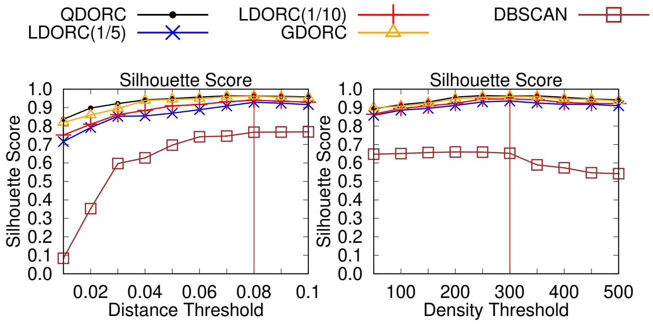


Fig. 10: Silhouette Score on FourSquare dataset

improves the purity and NMI scores while reducing the repair error. This is because the smaller distance between leaders and their followers tightens the approximation. However, the time cost of LDORC increases as τ decreases, since more leaders are selected, expanding the neighboring search space. W3

The root cause of GDORC's superior performance compared to QDORC and DBSCAN lies in its use of a grid structure, which transforms point-level neighboring search into cell-level search. By utilizing the grid, GDORC prunes points far from the target point, as explained in Equation 7 and the grid-based LP solution in Section V-B, avoiding unnecessary distance calculation between every pair of points. The grid also serves as a natural index for spatial proximity, accelerating the neighbor search process without sacrificing clustering quality, as detailed in Section V-C1 and Algorithm 1.

Compared to LDORC, GDORC outperforms in efficiency by eliminating the need for distance calculations between each point and the leaders, a step required by LDORC. While LDORC uses leaders for approximation, it still computes distances between data points and their assigned leaders, which can be computationally expensive. In contrast, GDORC utilizes the grid for pruning and indexing, significantly improving efficiency without compromising clustering quality. W3

D. Determining Clustering Parameters

GDORC shares the parameters η and ε with DBSCAN [8], which may affect clustering, especially with dirty data in Figure 8 (b). Misclassifications and errors can occur if these parameters are mis-tuned, leading to inefficiencies during tuning across different datasets. To mitigate this, we first use the cumulative density function of Poisson distribution [32] to estimate a coarse range of candidates η and ε. Then, we select the parameter combination among the candidates which achieve the highest Silhouette Score. The Silhouette Score [29] effectively evaluates clustering separation, density and compactness. A higher Silhouette Score means a better clustering quality.

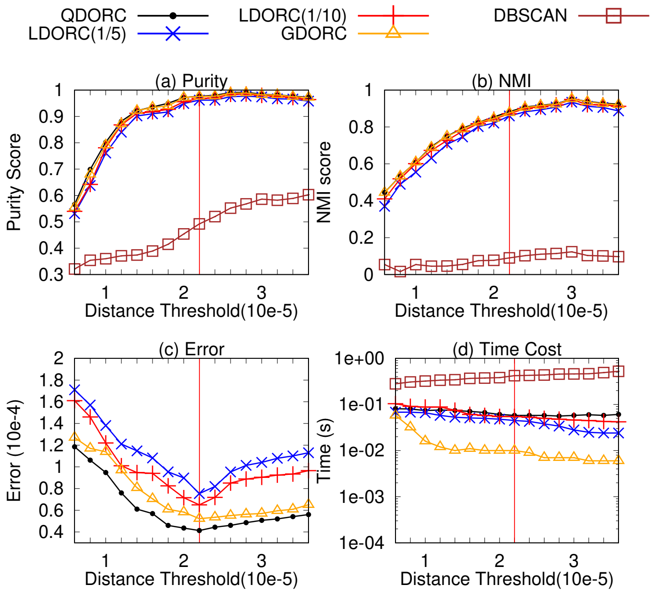


Fig. 11: Varying ε on GPS data with η=10

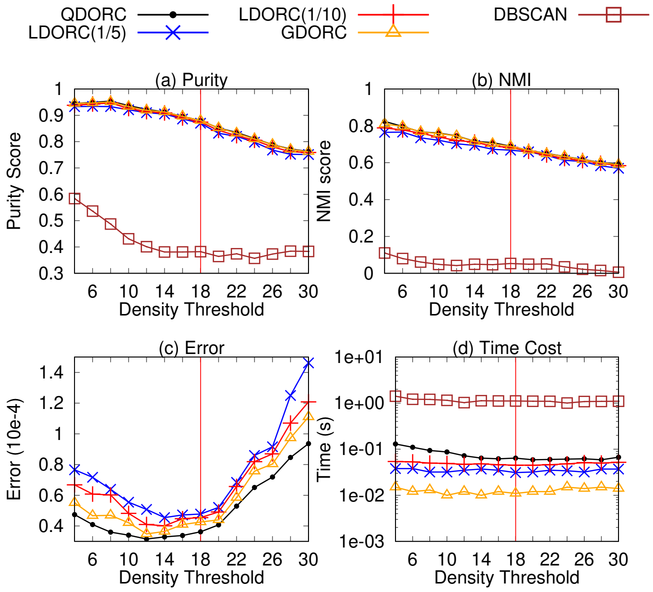


Fig. 12: Varying η on GPS data with $\varepsilon = 1.4e^{-5}$

To validate the parameter selection process, Figures 9 and 10 show the Silhouette Score on the GPS and Foursquare datasets, with the selected parameter combinations marked by the red line. For example, in Figure 9, the selected parameters for the GPS dataset is $\varepsilon = 2.2 \times 10^{-5}$ and $\eta = 18$, determined by the highest Silhouette Score. These selected parameters are

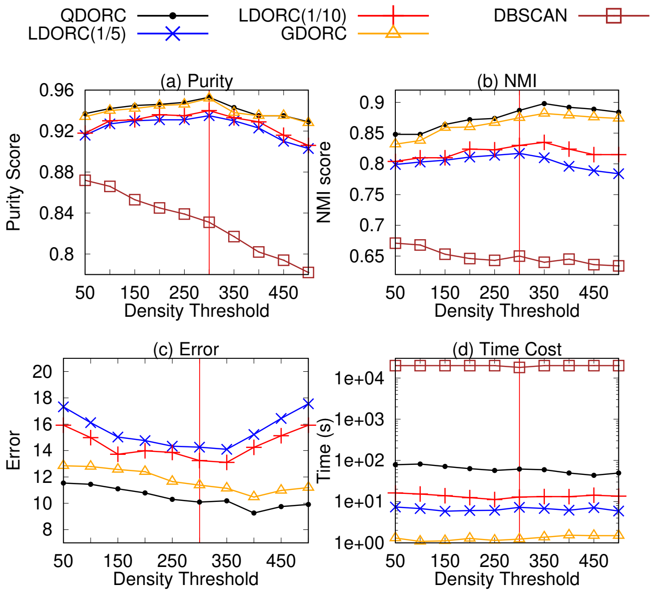


Fig. 13: Varying η on FourSquare data with $\varepsilon = 0.1$

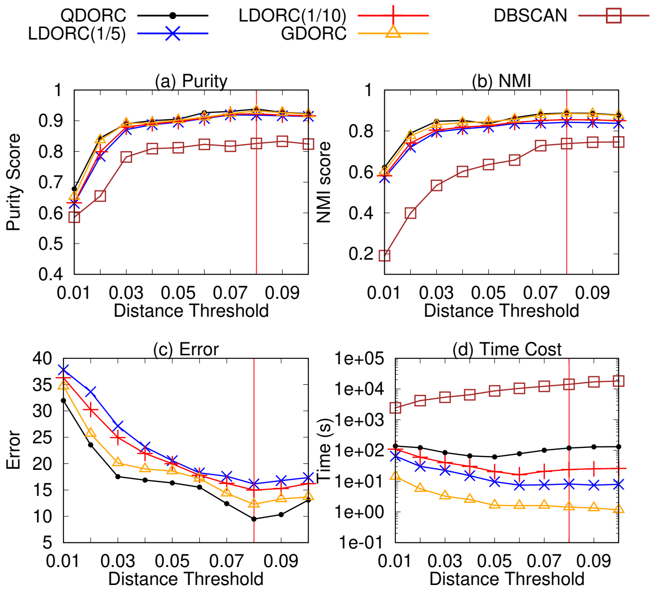


Fig. 14: Varying ε on FourSquare data with $\eta = 300$

further evaluated in Figures 11-14, in terms of clustering quality and repairing accuracy. For example, in Figures 11(a) and 11(b), when ε approaches the selected setting $\varepsilon = 2.2 \times 10^{-5}$, Purity and NMI can achieve a high score more than 0.95 and 0.8, respectively. Moreover, as shown in Figure 11(c), the repair error decreases to 0.6×10^{-4} when $\varepsilon = 2.2 \times 10^{-5}$. In contrast, for other parameters, e.g. $\varepsilon = 0.6 \times 10^{-5}$, the repair error may be as large as 1.2×10^{-4}. It demonstrates that GDORC can indeed achieve good performances with such selected parameters. These figures reveal that GDORC achieves superior efficiency in time cost while maintaining high accuracies.

W2

E. Varying Dirty Rate

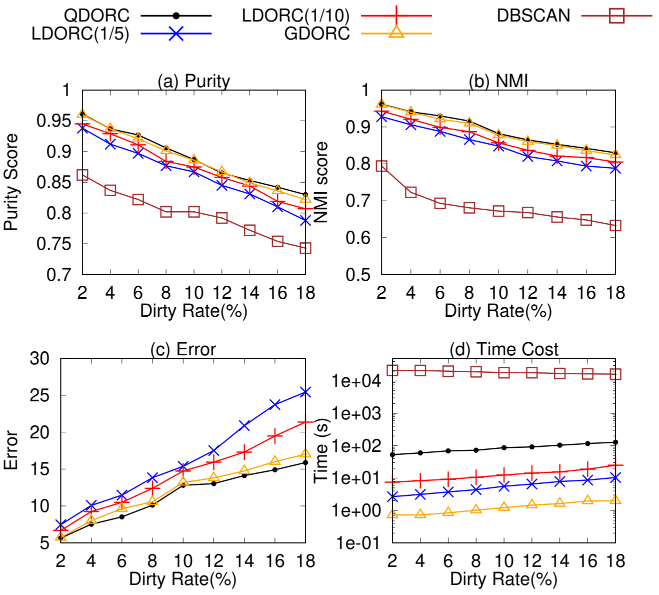


Fig. 15: Varying dirty rate on FourSquare (ε=0.1 and η=300)

Figure 15 demonstrates GDORC's performance on the Foursquare dataset with varying dirty rates. As the dirty rate increases, the accuracies of all methods decline at similar rate. GDORC and QDORC maintain high levels of clustering purity and NMI, higher than those of LDORC and DBSCAN. Additionally, GDORC and QDORC exhibit low repairing error compared to LDORC. Notably, the time costs of GDORC remain the lowest across all dirty rates.

F. Scalability

Figure 16 illustrates the scalability of GDORC on the Foursquare dataset. As the data size increases, both the clustering purity and NMI of GDORC and QDORC remain high, significantly outperforming LDORC and DBSCAN. Moreover, the repairing errors of GDORC and QDORC are substantially lower than those of LDORC and DBSCAN. Notably, GDORC's time costs are significantly lower than those of LDORC and QDORC across various data sizes, particularly QDORC, demonstrating GDORC's superior scalability.

G. Application Study

The GPS dataset is collected from various mobile phones in a real-world student attendance tracking context. In this scenario, students use their mobile phones to obtain their locations and then submit the locations via QR codes in classrooms. If a student's location is near the classroom building, the student is considered to be attending the class. Besides, in terms of sampling devices, we gather data points from various models of smartphones and tablets.

The reason why GPS data is selected is because its innate spatial characteristics and relatively low collecting quality. On the one hand, GPS data reflect the physical locations in the real world, which is ideal for distance-based clustering. For example, the locations of the students who attend the class form a cluster, since these students are all in the classroom.

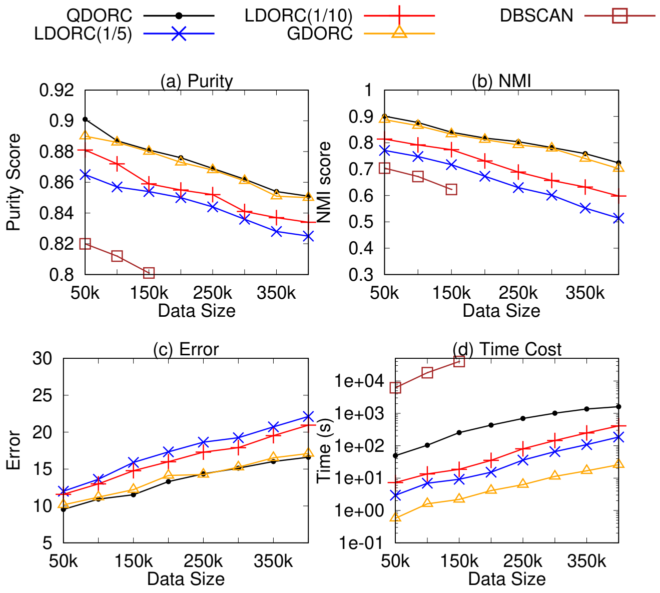


Fig. 16: Scalability on FourSquare dataset

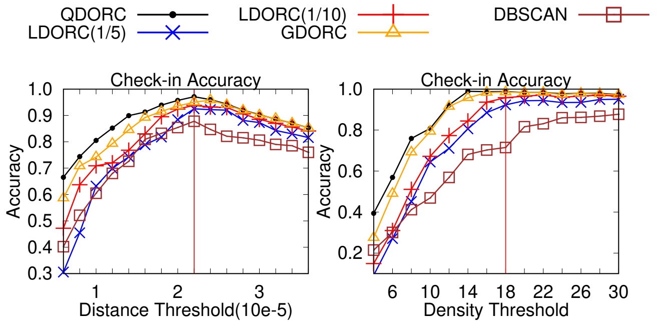


Fig. 17: Check-in application accuracy on GPS dataset

On the other hand, GPS data are often collected with low precision from mobile phones, and are often accompanied by noise and location shifting errors. For example, due to network fluctuations and the low locating precision of mobile phones, the collected locations may shift far from the classroom locations. Therefore, GPS data is useful for evaluating our proposed repairing methods.

Figure 8 provides a concrete example in the aforementioned scenario. Students are located in 3 different classrooms as in Figure 8(a). Due to network fluctuations and low locating precision, some locations shift, leading to poor clustering results in Figure 8(b). Fortunately, our proposal can repair these errors and obtain better clustering performance in Figure 8(c). We utilize an check-in accuracy score to evaluate the impact of erroneous location points and the effectiveness of the data repair process. The check-in accuracy measures the amount of students with correct attendance checks. As demonstrated in Figure 17, with a higher check-in accuracy score, GDORC can reduce the misidentification due to dirty data through repairing.

W2

VII. Conclusion

Density-based clustering can successfully identify noisy data but fails to clean them. Conversely, existing constraint-based repairing relies on external constraint knowledge and ignores the density information embedded within the data. Inspired by these limitations, this paper addresses a novel problem of clustering and repairing dirty data. We show that cleaning and clustering data simultaneously can achieve outstanding performance. With the happy marriage of clustering and repairing advantages, both the clustering and repairing accuracies are significantly improved as presented in the experimental evaluation. Our major technical contributions include: (1) the proof of NP-hardness of the DORC problem; (2) the formulation of DORC as an ILP problem; (3) a grid-based constant factor approximation solution with LP relaxation of the DORC problem.

Acknowledgement: This work is supported in part by National Natural Science Foundation of China (92267203, 62021002, 62072265, 62232005), National Key Research and Development Plan (2021YFB3300500), and Beijing Key Laboratory of Industrial Big Data System and Application. Shaoxu Song (https://sxsong.github.io/) is the corresponding author.

References

[1] A. Beer, A. Draganov, E. Hohma, P. Jahn, C. M. M. Frey, and I. Assent. Connecting the dots - density-connectivity distance unifies dbscan, k-center and spectral clustering. In *KDD*, pages 80–92. ACM, 2023.

[2] P. Bohannon, M. Flaster, W. Fan, and R. Rastogi. A cost-based model and effective heuristic for repairing constraints by value modification. In *SIGMOD Conference*, pages 143–154, 2005.

[3] L. Chen, D. Zhang, G. Pan, X. Ma, D. Yang, K. Kushlev, W. Zhang, and S. Li. Bike sharing station placement leveraging heterogeneous urban open data. In K. Mase, M. Langheinrich, D. Gatica-Perez, H. Gellersen, T. Choudhury, and K. Yatani, editors, *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp 2015, Osaka, Japan, September 7-11, 2015*, pages 571–575. ACM, 2015.

[4] J. Chomicki and J. Marcinkowski. Minimal-change integrity maintenance using tuple deletions. *Inf. Comput.*, 197(1-2):90–121, 2005.

[5] Q. Cui, W. Zheng, W. Hou, M. Sheng, P. Ren, W. Chang, and X. Li. Holocleanx: A multi-source heterogeneous data cleaning solution based on lakehouse. In *HIS*, volume 13705 of *Lecture Notes in Computer Science*, pages 165–176. Springer, 2022.

[6] M. M. G. Duarte and M. A. Sakr. Outlier detection and cleaning in trajectories: A benchmark of existing tools. In *Proceedings of the Workshops of the EDBT/ICDT 2023 Joint Conference, Ioannina, Greece, March, 28, 2023*, 2023.

[7] M. Ester, H.-P. Kriegel, J. Sander, M. Wimmer, and X. Xu. Incremental clustering for mining in a data warehousing environment. In *VLDB*, pages 323–333, 1998.

[8] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, pages 226–231, 1996.

[9] J. Gan and Y. Tao. DBSCAN revisited: Mis-claim, un-fixability, and approximation. In *SIGMOD Conference*, pages 519–530, 2015.

[10] J. Gan and Y. Tao. Dynamic density based clustering. In *SIGMOD Conference*, pages 1493–1507. ACM, 2017.

[11] J. Gan and Y. Tao. Dynamic density based clustering. In *SIGMOD Conference*, pages 1493–1507, 2017.

[12] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.

[13] A. Gunawan and M. de Berg. A faster algorithm for dbscan. *Master's thesis*, 2013.

[14] M. A. Hernández and S. J. Stolfo. Real-world data is dirty: Data cleansing and the merge/purge problem. *Data Min. Knowl. Discov.*, 2(1):9–37, 1998.

[15] V. J. Hodge and J. Austin. A survey of outlier detection methodologies. *Artif. Intell. Rev.*, 22(2):85–126, 2004.

[16] J. Jang and H. Jiang. DBSCAN++: towards fast and scalable density clustering. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 3019–3029. PMLR, 2019.

[17] S. R. Jeffery, M. N. Garofalakis, and M. J. Franklin. Adaptive cleaning for rfid data streams. In *VLDB*, pages 163–174, 2006.

[18] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[19] B. Kim, K. Koo, U. Enkhbat, and B. Moon. Denforest: Enabling fast deletion in incremental density-based clustering over sliding windows. In *SIGMOD Conference*, pages 296–309. ACM, 2022.

[20] B. Kim, K. Koo, J. Kim, and B. Moon. DISC: density-based incremental clustering by striding over streaming data. In *ICDE*, pages 828–839. IEEE, 2021.

[21] H.-P. Kriegel, P. Kröger, J. Sander, and A. Zimek. Density-based clustering. *Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery*, 1(3):231–240, 2011.

[22] J. Li, C. Jiang, K. Han, Q. Yu, and H. Zhang. High-resolution spatiotemporal inference of urban road traffic emissions using taxi GPS and multi-source urban features data: a case study in chengdu, china. *Urban Inform.*, 3(1):17, 2024.

[23] Z. Li, H. Wang, W. Shao, J. Li, and H. Gao. Repairing data through regular expressions. *Proc. VLDB Endow.*, 9(5):432–443, 2016.

[24] G. Liu, Q. Zheng, S. Niu, and J. Ma. Research and application of the global positioning system (GPS) clustering algorithm based on multilevel functions. *J. Comput. Methods Sci. Eng.*, 24(1):357–368, 2024.

[25] A. Lulli, M. Dell'Amico, P. Michiardi, and L. Ricci. NG-DBSCAN: scalable density-based clustering for arbitrary data. *Proc. VLDB Endow.*, 10(3):157–168, 2016.

[26] S. T. Mai, I. Assent, and M. Storgaard. Anydbc: An efficient anytime density-based clustering algorithm for very large complex datasets. In *KDD*, pages 1025–1034. ACM, 2016.

[27] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*. Cambridge University Press, 2008.

[28] X. V. Nguyen, J. Epps, and J. Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research*, 11:2837–2854, 2010.

[29] K. R. Shahapure and C. Nicholas. Cluster quality analysis using silhouette score. In *7th IEEE International Conference on Data Science and Advanced Analytics, DSAA 2020, Sydney, Australia, October 6-9, 2020*, pages 747–748. IEEE, 2020.

[30] K. Shoji, H. Terashima, N. Kawaguchi, S. Katayama, K. Urano, T. Yonezawa, and N. Tamura. Unveiling human attributes through life pattern clustering using GPS data only. In *Proceedings of the 32nd ACM International Conference on Advances in Geographic Information Systems, SIGSPATIAL 2024*, pages 621–624. ACM, 2024.

[31] H. Song and J. Lee. RP-DBSCAN: A superfast parallel DBSCAN algorithm based on random partitioning. In *SIGMOD Conference*, pages 1173–1187, 2018.

[32] S. Song, F. Gao, R. Huang, and Y. Wang. On saving outliers for better clustering over noisy data. In *SIGMOD Conference*, pages 1692–1704. ACM, 2021.

[33] S. Song, C. Li, and X. Zhang. Turn waste into wealth: On simultaneous clustering and cleaning over dirty data. In *SIGMOD Conference*, pages 1115–1124, 2015.

[34] C. Wang, J. Qiao, X. Huang, S. Song, H. Hou, T. Jiang, L. Rui, J. Wang, and J. Sun. Apache iotdb: A time series database for iot applications. *Proc. ACM Manag. Data*, 1(2):195:1–195:27, 2023.

[35] H. Wang and Y. Si. Detection of traffic abnormity based on clustering analysis of taxi GPS data. In *Proceedings of the 2nd International Conference on Data Science and Information Technology, DSIT 2019, Seoul, South Korea, July 19-21, 2019*, pages 219–224. ACM, 2019.

[36] Z. Wang, R. Zhang, J. Qi, and B. Yuan. DBSVEC: density-based clustering using support vector expansion. In *ICDE*, pages 280–291. IEEE, 2019.

[37] J. Wijsen. Database repairing using updates. *ACM Trans. Database Syst.*, 30(3):722–768, 2005.

[38] K. Yang, Y. Gao, R. Ma, L. Chen, S. Wu, and G. Chen. DBSCAN-MS: distributed density-based clustering in metric spaces. In *ICDE*, pages 1346–1357. IEEE, 2019.

[39] A. Zhang, S. Song, J. Wang, and P. S. Yu. Time series data cleaning: From anomaly detection to anomaly repairing. *Proc. VLDB Endow.*, 10(10):1046–1057, 2017.

Kenny Ye Liang is a master student in the School of Software, Tsinghua University. His research interests include data quality and data mining.

Yunxiang Su is a PhD student in the School of Software, Tsinghua University. His research interests include data quality.

Shaoxu Song (https://sxsong.github.io/) is an associate professor at Tsinghua University, Beijing, China. His research interests include data quality and data cleaning. He has published more than 80 papers in top conferences and journals such as SIGMOD, VLDB, KDD, ICDE, TODS, TKDE, VLDBJ, etc.

Chunping Li is an associate professor in the School of Software, Tsinghua University, Beijing, China. His research interests include data mining and artificial intelligence.

Summary of Difference

In this journal paper, which expands on the research presented at the 21st ACM SIGKDD Conference on Knowledge Discovery and Data Mining [33], we provide detailed descriptions and significant extensions in theoretical, technical, and experimental aspects.

The preliminary conference version [33] focuses on simultaneous repairing and clustering data, proposing two algorithms: QDORC and LDORC. QDORC directly obtains a solution to the corresponding LP relaxation without calling LP solvers, while LDORC offers a balance between effectiveness and efficiency with a notably lower time cost.

In this extended version, we enhance the simultaneous repairing and clustering algorithm by organizing data points into grids, a method we refer as GDORC. By dividing data points in the data space into cells with constant side length, we achieve time savings by searching from the cell level instead of searching from individual point level. Despite the similarity in repair costs to QDORC, GDORC demonstrates comparable repairing and clustering results with significantly reduced time costs—even lower than those of LDORC.

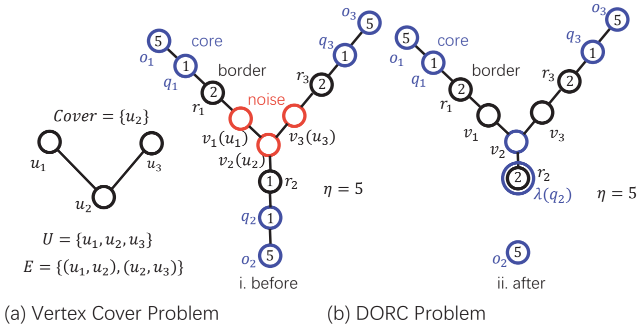
Major changes from the conference version include:

- In Section IV, in addition to describing the ILP formulation, we provide a hardness analysis of the DORC problem.
  - We prove that the DORC problem is NP-hard with Theorem 1.
  - With Proposition 2, we present a tractable special case to the DORC problem and prove that there exists a PTIME algorithm for the special case.
- In Section V, we present the new method GDORC for repairing and clustering simultaneously based on grid.
  - In Section V-A, we introduce the preliminary knowledge for the grid-based method. Specifically, we provide basic definitions for grid, cell, and noise cell, and define the minimum distance between two cells and neighboring cells. To better aid comprehension, we provide practical examples.
  - In Section V-B, we rewrite Formula 7 regarding the solution for the LP problem using the grid-based method. We focus on using grid-based strategy to improve computational efficiency. Proposition 5 is introduced, demonstrating how unnecessary repairs can be avoided in the grid-based method.
  - In Section V-C, we propose Algorithm 1 (INITIALIZATION) and Algorithm 2 (GDORC) for the grid-based method. Both Algorithms are provided with detailed explanations. Example 5 elaborates how to use grid-based method in repairing and clustering. Proposition 6 confirms that GDORC can successfully terminate, by repairing without introducing new noise points. Proposition 7 provides a time cost analysis for the entire GDORC method.
  - In Section V-D, we detail the approximation performance of our proposed GDORC method. Specifically, Proposition 8 demonstrates that GDORC is a factor-$\alpha\eta$ approximation to the ILP problem. Proposition 9 provides a special case of connected components of the GDORC method.
- In Section VI, we re-conduct the experiments on two real datasets: GPS and Foursquare. The GPS dataset with real-world dirty data is used primarily to test performance across different parameters, while the Foursquare dataset is utilized mainly to assess scalability and vary dirty rates. Since the baseline methods, such as FD and OPTICS, were already compared in the preliminary conference version [33], we are focusing on QDORC, which provides a quadratic-time approximate solution to the DORC problem, and LDORC, which offers an linear-time approximate solution to the DORC problem, along with our proposed GDORC method. Based on the experimental results, our proposed GDORC method achieves accuracy comparable to QDORC and significantly better than that of LDORC and DBSCAN. Additionally, GDORC consistently records the lowest time costs across all experiments.
  - In Figures 12- 14, we present the experiments with various distance and density thresholds.
  - In Figure 15, we assess the performance of our GDORC in comparison to others by varying the dirty rate.
  - In Figure 16, we examine the scalability of our GDORC method. Note that with data size over 150k, DBSCAN is unable to complete within 6 hours, and thus the experiments with DBSCAN on data sizes over 150k are omitted.
- In Section 6, we examine more recent related work on density-based clustering such as DBSVEC [36], DISC [20], and DBSCAN-DIST [1], and review more recent related work on data repairing such as HOLOCLEAN [5], RSR [23], and MISC [32]. We detail the reasons why these methods are carefully considered but not directly comparable in the experiments, providing comprehensive understandings of their capabilities and constraints.

Appendix

A. Proof of Theorem 1

Theorem 1. *The* DORC *problem is* NP*-hard.*



(a) Vertex Cover Problem (b) DORC Problem

Fig. 3: DORC Transformation to Vertex Cover

Proof. We prove the conclusion by constructing a reduction from the VERTEX COVER problem, which is one of Karp's 21 NP-complete problems [18]. Given an arbitrary graph $G(U, E)$ with $|U|$ vertices and $|E|$ edges, a vertex cover is a subset $C \subseteq U$ such that for each edge $(u_i, u_j) \in E$, C contains at least one of u_i or u_j.

We next transform the aforementioned VERTEX COVER problem on $G(U, E)$ into a specific DORC problem on a set of points $\mathcal{P}$, with the following mapping from $G(U, E)$ to $\mathcal{P}$. For each edge $(u_i, u_j) \in G(U, E)$, we map it to two points $v_i, v_j \in \mathcal{P}$ with $\delta(v_i, v_j) = \varepsilon$. Moreover, for each point $v_i \in \mathcal{P}$ mapped from $u_i \in G(U, E)$, we further introduce $\eta - l - 2$ data points overlaid at a location $r_i \in \mathcal{P}$, one data point at a location $q_i \in \mathcal{P}$, and η data points overlaid at a location $o_i \in \mathcal{P}$, where l is the number of edges connected to u_i in $G(U, E)$ and $l < \eta < n$. Simultaneously, for each $v_i \in \mathcal{P}$, we define the distances between these newly introduced points in $\mathcal{P}$ as follows: $\delta(v_i, r_i) = \varepsilon$, $\delta(r_i, q_i) = \varepsilon$, and $\delta(q_i, o_i) = \varepsilon$. Note that all other pairs of points have distances $\delta(*, *) \gg \varepsilon$. This transformation can be done in polynomial time.

Note that all newly introduced points are non-noise points and all v_i are noise points. This is because points in locations o_i and q_i are all core points, with $\eta+1$ and $2\eta-l-1$ neighbors, respectively. Points in location r_i within the ε-neighborhood of core point q_i are border points. However, each v_i, having $1 + l + (\eta - l - 2) = \eta - 1$ neighbors, is identified as a noise point. With such transformation, we can prove that $G(U, E)$ has a vertex cover if and only if there is a feasible repair on $\mathcal{P}$, by the following:

If graph G has a vertex cover C of size $|C| = k$, then a repair λ modifying k points with cost $\Delta(\lambda) = k\varepsilon$ is feasible. For each $u_i \in C$ (corresponding to $v_i \in \mathcal{P}$), setting $\lambda(q_i) = r_i$ ensures that v_i acquires η neighbors and becomes a core point. By the vertex cover definition, for each $u_j \notin C$, there must be a $u_i \in C$ having an edge with u_j. The corresponding v_j falls in the ε-neighborhood of core point v_i and becomes a border point, eliminating all noise points.

Conversely, suppose that there exists a λ such that $\Delta(\lambda) = k\varepsilon$ and $k < |C^*|$, where C^* is a minimum vertex cover. As aforementioned, only all v_i are noise points. First, the repairing must happen between (v_i, v_j), (v_i, r_i), (r_i, q_i) or (q_i, o_i), given distances of other pairs $\gg \varepsilon$. Moreover, repairing between $(v_i, v_j), (v_i, v_j)$ cannot eliminate noise points v_i, since the number of neighbors of v_i will not increase. Repairing between (q_i, o_i) or from r_i to q_i cannot upgrade the corresponding r_i to core point, and thus cannot make v_i non-noise. Thus, the only feasible repairing is $\lambda(q_i) = r_i$. This repair makes v_i a core point and makes all v_j with $\delta(v_i, v_j) \leq \varepsilon$, i.e., $(u_i, u_j) \in E$, become border points. Since we need to repair at least $|C^*|$ points to cover all the edges with cost $|C^*|\varepsilon$, it contradicts any claim that fewer repairs can achieve the same result. □

B. Proof of Proposition 2

Proposition 2. *For $\eta = 2$, there is a* PTIME *algorithm for solving the* DORC *problem.*

Proof. In the case of $\eta = 2$, a point can only either be a core (with at least two ε-neighbors) or a noise (with itself as the only ε-neighbor). By repairing a noise point p_i to any point $\lambda(p_i) = p_j$, both $\lambda(p_i)$ and p_j upgrade to core points.

In the PTIME algorithm for the minimum EDGE COVER problem [12], we interpret the relationships of noises with other points. The algorithm greedily picks a noise p_i with $\min_{1 \leq i \leq n, 1 \leq j \leq n, i \neq j} w(p_i, p_j)$ to repair in each step, and forms an optimal solution when no noise points remain. □

C. Proof of Proposition 3

Proposition 3. *The optimal solution $\mathbf{x}^{\text{ILP}}, \mathbf{y}^{\text{ILP}}$ of* ILP *forms an optimal repair λ^{ILP} with the minimum repairing cost*

$$\Delta(\lambda^{\text{ILP}}) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} x_{ij}^{\text{ILP}},$$

where $\lambda^{\text{ILP}}(p_i) = p_j$ iff $x_{ij}^{\text{ILP}} = 1, 1 \leq i \leq n, 1 \leq j \leq n$.

Proof. The correctness is obvious given that a point can only be repaired to one location (Formula 2) with cost weight $w_{ij} = w(p_i, p_j)$, and all the repaired points are either cores or in ε-neighborhood of some cores (Formulas 4 and 5). □

D. Proof of Proposition 4

Proposition 4. *For $\eta < n$, a feasible solution to the* ILP *problem always exists.*

Proof. By simply repairing all the points to a single location say p_1, we have $x_{i1} = 1$ and $y_1 = 1$, i.e., all the points become core points locating in p_1 after repairing. □

E. Proof of Proposition 5

Proposition 5. *When selecting noise point p_i to repair p_j,i.e.,$|\mathcal{N} \setminus \mathcal{N}(p_j)| \geq (1 - y_j)\eta$, noise point p_i does not need to be repaired from location p_i to location p_j if the distance between them is less than ε, i.e., $\delta(p_i, p_j) \leq \varepsilon$.*

Proof. Supposing that noise point p_i is repaired into location p_j, $C(p_j)$ would not change, since it already has $p_i \in C(p_j)$. And the count of ε-neighbors of p_j, i.e., $|C(p_j)|$ would not

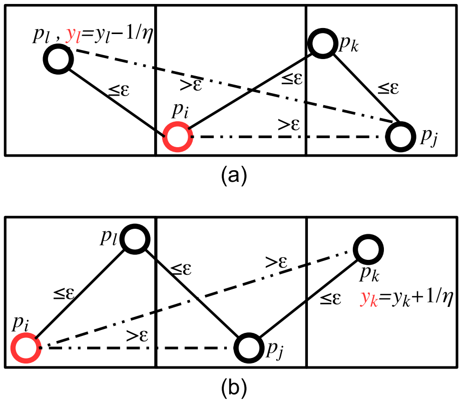


Fig.7: Cases for updating y_l and y_k for points p_l and p_k

increase. Repairing p_i into p_j will only increase the repair cost with $\delta(p_i, p_j)$. Referring to the minimum change principle in data repairing, this repair is unnecessary. Thus, there is no need to move noise point p_i into point p_j when the distance between them is less than ε. □

Proposition 6?

F. Proof of Proposition 6

Proposition 6. *Consider repairing point p_i into another point p_j, i.e., $\lambda(p_i) = p_j$. Let $\mathcal{N}_0$ denote the noise set before repair, and let $\mathcal{N}$ denote the noise set after repair. We have $\mathcal{N} \subset \mathcal{N}_0$.*

Proof. Referring to Algorithm 2, ε-neighbors of p_i and p_j, i.e., points $p_l \in C(p_i)$ and points $p_k \in C(p_j)$, will be influenced when repairing p_i into p_j.

For $p_l \in C(p_i)$, consider the distance between p_l and p_j. If $\delta(p_l, p_j) > \varepsilon$, as shown in Figure 7 (a), y_l will decrease to $y_l = y_l - \frac{1}{\eta}$ after repair, because p_i is removed from p_l's ε-neighborhood. Otherwise, if $\delta(p_l, p_j) \le \varepsilon$, as shown in Figure 7 (b), y_l will not change after repair, because p_i is still in p_l's ε-neighborhood. p_l can either be a border point or a noise point before repairing, since there is no need to repair p_i if p_l is a core point. With y_l stays the same or decrease, p_l's point type remains unchanged.

For $p_k \in C(p_j)$, consider the distance between p_k and p_i. If $\delta(p_k, p_i) > \varepsilon$, as shown in Figure 7 (b), y_k will increase to $y_k = y_k + \frac{1}{\eta}$ after repair, because p_i becomes p_k's new ε-neighbor. If $\delta(p_k, p_i) \le \varepsilon$, as shown in Figure 7 (a), y_k will not change after repair, because p_i is still in p_k's ε-neighborhood. There are three possible point types for p_k before repairing: core, border, or noise. If p_k is a core, it stays as a core. If p_k is a border, it either stays as a border or becomes a new core. If p_k is a noise, it either stays as as noise or becomes a border. This is because the number of ε-neighbors of p_k can only increase after repair.

In summary, after each repair, p_i either becomes a core or border point by moving into other locations, p_j becomes a core or stays as a border with y_j increased, $p_l \in C(p_i)$ stays as before, and $p_k \in C(p_j)$ has either the same y_k or an increased y_k. Thus, no new noise will be introduced. □

G. Proof of Proposition 7

Proposition 7. *The time complexity of the grid-based* GDORC *method is* $\mathrm{O}(nN_m + N\eta n)$, *with* $\mathrm{O}(nN_m)$ *being the time complexity of* Algorithm 1 (INITIALIZATION) *and* $\mathrm{O}(N\eta n)$ *being the time complexity of* Algorithm 2 (GDORC REPAIR).

Proof. Algorithm 1 first traverses each point on the cell level, with Line 1 going over each cell, and Line 3 going over each point in each cell, costing $\mathrm{O}(n)$ for n points. Then, it calculates each point's $\varepsilon-$neighbor cell in Line 8. Given cell width $\frac{\varepsilon}{\sqrt{d}}$, there are $\mathrm{O}((2\lceil\sqrt{d}\rceil + 1)^d)$ cells to check for each cell's $\varepsilon-$neighbor. After that, Line 9 checks for points in these neighbors, with each cell containing at most N_m points. In sum, we need to check $\mathrm{O}((2\lceil\sqrt{d}\rceil + 1)^d N_m)$ points to determine the status of each point. Since the dimensionality d is low, $\mathrm{O}((2\lceil\sqrt{d}\rceil + 1)^d)$ can be regarded as constant. Therefore, initialization on cell level takes $\mathrm{O}(nN_m)$ time. Similarly, initialization on point level also takes $\mathrm{O}(nN_m)$ time. To initialize supporting sets, every point is traversed again, with $\mathrm{O}(n)$ time. In summary, the time complexity of Algorithm 1 is $\mathrm{O}(nN_m)$.

For repairing in Algorithm 2 (GDORC), Line 2 for finding point $p_j \in \mathcal{P}$ with the maximum $y_j < 1$ can be solved in $\mathcal{O}(1)$ time, by amortizing points $\mathcal{P}$ into a constant space w.r.t. $\mathbf{y}$ values. After selecting a p_j to repair, there are at most η points to repair. Lines 4-12 will repeat $\mathrm{O}(\eta)$ times. When there are sufficient noises, it takes $\mathrm{O}(N_s)$ to find the nearest noise point with the minimum cell distance $\delta(u_j, u_i)$ in Line 5 every time. It costs $\mathrm{O}(1)$ to repair the selected p_i in Line 7 and update set status in Lines 8-11. In sum, when there are sufficient noise point for repairing, it takes $\mathrm{O}(\eta N_s)$ to repair a p_j. With $N_s < N$, the time cost can be reduced to $\mathrm{O}(\eta N)$.

When there are insufficient noises left to repair p_j into a core, Lines 17-20 allocate the remaining noises. For each remaining noise, it takes $\mathrm{O}(N_c)$ to find the nearest core with the minimum cell distance $\delta(u_j, u_i)$ in Line 19. With at most η noises remained, it takes $\mathrm{O}(\eta N_c)$ to repair all the remaining noises. With $N_c < N$, the time cost can be reduced to $\mathrm{O}(\eta N)$.

In the worst case, every border point needs only one noise point to repair, so Line 1 runs $|\mathcal{N}|$ times. Since $|\mathcal{N}| << n$, Algorithm 2 costs $\mathrm{O}(N\eta n)$. □

H. Proof of Proposition 8

Proposition 8. Algorithm 2 (GDORC REPAIR) *returns a feasible solution to the* ILP *problem, and is a factor-$\alpha\eta$ approximation with $\alpha = \frac{\delta_{\max}}{\delta_{\min}}$.*

Proof. The correctness of the grid-based method is verified by Proposition 6 since Algorithm 2 (GDORC) can repair all noises without introducing new noises.

The repairing cost every time is no greater than $\delta_{\max}$ since Algorithm 2 repairs only the points in $\mathcal{N}$, i.e.,

$$\Delta(\lambda^{\text{GDORC}}) = \sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}x_{ij} \le \delta_{\max}|\mathcal{N}|. \tag{8}$$

Furthermore, a noise point $p_i \in \mathcal{N}$ can be addressed with repairing p_i itself and its ε-neighbor p_j into a core point via repairing (with cost at least $\delta_{\min}$). When repairing p_i or p_j

into a core point, it eliminates at most η noises. Considering all the $|\mathcal{N}|$ noise points, there are at least $\frac{|\mathcal{N}|}{\eta}$ points repaired, with cost no less than $\frac{|\mathcal{N}|}{\eta}\delta_{\min}$, i.e.,

$$\Delta(\lambda^{\text{ILP}}) \geq \frac{|\mathcal{N}|}{\eta}\delta_{\min}. \tag{9}$$

With formulas (8) and (9), we can derive that

$$\frac{\Delta(\lambda^{\text{GDORC}})}{\Delta(\lambda^{\text{ILP}})} \leq \frac{\delta_{\max}}{\delta_{\min}}\eta,$$

it concludes the factor-$\alpha\eta$ approximation. □

I. Proof of Proposition 9

Proposition 9. *For $\eta = 2$, Algorithm 2 (*GDORC*) is a factor-α approximation with $\alpha = \frac{\delta_{\max}}{\delta_{\min}}$.*

Proof. With the proof of Proposition 2, a point is either a core point (with $y_j^{\text{LP}} = 1$) or a noise point (with $y_j^{\text{LP}} = \frac{1}{2}$) in this special case. That means Line 2 in Algorithm 2 always selects a noise point in $\mathcal{N}$. It was repaired with another $p_i \in \mathcal{N}$ in Line 5 to Line 7 or moved to another $p_i \in \mathcal{P}_c$ in Line 18 to Line 20, where $\mathcal{P}_c < \mathcal{P} \setminus \mathcal{N}$ We have $\Delta(\lambda^{\text{GDORC}}) \leq (\delta_{\max})\frac{|\mathcal{N}|}{2}$. Combining with Formula 9, where $\eta = 2$, it follows $\frac{\Delta(\lambda^{\text{GDORC}})}{\Delta(\lambda^{\text{ILP}})} \leq \alpha$. □