Automated Taxonomy Generation for Summarizing Multi-type Relational Datasets

Tao Li and Sarabjot S. Anand

***Abstract*— Taxonomy construction provides an efficient navigating and browsing mechanism to people by organizing large amounts of information into a small number of hierarchical clusters. Compared with manually editing taxonomies, Automated Taxonomy Generation has numerous advantages and has therefore been applied to categorize document collections. However, the utility of this technique to organize and represent relational datasets has not been investigated, because of its unaffordable computational complexity. In this paper we propose a new ATG method based on the relational clustering framework *DIVA*. By incorporating the idea of Representative Objects, the computational complexity can be greatly reduced. Moreover, we analyze the divergence of the data attributes and label the taxonomic nodes accordingly. The quality of the derived taxonomy is quantitatively evaluated by a synthesized criterion that considers both the intra-node homogeneity and inter-node heterogeneity. Theoretical analysis and experimental results prove that our approach is comparably effective and more efficient than other ATG algorithms.**

I. Introduction

Today's information technology makes it possible to produce a vast amount of data. For instance, on the World Wide Web millions of new webpages are published everyday. The problem of *information overload*, i.e. the incapability of people to digest such huge volumes of information, becomes very critical in many applications. Although some data mining techniques have been utilized to address this problem, they are not always effective. For example, search engines usually return too many pages of web documents that match a user query, but most users only have patience to read the top-20 results. Additionally, if the search result is not satisfactory, users have to manually adjust the target keywords or change the search strategies, which requires them to master advanced searching skills or domain-specific knowledge. Ideally, we expect a data model that is very effective in summarizing the characteristics of the whole dataset as well as intuitive enough for ordinary users to understand.

Taxonomy construction provides an efficient navigating and browsing mechanism by organizing large amounts of information into a small number of hierarchical clusters [21][13]. Some taxonomies as Yahoo! Directory (http://dir.yahoo.com) and Open Directory Project (http://www.dmoz.org) have been created on the Web to organize webpages into a global topical structure, but such manually edited taxonomies are only feasible for small or static datasets. In contrast, the technique of Automated Taxonomy Generation (ATG) owns obvious advantages and has therefore become an attractive research topic in recent years. Currently this technique is mainly used to organize a large collection of documents [17][14][5] or web pages/images [4][3][15][13]. Usually, the techniques of Natural Language Processing or Information Retrieval are applied in the first step to extract some linguistic features from the text, and then a hierarchical taxonomy is automatically synthesized by utilizing hierarchical clustering techniques. Given the derived taxonomy, users can easily browse the structure of the whole document collection and focus on the topics they are most interested in. A good example is the search engine vivisimo (http://www.vivisimo.com), which automatically groups the retrieved webpages into a topic hierarchy so as to facilitate users' browsing.

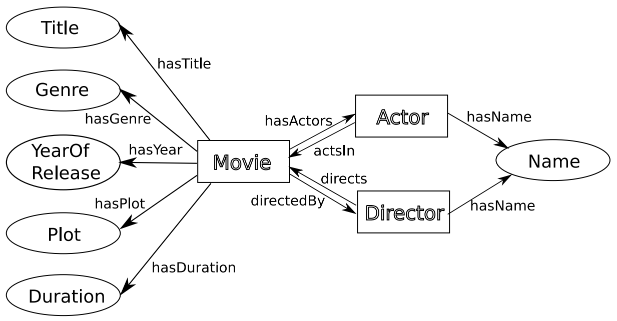

Fig. 1. Ontology of a movie dataset

Organizing multi-type relational datasets into hierarchical structures has potential value in both theoretical and practical aspects. Automatically derived taxonomies will not only facilitate people navigating the dataset but also accelerate many online applications. For example, Fig. 1 shows an ontology of a movie dataset used by a content-based filtering system, in which a bi-directional arrow means a recursive references between two concepts of the ontology, so the data object representing a movie will include all the participant actors and in turn other movies these actors participate in. The content-based filtering system expects to find a collection of candidate movies that are most similar to the movies the current user has visited. When comparing two movies, the related actors and in turn more movies have to be considered. Such recursive calculation is too expensive for the online recommendation service. If the dataset is organized by a hierarchical structure so that the movies contained in sibling nodes are more similar than those in non-sibling nodes,

Tao Li is with the Department of Computer Science, University of Warwick, Coventry CV4 7AL, United Kingdom (phone: +44 24 7657 3797; email: Li.Tao@warwick.ac.uk).

Sarabjot S. Anand is with the Department of Computer Science, University of Warwick, Coventry CV4 7AL, United Kingdom (phone: +44 24 7657 3778; fax: +44 24 7657 3024, email: Sarabjot.Anand@dcs.warwick.ac.uk).

the system can search a small portion of movies at the beginning and gradually expand the search scope when necessary, which will greatly reduce the online response time of the recommender system. Nevertheless, to the best of our knowledge, the utility of ATG techniques to organize and represent multi-type relational datasets has not been investigated till now, because most traditional hierarchical clustering algorithms have unaffordable computational complexity when processing relational datasets.

In this paper, we expand the applicability of ATG technique from document collections to relational datasets. Our algorithm does not depend on the extraction of linguistic features, but only requires a similarity or distance measure defined on the dataset. The idea of Representative Objects adopted in our algorithm guarantees the taxonomic nodes as homogeneous as possible and also speeds up the procedure of taxonomy generation for very large datasets. The labels of the taxonomic nodes are assigned by analyzing the divergence of the data attributes. Moreover, we propose a synthesized criterion, considering both the intra-node homogeneity and inter-node heterogeneity, to quantitatively evaluate the quality of the derived taxonomy. We also compare the robustness of our algorithm with others under different noise ratios, which is not studied in previous works.

The rest of the paper is organized as follows: Some related works about ATG algorithms and cluster analysis are presented in Section II. We review the relational clustering algorithm DIVA with more details in Section III before explaining our new ATG approach in Section IV. Experimental results are provided in Section V. And finally, Section VI summarizes some conclusions and future works.

II. Related Works

As the cornerstone of ATG techniques, cluster analysis aims to partition a dataset into a certain number of groups (clusters) by optimizing their internal homogeneity and external separation. Data objects in the same cluster are more similar to each other than those from different clusters [10]. Many clustering approaches build hierarchical trees to reflect the internal structure of the original dataset and preserves the similarity relationship between data objects, such as the traditional Hierarchical Agglomerative Clustering (HAC) approach or the bisecting K-Means Partitioning approach [18]. Another well-known algorithm BIRCH [20] utilizes the idea of *Clustering Feature* and incrementally builds a CF-tree to preserve the inherent clustering structure within a propositional dataset. Each node in the CF-tree summarizes the statistical characteristics of its corresponding cluster. However, BIRCH is only applicable for propositional datasets because it requires data to be additive and divisible. Yin et al. [19] developed a reinforcement hierarchical clustering model SimTree to approximate the original similarity values between pairs of objects, but their approach only utilizes frequent pattern mining techniques to build the initial SimTrees. Information contained in the property attributes of data objects are not considered, which definitely impacts the accuracy of the final cluster result. Li and Anand [16] proposed a multiple-phase clustering algorithm for relational datasets, which will be reviewed in the next section.

TABLE I
Main Framework of DIVA

```
DIVA (dataset $D$, number of ROs $r$, variance $v$)
    1) cluster set $\{C_k\}$ ← call the Divisive-Step, given $D$, $r$ and $v$ as
       the parameters.
    2) dendrogram $T$ ← call the Agglomerative-Step, given $\{C_k\}$ as
       the parameter.
```

In order to more effectively organize and discover knowledge within a large document collection, many algorithms of automatically constructing taxonomies upon document collections have been developed. Usually documents are first pre-processed by utilizing techniques in linguistics, information retrieval or statistics so that their textual contents are represented by some quantitative features. For example, a document is first converted into a term vector by the TF-IDF term weighting scheme. The semantic similarity measure between two documents is defined by the Pearson's Correlation Coefficient or the cosine value of their corresponding term vectors [1]. Based on that, traditional hierarchical (divisive or agglomerative) clustering algorithms can be used to construct a hierarchical taxonomy [5]. Boley developed a divisive partitioning algorithm PDDP in [2], which iteratively performs binary divisions in each cluster when its scatter value is greater than a user-defined threshold. The scatter criterion is determined by a linear discriminant function that is based on the first principal component of the covariance matrix. Clerkin et al. [6] applied the incremental conceptual clustering algorithm COBWEB [8] to construct class hierarchies. COBWEB performs the hill-climbing search in the space of possible taxonomies and uses *Category Utility* [9] to select the possible categorizations. After building the initial taxonomy as a single category, the algorithm uses CU to evaluate the taxonomies and perform one of the following operations accordingly: classify an instance into the existing cluster, create a new cluster, divide or merge the existing clusters. The output of COBWEB can be finally translated into a class hierarchy. Lawrie and Croft proposed the use of a set of topical summary terms for taxonomy construction. These topical terms are selected by maximizing the joint probability of their topicality and predictiveness, which is estimated by the statistical language models of the document collection [15]. Kummamuru et al. developed an incremental learning algorithm DisCover to maximize the coverage as well as the distinctiveness of the taxonomy. [13]. A general survey of various ATG techniques is provided in [12]. However, none of the above approaches can easily be extended to multi-relational datasets. For example, PDDP requires the availability of covariant matrix and probabilistic models beyond propositional domains, which are usually undefined in the relational datasets.

III. DIVA Framework

In this section we review in detail the general clustering framework *DIVA* for relational datasets [16], since it is the basis of our ATG algorithm. DIVA is a multiple-phase clustering algorithm [10], composed of two steps: divisive and agglomerative. The whole dataset is first divided into a number of clusters so that the variance of each cluster is equal to or less than a particular threshold value v. Based on these clusters, a hierarchical dendrogram is built using an agglomerative approach. The procedure of the DIVA framework is summarized in Table I.

A. Representative Objects

Traditionally most partitional clustering algorithms need to compute the centroid of the clusters during their execution. In propositional datasets, the computational complexity of obtaining this geometric feature is linear to the size of the dataset. However, in the relational dataset the complexity becomes quadratic because the relational objects are not additive or divisible. To accelerate the clustering procedure for relational datasets, it is necessary to develop new methods for delimiting the shape of clusters. The idea of *Representative Object* (RO) is proposed to address this problem. ROs are defined as a set of maximum-spread objects in the dataset D, denoted as $\{ro_i\}$ $(1 \leq i \leq r)$. Since they are maximum-spread to each other, the selected ROs can well represent the shape of D. Furthermore, the distance between the farthest pair of ROs approximates to the diameter of the data space. We define the *variance* of the dataset D as:

$$Var(D) = \max_{1 \leq i,j \leq r} fd_{obj}(ro_i, ro_j) \tag{1}$$

Small variance means data objects reside in a compact data space and thus are more similar to each other, which also means the data are of higher homogeneity.

An efficient method for determining the ROs is proposed in [16]. After a start object x_s is randomly chosen, the i-th RO is determined as follows [1]:

$$ro_i = \begin{cases} \arg\max\limits_{x \in D} fd_{obj}(x, x_s) & \text{if } i = 1 \\ \arg\max\limits_{x \in D} \left(\min\limits_{1 \leq j < i} fd_{obj}(x, ro_j) \right) & \text{if } 2 \leq i \leq r \end{cases} \tag{2}$$

B. Divisive and Agglomerative Steps

The divisive step implemented in [16] partitions the dataset in a recursive fashion: Given the whole dataset D as the initial cluster C_0, its ROs are selected using Eq. 2. The variance of C_0 is evaluated by Eq. 1. If $Var(C_0)$ is greater than the pre-specified variance threshold v, C_0 will be divided. The furthest pair of ROs are used as the absorbent objects of sub-clusters respectively, and the other data objects in C_0 are distributed according to their distance to the absorbent objects. Then for each of derived sub-clusters, the above procedure is recursively launched if its variance does not satisfy the requirement.

The agglomerative step is launched after the divisive one to improve the quality of the resulting clusters. The cluster set $\{C_k\}$ obtained from the divisive step constitute the leaf nodes of the dendrogram. In each iteration, the most similar pair of sub-clusters (child-nodes) are merged to form a new super-cluster (parent-node). In this paper the words "cluster" and "node" are used interchangeable. The similarity value between nodes t and t' is evaluated by their RO set:

$$fs_{node}(t, t') = \min_{i,j} fs_{obj}(ro_i^{(t)}, ro_j^{(t')}) \tag{3}$$

where $\{ro_i^{(t)}\}$ and $\{ro_j^{(t')}\}$ are the set of ROs contained in node t and in node t' respectively. Conventionally, most agglomerative clustering algorithms use similarity values as the criterion to merge nodes, so we use the similarity function $fs(\cdot,\cdot)$ in Eq. 3 instead of the distance function $fd(\cdot,\cdot)$.

One key advantage of the DIVA clustering approach is that its computational complexity is linear to the size of D. Formal complexity analysis is provided in [16]. Furthermore, the maximum-spread set of ROs well represents the shape of clusters in the data space, which will bring benefits for the taxonomy construction procedure in the next section.

[1]We use $fs_{obj}(\cdot,\cdot)$ and $fd_{obj}(\cdot,\cdot)$ to denote the similarity and distance measures for relational data objects respectively. Commonly they hold the relationship $fs_{obj}(\cdot,\cdot) + fd_{obj}(\cdot,\cdot) = 1$.

IV. Automated Taxonomy Generation

The main purpose of building a taxonomy is "to provide a meaningful navigational and browsing mechanism by organizing large amount of information into a small number of hierarchical clusters" [21]. Kummamuru et al. [13] suggests six desirable properties of taxonomies generated from a document collection, which can be easily expanded for processing general datasets :

1) *Coverage*: Ideally all the data should be contained in at lease one node in the taxonomy. That is to say, for each data object, we can find a path in the taxonomy which starts from the root node and ends at the node that the data object belongs to. The derived taxonomy that covers more data objects is considered to be better.
2) *Compactness*: The property restricts the size of the derived taxonomy. Since the ATG techniques are used for the purpose of summarizing and navigating datasets, taxonomies with too many levels or too many branches in each level are not encouraged.
3) *Sibling Node Distinctiveness*: At any level of the hierarchy, the sibling nodes should be as different as possible from each other.
4) *Node Label Predictiveness*: Labels are used to summarize the contents or the characteristics of their corresponding nodes. A taxonomy with good node labels can help users to find their interested sub datasets with minimum efforts.
5) *Reach Time*: The average time spent by users to find their interested sub datasets is important. This criterion can be qualified by the size of the derived taxonomy, because both the depth and the width of the taxonomy will impact user's reach time.

6) *General to Specific*: The node labels should be selected to follow the general-to-specific relationship within the hierarchical taxonomy from top to bottom.

Obviously, since each data object is contained in one of the leaf nodes and each super-node contains all the contents of its sub-nodes, the dendrogram derived by the agglomerative step of DIVA satisfies Properties 1 and 6. However, its binary split structure does not meet the requirement of Property 2. In Section IV-A, we will describe our method of optimizing the hierarchical structure of the derived taxonomy. Comprehensive experiments for evaluating Properties 2, 3 and 5 are presented in Section V. Property 4 is originally proposed for document collections, where the node labels are usually assigned by parsing the textual content of the documents. Even in that case, labeling taxonomic nodes is rather subjective in nature, because it is hard to qualify the goodness of labels for summarizing taxonomic nodes. In the scenario of generating taxonomies for multi-type relational datasets, this work becomes more difficult, so we will provide some empirical analysis in Section IV-B.

A. Optimize the taxonomy structure

The binary split dendrogram, although strictly records the order of merging nodes, is not convenient for users to navigate, so it is necessary to reduce the total number of levels within the dendrogram and adjust its structure [13][11]. We expect the optimized hierarchy can better reflect the natural structure of the dataset, i.e. all the taxonomic nodes located at the same level should be approximately of the same granularity and all the sibling nodes belonging to the same parent node be as distinctive as possible.

Some heuristic strategies have been proposed in literature for this purpose. Duda et al. suggest that the node-pair similarity used in each agglomerative iteration indicates whether the formed cluster is natural or forced: an unusually large gap within node-pair similarities indicate a natural partitioning of the dataset [7]. Kashyap et al. used a series of thresholds, according to the suggestion of the taxonomy creator or the user, to select nodes and extract a taxonomy from the dendrogram [11]. Chuang and Chien developed a top-down algorithm to cut the dendrogram at some levels and merge the rest. The determination of a cut level depends on the quality of the cluster set at that level, which is calculated by a criterion of combining intra- and inter-cluster similarities as well as the size of the cluster set [4]. Cheng and Chien used the distance between two adjacent levels, which is the ratio of the change of intra-cluster distance to that of inter-cluster distance in two levels, as the cutting criterion [3].

As in [3], we use the following function to determine the cutting levels and thus optimize the taxonomic structure: given the original dendrogram T, in which all the levels have been numbered $1, 2, \ldots, L$ from top to bottom, function $G(l)$ for level l $(2 \leq l \leq L)$ within T is defined as:

$$G(l) = \frac{g_{intra}(l) - g_{intra}(l-1)}{g_{inter}(l) - g_{inter}(l-1)} \tag{4}$$

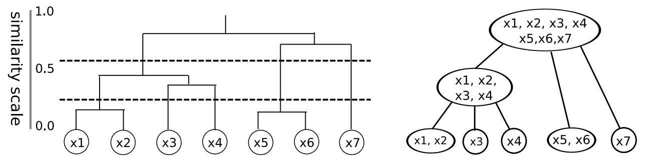

Fig. 2. Example of adjusting taxonomy

where $g_{intra}(l)$ and $g_{inter}(l)$ compute the intra-node and inter-node similarity of level l respectively. A set of cutting levels in T are determined when their corresponding values of $G(l)$ is greater than 1. An intuitive explanation is: if the change of intra-cluster similarity between level l and $l-1$ is greater than the change of inter-cluster similarity between them, which means the the data grouped by the nodes of level l is more cohesive or natural than that of level $l-1$, it is reasonable to set a cutting level between l and $l-1$; otherwise, levels l and $l-1$ would be merged. Because each node is represented by a set of ROs in DIVA, we re-define $g_{intra}(l)$ and $g_{inter}(l)$ as follows: assume there are K_l nodes located at the l-th level of T, denoted as $\{t_k^l\}$ $(1 \leq k \leq K_l)$, and function $P(t_k^l)$ obtains the parent node of t_k^l, then:

$$g_{intra}(l) = \frac{1}{K_l} \sum_{k=1}^{K_l} \Big(\min_{i,j} fs_{obj}(ro_i^{(t_k^l)}, ro_j^{(t_k^l)}) \Big)$$

and

$$g_{inter}(l) = \frac{1}{K_{l-1}} \sum_{p=1}^{K_{l-1}} \Big(\min_{\substack{k_1, k_2 \\ P(t_{k_1}^l) = P(t_{k_2}^l) = t_p^{l-1}}} fs_{node}(t_{k_1}^l, t_{k_2}^l) \Big)$$

The similarity function $fs_{node}(\cdot,\cdot)$ is given in Eq. 3. One special case is the calculation of $g_{inter}(1)$, because the first level only contains the root node and thus its inter-node similarity is undefined. To solve this problem we simply specify $g_{inter}(1)$ to be 0.

After determining all the cutting levels in T, we can optimize the structure of T accordingly: when a super-node t_p and its sub-node t_c are together located between two neighboring cutting levels or below the lowest cutting level, t_c will be removed and all t_c's sub-nodes will be linked to t_p, otherwise the linkage between t_p and t_c is kept in the final taxonomy T'. This procedure continues until all the nodes are unremovable. Fig. 2 gives a simple example: the left graph is a binary dendrogram T built by the agglomerative step and the right one is the taxonomy T' after optimization.

Compared with the approaches in [4] and [3], our approach is far less expensive to determine the cutting levels. Since each node in T is represented by ro_i $(1 \leq i \leq r)$, the complexity of calculating $g_{intra}(l)$ and $g_{inter}(l)$ is not related to the size of the whole dataset D: Given the number of nodes $\{t_k\}$ that are located at level l is K_l, to compute $g_{intra}(l)$ we will compare each pair of ROs belonging to the same node, so the total number of comparison is $O(r^2 \cdot K_l)$. Similarly the complexity of computing $g_{inter}(l)$ is $O\left(r^2 \cdot (\frac{K_l}{K_{l-1}})^2\right)$ in a balanced hierarchy or $O(r^2 \cdot K_l^2)$ in the worst case. When the number of nodes K_l is far less than the size of D, the time spent by the optimization step of our approach is approximately constant. In contrast, both of the approaches in [4] and [3] need to compare each pair of data objects when determining the cutting levels, so their computational complexities are quadratic to the size of D. This conclusion is supported by our experimental results in Section V.

B. Assign labels to taxonomic nodes

Another non-trivial issue is how to label the derived taxonomic nodes, since this task is rather subjective in nature. There is no commonly accepted criterion to evaluate the goodness of labels for summarizing taxonomic nodes till now. If the taxonomy is built for document collections, some key words or phrases extracted by NLP or IR techniques can be used as the labels of the taxonomic nodes [17][3][11], but labeling a taxonomy that represents general relational datasets is much more difficult. Our solution of addressing this issue is: by analyzing the attribute distribution of all data objects belonging to a taxonomic node t, find some attribute values of t that are the most different from those of t's sibling nodes and use them as the label of t. Formally, the Kullback-Leibler Divergence [7] is used to evaluate the distinctiveness between t and t's sibling nodes on attribute a:

$$D_{KL}(t,a) = \sum_h P_a(h) \log_2 \frac{P_a(h)}{Q_a(h)} \tag{5}$$

where $P_a(h)$ and $Q_a(h)$ are respectively the real and the "default" proportion of data objects in t whose attribute a has the value h. $Q_a(h)$ is estimated by using t's parent node. Because t's parent node contains the data objects of t as well as t's sibling-nodes, Eq. 5 reflects the distinctiveness between the current node t and its sibling nodes on attribute a. When an attribute contains a set of related data objects instead of simple numeric or nominal values, e.g. in the movie dataset a movie is related to many actors, we will recursively compute the attribute divergence of all the related objects that are belong to t and t's parent node using Eq. 5 and use the average value as the result. After obtaining the divergence value D_{KL} for all the attributes, those attributes with the maximum divergence are assigned to t as the label.

V. Experimental Results

To examine the effectiveness and efficiency, our DIVA-based ATG approach was compared with two representative taxonomy construction methods using pure binary agglomeration or binary division: (1) HAC-based approach [4] and (2) bisecting K-Means Partitioning approach [18] (we adopted k-Medoids as the meta-clustering algorithm here, since our experiments were conducted on relational datasets instead of document collections) until every leaf node contains one data object. For the sake of simplicity, in this section we use words "DIVA", "HAC" and "CBP" to represent the above three approaches respectively. For the CBP approach, we set the number of iterations within k-Medoids to 10 when splitting any taxonomic node, and the number of sub-clusters is fixed to 2 since the approach uses binary partitioning split. For our DIVA approach, we set the size of RO sets

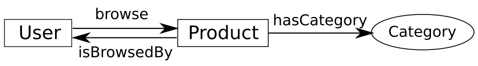

Fig. 3. Ontology of the Synthetic Dataset

r to 3 and the variance v to 0.4. Such parameter settings guarantee that all the leaf nodes in the derived taxonomy are homogeneous enough while the total number of nodes is far less than the size of dataset D. Both CBP and DIVA approaches are launched 5 times with different random seeds, and the final result is the average value of those experimental results obtained each time.

The experiments are conducted on two relational datasets: (1) a synthetic dataset of user visits, (2) a real movie dataset.

A. Synthetic Amazon Dataset

This synthetic dataset simulates the user visits to the Amazon website based on their interests. Fig. 3 shows the ontology of the synthetic dataset. First of all, 10,000 virtual products were created and randomly assigned into the Amazon product categories. In parallel, 2,000 users were created and assigned into 100 groups. For each user group, a probability distribution was constructed to simulate the users' preferences on the Amazon product categories, i.e. the likelihood of a user in that group to browse a product in that category. According to the information of users, groups, categories and products, in total 100,000 browsing actions were created by: (i) randomly select a user and get his group; (ii) randomly select an interest and get the related product category; (iii) randomly select a product that belongs to this category; (iv) create a browsing action between the user and the product. In order to test the robustness of the ATG approaches, we also introduced some noise data as follows: For each user, (i) uniformly choose four noise interests; (ii) randomly select a product that belongs to one of the four noise interests; (iii) create a noise browsing action between the user and the product. More details about the data generation procedure are included in [16].

As discussed in Section IV, the optimized taxonomy should maximize intra-node-uniformity and sibling-nodes-distinctiveness. Here we use an entropy-based criterion to evaluate the uniformity (purity) of all nodes in the derived taxonomy, because the classes of test data objects have been known. For node t_k, its intra-node entropy is defined as:

$$E_{intra}(t_k) = -\sum_h P_{h,k} \log_2 P_{h,k}$$

where $P_{h,k}$ is the proportion of data objects of class h in node t_k. The Kullback-Leibler Divergence is used to evaluate the sibling-nodes distinctiveness (named as inter-node entropy):

$$E_{inter}(t_k) = \sum_h P_{h,k} \log_2 \frac{P_{h,k}}{Q_{h,k}}$$

where $Q_{h,k}$ is the default proportion of data objects of class h. We use t_k's parent node to estimate the default distribution as in Section IV-B, and hence $E_{inter}(t_k)$ measures the

TABLE II

EVALUATION RESULTS USING SYNTHETIC DATASET

	HAC	CBP	DIVA
Time Spent (sec)	5286	5914	2532
Size of Taxonomy	1575.0	1218.3	438.3
Number of Leaves	1186.0	781.0	253.7
Intra-Node Entropy	0.062	0.455	0.471
Inter-Node Entropy	0.038	0.238	0.227
Quality	0.500	0.488	0.479

distinctiveness between the current node t_k to its sibling nodes. Generally speaking, lower intra-node entropy value means taxonomic nodes are more homogeneous and higher inter-node entropy value means they are more distinctive to each other. Another important issue is the size of the taxonomy. We expect the taxonomy would not be very large or very small, but it is very difficult to estimate the accurate value for that in a relational dataset. As in [4], we take the square root of N as the expected number of leaf nodes. To synthesize all the above issues, we use the following formula to evaluate the quality of the whole taxonomy:

$$Quality(T) = \frac{\sum_{t_k} E_{inter}(t_k)}{\sum_{t_k} E_{intra}(t_k)} \times \sqrt{1 - \left(\frac{M - \sqrt{N}}{N}\right)^2} \quad (6)$$

where M is the number of leaf nodes and N is the size of dataset. According to the definition of $E_{inter}(t_k)$ and $E_{inter}(t_k)$, we prefer the derived taxonomy with higher value of $Quality(T)$.

The first line of Table II records the time spent by all ATG approaches we examined. CBP and HAC are both very time consuming compared to DIVA. Actually, when generating a new taxonomic level of T, CBP will launch the k-Medoids algorithm to split every leaf node in T. And then each pair of data objects in the same cluster would be compared to select the medoids for the derived sub-clusters. Hence, the CBP approach has quadratic computational complexity as the HAC approach. In contrast, the DIVA approach is executed very fast due to its use of RO sets and assigning non-RO objects in linear time, and the computational complexity of optimizing the taxonomic structure is linear as well.

Table II also shows that the size of the taxonomy generated by DIVA is substantially smaller than that of the other two approaches. In total it contains 438 taxonomic nodes, less than half of the number of nodes in HAC or CBP, which means the taxonomy generated by DIVA better satisfied the Properties 2 (Compactness) and 5 (Reach Time) discussed in Section IV. As the tradeoff of the above advantages, the quality of the taxonomy derived by DIVA is slightly worse than that of HAC and CBP. When evaluated by the synthesized criterion in Eq. 6, the quality loss of the DIVA-derived taxonomy is 4.2% and 1.8% compared with HAC and CBP respectively.

Fig. 4 illustrates the robustness of all approaches under different noise ratios of browsing actions, ranging from 20% to 100%. In general, the quality of the taxonomy derived by all approaches is reduced as the noise ratio increases.

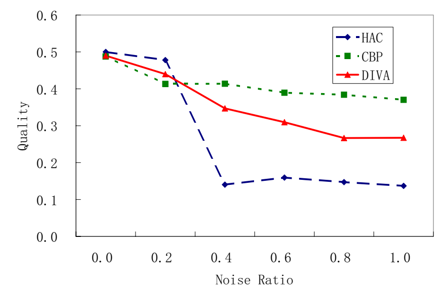

Fig. 4. Synthetic Dataset - Evaluate Taxonomy w.r.t noise

When the noise ratio is small (less than 0.4 in this case), HAC performs the best; otherwise, the accuracy of HAC decreases sharply. Instead, CBP performs more stable and keeps generating taxonomies of high quality under different noise ratio.

B. Real Dataset

All the taxonomy construction approaches were also evaluated on a real-world dataset, a movie knowledge base defined by the ontology in Fig. 1. After data pre-processing, there are 62,955 movies, 40,826 actors and 9,189 directors. The dataset also includes a genre taxonomy of 186 genres. Additionally, we have 542,738 browsing records included in 15,741 sessions from 10,151 users. For the original dataset neither HAC nor CBP are applicable due to their quadratic computational complexity, we have to reduce the size of the dataset. Based on the user visits, in total 5000 most popular movies are selected for the purpose of taxonomy construction.

In this dataset, since no pre-specified or manually-labeled class information for movies is available, we have to utilize the visit information from users to indirectly evaluate the derived taxonomies. The basic assumption is: two movies are likely to belong to the same category if they are co-visited by the same user, which reflects the organization of the dataset from the viewpoint of users. Hence the intra-node similarity S_{intra} can be evaluated by a variant of Jaccard Coefficient: the number of objects pairs that are co-visited by the same user over all possible objects pairs in the same node. The sibling-node-similarity (i.e. the inter-node similarity S_{inter}) is defined in the same way: the number of objects pairs belonging to two sibling nodes that are co-visited by the same user over all possible objects pairs in these two node. Different from the intra- and inter-node entropy, here high intra-node similarity value means taxonomic nodes are more homogeneous and lower inter-node similarity value means they are more distinctive to each other. Finally the quality of the whole taxonomy is defined as:

$$Quality(T) = \frac{\sum_{t_k} S_{intra}(t_k)}{\sum_{t_k} S_{inter}(t_k)} \times \sqrt{1 - \left(\frac{M - \sqrt{N}}{N}\right)^2} \quad (7)$$

Table III lists the experimental results of all approaches. Again the taxonomy built by DIVA is of the most compact-

TABLE III

EVALUATION RESULTS USING REAL DATASET

	HAC	CBP	DIVA
Size of Taxonomy	3343	5586	2548
Number of Leaves	2040.0	3041.0	1670.7
Intra-Node Similarity	0.537	0.457	0.572
Inter-Node Similarity	0.0188	0.0188	0.0168
Quality	0.026	0.020	0.033

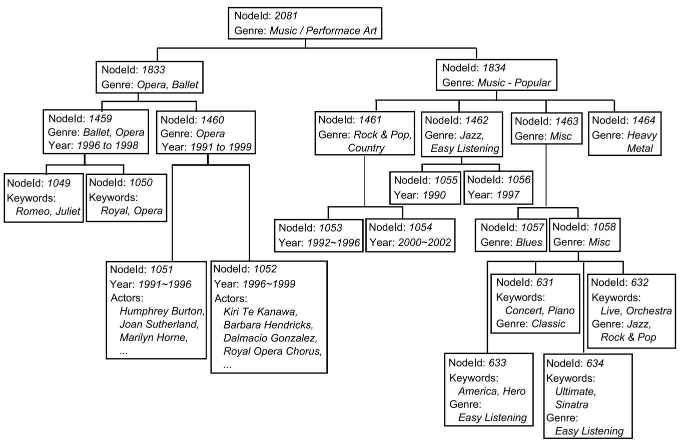

Fig. 5. Part of the Labeled Movie Taxonomy

ness. Moreover, we found in Table III that the intra- and inter-node similarities within the DIVA-derived taxonomy are both better than that of the HAC- and CBP-derived taxonomies, which leads to 26.9% and 65% improvements in taxonomy quality (evaluated by Eq. 7) when compared with HAC and CBP respectively.

Fig. 5 shows a part of the derived taxonomy labeled by the approach introduced in Section IV-B. At the first few levels, the attribute "Genre" can well distinguish a taxonomy node from its sibling nodes, When navigating deeper levels within the taxonomy, the movie genres of sibling nodes become more homogeneous and hence other attributes, such as keywords in the title, the year of movie released or the name of actors, are used as the labels of the taxonomic nodes.

VI. CONCLUSIONS AND FUTURE WORKS

Although many techniques of Automated Taxonomy Generation have been applied successfully to categorize document collections into a hierarchical structure for the purpose of facilitating people's navigation, they are not applicable for processing relational datasets due to their unaffordable computational complexity. In this paper we propose a new approach, based on the multi-type relational clustering algorithm *DIVA*, to automatically construct taxonomies for relational datasets. By incorporating the idea of Representative Object, our approach has linear time complexity. A heuristic method is used to determine the cutting levels within the derived taxonomy and optimize the hierarchical structure accordingly. Comprehensive experiments conducted on synthetic and real datasets show that our approach can build very compact taxonomies compared with traditional agglomerative or partitional approaches. The quality of the taxonomies derived by our approach are comparable or even better than that of the traditional approaches. In the future, we will continue to investigate more methods to determine the optimal structure of the derived taxonomy, and also explore other criterion to better evaluate the quality of the taxonomy by synthesizing all the desired properties.

REFERENCES

[1] R. A. Baeza-Yates and B. A. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

[2] D. Boley. Hierarchical taxonomies using divisive partitioning. Technical Report TR-98-012, Department of Computer Science, University of Minnesota, Minneapolis, 1998.

[3] P.-J. Cheng and L.-F. Chien. Auto-generation of topic hierarchies for web images from users' perspectives. In *Proceedings of ACM CIKM Conference*, 2003.

[4] S.-L. Chuang and L.-F. Chien. Towards automatic generation of query taxonomy: A hierarchical query clustering approach. In *Proceedings of ICDM'02*, pages 75–82, USA, 2002.

[5] P. Cimiano, A. Hotho, and S. Staab. Comparing conceptual, partitional and agglomerative clustering for learning taxonomies from text. In *Proceedings of ECAI'04*, pages 435–439. IOS Press, 2004.

[6] P. Clerkin, P. Cunningham, and C. Hayes. Ontology discovery for the semantic web using hierarchical clustering. In *Proceedings of Semantic Web Mining Workshop co-located with ECML/PKDD*, Germany, 2001.

[7] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience Publication, 2001.

[8] D. H. Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2(2):139–172, 1987.

[9] M. A. Gluck and J. E. Corter. Information, uncertainty, and the utility of categories. In *Proceedings of the 7th Annual Conference on Artificial Intelligence*, pages 831–836, Detroit, USA, 1985.

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

[11] V. Kashyap, C. Ramakrishnan, C. Thomas, and A. Sheth. TaxaMiner: an experimentation framework for automated taxonomy bootstrapping. *International Journal of Web and Grid Services*, 1(2):240–266, 2005.

[12] R. Krishnapuram and K. Kummamuru. Automatic taxonomy generation: Issues and possibilities. In *Proceedings of Fuzzy Sets andSystems (IFSA)*, LNCS 2715, pages 52–63. Springer-Verlag Heidelberg, 2003.

[13] K. Kummamuru, R. Lotlikar, S. Roy, K. Singal, and R. Krishnapuram. A hierarchical monothetic document clustering algorithm for summarization and browsing search results. In *Proceedings of WWW'04*, 2004.

[14] D. Lawrie, W. B. Croft, and A. Rosenberg. Finding topic words for hierarchical summarization. In *Proceedings of the 24th ACM SIGIR Conference*, pages 349–357, New York, NY, USA, 2001. ACM.

[15] D. J. Lawrie and W. B. Croft. Generating hierarchical summaries for web searches. In *Proceedings of the 26th ACM SIGIR Conference*, pages 457–458, New York, NY, USA, 2003. ACM.

[16] T. Li and S. S. Anand. DIVA: A variance-based clustering approach for multi-type relational data. In *Proceedings of the 16th ACM CIKM Conference*, Lisboa, Portugal, 2007.

[17] A. Muller, J. Dorre, P. Gerstl, and R. Seiffert. The TaxGen framework: Automating the generation of a taxonomy for a large document collection. In *Proceedings of the Int'l Conference on System Sciences*, 1999.

[18] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *Proceedings of KDD Workshop on Text Mining*, 2000.

[19] X. Yin, J. Han, and P. S. Yu. LinkClus: efficient clustering via heterogeneous semantic links. In *Proceedings of the 32nd VLDB Conference*, pages 427–438. VLDB Endowment, 2006.

[20] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: an efficient data clustering method for very large databases. In *Proceedings of 1996 ACM SIGMOD Conference*, pages 103–114, Montreal, Canada, 1996.

[21] Y. Zhao and G. Karypis. Evaluation of hierarchical clustering algorithms for document datasets. In *Proceedings of the 11th CIKM Conference*, pages 515–524, New York, NY, USA, 2002. ACM.