Tailoring Linked Data Exploration through inCloud Filtering

ABSTRACT
This paper presents techniques for tailoring linked data exploration by relying on the use of high-level, intuitive organization structures called inClouds (information Clouds). Techniques based on content filtering and relevance filtering are proposed for enabling a user to adjust the inCloud visualization according to her/his own preferences.

Categories and Subject Descriptors
H.5 [Information Systems]: Information Interfaces and Presentation; H.3 [Information Systems]: Information Storage and Retrieval

General Terms
Linked data exploration and filtering

1. INTRODUCTION
Due to the growing success of the linked data paradigm, the problem of effectively exploring and browsing the Linked Data Cloud according to specific selection criteria is getting more and more importance [1]. On one side, SPARQL endpoints and other frameworks like SQUIN and iSPARQL are being developed to provide visual and intuitive interfaces for querying a given linked data repository [4, 5]. On the other side, methods and techniques based on matching and aggregation are being proposed in the literature to overcome the inherent flat organization of linked data repositories by providing advanced exploration functionalities [2, 6].

In this paper, we present techniques for tailoring linked data exploration by relying on inClouds (information Clouds). We introduced the notion of inCloud in [3] to provide a high-level, intuitive organization structure capable of representing at a glance a (generally wide) collection of linked data. In the following, we first recall the definition of inCloud as well as its main featuring properties (Section 2). Then, we present techniques for content-based filtering and relevance-based filtering capable of adjusting the inCloud structure according to the exploration needs specified by a final user (Sections 3 and 4). Techniques for customizing the visualization of the filtering results are also discussed in the paper (Sections 5). Concluding remarks are finally provided (Section 6).

2. LINKED DATA CLOUDS
An inCloud originates from a set $S$ of linked data extracted from a repository $R$ (e.g., Freebase, DBpedia) through a combination of SPARQL queries. Starting from a seed $s$, that is an URI of $R$ chosen by the user as the “point of origin”, the extraction of $S$ consists in determining the set of linked data that are most pertinent to $s$. We select as pertinent resources those linked data that are connected to $s$ through a property path of length $\leq d$, where $d$ is a user-defined distance to set the extension of linked data extraction$^{1}$.

Definition of inCloud. Given a seed $s$, an inCloud is defined as a graph $iC_s = (N, E)$, where a node $n_i \in N$ represents a concept with an associated cluster of similar linked data belonging to $S$ and an edge $(n_i, n_j) \in E$ represents a relation of proximity between $n_i$ and $n_j$, denoting the fact that the two concepts and the respective clusters are somehow related.

A concept $n_i \in N$ is defined as $n_i = (e_i, cl_i)$ where $e_i$ is the essential of $n_i$, namely a concise and convenient summary of the concept, and $cl_i$ is a cluster of linked data of $S$. The essential $e_i$ is defined as a pair $e_i = (K_i, T_i)$, where $K_i$ is a set of keywords and $T_i$ is a set of type names. $K_i$ and $T_i$ are defined over the linked data of $cl_i$ by choosing the most frequently occurring terms and type names in the specification of the linked data of $cl_i$, respectively. To build $cl_i$, techniques for linked data matching are employed to discover all the pairs of similar resources in the set $S$. Then, techniques for linked data aggregation are invoked to determine $cl_i$ by putting together those resources of $S$ that are characterized by a high level of similarity according to the matching results [2]. Each concept $n_i \in N$ is characterized by a prominence $p_i$, which denotes the relative importance of $n_i$ within the overall inCloud. The concept prominence is proportional to the number and the strength of proximity relations holding between $n_i$ and the other concepts of the inCloud. The prominence affects the visual organization of the corresponding concept/cluster of the inCloud, in that the most prominent concepts are highlighted in foreground.

$^{1}$More details about linked data extraction are given in [3].
A proximity relation \( e(n_i, n_j) \) represents a similarity-based relationship between the concepts \( n_i \) and \( n_j \). Proximity relations suggest possible exploration paths across the concepts of the inCloud, thus enabling a user to navigate from one concept to another according to a similarity-based criterion. A proximity relation \( e(n_i, n_j) \) is associated with a degree of proximity \( x_{ij} \) which denotes the strength of the relationship between the concepts \( n_i \) and \( n_j \). The proximity degree \( x_{ij} \) depends on the number of elements that are similar (i.e., matching) across the clusters \( c_l \) and \( c_l' \), respectively. The higher the number of similar elements, the higher the proximity degree \( x_{ij} \).

**Example.** An example of inCloud extracted from the Freebase repository for the seed \( s=/en/italy \) is shown in Figure 1, providing concepts about Italy and some related countries in Europe (e.g., France, Germany). In the figure, we highlight some concepts containing data about cities, regions, tourist attractions, and films. Keywords \( K_t \) and type names \( T_i \) of a concept \( n_t \) are represented as boxes, while a circle with an ID is used to represent the corresponding cluster \( c_l \). For a concept \( n_t \), a portion of the linked data resources contained in the cluster \( c_l \) is also shown. The size of a cluster \( c_l \) can vary from one cluster to another, and it is proportional to the prominence \( p_t \) associated with the concept \( n_t \). A proximity relation \( e(n_i, n_j) \) is represented as a solid line whose thickness is proportional to the degree of proximity \( x_{ij} \) holding between the concepts \( n_i \) and \( n_j \).

### 3. Filtering inClouds by Content

Content-based filtering techniques allow a user to restrict the exploration of a considered inCloud to a subset of concepts that satisfy a given user interest. Such an user interest is expressed through a set of keywords \( Q \) where AND/OR operators are employed to denote conjunctive/disjunctive conditions, respectively. In other words, these techniques transform an initial inCloud \( iC \) into a resulting inCloud \( iC' = (N', E') \) (i.e., \( iC' \) is a subgraph of \( iC \)), where only the concepts/nodes of \( iC' \) that satisfy the specified user interest are inserted in \( N' \) and a proximity link \( e(n_i, n_j) \in E \) is inserted in \( E' \) if \( n_i, n_j \in N' \).

To determine whether a concept \( n_i \in N \) of an inCloud satisfies the user interest, each keyword \( q_i \in Q \) is matched against the corresponding essential \( e_i \). In particular, the user can specify the elements of the essential \( e_i \) to be considered for matching, in that, the set \( K_t \), the set \( T_i \) or both can be exploited in such a matching operation.

Content-based filtering techniques are based on a RETRIEVE instruction defined through the following SQL-like syntax.

```sql
RETRIEVE expr
FROM K | T | K,T
```

where the RETRIEVE clause specifies the conjunctive/disjunctive expression to search, and the FROM clause specifies the elements of the inCloud essentials upon which the retrieval operation has to be performed.

- \( expr \): \( q_i \mid (expr \ OR \ expr) \mid (expr \ AND \ expr) \) is the list of keywords with AND/OR operators specified by the user\(^2\).
- \( K = \{ \cup_{t=1}^n K_t \} \) is the set of keywords extracted from the concept essentials of the overall inCloud where \( n = |N| \) is the cardinality of \( N \).
- \( T = \{ \cup_{t=1}^n T_i \} \) is the set of type names extracted from the concept essentials of the overall inCloud.

\(^2\)If \( expr \) is not specified, the whole inCloud \( iC \) is returned as a result. In this case \( iC \equiv iC' \).
K, T is the set of keywords and type names of the inCloud essentials, namely K ∪ T.

Example. As an example of filtering-by-content, we consider the inCloud in Figure 1 and we suppose that a user is interested in retrieving contents concerning films located in or depicting the city of Rome. In order to express her/his interest, the user formulates the following RETRIEVE instruction:

RETRIEVE ‘film’ AND ‘rome’
FROM K, T

This RETRIEVE instruction filters the inCloud by selecting those concepts that contain both film and rome in their essentials (i.e., keywords or type names). In particular, the concepts that satisfy the request contain information about films located in Rome (ID=9), ID=12), tourist attractions (ID=120), and other general information about the town (ID=194). The concept about tourist attractions is bigger in size due to its higher prominence with respect to the other concepts in the inCloud.

4. FILTERING INCLOUDS BY RELEVANCE
Relevance-based filtering techniques allow a user to restrict the exploration of an inCloud to those concepts that are more relevant with respect to prominence and proximity requirements. These techniques can perform two different kinds of filtering, namely filtering-on-prominence and filtering-on-proximity, that are expressed through appropriate conditions. Filtering-on-prominence allows to select those concepts ni that have a prominence pi higher than a user-specified threshold. Filtering-on-proximity allows to select those proximity relations e(ni,nj) that have a degree of proximity xij higher than a user-specified threshold. In other words, these techniques transform an incoming inCloud IC’ (produced by the content filtering techniques) into a resulting inCloud IC″ where only the concepts and proximity relations that satisfy the specified conditions on prominence and/or proximity are represented.

Relevance-based techniques are specified in the RETRIEVE instruction through the WHERE clause as follows:

RETRIEVE expr
FROM K | T | K, T
[WHERE [p > valuep][|x > valuex]]

where valuep and valuex are user-specified thresholds to set the minimum level of requested prominence and proximity, respectively.

When the condition p > valuep is specified, a concept ni is included in the resulting IC″ iff the associated prominence value pi is higher than the valuep.

When the condition x > valuex is specified, a proximity relation e(ni,nj) is included in the resulting IC″ iff the associated proximity value xij is higher than valuex. Moreover, each node ni ∈ IC is included in IC″ iff a relation e(ni,nj) exists where nj ∈ IC and the proximity value xij is higher than valuex.

Example. An example of filtering-by-relevance is given in Figure 2(b), where the following WHERE clause is added to the RETRIEVE instruction of section 3:

RETRIEVE ‘film’ AND ‘rome’
FROM K, T
WHERE p > 0.5, x > 0.5

The result of this RETRIEVE instruction is that the concepts of Figure 2(b) are a subset of those represented in Figure 2(a), and some proximity relations are not reported in the resulting inCloud. In particular, we focus on the concept about the city of Rome (id=194), which is filtered out from the results due to its low level of prominence. In this case, the concept is pertinent with respect to the keywords Q specified by the user (i.e., film and rome) but it is not sufficiently prominent with respect to the specified valuep (p194 < 0.5).
5. Displaying inClouds

The visualization of the inCloud generated through filtering-by-content and filtering-by-relevance can be adjusted by the user according to her/his desired level of detail. The user can choose the elements of the inCloud to consider for display. In particular, the inCloud visualization can be focused on showing only the concept essentials of the result, which is a more suitable solution for users interested in learning general knowledge about a certain topic from linked data. Instead, the visualization can be focused on displaying only the content of clusters for more detailed data exploration, which is a more suitable solution for users interested in using inCloud filtering as a query facility for linked data repositories. Finally, the visualization of proximity relations between the concepts can be enabled/disabled to include connections between concepts in the final display.

The display options are specified through a SHOW clause as follows:

```sql
SHOW [essentials[,]] [relations] ON expr
RETRIEVE FROM K | T | K, T
WHERE [p > value[,][x > value,]]
```

The SHOW clause allows to specify the elements to display, namely clusters, essentials and relations. As a final remark, we stress that when only the relations option is specified, the visualization of clusters is also enforced since the only display of proximity links without essentials or clusters does not provide any useful information.

Example. An example of inCloud displaying is provided in Figure 3, where the result of the following SHOW-RETRIEVE instructions is given.

<table>
<thead>
<tr>
<th>Display essentials (a)</th>
<th>Display clusters (b)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHOW essentials, relations ON (</td>
<td>SHOW clusters, relations ON (</td>
</tr>
<tr>
<td>RETRIEVE 'film' AND 'rome' FROM K, T WHERE p &gt; 0.5, x &gt; 0.5)</td>
<td>RETRIEVE 'film' AND 'rome' FROM K, T WHERE p &gt; 0.5, x &gt; 0.5)</td>
</tr>
</tbody>
</table>

In particular, Figure 3 shows the main difference between two different display options. In Figure 3(a), the inCloud view is focused on essentials display to summarize the inCloud contents in terms of keywords and type names. In Figure 3(b), the inCloud view is focused on cluster display to provide a direct link to the underlying resources.

6. Concluding Remarks

In this paper, techniques for linked data exploration through inCloud filtering have been presented together with real examples on the Freebase repository. Ongoing work is devoted to integrate the proposed filtering techniques in the existing clouding Java prototype for inCloud construction and visualization (http://islab.dico.unimi.it/inCloud).

Figure 3: Results of (a) essential displaying, and (b) cluster displaying

7. References