2011 11th IEEE International Conference on Data Mining

Semi-supervised Hierarchical Clustering

Li Zheng
School of Computing and Information Sciences
Florida International University
Miami, FL, USA
Email: lzhen001@cs.fiu.edu

Tao Li
School of Computing and Information Sciences
Florida International University
Miami, FL, USA
Email: taoli@cs.fiu.edu

***Abstract*—Semi-supervised clustering (i.e., clustering with knowledge-based constraints) has emerged as an important variant of the traditional clustering paradigms. However, most existing semi-supervised clustering algorithms are designed for partitional clustering methods and few research efforts have been reported on semi-supervised hierarchical clustering methods. In addition, current semi-supervised clustering methods have been focused on the use of background information in the form of instance level must-link and cannot-link constraints, which are not suitable for hierarchical clustering where data objects are linked over different hierarchy levels. In this paper, we propose a novel semi-supervised hierarchical clustering framework based on ultra-metric dendrogram distance. The proposed framework is able to incorporate triple-wise relative constraints. We establish the connection between hierarchical clustering and ultra-metric transformation of dissimilarity matrix and propose two techniques (the constrained optimization technique and the transitive dissimilarity based technique) for semi-supervised hierarchical clustering. Experimental results demonstrate the effectiveness and the efficiency of our proposed methods.**

***Keywords*-Hierarchical clustering; semi-supervised clustering; triple-wise relative constraints;**

I. Introduction

The clustering problem arises in many disciplines and has a wide range of applications. Basically clustering aims to group the given samples into clusters such that samples in the same cluster are similar to each other while samples in different clusters are dissimilar [1]. Based on the way the clusters are generated, clustering methods can be divided into two categories: partitional clustering and hierarchical clustering [2] [3]. Generally *partitional clustering* decomposes the dataset into a number of disjoint clusters which are usually optimal in terms of some predefined objective functions. *Hierarchical clustering* groups the data points into a hierarchical tree-like structure using bottom-up or top-down approaches.

In many situations when we discover new patterns using clustering, there are known prior knowledge about the problem. Recently, semi-supervised clustering (i.e., clustering with knowledge-based constraints) has emerged as an important variant of the traditional clustering paradigms [4] [5]. Given the data representation, existing semi-supervised methods have utilized background knowledge to learn a distance/dissimilarity measure, to modify the objective criterion for evaluating clustering, and to improve the optimization procedures [6]–[11].

There are two limitations in current studies of semi-supervised clustering. First, most of these existing semi-supervised clustering algorithms are designed for partitional clustering methods and *few research efforts have been reported on semi-supervised hierarchical clustering methods*. Different from partitional clustering where the clustering results can be easily represented using vectors, clustering indicators, or connectivity matrices for optimization [10], the results of hierarchical clustering are more complex and typically represented as dendrograms or trees. In addition, hierarchical clustering methods have no global objective functions. These properties have made the semi-supervised hierarchical clustering problem more challenging.

Another limitation is *on the types of constraints*. Existing semi-supervised clustering methods have been focused on the use of background information in the form of instance level must-link and cannot-link constraints. A must-link (ML) constraint enforces that two instances must be placed in the same cluster while a cannot-link (CL) constraint enforces that two instances must not be placed in the same cluster. However, *both ML and CL constraints are not suitable for hierarchical clustering methods since objects are linked over different hierarchy levels* [12] [13].

In this paper, we propose a semi-supervised hierarchical clustering framework based on the ultra-metric dendrogram distance. The characteristics of our proposed framework are summarized below:

1) Triple-wise relative constraints: In the proposed framework, we consider the triple-wise relative constraints in the form of (x_i, x_j, x_k) which indicates the dissimilarity (or the distance) between x_i and x_j, noted as $d(x_i, x_j)$, should be smaller than $d(x_i, x_k)$. The relative constraint, also referred as must-link-before (MLB) constraint, specifies the order in which the objects are merged (or linked) and can be naturally incorporated into the hierarchical clustering process.
2) Ultra-metric dendrogram distance: Our proposed framework is based on ultra-metric dendrogram dis-

1550-4786/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDM.2011.130

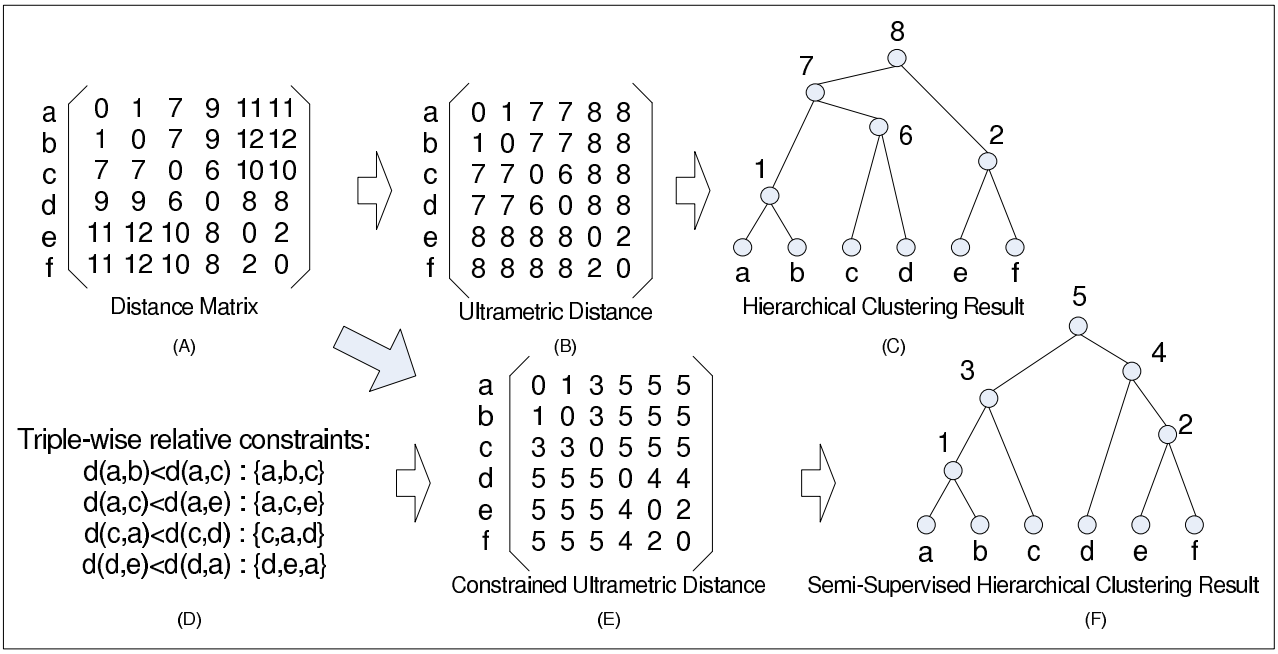


Figure 1. An illustrative example of semi-supervised hierarchical clustering with triple-wise relative constraints. The original data dissimilarity matrix is shown in (A). (B) shows a standard transitive dissimilarity matrix obtained from the original dissimilarity and (C) is the corresponding hierarchical clustering result without constraints. The triple-wise relative constraints are given in (D). By combining both (A) and (D), the constrained ultra-metric distance matrix is shown in (E) with its corresponding hierarchial clustering result in (F).

tance. Note that the results of hierarchical clustering can be represented using ultra-metric distance matrices [14]. Using the ultra-metric distance matrices, we propose two techniques for solving semi-supervised hierarchical clustering problem: the optimization-based technique and the transitive dissimilarity based technique.

3) Effectiveness and efficiency: Extensive experimental results demonstrate the effectiveness and efficiency of our proposed framework.

An illustrative example of semi-supervised hierarchical clustering is given in Figure 1. The original dissimilarity is shown in Figure 1(A). Its ultra-metric distance matrix is shown in Figure 1(B) and the corresponding hierarchical clustering result (without constraints) is shown in Figure 1(C). Four triple-wise relative constraints are given in Figure 1(D). A constrained ultra-metric distance matrix is obtained in Figure 1(E) and its corresponding hierarchical clustering result (with constraints) is shown in Figure 1(F).

To sum up, different from existing research efforts on semi-supervised (hierarchical) clustering, in our work, we explicitly establish the equivalence between ultra-metrics and hierarchical clustering and also provide a unified framework integrating both ultra-metric fitting and triple-wise relative constraints. Our proposed framework seeks an approximate dissimilarity metric (ultra-metric) which represents a tuned dendrogram that satisfies the given constraints. Two different solutions based on iterative projection and heuristic (modified Floyd-Warshall) algorithms are proposed and empirically evaluated. The rest of the paper is organized as follows: Section II discusses the related work; Section III formally defines the semi-supervised hierarchical problem; Section IV explicitly establishes the connection between ultra-metric distance matrix and hierarchical clustering dendrogram; Section V presents two different techniques for semi-supervised hierarchical clustering based on ultra-metric distance; Section VI describes the experimental results; and finally Section VII concludes the paper.

II. Related Work

A. Hierarchical Clustering

Hierarchical clustering algorithms are unsupervised methods to generate tree-like clustering solutions. They group the data points into a hierarchical structure using bottom-up (agglomerative) or top-down (divisive) approaches [3]. The typical bottom-up approaches take each data point as a single cluster to start with and then builds bigger clusters by grouping similar clusters together until the entire data set is grouped into one final cluster. The divisive approaches start with all data points in one cluster and then split the largest cluster recursively. Many research efforts have been reported on algorithm-level improvements of the hierarchical

clustering process and on understanding of hierarchical clustering [15] [16].

B. Semi-supervised Clustering

Integrating background knowledge into the clustering process has been investigated extensively. Many researchers have explored the use of instance-level background information, such as pairwise must-link and cannot-link constraints for learning a distance/dissimilarity measure, or modifying the objective criterion, or improving the optimization procedures [6]–[11]. Other types of knowledge hints (such as size of the clusters, partial labels of the data points, and user-provided external prototypes/representatives) have also been used for clustering [17]. However, most of these works are designed for partitional clustering and few research efforts have been reported on semi-supervised hierarchical clustering methods. We note the very recent works of Zhao et al. [18] and Bade et al. [12] which perform hierarchical clustering with order constraints and partially known hierarchy. Conceptually our triple-wise constraints are special cases of the order constraints. However, different from their works, our proposed semi-supervised hierarchical clustering framework is based on ultra-metric dendrogram distance. Experimental studies demonstrate the effectiveness and efficiency of our proposal.

C. Metric Fitting

The problem of fitting a tree metric to the (dis-)similarity data on pairs of objects from a given set has been studied quite extensively [19]. Ultra-metric is a special kind of tree metric where all elements of the input dataset are leaves in the underlying tree and all leaves are at the same distance from the root. Ultra-metric naturally corresponds to a hierarchy of clusterings of the data. Given a dissimilarity D on pairs of objects, the problem of finding the best ultra-metric d_u such that $||D - d_u||_p$ is minimized is NP-hard for L_1 and L_2 norms (e.g., when $p = 1$ and $p = 2$) [20]. In this paper, we propose two techniques for fitting an ultra-metric using the given relative constraints.

III. Problem Statement

A. Problem Definition

Given set of instances $X = \{x_1, x_2, \cdots, x_n\}$, their pairwise dissimilarities $D = \{d(x_i, x_j) | x_i, x_j \in X\}$ and a set of constraints $C = \{(x_i, x_j, x_k) \ | d(x_i, x_j) < d(x_i, x_k) , x_i, x_j, x_k \in X\}$. The semi-supervised hierarchical clustering problem aims to output a clusters hierarchy/dendrogram $\mathcal{H}$ to *satisfy as many constraints as possible* and meanwhile to *maintain the merge order based on sample dissimilarities as close as possible*.

Note that hierarchical clustering results can be represented graphically on dendrograms as shown in Figure 2. The vertical line along with the clustering dendrogram is labeled by the value of the updated dissimilarity between the merged clusters, which can be treated as a measure of separation of paired samples. The dissimilarity of sample a and c in the dendrogram is noted by $level(a, c)$. Note that some relative constraints (e.g., constraint (a, b, c) in Figure 2) are consistent with the given dissimilarity matrix while many constraints are not (e.g., constraint (a, d, b)).

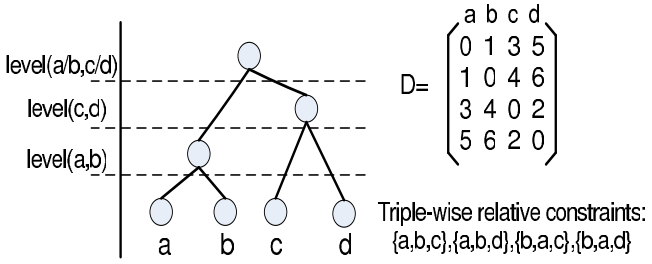


Figure 2. Triple-wise relative constraints for samples a and b in 4-point sample set.

B. Constraint Pre-processing

Transitive Closure: Constraints given by human experts or by partially known data hierarchy may be *incomplete*. Some constraints are not explicitly given, for example, two given constraints $c_1 = (x_i, x_j, x_k)$ and $c_2 = (x_i, x_k, x_l)$ imply an additional constraint $c_3 = (x_i, x_j, x_l)$ which might not be explicitly stated. In our framework, given the initial constraint set, we perform Floyd-Warshall algorithm to find its transitive closure and extend the constraint set.

Conflict Removal: In practice, the given constraints may be *conflicting*. For example $c_1 = (x_i, x_j, x_k)$ and $c_2 = (x_j, x_k, x_i)$ are explicitly conflicting with each other or $c_1 = (x_i, x_j, x_k)$, $c_2 = (x_i, x_k, x_l)$ and $c_3 = (x_i, x_l, x_j)$ form a circle of merge orders. Conflicts in the constraint set can form deadlocks, and the clustering algorithm may fail to identify a valid merging pair of clusters. To remove conflicts, we randomly and iteratively remove one of the conflicting constraint until there is no conflict.

IV. Ultra-metric Distance

Our proposed framework is based on ultra-metric distance matrices. In this section, we establish the connection between hierarchical agglomerative clustering and ultra-metric transformation of pairwise dissimilarity matrix.

A. Ultra-metric Interpretation of Hierarchical Clustering

In each merge of an agglomerative hierarchical clustering, the algorithm finds two closest clusters to merge as a new one. So, the merged clusters C_i and C_j satisfy:

$$d(C_i, C_j) \leq \min(d(C_i, C_k), d(C_j, C_k)), \quad (1)$$

and it also implies

$$\forall i, j, k, \min(d(C_i, C_k), d(C_j, C_k)) \leq d(C_i \cup C_j, C_k). \quad (2)$$

The above **reducibility property** has been studied in [21]. A brief interpretation is that each merge of clusters always combines the pair of reciprocal nearest neighbors and does not change the initial merge orders. Once the reducibility condition holds, the updated dissimilarities satisfy the ultra-metric inequality [21]:

$$d(x_i, x_j) \leq \max(d(x_i, x_k), d(x_j, x_k)), \forall x_i, x_j, x_k \in X. \tag{3}$$

Therefore, a hierarchical clustering transforms the original dissimilarity matrix D into an ultra-metric distance $\hat{D}$. In other words, hierarchical clustering can be understood as a ultra-metric transformation $\hat{D}$ on the original dissimilarity matrix D which uniquely characterizes a ultra-metric tree. Such transformation can be formulated as using mathematical programming. For simplicity, let D_{ij} denote the ij-th entry of D, i.e., $d(x_i, x_j)$. The hierarchical clustering problem can be modeled as seeking for an optimal ultra-metric distance matrix $\hat{D}$:

$$\arg\min_{\hat{D}} \sum_{x_i, x_j \in X} (D_{ij} - \hat{D}_{ij})^2. \tag{4}$$

The above least-squares loss (L2) optimization problem is known to be NP-hard [22]. Many approximation approaches from both combinatorial/regression [23] and gradient-based [24] [25] perspectives have been proposed to identify the local optimal solution.

B. Transitive Dissimilarity

The ultra-metric distance matrix can also be obtained using transitive dissimilarity. In order to construct the transitive dissimilarity on D, we first treated D as the transition matrix on a graph in which each column/row represents an sample/node in the graph and each entry represent the edge weight associated with a pair of nodes. The idea of transitive dissimilarity is to **preserve transitivity** of graph. For example, the transitivity of three nodes $x_1 \to x_2 \to x_3$ in terms of dissimilarity can be defined as

$$d_{13} \leq \max(d_{12}, d_{23}),$$

For simplicity, $d(x_i, x_j)$ is noted as d_{ij} in this section, so the distance d_{13} should be no worse than either d_{12} or d_{23}, which clearly enforces a tighter bound than triangle inequality. In general, such relation refers to the ultra-metric inequality:

$$d_{ij} \leq \max(d_{ik}, d_{kj}), \forall x_i, x_j, x_k.$$

To extend this definition, we consider a(any) path P_{ij} between x_i and x_j. The **transitive dissimilarity** on path P_{ij} can be defined as

$$T(P_{ij}) = \max(d_{i,k_1}, d_{k_1,k_2}, d_{k_2,k_3}, \cdots, d_{k_{n-1},k_n}, d_{k_n,j}). \tag{5}$$

Among all existing paths for any given pair of vertices x_i and x_j, the **minimal transitive dissimilarity** is defined as:

$$m_{ij} = \min_{P_{ij}}(T(P_{ij})), \text{for given vertices } x_i \text{ and } x_j. \tag{6}$$

It is clear that $m_{ij} \geq d_{ij}, \forall x_i$ and x_j, which implies that minimal transitive dissimilarity extends vertices further than the original distance matrix and the minimal transitive dissimilarity between any pair of vertices holds the ultra-metric inequality [26]:

Proposition 1: For any weighted dissimilarity graph, the minimal transitive dissimilarity between any pair of vertices holds the ultra-metric inequality:

$$m_{ij} \leq \max(m_{ik}, m_{kj}), \forall x_i, x_j, x_k.$$

Proof: Let P_{ij} is a set of all paths in which each element indicates an existing path connecting V_i and V_j as its end points. (P_{ik}, P_{kj}) is describing a path starting from V_i to V_j via V_k in a weighted graph. It is clear that (P_{ik}, P_{kj}) is a subset of P_{ij}. We define $W(P_{ij})$ as edge weights of any directly connected vertices in all possible paths Pij.

$$\begin{aligned} m_{ij} &= \min_{P_{ij}} \max[W(P_{ij})] \\ &\leq \min(P_{ik}, P_{kj}) \max(W(P_{ik}, P_{kj})) \\ &= \min(P_{ik}, P_{kj}) \max[\max[W(P_{ik})], \max[W(P_{kj})]] \\ &= \max[\min_{P_{ik}}(\max[W_{P_{ik}}]), \min_{P_{kj}}(\max[W_{P_{kj}}])] \\ &= \max(m_{ik}, m_{kj}). \end{aligned}$$

■

C. Cluster Separation Enhancement and Transitive Dissimilarity

Here we demonstrate the enhancement of cluster separation due to the transitive dissimilarity. We use a small dataset shown in Figure 3, where the two clusters (i.e., (1,2,3) and (4,5)) are reasonably visible.

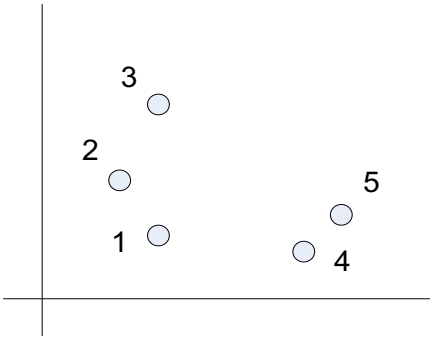


Figure 3. Illustration of cluster separation due to the transitive dissimilarity for a simple dataset of 5 points in 2D space.

The original distance of the dataset is shown in Table I and the transitive distance is shown in Table II. The distance

Table I
ORIGINAL DISTANCE AMONG THE 10 OBJECTS SHOWN IN FIGURE 3.

i, j	1	2	3	4	5
1	0	138	300	262	230
2	138	0	164	289	272
3	300	164	0	400	384
4	262	289	400	0	85
5	230	272	468	85	0

Table II
TRANSITIVE DISTANCE AMONG THE 10 OBJECTS SHOWN IN FIGURE 3.

i, j	1	2	3	4	5
1	**0**	**138**	**164**	230	230
2	**138**	**0**	**164**	230	230
3	**164**	**164**	**0**	230	230
4	230	230	230	**0**	**85**
5	230	230	230	**85**	**0**

is computed using Euclidean distance and the value is scaled by multiplying 1000 for readability. It is clear that the distance in Table II provides an enhanced/improved 2-cluster structure, because the diagonal block (points 1-3) and (points 4 and 5) elements (distance among the same cluster) are visibly reduced. while the distance between the two clusters remain at the fixed value 230.

For example, the original distance between x_1 and x_3 (they are in the same cluster) $d_{13}^{original} = 300$, while the original distance between x_1 and x_4 or x_1 and x_5(they are in different cluster) $d_{14}^{original} = 262$, $d_{15}^{original} = 230$. This is not **normal** because it implies that members of the same cluster could have **larger** distance than the distance between members of different clusters.

With transitive distance, this abnormal situation is **corrected** because now $d_{13}^{transtive} = 164$, while $d_{14}^{original} = d_{15}^{original} = 230$.

V. SEMI-SUPERVISED HIERARCHICAL CLUSTERING

We propose two approaches for semi-supervised hierarchical clustering based on ultra-metric distance matrices: the optimization-based approach and the transitive dissimilarity based approach. The optimization-based approach models the semi-supervised hierarchical clustering as a constrained optimization problem of constructing an optimal distance matrix satisfying both the ultra-metric constraints and relative constraints. The transitive dissimilarity based approach aims to incorporate the relative constraints into the process of constructing the transitive dissimilarity.

A. Constraint-based Optimization

In semi-supervised hierarchical clustering, besides satisfying ultra-metricity, the clustering results should also consider relative constraints. We assume the dissimilarity matrix is non-negative and symmetric in our proposed algorithm, so we can adopt a vector representation. Suppose we have n samples and r relative constraints. For simplicity, the $n \times n$ symmetric dissimilarity matrix D can be represented by an $m \times 1$ vector $\vec{d}$ with $m = n * (n-1)/2$ entries of D's upper/lower triangle elements. Thus, each relative constraint $(x_i, x_j, x_k) \in C$ can also be represented by an $m \times 1$ vector $\vec{c}$ where the index corresponds to D_{ij} is set to 1 and the index of D_{ik} is set to -1. So, for any constraint c that is not consistent with the given dissimilarity matrix, we have $d^T c \geq 0$. An illustrative example is shown in Figure 4. Following the vector representation of dissimilarity and constraints, semi-supervised hierarchical clustering problem can be represented in the form below:

$$\arg\min_{\hat{d}} (\vec{d} - \vec{\hat{d}})^T E (\vec{d} - \vec{\hat{d}}), \tag{7}$$

subject to

$$\hat{D}_{ij} \leq \max\{\hat{D}_{ik}, \hat{D}_{jk}\}, \forall x_i, x_j, x_k \in X, \tag{8}$$

$$C\vec{d} \leq \vec{0}, \tag{9}$$

where $\vec{d}$ and $\vec{\hat{d}}$ are vectors representing pair-wise dissimilarities, E is a $m \times m$ identity matrix, and $C = [c_1^T; c_2^T; \cdots; c_r^T]$ is an $r \times m$ matrix containing all r relative constraints.

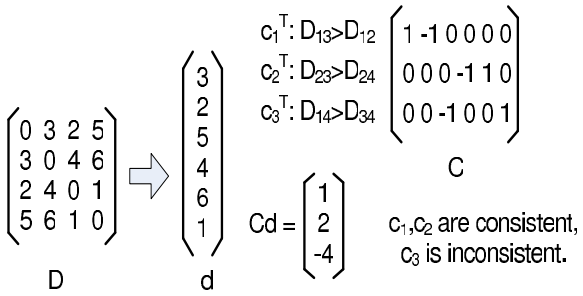


Figure 4. Utilizing constraints in the optimization framework.

The above optimization problem can be solved by conducting iterative projection approach which provides optimal solution to minimize the least-square loss function under inequality constraints [27] [28] [29]. Different from related approaches, our problem formulation considers both ultra-metric and triple-wise relative constraints and seeks an approximate dissimilarity metric (ultra-metric) that satisfies the given constraints. The ultra-metricity of the dissimilarity is taken as the underlying constraints to generate a tree-like hierarchy. Iterative projection can be generally conducted by repeatedly following the iterative "augmenting" steps. At each iteration, the parameter estimates are first projected onto closed convex sets defined by the inequality constraints $C\vec{d} \leq \vec{0}$, and are then updated by subtracting a vector of the changes made in the previous projection. Iterative projection carried out with this augmentation step is guaranteed to converge to the least squares optimal solution for a given fixed set of constraints [30].

Algorithm 1 shows a simple implementation of iterative projection used in [31]. The procedure simultaneously generates sequence of estimated solutions $a(t)$ and a sequence of Kuhn-Tucker vectors $u(t)$ where $a(t)$ and $u(t)$ denote the $\vec{a}$ and $\vec{u}$ in iteration t [32].

Algorithm 1 Iterative Projection to minimize least-square error

Input: $\vec{d}, C, E$
Output: $\hat{\vec{d}}$
Init: $\vec{a} = \vec{d}$ and $\vec{u} = \vec{0}$.

1: **while** not converge **do**
2: $p = t \bmod r$
3: $\vec{s} = \vec{a}(t-1) + E\vec{c_p}\vec{u}(t-1)_p/2$
4: **for** $q = 1$ to r **do**
5: **if** $q = p$ **then**
6: $\vec{u}(t)_q = \max(0, 2 * \vec{c_q^T}\vec{s}/\vec{c_q^T}E\vec{c_q}))$
7: **else**
8: $\vec{u}(t)_q = \vec{u}(t-1)_q$
9: **end if**
10: **end for**
11: $\vec{a}(t) = \vec{s} - E\vec{c_q}\vec{u}(t)_q/2$
12: $t = t + 1$
13: **end while**
14: **return** $\hat{\vec{d}} = \vec{a}$

Note that the iterative projection approach can be extended to an L1 minimization algorithm by using iteratively reweighted least-squares (IRLS) framework [28] [33].

B. Transpositive Dissimilarity

The Floyd-Warshall algorithm can be used to compute the minimum transitive dissimilarity. In this paper, we modified the Floyd-Warshall algorithm to fit the original dissimilarity matrix to a ultra-matrix and at the meantime to incorporate the given relative constraints. Algorithm 2 shows the algorithm procedure to incorporate the relative constraints into the ultra-metric transformation process. Its difference from standard Floyd-Warshall algorithm is that the updated value for m_{ij} is not only determined by the pairwise dissimilarities related to x_i and x_j, but also restricted by any constraints specifying merge orders about them (see Lines 4-7).

VI. Experiments

In this section, we conduct experiments on various datasets to evaluate our proposed semi-supervised hierarchical clustering framework. We compare our proposed techniques in Section V (the iterative projection algorithm (IPoptim) and the transitive dissimilarity transformation algorithm (UltraTran)) with two baseline algorithms: the standard agglomerative hierarchical clustering(HAC) without constraints and the constraint-based HAC (denoted as HACoc) proposed in [18].

Algorithm 2 Modified Floyd-Warshall algorithm to compute the minimum transitive dissimilarity of weighted graph G

Input: G: Pair-wise distance matrix of data set.
C: Merge order constraints.
Output: M: Minimum Transitive dissimilarity matrix closure of G.
Init: M = G.

1: **for** $k \leftarrow 0$ **to** N **do**
2: **for** $i \leftarrow 0$ **to** N **do**
3: **for** $j \leftarrow 0$ **to** N **do**
4: **for all** $c = (x_i, x_j, x_l)$ **do**
5: $minCon = \min(minCon, d(x_i, x_l))$
6: **end for**
7: $m_{ij} = \min\{m_{ij}, \max(m_{ik}, m_{kj}), minCon\}$
8: **end for**
9: **end for**
10: **end for**
11: **return** M

A. Dataset Description

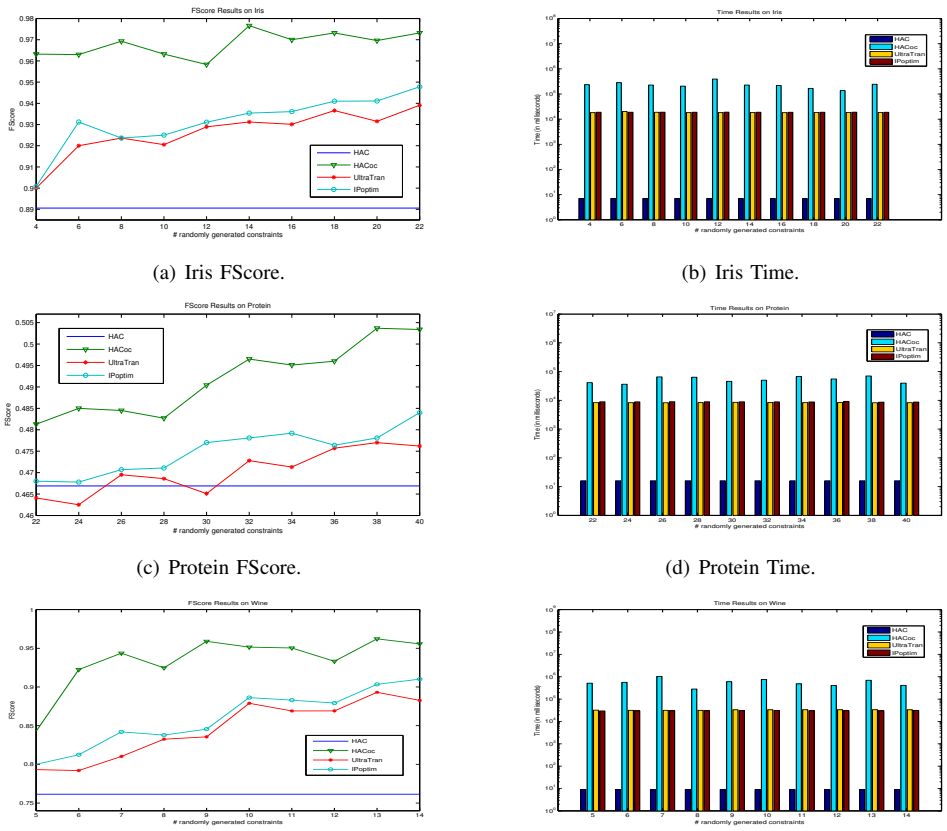
Name	# samples	# attributes	# classes
Iris	150	4	3
Wine	178	13	3
Protein	116	20	6
Ionosphere	351	34	2
CSTR	475	1000	4
Log	1367	1000	8
WebACE	2340	1000	20
Reuters	2900	1000	10

Table III
DATASET DESCRIPTIONS

Table III shows the summary of the datasets used in the experiments. We use 8 datasets with the number of classes ranges from 2 to 20, the number of samples ranges from 116 to 2900 and the number of dimensions ranges from 4 to 1000. The details of the datasets are: (1) Four datasets (Ionosphere, Iris, Protein and Wine) are from UCI data repository [34]. (2) Four datasets (CSTR, Log, Reuters, WebACE) are benchmark text datasets for document clustering. Each document is represented as a term vector using vector space model. All document datasets are pre-processed by removing the stop words and unnecessary tags and headers. More information of the datasets can be found in [35].

B. Evaluation Measures

All the eight datasets have data labels which will be used in clustering performance evaluation. The accuracy of a hierarchical clustering is evaluated by considering the entire hierarchy [16]. A single cut on the hierarchy produces a possible partition of the data set and such partition can be measured by FScore proposed in [36]. Supposing G_i is one of the clusterings generated by cutting on the hierarchy H



(a) Iris FScore.

(b) Iris Time.

(c) Protein FScore.

(d) Protein Time.

(e) Wine FScore.

(f) Wine Time.

Figure 5. Results on Iris, Protein, and Wine datasets. The performance as a function of the number of constraints.

and D_j is a group of data sharing the same label over L classes, then

$$FScore(G_i, D_j) = \frac{2 * Recall(G_i, L_j) * Precision(G_i, L_j)}{Recall(G_i, L_j) + Precision(G_i, L_j)}. \quad (10)$$

The FScore of group G_i is defined as the maximum FScore over all L classes

$$FScore(G_i) = \max_{j \in L} FScore(G_i, D_j). \quad (11)$$

For a hierarchical clustering with $|D|$ samples. Totally $N = \frac{(1+|D|)*|D|}{2}$ possible groups can be generated by cutting at different levels. The FScore defined on the entire hierarchy is computed as the weighted sum of each group's FScore:

$$FScore(H) = \sum_{i=1}^{N} \frac{|G_i|}{|D|} FScore(G_i). \quad (12)$$

We also compare the running time of different algorithms. The running time is recorded at milliseconds (1/1000s).

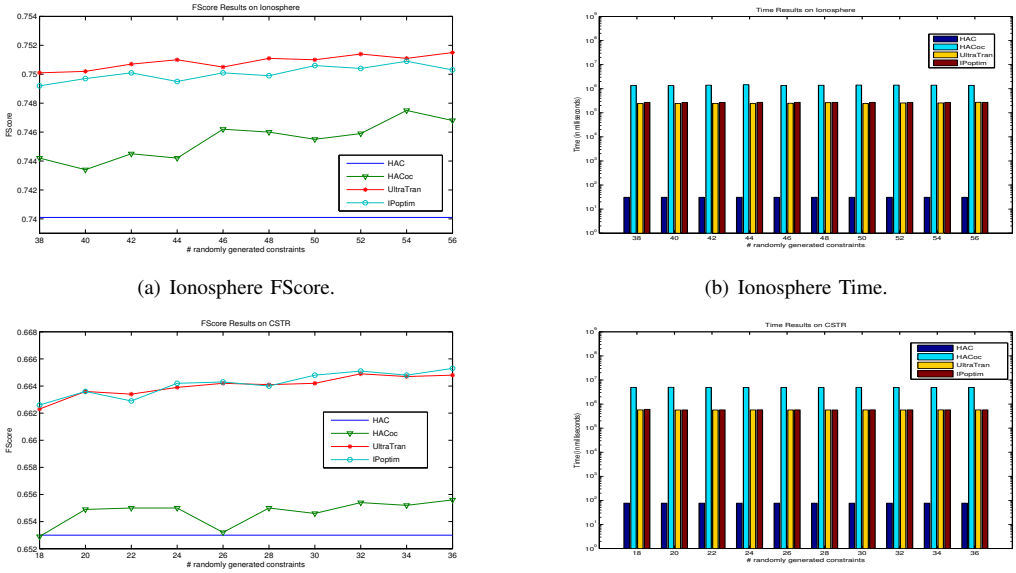
C. Experiment Setup

According to the given class label of each sample, we randomly select three samples from two different classes to generate a constraint. For example, if $x_i, x_j \in Class1$ and $x_k \in Class2$, then $c = (x_i, x_j, x_k)$ is a relative constraint. So each generated constraint is based on the actual class label information and should reflect the domain knowledge. As a result, we can expect the clustering performance should be generally improved when these constraints are utilized. In our experiments, the reported results are computed by averaging 10 runs. For the first five small datasets (the number of samples $\leq$ 1000), we randomly generate 100 constraints for each run. For the other three large datasets, we randomly generate 200 constraints for each run. All constraint sets are preprocessed to eliminate the conflicts. The experiments are conducted under the environment of Linux 2.6 plus 8 Intel(R) Xeon(R) CPU E5420 2.50GHz and 16 GB of RAM.

D. Result Analysis

The experimental results are shown in Table IV. Note that the running time of HACoc is much longer than the other algorithms, especially on large datasets. So we do not include the results of HACoc on Log, WebACE and Reuter datasets for comparison. From Table IV, we observe that:

- By incorporating relative constraints, semi-supervised



(a) Ionosphere FScore.

(b) Ionosphere Time.

(c) CSTR FScore.

(d) CSTR Time.

Figure 6. Results on Ionosphere and CSTR datasets. The performance as a function of the number of constraints.

dataset	Algorithm	FScore	Time
Iris	HAC	0.8906	107
	HACoc	**0.96**	233694
	IPoptim	0.9293	18917
	UltraTran	0.9211	**18490**
Wine	HAC	0.7614	109
	HACoc	**0.9346**	573002
	IPoptim	0.86	**30034**
	UltraTran	0.8456	32636
Protein	HAC	0.4669	196
	HACoc	**0.5131**	53580
	IPoptim	0.4730	8889
	UltraTran	0.4669	**8342**
Ionosphere	HAC	0.7401	361
	HACoc	0.7446	1392259
	IPoptim	**0.7503**	270198
	UltraTran	0.7501	**251164**
CSTR	HAC	0.653	784
	HACoc	0.6524	4911106
	IPoptim	**0.6632**	577451
	UltraTran	0.6631	**570320**
Log	HAC	0.8871	3255
	IPoptim	**0.9001**	1.984e+7
	UltraTran	0.8973	**1.9698e+7**
WebACE	HAC	0.5471	19580
	IPoptim	0.5492	**1.0081e+8**
	UltraTran	**0.5514**	1.0090e+8
Reuter	HAC	0.6154	33000
	IPoptim	**0.6187**	**1.7682e+8**
	UltraTran	0.6178	1.7694e+8

Table IV
PERFORMANCE COMPARISON ON 8 DATASETS.

hierarchical clustering outperforms hierarchical clustering without constraints. In all datasets, the Fscore values of HAC are consistently lower than those of other semi-supervised hierarchical clustering frameworks with constraints. The performance improvement is significant on Iris, Wine and CSTR as shown in Figure 5.

- Although HACoc achieves the best clustering performance on three small datasets (Iris, Wine, Protein), it is not efficient and needs long execution time.
- Our proposed techniques (IPoptim and UltraTran) are much more efficient than HACoc. In terms of clustering performance, IPoptim and UltraTran outperform HACoc on Ionosphere and CSTR datasets as shown in Figure 6.
- In general, IPoptim outperforms UltraTran in clustering performance while UltraTran is more efficient than IPoptim.

To further investigate the performance of semi-supervised hierarchical clustering, we conduct experiments by varying the number of relative constraints. Figure 5, Figure 6 and Figure 7 plot the clustering performance and execution time as a function of the number of constraints on six datasets (Iris, Protein, Wine, Ionosphere, CSTR, and Reuter). Note that the computation time of the algorithms does not increase much as the number of constraints increases. We also observe that the performance enhancement obtained by the semi-supervised clustering is generally greater as the number of constraints increases. However, the performance is not monotonically increasing with the number of constraints. There are two possible reasons. First, our proposed framework is not aiming to satisfy all constraints but to find a good approximation of the constrained ultra-metric. Second, the clustering performance is also depending on the quality

of generated constraints. It is intuitive to say that not all constraints have the same importance to the performance of semi-supervised hierarchical clustering. And the constraints we applied are directly generated from the instance similarities and the true class labels. How to discover important constraints would be a valuable consideration in our future work.

VII. Conclusion

In this paper we propose a semi-supervised hierarchical clustering framework based on ultra-metric dendrogram distance. The triple-wise relative constraints are introduced, particularly for hierarchial clustering, to describe the merge preference among instances. Two techniques are developed to solve semi-supervised hierarchical clustering problem. The optimization-based technique minimizes the distance between the original dissimilarity matrix and the target ultra-matrix using the ultra-metricity and relative constraints. The transitive dissimilarity based technique takes those relative constraints into the ultra-metric transformation process. Experiments are conducted to demonstrate the effectiveness and efficiency of our proposed methods.

Acknowledgment

The work is partially supported by NSF grants DBI-0850203 and HRD-0833093 and DHS grants 2009-ST-062-000016 and 2010-ST-062-000039.

References

[1] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc.

[2] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, 2nd ed. Morgan Kaufmann, 2006.

[3] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*, 1st ed. Addison Wesley, May.

[4] I. Davidson and S. S. Ravi, "Clustering with constraints: Feasibility issues and the k-means algorithm," 2005.

[5] Z. Lu and T. K. Leen, "Semi-supervised learning with penalized probabilistic clustering," in *Advances in Neural Information Processing Systems 17*. MIT Press, 2005, pp. 849–856.

[6] A. Bar-hillel, T. Hertz, N. Shental, and D. Weinshall, "Learning a mahalanobis metric from equivalence constraints," *Journal of Machine Learning Research*, vol. 6, pp. 937–965, 2005.

[7] S. Basu, M. Bilenko, and R. J. Mooney, "A probabilistic framework for semi-supervised clustering," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '04, 2004, pp. 59–68.

[8] D. Klein, S. Kamvar, and C. Manning, "From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering."

[9] K. L. Wagstaff, "Intelligent clustering with instance-level constraints," Ph.D. dissertation, Ithaca, NY, USA, 2002, aAI3059148.

[10] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell, "Distance metric learning, with application to clustering with side-information," in *Advances in Neural Information Processing Systems 15*. MIT Press, 2002, pp. 505–512.

[11] X. Zhu, "Semi-supervised learning literature survey," Computer Sciences, University of Wisconsin-Madison, Tech. Rep. 1530, 2005.

[12] K. Bade and A. Nurnberger, "Personalized hierarchical clustering," in *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, ser. WI '06, 2006, pp. 181–187.

[13] K. Bade and A. Nrnberger, "Creating a cluster hierarchy under constraints of a partially known hierarchy." in *SDM'08*, 2008, pp. 13–24.

[14] "Simulation of random dendrograms and comparison tests: some comments," vol. 17.

[15] H. Yuan, G. Chen, J. Wu, and H. Xiong, "Towards controlling virus propagation in information systems with point-to-group information sharing," *Decision Support Systems*, vol. 48, no. 1, pp. 57–68, 2009.

[16] Y. Zhao and G. Karypis, "Evaluation of hierarchical clustering algorithms for document datasets," in *CIKM*, 2002, pp. 515–524.

[17] W. Pedrycz, "Fuzzy clustering with a knowledge-based guidance," *Pattern Recognition Letters*, vol. 25, pp. 469–480, 2004.

[18] H. Zhao and Z. Qi, "Hierarchical agglomerative clustering with ordering constraints," in *WKDD*, 2010, pp. 195–199.

[19] N. Ailon and M. Charikar, "Fitting tree metrics: Hierarchical clustering and phylogeny," in *In Proceedings of the Symposium on Foundations of Computer Science*, 2005, pp. 73–82.

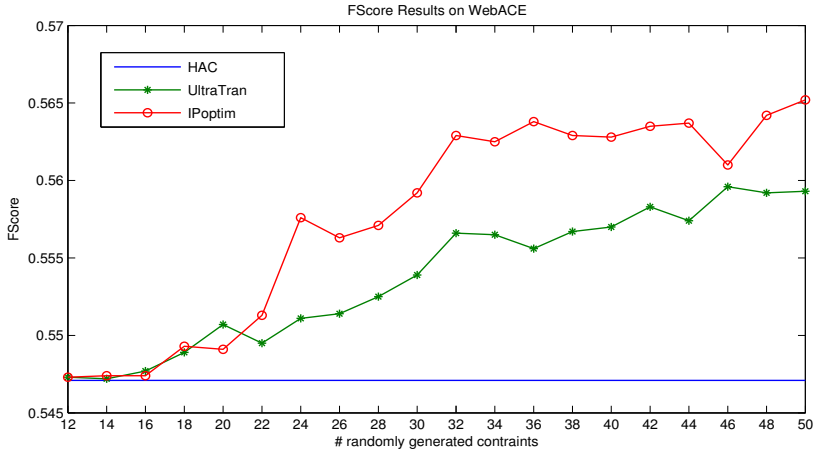
[20] R. Agarwala, V. Bafna, M. Farach, B. Narayanan, M. Paterson, and M. Thorup, "On the approximability of numerical taxonomy (fitting distances by tree metrics)," 1995.

[21] P. Hansen and B. Jaumard, "Cluster analysis and mathematical programming," *Mathematical Programming*, vol. 79, pp. 191–215.

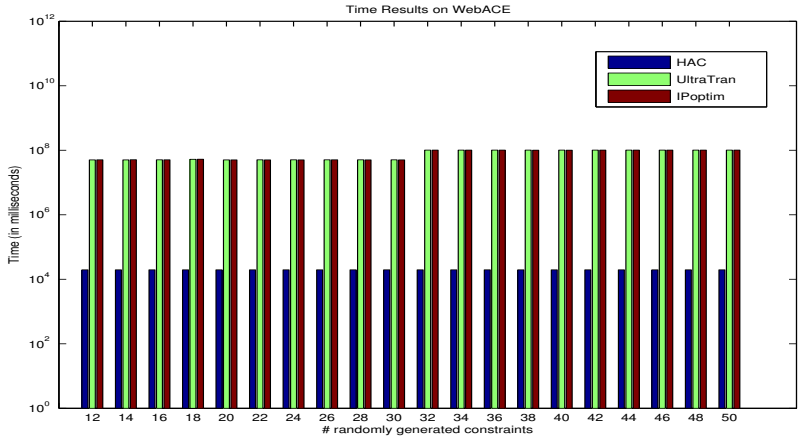
[22] M. Krivanek and J. Moravek, "Np-hard problems in hierarchical-tree clustering," *Acta Informatica*, no. 3, pp. 311–323.

[23] S. Sattath and A. Tversky, "Additive similarity trees," *Psychometrika*, no. 3, pp. 319–345.

[24] L. Hubert, P. Arabie, and J. Meulman, *Combinatorial data analysis: optimization by dynamic programming*, ser. SIAM monographs on discrete mathematics and applications.



(a) WebACE FScore.



(b) WebACE Time.

Figure 7. Results on WebACE dataset. The performance as a function of the number of constraints.

[25] G. D. Soete, "A least squares algorithm for fitting an ultrametric tree to a dissimilarity matrix," *Pattern Recognition Letters*, 1986.

[26] T. L. Li Zheng and C. Ding, "Hierarchical Ensemble Clustering," pp. 1199 – 1204, 2010.

[27] L. Hubert and P. Arabie, "Iterative projection strategies for the least-squares fitting of tree structures to proximity data," *Br J Math Stat Psychol*, vol. 48, pp. 281–317, 1995.

[28] S. T. J., "L1 Optimization under Linear Inequality Constraints," *Journal of classification*, vol. 17, pp. 225–242, 2000.

[29] G. Soete, "Ultrametric tree representations of incomplete dissimilarity data," *Journal of Classification*, vol. 1, no. 1, pp. 235–242, December 1984.

[30] S. T. J., "Constructing ultrametric and additive trees based on the L1 norm," *Journal of classification*, vol. 18, pp. 185–207, 2001.

[31] R. L. Dykstra, "An algorithm for restricted least squares regression," 1983.

[32] H. W. Kuhn and A. W. Tucker, "Nonlinear programming," in *Proceedings of the 2nd Berkeley Symposium on Mathematical Statistics and Probability*, J. Neyman, Ed. University of California Press, Berkeley, CA, USA, 1950, pp. 481–492.

[33] R. D. B., "Iteratively Reweighted Least Squares," *Encyclopedia of Statistical Sciences*, pp. 272–275, 1983.

[34] C. L. Blake and C. J. Merz, "UCI repository of machine learning databases," 1998.

[35] T. Li and C. Ding, "Weighted consensus clustering," in *Proceedings of 2008 SIAM International Conference on Data Mining*, ser. SDM 2008, 2008, pp. 798–809.

[36] B. Larsen and C. Aone, "Fast and effective text mining using linear-time document clustering," in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '99, 1999, pp. 16–22.