Distributed and Parallel Databases, 11, 157–180, 2002

RACHET: An Efficient Cover-Based Merging of Clustering Hierarchies from Distributed Datasets*

NAGIZA F. SAMATOVA samatovan@ornl.gov
GEORGE OSTROUCHOV ost@ornl.gov
AL GEIST gst@ornl.gov
Computer Science and Mathematics Division, Oak Ridge National Laboratory,† *P.O. Box 2008, Oak Ridge, TN 37831, USA*

ANATOLI V. MELECHKO melechko@unix.cas.utk.edu
Molecular-Scale Engineering and Nanoscale Technologies Group, Oak Ridge National Laboratory, P.O. Box 2008, Oak Ridge, TN 37831, USA

Abstract. This paper presents a hierarchical clustering method named RACHET (Recursive Agglomeration of Clustering Hierarchies by Encircling Tactic) for analyzing multi-dimensional distributed data. A typical clustering algorithm requires bringing all the data in a centralized warehouse. This results in $O(nd)$ transmission cost, where n is the number of data points and d is the number of dimensions. For large datasets, this is prohibitively expensive. In contrast, RACHET runs with at most $O(n)$ time, space, and communication costs to build a global hierarchy of comparable clustering quality by merging locally generated clustering hierarchies. RACHET employs the encircling tactic in which the merges at each stage are chosen so as to minimize the volume of a covering hypersphere. For each cluster centroid, RACHET maintains descriptive statistics of constant complexity to enable these choices. RACHET's framework is applicable to a wide class of centroid-based hierarchical clustering algorithms, such as centroid, medoid, and Ward.

Keywords: clustering distributed datasets, distributed data mining

1. Introduction

Clustering of multidimensional data is a critical step in many fields including data mining [6], statistical data analysis [1, 12], pattern recognition and image processing [7], and business applications [2]. *Hierarchical clustering* based on a dissimilarity measure is perhaps the most common form of clustering. It is an iterative process of merging (agglomeration) or splitting (partition) of clusters that creates a tree structure called a *dendrogram* from a set of data points. *Centroid-based hierarchical clustering* algorithms, such as centroid, medoid, or minimum variance [1], define the dissimilarity metric between two clusters as some function (e.g., Lance-Williams [13]) of distances between cluster centers. Euclidean distance is typically used.

*This work has been supported by the MICS Division of the US Department of Energy.
†Oak Ridge National Laboratory is managed by UT-Battelle for the LLC U.S. D.O.E. under Contract No. DE-AC05-00OR22725.

We focus on the *distributed* hierarchical clustering problem. We create a hierarchical decomposition of clusters of massive data sets inherently distributed among various sites connected by a network. For practical reasons, the application to distributed and very massive (both in terms of data points and the number of features, or dimensions, for each point) datasets raises a number of major requirements for any solution to this problem:

1. *Qualitative comparability.* The quality of the hierarchical clustering system produced by the distributed approach should be comparable to the quality of the clustering hierarchy generated from centralized data.
2. *Computational complexity reduction*. Asymptotic time and space complexity of a distributed algorithm should be less than or equal to the asymptotic complexity of the corresponding centralized approach.
3. *Scalability*. The algorithms should be scalable with the number of data points, the number of features, and the number of data stores.
4. *Communication acceptability*. The data transfer/communication overheads should be modest. Doing this with minimal communication of data is a challenge.
5. *Flexibility*. If the solution is based on an existing clustering algorithm, then it should be applicable to a wide class of clustering algorithms.
6. *Visual representation sufficiency*. The summarized description of the resulting global hierarchical cluster structure should be sufficient for its accurate visual representation.

Current clustering approaches do not offer a solution to the distributed hierarchical clustering problem that meets all these requirements. Most clustering approaches [3, 9, 14] are restricted to the centralized data situation that requires bringing all the data together in a single, centralized warehouse. For large datasets, the transmission cost becomes prohibitive. If centralized, clustering massive centralized data is not feasible in practice using existing algorithms and hardware.

Distributed clustering approaches necessarily depend on how the data are distributed. Possible combinations are: *vertical* (features), *horizontal* (data points), and *block* fragmentations. For vertically distributed data sets, Johnson and Kargupta [10] proposed the Collective Hierarchical Clustering (CHC) algorithm for generating hierarchical clusters. The CHC runs with a $O(|S|n)$ space and $O(n)$communication requirement, where n is the number of data points and |S| is the number of data sites. Its time complexity for the agglomeration phase is $O(|S|n^2)$, and the implementation is restricted to single link clustering [1], also referred to as nearest neighbor clustering. This does not include the complexity for generating local hierarchies. Parallel based hierarchical clustering approaches [4, 15] can be considered as a special case of horizontal data distribution. However, these algorithms are tailored to a specific hardware architecture (e.g., PRAM) or restricted to a certain number of processors. Moreover, there is a major distinction between parallel and horizontally distributed approaches: the data are already distributed so that we do not have the luxury of distributing data for optimal algorithm performance as is often done for parallel computation.

We present a clustering algorithm named RACHET that is especially suitable for very large, high-dimensional, and horizontally distributed datasets. RACHET builds a global hierarchy by merging clustering hierarchies generated locally at each of the distributed data sites. Its time, space, and transmission costs are at most $O(n)$ (linear) in the size of the dataset. This includes only the complexity of the transmission and agglomeration phases and does not include the complexity of generating local clustering hierarchies. Finally, RACHET's summarized description of the global clustering hierarchy is sufficient for its accurate visual representation that maximally preserves the proximity between data points.

The rest of the paper is organized as follows. Section 2 describes the details of the RACHET algorithm. It first introduces the concept of descriptive statistics for cluster centroids and then derives an approximation to the Euclidean metric based on this concept. The description of the process of merging two clustering hierarchies concludes this section. Section 3 presents time/space/transmission cost analysis of the RACHET algorithm. Section 4 provides some empirical results on real and synthetic datasets. Error of the approximation to the Euclidean metric is discussed in Section 5. Finally, prospects for future work conclude this paper.

1.1. Definitions and notation

Here, we introduce notation and definitions used throughout the paper. Let n denote the total number of data points, d denote the dimensionality of the data space, and $|S|$ denote the number of data sites. First, we provide a formulation of the distributed hierarchical clustering problem.

Definition 1. The *dendrogram* [5] is a tree-like representation of the result of a hierarchical clustering algorithm. It can be viewed as a sequence of partitions of the data into clusters beginning with the whole data set at the root node of the tree at the top.

Problem Definition. The *distributed hierarchical clustering problem* is the problem of creating a global hierarchical decomposition into clusters (represented by a dendrogram) of a data set distributed across various data sites connected by a network. More formally,

Given:
1. n data objects with d features each
2. a distribution of these data objects across $S = \{S_1, S_2, \ldots, S_{|S|}\}$ data sites (a horizontal distribution), and
3. a set $D = \{D_1, D_2, \ldots, D_{|S|}\}$ of local hierarchical decompositions (or *local dendrograms*) of clusters of data objects in S_i, $i = 1, \ldots, |S|$

Find: A global hierarchical decomposition (or *global dendrogram*) of clusters of n data objects,
such that the global dendrogram generated from $|S|$ data sites is similar to the dendrogram generated from the centralized dataset of n data objects as much as possible.

For a horizontally distributed case, the ideal creation of a global dendrogram should fulfill the following requirements:

1. It should require minimum data transfer across the network: $O(n)$ or $O(n \log n)$ but not $O(nd)$ or higher, because the communication cost will be prohibitive for high-dimensional datasets.

2. It should be fast to merge local dendrograms: $O(n)$ or $O(n \log n)$ but not $O(n^2)$ or higher, because time and space cost will be too high for massive datasets.
3. It should be of a comparable quality relative to the centralized dendrogram.

Next, we will define the Descriptive Statistics, or summarized cluster representation. This is one of the key concepts of this paper. Let $C = \{\vec{p}_1, \vec{p}_2, \dots, \vec{p}_{N_c}\} \subset R^d$ be the set of data points in a cluster.

Definition 2. The *cluster centroid* $\vec{c} = (f_{c1}, f_{c2}, \dots, f_{cd})$ is the mean vector of all the data points in the cluster. Hence, the centroid of cluster C is defined as:

$$\vec{c} = \frac{1}{N_c} \sum_{i=1}^{i=N_c} \vec{p}_i \tag{1}$$

Let p_{ij} denote the j-th component of the data point $\vec{p}_i$. Thus, the j-th component of $\vec{c}$ is given by:

$$f_{cj} = \frac{1}{N_c} \sum_{i=1}^{N_c} p_{ij}, \quad j = \overline{1,d}. \tag{2}$$

Definition 3. The *radius of the cluster* R_c is defined as the average squared Euclidean distance of a point from the centroid of the cluster. More formally, R_c is given by

$$R_c := \left[\frac{\sum_{i=1}^{N_c} (\vec{p}_i - \vec{c})^2}{N_c} \right]^{\frac{1}{2}} \tag{3}$$

Definition 4. The *covering hypersphere* $(\vec{c}, R_c)$ of the cluster C is defined as the hypersphere with the center $\vec{c}$ and the radius R_c. Each cluster C can be represented by a covering hypersphere $(\vec{c}, R_c)$. In what follows, the terms "cluster" and its "hypersphere" will be used interchangeably throughout the paper.

Selection and effective description of cluster Descriptive Statistics (DS), or summarized cluster representation, is an important step in merging local clustering hierarchies and in visualization of the global hierarchy. DS have to meet a number of major requirements:

- They should occupy much less space than the naive representation, which maintains all objects in a cluster.
- They should be adequate for efficiently calculating all measurements involved in making clustering decisions such as merging or reconfiguration.
- They should be sufficient to visually represent the global hierarchy.

Definition 5. The *Descriptive Statistics* (*DS*) of the cluster centroid $\vec{c}$ are defined as a 6-tuple $DS(\vec{c}) = (N_c, NORMSQ_c, R_c, SUM_c, MIN_c, MAX_c)$, where

1. N_c is the number of data points in the cluster.
2. $NORMSQ_c$ is the square norm of the centroid defined as:

$$NORMSQ_c := \sum_{j=1}^{j=d} f_{cj}^2 \tag{4}$$

3. R_c is the radius of the cluster
4. SUM_c is the sum of the components of the centroid defined as:

$$SUM_c := N_c \sum_{j=1}^{j=d} f_{cj} \tag{5}$$

5. MIN_c is the minimum value of the centroid components defined as:

$$MIN_c := N_c \min_{1 \le j \le d} f_{cj} \tag{6}$$

6. MAX_c is the maximum value of the centroid components defined as:

$$MAX_c := N_c \max_{1 \le j \le d} f_{cj} \tag{7}$$

Finally, we define some other notations used for building a global dendrogram:

1. $d^2(\vec{c}_1, \vec{c}_2)$ is the squared Euclidean distance between two cluster centroids, $\vec{c}_1$ and $\vec{c}_2$. It is given by:

$$d^2(\vec{c}_1, \vec{c}_2) = \sum_{j=1}^{d} \left(f_{c_1 j} - f_{c_2 j}\right)^2. \tag{8}$$

2. $d^2_{\text{approx}}(\vec{c}_1, \vec{c}_2)$ is the approximation to the squared Euclidean distance. It is defined by Eq. (19). $d_{\text{approx}}(\vec{c}_1, \vec{c}_2)$ denotes the square root of $d^2_{\text{approx}}(\vec{c}_1, \vec{c}_2)$.
3. $NN(i)$ is the nearest neighbor of the ith object.
4. $DISS(i)$ is the value of dissimilarity (e.g., Euclidean distance) between the ith object and its nearest neighbor.

2. The RACHET algorithm

We assume the data are distributed across several sites where each site has the same set of features but on different items. Note that this is a horizontal distribution of the data. Homogeneity is assumed not only for the type of features of the problem domain but also for the units of measurements of those features. Next, we use Euclidean distance as the measure of dissimilarity between individual points. Finally, the implementation of RACHET assumes a centroid-based hierarchical clustering algorithm, such as centroid, medoid, or

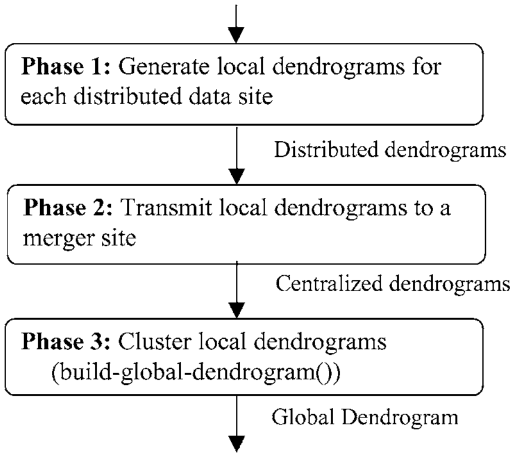

Figure 1. Control flow of RACHET.

minimum variance (Ward's). An overview of these hierarchical clustering methods can be found in [1].

Figure 1 presents the control flow of RACHET. Phase 1 is designed to generate a local dendrogram from each of the distributed data sites using a given off-the-shelf centroid-based hierarchical clustering algorithm. For each node of the dendrogram, RACHET maintains Descriptive Statistics (DS) about the cluster at that node. The space complexity of the DS is constant. This summary is not only efficient because it requires much less space than storing all the data points in the cluster, but also effective because it is sufficient for calculating all measurements involved in making clustering decisions in consecutive phases. More details on DS are presented in Section 2.1. We adapted Ward's agglomerative hierarchical clustering algorithm to generate and maintain the DS.

After Phase 1, we obtain a list of dendrograms that captures the major information about each cluster centroid needed for making clustering decisions in Phase 3. In Phase 2, local dendrograms are transmitted to a single *merger* site. The agglomeration of these dendrograms is performed at the merger site.

Phase 3 is the core of RACHET. The main task of Phase 3 is to cluster local dendrograms into a global dendrogram. We adapted an agglomerative hierarchical algorithm for clustering data points [15] by applying it directly to local dendrograms. The algorithm is shown in figure 2. One of the key components in this algorithm is the call of the merge-dendrograms( ) method that merges two dendrograms. The details of this method are discussed in Section 2.3. Due to the lack of space, we omit description of the find_best_match( ) method. Its pseudo code is available at http://www.csm.ornl.gov/~ost/RACHET.

2.1. Centroid descriptive statistics

Given the descriptive statistics (see Definition 5) of two cluster centroids $\vec{c}_1$ and $\vec{c}_2$, this section provides a mechanism for updating the descriptive statistics of cluster $\vec{c}$ formed by merging clusters $\vec{c}_1$ and $\vec{c}_2$ without regenerating them from "scratch".

Dendrogram build-global-dendrogram(Dendrogram $D[\,]$)
{
For each $\{i, j : 0 \le i < j \le |S|\}$ compute $d_{approx}(\vec{c}_i, \vec{c}_j)$
For each $\{i : 0 \le i \le |S|\}$ compute
    $NN(i)$ = **find-best-match**($D[\,i]$, $D[\,]$)
    $DISS(i) = d_{approx}(\vec{c}_i, \vec{c}_{idexOf(NN(i))})$
Initialize *GlobalDendrogram*
Repeat $|S| - 1$ times
    Determine i such that *DISS(i)* is minimized
    $Dendrogram1 \leftarrow D[i]$
    $Dendrogram2 \leftarrow NN(i)$
    Dendrogram3 = **merge-dendrograms**(*Dendrogram1*, *Dendrogram2*)
    Update *GlobalDendrogram*, each $DISS(i)$ and $NN(i)$ as necessary
return *GlobalDendrogram*
}

Figure 2. An efficient algorithm to build a global dendrogram.

Theorem 2.1. *Assume that $DS(\vec{c}_1) = (N_{c_1}, NORMSQ_{c_1}, R_{c_1}, SUM_{c_1}, MIN_{c_1}, MAX_{c_1})$ and $DS(\vec{c}_2) = (N_{c_2}, NORMSQ_{c_2}, R_{c_2}, SUM_{c_2}, MIN_{c_2}, MAX_{c_2})$ are the descriptive statistics of two disjoint clusters $\vec{c}_1$ and $\vec{c}_2$, respectively. Then the following statements hold for the descriptive statistics of cluster $\vec{c}$ that is formed by merging these clusters*:

1. $N_c = N_{c_1} + N_{c_2}$.
2. $NORMSQ_c = \frac{1}{N_{c_1}+N_{c_2}}\{N_{c_1}NORMSQ_{c_1} + N_{c_2}NORMSQ_{c_2} - \frac{N_{c_1}N_{c_2}}{N_{c_1}+N_{c_2}}d^2(\vec{c}_1, \vec{c}_2)\}$, *where $d^2(\vec{c}_1, \vec{c}_2)$ is the squared Euclidean distance between the two centroids.*
3. $R_c = [\frac{1}{N_{c_1}+N_{c_2}}\{N_{c_1}R_{c_1}^2 + N_{c_2}R_{c_2}^2 + \frac{N_{c_1}N_{c_2}}{N_{c_1}+N_{c_2}}d^2(\vec{c}_1, \vec{c}_2)\}]^{\frac{1}{2}}$
4. $SUM_c = SUM_{c_1} + SUM_{c_2}$
5. $MIN_c \ge MIN_{c_1} + MIN_{c_2}$
6. $MAX_c \le MAX_{c_1} + MAX_{c_2}$

Proof: In order to evaluate the square norm of centroid $\vec{c}$ that is formed by merging disjoint clusters $\vec{c}_1$ and $\vec{c}_2$, we first note that based on Eq. (2) the j-th component of $\vec{c}$ can be defined by the relation

$$N_c f_{cj} = N_{c_1} f_{c_1 j} + N_{c_2} f_{c_2 j}$$

Squaring both sides of this equation gives

$$N_c^2 f_{cj}^2 = N_{c_1}^2 f_{c_1 j}^2 + N_{c_2}^2 f_{c_2 j}^2 + 2N_{c_1}N_{c_2} f_{c_1 j} f_{c_2 j} \tag{9}$$

The cross-product term can be written as

$$2 f_{c_1 j} f_{c_2 j} = f_{c_1 j}^2 + f_{c_2 j}^2 - \left(f_{c_1 j} - f_{c_2 j}\right)^2 \tag{10}$$

Substituting Eq. (10) into Eq. (9) and dividing both sides by N_c^2 then gives

$$f_{cj}^2 = \frac{1}{N_{c_1} + N_{c_2}} \left\{ N_{c_1} f_{c_1 j}^2 + N_{c_2} f_{c_2 j}^2 - \frac{N_{c_1} N_{c_2}}{N_{c_1} + N_{c_2}} \left(f_{c_1 j} - f_{c_2 j} \right)^2 \right\}$$

Summing both sides of this equation over j results in

$$\sum_{j=1}^{d} f_{cj}^2 = \frac{1}{N_{c_1} + N_{c_2}} \left\{ N_{c_1} \sum_{j=1}^{d} f_{c_1 j}^2 + N_{c_2} \sum_{j=1}^{d} f_{c_2 j}^2 - \frac{N_{c_1} N_{c_2}}{N_{c_1} + N_{c_2}} \sum_{j=1}^{d} \left(f_{c_1 j} - f_{c_2 j} \right)^2 \right\} \tag{11}$$

This proves the update formula for the $NORMSQ_c$.

From the definition of cluster centroid $\vec{c}$, it follows that its squared radius can be written as:

$$N_c R_c^2 = \sum_{i=1}^{N_c} \sum_{j=1}^{d} (p_{ij} - f_{cj})^2 = \sum_{j=1}^{d} \sum_{i=1}^{N_c} p_{ij}^2 - N_c \sum_{j=1}^{d} f_{cj}^2 \tag{12}$$

If cluster $\vec{c}$ is formed by merging two disjoint clusters $\vec{c}_1$ and $\vec{c}_2$, then the first term in Eq. (12) can be decomposed into the sum of squared coordinates of data points in the first cluster and the sum of squared coordinates of points in the second cluster. That is,

$$\sum_{j=1}^{d} \sum_{i=1}^{N_c} p_{ij}^2 = \sum_{j=1}^{d} \sum_{i=1}^{N_{c_1}} p_{ij}^2 + \sum_{j=1}^{d} \sum_{i=1}^{N_{c_2}} p_{ij}^2 \tag{13}$$

Substituting Eqs. (11) and (13) into Eq. (12) and regrouping the terms then gives

$$\begin{aligned} N_c R_c^2 = {} & \left(\sum_{j=1}^{d} \sum_{i=1}^{N_{c_1}} p_{ij}^2 - N_{c_1} \sum_{j=1}^{d} f_{c_1 j}^2 \right) + \left(\sum_{j=1}^{d} \sum_{i=1}^{N_{c_2}} p_{ij}^2 - N_{c_2} \sum_{j=1}^{d} f_{c_2 j}^2 \right) \\ & + \frac{N_{c_1} N_{c_2}}{N_{c_1} + N_{c_2}} \sum_{j=1}^{d} \left(f_{c_1 j} - f_{c_2 j} \right)^2 \end{aligned}$$

Applying Eq. (12) to clusters $\vec{c}_1$ and $\vec{c}_2$, the last equation can be written as

$$N_c R_c^2 = N_{c_1} R_{c_1}^2 + N_{c_2} R_{c_1}^2 + \frac{N_{c_1} N_{c_2}}{N_{c_1} + N_{c_2}} d^2(\vec{c}_1, \vec{c}_2)$$

This proves the update formula for the R_c.

To derive the lower bound on MIN_c of the centroid $\vec{c}$, we note that each component j of $\vec{c}$ can be represented as

$$N_c f_{cj} = \sum_{i=1}^{N_c} p_{ij} = \sum_{i=1}^{N_{c_1}} p_{ij} + \sum_{i=1}^{N_{c_2}} p_{ij} = N_{c_1} f_{c_1 j} + N_{c_2} f_{c_2 j} \tag{14}$$

By definition of MIN_{c_1} and MIN_{c_2}, it follows that

$$f_{c_1 j} \geq \frac{1}{N_{c_1}} MIN_{c_1} \quad \text{and} \quad f_{c_2 j} \geq \frac{1}{N_{c_2}} MIN_{c_2} \quad \text{for} \;\; j = 1, \ldots, d.$$

Hence, Eq. (14) can be estimated as

$$N_c f_{cj} \geq MIN_{c_1} + MIN_{c_2}$$

Taking the minimum from both sides of this inequality over j proves the lower bound for the MIN_c. The update formulas for the other parameters of $DS(\vec{c})$ can be proven similarly. □

2.2. *Euclidean distance approximation*

From Eq. (8), it follows that in order to compute the Euclidean distance between centroids from different local datasets would require the transmission of all d centroid components. This approach would involve the transmission of cluster centroids represented by each node of the dendrogram generated at each of the $|S|$ local datasets. This would result in a transmission cost of $O(nd)$, which can be prohibitively high.

Given the DS of each cluster, we can derive an approximated distance between the two cluster centroids. Equation (8) can be expanded as follows:

$$d^2(\vec{c}_1, \vec{c}_2) = \sum_{j=1}^{d} f_{c_1 j}^2 + \sum_{j=1}^{d} f_{c_2 j}^2 - 2 \sum_{j=1}^{d} f_{c_1 j} f_{c_2 j} \tag{15}$$

$$d^2(\vec{c}_1, \vec{c}_2) = NORMSQ_{c_1} + NORMSQ_{c_2} - 2 \sum_{j=1}^{d} f_{c_1 j} f_{c_2 j} \tag{16}$$

If the cross-product term is ignored, then the distance can be approximated by the sum of square norms of the centroids. This results in a significant error. To reduce this error, we can place a non-zero upper and lower bound on the cross-product term:

$$\frac{1}{N_{c_1} N_{c_2}} MIN_{c_1} SUM_{c_2} \leq \sum_{j=1}^{d} f_{c_1 j} f_{c_2 j} \leq \frac{1}{N_{c_1} N_{c_2}} MAX_{c_1} SUM_{c_2} \tag{17}$$

or

$$\frac{1}{N_{c_1} N_{c_2}} MIN_{c_2} SUM_{c_1} \leq \sum_{j=1}^{d} f_{c_1 j} f_{c_2 j} \leq \frac{1}{N_{c_1} N_{c_2}} MAX_{c_2} SUM_{c_1} \tag{18}$$

Inequalities (17) and (18) hold, if each component of the cluster centroid is positive, i.e. $f_{cj} > 0$, $\forall c$ and $j = 1, \ldots, d$. Otherwise, for $O(|S|)$ communication cost, we can broadcast the global constant *CONST* such that

$$p_{ij}^{\text{new}} = p_{ij}^{\text{old}} + CONST > 0$$

for each component j of the data point $\vec{p}_i$. Taking the maximum of the lower bounds and the minimum of the upper bounds in (17) and (18) leads to the following bounds on the Euclidean distance:

$$\begin{aligned} d_{\text{lower}}^2(\vec{c}_1, \vec{c}_2) &= \max\Bigg\{0, NORMSQ_{c_1} + NORMSQ_{c_2} \\ &\quad - 2\frac{1}{N_{c_1}N_{c_2}} \min\{MAX_{c_1}SUM_{c_2}, MAX_{c_2}SUM_{c_1}\}\Bigg\} \\ d_{\text{upper}}^2(\vec{c}_1, \vec{c}_2) &= NORMSQ_{c_1} + NORMSQ_{c_2} \\ &\quad - 2\frac{1}{N_{c_1}N_{c_2}} \max\{MIN_{c_1}SUM_{c_2}, MIN_{c_2}SUM_{c_1}\} \end{aligned}$$

Taking the simple mean of the minimum and the maximum square distances gives an approximation of the squared Euclidean distance between two centroids

$$d_{\text{approx}}^2(\vec{c}_1, \vec{c}_2) = \frac{d_{\text{lower}}^2 + d_{\text{upper}}^2}{2} \tag{19}$$

2.3. *Merging two dendrograms*

Given two datasets S_1 and S_2 and their dendrograms D_1 and D_2 generated by a hierarchical clustering algorithm applied locally to each data set, figure 3 illustrates four different cases (out of six possible) of merging the two dendrograms (figure 3(a)) into dendrogram D_{new}.

Case 1 (figure 3(b)). This case is designed to merge two well separated datasets. Two clusters, $\vec{c}_1$ and $\vec{c}_2$, are well separated if their hyperspheres do not intersect. That is,

$$d(\vec{c}_1, \vec{c}_2) \geq R_{c_1} + R_{c_2}.$$

In this case, a new parent node, D_{new}, is created and dendrograms D_1 and D_2 become the children of the new node. The descriptive statistics $DS(\vec{c}_{\text{new}})$ of the new cluster are updated according to Theorem 2.1.

Case 2. Here, the data points of the first cluster are contained in the hypersphere with center $\vec{c}_2$ and radius R_{c_2}, i.e.

$$d(\vec{c}_1, \vec{c}_2) < R_{c_2}.$$

This case is further subdivided into two subcases:

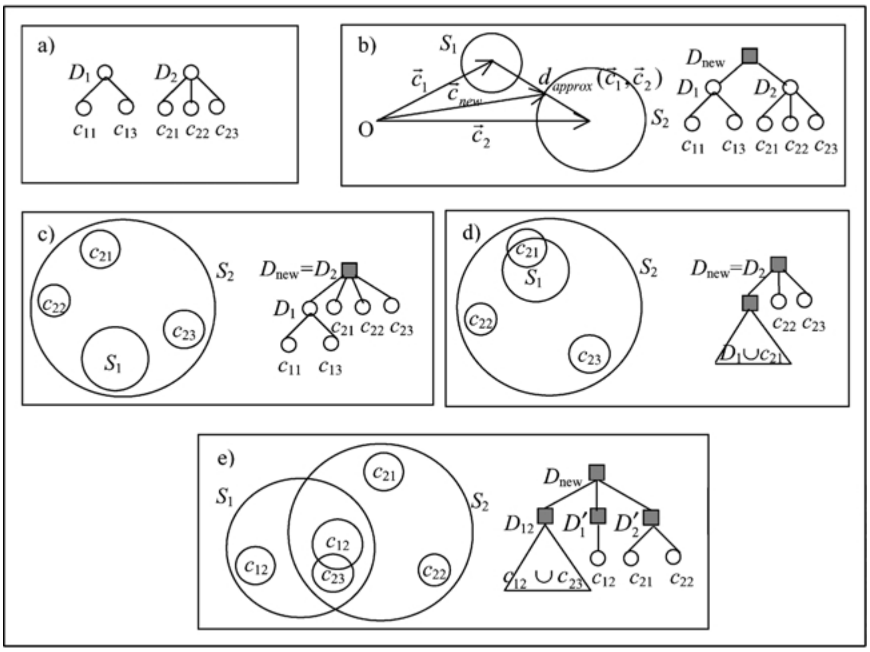

Figure 3. Four cases of merging two dendrograms. (a) Two dendrograms D_1 and D_2. (b) Merging two well separated clusters (Case 1). (c) Making cluster S_1 a subcluster of cluster S_2 provided a proper containment of cluster S_1 in cluster S_2 (Case 2a). (d) Merging cluster S_1 with the best matched subcluster of cluster S_2 provided a proper containment of cluster S_1 in cluster S_2 (Case 2b). (e) Merging two overlapping clusters (Case 4).

Case 2a (figure 3(c)). The first cluster $(\vec{c}_1, R_{c_1})$ is well separated from any other child cluster $(\vec{c}_{2j}, R_{c_{2j}})$ of the second cluster $(\vec{c}_2, R_{c_2})$, $j = 1, 2, \ldots$. In this case, dendrogram D_1 becomes a new child of dendrogram D_2. The descriptive statistics $DS(\vec{c}_{\text{new}})$ are updated similarly to Case 1.

Case 2b (figure 3(d)). The first cluster $(\vec{c}_1, R_{c_1})$ overlaps with one or more child clusters $(\vec{c}_{2j}, R_{c_{2j}})$ of the second cluster $(\vec{c}_2, R_{c_2})$, $j = 1, 2, \ldots$. Here the child cluster that matches best with dendrogram D_1 is selected to be merged with this dendrogram using a recursive call to the merge_dendrograms( ) process. There are a number of possible choices for defining a "best match". One choice for the best match is the cluster that has the largest intersection volume with the candidate cluster. The new node that is returned by the merge_dendrograms( ) process replaces the selected child in dendrogram D_2. If the new node D_{new} has more than two children, then its descriptive statistics are obtained by repeatedly applying Theorem 2.1 to two children at a time.

Case 3. This case addresses the situation when data points of the second cluster are contained in the hypersphere with center $\vec{c}_1$ and radius R_{c_1}, i.e.

$$d(\vec{c}_1, \vec{c}_2) < R_{c_1}.$$

This case is a degenerate example of Case 2.

Case 4 (figure 3(e)). This last case is designed to merge partially overlapped clusters, i.e.

$$\left(d(\vec{c}_1, \vec{c}_2) < R_{c_1} + R_{c_2}\right) \quad \text{and} \quad \left(d(\vec{c}_1, \vec{c}_2) > R_{c_1} \text{or } d(\vec{c}_1, \vec{c}_2) > R_{c_2}\right).$$

This case tries to improve the quality of the clustering by reconfiguring the children of both dendrograms D_1 and D_2. First, a new parent node, D_{new}, with D_1 and D_2 as its children is created and its DS are updated like in case 1. Second, the children of both dendrograms are partitioned into three categories:

1. The D'_1 category that contains all the children of D_1 that do not overlap with $(\vec{c}_2, R_{c_2})$
2. The D'_2 category that contains all the children of D_2 that do not overlap with $(\vec{c}_1, R_{c_1})$
3. The D_{12} category that includes the children that overlap with both $(\vec{c}_1, R_{c_1})$ and $(\vec{c}_2, R_{c_2})$.

Next, all children that are not in the D'_1 category are removed from D_1. The DS about D_1 are updated to reflect these changes. If the modified node D_1 has more than one child, then its descriptive statistics are obtained by repeatedly applying Theorem 2.1 to two children at a time. Otherwise, the D_1 node is replaced by its only child. Similar steps are done for D_2. Finally, the build-global-dendrogram( ) process is called using the children in the D_{12} category. The node that is returned by this method becomes a new child of D_{new}.

Figure 4 describes an algorithm that merges two clustering hierarchies.
Note that definitions of several methods in figure 4 such as create-parent( ), add-child( ), find_best_match( ), delete-children( ), update-DS( ) are omitted in the paper due to the lack of space. The pseudo codes of these methods can be found at http://www.csm.ornl.gov/~ost/RACHET.

3. Complexity analysis

This section presents complexity analysis for Phase 2 and Phase 3 (figure 1) of the RACHET algorithm. The overall cost of transmitting the local dendrograms to the merger site (Phase 2) is given by:

$$\mathit{Transmission}_{\text{total}} = \sum_{i=1}^{|S|} \mathit{Transmission}(i)$$

where $\mathit{Transmission}(i)$ is the cost of transmission of a given local dendrogram i. Given the nature of the dendrogram, there is a total of $2n_i - 1$ nodes in the dendrogram with n_i leaf nodes. We use an array-based format for the dendrogram representation as described in [10]. In this format, there are $2n_i - 1$ elements in the array. Each element in the array contains at most 4 items to represent each node with additional 6 items to represent the descriptive statistics about the cluster centroid associated with each node. Thus, the cost of transmission of a given local dendrogram i can be written as:

$$\mathit{Transmission}(i) = O((4 + 6) \times (2n_i - 1)) = O(n_i)$$

```
Dendrogram    merge-dendrograms(Dendrogram D1, Dendrogram D2)
{
  Case = which-case(D1, D2)
  if (Case == 1)
    ParentDendrogram = create-parent(D1, D2)
  else if (Case == 2.a)
    ParentDendrogram = D2
    add-child(ParentDendrogam, D1)
  else if (Case == 2.b)
    ParentDendrogram = D2
    BestChild = find-best-match(D1, D2->children)
    NewChild = merge-dendrograms(D1, BestChild)
    add-child(ParentDendrogam, NewChild)
  else if (Case == 3.a)
    ParentDendrogram = D1
    add-child(ParentDendrogam, D2)
  else if (Case == 3.b)
    ParentDendrogram = D1
    BestChild = find-best-match(D2, D1->children)
    NewChild = merge-dendrograms(D2, BestChild)
    add-child(ParentDendrogam, NewChild)
  else        /* Case 4 */
    ParentDendrogram = create-parent(D1, D2)
    Children[] = find-overlaps(D1, D2)
    delete-children(D1, Children[])
    update-DS(D1)
    delete-children(D2, Children[])
    update-DS(D2)
    NewChild = build-global-dendrogram(Children[])
    add-child(ParentDendrogam, NewChild)
  update-DS(ParentDendrogram)
  return ParentDendrogram
}
```

Figure 4. An algorithm to efficiently merge two dendrograms.

Therefore, the total cost of transmission for Phase 2 is given by:

$$Transmission_{\text{total}} = O(n) \tag{20}$$

In order to evaluate time and space complexity of generating the global dendrogram (Phase 3), first we analyze the time and space complexity of the merge-dendrograms( ) method (figure 4), the key component of Phase 3. The merge-dendrograms( ) algorithm uses a recursive top-down strategy (with no backtracking) to efficiently merge two dendrograms. The recursion stops when Case 1, Case 2a, or Case 3a ("stopping" cases) happens or a leaf node is reached. Otherwise, the merge-dendrograms( ) or the build-global-dendrogram( ) process proceeds recursively. In constant time we can decide which of the six cases for merging the two dendrograms occurs. Assuming that the branching factor B of a dendrogram is a constant, we can perform basic operations on dendrograms (adding and deleting a child, updating descriptive statistics, finding best match, etc.) required by each case in constant time as well. Every time a non-stopping case is selected, we descend a level in the dendrogram. If the average depth of a dendrogram is $O(\log_B n)$, merging two dendrograms requires on average $O(\log_B n)$ time. However, for very unbalanced dendrograms, the worst case for merging two dendrograms requires $O(n)$ time. This is an upper bound. As for the space cost of the merge-dendrograms( ) method, we need $O(n)$ space to store both dendrograms and $O(\log_B n)$ average stack space to process the recursion. Thus, the merge-dendrograms( ) method requires $O(n)$ space.

To compute the overall time and space costs of Phase 3 described by the build-global-dendrogram( ) algorithm (figure 2), we first note that this algorithm is an adaptation of the hierarchical clustering algorithm [15]. There are two main parts in this algorithm: the *initialization* part represented by lines (1) through (4) and the *agglomeration* part represented by lines (6) through (11). Computing the array storing each $d_{\text{approx}}(\vec{c}_i, \vec{c}_j)$ (line (1)) requires $O(|S|^2)$ space and time. Given this array, the initialization of arrays storing each $NN(i)$ and $DISS(i)$ adds a factor of $O(|S|)$ and $O(|S|^2)$ to the overall space and time complexity, respectively. Thus, the time and space complexities of the initialization part of the algorithm are given by:

$$Time_{\text{init}} = O(|S|^2) \tag{21}$$

$$Space_{\text{init}} = O(|S|^2) \tag{22}$$

The agglomeration part that starts on line (6) repeats $|S| - 1$ times. Determining the two closest dendrograms (lines (7) through (9)) can be performed in $O(|S|)$ time by examining each dendrogram's best match. Based on the complexity analysis of the merge-denrograms( ) algorithm, the agglomeration in line (10) requires $O(n)$ time and $O(n)$ space. Finally, the updating step (line (11)) can be performed in $O(|S|)$ time for metrics that satisfy the *reducibility property* [14]. Otherwise, the algorithm requires at most $O(|S|^2)$ per iteration to update the arrays. Thus, the time complexity of the agglomeration part of the algorithm is given by:

$$\begin{aligned} Time_{\text{agglom}} &= O((|S|-1)\cdot|S|) + O((|S|-1)\cdot n) + O((|S|-1)\cdot|S|^2) \\ &= O(|S|^2) + O(|S|n) \end{aligned} \tag{23}$$

The space complexity of the agglomeration part of the algorithm is given by:

$$Space_{\text{agglom}} = O(|S|^2) + O(n) \tag{24}$$

Hence, the overall time and space complexity for hierarchical clustering of local dendrograms presented in figure 2 is given by:

$$\begin{aligned} Time_{\text{total}} &= Time_{\text{init}} + Time_{\text{agglom}} \\ Space_{\text{total}} &= Space_{\text{init}} + Space_{\text{agglom}} \end{aligned}$$

Using the time and space costs as given in Eqs. (21) through (24), the total time and space cost for Phase 3 is given by:

$$\begin{aligned} Time_{\text{total}} &= O(|S|^2) + O(|S|n) \\ Space_{\text{total}} &= O(|S|^2) + O(n) \end{aligned}$$

Which are effectively $O(n)$, when $|S|$ is constant and $|S| \ll n$.

4. Empirical evaluation

In this section, we evaluate the effectiveness of RACHET on several datasets. Tests are done on synthetic datasets and also on "real world" datasets from the ML Repository at UC Irvine, available at http://www.ics.uci.edu/AI/ML/MLDBRepository.html. We use Ward's agglomerative hierarchical clustering algorithm [1] for generating local dendrograms in all of our experiments.

4.1. Experimental methodology

In order to evaluate the effectiveness of the RACHET algorithm relative to the centralized Ward's clustering algorithm, we use the method described by Kargupta et al. [11]. The dendrograms generated in the centralized and distributed fashion are "cut" at different levels such that the same number of clusters, l, results from each. Then, an adjacency matrix is constructed as follows. The element a_{ij} of the adjacency matrix is one if the ith and jth data points belong to the same cluster. Otherwise it is zero. The error $E(l)$ of misclassifications comparing the centralized and distributed algorithms is measured as the ratio of the sum of absolute differences between elements of adjacency matrices to the total number of elements. More formally, $E(l)$ is defined as:

$$E(l) = \frac{\sum_{j=1}^{n} \sum_{i=1}^{n} |c_{ij} - d_{ij}|}{n^2}, \tag{25}$$

where c_{ij} and d_{ij} are elements of the adjacency matrix for centralized and distributed algorithm, respectively.

4.2. Results for synthetic data sets

The main purpose of this section is to study the sensitivity of RACHET to various characteristics of the input. The characteristics include various partitions of data points across data sites, the number of data sites, and different dimensionality of data. We first introduce the synthetic data sets.

Synthetic data was created for dimensionality $d = 2$, 4, 8, and 16. For a given value of d, data was sampled from four Gaussian distributions (hence number of clusters $K = 4$).

The number of points in each Gaussian is n/K and its mean vector is sampled from a uniform distribution on $[min\_val + k \cdot max\_val,\ max\_val + k \cdot max\_val]$ for $k = 0, \ldots, K-1$. The values for min_val and max_val are 5 and 15, respectively. Elements of the diagonal covariance matrix are sampled from a uniform distribution on $[0, min\_val/6]$. Hence, these are fairly well-separated Gaussians, an ideal situation for a centralized Ward's clustering algorithm. Note that $E(l)$ for $l = K$ in (25) is calculated with the correct classification of points by the centralized algorithm. In many real world data sets, the behavior of the centralized algorithm is not "best-case" for a given K, hence the misclassification error of the distributed algorithm relative to the centralized algorithm is not necessarily an appropriate measure (and indeed it is as demonstrated in Table 4). In this case, a comparison with a known classifier might be preferred.

We test the performance of the algorithm with different numbers of data sites and various distributions of data points across data sites. Note that the "best-case" scenario for RACHET is when all points of the same cluster are assigned to be in a single data site (homogeneous assignment). The "worst case" scenario most likely occurs when each data site contains a subset of points from each of the K clusters (heterogeneous assignment). Table 1 shows the percentage of misclassifications of our algorithm relative to the centralized algorithm for 2, 4, and 6 data sites with a heterogeneous assignment of data points. More precisely, n_i $(i = 0, \ldots, K-1)$ points are randomly selected from each Gaussian and assigned so that each data site has points from each of the Gaussians. For our experiments, n_i is chosen as $(n/K)/|S|$ so that each data site has roughly the same number of points from each Gaussian. The total number of data points $n = 1024$. The dendrograms were split at level $l = 2, 4, 6, 8$, and 10. Since we use random sampling in creating synthetic data sets and assigning points to data sites, here we present the average of twenty-five different runs of the distributed algorithm. We generate a synthetic data set and prepare five different random assignments of data points to sites. We take the average of the five resulting adjacency matrices. Each average value is rounded off to the nearest Boolean value. Then we average the obtained misclassification errors across five different synthetic data sets.

We can make an observation from the Table 1 that RACHET achieves good performance at a higher division level of the dendrogram. Its behavior on the synthetic data remains

Table 1. Percentage of misclassifications at different level of division of the global dendrogram generated from $|S| = 2$, 4, and 6 sites compared to the centrally generated dendrogram.

Divison	$d = 2$			$d = 4$			$d = 8$			$d = 16$		
level	$\|S\| = 2$	$\|S\| = 4$	$\|S\| = 6$	$\|S\| = 2$	$\|S\| = 4$	$\|S\| = 6$	$\|S\| = 2$	$\|S\| = 4$	$\|S\| = 6$	$\|S\| = 2$	$\|S\| = 4$	$\|S\| = 6$
2	46%	32%	36%	22%	36%	36%	46%	41%	39%	28%	20%	28%
4	9%	24%	27%	19%	9%	22%	14%	12%	12%	9%	13%	10%
6	9%	12%	8%	9%	12%	10%	12%	13%	12%	5%	12%	12%
8	14%	10%	10%	10%	14%	13%	11%	15%	14%	8%	14%	16%
10	17%	11%	12%	11%	12%	13%	13%	13%	14%	8%	16%	17%

Results are for synthetic data sets of size $n = 1024$ and dimension $d = 2, 4, 8$, and 16 with heterogeneous assignment of points to data sites.

roughly unchanged with the number of dimensions and the number of data sites. RACHET shows the worst performance for division level 2, which seems quite natural for the "worst-case" scenario of heterogeneous assignment of data items to data sites. We have not observed the degradation of performance at this division level for the homogeneous assignment of data points.

4.3. Results on real-world data sets

We have tested the algorithm on 3 publicly available "real-world" datasets obtained from UCI ML Repository: the Boston Housing data, the E.coli data, and the Pima Indians Diabetes data. Table 2 provides a brief summary of these data sets (see Appendix A for more detailed dataset statistics by features).

The *E.coli* dataset contains 336 data points in 7 dimensions that are classified into 8 clusters. Figure 5 shows the density plot constructed based on the adjacency matrix. The matrix is obtained by the centralized algorithm with the division level set to 8. Figure 6 shows the density plot obtained by our algorithm for the same division level with two data sites. Figure 7 shows their difference. The error of misclassification relative to the centralized algorithm is 9%.

Table 3 summarizes the comparative results at different levels of division of the dendrogram between the centrally generated dendrogram and the global dendrogram generated from two and four data sites for all three data sets. For each data set, experiments are run

Table 2. Brief summary of the three data sets from the UCI ML Repository.

Data set	No. of items n	No. of features d	No. of classes
E.coli	336	7	8
Boston Housing	506	14	N/A
Pima Indians Diabetes	768	8	2

Table 3. Percentage of misclassifications at different level of division of the global dendrogram generated from $|S| = 2$ and 4 sites compared to the centrally generated dendrogram.

Divison level	Boston Housing		*E.coli*		Pima Indians Diabetes	
	$\|S\| = 2$	$\|S\| = 4$	$\|S\| = 2$	$\|S\| = 4$	$\|S\| = 2$	$\|S\| = 4$
2	49%	34%	32%	45%	49%	47%
4	29%	34%	8%	32%	36%	43%
6	26%	24%	9%	29%	31%	36%
8	20%	18%	9%	21%	17%	37%
10	18%	17%	10%	22%	16%	29%
12	14%	13%	10%	22%	15%	26%

Results are for real data with random assignment of points to data sites.

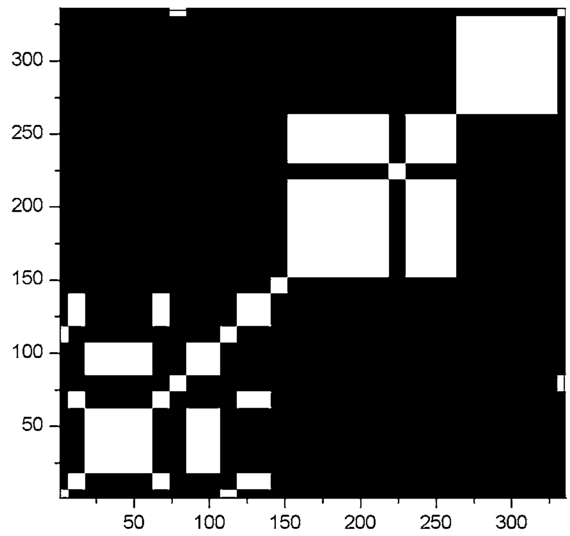

Figure 5. Density plot at division level 8 for centralized clustering of the *E.coli* data.

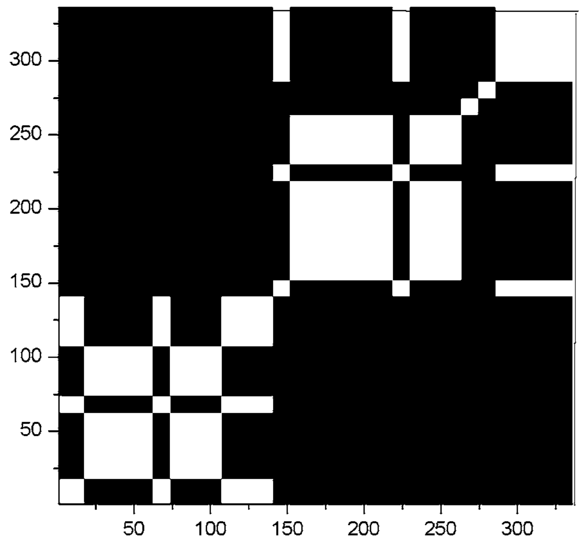

Figure 6. Density plot at division level 8 for distributed clustering of the *E.coli* data with two data sites.

five times with different random assignment of data points to data sites and the results are averaged over these runs. No class labels have been used. Note that the performance does not change with the number of dimensions. It improves at higher division levels as opposed to the results on the Boston Housing data provided in [10] for vertical (by features) distribution of data items across data sites.

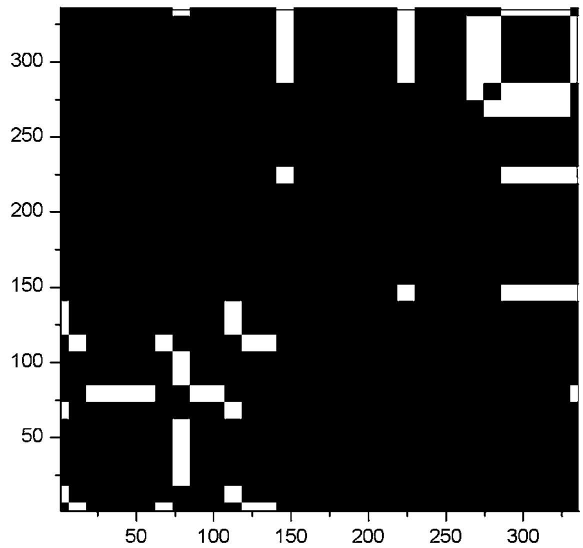

Figure 7. Difference between adjacency matrices obtained for centralized and distributed algorithm at division level 8 for the *E.coli* data with two data sites.

Comparing the results for real (Table 3) and synthetic (Table 1) data sets, we see that the accuracy of RACHET is slightly worse for real data sets. There are several reasons for the performance deterioration. First, the real data sets are more heterogeneous in terms of the units of measurements and the range of values for the parameters of descriptive statistics. This is illustrated by the summary statistics in Appendix A. Second, the error of misclassification is evaluated relative to the centralized algorithm (see (25)). It may not necessarily be a good measure of RACHET's performance since the behavior of the centralized algorithm may be poor for some real world data sets. For example, Table 4 demonstrates that the centralized algorithm performs worse than the distributed algorithm when compared with the clustering results for known class labels.

We also test the scalability of RACHET with the number of data sites. While the performance for the Boston Housing data (see Table 3), like the performance for synthetic data sets, remains roughly unchanged as the number of data sites increases, the performance

Table 4. Comparative results have between clusters identified by class labels and clusters obtained by the centralized algorithum at the division level of the dendrogram set to the number of classes and the global dendrogram generated from $|S| = 2$ and 4 sites at the same division level.

		Clusters with known class labels compared to:		
Data set	# Classes	Centralized	$\|S\| = 2$	$\|S\| = 4$
E.coli	8	22%	17%	20%
Pima Indians Diabetes	2	50%	39%	49%

for the E.coli and Pima Indians Diabetes data sets degrades. We have not identified the reasons for the performance degradation of the latter data sets; however, we expect better scalability with the number of data sites by enriching the descriptive statistics so that a desirable trade-off between transmission cost and accuracy is achieved. Currently, we are investigating approaches in this direction such as using Principal Component Analysis [8], providing low-degree polynomial approximations to feature vectors, as well as explicitly adding coordinates of cluster centroids corresponding to some division level k, $k \ll n$, of each local dendrogram.

5. Discussion

5.1. Error analysis

In this section, we present a brief discussion of the error in our Euclidean distance approximation (19). Let $\varepsilon(\vec{c}_1, \vec{c}_2)$ denote an absolute error of the approximation for the Euclidean distance between two centroids $\vec{c}_1$ and $\vec{c}_2$ defined as:

$$\varepsilon(\vec{c}_1, \vec{c}_2) = |d(\vec{c}_1, \vec{c}_2) - d_{\text{approx}}(\vec{c}_1, \vec{c}_2)| \tag{26}$$

First, we make several observations about the behavior of $\varepsilon(\vec{c}_1, \vec{c}_2)$.

Observation 1. If $MIN_{c_1} = MAX_{c_1}$ then $\varepsilon(\vec{c}_1, \vec{c}_2) = 0$ for any centroid $\vec{c}_2$.

In other words, if one of the centroids lies on the bissecting line (i.e., all the vector coordinates are the same), then the approximated distance equals the exact distance.

Observation 2. If $MIN_{c_1} \neq MAX_{c_1}$ and $MIN_{c_2} \neq MAX_{c_2}$ then

$$\max_{\vec{c}_2} \varepsilon(\vec{c}_1, \vec{c}_2) = \varepsilon(\vec{c}_1, \vec{c}_1) \quad \text{and} \quad \max_{\vec{c}_1} \varepsilon(\vec{c}_1, \vec{c}_2) = \varepsilon(\vec{c}_2, \vec{c}_2).$$

In other words, the absolute error achieves its maximum value when centroids are very close to each other provided neither of them lies on the bissecting line.

Observation 3. Let $\vec{c}$ be any data point on the bissecting line (e.g., unit vector). By Observation 1, descriptive statistics of $\vec{c}_1$ and $\vec{c}_2$ are sufficient for the exact calculation of the low and upper bounds for $d_{\text{approx}}(\vec{c}_1, \vec{c}_2)$ defined as:

$$|d(\vec{c}_1, \vec{c}) - d(\vec{c}_2, \vec{c})| \leq d_{\text{approx}}(\vec{c}_1, \vec{c}_2) \leq d(\vec{c}_1, \vec{c}) + d(\vec{c}_2, \vec{c})$$

Hence,

$$\varepsilon(\vec{c}_1, \vec{c}_2) \leq 2 \cdot \min\{d(\vec{c}_1, \vec{c}), d(\vec{c}_2, \vec{c})\} \tag{27}$$

The proofs of these observations follow from elementary algebra and are omitted here.

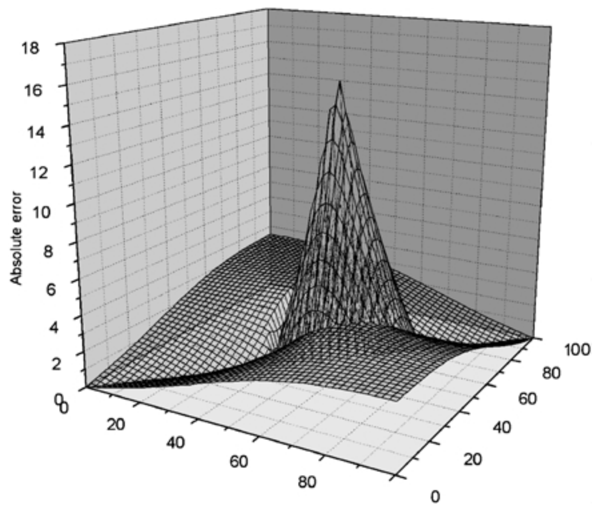

Figure 8. Absolute error plot for the first centroid set to (50,55) and the second centroid set to all integers between 0 and 100.

Figure 8 illustrates the behavior of the absolute error in 2D when the coordinates of $\vec{c}_1$ are set to (50, 55) and the coordinates of $\vec{c}_2$ are set to all possible integers between 0 and 100. We can see that the absolute error increases when $\vec{c}_2$ approaches $\vec{c}_1$ and equals zero when $\vec{c}_2$ reaches the bissecting line ($x = y$). The maximum value of the absolute error depends on the location of $\vec{c}_1$ as can be seen in figure 9 when $\vec{c}_1 = (99, 89)$ and is bounded by its distance to the bissecting line.

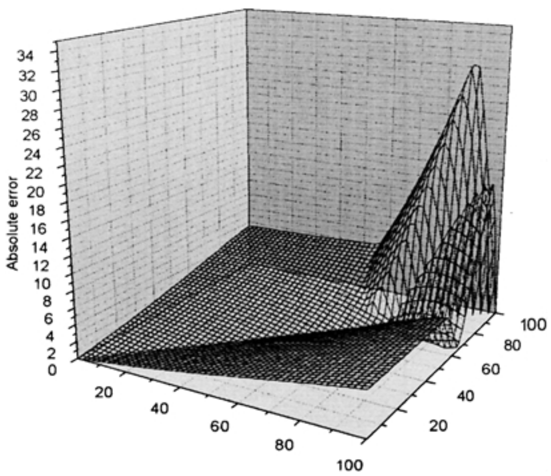

Figure 9. Absolute error plot for the first centroid set to (99,89) and the second centroid set to all integers between 0 and 100.

In spite of the fact that the absolute error (see (27)) of the Euclidean distance approximation can be large depending on the location of centroids, the overall performance of RACHET is still reasonably good as demonstrated on both synthetic and real datasets in Section 4. There is a heuristic explanation for this behavior. Note that if two centroids are very close to each other, then the merge_dendrograms( ) subroutine (see figure 4) will misclassify them as belonging to Case 1, i.e. well separated clusters, as opposed to one of the overlapping cases (Case 2, 3, or 4). In this situation, the recursion stops instead of refining the process of merging local dendrograms. However, this situation is more likely to occur closer to the leaves of local dendrograms rather than the roots. Many of our experiments on synthetic and real data sets support this statement with a few exceptions for some specifically designed assignments of data points to data sites. Hence, the performance of RACHET will degenerate more at the division level close to the leaves of the global dendrogram and will remain acceptable at moderate division levels.

5.2. *Future work*

Empirical results on synthetic Gaussian data indicate that RACHET provides a comparable quality solution to the distributed hierarchical clustering problem while being scalable with both the number of dimensions and the number of data sites. Results on the small real-world UCI ML data sets indicate that RACHET can provide a more effective clustering solution than the solution generated by the centralized clustering. The reason for using small real data sets is that the goal at this stage is to demonstrate ability to create a comparable quality global dendrogram from distributed local dendrograms within reasonable requirements for the time, space, and communication cost. However, based on the theoretical results for linear time/space/communication complexity of RACHET, the next step is to study the efficiency of RACHET in dealing with very large (gigabytes or terabytes) and very high-dimensional (thousands of features) real data sets. Example of such massive datasets might be the Reuters text classification database consisting of the documents with hundreds of thousands of words (i.e., hundreds of thousands of dimensions) or the PCMDI archive of climate simulation model outputs with each output in the order of a couple of terabytes and 2500 or more dimensions. We believe that the RACHET algorithm is scalable to such sizes of the problem because it transforms a large problem into a set of small subproblems with cumulative computational cost much less than the aggregate problem.

The distributed hierarchical clustering algorithm proposed here is in the context of centroid-based hierarchical algorithms using Euclidean distance as a dissimilarity measure between two data objects. We note that similar ideas can be extended to other hierarchical clustering algorithms as well as to non-Euclidean dissimilarity measures.

6. Summary

This paper presents RACHET, a hierarchical clustering method for very large, high-dimensional, horizontally distributed datasets. Most hierarchical clustering algorithms suffer from severe drawbacks when applied to very massive and distributed datasets: 1) they require prohibitively high communication cost to centralize the data to a single site and 2) they do not scale up with number of data items and with dimensionality of data sets. RACHET makes the scalability problem more tractable. This is achieved by generating local clustering hierarchies on smaller data subsets and using condensed cluster summaries for the consecutive agglomeration of these hierarchies while maintaining clustering quality. Moreover, RACHET has significantly lower (linear) communication costs than traditional centralized approaches.

Appendix A: Summary of datasets statistics

Feature number	Minimal feature value	Maximal feature value	Mean feature value	Standard deviation
		E.coli		
1	0.00	0.89	0.50	0.19
2	0.16	1.00	0.50	0.15
3	0.48	1.00	0.50	0.09
4	0.50	1.00	0.50	0.03
5	0.00	0.88	0.50	0.12
6	0.03	1.00	0.50	0.22
7	0.00	0.99	0.50	0.21
		Boston Housing		
1	0.01	88.98	3.61	8.60
2	0.00	100.00	11.36	23.32
3	0.46	27.74	11.14	6.86
4	0.00	1.00	0.07	0.25
5	0.39	0.87	0.55	0.12
6	3.56	8.78	6.28	0.70
7	2.90	100.98	68.57	28.15
8	1.13	12.13	3.80	2.11
9	1.00	24.00	9.55	8.71
10	187.00	711.00	408.24	168.54
11	12.60	22.00	18.46	2.16
12	0.32	396.90	356.67	91.29
13	1.73	37.97	12.65	7.14
14	5.00	50.00	22.53	9.20
		Pima Indians Diabetes		
1	0.00	17.00	3.8	3.4
2	0.00	199.00	120.9	32

(*Continued on next page.*)

(Continued).

Feature number	Minimal feature value	Maximal feature value	Mean feature value	Standard deviation
3	0.00	122.00	69.1	19.4
4	0.00	99.00	20.5	16
5	0.00	846.00	79.8	115.2
6	0.00	67.10	32	7.9
7	0.08	2.42	0.5	0.3
8	21.00	81.00	33.2	11.8

Acknowledgment

This research was supported in part by an appointment to the Oak Ridge National Laboratory Postdoctoral Research Associates Program administered jointly by the Oak Ridge Association of Universities and Oak Ridge National Laboratory.

References

1. M.R. Anderberg, Cluster Analysis and Applications, Academic Press: New York, 1973.
2. R. Brachman, T. Khabaza, W. Kloesgen, G. Piatetsky-Shapiro, and E. Simoudis, "Mining business databases," Communications of ACM, vol. 39, no. 11, pp. 42–48, 1996.
3. W.H.E. Day and H. Edelsbrunner, "Efficient algorithms for agglomerative hierarchical clustering methods," Journal of Classification, vol. 1, pp. 7–24, 1984.
4. I. Dhillon and D. Modha, "A data clustering algorithm on distributed memory multiprocessors," in Large-Scale Parallel Data Mining, Workshop on Large-Scale Parallel KDD Systems, Mohammed Javeed Zaki and Ching-Tien Ho (Eds.), SIGKDD, Aug. 15, 1999, San Diego, CA, USA, pp. 245–260.
5. R. Dubes and A. Jain, "Clustering methodologies in exploratory data analysis," Advances in Computers, vol. 19, pp. 113–228, 1980.
6. U. Fayyad, D. Haussler, P. Stolorz, and R. Uthurusamy (Eds.), Advances in Knowledge Discovery and Data Mining, MIT Press: Cambridge, MA, 1996.
7. K. Fukunaga, Introduction to Statistical Pattern Recognition, Academic Press: New York, 1990.
8. J.E. Jackson, A User's Guide to Principal Components, John Wiley & Sons: New York, 1991.
9. A.K. Jain, M.N. Murty, and P.J. Flynn, "Data clustering: A review," ACM Computing Surveys, vol. 31, pp. 264–323, 1999.
10. E. Johnson and H. Kargupta, "Collective, hierarchical clustering from distributed, heterogeneous data," Lecture Notes in Computer Science, vol. 1759, Springer-Verlag: Berlin, 1999.
11. H. Kargupta, W. Huang, K. Sivakumar, and E. Johnson, "Distributed clustering using collective principal component analysis," Knowledge and Information Systems, vol. 3, no. 4, pp. 422–448, 2001.
12. L. Kaufman and P. Rousseeuw, Finding Groups in Data, John Wiley and Sons: New York, 1989.
13. G.N. Lance and W.T. Williams, "A general theory of classificatory sorting strategies. 1: Hierarchical systems," Computer Journal, vol. 9, pp. 373–380, 1967.
14. F. Murtagh, "A survey of recent advances in hierarchical clustering algorithms," Computer Journal, vol. 26, pp. 354–359, 1983.
15. C. Olson, "Parallel algorithms for hierarchical clustering," Parallel Computing, vol. 8, pp. 1313–1325, 1995.