Multi-Probe Random Projection Clustering to Secure Very Large Distributed Datasets

Lee A. Carraher
University of Cincinnati
Cincinnati, OH 45221–0030
Email: carrahle@mail.uc.edu

Philip A. Wilsey
University of Cincinnati
Cincinnati, OH 45221–0030
Email: philip.wilsey@uc.edu

Anindya Moitra
University of Cincinnati
Cincinnati, OH 45221–0030
Email: moitraaa@mail.uc.edu

Sayantan Dey
University of Cincinnati
Cincinnati, OH 45221–0030
Email: deysn@mail.uc.edu

***Abstract*—This paper presents a solution to the approximate k-means clustering problem for very large distributed datasets. Distributed data models have gained popularity in recent years following the efforts of commercial, academic and government organizations, to make data more widely accessible. Due to the sheer volume of available data, in-memory single-core computation quickly becomes infeasible, requiring distributed multi-processing. Our solution achieves comparable clustering performance to other popular clustering algorithms, with improved overall complexity growth while being amenable to distributed processing frameworks such as Map-Reduce. Our solution also maintains certain guarantees regarding data privacy deanonimization.**

I. Introduction

Data clustering is an inherent problem in data analysis. It is the principle workhorse of many types of pilot data analysis and various data retrieval methods. Due to its importance in machine learning, data clustering is studied extensively in computing and mathematics [1]–[7]. Recent advances in data aggregation of online and distributed source provide nearly limitless sources of data to be clustered. For this reason we developed an algorithm to solve the popular version of clustering called the k-means clustering problem for distributed datasets. While providing certain data security assurances for data in transit and at rest, our solution offers comparable clustering performance to offline solutions on a variety of synthetic and real world datasets.

Clustering has long been the standard method used for the analysis of labeled and unlabeled data. The effect of clustering data allows for identifying dissimilar and similar objects in a dataset, that are often unattainable through standard single pass statistical methods. Single pass, data intensive, statistical methods are often the primary workhorses for database processing of business logic and scientific domains, while clustering is often overlooked due to issues with scalability [3]. A multitude of surveys [3] have been made available comparing different aspects of clustering algorithms, in regard to accuracy, complexity, and application domain. Of the multitude of fields that benefit from data clustering, medical data analysis is one that would benefit most from a secure scalable clustering platform. Such fields as Micro Array clustering, Protein-Protein interaction clustering, medical resource decision making, medical image processing, and clustering of epidemiological events all serve to benefit from larger dataset sizes.

Our algorithm, called *Random Projection Hash* (*RPHash*), was expressly created to provide algorithmic scalability by operating on distributed datasets. Many clustering algorithms have been converted to function efficiently on distributed data, however they often have potential issues regarding asymptotic scalability [8], dimensionality in which they are effective [9], and robustness to noise. Although many algorithms have been proposed for parallel clustering, many issues are still present when applied to very large, high dimensional, datasets [8]–[12].

The *RPHash* algorithm combines approximate and randomized methods in a new way to solve issues of scalability and data security for real world distributed data analysis. The focus on a randomized, and thus non-deterministic, algorithms is somewhat uncommon in computing, but is quite common for ill-posed, or combinatorially restrictive problems such as data clustering. This assertion is complemented by a similar inversion regarding algorithms in computing, in which data from real world datasets tends to converge much faster than synthetic datasets. Naturally adversarially crafted worse-case scenarios [13] can approach exponential time requirements. Our principle assumption with *RPHash* is that approximate clustering is qualitatively similar to exact clustering methods, due to noise, redundancy, data loss, and the curse of dimensionality [14].

The remainder of this article is organized as follows. Section II provides some background information on material necessary for covering various preliminaries important to the description of our *RPHash* algorithm. Section III reviews some of the recent work with similar methods for scalable data clustering. Section IV provides a detailed description of the *RPHash* algorithm. Section V describes how *RPHash* can cluster data across a collection of distributed databases while maintaining the security of the data therein. Section VI contains some experimental results and a brief summary of findings. Finally, Section VII contains some concluding remarks.

II. Background

Data clustering is a useful tool in pilot data and knowledge engineering. It is used in a range of problems from practical data localization and search, to helping augment the understanding of complex biological, social, and physical structures. The k-means clustering problem is often defined by the following

optimization over the set of k cluster centroids $\mu \in C$ on vectors $x \in X$.

$$\underset{C}{\operatorname{argmin}} \sum^{k} \sum_{x \in C} ||x - \mu_i||^2$$

k-means clustering has been shown to be NP-Hard for dimensions ≥ 2 [13], [15]. However much faster runtime algorithms exist achieving sub-optimal solutions to the k-means clustering problem for real world datasets, famously the classic Lloyd's algorithm.

A. Random Projection Methods

Random projection is a method where vectors are projected from their original embedding to a lower dimensional subspace by applying a projection matrix transform (where the projection matrix is composed of orthogonal vectors from some random distribution). Random projection embeddings of data can achieve bounded error distortion very near the L_2 norm optimal subspace embedding that would otherwise result from the far more computationally intensive principal component decomposition [16]. The resurgence of the random projection method of Johnson and Lindenstrauss was reinvigorated with the work of Achlioptas on Database Friendly Projection that provided good subspace embeddings requiring minimal computation costs [17]. The formal definition of a projection $f(\cdot)$ that preserves discrepancy between any two vectors u and u' with probability ϵ is given below.

Theorem 2.1 (JL - Lemma [18]):

$$(1-\epsilon)\|u-u'\|^2 \leq \|f(u)-f(u')\|^2 \leq (1+\epsilon)\|u-u'\|^2$$

Under the optimal ϵ-preserving mapping $f(\cdot)$, the Johnson-Lindenstrauss lemma results in a tight bound for $u, u' \subseteq U$ and $n = |U|$, of $d \sim \Theta(\frac{log(n)}{\varepsilon^2 log(1/\varepsilon)})$. The bound was later applied to random orthogonal projections in Frankl and Maehara, and found to have a similar order on the bound for the projected subspace dimensionality [19]. Vempala gives a relaxation of the JL-bound for random orthogonal projections, arriving at $d \sim \Omega(log(n))$ [18], with a scaling factor of $\frac{1}{\sqrt{n}}$ to preserve approximate distances between projected vectors. An additional benefit of using Random Projection for mean clustering is that randomly projected asymmetric clusters tend to become more spherical in lower dimensional subspace representations [20]. Mean and medoid clustering algorithms, such as *RPHash*, are predisposed to spherical clusters.

B. Discrete Subspace Quantizers

Switching from the continuous spaces of random projections, a discussion on discrete space partitioning lattices is pursued. For optimal implementations of the clustering problem, the data space must be partitioned as evenly as possible. Furthermore to avoid costly interprocess communication overhead, a universally generative naming scheme must be established. For known datasets, the perfect partition of the data space is produced by the Voronoi partitioning [21]. In 2-dimensional space the Voronoi Diagram can be generated in $\Theta(nlog(n))$-time [22]. However higher dimensional algorithms have less favorable run time complexities [23], making them inefficient for partitioning arbitrarily high dimensions. Instead, a finite lattice structure is considered, with rotation and shifting transformations to compensate for gaps in the covering radius of the lattice. A formal definition of a lattice constructed from a linear code is:

Definition 2.2 (Lattice in $\mathbb{R}^n$ [24]): let $v_1, ..., v_n$ be n linear independent vectors where $v_i = v_{i,1}, v_{i,2}, ..., v_{i,n}$. The lattice Λ with basis $\{v_1, ..., v_n\}$ is the set of all integer combinations of $v_1, ..., v_n$ the integer combinations of the basis vectors are the points of the lattice.

$$\Lambda = \{z_1 v_1 + z_2 v_2 + ... + z_n v_n | z_i \in \mathbb{Z}, 1 \leq i \leq n\}$$

Regular lattices formed from binary codes (such as the the E8 Lattice) have natural enumerations that can be used as labels for the partitioned regions. The mapping from a continuous space region to discrete binary sequence is a hash function. Furthermore, a hash functions with collision probability dependent on a distance function is called a Locality Sensitive Hash (LSH) function.

Definition 2.3 (Locality Sensitive Hash Function [25]): let $\mathbb{H} = \{h : S \rightarrow U\}$ is $(r_1, r_2, p_1, p_2)-$sensitive if for any $u, v \in S$

1) if $d(u,v) \leq r_1$ then $Pr_{\mathbb{H}}[h(u) = h(v)] \geq p_1$
2) if $d(u,v) > r_2$ then $Pr_{\mathbb{H}}[h(u) = h(v)] \leq p_2$

C. Leech Lattice Decoder

The Leech Lattice is a unique 24 dimensional lattice with many exceptional properties [26], [27]. Of particular interest to this work is the Leech Lattice's packing efficiency. The Leech Lattice defines an optimal regular sphere packing of 24 dimensional space [28] and will serve nicely as a space quantizer for *RPHash*. Furthermore, an efficient decoders for the Leech Lattice developed by Amrani and Be'ery's [29] has a worse-case decoding of only 519 floating point operations. Although higher dimensional lattices with comparable packing efficiency exist, in general the decoding complexity scales exponentially with dimension [30], [31]. An alternate decoders are also shown in the results section and in general originate from LSH and nearest neighbor algorithm research communities.

III. Related Work

Despite theoretical results showing that k-means has an exponential worse case complexity [32], many real world problems tend to fair much better under k-means and other similar algorithms. For this reason, clustering massive datasets with k-means, although suffering from unbounded complexity guarantees, often yields qualitatively good results near the global optimal solution. Due to approximate solution's real world proclivity toward revealing useful results, randomized methods such as sampling and random dimensional reduction are often utilized in overcoming complexity growth. The concept of random projection clustering is not new, having been explored in a variety papers involving high dimensional data clustering [8], [9], [33], [34].

The first set of clustering algorithms began with density based scanning methods. These methods tend to work well

on spatially separated datasets with relatively low dimension. A common clustering problem for these types of algorithms would be on geo-spatial data, in geographic data systems (GIS) and image segmentation. The algorithms DBScan [33], Clique [34], and CLARANS [9], respectively, represent a successful progression of the density scanning techniques. Although density scan algorithms are often scalable, they often show weaknesses in accuracy when scaling the number of dimensions.

Another important feature of our algorithm (*RPHash*) is the use of random projection clustering. A proof of the convergence of projected k-means clustering is given in Boutsidis [35]. Proclus [8] was an even earlier application that explored random projection for clustering. The merits of random projection are discussed in [36] who suggest that random projection not only compresses sparse datasets making them computationally more tractable but also may help overall accuracy by alleviated round-off issues caused by non-homoscedastic variance by generating more spherical clusters in a more dense subspace.

In addition to Proclus, various other methods and analysis have been proposed for clustering with random projections that provide bounds on the convergence and limits of random projection clustering. Florescu gives bounds on the scaling and convergence of projected clustering [37]. Their results closely follow the logic of Urruty [38], and find that the number of orthogonal projections required, is logarithmic in n, the number of vectors to be clustered. Following that the probability of a random projection plane offering a good partitioning increases exponentially as the number of dimensions in the projected subspace increases. Bingham *et al* provide examples of projected clustering well below the JL bound [20] and Bartal *et al* make these assertions mathematically rigorous showing that the projected subspace is independent of the data's original dimensionality [39].

Other clustering by random projection algorithms have been explored that are similar to *RPHash*, but for projections on the line. These so called cluster ensemble approaches [40]–[42] use histograms to identify cluster regions of high density much like *RPHash*. Although, as suggested in Florescu and Urruty, the single dimension approach may be plagued by issues of occultation, and exponential convergence as d increases. Figure 1 shows a brief example of projection occultation for Gaussian clusters in $\mathbb{R}^3$ space, projected to $\mathbb{R}^2$ space. In Figure 1 it is clear that even modest reductions of 3 dimensions to 2 dimensions can yield unwanted results.

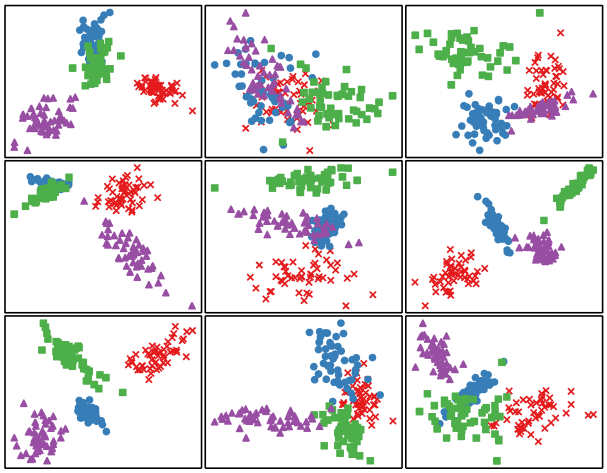


Fig. 1: Random Projection of Gaussian Clusters in $\mathbb{R}^3 \rightarrow \mathbb{R}^2$

IV. RPHash Algorithm

RPHash is a distributed algorithm for dense region identification as a precursor to a more computationally intensive clustering algorithms (k-Means, LDA, Mean Shift) or as a standalone approximate clustering algorithm. It is the latter case is that will be more thoroughly investigated in this discussion, with a brief discussion on *RPHash*'s potential as a density selection pre-conditioner. In the (*RPHash*) algorithm, both approximate and randomized techniques are employed to provide a stochastic element to our clustering algorithm. To combat the curse of dimensionality (COD), *RPHash* performs multi-probe, random projection with Gaussian blurring of high dimensional vectors to the unique partitions of the Leech Lattice (Λ_{24}) [43].

A. Overview of RPHash

An outline of the steps in the *RPHash* algorithm (Algorithm 1) is given below. Some of the aspects of its function in regards to randomness and approximation are highlighted here. One way that the *RPHash* algorithm achieves scalability is through the generative nature of its region assignment. Clustering region assignments are performed by decoding vector points into partitions of the Leech Lattice.

In most cases the problem space will not be exactly 24 dimensional. The Johnson-Lindenstrauss (*JL*) lemma and subsequently, random projection provides a solution to this problem and provides additional benefits II-A. *JL* states that for an arbitrary set of n points in m dimensional space, a projection exists onto a d-dimensional subspace such that all points are linearly separable with ϵ-distortion following $d \propto \Omega(\frac{log(n)}{\epsilon^2 log 1/\epsilon})$. Although many options for projections exists, a simple and sufficient method for high dimensions is to form the projection matrix $r_{ij} \in \mathbf{R}$ is $m \times d$ as follows:

$$r_{ij} = \begin{cases} +1, & \text{with probability } \frac{1}{6} \\ 0, & \text{with probability } \frac{2}{3} \\ -1, & \text{with probability } \frac{1}{6} \end{cases} \quad [17]$$

Although the Leech lattices partitions provide optimal sphere packing in 24 dimensions for regular lattices, (an unrelated version of the curse of dimensionality) the overall density of the lattice is sparse at 0.001930. To overcome this, RPHash "blurs" the projected data by apply shifts to projected vectors to more fully cover the $\mathbb{R}^{24}$ subspace and performing multiple probes of the Leech Lattice partitions in addition to the vector $v_{24} : v_{24} = \{v_{24}, v_{24} + N(0,1)_{24} ...\}$.

The approximation of a random projection is computationally efficient for large datasets, and unlike a truly Gaussian

projection matrix, yields a semi-positive definite projection matrix, that will likely be useful in proving the convergence of *RPHash*.

Data

K - number of clusters
$X = \{x_1, ..., x_n\}$, $x_k \in \mathbb{R}^m$ - data vectors
C- a k-HH counter
$\mathbb{H}$ - LSH Function
$\mathbb{P} = \{p_1, ...p_n\}$ - set of projection matrices
$L = \{\{\varnothing\}...\}$
$M = \{C, [0, ...0]\}$

Algorithm 1: RPHash Algorithm

```
forall x_k ∈ X do
    forall p_i ∈ P do
        x̃_k ← sqrt(m/d) p_i^T x_k
        t = H(x̃_k)
        L[k][i] = t
        C.add(t)
    end
end
forall x_k ∈ X do
    forall c_i ∈ C.top(K) do
        if L[k] ∩ M[i][0] ≠ 0 then
            Δ = M[k] − x_k
            M[k] = M[k] + Δ/count
            L[k].add(M[i][0])
        end
    end
end
Result: M
```

In addition to Achlioptas efficient random projection method for databases, a further reduction in the number of operation required for random subspace projection called the Fast Johnson Lindenstrauss Transform (FJLT) [44]–[46] is currently an active area of research. FJLT, and similar nearly optimal projection methods, utilize the local-global duality (Heisenberg Principle) of the Discrete Fourier Transform to precondition the projection matrix resulting in a nearly optimal number of steps to compute an ϵ-distortion random projection [44]–[46]. A sub-linear bound on the number of operations required for the dominant projection operation may further improve *RPHash*'s overall runtime complexity.

B. Overview of the RPHash algorithm

The basic intuition of *RPHash* is to combine multi-probe random projection with discrete space quantization. Following this intuition, near-neighbor vectors are often projected to the same partition of the space quantizer, which is regarded as a hash collision in LSH parlance. As an extension, multi-probe projection ensures that regions of high density in the original vector space are projected probabilistically more often to the same partitions that correspond to density modes in the original data. In other words, partitions with high collision rates are good candidates for cluster centroids. To follow common parameterized k-means methods, the top k densest regions will be selected.

According to the *JL* lemma, the sub-projections will conserve the pairwise distances in the projected space for points with ϵ-distortion in which the size of the dataset is proportional to the logarithm of the number of dimensions in the randomly projected subspace. In addition to compressing a dataset to a computationally more feasible subspace for performing space quantization, random projection can also make *eccentric* cluster more spherical [18], [36].

Discrete space quantizers play a central role in the *RPHash* algorithm. The sequential implementation of the *RPHash* algorithm will rely on the efficient Leech lattice decoder of Vardy, Sun, Be'ery, and Amrani [29], [47]–[49] used as a space quantizer. The lattice decoder implementation relies on the decomposition of the binary $(24, 22, 8)$ extended Golay Code into 4 partitions of the $(6, 3, 4)$ quaternary hexacode and its relation to the Leech Lattice as a type B lattice construction. This decomposition and relation to the Golay Code provides a real worse case decoding complexity well below the asymptotically exponential bound for trellis decoders as the dimension d increases [30], [50]. The Leech lattice is a unique lattice in 24 dimensions that is the densest lattice packing of hyperspheres in 24 dimensional space [27], [28]. The Leech lattice is a unique lattice in 24 dimensional space with many exceptional properties, however, of importance to *RPHash* is that it is the densest regular lattice possible in 24 dimensions and nearly optimal among theoretical non-regular packings [51]. The 24 dimensional subspace partitioned by the Leech Lattice is small enough to exhibit the spherical clustering benefit of random projection. Low distortion random embeddings are also feasible for very large dataset ($n = \Omega(c^{24})$) objects while avoiding the occultation problem [38]. Furthermore, the decoding of the Leech lattice is a well studied subject, with a constant worse case decoding complexity of 331 operations [47].

Space quantizers have hard margin boundaries and will only correctly decode points that are within the error correcting radius of its partitions. This is an an issue found in approximate nearest neighbor search [43], [52] and is overcome in a manner similar to Panigrahy [52] — by performing multiple random projections of a vector and then applying the appropriate locality sensitive to provide a set of hash IDs. Using multiple random projections of a vector allows the high dimensional vector to be represented as 'fuzzy' regions that are probabilistically dependent on the higher dimensional counterpart. From Panigrahy [52], the requirement of $(\Theta(log(n)))$ random projection probes is given to achieve c-approximate hash collisions for the bounded radius, r-near vectors. Random projection probing adds a $\Theta(log(n))$-complexity coefficient to the clustering algorithm. The top k cardinality set of lattice hash ID vector subsets represent regions of high density.

Projected clustering of representative cluster centroids will not in general be correlated with other projections of data into projected cluster centroids. To recover data from the projection

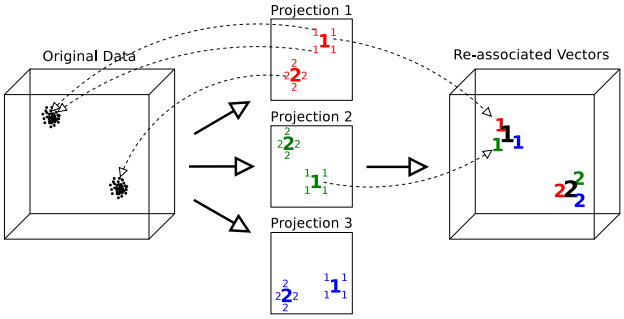


Fig. 2: Multiple projections $\mathbb{R}^3 \rightarrow \mathbb{R}^2 \rightarrow \mathbb{R}^3$

step, we must map projected vectors back to their original un-projected data space counterparts. The original data space vectors can then used to compute centroids corresponding to the clusters in the projected space. Figure 2 shows an example of this process for 3 projection probes of $\mathbb{R}^3 \rightarrow \mathbb{R}^2 \rightarrow \mathbb{R}^3$. standard Lloyd's algorithm is performed to resolve the overestimate of k the number of desired clusters, effectively merging the $k \times$ number of projections representations of centroids in the original data space.

V. Data Security: At no added cost

Recent United States government initiatives pushing for the large scale availability of data resources have made vast quantities of de-identified health information available to the public. These resources however have prompted advances in attacks on de-identification of whole genome sequence data. Such attacks have been used to associate, thought to be, anonymous medical records with specific individuals [53]. Similar de-anonymization attacks [54], [55] along with a presidential commission (privacy and progress in WGS) have prompted a need for better data security of medical records data. The *RPHash* algorithm provides an intrinsic solution to this problem in both the distribution of data among servers as well as during the communication steps required by the algorithm.

While attempting to mitigate communication restrictions, *RPHash* intrinsically provides some security in the data it exchanges. Previous attempts at securing data in distributed systems required additional cryptographic steps [56]. Namely, the randomly projected centroid IDs, and the aggregation of only the k-largest cardinality vector sets. Non-distributed data clustering requires the entire dataset to reside on the processing system and distributed methods often require communication of full data records between nodes.

In the subspace projection step of *RPHash*, nearly-orthogonal random projection is utilized as a destructive operation, providing vector anonymity. As a consequence of projecting the real data vectors to a random subspace via a near, but not completely orthogonal matrix, destructive data loss occurs providing a cryptographic "trapdoor" function. The data loss is an intrinsic part of the *RPHash* clustering algorithm that has little adverse effect on its model generation and subsequent recall accuracy. Given the likelihood that RPHash is applied to a dataset where the number of vectors n is much greater than the desired k centroids, recovering an individual's private information would require knowledge of $\frac{n}{k}$ (on average) records in the representative centroid.

A. Finding good random subspaces

For a fixed number of random projections the probability of finding a good projections converges exponentially to zero as the number of dimensions d increases [37]. In this work, the probability of finding 'good' (or low distortion) versus unfavorable (high distortion) subspace projections is improved by both random probing and the use of a high dimensional projection subspace (following the results of Urruty [38]).

B. Constellation Shaping

One issue encountered with the uniform distribution decoding lattice was a misappropriation of decoding symbols in the QAM lattice. Commonly in communication theory, a shaping code would be applied to data in order to redistribute symbols in order to minimize the signal energy needed to encode symbols in regard to the AWGN perturbed communication channel. In *RPHash* a similar circumstance is encountered resulting in data that doesn't fully utilize the QAM quantization step of the Leech decoder. To overcome this issue, we employ a quasi-inverse Gaussian transformation subject the data's sampled variance. To save redundant calculation, the inversion is applied to the QAM constellation points.

VI. Experimental Approach

To assure *RPHash*'s accuracy and performance, tests for similarity to the standard k-means algorithm, on various real and synthetic datasets. The experimental approach for testing *RPHash* will address several major areas of *RPHash*'s utility, namely: Synthetic Algorithm Accuracy, Real World Data Set Accuracy, and Overall Scalability.

A required analysis of any k-means algorithm is in its ability to correctly categorize unseen data. Therefore it is imperative that *RPHash* perform comparable to the standard k-means algorithm in regard to precision recall. In order to evaluate *RPHash* over a continuously changing dataset we first consider the clustering of synthetic Gaussian clusters. Using synthetic data we can arbitrarily scale the dataset variance, number of clusters, and dimensionality and measure overall clustering and timing performance.

In addition, we include a more thorough analysis of *RPHash* metrics in regard to various real world datasets, against a collection of standard clustering algorithms. To evaluate the real world performance of *RPHash*, we applied it to five different datasets and compared the results of *RPHash* to those produced by six other well established clustering algorithms, each of which is known to produce fairly accurate results on different types of data distributions. The R implementations used for the six other clustering algorithms are listed below:

- K-Means:It is implemented with 'kmeans' in R [57].
- Four methods of Agglomerative Hierarchical clustering: Single Linkage, Complete Linkage, Average Linkage

and Ward's minimum variance method. At each stage distances between clusters are recomputed by the Lance-Williams dissimilarity update formula according to the particular clustering method being used. Ward's (1963) clustering criterion, where the dissimilarities are squared before cluster updating, is used in the implementation of Ward's method (Murtagh and Legendre 2014). All of these agglomerative clustering methods are implemented with 'hclust' in R.

- Self-organizing Tree Algorithm (SOTA): It is implemented with 'sota' (Package 'clValid') in R [58].

For each of the five datasets, ground-truth partitions are known *a priori*. The ground-truth labels are compared with labels produced by each clustering algorithm to compute the values of three different external measures of clustering validation, viz., *Adjusted Rand Index*, *Accuracy* and the *Kappa Statistic*. We have run each of K-Means and *RPHash* 20 times on each dataset and have constructed 95% confidence intervals for the obtained results. The other clustering algorithms have been run only once as their outputs do not change from one run to another.

Along with external measures, we have captured average runtimes for each algorithm ('user time' in R) on a computer having Intel(R) Core(TM) i7-4770 CPU with 4 cores @ 3.40GHz supporting 8 hardware threads with 32 GB memory.

A. Datasets and Results

RPHash and Projected k-means to 24 dimension was performed on real and synthetic datasets. The limit to 24 dimensions is equal to the dimensionality of the Leech lattice decoder used in *RPHash* and was chosen for two reasons. First, k-means on very large datasets requires excessive amounts of time and space to complete. Second, the performance on non-projected data becomes nonequivalent, as the k-means algorithm would in effect be processing a different dataset. In order to mimic, multi-projection *RPHash*, k-means was given an equivalent multi-projection ratio and allowed to over-sample the k parametric k- clusters, similar to [59].

The synthetic datasets consists of k homoscedastic Gaussian clusters in $\mathbb{R}^d$ with centroids chosen from a uniform $[-1, 1]$ random distribution. For each cluster center an n/k set of vectors will be generated by an additive Gaussian noise permutation of variance (var) added to the centroid. This results in n vectors partitioned into k Gaussian clusters. From this half of the vectors will be reserved for training, and the other half for testing accuracy. The synthetic dataset compares precision recall performance and processing time on variable variance, number of clusters and number of vectors. With exception of the varying parameter, the dataset was generated from $\mathbb{R}^{5000}$, variance $= 1.0$, and number of vectors $= 100000$, and $k = 30$. k-means was tested against 2 variants of *RPHash*. The first variant(rep) selects a single representative vector from the set of vectors within a high density partition while the second(avg) averages all vectors within the partition.

Results on synthetic data in Figure 3 shows projected k-means (red $\times$), single projection *RPHash* (blue $\circ$), multiple projection *RPHash* with a single representative centroid (green $\square$), and multiple projection *RPHash* with mean representatives(purple Δ) performed on k clusters n Gaussian distributed vectors in d dimensions. Logistic regression (solid lines) with error bounds (shaded area) is given for all simulations.

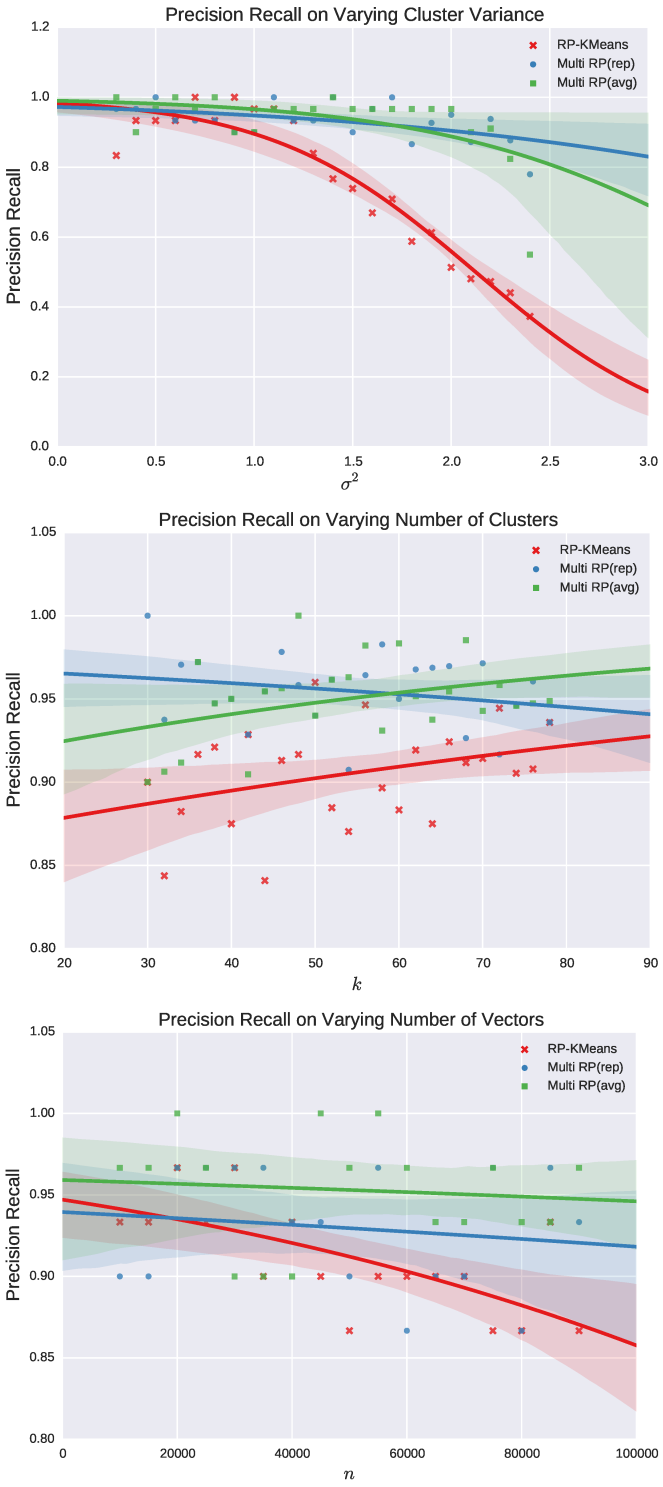
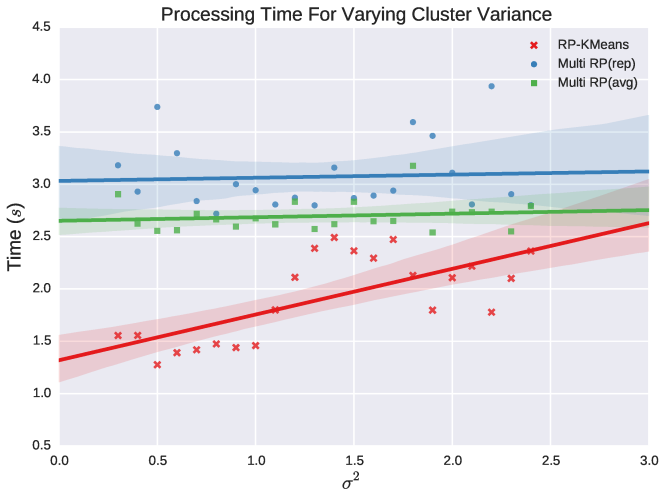


Fig. 3: Precision Recall Performance for Various Synthetic Data Sets



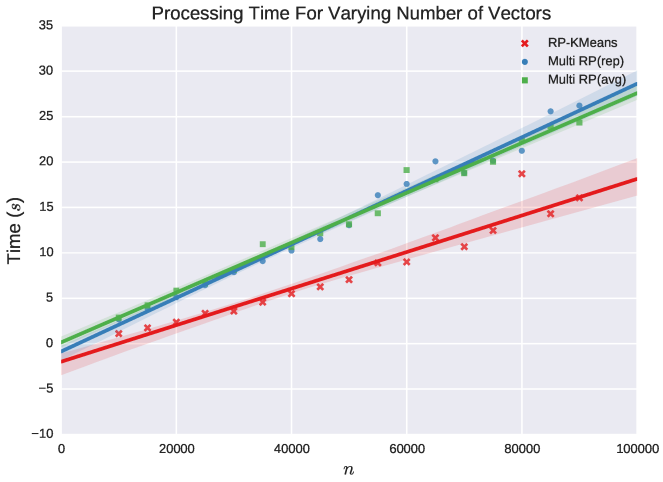


Fig. 4: Computation Time for Various Synthetic Data Sets

In Figure 4 an analysis of the time complexity for projected k-means (red $\times$), single projection *RPHash* (blue $\circ$), multiple projection *RPHash* with a single representative centroid (green $\square$), and multiple projection *RPHash* with mean representatives(purple Δ) on k clusters with n Gaussian distributed vectors in d dimensions. Logistic regression (solid lines) with error bounds (shaded area) is given for all simulations.

Dataset	Vectors	Attributes	Clusters
WebKB	256	1703	5
Cora	2708	1443	7
Internet Ad	3279	1558	2
CNAE-9	1080	856	9
CiteSeer	3312	3703	6

TABLE I: Real-World Datasets

Algorithm	Runtime	ARI	Accuracy	Kappa
k-means	6	1	6	3
Single	3	6	4	6
Complete	5	7	5	7
Average	3	6	4	6
Ward's	4	3	2	2
SOTA	7	2	1	1
RPHash	1	4	7	4

TABLE II: WebKB Wisconsin Dataset

For both *RPHash* variants, processing time is slightly longer, but the Precision Recall performance is better than k-means. The success of the *RPHash* algorithms in comparison suggests that it is within a constant factor processing time, and would surpass it for distributed architectures.

The real world datasets consist of 5 high dimensional datasets from various technical fields. Parameters for the datasets are given in Table I, and consist of WebKB (Wisconsin) Dataset [60], Cora Dataset [60], CiteSeer Dataset [60] and CNAE-9 Dataset [61] of natural language data. In addition, we tested our algorithm against the Internet Advertisements Datasets [62] containing mixed natural language and image data.

We applied k-means, Hierarchical Agglomerative (single, complete, average, Ward's), SOTA and *RPHash* to our real world clustering tasks. Results are given in Tables II, III, IV, V, and VI with runtime, ARI, Accuracy and Kappa statistic ranked for the clustering techniques. Gold entries represent top performance while blue represents worst performance for each dataset and metric.

Our results show that RPhash performs equivalently to the test clustering techniques with ranking results being mixed among the datasets and for all categories except Runtime. Our goal of RPHash was to create a fast distributed algorithm, which is evidenced in its dominance of this category. Ranking also reinforces the continued effectiveness of the standard k-means clustering algorithm, as both a fast and accurate algorithm. Furthermore SOTA is also shown to be a good alternative but a bit less stable than k-means.

Algorithm	Runtime	ARI	Accuracy	Kappa
k-means	3	1	2	4
Single	4	7	3	2
Complete	5	4	7	6
Average	6	7	6	3
Ward's	7	2	6	7
SOTA	2	3	6	5
RPHash	1	5	2	1

TABLE III: Cora Dataset

Algorithm	Runtime	ARI	Accuracy	Kappa
k-means	3	2	2	2
Single	4	6	7	6
Complete	5	6	7	6
Average	6	6	7	6
Ward's	7	1	4	7
SOTA	2	3	1	1
RPHash	1	7	3	3

TABLE IV: Internet Advertisement Dataset

Algorithm	Runtime	ARI	Accuracy	Kappa
k-means	2	2	2	2
Single	6	7	5	5
Complete	4	5	3	3
Average	3	7	5	5
Ward's	5	1	7	7
SOTA	7	4	1	1
RPHash	1	3	6	6

TABLE V: CNAE-9 Dataset

B. Security Performance

Due to the inability to anticipate all possible cryptographic attacks on deanonimization, a qualitative measure of data obfuscation is developed for comparing a fully qualified vector $v \in \mathbb{V}^d$ with its corresponding projected vectors $v' \in \mathbb{V}^d$. Where v' is the inverse projection of $u \in \mathbb{V}^s$ that results from the random projection of v. Destructive data obfuscation occurs if the distance between v' and v is greater than the distance between v and some other vector $\hat{v} \in \mathbb{V}^d$. The inverse projection matrix $R^{-1}_{d\to s}$ will be used to map u back to v's original subspace. The two equations below describe the projection of v to u and the theoretical inverse of the projection from u to v' under the matrix transform $R_{d\to s}$ where $d > s$.

$$u = \sqrt{\frac{n}{k}} R^T_{d\to s} v, v' = \sqrt{\frac{k}{n}} u^T R^{-1}_{d\to s}$$

The above inversion is theoretical however due to the orthogonal projection $R_{d\to s}$ being non-square and not invertible. Therefore, for a projection matrix $R_{d\to s}$ the least squares solution $\hat{R}^{-1}_{d\to s}$ will serves as the optimal inverse of the projection. Even in the not strictly orthogonal random projection case (as in Achlioptas [17] and *RPHash*) the least-squares solution will result in an over-determined system of equations. Which implies that any pseudo-inverse projection of a vector in V^s to V^d will result in unrecoverable data loss for non-trivial (*i.e.*, <**0**>,<**1**>) cases. The goal in testing the security of *RPHash* is to show that the data loss is sufficient to make it impossible for an attacker to re-associate the projected vectors. A formal definition of the requirement for destructive data obfuscation follows:

$$s(v, v') = ||v, v'||_2,$$

Algorithm	Runtime	ARI	Accuracy	Kappa
k-means	3	2	6	6
Single	4	5	4	5
Complete	6	7	2	2
Average	5	6	3	4
Ward's	7	3	1	1
SOTA	2	1	7	7
RPHash	1	4	5	3

TABLE VI: CiteSeer Dataset

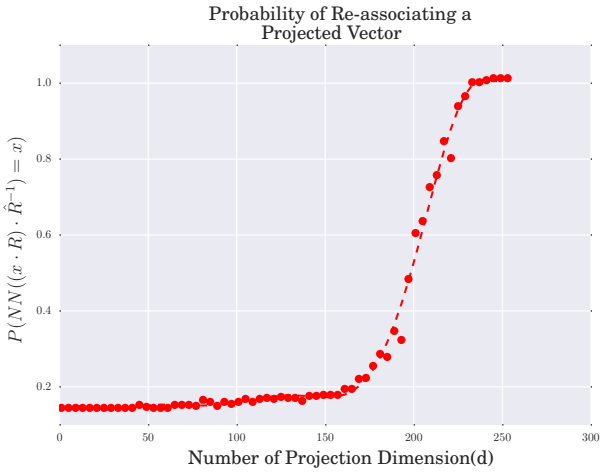


Fig. 5: Probability of Vector Re-association for MIMIC II BioMetric Signatures

$$\forall\{v, v'\} \in V, \exists \hat{v} \in V : s(v, v') > s(\hat{v}, v) \text{ where } \hat{v} \neq v \text{ or}$$

$$Pr(NN((v \cdot R) \cdot \hat{R}^{-1}, V) = v) \lesssim \frac{1}{||V||}$$

In Figure 5 a subset of the publicly available MIMIC II biometric data [63] was used to generate patient signatures. The top 64 most prevalent frequencies as identified via power spectral density estimation were take from the signals "RESP", "ABP", "ECG II" and "PLETH" of 38 patients were taken and composited into a 38×256 data matrix, on which clustering was performed. The overall bi-clustering in full and reduced dimensions showed little to no degradation over full and reduced subspaces down to 10 dimensions. This is further corroborated on additional data in Bingham [20]. Following the above definition of destructive data obfuscation, we constructed a test on the MIMIC II data. The Moore-Penrose inverse generated from the reciprocal of the singular value decomposition diagonal matrix is generated from the projection matrix and used as the least squares inverse projection to remap the projected vectors to their original subspace. The nearest neighbor method was then used to query the original set of vectors to see if a vector can be re-associated with itself. In the results it is clear that as dimensionality increases, the probability of re-association converges to random coincidence of a Bernoulli trial $((1 - (1/n))^n) \approx .363$. The successful re-associate rate follows an inverse power-law like distribution.

In regard to security performance, It is fairly clear that data becomes unrecoverable when the difference between the original data embedding and the projected space exceeds 75 dimensions. Given that RPHash's native clustering space is 24 dimensions, this case occurs for a wide variety of high dimensional datasets, namely those that exceed 99 dimensions.

VII. Conclusion

The *RPHash* algorithm combines approximate and randomized methods in a new way to solve issues of scalability and data security for cluster analysis on distributed data. The runtime

and Precision-Recall performance of *RPHash* is similar to that of the standard k-means clustering algorithm, with the added benefit of being scalable to very large datasets. This randomized, clustering algorithm is well suited for ill-posed, combinatorially restrictive problems such as clustering and partitioning. This assertion is complemented by a similar inversion regarding clustering and computing, in which real world problems tend to converge much faster than adversarially crafted worse-case problems.

The principle assumption in this work is that approximate and exact clustering, are qualitatively similar due to noise, redundancy, data loss, and the curse of dimensionality. Furthermore, the addition of random noise to the clustering dataset resulting from the random subspace projection requirement, provides some degree of non-deterministic process, so subsequent iterations of the algorithm could potentially find better results. Making the process of finding better clustering results, a matter of available processing time and resources. Furthermore, the destructive projection process affords us a certain degree of data privacy while requiring no additional processing.

Clustering has long been the standard method used for the analysis of labeled and unlabeled data. Clusterings effects intrinsically identify dissimilar and similar objects in a dataset, often unattainable through standard statistical methods. Single pass, data intensive statistical methods are often the primary workhorses for database processing of business logic and other domains, while clustering is often overlooked due to issue in its scalability.

Applications such as Micro Array clustering, Protein-Protein interaction clustering, medical resource decision making, medical image processing, and clustering of epidemiological events all serve to benefit from larger dataset sizes that *RPHash* enables.

VIII. Acknowledgments

Support for this work was provided in part by the National Science Foundation under grant ACI–1440420.

References

[1] M. R. Anderberg, *Cluster analysis for applications*. Academic Press, 1973.

[2] H. Samet, *Foundations of Multidimensional and Metric Data Structures*. Morgan Kaufmann, 2006.

[3] R. Xu and . I. I. Wunsch, D., "Survey of clustering algorithms," *Neural Networks, IEEE Transactions on*, vol. 16, no. 3, pp. 645–678, May 2005.

[4] J. Magidson and J. Vermunt, "Latent class models for clustering: A comparison with k-means," *Canadian Journal of Marketing Research*, vol. 20, no. 1, pp. 36–43, 2002.

[5] T. Hofmann, "Unsupervised learning by probabilistic latent semantic analysis," *Mach. Learn.*, vol. 42, no. 1-2, pp. 177–196, Jan 2001. [Online]. Available: http://dx.doi.org/10.1023/A:1007617005950

[6] U. V. Luxburg and S. Ben-david, "Towards a statistical theory of clustering," in *In PASCAL workshop on Statistics and Optimization of Clustering*, 2005.

[7] R. Zass and A. Shashua, "A unifying approach to hard and probabilistic clustering," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 1, Oct 2005, pp. 294–301.

[8] C. C. Aggarwal, J. L. Wolf, P. S. Yu, C. Procopiuc, and J. S. Park, "Fast algorithms for projected clustering," in *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '99. New York, NY, USA: ACM, 1999, pp. 61–72. [Online]. Available: http://doi.acm.org/10.1145/304182.304188

[9] R. T. Ng and J. Han, "Clarans: A method for clustering objects for spatial data mining," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 14, no. 5, pp. 1003–1016, 2002.

[10] I. Dhillon and D. Modha, "Lecture notes in computer science," in *Large-Scale Parallel Data Mining*, M. Zaki and C.-T. Ho, Eds. Springer Berlin Heidelberg, 2000, vol. 1759, ch. A Data-Clustering Algorithm on Distributed Memory Multiprocessors, pp. 245–260. [Online]. Available: http://dx.doi.org/10.1007/3-540-46502-2_13

[11] B.-H. Park and H. Kargupta, "Distributed data mining: Algorithms, systems, and applications," in *Data Mining Handbook*, N. Ye, Ed., 2002, pp. 341–358.

[12] T. Tamura, M. Oguchi, and M. Kitsuregawa, "Parallel database processing on a 100 node pc cluster: Cases for decision support query processing and data mining," *SC Conference*, vol. 0, p. 49, 1997.

[13] M. Mahajan, P. Nimbhorkar, and K. Varadarajan, "Lecture notes in computer science," in *WALCOM: Algorithms and Computation*, S. Das and R. Uehara, Eds. Springer Berlin Heidelberg, 2009, vol. 5431, ch. The Planar k-Means Problem is NP-Hard, pp. 274–285. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-00202-1_24

[14] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is "nearest neighbor" meaningful?" in *In Int. Conf. on Database Theory*, 1999, pp. 217–235.

[15] S. Dasgupta, *The hardness of k-means clustering*. Department of Computer Science and Engineering, University of California, San Diego, 2008.

[16] J. Bourgain, "On lipschitz embedding of finite metric spaces in hilbert space," *Israel Journal of Mathematics*, vol. 52, no. 1-2, pp. 46–52, 1985.

[17] D. Achlioptas, "Database-friendly random projections," in *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, 2001, pp. 274–281.

[18] S. S. Vempala, *The Random Projection Method*, ser. DIMACS Series. American Mathematical Society, 2004. [Online]. Available: http://books.google.com/books?id=L5L2J0UEY4QC

[19] P. Frankl and H. Maehara, "The johnson-lindenstrauss lemma and the sphericity of some graphs," *J. Comb. Theory Ser. A*, vol. 44, no. 3, pp. 355–362, jun 1987. [Online]. Available: http://dl.acm.org/citation.cfm?id=48184.48193

[20] E. Bingham and H. Mannila, "Random projection in dimensionality reduction: Applications to image and text data," in *in Knowledge Discovery and Data Mining*. ACM Press, 2001, pp. 245–250.

[21] R. Klein, "Lecture notes in computer science," in *Computational Geometry and its Applications*, H. Noltemeier, Ed. Springer Berlin Heidelberg, 1988, vol. 333, ch. Abstract voronoi diagrams and their applications, pp. 148–157. [Online]. Available: http://dx.doi.org/10.1007/3-540-50335-8_31

[22] S. Fortune, "A sweepline algorithm for voronoi diagrams," in *Proceedings of the Second Annual Symposium on Computational Geometry*, ser. SCG '86. New York, NY, USA: ACM, 1986, pp. 313–322. [Online]. Available: http://doi.acm.org/10.1145/10515.10549

[23] M. L. Gavrilova, "Lecture notes in computer science," in *Computational Science and Its Applications — ICCSA 2003*, V. Kumar, M. Gavrilova, C. Tan, and P. L'Ecuyer, Eds. Springer Berlin Heidelberg, 2003, vol. 2669, ch. An Explicit Solution for Computing the Euclidean d-dimensional Voronoi Diagram of Spheres in a Floating-Point Arithmetic, pp. 827–835. [Online]. Available: http://dx.doi.org/10.1007/3-540-44842-X_84

[24] P. V. Huffman W., *Fundamentals of Error-Correcting Codes*, 1st ed. Cambridge University Press, 2003.

[25] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proceedings of the twentieth annual symposium on Computational geometry*, ser. SCG '04. New York, NY, USA: ACM, 2004, pp. 253–262. [Online]. Available: http://doi.acm.org/10.1145/997817.997857

[26] R. T. Curtis, "A new combinatorial approach to m24," *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 79, no. 01, pp. 25–42, 1976. [Online]. Available: http://dx.doi.org/10.1017/S0305004100052075

[27] J. Conway and N. Sloane, *Sphere Packings Lattices and Groups Third Edition*. New York: Springer, 1998.

[28] J. Leech, "Notes on sphere packings," *Canadian Journal of Mathematics*, 1967.

[29] O. Amrani, Y. Be'ery, A. Vardy, F.-W. Sun, and H. C. A. van Tilborg, "The leech lattice and the golay code: bounded-distance decoding and

multilevel constructions," *Information Theory, IEEE Transactions on*, vol. 40, no. 4, pp. 1030–1043, Jul 1994.

[30] V. Tarokh and I. F. Blake, "Trellis complexity versus the coding gain of lattices. i," *Information Theory, IEEE Transactions on*, vol. 42, no. 6, pp. 1796–1807, Nov 1996.

[31] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger, "Closest point search in lattices," *Information Theory, IEEE Transactions on*, vol. 48, no. 8, pp. 2201–2214, Aug 2002.

[32] A. Vattani, "k-means requires exponentially many iterations even in the plane," in *Proceedings of the 25th annual symposium on Computational geometry*, ser. SCG '09. New York, NY, USA: ACM, 2009, pp. 324–332. [Online]. Available: http://doi.acm.org/10.1145/1542362.1542419

[33] 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*, vol. 96, 1996, pp. 226–231.

[34] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic subspace clustering of high dimensional data for data mining applications," in *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '98. New York, NY, USA: ACM, 1998, pp. 94–105. [Online]. Available: http://doi.acm.org/10.1145/276304.276314

[35] C. Boutsidis, A. Zouzias, and P. Drineas, "Random projections for k-means clustering," *CoRR*, vol. abs/1011.4632, 2010.

[36] S. Dasgupta, "Experiments with random projection," in *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, ser. UAI'00. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, pp. 143–151. [Online]. Available: http://dl.acm.org/citation.cfm?id=2073946.2073964

[37] I. Florescu, A. Molyboha, and A. Myasnikov, "Scaling and convergence of projection sampling," Tech. Rep., 2009.

[38] T. Urruty, C. Djeraba, and D. Simovici, "Clustering by random projections," in *Advances in Data Mining. Theoretical Aspects and Applications*, P. Perner, Ed. Springer Berlin Heidelberg, 2007, vol. 4597, ch. Lecture Notes in Computer Science, pp. 107–119. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-73435-2_9

[39] Y. Bartal, B. Recht, and L. J. Schulman, "Dimensionality reduction: beyond the johnson-lindenstrauss bound," in *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, 2011, pp. 868–887.

[40] X. Z. Fern and C. E. Brodley, "Random projection for high dimensional data clustering: A cluster ensemble approach," in *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003)*, aug 2003, pp. 186–193.

[41] M. Al-Razgan and C. Domeniconi, "Weighted clustering ensembles," in *Proceedings of the SIAM international conference on data mining*, 2006, pp. 258–269.

[42] R. Avogadri and G. Valentini, "Fuzzy ensemble clustering based on random projections for dna microarray data analysis," *Artificial Intelligence in Medicine*, vol. 45, no. 2, pp. 173–183, 2009.

[43] A. Andoni, "Nearest neighbor search: the old, the new, and the impossible," Ph.D. dissertation, Massachusetts Institute of Technology, September 2009.

[44] N. Ailon and B. Chazelle, "Approximate nearest neighbors and the fast johnson-lindenstrauss transform," in *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, 2006, pp. 557–563.

[45] A. Dasgupta, R. Kumar, and T. Sarlos, "A sparse johnson: Lindenstrauss transform," in *Proceedings of the Forty-second ACM Symposium on Theory of Computing*, ser. STOC '10. New York, NY, USA: ACM, 2010, pp. 341–350. [Online]. Available: http://doi.acm.org/10.1145/1806689.1806737

[46] N. Ailon and E. Liberty, "An almost optimal unrestricted fast johnson-lindenstrauss transform," *ACM Trans. Algorithms*, vol. 9, no. 3, pp. 21–1, Jun 2013. [Online]. Available: http://doi.acm.org/10.1145/2483699.2483701

[47] A. Vardy, "Even more efficient bounded-distance decoding of the hexacode, the golay code, and the leech lattice," *Information Theory, IEEE Transactions on*, vol. 41, no. 5, pp. 1495–1499, 1995.

[48] F.-W. Sun and H. C. A. van Tilborg, "The leech lattice, the octacode, and decoding algorithms," *Information Theory, IEEE Transactions on*, vol. 41, no. 4, pp. 1097–1106, Jul 1995.

[49] O. Amrani and Y. Beery, "Efficient bounded-distance decoding of the hexacode and associated decoders for the leech lattice and the golay code," *Communications, IEEE Transactions on*, vol. 44, no. 5, pp. 534–537, May 1996.

[50] V. Tarokh and I. F. Blake, "Trellis complexity versus the coding gain of lattices. i," *Information Theory, IEEE Transactions on*, vol. 42, no. 6, pp. 1796–1807, Nov 1996.

[51] H. Cohn and A. Kumar, "Optimality and uniqueness of the leech lattice among lattices," *Annals of Mathematics*, vol. 170, no. 3, pp. 1003–1050, 2009.

[52] R. Panigrahy, "Entropy based nearest neighbor search in high dimensions," in *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, ser. SODA '06. New York, NY, USA: ACM, 2006, pp. 1186–1195. [Online]. Available: http://doi.acm.org/10.1145/1109557.1109688

[53] M. Gymrek, A. L. McGuire, D. G. D, E. Halperin, and Y. Erlich, "Identifying personal genomes by surname inference," *Science*, no. 339, pp. 321–324, 2013.

[54] X. Ding, L. Zhang, Z. Wan, and M. Gu, "A brief survey on de-anonymization attacks in online social networks," in *Computational Aspects of Social Networks (CASoN), 2010 International Conference on*, 2010, pp. 611–615.

[55] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *Security and Privacy, 2008. SP 2008. IEEE Symposium on*, 2008, pp. 111–125.

[56] Y. Lindell and B. Pinkas, "Lecture notes in computer science," in *Advances in Cryptology –- CRYPTO 2000*, M. Bellare, Ed. Springer Berlin Heidelberg, 2000, vol. 1880, ch. Privacy Preserving Data Mining, pp. 36–54. [Online]. Available: http://dx.doi.org/10.1007/3-540-44598-6_3

[57] J. A. Hartigan and M. A. Wong, "A k-means clustering algorithm," *JSTOR: Applied Statistics*, vol. 28, no. 1, pp. 100–108, 1979.

[58] J. Herrero, A. Valencia, and J. Dopazo, "A hierarchical unsupervised growing neural network for clustering gene expression patterns," 2001.

[59] N. Ailon, R. Jaiswal, and C. Monteleoni, "Streaming k-means approximation," in *Advances in Neural Information Processing Systems 22*, Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, Eds. Curran Associates, Inc., 2009, pp. 10–18. [Online]. Available: http://papers.nips.cc/paper/3812-streaming-k-means-approximation.pdf

[60] P. Sen, G. M. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad, "Collective classification in network data," *AI Magazine*, vol. 29, no. 3, pp. 93–106, 2008.

[61] P. M. Ciarelli and E. Oliveira, "Agglomeration and elimination of terms for dimensionality reduction," in *Intelligent Systems Design and Applications, 2009. ISDA '09. Ninth International Conference on*, Nov 2009, pp. 547–552.

[62] N. Kushmerick, "Learning to remove internet advertisements," in *Proceedings of the Third Annual Conference on Autonomous Agents*, ser. AGENTS '99. New York, NY, USA: ACM, 1999, pp. 175–181. [Online]. Available: http://doi.acm.org/10.1145/301136.301186

[63] M. Saeed, M. Villarroel, A. T. Reisner, G. Clifford, L.-W. Lehman, G. Moody, T. Heldt, T. H. Kyaw, B. Moody, and R. G. Mark, "Multiparameter intelligent monitoring in intensive care ii (mimic-ii): A public-access intensive care unit database," *Critical Care Medicine*, vol. 39, pp. 952–960, May 2011. [Online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3124312/