Achieving Anonymity via Clustering

GAGAN AGGARWAL
Google Inc., Mountain View
TOMÁS FEDER
Computer Science Department
Stanford University, Stanford
KRISHNARAM KENTHAPADI
Microsoft Research, Mountain View
SAMIR KHULLER
Computer Science Department
University of Maryland, College Park
RINA PANIGRAHY
Microsoft Research, Mountain View
DILYS THOMAS
Oracle, Redwood Shores
and
AN ZHU
Google Inc.
Mountain View

A preliminary version of this paper was presented at the 2006 ACM Principles of Database System (PODS) Conference.
This work was done while G. Aggarwal, K. Kenthapadi, R. Panigrahy, D. Thomas and A. Zhu were Ph.D. students at Stanford University. All of them and T. Feder were supported in part by NSF Grant ITR-0331640 and also supported in part by TRUST (The Team for Research in Ubiquitous Secure Technology), which receives support from the National Science Foundation (NSF award number CCF-0424422) and the following organizations: Cisco, ESCHER, HP, IBM, Intel, Microsoft, ORNL, Qualcomm, Pirelli, Sun and Symantec.
S. Khuller was supported by NSF Award CCF-0430650.
Current contact Information for authors: Email:gagan@google.com, tomas@cs.stanford.edu, krishnaram.kenthapadi@microsoft.com, samir@cs.umd.edu rina@microsoft.com, dilys@cs.stanford.edu, anzhu@cs.stanford.edu.

Publishing data for analysis from a table containing personal records, while maintaining individual privacy, is a problem of increasing importance today. The traditional approach of de-identifying records is to remove identifying fields such as social security number, name etc. However, recent research has shown that a large fraction of the US population can be identified using non-key attributes (called quasi-identifiers) such as date of birth, gender, and zip code. The k-anonymity model protects privacy via requiring that non-key attributes that leak information are suppressed or generalized so that, for every record in the modified table, there are at least $k-1$ other records having exactly the same values for quasi-identifiers. We propose a new method for anonymizing data records, where quasi-identifiers of data records are first clustered and then cluster centers are published. To ensure privacy of the data records, we impose the constraint that each cluster must contain no fewer than a pre-specified number of data records. This technique is more general since we have a much larger choice for cluster centers than k-Anonymity. In many cases, it lets us release a lot more information without compromising privacy. We also provide constant factor approximation algorithms to come up with such a clustering. This is the first set of algorithms for the anonymization problem where the performance is independent of the anonymity parameter k. We further observe that a few outlier points can significantly increase the cost of anonymization. Hence, we extend our algorithms to allow an ϵ fraction of points to remain unclustered, i.e., deleted from the anonymized publication. Thus, by not releasing a small fraction of the database records, we can ensure that the data published for analysis has less distortion and hence is more useful. Our approximation algorithms for new clustering objectives are of independent interest and could be applicable in other clustering scenarios as well.

Categories and Subject Descriptors: H.2.8 [**Database Applications**]: Data mining; H.3.3 [**Information Search and Retrieval**]: Clustering

General Terms: Approximation Algorithms

Additional Key Words and Phrases: Privacy, Anonymity, Clustering, Approximation Algorithms

1. INTRODUCTION

With the rapid growth in database, networking, and computing technologies, a large amount of personal data can be integrated and analyzed digitally, leading to an increased use of data-mining tools to infer trends and patterns. This has raised universal concerns about protecting the privacy of individuals [Time 1997].

Age	Place	Disease
α	β	Flu
$\alpha + 2$	β	Flu
δ	$\gamma + 3$	Hypertension
δ	γ	Flu
δ	γ -3	Cold

(a) Original table

Age	Place	Disease
*	β	Flu
*	β	Flu
δ	*	Hypertension
δ	*	Flu
δ	*	Cold

(b) 2-anonymized version

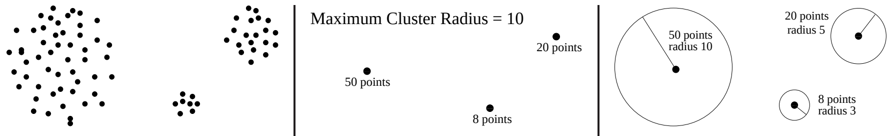
Age	Place	Points	Disease
α +1	β	2	Flu Flu
δ	γ	3	Hypertension Flu Cold

(c) 2-gather clustering, with maximum radius 3

Age	Place	Points	Radius	Disease
α +1	β	2	1	Flu Flu
δ	γ	3	3	Hypertension Flu Cold

(d) 2-cellular clustering, with total cost 11

Fig. 1. Original table and three different ways of achieving anonymity



(a) Original points (b) r-gather clustering (c) r-cellular clustering

Fig. 2. Publishing anonymized data

Combining data tables from multiple data sources allows us to draw inferences which are not possible from a single source. For example, combining patient data from multiple hospitals is useful to predict the outbreak of an epidemic. The traditional approach of releasing the data tables without breaching the privacy of individuals in the table is to de-identify records by removing the identifying fields such as name, address, and social security number. However, joining this de-identified table with a publicly available database (like the voters database) on columns like race, age, and zip code can be used to identify individuals. Recent research [Sweeney 2000] has shown that for 87% of the population in the United States, the combination of non-key fields like date of birth, gender, and zip code corresponds to a unique person. Such non-key fields are called *quasi-identifiers*. In what follows we assume that the identifying fields have been removed and that the table has two types of attributes: (1) the quasi-identifying attributes explained above and (2) the sensitive attributes (such as disease) that need to be protected.

In order to protect privacy, Samarati [Samarati 2001] proposed the k-Anonymity model, where some of the quasi-identifier fields are suppressed or generalized so that, for each record in the modified table, there are at least $k-1$ other records in the modified table that are identical to it along the quasi-identifying attributes. For the table in Figure 1(a), Figure 1(b) shows a 2-anonymized table corresponding to it. The columns corresponding to sensitive attributes, like disease in this example, are retained without change. The aim is to provide a k-anonymized version of the table with the minimum amount of suppression or generalization of the table entries. There has been a lot of recent work on k-anonymizing a given database table [Bayardo and Agrawal 2005; LeFevre et al. 2005]. An $O(k \log k)$ approximation algorithm was first proposed for the problem of k-Anonymity with suppressions only [Meyerson and Williams 2004]. This was recently improved to an $O(k)$ approximation for the general version of the problem [Aggarwal et al. 2005].

In this paper, instead of generalization and suppression, we propose a new technique for anonymizing tables before their release. We first use the quasi-identifying attributes to define a metric space (i.e., pairwise distances satisfying the triangle inequality) over the database records, which are then viewed as points in this space. This is similar to the approach taken in [Chawla et al. 2005], except that we do not restrict ourselves to points in $\mathcal{R}^d$; instead, we allow our points to be in an arbitrary metric space. We then cluster the points and publish only the final cluster centers along with some cluster size and radius information. Our privacy requirement is similar to the k-Anonymity framework – we require each cluster to have at least

r points[1]. Publishing the cluster centers instead of the individual records, where each cluster represents at least r records, gives privacy to individual records, but at the same time allows data-mining tools to infer macro trends from the database.

In the rest of the paper we will assume that a metric space has been defined over the records, using the quasi-identifying attributes. For this, the quasi-identifying attributes may need to be remapped. For example, zip codes could first be converted to longitude and latitude coordinates to give a meaningful distance between locations. A categorical attribute, i.e., an attribute that takes n discrete values, can be represented by n equidistant points in a metric space. Furthermore, since the values of different quasi-identifying attributes may differ by orders of magnitude, we need to weigh the attributes appropriately while defining the distance metric. For example, the attribute location may have values that differ in orders of 10 miles with a maximum of 1000 miles, while the attribute age may differ by a single year with a maximum of 100 years. In this case we assume that the attribute location is divided by 10 and the attribute age retained without change if both attributes are needed to have the same relative importance in the distance metric. For the example we provide in Figure 1, we assume that the quasi-identifying attributes have already been scaled. We assume that appropriate steps have been taken to map the data so that a metric is defined by choosing the scale factors carefully.

To publish the clustered database, we publish three types of features for each cluster: (1) the quasi-identifying attribute values for the cluster center (age and location in our example), (2) the number of points within the cluster, and (3) a set of values taken by the sensitive attributes (disease in our example). We'll also publish a measure of the quality of the clusters. This will give a bound on the error introduced by the clustering.

In this paper we consider two cluster-quality measures. The first one is the maximum cluster radius. For this we define the r-Gather problem, which aims to minimize the maximum radius among the clusters, while ensuring that each cluster has at least r members. As an example, r-Gather clustering with minimum cluster size $r = 2$, applied to the table in Figure 1(a) gives the table in Figure 1(c). In this example, the maximum radius over all clusters is 3. As another example, Figure 2(b) gives the output of the r-Gather algorithm applied to the quasi-identifiers, shown as points in a metric space in Figure 2(a). Our formulation of the r-Gather problem is related to, but not to be confused with, the classic k-Center problem [Hochbaum and Shmoys 1985]. The k-Center problem has the same objective of minimizing the maximum radius among the clusters, however, the constraint is that we can have no more than k clusters in total. The r-Gather problem is different from k-Center problem in that instead of specifying an upper bound on the number of clusters, we specify a lower bound on the number of points per cluster as part of the input. It's also worth noting that the constraint of at least r points per cluster implies that we can have no more than n/r number of clusters, where n is the total number of points in our data set.

We also consider a second (more verbose) candidate for indicating cluster-quality, whereby we publish the radius of each cluster, rather than just the maximum radius among all clusters. For each point within a cluster, the radius of the cluster gives

[1] We use r instead of k, as k is traditionally used in clustering to denote the number of clusters.

an upper bound on the distortion error introduced. Minimizing this distortion error over all points leads to the *cellular* clustering measurement that we introduce in this paper. More formally, the *cellular* clustering measurement over a set of clusters, is the sum, over all clusters, of the products of the number of points in the cluster and the radius of the cluster. Using this as a measurement for anonymizing tables, we define the r-CELLULAR CLUSTERING problem as follows: Given points in a metric space, the goal is to partition the points into cells, a.k.a. clusters, each of size at least r, and the cellular clustering measurement is minimized. Consider again the data in Figure 1(a). Figure 1(d) shows a r-cellular cluster solution with minimum cluster size $r = 2$. The total cost is $2 \times 1 + 3 \times 3 = 11$. Also, Figure 2(c) gives the output of the r-CELLULAR CLUSTERING algorithm applied to the quasi-identifiers shown as points in a metric space in Figure 2(a). The total cost of the solution in Figure 2(c) is: $50 \times 10 + 20 \times 5 + 8 \times 3 = 624$. We study a more generalized version of the problem: similar to the FACILITY LOCATION problem [Jain and Vazirani 1999], we add an additional setup cost for each potential cluster center, associated with opening a cluster centered at that point, but we don't have the lower bound on number of points per cluster. We call this the CELLULAR CLUSTERING problem. In fact, we will use the setup costs in the CELLULAR CLUSTERING problem formulation to help us devise an algorithm that solves r-CELLULAR CLUSTERING.

Comparison with k-Anonymity. While k-Anonymity forces one to suppress or generalize an attribute value even if all but one of the records in a cluster have the same value, the above clustering-based anonymization technique allows us to pick a cluster center whose value along this attribute dimension is the same as the common value, thus enabling us to release more information without losing privacy. For example, consider the table in Figure 3 with the Hamming distance metric on the row vectors. If we wanted to achieve 5-Anonymity, we will have to hide all the entries in the table, resulting in a total distortion of 20. On the other hand, a 5-CELLULAR CLUSTERING solution could use $(1, 1, 1, 1)$ as the cluster center with a cluster radius of 1. This will give a total distortion bound of 5 (the actual distortion is only 4).

	Attr1	Attr2	Attr3	Attr4
Record 0	1	1	1	1
Record 1	0	1	1	1
Record 2	1	0	1	1
Record 3	1	1	0	1
Record 4	1	1	1	0

Fig. 3. A sample table where there is no common attribute among all entries.

Just like k-Anonymity, r-GATHER and r-CELLULAR CLUSTERING is sensitive to outlier points, with just a few outliers capable of increasing the cost of the clustering significantly. To deal with this problem, we generalize the above algorithms to allow an ϵ fraction of the points to be deleted before publication. By not releasing a small fraction of the database records, we can ensure that the data published for analysis has less distortion and hence is more useful. This can be done as long as our aim is to infer macro trends from the published data. On the other hand, if the goal is to find out anomalies, then we should not ignore the outlier points. There has been no previous work for k-Anonymity with this generalization.

We note that, as in k-Anonymity, the objective function is oblivious to the sensitive attribute labels. Extensions to the k-Anonymity model, like the notion of l-diversity [Machanavajjhala et al. 2006], can be applied independently to our clustering formulation.

We provide constant factor approximation algorithms for both the r-Gather and r-Cellular Clustering problems. In particular, we first show that it is NP-hard to approximate the r-Gather problem better than 2 and provide a matching upper bound. We then provide extensions of both these algorithms to allow for an ϵ fraction of unclustered points, which we call the (r, ϵ)-Gather and (r, ϵ)-Cellular Clustering, respectively. These are the first constant factor approximation algorithms for publishing an anonymized database. The best known algorithms [Aggarwal et al. 2005; Meyerson and Williams 2004] for previous problem formulations had an approximation ratio linear in the anonymity parameter r.

The rest of the paper is organized as follows. First, in Section 2, we present a tight 2-approximation algorithm for the r-Gather problem and its extension to the (r, ϵ)-Gather problem, giving a 4-approximation for this case. We also present a 4-approximation for the (k, r, ϵ)-CENTER problem.

In Section 3, motivated by the desire to reduce the sum of the distortions experienced by the points, we introduce the problem of Cellular Clustering. We present a primal-dual algorithm for the problem without any cluster-size constraints that achieves an approximation ratio of 3. We then study the additional constraint of having a minimum cluster size of r, and for this case the approximation ratio is at most 36. Finally, we relax the problem by allowing the solution to leave at most an ϵ fraction of the points unclustered. We conclude in Section 4.

2. R-GATHER CLUSTERING

To publish the clustered database, we publish three types of features for each cluster: (1) the quasi-identifying attribute values for the cluster center, (2) the number of points within the cluster, and (3) a set of values taken by the sensitive attributes. The maximum cluster radius is also published to give a bound on the error introduced by clustering. This is similar to the traditionally studied k-Center clustering. In order to ensure r-Anonymity, we don't restrict the total number of clusters, instead, we pose the alternative restriction that each cluster should have at least r records assigned to it. We call this problem r-Gather, which we formally define below.

Definition 2.1. The r-Gather problem is to cluster n points in a metric space into a set of clusters, such that each cluster has at least r points. The objective is to minimize the maximum radius among the clusters.

We note that the minimum cluster size constraint has been considered earlier in the context of facility location [Karger and Minkoff 2000].

We first show the reduction for NP-completeness and hardness proofs.

2.1 Lower Bound

We show that this problem is NP-complete by a reduction from the 3-Satisfiability problem, where each literal belongs to at most 3 clauses [Garey and Johnson 1990].

Suppose that we have a boolean formula $\mathcal{F}$ in 3-CNF form with m clauses and n variables. Let $\mathcal{F} = C_1 \wedge \ldots \wedge C_m$, be a formula composed of variables $x_i, i = 1 \ldots n$ and their complements $\overline{x_i}$.

From the boolean formula, we create a graph $G = (V, E)$ with the following property: There is a solution to the r-Gather problem with a cluster radius of 1, with respect to the shortest distance metric on the graph G, if and only if $\mathcal{F}$ has a satisfying assignment.

We create the graph as follows: For each variable x_i, create two vertices v_i^T and v_i^F, and create an edge (v_i^T, v_i^F) between the two vertices; in addition create a set S_i of $(r-2)$ nodes and add edges from each node in S_i to both v_i^T and v_i^F. Picking v_i^T (v_i^F) as a center corresponds to setting $x_i = T$ (F). For each clause C_j, create a new node u_j that is adjacent to the nodes corresponding to the literals in the clause. For example, if $C_1 = (x_1 \vee \overline{x_2})$ then we add edges from u_1 to v_1^T and v_2^F. (Note that we cannot choose both v_i^T and v_i^F since there are not enough nodes in S_i.)

If the formula is indeed satisfiable, then there is a clustering by picking v_i^T as a center if $x_i = T$ and picking v_i^F otherwise. Each clause is true, and must have a neighbor chosen as a center. Moreover by assigning S_i to the chosen center, we ensure that each center has at least r nodes in its cluster.

Now suppose there is an r-gather clustering. If $r > 6$ then both v_i^T and v_i^F cannot be chosen as centers. In addition, the clause nodes u_j have degree at most 3 and cannot be chosen as centers. If exactly one of v_i^T or v_i^F is chosen as a center, then we can use this to find the satisfying assignment. The assignment is satisfying as each clause node has some neighbor at distance 1 that is a chosen center, and makes the clause true.

This completes the NP-completeness proof. Note that this reduction also gives us a hardness of 2. We just showed that there is a solution to the r-Gather problem with a cluster radius of 1 if and only if $\mathcal{F}$ had a satisfying assignment. The next available cluster radius is 2 in the metric defined by the graph G.

2.2 Upper Bound

We first use the threshold method used for k-Center clustering to guess R, the optimal radius for r-Gather. The choices for R are defined as follows. We will try all values $\frac{1}{2}d_{ij}$ where d_{ij} is the distance between points i and j. Note that this defines a set of $O(n^2)$ distance values. We find the smallest R for which the following two conditions hold:

Condition (1). Each point p in the database should have at least $r-1$ other points within distance $2R$ of p.

Condition (2). Let all nodes be unmarked initially. Consider the following procedure: Select an arbitrary unmarked point p as a center. Select all unmarked points within distance $2R$ of p (including p) to form a cluster and mark these points. Repeat this as long as possible, until all points are marked. Now we try to reassign points to clusters to meet the requirement that each cluster has size at least r. This is done as follows. Create a flow network as follows. Create a source s and sink t. Let C be the set of centers that were chosen. Add edges with capacity r from s to each node in C. Add an edge of unit capacity from a node $c \in C$ to a node $v \in V$ if their distance is at most $2R$. Add edges of unit capacity from nodes in V to t and check to see if a flow of value $r|C|$ can be found (saturating all the edges out of s). If so, then we can obtain the clusters by choosing the nodes to which r units of flow are sent by a node $c \in C$. All remaining nodes of V can be assigned to any node of C that is within distance $2R$. If no such flow exists, we exit with failure.

The following lemma guarantees that the smallest R that satisfies these conditions is a lower bound on the value of the optimal solution for r-GATHER. Suppose we have an optimal clustering $S_1, \ldots, S_\ell$ with ℓ clusters. Let the maximum diameter of any of these clusters be d^* (defined as the maximum distance between any pair of points in the same cluster).

LEMMA 2.2. *When we try $R = \frac{d^*}{2}$, then the above two conditions are met.*

PROOF. By the definition of r-GATHER, every point has at least $r-1$ other points within the optimal diameter, and hence within distance $2R$. Consider an optimal r-GATHER clustering. For each point i, all points belonging to the same optimal cluster c as the point i are within a distance $2R$ of i. Thus, in the procedure of Condition (2), as soon as any point in c is selected to open a new cluster, all unmarked points belonging to c get assigned to this new cluster. So at most one point from each optimal cluster is chosen as a center and forms a new cluster. We would now like to argue that the reassignment phase works correctly as well. Let S be the set of chosen centers. Now consider an optimal solution with clusters, each of size at least r. We can assign each point of a cluster to the center that belongs to that cluster, if a center was chosen in the cluster. Otherwise, since the point was marked by the algorithm, some center was chosen that is within distance $2R$. We can assign this point to the center that had marked it. Each chosen center will have at least r points assigned to it (including itself). □

Since we find the smallest R, we will ensure that $R \leq d^*/2 \leq R^*$ where R^* is the radius of the optimal clustering. In addition, our solution has radius $2R$. This gives us a 2-approximation.

THEOREM 2.3. *There exists a polynomial time algorithm that produces a 2-approximation to the r-GATHER problem.*

2.3 (r, ϵ)-Gather Clustering

A few outlier points can significantly increase the clustering cost under the minimum cluster size constraint. We consider a relaxation whereby the clustering solution is allowed to leave an ϵ fraction of the points unclustered, i.e., to delete an ϵ fraction of points from the published k-anonymized table. Charikar et al. [Charikar et al. 2001] studied various facility location problems with this relaxation and gave constant factor approximation algorithms for them.

For the (r, ϵ)-GATHER problem, where each cluster is constrained to have at least r points and an ϵ fraction of the points are allowed to remain unclustered, we modify our r-GATHER algorithm to achieve a 4-approximation. We redefine the condition to find R. We find the smallest R that satisfies the following condition: There should be a subset S of points containing at least $1-\epsilon$ fraction of the points, such that each point in S has at least $r-1$ neighbors within distance $2R$ in S.

This condition can be checked in $O(n^2)$ time by repeatedly removing any point in S that has fewer than $r-1$ other points in S within distance $2R$ of itself, with S initially being the entire vertex set. It is clear that the smallest R we found is no more than R^*, the optimal radius.

Let R be the value that we found. Let $N(v)$ denote the set of points in S within distance $2R$ of v, including v itself. We know then $N(v) \geq r$. We then consider the following procedure: Select an arbitrary point p from S. If there are at least $r-1$ other points within distance $2R$ of p, then form a new cluster and assign p and all points within distance $2R$ of p to this cluster. Remove all these points from further consideration and repeat this process until all remaining points have fewer than $r-1$ other points within distance $2R$ of them. Let U be the set of points left unclustered at the end of this process. For each $u \in U$, there exists a point $p \in N(u)$ such that p is assigned to some cluster c in the procedure of forming clusters. We can see this as follows. Since u was left unassigned at the end of the procedure, there are fewer than r unassigned points remaining in $N(u)$. This implies that there is at least one point p in $N(u)$ which is already assigned to some cluster c. We assign u to c, which already has at least r points.

Thus, we have assigned all points to clusters, such that each cluster has at least r points. Note that the radius of each cluster is no more than $4R$. This gives us the following theorem.

THEOREM 2.4. *There exists a polynomial time algorithm that produces a 4-approximation to the (r,ϵ)-GATHER problem.*

We note that in the problem formulation of (r,ϵ)-GATHER, if we require the cluster centers to be input points, instead of arbitrary points in the metric, then we can improve the approximation factor to 3 as follows. In the filtering step we define "candidates" as the set of points that have at least r points within radius R. The total number of points within distance R of the candidates should contain at least $1-\epsilon$ fraction of the points. Call this set S. Each point in S has at least $r-1$ neighbors within distance $2R$ in S. In the initial phase we greedily pick clusters of radius R (instead of $2R$) that have at least r points and mark those points covered. If a point in S is now uncovered, it must have a candidate within distance R that was unable to form a cluster. This is because some of the points within distance R of the candidate were covered in the first phase by disks of radius R. Hence each point in S can reach such a cluster center within distance $3R$ (through the candidate).

2.4 Combining r-Gather with k-Center

We can combine the r-GATHER problem with the k-CENTER problem and have the two constraints present at the same time. That is, we minimize the maximum radius, with the constraint that we have no more than k clusters, each must have at least r members. We call this the (k,r)-CENTER problem.

It is worth mentioning that a similar problem has been studied before in the k-CENTER literature. That is, instead of having a lower bound r on the cluster size as an additional constraint to the original k-CENTER formulation, an upper bound on the cluster size is specified. This is called the CAPACITATED k-CENTER problem [Khuller and Sussmann 2000]. Bar-Ilan, Kortsarz, and Peleg [Bar-Ilan et al. 1993] gave the first constant approximation factor of 10 for this problem. The bound was improved subsequently to 5 by Khuller and Sussmann [Khuller and Sussmann 2000]. In this subsection though we only concentrate on the (k, r)-CENTER problem defined above.

We note here that the algorithm developed for r-GATHER in Subsection 2.2 can be extended to provide a 2-approximation for the (k, r)-CENTER problem. We just have to add to Condition (2) the extra criteria that if the number of centers chosen exceeds k then exit with failure, i.e., try a different value for R. We can show that Lemma 2.2 holds for the modified conditions, hence an approximation factor of 2.

We also consider the outlier version of this problem, namely, the (k, r, ϵ)-CENTER problem.

We show that the following algorithm is a 4-approximation algorithm for the (k, r, ϵ)-CENTER problem.

Fix a guess for the optimal radius R (choose the smallest R that succeeds). For each such guess, we apply the following algorithm. Let $D(v, \delta)$ be the set of points within distance δ of v (including v).

Algorithm:

(Filtering Step) Let S be the set of points v such that $|D(v, 2R)| \geq r$. Check to see if $|S| \geq (1 - \epsilon)n$, otherwise exit with failure. From now on we only consider points in S.

(Greedy Step) We now choose up to k centers. We put the centers in the set Q. Initially Q is empty. All points are uncovered to start with. Let $N(v, \delta)$ be the set of *uncovered points* within distance δ of v (including v itself). Points are either uncovered, or covered. Once a point is covered it is removed from consideration. At each step i, we choose a center c_i that satisfies the following criteria:
(a) c_i is uncovered.
(b) $|N(c_i, 2R)|$ is maximum.
All uncovered points in $N(c_i, 4R)$ are then marked as covered.

After Q is completely decided, check that the total points covered is at least $(1 - \epsilon)n$, otherwise exit with failure.

(Assignment step): Form clusters as follows. For each $c_i \in Q$, form a cluster C_i centered at c_i. Each covered point is assigned to its closest cluster center.

For each c_i, we denote $G_i = N(c_i, 2R)$ and $E_i = N(c_i, 4R)$, which are uncovered points within distance $2R$ and $4R$ of c_i, when c_i is chosen.

In Figure 4 we illustrate this algorithm via an example. Let's start from the Greedy Step and assume for a moment that R is indeed the optimal radius just for illustration purposes. Let the optimal solution have clusters $O_1, \ldots, O_k$. In the figure, we only show cluster centers to be picked and omit the rest of the points. The left side illustrates the situation when we are picking c_1. Note that G_1 includes O_1 completely, and overlaps with O_2. Because of this, all points in O_1 and O_2 are in E_1 and marked covered and cannot be chosen as a center later. Note that E_1 in fact will cover points in other optimal clusters as well. For example, when we choose c_1 and cover all points in E_1, we also cover some points in O_3. However, we

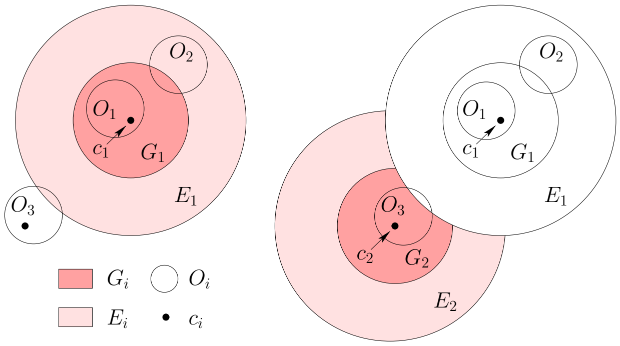


Fig. 4. Optimal Clusters and the Greedy Step

may still pick a remaining point in O_3 as the next cluster center c_2, as shown in the right side. Note that in the Greedy Step, we completely ignore the constraint of r, as we are not forming any clusters but only picking cluster centers. In fact, G_2 now could have fewer than r uncovered points. The key is that the G_i's are far apart. Hence in the Assignment Step, all the points in $D(c_2, 2R)$ that were initially covered by E_1 will eventually be assigned to the center c_2, giving the whole cluster C_2 at least r points. Detailed proofs are below.

LEMMA 2.5. *After the assignment step, each cluster formed has at least r points, and radius at most* $4R$.

PROOF. Every time a center c_i is selected, we only cover points within distance $4R$, thus the maximum radius is at most $4R$. In the end, each point is assigned to its closest chosen center in Q. Observe that the cluster centers are more than $4R$ apart. Thus for each center c_i and its corresponding cluster C_i, all the points within distance $2R$ of c_i are assigned to the same cluster C_i. By the filtering step, we know that $|C_i| \geq |D(c_i, 2R)| \geq r$. $\square$

LEMMA 2.6. *The optimal solution on set S is the same as the optimal solution on set V.*

PROOF. This is simply true by the filtering step, since every point in the optimal solution belongs to S. $\square$

LEMMA 2.7. *Consider the guess $R = \frac{d^*}{2}$, where d^* is the maximum distance between any two points in the same optimal cluster, our algorithm covers no less points than the optimal solution on set S.*

PROOF. We are going to prove a stronger statement, we'll show that our algorithm covers no less points than the following optimal solution OPT on set S: it has at most k clusters, and the maximum distance between any two points in the same optimal cluster is at most d^*, but there is no requirement on the number of points per cluster. Let $O_1, O_2, \ldots, O_k$ denote the set of optimal clusters in OPT. We claim that:

$$|E_1 \cup \ldots \cup E_k| \geq |O_1 \cup \ldots \cup O_k| \tag{1}$$

The proof is by induction on k. The claim is true for $k = 1$, since $|E_1| \geq |G_1| \geq |O_1|$. Assume that $k > 1$. Clearly,

$$\bigcup_{i=1}^{k} (E_1 \cap O_i) \subseteq E_1.$$

Assume that G_1 intersects one of the disks $O_1, \ldots, O_k$ (say, O_1). Then $O_1 \subseteq E_1$ and the following inequality is satisfied.

$$|E_1| \geq |O_1| + \sum_{i=2}^{k} |E_1 \cap O_i|. \tag{2}$$

The above inequality is satisfied even if G_1 does not intersect any of the disks $O_1, \ldots, O_k$, since then

$$\bigcup_{i=1}^{k} (E_1 \cap O_i) \cup G_1 \subseteq E_1.$$

Now since $|G_1| \geq \max\{|O_1|, |O_2|, \ldots, |O_k|\} \geq |O_1|$, we have

$$|E_1| \geq |G_1| + \sum_{i=1}^{k} |E_1 \cap O_i| \geq |O_1| + \sum_{i=2}^{k} |E_1 \cap O_i|.$$

In either case, inequality (2) is satisfied.

Now consider the $(k-1)$-center problem on the set $S - E_1$. On this set, our algorithm could have picked the sets $E_2, E_3, \ldots, E_k$. Also, for $S - E_1$, it is clear that $O_2 - E_1, O_3 - E_1, \ldots, O_k - E_1$ is a solution, although it is not necessarily an optimal one. By induction, we know that

$$|E_2 \cup \ldots \cup E_k| \geq |\bigcup_{i=2}^{k} (O_i - E_1)| \tag{3}$$

Combining inequalities (2) and (3) proves (1). □

Combining the above three lemmas we have the following theorem.

THEOREM 2.8. *Our algorithm gives a 4-approximation for the* (k, r, ϵ)-CENTER *problem.*

3. CELLULAR CLUSTERING

As mentioned in the introduction, a second approach is to publish the radius of each cluster in addition to its center and the number of points within it. In this case, for each point within a cluster, the radius of the cluster gives an upper bound on the distortion error introduced. The CELLULAR CLUSTERING problem aims to minimize the overall distortion error, i.e., it partitions the points in a metric space into cells, each having a cell center, such that the sum, over all cells, of the products of the number of points in the cell and the radius of the cell is minimized. We even allow each potential cluster center to have a facility (setup) cost f_c associated with opening a cluster centered at c. This will later allow us to solve the problem in the case when each cluster is required to have at least r points within it.

Definition 3.1. A cluster consists of a center along with a set of points assigned to it. The *radius* of the cluster is the maximum distance between a point assigned to the cluster and the cluster center. To open a cluster with cluster center c and radius r incurs a *facility cost* f_c. In addition, each open cluster incurs a *service cost* equal to the number of points in the cluster times the cluster radius. The sum of these two costs is called the *cellular cost* of the cluster. The Cellular Clustering problem is to partition n points in a metric space into clusters with the minimum total cellular cost.

We first show that the problem is NP-complete via a reduction from set cover.

Theorem 3.2. *The* Cellular Clustering *problem is* NP*-complete.*

Proof. We can view the set cover problem as a bipartite graph (S, X, E) where S contains a vertex corresponding to each set, X contains a vertex corresponding to each element, and E is the set of (set,element) pairs denoting membership. We then set up a Cellular Clustering problem as follows: Each vertex in the bipartite graph corresponds to a point in the metric space. The distance between the points is defined as follows: the points are adjacent to each other according to the edges defined by E. In addition, the points in the set S form a clique, i.e., they are adjacent to each other. All points adjacent to each other have distance 1. We can assume that the distance between any set s_i and a vertex $x_j \in X$ is 2 if $x_j \notin s_i$. The facility cost for each point in S is 1, and the facility cost for each point in X is very high.

We observe that there is a collection of k sets covering all the elements, if and only if there is a cellular clustering with cost $k + |X| + |S|$. If a collection of k sets chosen from S exist that cover all elements in X, then those k sets are chosen as clusters of radius 1, with each element belonging to such a cluster. We pay a total of k for the facility cost, and every point also incurs a radius cost of 1 since every point belongs to a cluster of radius 1.

We now show that a cellular clustering cost of $|X| + |S| + k$ also corresponds to a set cover of size k. From the setup, it is obvious that cluster centers can only reside on the points in S. Assume that in the solution, k' such clusters centers are chosen. Without loss of generality, we can assume that each of the k' clusters have radius at least 1, thus each point has to pay a radius cost of at least 1. Otherwise we can perform the following modification for each cluster of radius 0. Such a cluster has a single member, which corresponds to a point in S. We close down this cluster, saving a facility cost of 1. And assign this point to a cluster of radius 1, encountering a radius cost of 1. A cluster of radius 1 has to exist. Otherwise the total cost of the solution would be at least $2|X| + k' + 2(|S| - k')$, where each point in X has to pay a radius of cost 2, and each point in S has to either pay a facility cost of 1 or a radius cost of 2. We can rewrite it as $|X| + |S| + |X| + (|S| - k')$ to see that obviously this can't happen.

Obviously if $k' > k$, then the total cluster cost will be at least $k' + |X| + |S|$. Assume $k' < k$. We'll modify the solution without increasing the cost so that each cluster has a radius of 1, which will yield a set cover solution. Consider any cluster of radius 2, centered at s_i. Let X_i denote the set of points in X that belong to this cluster and are of distance 2 from s_i. For each point $x_j \in X_i$, we simply find a set s_{ij} such that $x_j \in s_{ij}$, and assign x_j to the cluster centered at s_{ij}. If s_{ij} was already chosen as a cluster center, then x_j only has to pay a radius cost of 1, and we don't incur any additional facility cost. Otherwise, we save 1 on the radius cost for x_j, and pay for an additional facility cost of 1 for s_{ij}. $\square$

We present a primal-dual algorithm for the CELLULAR CLUSTERING problem that achieves an approximation factor of 3.

Let $c = (v_c, d_c)$ denote a cluster c whose cluster center is the node v_c and whose radius is d_c. By definition, the setup cost f_c for a cluster $c = (v_c, d_c)$ depends only on its center v_c; thus $f(c) = f(v_c)$. For each possible choice of cluster center and radius $c = (v_c, d_c)$, define a variable y_c, a 0/1 indicator of whether or not the cluster c is open. There are $O(n^2)$ such variables. For a cluster $c = (v_c, d_c)$, any point p_i within a distance of d_c of its center v_c is said to be a *potential member* of the cluster c. For all potential members p_i of a cluster c, let x_{ic} be a 0/1 indicator of whether or not point p_i joins cluster c. Note that the pair (i, c) uniquely identifies an edge between p_i and the center of cluster c. We relax the integer program formulation to get the following linear program:

$$
\begin{array}{rll}
\text{Minimize:} & \sum_c (\sum_i x_{ic} d_c + f_c y_c) & \\
\text{Subject to:} & \sum_c x_{ic} \geq 1 & \forall i \\
& x_{ic} \leq y_c & \forall i, c \\
& 0 \leq x_{ic} \leq 1 & \forall i, c \\
& 0 \leq y_c \leq 1 & \forall c
\end{array}
$$

And the dual program is:

$$
\begin{array}{rll}
\text{Maximize:} & \sum_i \alpha_i & \\
\text{Subject to:} & \sum_i \beta_{ic} \leq f_c & \forall c \\
& \alpha_i - \beta_{ic} \leq d_c & \forall i, c \\
& \alpha_i \geq 0 & \forall i \\
& \beta_{ic} \geq 0 & \forall i, c
\end{array}
$$

The above formulation is similar to the approach used for facility location [Jain and Vazirani 1999]. However, since the assignment of additional points to clusters increases the service cost incurred by existing members of the cluster, we need a slightly different strategy to form clusters and assign points to clusters.

The procedure of raising the dual variables until we find a feasible dual solution is identical to the procedure for facility location given by Jain and Vazirani [Jain and Vazirani 1999]. For completeness we present the procedure here. In this procedure, each point will raise its dual variable α_i until it is connected to an open cluster.

Initially, each point is *unconnected*, and its dual variables α_i and β_{ic}'s are set at zero. The algorithm will raise the dual variable α_i for all *unconnected* points uniformly. For convenience, we introduce the notion of time, the dual variables increase by 1 per time unit. When $\alpha_i = d_c$ for some edge (i, c), the edge (i, c) is declared to be tight. At this point, the dual variable β_{ic} is raised at the uniform rate with the rest of the α's. β_{ic} goes towards paying for f_c.

Cluster c is said to be paid for when $\sum_i \beta_{ic} = f_c$. The algorithm then checks to see if there are any *unconnected* points that have a tight edge to c. If so, then c is declared *temporarily open* and all *unconnected* points with a tight edge to c are declared *connected*. Cluster c is said to be the *connecting witness* for each of these points. Notice that the dual variables α_i and β_{ic}'s for points with tight edges to c are no longer growing.

We now show how to construct a primal solution with cost at most 3 times the value of the dual solution found above. For this, we note the following properties:

. (1) At any time, the value of α_i for all *unconnected* points i is the same. Moreover, this value is no less than the value of α_j for any *connected* point j.

. (2) Once a point has a tight edge to a particular cluster c, all *unconnected* potential members of that cluster (i.e., points within a distance d_c of the cluster center v_c) have tight edges to it.

. (3) Examine any *temporarily opened* cluster c. All points for which c is a *connecting witness* have the largest α values among all points that have tight edges to c.

Property (1) follows from the definition of our procedure. Property (2) follows from property (1) and the fact that the edge (i, v_c) becomes tight when the dual variable α_i equals d_c. Property (3) then follows from (2).

We construct a primal solution as follows: initially all points are *unassigned*. Order the *temporarily open* clusters in decreasing order of radius. Consider the clusters one by one, we will *permanently open* a cluster c if all points with tight edges to c are still unassigned. Once c is open, all points with tight edges to c are *directly assigned* to c. On the other hand, if the cluster c being considered has a point i, such that i is already assigned elsewhere and i has a tight edge to c, we'll shut down c, and consider the next cluster.

After all the clusters are considered, we might have some points still left unassigned. Consider any such a point i, and consider the cluster c' that is the *connecting witness* for i. If i is still *unassigned*, then c' was shut down. There must exist a point j with tight edges to c', such that j was *directly assigned* to some cluster c, which has a larger radius compared to c'. We will have i join c, and i becomes *indirectly assigned* to c. Notice that after i joins c, the cluster c may now have a new radius, since i could be potentially more than d_c away from c. Rename c as c° instead. The radius of the new cluster c° can be bounded as follows:

LEMMA 3.3. *For a specific cluster c°, $d_{c^\circ} \leq 3d_c$. Recall that d_c is the original cluster radius of cluster $c = (v_c, d_c)$.*

PROOF. We inherit the notation of i, j, c', and c° as above. we have: $d(i, v_c) \leq d(i, v_{c'}) + d(v_{c'}, j) + d(j, v_c) \leq 2d_{c'} + d_c \leq 3d_c$. □

We now bound the cost of the primal solution. Let $\mathcal{C}$ be the set of open centers. Some of the points are directly assigned, and others are indirectly assigned to a cluster centered at c°. We consider the set of points $\mathcal{P}_{c^\circ}$ assigned to a particular open cluster c°.

The following lemma gives a bound on the service cost of all points assigned to c° and three times its facility cost.

LEMMA 3.4. $(\sum_{i \in \mathcal{P}_{c^\circ}} x_{ic^\circ} d_{c^\circ} + 3 f_{c^\circ} y_{c^\circ}) \leq 3 \times \sum_{i \in \mathcal{P}_{c^\circ}} \alpha_i$.

PROOF. Consider the cluster c° to which point i is *indirectly assigned*. By property (3), $\alpha_i \geq \alpha_j$ for some j with tight edges to v_c and $\alpha_j \geq d_c$. Combined with

Lemma 3.3, we have $3\alpha_i \geq 3\alpha_j \geq 3d_c \geq x_{ic^\circ} d_{c^\circ}$. Consider the point j which is *directly assigned* to c°, $3\alpha_j = 3d_c + 3\beta_{jc} \geq 3x_{ic^\circ} d_{c^\circ} + 3\beta_{jc^\circ}$. Since we only *permanently open* the cluster c° when we have all the points with tight edges to v_c *directly assigned* to c°, $3\sum_j \beta_{jc^\circ} = 3f_{c^\circ} y_{c^\circ}$. □

Summing over all open clusters in $\mathcal{C}$ gives the bound. We thus obtain the following theorem. Note that we are actually paying 3 times of the original cluster cost for each of the original cluster c. And the final clusters constructed have radius at most 3 times that of the original clusters. These facts will become important for the r-Cellular Clustering problem discussed in Subsection 3.1.

Theorem 3.5. *The primal-dual method described above produces a 3-appro-ximation solution to the* Cellular Clustering *problem.*

3.1 r-Cellular Clustering

We now extend the above primal-dual algorithm to get an approximation algorithm for the r-Cellular Clustering problem which has the additional constraint that each cluster is required to have at least r members. The notation (r, C) is used to denote a solution having a total cost of C, and having at least r members in each cluster.

Comparison with prior clustering work. Since our algorithm can be viewed as an extension of facility location, we briefly discuss related results. The facility location (and k-median) problems have been studied with the minimum cluster size constraint [Karger and Minkoff 2000], as well as in the context of leaving an ϵ fraction of the points unclustered [Charikar et al. 2001]. Let OPT_r be the optimal facility location cost with minimum cluster size r. Recall that (r, C) denotes a solution with minimum cluster size r and solution cost C. Bi-criteria approximations for the facility location problem of the form $(\alpha \cdot r, \frac{1+\alpha}{1-\alpha}\beta \cdot OPT_r)$ were achieved independently by Guha, Meyerson and Munagala and by Karger and Minkoff [Guha et al. 2000; Karger and Minkoff 2000]. Here, β refers to the best approximation factor for facility location, and α is an adjustable parameter between 0 and 1. It is not known whether it is possible to achieve a one-sided approximation on facility location cost alone. In contrast, for the r-Cellular Clustering problem, we provide an one-sided approximation algorithm, specifically we obtain a $(r, 36OPT_r)$ solution, where OPT_r is the cost of the optimal solution with cluster size at least r,

To achieve this, we first study a *sharing* variant of this problem, where a point is allowed to belong to multiple clusters, thus making it easier to satisfy the minimum cluster size constraint. Interestingly, allowing sharing changes the value of the optimal solution by at most a constant factor. We note that this observation does not hold for facility location, where a shared solution might be arbitrarily better than an unshared one. The algorithm consists of three main steps:

1. Augmenting with Setup Costs. Given an instance of r-Cellular Clustering, we first construct an instance of Cellular Clustering as follows: augment the cluster cost f_c of a cluster c by $r \times d_c$. In addition, if a cluster $c = (v_c, d_c)$ has fewer than r points within distance d_c of its center v_c, this cluster is eliminated from the instance. If the original r-Cellular Clustering instance has an optimal solution with cost OPT_r, it is not hard to see that the same solution works for the Cellular Clustering instance constructed above with a total cost of at most $2OPT_r$. We invoke the 3-approximation algorithm for Cellular Clustering on this new instance to find a solution with cost at most $6OPT_r$.

2. Sharing Points between Clusters. We now describe the notion of a *shared* solution for r-Cellular Clustering. In a shared solution, points are allowed to be assigned to multiple clusters, as long as they pay the service cost for each cluster they are assigned to. A shared solution is *feasible* if all clusters have at least r (potentially shared) members. We modify the solution obtained above to get a feasible shared solution for r-Cellular Clustering as follows: for each open cluster c with center P, assign the r closest neighbors of P to c as well, regardless of where they are initially assigned. The extra service cost of at most $r \times 3d_c$ for these r points can be accounted for by the extra 3 times the cluster cost of $r \times d_c$ being paid by the open cluster c in the Cellular Clustering solution. Thus, we have obtained an $(r, 6OPT_r)$ shared solution for the r-Cellular Clustering instance.

3. Making the Clusters Disjoint. Finally we show how to convert a shared solution to a valid solution where each point is assigned to only one cluster, with only a constant blowup in cost. We note that for the corresponding facility location problem, it is not feasible to do this "unsharing" without a large blowup in cost in the worst case.

Initially, all points are labeled *unassigned*. We consider the clusters in order of increasing cluster radius d_c. If a cluster c has at least r *unassigned* members, then it is opened and all its *unassigned* members are assigned to c and labeled *assigned*. We stop this process when all the remaining clusters have fewer than r *unassigned* members each. The remaining clusters are called *leftover* clusters. We temporarily assign each of the *unassigned* points arbitrarily to one of the leftover clusters it belongs to. Since each cluster had at least r members in the shared solution, each leftover cluster c' must have a member in the shared solution, which is now *assigned* to an open cluster o, s.t. $d_{c'} \geq d_o$. We thus have the situation illustrated in Figure 5.

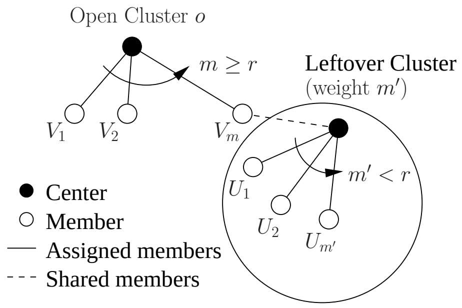


Fig. 5. Structures of open and leftover clusters

The points are organized in a forest structure, where each tree has two "levels". We can regroup points into clusters, on a per tree basis. It is obvious that each tree has at least r points, since it contains at least one open cluster o. We further simplify the structure into a true two-level structure as in Figure 5, by collapsing each leftover cluster into a single node with weight equal to the number of points temporarily assigned to it. Nodes in the first level of the tree have weight 1. We apply the following greedy grouping procedure: first consider only the nodes at the second level of the tree and collect nodes until the total weight exceeds r for the first time. We group these nodes (belonging to leftover clusters) into a cluster, and repeat the process. Notice that since we did not touch the first-level nodes, the total weight of remaining nodes in the tree is at least r. If the total weight of remaining nodes in the second level, W_s, is less than r, then we extend the grouping into the first level nodes. Let m denote the total weight of nodes in the first level. If $W_s + m \geq 2r$, then we group the nodes in the second level with $r - W_s$ first level nodes together into a cluster; the remaining nodes in the first level form a cluster. Otherwise, all the remaining nodes (both the first and second level) are grouped into a cluster. If we break up the tree using the procedure above, each resulting cluster has size at least r.

LEMMA 3.6. *For a cluster that contains any second-level nodes, the total number of points in the cluster is no more than* $2r - 1$.

PROOF. Since a single second-level node has weight less than r, a cluster containing only second-level nodes has at most $2r - 1$ members. If the cluster contains both the first and second-level nodes, then we must have reached the case where the total weight of remaining nodes in the second level is less than r. In that case, by definition, the cluster formed containing these second-level nodes has size either r or less than $2r - 1$. □

There could be a cluster that only contains the first level nodes, and its entire cost (both the service and cluster cost) can be accounted for by its cost in the original $(r, 6OPT_r)$ shared solution. We now bound the cost of clusters containing the second-level nodes.

LEMMA 3.7. *For each cluster c formed that contains second level nodes, there exists a leftover cluster c' unique to c, such that the following holds: let p be the center of the initial open cluster o at the first level, if we center the cluster c at p, then the radius of cluster c,* $d_c \leq 3d_{c'}$.

PROOF. Among all the leftover clusters that contributed to c, let c' be the one with the maximum radius. By definition, all nodes assigned to a leftover cluster get assigned to a single cluster, guaranteeing the uniqueness of c'. Let d_o be the radius of the open cluster at level 1 of this tree. Consider a point $q \in c$. If q is a first-level node, then $d(q,p) \leq d_o \leq d_{c'}$. If q is a second-level node, then let c'' be the leftover cluster that q was assigned to, then $d(q,p) \leq 2d_{c''} + d_o \leq 3d_{c'}$. □

The above lemma implies that by choosing p as the cluster center, the service cost of each point in c is no more than $3d_{c'}$ and the total facility cost incurred within our solution is no more than that of the shared solution. Together with Lemma 3.6, we conclude that the service cost of points in c is no more than $6r \times d_{c'}$. Notice that in the shared solution, points in cluster c' are paying a total service cost of at least $r \times d_{c'}$. We thus have the following theorem.

THEOREM 3.8. *The above procedure produces a solution with minimum cluster size r and total cost no more than $36OPT_r$, i.e., a $(r, 36OPT_r)$ solution, where OPT_r is the value of the optimal solution with a minimum cluster size of r.*

We note that the above algorithm and analysis can be combined with the technique developed in [Charikar et al. 2001] to give a constant approximation to the (r, ϵ)-CELLULAR CLUSTERING problem.

4. CONCLUSIONS

Publishing data about individuals without revealing sensitive information is an important problem. The notion of privacy called k-Anonymity has attracted a lot of research attention recently. In a k-anonymized database, values of quasi-identifying attributes are suppressed or generalized so that for each record there are at least $k-1$ records in the modified table that have exactly the same values for the quasi-identifiers. However, the performance of the best known approximation algorithms for k-Anonymity depends linearly on the anonymity parameter k. In this paper, we introduced clustering as a technique to anonymize quasi-identifiers before publishing them. We studied r-GATHER as well as a newly introduced clustering metric called r-CELLULAR CLUSTERING and provided the first constant factor approximation algorithms for publishing an anonymized database table. Moreover, we generalized these algorithms to allow an ϵ fraction of points to remain unclustered.

REFERENCES

AGGARWAL, G., FEDER, T., KENTHAPADI, K., MOTWANI, R., PANIGRAHY, R., THOMAS, D., AND ZHU, A. 2005. Approximation algorithms for k-anonymity. *Journal of Privacy Technology*, Number 20051120001.

BAR-ILAN, J., KORTSARZ, G., AND PELEG, D. 1993. How to allocate network centers. *Journal of Algorithms 15*, 385–415.

BAYARDO, R. J. AND AGRAWAL, R. 2005. Data privacy through optimal k-anonymization. In *International Conference on Data Engineering*. 217–228.

CHARIKAR, M., KHULLER, S., MOUNT, D., AND NARASIMHAN, G. 2001. Algorithms for facility location with outliers. In *ACM-SIAM Symposium on Discrete Algorithms*. 642–651.

CHAWLA, S., DWORK, C., MCSHERRY, F., SMITH, A., AND WEE, H. 2005. Toward privacy in public databases. In *Theory of Cryptography Conference*. 363–385.

GAREY, M. R. AND JOHNSON, D. S. 1990. *Computers and intractability, a guide to the theory of NP-completeness*. W. H. Freeman and Company, New York.

GUHA, S., MEYERSON, A., AND MUNAGALA, K. 2000. Hierarchical placement and network design problems. In *IEEE Symposium on Foundations of Computer Science*. 603–612.

HOCHBAUM, D. AND SHMOYS, D. 1985. A best possible approximation algorithm for the k-center problem. *Mathematics of Operations Research 10*, 180–184.

JAIN, K. AND VAZIRANI, V. V. 1999. Primal-dual approximation algorithms for metric facility location and k-median problems. In *IEEE Symposium on Foundations of Computer Science*. 2–13.

KARGER, D. AND MINKOFF, M. 2000. Building steiner trees with incomplete global knowledge. In *IEEE Symposium on Foundations of Computer Science*. 613–623.

KHULLER, S. AND SUSSMANN, Y. 2000. The capacitated k-center problem. *SIAM Journal on Discrete Mathematics 13,* 3, 403–418.

LEFEVRE, K., DEWITT, D., AND RAMAKRISHNAN, R. 2005. Incognito: Efficient full-domain k-anonymity. In *ACM SIGMOD International Conference on Management of Data*. 49–60.

MACHANAVAJJHALA, A., KIFER, D., GEHRKE, J., AND VENKITASUBRAMANIAM, M. 2006. l-diversity: Privacy beyond k-anonymity. In *International Conference on Data Engineering*. 24.

Meyerson, A. and Williams, R. 2004. On the complexity of optimal k-anonymity. In *Symposium on Principles of Database Systems*. 223–228.

Samarati, P. 2001. Protecting respondent's privacy in microdata release. *IEEE Transactions on Knowledge and Data Engineering 13,* 6, 1010–1027.

Sweeney, L. 2000. Uniqueness of simple demographics in the u.s. population. *LIDAP-WP4. Carnegie Mellon University, Laboratory for International Data Privacy, Pittsburgh, PA*.

Time. 1997. The death of privacy.

Received August 2007; revised June 2008; accepted July 2008