Iterative Entity Navigation via Co-clustering Semantic Links and Entity Classes

Liang Zheng, Jiang Xu, Jidong Jiang, Yuzhong Qu(✉), and Gong Cheng

National Key Laboratory for Novel Software Technology,
Nanjing University, Nanjing 210023, People's Republic of China
{zhengliang,jiangxu,jdjiang}@smail.nju.edu.cn, {yzqu,gcheng}@nju.edu.cn

Abstract. With the increasing volume of Linked Data, the diverse links and the large amount of linked entities make it difficult for users to traverse RDF data. As semantic links and classes of linked entities are two key aspects to help users navigate, clustering links and classes can offer effective ways of navigating over RDF data. In this paper, we propose a co-clustering approach to provide users with iterative entity navigation. It clusters both links and classes simultaneously utilizing both the relationship between link and class, and the intra-link relationship and intra-class relationship. We evaluate our approach on a real-world data set and the experimental results demonstrate the effectiveness of our approach. A user study is conducted on a prototype system to show that our approach provides useful support for iterative entity navigation.

Keywords: Entity navigation · Semantic link · Entity class · Co-clustering

1 Introduction

With the enrichment of available Linked Data on the Web, challenges in navigating the data space arise: large numbers of linked entities and high diversity of links, often make it hard for users to explore and find the entities of interest quickly. As semantic links and classes of linked entities are two key aspects to help users navigate, clustering links and classes may provide effective organizations about large numbers of links and classes.

As shown in Fig. 1, `Steven Spielberg` in DBpedia [2] is linked to 4 entities (`Falling Skies`, `Men in Black`, `A.I.` and `Medal of Honor`) through 3 semantic links (`executive producer`, `producer` and `writer`). Links can be clustered, such as {`producer`, `writer`} (based on the common linked entities they are linked to) and {`executive producer`, `producer`} (based on lexical similarity between their labels). 4 linked entities are associated with 3 entity classes (`Television Show`, `Film` and `Video Game`). Classes can also be clustered, such as {`Television Show`, `Film`} according to semantic similarity. Link clusters and class clusters offer effective organizations and provide a overview of overall information during users' navigation. However, link clusters and class clusters are utilized separately, and potential relationships between them are not taken into account.

H. Sack et al. (Eds.): ESWC 2016, LNCS 9678, pp. 168–181, 2016.
DOI: 10.1007/978-3-319-34129-3_11

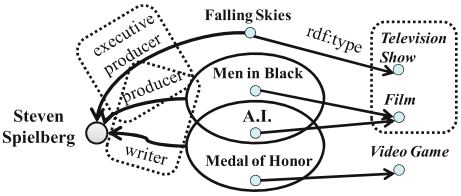

Fig. 1. The context of browsing an entity

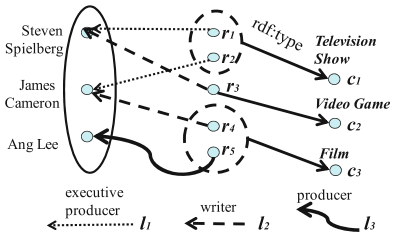

Fig. 2. The context of browsing a set of entities

In order to improve the efficiency of navigation, jointly utilizing link clusters and class clusters may provide users with an iterative refinement mode during entity navigation. For instance, through link cluster {`producer`, `writer`}, users find 3 linked entities (`Men in Black`, `A.I.` and `Medal of Honor`). With respect to the 3 linked entities, there are two class clusters ({`Film`} and {`Video Game`}). Then, users can locate `Medal of Honor` by using class cluster {`Video Game`}. This iterative navigation process can assist users to explore and understand the overall information space. Besides, there is a necessity for navigation paradigm to take into account not only single-entity-oriented transition, but also entity-set-oriented transition. Figure 2 shows the context of browsing the three best director academy award winners. We can capture the strength of the relationship between a link and a class. For instance, 2 directors (`Steven Spielberg` and `James Cameron`) are the `executive producer` of `Television Show`. These rich and meaningful inter/intra relationships among links and classes could be leveraged to improve entity navigation.

In this paper, we propose a co-clustering approach that organizes semantic links and entity classes to support iterative entity navigation. For a given context of entity browsing, we propose a notion of navigation graph to model the three aspects of navigation (links, linked entities and their classes) based on tripartite graph [12]. Then, links and classes are clustered simultaneously based on information theoretic co-clustering (ITCC) [4] over navigation graph. To improve the effect of co-clustering, we define a measure of intra-link similarity and intra-class similarity and incorporate them into ITCC.

The remainder of the paper is structured as follows. Section 2 defines the basic notion to be used and describes our co-clustering problem. Section 3 introduces similarity measuring scheme and our proposed approach. Section 4 provides our experimentation. The related work is reported in Sect. 5. Section 6 concludes this paper.

2 Problem Statement

In this section, we define the notion of navigation graph based on tripartite graph [12] to model the three aspects of navigation (links, linked entities and

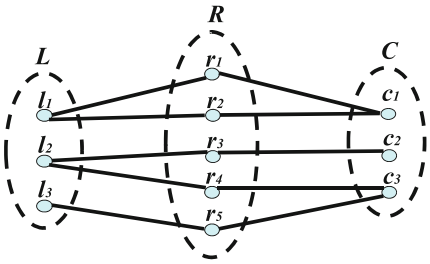

Fig. 3. An example of navigation graph.

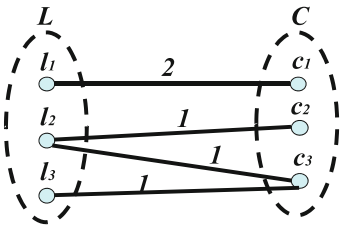

Fig. 4. An example of link-class graph.

their classes) over an RDF graph and introduce the problem of co-clustering semantic links and entity classes.

In this paper, we do not consider literal and blank node in the RDF data model. Let U be the set of URI named entities and classes, L be a set of links including object properties and inverse of them and $T \subseteq U \times L \times U$ be a set of triples.

Definition 1 (Navigation Graph). *Given a set of entities $S \subseteq U$ being the focus, a navigation graph $G =< H, E >$ consists of a set of vertices $H = L' \cup R \cup C$ where $L' \subseteq L$ denotes a set of links, $R \subseteq U$ a set of linked entities and $C \subseteq U$ a set of classes, and a set of edges $E = \{(l, r) \mid \exists\ s \in S,\ (s, l, r) \in T\} \bigcup \{(r, c) \mid (r, rdf : type, c) \in T\}$.*

Suppose the user explores 3 best director academy award winners, Fig. 3 shows the navigation graph associated with Fig. 2. Furthermore, there are three bipartite graphs deduced from navigation graph G. These three graphs can model the associations between links and entities (graph LR), classes and entities (graph CR) and links and classes (graph LC).

Definition 2 (Link-Class Graph). *Given a navigation graph $G =< H, E >$, a weighted bipartite graph LC can be defined as follows: $LC =< V, E_{lc} >$ where $V = L' \cup C$ and $E_{lc} = \{(l, c)\ |\exists\ r \in R,\ (l, r) \in E \wedge (r, c) \in E\}$. $w : E_{lc} \rightarrow N^+$, $\forall e \in E_{lc},\ w(e) = |\{s|\ \exists\ s \in S,\ (s, l, r) \in T \wedge (r, rdf : type, c) \in T\}|$.*

There is an example of link-class graph LC derived from navigation graph G, as shown in Fig. 4. $w(l_1, c_2) = 2$ represents that 2 directors are the `executive producer` of `Television Show`.

Problem 1 (Links-Classes Co-clustering). *Given a navigation graph $G =< H, E >$ and a similarity function sim, find k disjoint clusters of links $\{\hat{x}_1, \hat{x}_2, ..., \hat{x}_k\}$ and l disjoint clusters of classes $\{\hat{y}_1, \hat{y}_2, ..., \hat{y}_l\}$ such that $\sum_{i=1}^{k} \sum_{x,x' \in \hat{x}_i} sim(x, x')$ and $\sum_{j=1}^{l} \sum_{y,y' \in \hat{y}_j} sim(y, y')$ are maximized.*

Clearly the similarity function is a key facet in solving this problem and we define a measuring scheme in Sect. 3.1.

3 Co-clustering Links and Classes

In this section, we define a measuring scheme to compute the link similarity and class similarity in Sect. 3.1. Then we give a solution to the co-clustering problem based on information theoretic co-clustering (ITCC) [4] in Sect. 3.2. Finally we introduce a method for cluster labeling in Sect. 3.3.

3.1 Measuring Similarity

In our work, we focus on cosine similarity, lexical similarity and semantic similarity.

Cosine Similarity. As described in Sect. 2, we use the bipartite graph LC to model the link and class collections. First, link and class can be represented by each other based on vector space model [11]. Each class c can be modeled as a vector over the set of links. Likewise, each link l can be modeled as a vector over the set of classes. For example, as shown in Fig. 4, $c_1 = (2, 0, 0)$, $c_2 = (0, 1, 0)$ and $c_3 = (0, 1, 1)$. Then, the class similarity is defined as the cosine function of angle between two vectors of c_i and c_j:

$$sim_{cos}(c_i, c_j) = \frac{c_i \cdot c_j}{||c_i|| \cdot ||c_j||} \tag{1}$$

and the value of sim_{cos} is in the range (0, 1]. Likewise, the cosine similarity between links can be captured according to the above method.

Lexical Similarity. The label of a link or a class is useful for human understanding. When the labels of two links or classes share many common words, it may indicate some kind of similarity between them. Given two classes c_i and c_j, the lexical similarity is defined as the edit distance between two strings of c_i's label (s_{c_i}) and c_j's label (s_{c_j}):

$$sim_{edit}(c_i, c_j) = \frac{1}{1 + editDist(s_{c_i}, s_{c_j})} \tag{2}$$

where the value of sim_{edit} is in the range (0, 1] and $editDist(s_{c_i}, s_{c_j})$ is the minimum number of character insertion and deletion operations needed to transform one string to the other [9]. Likewise, the lexical similarity between links can be captured according to the above method.

Semantic Similarity. In our work, semantic similarity refers to similarity between two concepts in a taxonomy. Due to a hierarchical structure in many taxonomies, we adopt a similarity measure based on path lengths between concepts [15]. The semantic similarity between class c_i and c_j is calculated as:

$$sim_{sem}(c_i, c_j) = \frac{2 \cdot depth(LCA)}{depth(c_i) + depth(c_j)} \tag{3}$$

where the value of sim_{sem} is in the range (0, 1] and $depth(c_i)$ is the shortest path length from the root to c_i and LCA is the least common ancestor of c_i and c_j. The semantic similarity between semantic links can be also computed according to the above method.

Combination of Similarity. We discuss three similarity measures from different points of view. In order to get the total similarity, we combine these measures based on a natural way (linear combination). Thus, we define the similarity scoring function between two classes c_i and c_j as follows.

$$sim(c_i, c_j) = \alpha \cdot sim_{cos}(c_i, c_j) + \beta \cdot sim_{edit}(c_i, c_j) + \gamma \cdot sim_{sem}(c_i, c_j) \quad (4)$$

where $\alpha + \beta + \gamma = 1$ and $\alpha, \beta, \gamma \in [0, 1]$ indicate the weights for each similarity measure to be tuned empirically.

Also, the similarity between two links l_i and l_j can be captured according to the above method.

3.2 Information-Theoretic Co-clustering Links and Classes

In [4], Dhillon et al. define co-clustering as a pair of maps from rows to row-clusters and from columns to column-clusters based on information theory. The optimal co-clustering is one that minimizes the difference ("loss") in mutual information between the original random variables and the mutual information between the clustered random variables.

Let $X = \{x_1, ..., x_m\}$ and $Y = \{y_1, ..., y_n\}$ be discrete random variables respectively. Let $p(X, Y)$ denote the joint probability distribution between X and Y. Let the k disjoint clusters of X be written as: $\hat{X} = \{\hat{x}_1, \hat{x}_2, ..., \hat{x}_k\}$, and the l disjoint clusters of Y be written as: $\hat{Y} = \{\hat{y}_1, \hat{y}_2, ..., \hat{y}_l\}$. An optimal co-clustering minimizes the loss of mutual information, defined as

$$I(X; Y) - I(\hat{X}; \hat{Y}) = D(p(X, Y)||q(X, Y)) \quad (5)$$

where $I(X; Y)$ is the mutual information between sets X and Y, $D(\cdot||\cdot)$ denotes the Kullback-Leibler(KL) divergence, and $q(X, Y)$ is a distribution of the form $q(x, y) = p(\hat{x}, \hat{y})p(x|\hat{x})p(y|\hat{y})$, $x \in \hat{x}$, $y \in \hat{y}$.

In our context, the link set L' can be considered as X and the classes C as Y. The joint probability distribution between links and classes can be captured based on entity set R over navigation graph $G =< L' \cup R \cup C, E >$. The joint probability of a link x and a class y is defined as follows:

$$p(x, y) = \frac{|\{r|\exists r \in R, (x, r) \in E\} \cap \{r|\exists r \in R, (r, y) \in E\}|}{|R|} \quad (6)$$

To improve co-clustering, many research studies utilize the intra-relationships (e.g., interdocument similarity, interword similarity) [14]. We take the similarity

between links, and the similarity between classes as the intra-relationships, and incorporate them into the information theoretic co-clustering. We rewrite Eq. (5) as

$$I(X;Y) - I(\hat{X};\hat{Y}) - \lambda LCS - \mu CCS \tag{7}$$

$$LCS = \frac{1}{k}\sum_{i=1}^{k}\sum_{x,x' \in \hat{x_i}} \frac{sim(x,x')}{|\hat{x_i}| * (|\hat{x_i}| - 1)} \tag{8}$$

$$CCS = \frac{1}{l}\sum_{j=1}^{l}\sum_{y,y' \in \hat{y_j}} \frac{sim(y,y')}{|\hat{y_j}| * (|\hat{y_j}| - 1)} \tag{9}$$

LCS and CCS are the total similarity within link clusters and class clusters respectively. $\lambda + \mu = 1$ and $\lambda, \mu \in [0, 1]$ indicate the weights for the trade off among the loss of mutual information, LCC and CCS.

In our implementation, we use the ITCC algorithm with time complexity $O((nz(k+l) + km^2 + ln^2)\tau)$ provided by [14], where nz is the number of non-zeros in $p(X, Y)$ and τ is the number of iterations.

3.3 Making the Clusters Easy to Browse

To help user decide at a glance whether the contents of a cluster are of interest, we aim to provide concise and accurate cluster description. A heuristic method is to find the cluster's centroid which is most similar with other elements in the cluster as the label of cluster.

Given a class cluster $\hat{y}$ and a class $y \in \hat{y}$, we define the centricity of y as follows:

$$centricity(y) = \sum_{y' \in \hat{y}} sim(y, y'). \tag{10}$$

Thus, the centroid of cluster $\hat{y}$ can be defined as follows:

$$centroid(\hat{y}) = \arg\max_{y \in \hat{y}} centricity(y) \tag{11}$$

Also, the link cluster's centroid can be captured according to the above method.

4 Evaluation

In this section, we evaluate our approach compared with three baseline algorithms on real-world datasets. Our proposed approach is implemented in a prototype system and then compared with two Linked Data browsers via a user study.

Table 1. Statistics of experimental datasets

	Artist	City	Company	University
Number of entities	1233	2243	304	510
Number of links	139	280	174	163
Number of linked entities	59654	402580	88003	57487
Average num. of links per entity	25.8	30.1	27.2	28.9
Average num. of linked entities per entity	113.2	217.5	338.1	151.7
Average num. of classes per linked entity	6.9	8.7	5.8	8.9

4.1 Experimental Evaluation

Datasets. we used the DBpedia (version:2015-04)[1] *Mapping-based Properties* dataset, excluding RDF triples containing literals. We selected 4 common classes (i.e., Artist, City, Company, University). For each class, we collected those entities that each one has more than 15 semantic links. As to entity classes, we used the *Mapping-based Types* dataset. For the purpose of our task, we used the *DBpedia Ontology* dataset. The statistical results of these datasets are listed in Table 1.

Baselines. We compare our method (ITCC+) with three baselines (ITCC [4], co-clustering via bipartite spectral graph partition(BSGP) [3] and K-means [10]).

ITCC only focuses on the relationship between row and column in the co-occurrence matrix from mutual information aspect but neglects the intra-row and intra-column relationships.

BSGP considers the co-clustering problem in term of finding minimum cut vertex partitions in a weighted bipartite graph. In our context, the link-class bipartite graph $LC =< L' \cup C, E_{lc} >$ can be captured. We use the edge weighting method provided by [5] and the weight between a link l and a class c is computed with

$$w(l,c) = \max_{c_i \in l} sim(c_i, c) + \max_{l_j \in c} sim(l, l_j) \tag{12}$$

where $l = \{c_i | \exists c_i \in C, (l, c_i) \in E_{lc}\}$ and $c = \{l_j | \exists l_j \in L', (l_j, c) \in E_{lc}\}$.

K-means measures the distance between two data points according to the similarity by Eq. (4). Since K-means is a one-sided clustering algorithm, the link and class collections are clustered separately.

Evaluation Metrics. We define three metrics (*cohesion*, *separation* and *overall*) to measure the quality of clustering. Given a cluster set $O = \{O_1, ..., O_k\}$ of N elememts, the three metrics of O are defined as follows.

$$cohesion(O) = \frac{1}{k} \sum_{i=1}^{k} coh(O_i), \; coh(O_i) = \frac{\sum_{o \in O_i, o' \in O_i} sim(o, o')}{|O_i| \cdot (|O_i| - 1)}. \tag{13}$$

[1] http://wiki.dbpedia.org/Downloads2015-04.

Table 2. Comparsion of *overall* against different k and (α, β, γ)

		t_1	t_2	t_3	t_4	t_5	t_6
k=3	ITCC+	**0.874**	0.566	0.401	**0.467**	**0.911**	0.642
	ITCC	0.872	0.377	**0.486**	0.273	0.688	**0.729**
	BSGP	0.835	0.226	0.179	0.238	0.465	0.43
	K-means	0.852	**0.643**	0.463	0.356	0.673	0.577
k=5	ITCC+	**0.891**	0.566	**0.587**	**0.428**	**0.925**	0.535
	ITCC	0.887	0.384	0.427	0.372	0.749	0.471
	BSGP	0.797	0.273	0.209	0.193	0.522	0.383
	K-means	0.814	**0.762**	0.479	0.383	0.765	**0.61**
k=8	ITCC+	**0.857**	**0.672**	**0.52**	0.433	**0.887**	0.668
	ITCC	0.832	0.359	0.497	0.324	0.886	**0.729**
	BSGP	0.686	0.238	0.419	0.273	0.596	0.365
	K-means	0.823	0.663	0.478	**0.481**	0.829	0.649

$$separation(O) = \frac{1}{k}\sum_{i=1}^{k} sep(O_i),\ sep(O_i) = \frac{\sum_{o\in O_i, o'\notin O_i} sim(o, o')}{|O_i| \cdot (N - |O_i|)}. \quad (14)$$

$$overall(O) = 1 - \frac{separation(O)}{cohesion(O)}. \quad (15)$$

Obviously, the higher the overall score the better the clustering quality is.

Results and Discussions. In the experiments, we randomly selected 200 entities from our experimental dataset. For each selected entity, we empirically conducted 10 runs using four algorithms (ITCC+, ITCC, BSGP and K-means) respectively and reported the average *overall*.

For parameter settings, we investigated the sensitivity with respect to the disjoint clusters size k (=3, 5, 8) and the balanced parameters (α, β, γ, λ, μ). As to the parameter (α, β, γ) in similarity Eq. (4), we set $t_1 = (1, 0, 0)$, $t_2 = (0, 1, 0)$, $t_3 = (0, 0, 1)$, $t_4 = (0.33, 0.33, 0.33)$, $t_5 = (0.6, 0.1, 0.3)$ and $t_6 = (0.3, 0.1, 0.6)$ in computing link similarity and class similarity. As to the parameters (λ, μ) of loss in mutual information Eq. (7), we set $\lambda = \mu = 0.5$ based on equity. More parameters settings will be experimented in future work. Besides, we set the number of iterations $\tau = 20$ which is enough for convergence [4].

From Table 2, we have the following observations. In most cases, ITCC+ outperforms ITCC, BSGP and K-means. Since the core of ITCC+ and ITCC depends on the joint probability distribution between links and classes, ITCC+ and ITCC can bring better results when the cosine similarity has higher weight (e.g., t_1 and t_5).

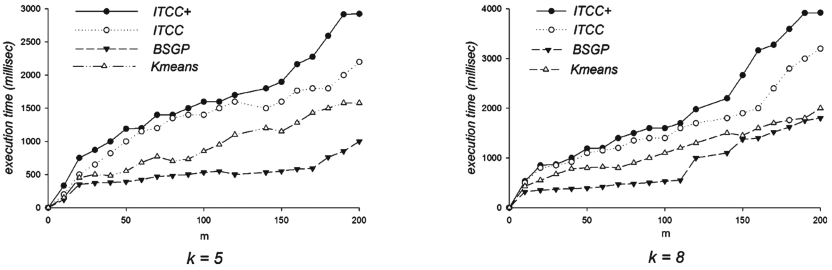

Fig. 5. Execution time of four algorithms with varying k and m.

Efficiency Evaluation. We evaluated the performance of ITCC+, ITCC, BSGP and K-means by measuring the average execution time for varying size of entities denoted by m. We randomly selected entities from our experimental dataset. The four algorithms were implemented in Java and carried out on an Intel Core2 Quad 2.66 GHz CPU, Windows 7 with 1.2 GB JVM.

As can be seen from Fig. 5, the four algorithms were reasonably fast in practice. BSGP was faster than ITCC+, ITCC and K-means. ITCC+ and ITCC need multiple iterations and recompute the distributions on every iteration. Besides, ITCC+ compute the link cluster similarity and class cluster similarity respectively on every iteration.

4.2 User Study

We conducted a user study to compare our approach with two Linked Data browsers (Rhizomer[2], SView[3]) and to evaluate the effectiveness of our approach.

Participant Systems. We implemented our proposed approach in a prototype system called CoClus[4], as shown in Fig. 6. Users can start browsing with an entity URI by entering into the input box (A). The left-hand side of the interface lists the link clusters (B). The right-hand side lists class clusters (C). There are connections between link clusters and class clusters (D). Users can click the button "browse all" to explore all the linked entities (E). Also, users can choose some link/class clusters to iteratively filter the target entities. Selected filters can be cancelled (F).

Rhizomer provides users with facet navigation to explore RDF data in DBpedia. Once users have zoomed by selecting the kind of entities from the navigation bar, facets are generated automatically and help users filter out those that are not interesting.

[2] http://rhizomik.net/html/rhizomer/.
[3] http://ws.nju.edu.cn/sview/.
[4] http://ws.nju.edu.cn/coclus/.

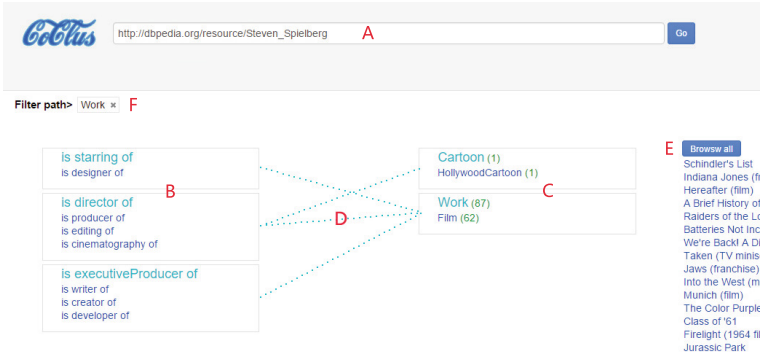

Fig. 6. A screenshot of CoClus.

Table 3. An example of navigation tasks about `John Lennon`

		Tasks
G1	*E1*	Explore the information related to `John Lennon`, and describe three main aspects of him
	F1	Find the albums written by `John Lennon`
G2	*E2*	Explore the information related to the band members of the Beatles, and describe three main aspects of them
	F2	Find the films starred by the band members of the Beatles

SView provides a navigation module (called "Link") which organizes the semantic links in the form of link patterns [16]. It supports a hierarchical and interactive interface to help users to find target linked entities.

Tasks. We selected 8 entities from experimental dataset in Sect. 4.1 at random as the starting points of user navigation. In a browsing scenario, navigation tasks can be divided into two types: *Explore* (a user has a fuzzy need) and *Find* (a user has a clear need) tasks. According to navigation paradigm, tasks can also be divided into two groups: single-entity-oriented (*G1*) and entity-set-oriented (*G2*) tasks. For each starting point, we established 4 navigation tasks. The navigation tasks about `John Lennon` are shown in Table 3.

Procedure. We invited 12 student subjects majoring in computer science who were familiar with the Web, but with no knowledge of our project. The evaluation was conducted in three phases. First, the subjects learned how to use the given systems through a 20 min tutorial, and had additional 10 min for free use and questions. Second, the subjects used each of the three systems arranged in random order. For each system, the subjects were randomly assigned to one starting point, and required to complete 4 navigation tasks. Meanwhile, the starting points of user navigation among the three systems were different. To enable

Table 4. Navigation questionnaire

	Questions
Q1:	The system helped me get an overview of all the information
Q2:	The number of navigation options was overwhelming
Q3:	The navigation options were well organized
Q4:	The navigation option titles were understood well
Q5:	It was easy to reorient myself in the navigation

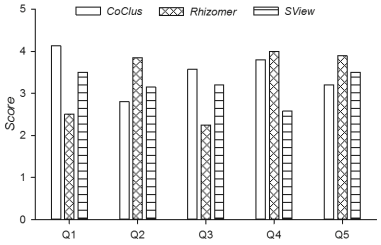

Fig. 7. Results of navigation questionnaire.

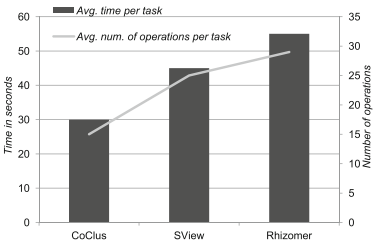

Fig. 8. Average time and number of operations for each task.

users to use the systems (e.g., Rhizomer is used to browse a whole dataset in a faceted manner) for carrying out such a task, prior to testing, we navigate to the desired starting point, from which users start their tasks. The subjects were asked to complete all the tasks in 10 min. We recorded their answers, their operations and the time they spent on each task.

With regard to each system, the subjects responded to the navigation questionnaire, as shown in Table 4. The questions in the questionnaires were responded by using a five-point Likert scale ranging from 1 (strongly disagree) to 5 (strongly agree). Finally, the subjects were asked to comment on the three systems.

Results and Discussions. Navigation questionnaire Q1–Q5 captured subjects' navigation experience with different systems in Fig. 7. Repeated measures ANOVA revealed that the differences in subjects' mean ratings were all statistically significant ($p < 0.01$). LSD post-hoc tests ($p < 0.05$) revealed that, according to Q1 ($F = 11.543$, $p = 0.0$), CoClus ($mean = 4.18$, $sd = 0.54$) provided a better overview of all the information than SView ($mean = 3.8$, $sd = 1.056$) and Rhizomer ($mean = 2.75$, $sd = 0.85$) due to the use of link and class clusters. According to Q2 ($F = 9.56$, $p = 0.0$), Rhizomer ($mean = 3.98$, $sd = 1.06$) provided too many faceted filtering views and links compared with CoClus ($mean = 2.85$, $sd = 0.56$) and SView ($mean = 3.42$, $sd = 1.04$). According to Q3 ($F = 13.65$, $p = 0.002$), CoClus ($mean = 3.78$, $sd = 0.96$) provided a better organization of links and classes than SView ($mean = 3.41$, $sd = 0.83$)

and Rhizomer ($mean = 2.4$, $sd = 0.61$). According to Q4 ($F = 10.36$, $p = 0.0$), Rhizomer ($mean = 4.12$, $sd = 1.03$) helped subjects more easily understand the label of links. Finally, according to Q5 ($F = 11.67$, $p = 0.0$), Rhizomer ($mean = 4.08$, $sd = 0.97$) helped subjects keep track of browsing and provided easy rollback.

Figure 8 shows the average time spent on each task and the number of operations which participants had conducted in one task. Since CoClus provided a overview using link clusters and class clusters, subjects required far less time and fewer interactions to complete these tasks.

We also summarized all the major comments of the subjects. As to Rhizomer, 10 subjects (83 %) said that faceted navigation helped them filter out those entities that were not interesting but needed them to input manually and multiple interactions. As to SView, 8 subjects (67 %) said that link patterns provided useful support to help users to locate target linked entities, but target entity collection was still large. As to CoClus, 11 subjects (92 %) said that it was distinguished by its co-clustering of links and classes of linked entities, and it offered a different, more useful set of navigation functions. The link and class clusters provided a overview of information and enabled users to navigate iteratively. But 4 subjects (33 %) said that it had some risks, such as misleading information because of some inappropriate cluster labels.

5 Related Work

Navigation as an important feature of Linked Data, has been supported by many Linked Data browsers. Tabulator [1] allows users to browse data by starting from a single resource and following links to other resources. It also allows users to select a resource for further exploration in a nest tree view. gFacet [8] is a tool that supports the exploration of the Web of data by combining graph-based visualization with faceted filtering functionalities. With gFacet it is possible to choose one class and then pivot to a related class keeping those filters for the instances of the second class connected to the filtered instances in the first class. VisiNav [7] is a system based on an interaction model designed to easily search and navigate large amounts of Web data. It provides four atomic operations over object structured datasets: keyword search, object focus, path traversal, and facet specification. Users incrementally assemble complex queries that yield sets of objects. Rhizomer [6] addresses the exploration of semantic data by applying the data analysis mantra of overview, zoom and filter. Users can interactively explore the data using facets. Moreover, facets also feature a pivoting operation. Visor [13] is a generic RDF data explorer that can work over SPARQL endpoints. In Visor, exploration starts by selecting a class of interest from the ontology. Then, users can pivot to related collections and continue browsing. Visor provides a hierarchical overview of the collections and also provides a spreadsheet requiring manual customization to filter the collection.

Whereas the above efforts mainly focus on providing the user with powerful interaction modes, we aim to cluster semantic links and entity classes simultaneously to support iterative entity navigation, which is complementary to all of them.

6 Conclusion

In this paper, we propose a co-clustering approach which clusters both links and classes simultaneously to provide users with an iterative entity navigation. To achieve this, we define a new notion of navigation graph based on tripartite graph to model the three aspects of navigation (links, linked entities and their classes). We also measure the link similarity and the class similarity, and incorporate them into the co-clustering algorithm. The proposed approach is implemented in a prototype system. The evaluation results demonstrate that it provides a better overview of all the information, and supports users' iterative entity navigation.

Acknowledgements. This work is supported in part by the 863 Program under Grant 2015AA015406, in part by the National Science Foundation of China under Grant Nos. 61223003 and 61572247, and in part by the Fundamental Research Funds for the Central Universities. We are also grateful to all the participants in the experiments of this work.

References

1. Berners-lee, T., Chen, Y., Chilton, L., Connolly, D., Dhanaraj, R., Hollenbach, J., Lerer, A., Sheets, D.: Tabulator: exploring and analyzing linked data on the semantic web. In: 3rd International Semantic Web User Interaction Workshop (2006)
2. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia-a crystallization point for the web of data. J. Web Sem. **7**(3), 154–165 (2009)
3. Dhillon, I.S.: Co-clustering documents and words using bipartite spectral graph partitioning. In: 7th International Conference on Knowledge Discovery and Data mining, pp. 269–274 (2001)
4. Dhillon, I.S., Mallela, S., Modha, D.S.: Information-theoretic co-clustering. In: 9th International Conference on Knowledge Discovery and Data mining, pp. 89–98 (2003)
5. Giannakidou, E., Koutsonikola, V.A., Vakali, A., Kompatsiaris, Y.: Co-clustering tags and social data sources. In: 9th International Conference on Web-Age Information Management, pp. 317–324. IEEE (2008)
6. Garcia, R., Gimeno, J.M., Perdrix, F., et al.: Building a usable and accessible semantic web interaction platform. World Wide Web **13**(1–2), 143–167 (2010)
7. Harth, A.: VisiNav: a system for visual search and navigation on web data. J. Web Sem. **8**(4), 348–354 (2010)
8. Heim, P., Ertl, T., Ziegler, J.: Facet graphs: complex semantic querying made easy. In: Aroyo, L., Antoniou, G., Hyvönen, E., Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010, Part I. LNCS, vol. 6088, pp. 288–302. Springer, Heidelberg (2010)
9. Lin, D.: An information-theoretic definition of similarity. In: International Conference on Machine Learning, pp. 296–304 (1998)
10. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In: Fifth Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1967)

11. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to information retrieval. Cambridge University Press, Cambridge (2008)
12. Mika, P.: Ontologies are us: a unified model of social networks and semantics. J. Web Sem. **5**(1), 5–15 (2007)
13. Popov, I.O., Schraefel, M.C., Hall, W., Shadbolt, N.: Connecting the dots: a multi-pivot approach to data exploration. In: Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N., Blomqvist, E. (eds.) ISWC 2011, Part I. LNCS, vol. 7031, pp. 553–568. Springer, Heidelberg (2011)
14. Wu, J.S., Lai, J.H., Wang, C.D.: A novel co-clustering method with intra-similarities. In: 11th International Conference on Data Mining Workshops, pp. 300–306 (2011)
15. Wu, Z., Palmer, M.: Verbs semantics and lexical selection. In: 32nd Annual Meeting on Association for Computational Linguistics, pp. 133–138 (1994)
16. Zheng, L., Qu, Y., Jiang, J., Cheng, G.: Facilitating entity navigation through top-k link patterns. In: 14th International Semantic Web Conference, pp. 163–179 (2015)