Clustering Pair-wise Dissimilarity Data into Partially Ordered Sets

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

ABSTRACT

Ontologies represent data relationships as hierarchies of possibly overlapping classes. Ontologies are closely related to clustering hierarchies, and in this article we explore this relationship in depth. In particular, we examine the space of ontologies that can be generated by pairwise dissimilarity matrices. We demonstrate that classical clustering algorithms, which take dissimilarity matrices as inputs, do not incorporate all available information. In fact, only special types of dissimilarity matrices can be exactly preserved by previous clustering methods. We model ontologies as a partially ordered set (poset) over the subset relation. In this paper, we propose a new clustering algorithm, that generates a partially ordered set of clusters from a dissimilarity matrix.

Categories and Subject Descriptors: H.2.8 [Database Management]: Database Applications - Data Mining
General Terms: Algorithms
Keywords: PoCluster, Poset, Dissimilarity, Clustering

1. INTRODUCTION

Classification hierarchies are natural ways of organizing data. Such hierarchies can range from taxonomies, where all subclasses are disjoint subsets of the parent class, to ontologies that allow arbitrary overlaps between subclasses as well as allowing any subclass to have multiple parents. Typically, classification hierarchies are designed by domain experts. In this article, we address the problem of constructing ontologies automatically by computational means. Moreover, we attempt to derive both categories and their subclass relationships when given only pairwise relationships, dissimilarities, between elements. As a result, we treat ontology construction as a data clustering generalization where the set of objects is grouped into clusters, and the clusters are partially ordered by the subset relation.

Dissimilarity is a common intermediary used by clustering methods to classify data. Applications range from analyzing microarray gene expression levels collected under multiple conditions[15], to analyzing word usage statistics from a corpus of documents[6]. Dissimilarities represent relative pairwise relationships between data objects. Often, they are derived, by various means, from data features. Dissimilarity matrices simplify some of the problems associated with clustering high-dimensional datasets, since their size is only a function of the number of objects ($O(|\mathcal{N}|^2)$), and independent of the objects' dimensions. Many clustering approaches have been developed that take dissimilarities as inputs, and generate hierarchies of clusters. Such hierarchies can be viewed as categorizations or taxonomies if the clusters form a hierarchy of data partitions.

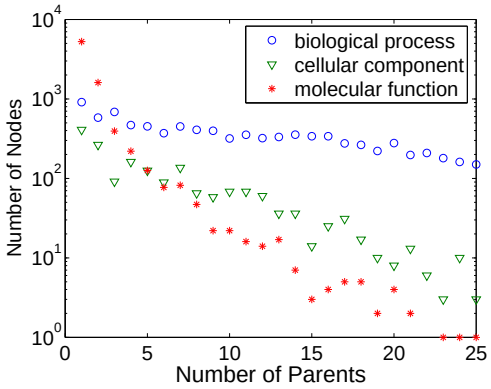


Figure 1: Frequencies of nodes with multiple parents in three GO files (biological process, cellular component, and molecular function).

Classification ontologies are an important tool in biology. Biological ontologies, such as Gene Ontology[1] (GO), are carefully curated and encapsulate both functional knowledge and important relationships between genes. The class-subclass relationships in GO are neither a simple tree, nor a lattice structure. Instead, it is a directed acyclic graph, where any child can have multiple parents. The frequency of categories with multiple parents in GO is illustrated in Figure 1. Ontologies are a rich source of information for comparing functions and relationships between various subsets of genes. Clearly, genes in the same category are expected to be similar. Likewise, genes whose categories share a common parentage would also be expected to exhibit some similarity, albeit to a lesser extent than members of a common category. Recent efforts have tried to extract the functional relationships of the expressed genes seen in microarray studies based on their classification in GO[11].

Ideally, one would expect that the categorical similarities and dissimilarities derived from a domain expert's knowledge of a gene's function could be used as the basis for extracting biologically meaningful clusters[15]. We explore the potential for extracting such meaningful clusters directly from pairwise

KDD'06, August 20–23, 2006, Philadelphia, Pennsylvania, USA.

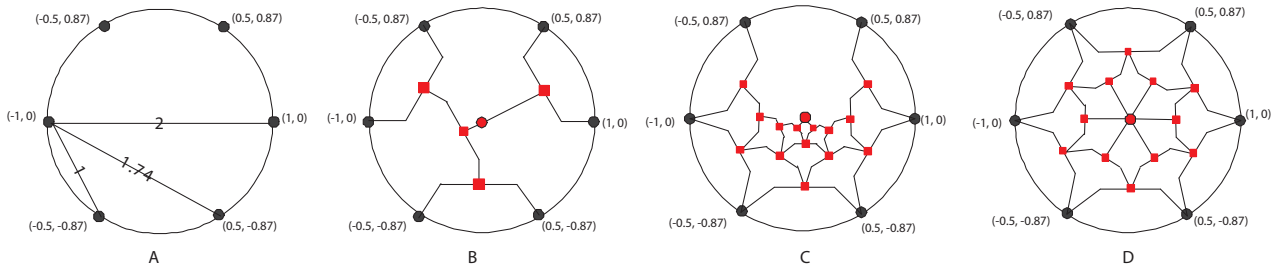


Figure 2: Comparison of clustering algorithms on an example of 6 points in 2D. (A) Distribution of the points and 3 distances appearing in the figure; (B) Hierarchical Clustering; (C) Pyramidal Clustering; (D) PoCluster. Note: black dots: points; red rectangle: intermediate clusters; red dots: cluster of universal set.

dissimilarity measures. Classic clustering algorithms are typically (with some exceptions noted later) either flat, or hierarchical, data partitions, whereas the categories of an ontology allow objects to be members of multiple categories or clusters, and allow clusters to have multiple parents.

Before proceeding, we examine the specific classification notion assumed in this paper. Mathematically, a general classification system, namely, ontology, is a partially ordered set[1], or poset. It represents the relationships among multiple categories of objects. For our discussion, we consider a category as a set of objects and the more "specific" relationship between a parent and a child as a "subset" relationship. Therefore, a poset contains the sets of objects as the elements ordered according to their subset relationships. A poset can be constructed from any combination of subsets taken from the set's power set. Therefore, the set of posets has a maximal cardinality of $2^{2^{|\mathcal{N}|}}$, where $\mathcal{N}$ is the object set.

A poset generalizes hierarchical clustering structures by allowing overlaps. This generalization poses a challenge to traditional clustering methods. Classical clustering algorithms generate disjoint subsets, such as graph-theoretical clustering, density-based clustering and k-means type clustering, etc. Even agglomerative hierarchical clustering methods maintain the invariant that child subsets of a common parent are disjoint.In this paper, we focus on incorporating all of the available information from a given dissimilarity matrix into a clustering algorithm, and derive partially ordered sets from it.

From an application standpoint, the goal of our paper is to derive plausible ontology-like categorizations of objects from a pairwise dissimilarity matrix via a clustering algorithm. We adopt the natural definition of the cluster in graph theory, maximal clique. A *clique cluster* is a maximal subset of objects whose maximal pair-wise dissimilarity is below a certain threshold. The *PoCluster* is a collection of maximal cliques arrived at by smoothly varying the threshold from 0 to the maximum dissimilarity within the dataset.

In order to construct the PoCluster from a dissimilarity matrix, we map our problem to a dual graph problem. We start with a graph with no edges and gradually insert the edges in the increasing order according to their dissimilarity values. After all edges less-than-or-equal to a given dissimilarity threshold are inserted, the graph is searched for *cliques*. These cliques represent potential categories (subsets). In subsequent passes the threshold is increased and the process repeats until all objects are combined into a single clique. As a result, the series of cliques form a PoCluster. Our experiments on real data have shown effectiveness and efficiency compared with 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 taxonomies. Section 3 defines PoCluster and its properties. Section 4 constructs the poset from the dissimilarity data. A performance study is reported in Section 5. Section 6 concludes the paper and discusses some future work.

2. RELATED WORK

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

2.1 Hierarchial and Pyramidal Clustering

Both hierarchical[10, 3] and pyramidal clustering [7, 4] generate clusters that have bijections to some special sub-classes of dissimilarity matrices.

Hierarchical clustering[10, 3] refers to the formation of a recursive clustering of data objects: a partition into two clusters, each of which is itself hierarchically clustered. It is often represented by a *dendrogram*, that is, a binary 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 maximal dissimilarities between the descendant leaves. It has been proven that a bijection exists between hierarchical clustering and an *ultrametric*[7] — a special type of metric in which the dissimilarities satisfy the *ultrametric triangle inequality* $D(a,c) < max\{D(a,b), D(b,c)\}$. An equivalent way of defining an ultrametric is that there exists a linear order of all objects such that their dissimilarities are the distances between them.

Pyramidal clustering[7, 4] 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 structure can be represented by a directed acyclic graph. It is known that a bijection exists between pyramidal clustering and 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 case, neither hierarchical clustering nor pyramidal clustering is able to generate clustering from which the original dissimilarity matrix can be re-derived. That is, the bijection no longer exists.

Information of dissimilarity will be lost during the clustering procedure.

Consider the example shown in Figure 2. Figure 2(A) shows 6 points on a circle in a 2D space. The non-overlapping property of hierarchical clustering (Figure 2(B)) prohibits the clustering of (-1, 0) with (-0.5, -.87) once it is clustered with (-0.5, 0.87), although it has the same distance to both (-0.5, -.87) and (-0.5, 0.87). Pyramidal clustering (Figure 2(C)) alleviates this problem by allowing (-1, 0) to be clustered with both (-0.5, -0.87) and (-0.5, 0.87). However, a strict ordering of points based on Robinson matrix criterion is impossible in this case. With the optimized ordering $\langle(-0.5, 0.87)$, (-1, 0), (-0.5, -0.87), (0.5, -0.87), (1, 0), (0.5, 0.87)$\rangle$, points (-0.5, 0.87) and (0.5, 0.87) are not connected although they have the minimum distance. Our method (Figure 2(D)) considers only the different dissimilarities in the data. In this example, they are $\{1, 1.74, 2\}$. For each dissimilarity d, we look for the maximal sets of points whose maximum pair-wise dissimilarity is less than or equal to d. When $d = 1$, the set of clusters are those intermediate clusters in the first level shown in Figure 2(D).

2.2 Dissimilarity Derived from an Ontology

Similarities or dissimilarities among objects organized in a hierarchical structure are often easy to compute than those of the objects in a DAG(Directed Acyclic Graph), such as wordNet and GO. Semantic similarity[6] was introduced to measure the similarity between two concepts in the WordNet. Similar measures were also applied to determine the dissimilarity between a pair of genes in Gene Ontology[11, 15].

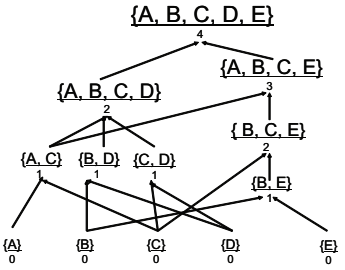
The following are a few widely adopted measures. Guided by the intuition that the similarity between a pair of concepts may be assessed by "the extent to which they share information", Resnik defined the dissimilarity between two concepts c_1 and c_2 as the information content of their lowest subsumer(which is measured by a probability p), i.e., $sim(c1, c2) = -\log p(ls(c1, c2))$. Leacock and Chodorow proposed a very different similarity measure that relies on the length $len(c1, c2)$ of the shortest path between two concepts. However, they limit their attention to specific links and scale the path length by the overall depth D of the taxonomy: $dis(c_1, c_2) = \frac{-log(len(c_1,c_2))}{2D}$. It is unclear how clusters derived from these dissimilarities relate to the original ontology. In Section 5, we provide a comparison of those algorithms in how suitable they are when used for recovering the original ontology. Jiang *et. al*[6] and Lin[6] also developed other two alternative similarity measures, which are a variation of Resnik's method.

3. MODEL

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 pairs in S is denoted by $S \times S$ or S^2.

A *dissimilarity matrix* describes the pair-wise relationships between objects. It is a mapping D from $(\mathcal{N} \times \mathcal{N})$ to a real nonnegative value. A dissimilarity matrix has the following two properties (1) reflectivity: $\forall x,\ D(x, x) = 0$; (2) symmetry: $\forall x, y,\ D(x, y) = D(y, x)$. 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 depicts the dissimilarity $D(x, y)$ between the two objects it connects. We denote the graph implied by the dissimilarity D as $G(D)$.

Example: Figure 3 (B) shows a dissimilarity matrix of object set $\{A, B, C, D, E\}$. It satisfies both reflectivity(0 diagonal) and symmetry. This dissimilarity matrix can be mapped to the undirected weighted graph in Figure 4 ($d = 4$). Each node in graph corresponds to an object, each edge corresponds to a pair and the weight of the edge is the dissimilarity between the pairs of objects.



(A) An example PoCluster

	A	B	C	D	E
A	0	2	1	2	3
B	2	0	2	1	1
C	1	2	0	1	2
D	2	1	1	0	4
E	3	1	2	4	0

(B) Dissimilarity Matrix

Figure 3: A running example. (A) shows a PoCluster which contains 13 clusters over the object set {A, B, C, D, E}. Each node in the PoCluster represents a clique cluster with its diameter. 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 $S_1 \subset S_2$; (B) shows a dissimilarity matrix which corresponds to the PoCluster in (A). Applying Algorithm in Sec 4 can completely construct the PoCluster in (A) from (B).

A clique is a fully connected subgraph in an undirected graph.The *diameter* of a clique is the maximum edge weight within the clique. A *clique cluster* is defined as a maximal clique with a diameter d. A diameter indicates the level of dissimilarity of the set of objects in the clique cluster.

DEFINITION 3.1. *(Clique Cluster). Let $G(D)$ be an undirected weighted graph of a dissimilarity matrix D. A clique cluster $C = \langle S, d\rangle$ is a maximal clique S with diameter d in graph $G(D)$.*

When there are multiple cliques within the graph with the same diameter d, we denote this set of clique clusters as cliqueset(d).

Example: Given the dissimilarity matrix shown in Figure 3(B), $ABCD$ forms a clique with maximum edge weight 2, as shown in Figure 4. Therefore, $ABCD$ is a clique cluster with diameter 2. So is BCE. We denote them as cliqueset(2)= $\{ABCD, BCE\}$.

PoCluster

The notion of clique cluster is not new. The intermediate clusters generated by hierarchical clustering using a complete linkage criterion is similar to clique clusters in spirit, since they both look for a cluster with minimum diameter. However, when two clusters are merged in hierarchical clustering, the relationship(linkage) between two clusters to be merged solely depends on the maximum or minimum dissimilarity between a pair of objects within two clusters. This oversimplified similarity measure ignores many pair-wise relationships

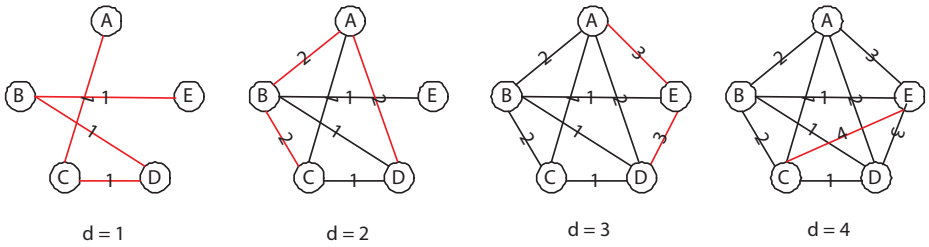


Figure 4: Four directed weighted graphs corresponding to the dissimilarity matrix in Figure 3 (B) with maximum edge weight $\{d = 1, 2, 3, 4\}$.

between objects in the two clusters. To best explore and preserve the information carried by a dissimilarity matrix, in this paper, we present PoCluster. PoCluster reveals clique clusters with all possible diameters present in the dissimilarity matrix. The non-disjoint feature allows us to explore richer and deeper relationships among objects. We formally define PoCluster in Definition 3.2.

DEFINITION 3.2. *(PoCluster) Let D be a dissimilarity matrix, a PoCluster P of D is defined as*

$$P = \bigcup_{\forall d \in W(D)} cliqueset(d). \quad (1)$$

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

Example: The dissimilarity matrix in Figure 3(B) consists of 4 possible diameters, they are $\{1, 2, 3, 4\}$. For each diameter, we map them into an undirected graph, namely, diameter graph, where there exists an edge between two nodes only if their dissimilarities are smaller than or equal to the diameter. For each graph, there exists a set of cliques in it. For example, in Figure 4, when $d = 1$, there are four 2-cliques in the graph. The set of the clique clusters generated in each of the diameter graph is shown as poClusters in Figure 3(A).

Now, we examine the properties of PoClusters. Similar to hierarchical clustering, PoCluster also includes the set $\mathcal{N}$ containing all the objects. This set has the maximum dissimilarity in D as its diameter. PoCluster does not ignore dissimilarity measures as hierarchical clusters do since each pair-wise dissimilarity is covered by at least one clique cluster whose diameter equals to the pair-wise dissimilarity. In addition, the maximal clique cluster insures that if one cluster is a subset of the other, one's diameter will be strictly lower than the other. This property generates a partial order of the clusters in the PoCluster as shown in Figure 3(A).

PROPERTY 3.1. *Let D be a dissimilarity matrix of object set N, Let P be a PoCluster of dissimilarity matrix D, P has the following properties.*

1. $N \in P$
2. $\forall C_1, C_2 \in P$, *if* $C_1 \subset C_2$, *then* $diam(C_1) < diam(C_2)$
3. $\forall x, y \in N$, *there exists a cluster* $C \in P$, *such that* $\{x, y\} \subseteq C$ *and* $diam(C) = D(x, y)$;
4. $\forall x \in N$, $\{x\} \in P$.

d	cliqueset(d)
d=1	AC, BD, CD, BE
d=2	$ABCD, BCE$
d=3	$ABCE$
d=4	$ABCDE$

Table 1: PoCluster generated based on dissimilarity matrix in Figure 3(B).

4. CONSTRUCTION OF A POCLUSTER

Given a dissimilarity matrix D, the corresponding PoCluster P (i.e., $P = \{\text{cliqueset}(d) | \forall d \in W(D)\}$) can be found by repeating a simple procedure. 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. Finding a clique of size k in a graph is one of the original NP-complete problems identified in Karp's seminal paper [12]. The k-clique problem can be reduced in polynomial time to a PoClustering problem, hence the PoClustering problem is NP-hard. In order to make a PoCluster reconstruction practical for large datasets, we present an incremental algorithm which takes advantage of already constructed clique clusters to generate their subsequent parent clusters.

Algorithm 1 gen_poCluster(E)
Input E:an ordered list of edges.
Output P: a PoCluster

1: $t = 0$; $G = \langle N, \emptyset \rangle$.
2: $E^0 = E$
3: **while** $E \neq \emptyset$ **do**
4: $\quad e \leftarrow min(E^t)$;$E^{t+1} \leftarrow E^t - e$
5: $\quad C^{t+1} \leftarrow$gen_clique_clusters(C^t, e)
6: $\quad P \leftarrow P \cup C^{t+1}$; $t \leftarrow t + 1$
7: **end while**
8: return P

The Optimized Incremental Algorithm

The incremental algorithm only computes cliques that are affected by the introduction of new edges. The algorithm keeps a pool of all cliques in the previous graph. Given the next graph with more edges, the pool of cliques can be updated as follows: First, find all the cliques in the pool that share a vertex with the new edges. Second, if a clique in the pool can be extended by adding one or more of the new edges, the extended maximal cliques are added into the pool, and the cliques in the old pool that are subgraphs of the newly added cliques are removed. The parent-child relationships can be established between new cliques and removed cliques.

Let $x \in V$ be a node in graph G, and let $\pi(x)$ denote all the cliques containing x.

LEMMA 4.1. *Let $G = (V, E)$ be an undirected graph. Let C be the cliques contained in G, and let $e = (x, y)$ be the edge added to G. The cliques in the new graph G' can be obtained based on the cliques in the graph G.*

- *The added cliques are the maximal complete subgraphs of $\{c_1 \cap c_2 \cup \{x, y\} | \forall \langle c_1, c_2 \rangle \in \pi(x) \times \pi(y)$, where $\pi(x), \pi(y) \in G\}$.*
- *The removed cliques are those $\{c | \exists c', \{\langle c, c' \rangle \in \pi(x) \times \pi(y)\} \wedge \{|c \setminus c'| = 1\}\}$.*

When a new edge is inserted, a new set of cliques are generated based on the previous graph. Algorithm 2 implements the second part of the Lemma 4.1 while removing cliques in $G^{(t-1)}$ from which the new cliques are derived.

Algorithm 2 gen_clique_clusters(C, e)
Input e: an new edge and C: current cliques.
Output C_{new}: cliques after adding e.

```
1: (x, y) ← get_vertices(e)
2: C_new = ∅
3: for all ⟨c_1, c_2⟩ ∈ π(x) × π(y) do
4:    c ← c_1 ∩ c_2
5:    if {c = c_1} ∨ {c = c_2} then
6:       C ← C − {c}
7:    else if ∄c' ∈ C_new, c ∪ {x, y} ⊂ c' then
8:       C_new ← C_new ∪ {c ∪ {x, y}}
9:    end if
10: end for
11: C_new = C_new \ {c | c ⊂ c', c, c' ∈ C_new}
12: return C_new = C ∪ C_new
```

5. EXPERIMENTS

Our experiments are done on a subset of gene function categories obtained from Gene Ontology. We compare hierarchical clustering(Hierarchy), pyramidal clustering(Pyramid), and incremental optimal poCluster algorithm, to evaluate the their capabilities in preserving the structures.

5.1 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 P_1 and P_2, the match score of P_2 to P_1 is computed as:

$$match(P_1, P_2) = mean_{s_1 \in P_1}(max_{s_2 \in P_2}(\frac{s_1 \cap s_2}{s_1 \cup s_2}) \quad (2)$$

5.2 Gene Ontology

Our experiment compares the quality of the three clustering algorithms given a real ontology categorization. We also evaluate our method for deriving dissimilarities against previous approaches reviewed in Section 2.

We use 799 genes which are the most active and cell cycle co-regulated in the yeast cell cycle data of Spellman et al.(1998)[14]. We consider three GO files on biological process (BP), cellular component (CC), and molecular function (MF), from the Gene Ontology database. We extract all GO categories that contain at least two genes and remove duplicate categories.

The remaining GO categories are taken to generate dissimilarity matrices as the input to the clustering algorithms.

GO files	#Known genes	#Terms	Maxlevel	Mean Overlap
CC	64	349	7	46%
BP	159	523	7	21%
MF	230	451	10	21%

Table 3: Statistics for the three GO files. MF: Molecular Function, CC: Cellular Component; BP: Biological Process

Table 3 presents size and overlapping statistics of our data. The statistics of the three GO files are listed in Table 3.

We first compare the two possible similarity measures including Resnik, and Leacock and Chodorow (LC) [6]. These methods are applied to generate dissimilarity matrices of genes based on the structure of categories in GO. We derive two dissimilarity matrices, one by each method. We then apply the PoCluster algorithm to the dissimilarity matrices in order to reconstruct the categorization structure of GO. The matching scores of three different files are shown in Table 4. The result shows that Resnik method renders consistently higher recoverability result than LC method for all three GO files. Therefore, in the rest of experiment, we take Resnik method to measure the affinity of genes based on the GO categorization.

Similarity	CC	BP	MF
Resnik	0.8471	0.8680	0.7261
LC	0.5616	0.7157	0.6684

Table 4: Reconstructed poset match score to original GO based on various similarity measures

We then apply the three algorithms: PoCluster, hierarchical clustering and pyramidal clustering to reconstruct the original categorization. Among them, PoCluster performs the best in recovering the GO categories, while hierarchical and pyramidal clustering seems to miss many of the categories. In this case, two additional measurements are used to evaluate the relationships between a reconstructed poset(P) and GO files(go). They are recovery rate and accuracy. The recovery rate is the percentage of GO categories recovered; the accuracy is the percentage of the clusters discovered that truly appear in GO categories. The two measurements provide more information about those clusters. The recovery rate in the second column of Table 2 is more than 50% and even closer to 79%(MF) of the categories that cannot be properly discovered by hierarchical and pyramidal clustering. According to the third column in the same table, over 96% of the discovered clusters by PoCluster are actual matches to the GO categories. In comparison, the spurious clusters in pyramidal clustering and hierarchical clustering may exceed 50%, which is unacceptable in real applications.

6. CONCLUSION AND FUTURE WORK

We have presented a new clustering algorithm for the automatical generation a set of partially ordered clusters(PoCluster) based on the pairwise relationships between objects. The structure of a PoCluster is analogous to that of the classification ontology such as Gene Ontology by allowing overlapping between sibling categories and allowing one child category to have multiple parent categories.

PoClustering is a generalization of both hierarchical clustering and pyramid clustering. PoCluster provides both homogeneity within a cluster, as measured by the cluster's diameter, and separation between clusters. Different from disjoint clustering algorithms, PoClusters allow overlaps be-

	match score			recovery rate			accuracy		
Algorithm	CC	BP	MF	CC	BP	MF	CC	BP	MF
poCluster	0.8493	0.8776	0.7263	0.7813	0.8553	0.6478	0.9615	0.9379	1.0000
Pyramid	0.6621	0.5342	0.6335	0.4531	0.2704	0.4652	0.6905	0.4574	0.8168
Hierarchy	0.5086	0.4960	0.5046	0.4063	0.2893	0.3696	0.6047	0.3262	0.4545

Table 2: Reconstructed poset match score to original GO by the three algorithms. *go* represents the GO file and P is the reconstructed poset

tween clusters in a meaningful way. However, given an arbitrary poset containing the whole object set and singleton set, it might not be a valid PoCluster. For example, given a set of objects $\{A, B, C\}$, a poset of this object set as shown in Figure 5, is not a valid PoCluster. Assume the poset shown is a valid poset, according the second property listed in Property 3.1, we have diam(AB) < diam(ABC), diam(AC) < diam(ABC), and diam(BC) < diam(ABC). However, the three conditions cannot be true at the same time since diam(ABC) = max (diam(AB), diam(BC), diam (AC)). Therefore, this poset is not a valid PoCluster. One of the interesting questions that is worth further investigation is how to identify an arbitrary poset as a valid poCluster.

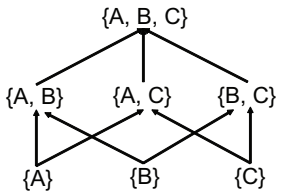


Figure 5: An example of poset which is not a valid PoCluster

As we know, the hierarchical clustering only preserves the information of the ultrametric dissimilarity matrix. By preserving, we mean there exists one to one correspondence between the set of hierarchical clusters and the set of ultrametric dissimilarity matrices. Similarly, the pyramidal clustering, which is an extension of hierarchical clustering preserves another special type of matrices, namely, the Robinson matrices. Applying pyramidal clustering on other matrices may cause information contained in the dissimilarity matrices to be ignored.

This inspires us to pursue the following questions: Is PoCluster able to fully preserve the information provided in a dissimilarity matrix? If so, what are the types of dissimilarity matrix? And how should we derive it from a PoCluster? In addition, how do they relate to the Robinson matrices or ultrametric matrices? The answers to these theoretical questions may lead to deeper understanding of the relationships between the sets of PoClusters and the set of dissimilarity matrices.

The formal definition of PoCluster is primarily of theoretical interest, since computing the exact solution is likely to be intractable for large problems. In order to address the challenge, it is important to investigate an approximation algorithm which is scalable to large datasets in our future work.

7. REFERENCES

[1] M. Ashburner, CA. Ball, JA. Blake, D. Botstein, H. Butler, JM. Cherry, AP. Davis, K. Dolinski, SS. Dwight, JT. Eppig, MA. Harris, DP. Hill , L. Issel-Tarver, A. Kasarskis, S. Lewis, JC. Matese, JE. Richardson, M. Ringwald, GM. Rubin, G. Sherlock: Gene Ontology: tool for the unification of biology. Nat Genet 2000, 25:25-29.

[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] Budanitsky, A., and G. Hirst, "Semantic Distance in WordNet: An Experimental, Application-oriented Evaluation of Five Measures", Workshop on WordNet and Other Lexical Resources, in the North American Chapter of the Association for Computational Linguistics (NAACL-2001), Pittsburgh, PA, June 2001.

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

[8] R. O. Duda, P. E. Hart, and D. G. Stork, Pattern Classification, John Wiley and Sons, Inc., 2001.

[9] L. K. Hua, Introduction to Number Theory. Springer-Verlag, New York, 1982.

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

[11] J. L. Sevilla, V.Segura, A. Podhorski, E. Guruceaga, J. Mato, L.A. Martinez-Cruz, F. J. Corrales, and A. Rubio. Correlation between gene expression and GO semantic Similarity. IEEE/ACM transactions on computational biology and bioinformatics, vol2, No4, 2005.

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

[13] P.W. Lord, R. Stevens, A. Brass, and C.A.Goble. Investigating semantic similarity measures across the Gene Ontology: the relationship between sequence and annotation. Bioinformatics, 19(10):1275-83, 2003.

[14] P.T. Spellman, G.Sherlock, M.Q.Zhang, V.R.Lyer, K.Anders, M.B.Eisen, P.O.Brown, D.Botstein, and Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast sacccharomyces cerevisiae by microaray hybidization. Molecular Biology of the Cell, 9:3273-2297, 1998.

[15] H. Wang, F. Azuaje, O. Bodenreider. An ontology-driven clustering method for supporting gene expressio analysis. Proceedings of the 18th IEEE International Symposium on Computerh-Based Medical Systems, pp. 389-394. 2005.