PoClustering: Lossless Clustering of Dissimilarity Data

Jinze Liu, Qi Zhang, Wei Wang, Leonard McMillan, Jan Prins
Department of Computer Science
University of North Carolina, Chapel Hill, NC 27599
{liuj, zhangq, weiwang, mcmillan, prins}@cs.unc.edu

Abstract

Given a set of objects V with a dissimilarity measure between pairs of objects in V, a *PoCluster* is a collection of sets $P \subset powerset(V)$ partially ordered by the $\subset$ relation such that $S \subset T$ iff the maximal dissimilarity among objects in S is less than the maximal dissimilarity among objects in T. PoClusters capture categorizations of objects that are not strictly hierarchical, such as those found in ontologies. PoClusters can not, in general, be constructed using hierarchical clustering algorithms. In this paper, we examine the relationship between PoClusters and dissimilarity matrices and prove that PoClusters are in one-to-one correspondence with the set of dissimilarity matrices. The PoClustering problem is NP-Complete, and we present a heuristic algorithm for it in this paper. Experiments on both synthetic and real datasets demonstrate the quality and scalability of the algorithms.

1 Introduction

Categorizations are natural ways to organize a set of objects. The structure of categorization ranges from hierarchies(taxonomies), where subclasses are disjoint partitions of their parent class, to ontologies, which allow overlapping subclasses as well as multiple parents. Unsupervised clustering is an important classification approach. Deriving hierarchical classifications by clustering pair-wise dissimilarity data has been studied extensively. The problem is referred to as *numerical taxonomy*[1]. Numerical taxonomies are useful in a number of applications, such as estimating evolutionary branching processes in biology. Since strict taxonomies form disjoint partitions, these structures are insufficient for capturing categorizations with richer relationships, such as ontologies.

In this paper, we consider the problem of automatically constructing *numerical ontologies* by clustering dissimilarities between object pairs from a given set. Numerical ontologies provide a more general categorization approach than taxonomies, and their added categorization power may benefit applications in multiple disciplines. For example, in biology, a gene may be in-volved in multiple pathways due to a common biological mechanism called trans-regulation. Building a gene hierarchy from observed pairwise dissimilarities forces each gene to one specific function. As a result, the hierarchical classification is largely inconsistent with existing gene function classifications, such as Gene Ontology, where gene subclasses may overlap or belong to multiple parents.

Before proceeding, we clarify the specific classification notion assumed in this paper. We consider the most general classification system, which is a partially ordered set, or poset. A poset contains the sets(clusters) of objects as the elements, ordered according to their subset relationships. Since a given poset can be constructed from any combination of subsets taken from the set's power set, the set of posets has a cardinality of $2^{2^{|\mathcal{N}|}}$, where $\mathcal{N}$ is the object set. The set of hierarchical clusters, for example, is a special subset of the set of posets.

It has been proven that any given ultrametric dissimilarity matrix corresponds to a unique hierarchy[1]. A dissimilarity matrix D is ultrametric, if for any three objects A, B and C in the set, $D(A, C) \leq \max(D(A, B), D(B, C))$. The correspondence means that the same dissimilarity matrix can be recovered from the hierarchy. For example, Figure 1 (a.1) shows an ultrametric dissimilarity matrix. A hierarchy shown in (a.2) is constructed from it by hierarchical clustering with a complete linkage criterion. Let the diameter of a cluster be the maximum pair-wise dissimilarity within the cluster. The pair-wise dissimilarity between any pair of objects shown in (a.1) can be recovered by assigning it the minimum diameter of the clusters containing the pair. On the other hand, the dissimilarity matrix shown in (b.1) is not an ultrametric dissimilarity matrix, because for objects A, B and C, $D(A, C) > \max(D(A, B), D(B, C))$. Applying the same clustering algorithm to this dissimilarity matrix generates the same hierarchy shown in (a.2). But the dissimilarities in (b.1) cannot be derived from the hierarchy, which corresponds to the ultrametric dissimilarities in (a.1). Therefore, building a hierarchy from a
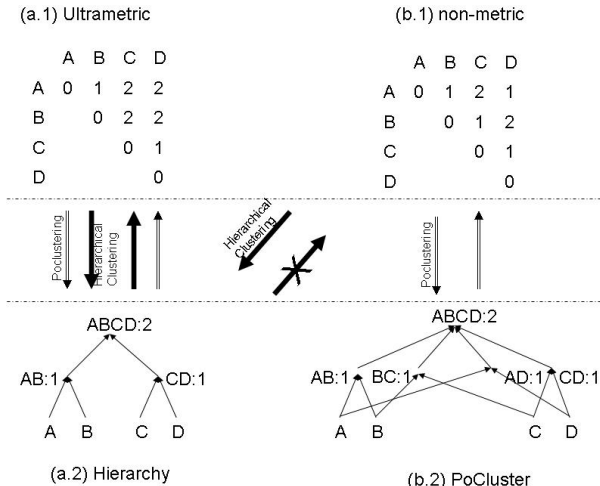
Figure 1: (a.1) An ultrametric dissimilarity matrix; (a.2) Hierarchy constructed from (a.1) by either hierarchical clustering or PoClustering; (b.1) A non-metric dissimilarity matrix. (b.2) PoCluster constructed from (b.1) by PoClustering. Note: (a.1) can be derived from the hierarchy in (a.2) by assigning each pair the minimum diameter of the sets containing it; (b.2) can be used to derive dissimilarities of (b.1) in the same way; Applying hierarchical clustering to (b.1) can also construct the hierarchy in (a.2), but (b.1) cannot be derived from (a.2)

dissimilarity matrix that does not satisfy the ultrametric property potentially loses information. The problem of interest in this paper is *whether there exists a clustering approach that preserves the information of any given dissimilarity data?*

PoClustering has been proposed by Liu *et.al.* in [11]. A *PoCluster* is a collection of *clique clusters* arrived at by smoothly varying the threshold from 0 to the maximum pair-wise dissimilarity in the set. It adopts a definition of the cluster as a maximal clique from graph theory. A *clique cluster* is a maximal subset of objects whose maximum pair-wise dissimilarity does not exceed a given threshold. An example of PoCluster is shown in Figure 1 (b.2), which is generated from dissimilarity matrix (b.1) by PoClustering. PoClusters differ from hierarchies by incorporating all clique clusters rather than only disjoint clusters. As a result, it allows overlaps between clusters that are not strict subsets, as shown in Figure 1 (b.2). In addition, it preserves the information provided in the dissimilarity data. The dissimilarity matrix shown in Figure 1 (b.1), which could not be recovered from the hierarchy in (a.2), can be derived from the PoCluster in (b.2). In this paper, we formally prove that, there exists a one-to-one correspondence between the set of PoClusters and the set of dissimilarity matrices. The set of PoClusters is, therefore, the most

general notion of clustering that can be derived from dissimilarities alone. In addition, we prove that the set of PoClusters contains all possible pyramidal[6] and hierarchical clusters as special instances.

PoClustering algorithm was presented in [11]. However, the PoClustering problem is NP-complete. In this paper, we present a greedy approximation algorithm for PoClustering which replaces maximum clique finding in the original algorithm by solving a minimum edge clique cover problem. Our experiments on both synthetic and real data show the effectiveness and efficiency of this approximation algorithm in comparison to the conventional hierarchical and pyramidal clustering algorithms.

The remainder of this paper is organized as follows. Section 2 addresses related work in clustering, automated taxonomy construction, and dissimilarity measures appropriate for ontologies. Section 3 presents the preliminary definitions of PoClusters, followed by Section 4 which examines their properties and the relationships with existing clustering algorithms. Section 5 provides an approximation algorithm for constructing PoClusters from dissimilarity data. A performance study is reported in Section 6. Section 7 concludes the paper and discusses future work.

2 Related Work

Many clustering algorithms take a dissimilarity matrix as input. However, relatively few investigations have been conducted to establish the relationship between the dissimilarity matrix input and the clustering results. In this section, we review previous studies on clustering algorithms that have known relationships to special classes of dissimilarity matrices.

Both hierarchical[3, 7] and pyramidal clustering [4, 6] generate clusters that have bijections to special sub-classes of dissimilarity matrices.

Hierarchical clustering[3, 7] refers to the formation of a nested disjoint partition of data objects. It is often represented by a *dendrogram*, that is, a tree with the objects at its leaves and a root corresponding to the universal set (of all objects). The heights of the internal nodes represent the maximum dissimilarities between the descendant leaves. It has been proven that a bijection exists between hierarchical clustering and the set of *ultrametric*[6] dissimilarity matrices which satisfy the *ultrametric triangle inequality*, i.e., for any set of three objects $\{a, b, c\}$, $D(a, c) \leq max\{D(a, b), D(b, c)\}$. An equivalent statement of the ultrametric condition is that there exists a linear order of all objects such that their dissimilarities are the distances between them.

Pyramidal clustering[4, 6] allows for a more general model than hierarchical clustering. A child cluster may have up to two parent clusters. Two clusters may overlap by sharing a common child cluster. The
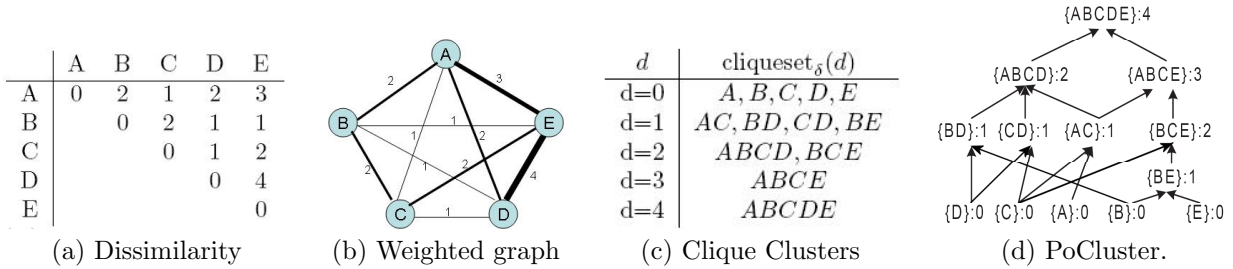
	A	B	C	D	E
A	0	2	1	2	3
B		0	2	1	1
C			0	1	2
D				0	4
E					0

(a) Dissimilarity

(b) Weighted graph

d	cliqueset$_\delta(d)$
d=0	A, B, C, D, E
d=1	AC, BD, CD, BE
d=2	$ABCD, BCE$
d=3	$ABCE$
d=4	$ABCDE$

(c) Clique Clusters

(d) PoCluster.

Figure 2: A running example. (a) shows a dissimilarity matrix of 5 objects {A, B, C, D, E}; (b) shows an undirected weighted graph implied by (c); Table (c) contains the list of clique clusters with all diameters; (d) shows a PoCluster which contains 13 clusters and their subset relationships (Each cluster in the PoCluster represents a clique cluster with its diameter in (c). The PoCluster is organized in DAG with subset relationship between the nodes. There is a directed path from node S_1 to S_2 if $S_1 \subset S_2$). Note: Applying PoClustering algorithm can construct PoCluster shown in (d) given dissimilarity matrix (a).

structure can be represented by a directed acyclic graph. It is known that a bijection exists between pyramidal clustering and the set of dissimilarity matrices that are Robinson matrices. A matrix is a *Robinson matrix* if there exists an ordering among all objects such that the dissimilarities in the rows and columns do not decrease when moving horizontally or vertically away from the main diagonal. An ultrametric matrix is a special case of Robinson matrix and hierarchical clustering is a special case of pyramidal clustering. Note that a dissimilarity matrix may not always be a Robinson matrix, and in such cases, neither hierarchical clustering nor pyramidal clustering is able to generate clustering from which the original dissimilarity matrix can be re-derived. That is, no bijection exists.

We prove in this paper, that PoClustering preserves the bijection between PoCluster and a given dissimilarity matrix. It also includes both hierarchical clustering and pyramidal clustering as special cases.

3 Preliminaries on Pocluster

In the following discussion, we assume a universal set of objects denoted by $\mathcal{N}$. A *pair* in $\mathcal{N}$ refers to an object pair $\{x, y\}$, where $x, y \in \mathcal{N}$. Given a set $S \subseteq \mathcal{N}$, the set of all pairs in S is denoted by $S \times S$. A *dissimilarity matrix* describes the pair-wise relationships between objects. A dissimilarity matrix can be directly mapped to an undirected weighted graph $G = \langle V, E, W \rangle$, where each node in V corresponds to an object in $\mathcal{N}$, and each edge $e = \langle x, y \rangle$ with weight w denotes that the dissimilarity $D(x, y)$ between the two objects x and y is w. The graph implied by the dissimilarity D is denoted as $G(D)$.

A fully connected subgraph in the weighted graph G is called a clique. The *diameter* of a clique is the maximum edge weight of the clique. Given $G(D)$, a *clique cluster* $C = \langle S, d \rangle$ is defined as a maximal clique S with diameter d. When there are multiple cliques within the graph with the same diameter d, we denote this set of clique clusters as cliqueset$_\delta(d)$.

Next, we present a brief overview of the definition of PoCluster proposed by Liu *et.al.* in paper [11]. Let D be a dissimilarity matrix, and let $W(D)$ be the set of diameters indicated by D. A *PoCluster* P of D is defined as

$$(3.1) \qquad P = \bigcup_{\forall d \in W(D)} \text{cliqueset}_\delta(d).$$

which is the collection of clique clusters of all possible diameters in W of $G(D)$.

For example, the dissimilarity matrix in Figure 2(a) consists of 4 different dissimilarities and therefore, 4 possible diameters, $\{1, 2, 3, 4\}$. The set of the clique clusters generated for each diameter threshold is shown in Figure 2(c). The PoCluster (subset) relationships between these clique clusters are shown in Figure 2(d).

4 Properties of PoCluster

The basic properties of PoClusters appear in [11]. Similar to hierarchical clustering, PoClustering also includes the universal set, that has the maximum dissimilarity in D as its diameter, and singletons, which have the minimum dissimilarity(0). PoClustering does not ignore dissimilarity measures like hierarchical clustering, since each pair-wise dissimilarity is covered by at least one clique cluster whose diameter equals the pair-wise dissimilarity. In addition, the maximal clique cluster insures that if one cluster is a subset of another, its diameter is strictly lower.

We further examine the properties of PoCluster with regard to dissimilarity matrices. Although it is possible to run classical hierarchal clustering or pyramidal clustering on any dissimilarity matrix, this mapping is not, in general, invertible. However, PoClus-

tering goes beyond them by generating a PoCluster, which provides a one-to-one correspondence with the input dissimilarity matrix. This property is presented and proven in Theorem 4.1.

THEOREM 4.1. *There exists a bijection between the set of PoClusters $\mathcal{P}$ and the set of dissimilarity matrices $\mathcal{D}$.*

Proof of Theorem 4.1 can be found in technical report[12].

Hierarchical and pyramidal clusterings are known to have one-to-one correspondences with ultrametric and Robinson matrices respectively. In addition, hierarchical clustering is a special case of the pyramidal clustering as shown in [6]. We answer a similar question, i.e., *Does PoCluster recover the same hierarchy or pyramid as pyramidal clustering given an ultrametric or a Robinson matrix?*
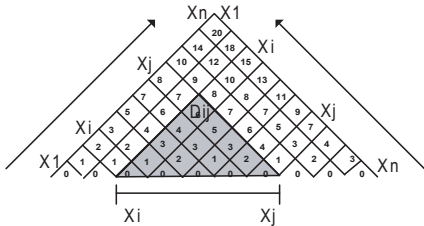

Figure 3: The structure of Robinson matrix. There exists a linear ordering of the objects, such that the dissimilarities never decrease as they move away from diagonal along both row and column.

LEMMA 4.1. *The set of pyramids is included in the set of PoCluster.*

Proof. (Sketch of proof) As shown by Diday[6], there exists a bijection between a pyramid and a Robinson matrix. A matrix is a Robinson matrix if there exists an ordering θ of the objects, such that the rows and columns are in non-decreasing order as they are moving away from the diagonal. Any given pyramid cluster is an interval of such an ordering. Now let $\{x_1, x_2, ..., x_n\}$ be an ordered list of objects according to θ. Let D be the Robinson dissimilarity matrix shown in Figure 3. Let P be a PoCluster. For any entry $\{x_i, x_j\}$ in a Robinson matrix D, the sub-triangle below $\{x_i, x_j\}$ above the main diagonal corresponding to the rows $\langle x_i, ..., x_j \rangle$ and the columns $\langle x_i, ..., x_j \rangle$ will contain lesser dissimilarity values than $D(x_i, x_j)$, hence, $\{x_i, ..., x_j\}$ will be a cluster in P, which is an interval of θ. Therefore, starting from the entries from the upper right corner, then recursively traversing the two lower sub-triangles $\langle x_i, ..., x_{j-1} \rangle$ and $\langle x_{i+1}, ..., x_j \rangle$ returns the whole pyramid, which is also a PoCluster.

5 PoClustering Algorithm

Given a dissimilarity matrix D, the corresponding PoCluster P (i.e., $P = \{\text{cliqueset}_\delta(d) | \forall d \in W(D)\}$) can be found by repeating a simple procedure. In the naive algorithm, one needs only to find all cliques in a subgraph of $G(D)$ that includes only those edges corresponding to the pair-wise dissimilarities less than or equal to a threshold d as the threshold varies from the smallest to the largest dissimilarity in D.

An incremental and exact PoClustering algorithm was proposed by Liu *et.al.* in [11]. The algorithm only computes clique clusters that are affected by the introduction of new edges. The algorithm maintains a pool of all clusters in the previous graph. Given the next graph with more edges, the pool of cliques can be updated as follows: First, all the cliques in the pool that share a vertex with the new edges are found. Second, if a clique in the pool can be extended by adding one or more of the new edges, the extended clique is added to the pool, and the cliques that are subgraphs of the new clique are removed. The parent-child relationships can be established between new cliques and removed cliques.

Though more efficient, the exact algorithm still bears the NP-Complete complexity of the PoClustering problem. Although theoretically the number of clusters in a PoCluster can be exponential in the number of objects[1], most real classifications contain only *polynomial* number of clusters with respect to the number of objects. In this section, we present a heuristic algorithm for the construction of approximate PoClusters. This approach addresses both the NP-completeness of finding all cliques and the huge number of clique clusters generated.

Instead of searching for all clique clusters in a graph, we approximate them with a minimum set of cliques that covers all the edges in the graph, i.e, a minimum edge clique cover(ECC). The minimum number of clique clusters covering all edges are a subset of all clique clusters.

DEFINITION 5.1. *Given a graph $G =< V, E >$, an* **edge clique cover***(ECC) R of G is defined as a set of maximal cliques induced by G, such that for any $e \in E$, there exists a maximal clique $c \in R$, where e is an edge in c.*

We describe an incremental greedy algorithm to approximate the minimum edge clique cover in order to construct the poset. Algorithm 1 is similar to the original algorithm proposed in [11] in detecting the new cliques. The difference is that at each step, it only keeps a minimum number of cliques that cover all the

---

[1]The number of clique clusters in an undirected graph with n nodes is $o(3^{1/3*n})$.

Algorithm 1 $R_{new} = \text{gen\_ECC}(G, R, e)$
Input: G: graph; R: an edge clique cover does not cover e, $e = \{x, y\}$: a new edge
Output: R_{new}: a new clique cover covering $E(G) \cup \{e\}$

1: $C_{x,y} \leftarrow maximal\{c_1 \cap c_2 | c_1 \in \pi(x), c_2 \in \pi(y)\}$.
2: $R \leftarrow \{R \cup C_{x,y}\}$
3: $R_{new} \leftarrow \text{argmin}_{|R'|}\{R'|R' \subseteq R, R \text{ covers } E(G) \cup \{e\}\}$
4: return R_{new}

edges. Given an input clique cover from time t and an inserted edge $\{x, y\}$, the algorithm first goes through each clique in the clique cover that is connected with one of the edges, let's say, x, and let $\pi(x)$ be the set of maximal cliques containing x. The algorithm then looks for the maximum sets of points in $\pi(x)$ that are also connected to y. The subgraph, though complete, might not be maximal. The algorithm then further extends it into a maximal clique. In the end, the algorithm goes through the current list of cliques sequentially, and removes redundant cliques whose edges have already been covered.

The total number of clique covers in the graph is bounded by the number of edges since, at most, only one clique will be added into the cover, at each step an edge is inserted into the graph. In practice, the number is considerably smaller than the number of edges.

The number of outer loop iterations of the algorithm is determined by the number of edges, which is $O(n^2)$. For a new clique, the greedy algorithm goes through the cliques in the existing clique cover, and find the maximal clique cover. Assume the number of cliques in the cover is k, creating a maximal clique takes $O(kn)$. Checking the coverage of edges by the cliques also takes $O(kn^2)$ time.

6 Experiments

We have experimented with both synthetic comparing hierarchical clustering (Hierarchy), pyramidal clustering (Pyramid), and edge clique cover (ECC). We do not present the results of the exact algorithm because of its prohibitively long running time. The results of real dataset experiments are reported in [12].

6.1 Evaluation Criteria The *match* score is used to measure the approximation of the recovered poset to the original poset. We take each poset as a set of sets. Given two PoClusters P_1 and P_2, the match score of P_2 to P_1 is computed as:

$$match(P_1, P_2) = mean_{s_1 \in P_1}(max_{s_2 \in P_2}(\frac{|s_1 \cap s_2|}{|s_1 \cup s_2|}))$$

For each set in P_1, only the best match in P_2 is

taken account of by Jaccard coefficient. The overall match score is determined by the mean match score of the whole sets.
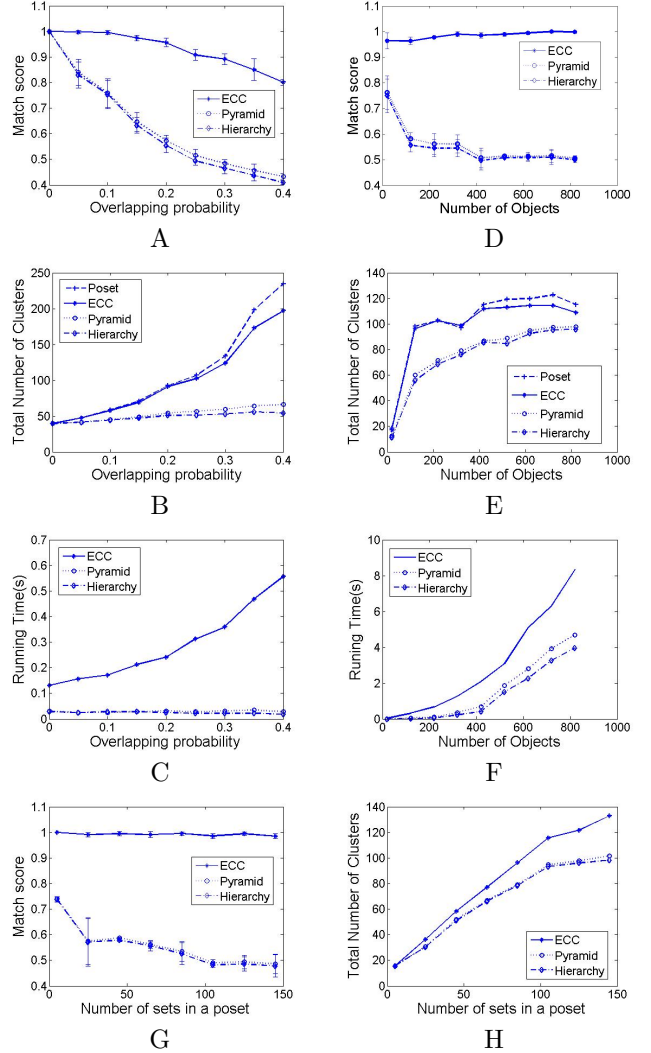

Figure 4: Experimentation on synthetic data

6.2 Synthetic Data We created a synthetic poset generator that is controlled by three parameters. They are the number of objects of the universal set N, the total number of clusters (sets), and the overlap probability between clusters. The overlap probability follows a normal distribution with a user-specified mean value and a default standard deviation of $\sqrt{(0.2)}$. It defines the likelihood that an element in a parent cluster appears in more than one child cluster. The synthetic generator of the poset then works as follows: starting from the root node of the poset with n objects, the program recursively generates the child sets that respect the overlap probability distribution. For each

parent set, whether an object should appear in multiple child sets is determined by coin-flipping with overlap probability $p_{overlap}$. After the poset is generated, we computed the rank and assigned the pair-wise distances based on the algorithm presented in [12] We then take the distance matrix as the input to the three clustering algorithms.

We first examined how the overlap probability affects the performance. In this setting, the total number of objects in N is set to 500, and the maximum number of clusters in P is 200. Figure 4(A) shows that overlap probability affects match score of all three algorithms. Starting from 0 overlap, all of the three algorithms have almost 100% match score. However, ECC does a much better job than the other two. The reason is that the number of overlapping clusters increases as the overlap probability increases. The match score drops significantly for hierarchical and pyramidal clustering because of their inability to allow arbitrary overlaps between clusters. Figure 4(B) and (C) show the total number of clusters generated and the running time comparison. Both hierarchical clustering and pyramidal clustering have shorter running time than ECC. It is due to the fact that both algorithms fail to recover most clusters in the original poset. This can be observed from Figure 4(B) where, among the three algorithms, ECC is the only one that is able to recover most clusters in the original poset (its number of clusters is shown as the top curve in Figure 4(B)) .

Our second experiment demonstrated how the number of objects affects the performance. The overlap probability is set to 0.2 and maximum number of clusters is 120. The results are shown in Figure 4 (D), (E), and (F). The match score is close to 1 for ECC but drops below 0.5 for the other two algorithms. Again, the difference in the running time (Figure 4 (F)) between ECC and others is because both hierarchical clustering and pyramidal clustering recover only a small fraction of the sets(clusters) in the original poset (the top curve shown in Figure 4 (E)).

In the last experiment, we fixed the number of objects to 500 and the overlap probability to 0.2, and varied the number of clusters in the poset. The result are shown in Figure 4 (G) and (H). A side effect of increasing the number of clusters defined on a fixed set of objects is an increase in the degree of overlap. Therefore, the match scores of hierarchical clustering and pyramidal clustering fall as the number of clusters increases, whereas ECC maintains a high match score. As shown in Figure 4 (H), ECC is able to recover almost all clusters in the original poset. However, pyramidal clustering and hierarchical clustering generate considerably fewer clusters.

7 Conclusions

PoClusters are a generalization of both hierarchical clustering and pyramid clustering. PoClusters provide both homogeneity within a cluster, as measured by the cluster's diameter as well as separation between clusters. They also handle overlaps in a meaningful way. The formal definition of PoClusters is primarily of theoretical interest, since the problem of computing them exactly is likely to be intractable for large problems. To address this shortcoming we have introduced a polynomially bounded approximation algorithm to automatically generate classification hierarchies.

References

[1] N. Ailon, M. Charikar, Fitting tree metrics: Hierarcical clustering and phylogeny. Proceedings of 46th IEEE Symposium on Fundation of Computer Science, 2005.

[2] Applications of the pyramidal clustering method to biological objects. Comput Chem, 23(3-4):303-15, Jun15, 1999.

[3] P. Berkhin. Survey of clustering data mining techniques https://umdrive.memphis.edu/vphan/public/berkhin-survey.pdf, Accrue Software, 2002.

[4] P. Bertrand and M. F. Janowitz. Pyramids and weak hierarchies in the ordinal model for clustering. Discrete Applied Mathematics, Volume 122, Issues 1-3,Pages 55-81, 15 October 2002

[5] C. Bron and J. Kerbosch, Algorithm 457: Finding all cliques of an undirected graph, Commun. ACM, vol. 16, no. 9, pp. 575-577, 1973.

[6] E. Diday, Orders and overlapping clusters in pyramids. In: J. De Leeuw et al. Multidimensional Data Analysis, DSWO Press, Leiden (1986), pp. 201-234.

[7] A. JAIN and R. Dubes. Algorithms for clustering data. Prentice-Hall, 1988.

[8] R.M. Karp Reducibility among combinatorial problems. Complexity of computer computations, Plenum Press, New YOrk, pp.85-103, 1972.

[9] L. T. Kou , L. J. Stockmeyer , C. K. Wong, Covering edges by cliques with regard to keyword conflicts and intersection graphs, Communications of the ACM, v.21 n.2, p.135-139, Feb. 1978

[10] Y. Linde, A. Buzo, R. Gray. An algorithm for vector quantization design. Proc. IEEE Transactions on Communications, Vol. 28, pp. 84 95, January 1980.

[11] J. Liu, Q. Zhang, W. Wang, L. Mcmillan, J. Prins. Clustering dissimilarity data into partially ordered set. Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, to appear, 2006.

[12] J. Liu, Q. Zhang, W. Wang, L. Mcmillan, J. Prins. Pocluster: loseless clustering of similarity data. UNC-Technical Report, 2007.