Clouding Services for Linked Data Exploration

Silvana Castano, Alfio Ferrara, and Stefano Montanelli

Università degli Studi di Milano,
DICo - Via Comelico, 39 - 20135 Milano
{silvana.castano,alfio.ferrara,stefano.montanelli}@unimi.it

Abstract. Exploration of linked data aims at providing tools and techniques that enable to effectively explore a dataset through concepts, relationships, and properties by means of SPARQL endpoints and visual interfaces. In this paper, we present a set of clouding services for linked data exploration, to enable the end-user to personalize and focus her/his exploration by interactively configuring high-level conceptual structures called *in*Clouds.

Keywords: Linked data, exploration, clouding services.

1 Introduction

The Linked Data paradigm promoted a new way of exposing, sharing, and connecting pieces of data, information, and knowledge on the Semantic Web, based on URIs (Universal Resource Identifier) and RDF (Resource Description Framework) [1]. However, due to the inherent flat organization of linked data repositories, the user interested in getting data about a certain search target has usually to face a multi-step and loosely-intuitive browsing activity to build a (more or less) comprehensive picture of the data of interest [5,7,13].

In this paper, we address the exploration of linked data, namely the activity aiming at providing tools and techniques that enable to effectively explore a dataset through concepts, relationships, and properties by means of SPARQL endpoints and visual interfaces. We define a set of clouding services to enable the end-user to personalize and focus her/his exploration by interactively configuring high-level conceptual structures called *in*Clouds. *in*Clouds and associated clouding services allow the exploration of underlying data through i) *concepts*, that are representative of sets of linked data built through similarity-based clustering and that can be browsed and dynamically reconfigured upon user request; ii) *proximity relations*, that express the existence of a similarity relationship between concepts and related clusters and that are used as navigation paths to browse and combine data pertaining to different but related concepts; iii) *ranking properties*, expressed in form of prominence values associated with concepts and proximity values associated with relations, that enable the user to intuitively focus on the most relevant concepts and their most significant relations to explore the underlying data.

J. Ralyté et al. (Eds.): CAiSE 2012, LNCS 7328, pp. 486–501, 2012.

After introducing the main features of *in*Clouds and their life-cycle (Section 2), the paper focuses on describing the clouding services to support the initial bootstrapping and the subsequent interactive tailoring of *in*Clouds for enabling more effective and focused exploration modalities (Sections 3, 4, and 5). Related work (Section 6) and concluding remarks (Section 7) finally close the paper.

2 Linked Data Clouds

To go beyond the flat organization of linked data repositories, in [4], we introduced the *in*Cloudas a high-level, intuitive data structure capable of representing at a glance a (generally wide) collection of linked data. After briefly recalling the main features of *in*Clouds, we then focus on the main contribution of the paper, namely on the definition of the *in*Cloud life-cycle and associated clouding services.

2.1 The inCloud Structure

An *in*Cloud originates from a set $\mathcal{S}$ of linked data extracted from a repository $\mathcal{R}$ (e.g., Freebase[1], DBpedia[2]) through a combination of SPARQL queries starting from a *seed* s, that is an URI of $\mathcal{R}$ chosen by the user as the "point of origin" for linked data extraction (see Section 2.2).

Definition 1. *in*Cloud. Given a seed s, an *in*Cloud 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 $\mathcal{S}$ and an edge $e(n_i, n_j) \in E$ 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.

In the *in*Cloud graph, a *concept* $n_i \in N$ is defined as $n_i = (K_i, T_i, cl_i)$ where K_i is a set of keywords, T_i is a set of types, and cl_i is a cluster of linked data, respectively. The cluster cl_i contains a subset of the linked data in $\mathcal{S}$ and it is built through aggregation of those linked data that are similar according to a considered matching function. K_i and T_i are defined over the linked data of cl_i by choosing the most frequently occurring terms and types in the specification of the linked data of cl_i, respectively. Each concept $n_i \in N$ is characterized by a *prominence* p_i, which denotes the relative importance of n_i within the overall *in*Cloud. The concept prominence is proportional to the number and the strength of proximity relations holding between n_i and the other concepts of the *in*Cloud. The prominence affects the visual organization of the corresponding concept/cluster of the *in*Cloud, in that the most prominent concepts are highlighted in foreground.

A *proximity relation* $e(n_i, n_j)$ represents a similarity-based relationship between the concepts n_i and n_j. The nature of the proximity relation depends on the matching function employed for similarity evaluation. For instance,

[1] `http://www.freebase.com/`
[2] `http://dbpedia.org/`

when geo-spatial properties are considered to calculate linked data similarity, a proximity relation $e(n_i, n_j)$ denotes that the concepts n_i and n_j contain a number of geographically close elements in their respective clusters cl_i and cl_j. 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 cl_i and cl_j, respectively. The higher the number of similar elements, the higher the proximity degree x_{ij}[3]. Proximity relations suggest possible exploration paths across the concepts of the *in*Cloud, thus enabling a user to navigate from one concept to another following a similarity-based criterion.

Example. An example of *in*Cloud 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 clustering data about cities, regions, tourist attractions, and films. In Figure 1, keywords K_i and types T_i of a concepts n_i are represented as boxes, while a circle with an ID is used to represent the corresponding cluster cl_i. The circle size of a cluster cl_i can vary from one cluster to another, and it is proportional to the prominence p_i associated with the concept n_i. 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.

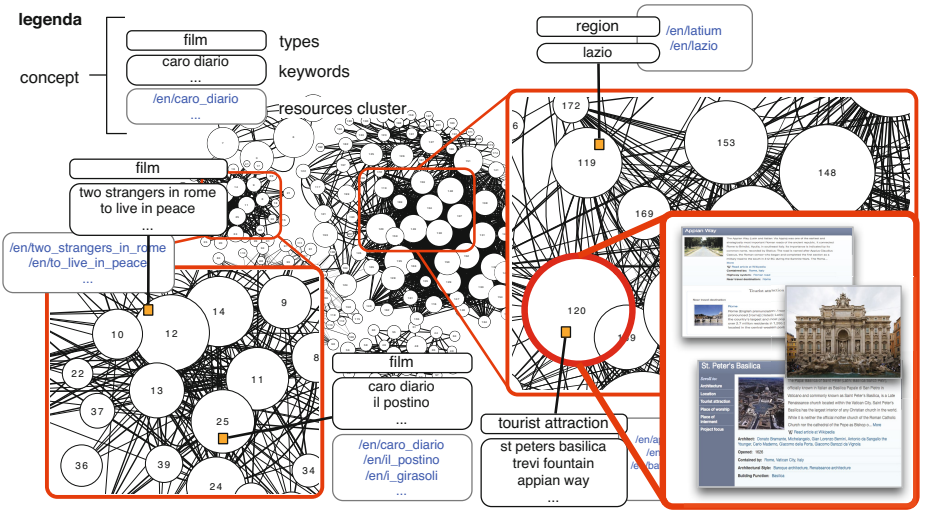

Fig. 1. An example of *in*Cloud extracted from the Freebase repository for the seed $s =$/en/italy

[3] A proximity relation $e(n_i, n_j)$ and its degree of proximity x_{ij} are defined between concepts n_i and n_j and are calculated over the matching elements of the corresponding clusters cl_i and cl_j.

For a concept n_i, a portion of the linked data resources contained in the cluster cl_i is also shown. For instance, the concept n_{120} of Figure 1 is described by the keyword set $K_{120} = \{\text{st peters basilica, trevi fountain, appian way}\}$, the the type set $T_{120} = \{\text{tourist attraction}\}$, and the cluster cl_{120}.

2.2 The inCloud Life-Cycle

An *in*Cloud is characterized by a two-phase life-cycle based on *bootstrapping* and *tailoring* activities (see Figure 2).

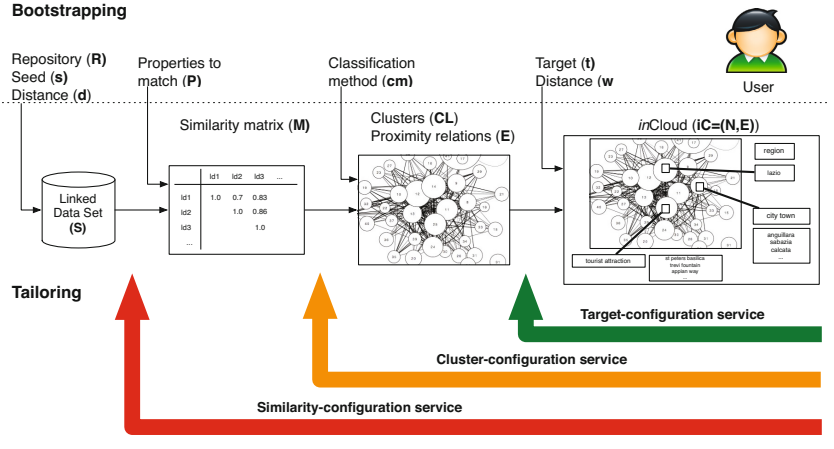

Fig. 2. The *in*Cloud life-cycle

Bootstrapping. This phase has the goal to generate the initial *in*Cloud about a given user interest. To this end, the user specifies the repository $\mathcal{R}$ and the seed s to use for extracting the set of linked data $\mathcal{S}$ that is the basis for the *in*Cloud construction. The set $\mathcal{S}$ is constituted by those linked data of $\mathcal{R}$ that are pertinent to the seed s, namely those resources that are connected to s through a property path of length $\leq d$. The distance d is a parameter to set the extension at which linked data extraction has to be enforced and thus the size of the resulting *in*Cloud[4]. The *in*Cloud creation is articulated in three steps.

Similarity Evaluation. A *similarity matrix* (M) is defined by calculating all the pairs of similar linked data resources in the set $\mathcal{S}$. An entry $M[i, j] \in (0, 1]$ denotes the similarity value between the *i-th* and the *j-th* linked data resources in $\mathcal{S}$. Techniques for linked data matching based on property names and values are employed for calculation of the similarity values of M. By default, all the properties of the linked data resources in $\mathcal{S}$ are considered for matching.

[4] More details about linked data extraction from a repository $\mathcal{R}$ are provided in [4].

Cluster Definition. A *classification method* (cm) is employed to generate a set of similarity-based clusters $CL = \{cl_1, \ldots, cl_k\}$ out of the matrix M. Different classification methods can be chosen by the user. In particular, we enforce classification methods that allow the insertion of a linked data resource in a single cluster only (i.e., *single-partitioning*), the default one, and those supporting the insertion in multiple clusters (i.e., *multiple-partitioning*).

Concept Abstraction. A set of concepts N and a set of proximity links E featuring the *in*Cloud are finally generated out of similarity clusters. In particular, a concept $n_i \in N$ is defined by extracting a set of keywords K_i and a set of types T_i from the linked data contained in a cluster $cl_i \in CL$. Prominence value p_i and degree of proximity x_{ij} are then calculated for each concept n_i. Based on the value x_{ij}, the proximity relation $e(n_i, n_j) \in E$ is defined for each pair of concepts n_i and n_j.

Tailoring. This phase has the goal to enable the user to dynamically configure the *in*Cloud for adaptation and re-organization according to her/his exploration preferences. The *similarity-configuration*, *cluster-configuration*, and *target-configuration* services are defined enforcing tailoring at three different levels of deepness.

Target-Configuration Service. This service enables the user to re-organize the *in*Cloud structure according to a specific target query of interest. Target-configuration is a "lightweight" re-organization service of the *in*Cloud that works only on prominence while keeping the similarity matrix M and related clusters CL unaltered.

Cluster-Configuration Service. This service enables the user to change the clustering method to produce new clusters/concepts providing a different view on the underlying linked data. Cluster-configuration is a "middleweight" re-organization service, in that it allows to produce a different aggregation of data while keeping the similarity matrix M unaltered.

Similarity-Configuration Service. This service enables the user to change the properties to consider for evaluating the similarity of linked data, and thus to set the "dimensions" of similarity to emphsize. Similarity-configuration is a "heavyweight" re-organization service that deeply changes the structure of the *in*Cloud since it directly works on the set of linked data $\mathcal{S}$ extracted from the repository $\mathcal{R}$.

3 Target-Configuration Service

The target-configuration service makes it possible for a user to restrict the exploration of the *in*Cloud according to a target query, which expresses in form of keywords a specific theme/topic, or a specific entity described by the linked data which compose the *in*Cloud, such as a real-world object/person, an event, a situation, or any similar subject. For example, in our *in*Cloud example of Figure 1,

a user may be interested in focusing on a specific city, like Rome, or either on the notion of "city", seen as the collection of all the cities described in the cloud. The target query is processed against the *in*Cloud concepts, and a new *in*Cloud is produced containing only the concepts matching the target, as shown in the example of Figure 3.

Given a target query t and an *in*Cloud iC, the target-configuration service produces a new *in*Cloud iC' which contains one or more cut-outs of iC composed by the subset of the concepts of iC that "have to do" with t. In particular, a cut-out contains the concepts $\overline{C}$ of iC that directly match t, and can also contain the concepts of iC at a distance w from at least one concept in $\overline{C}$. Moreover, concepts in the new *in*Cloud are characterized by a new value of prominence for iC'.

The core functions of target-configuration are the *filtering* function and the *prominence* function.

3.1 The Filtering Function

In order to satisfy an interest, a user formulates her target as a keyword-based query t, composed over a controlled vocabulary of terms extracted from the sets of keywords and types associated with the concepts of the *in*Cloud. The filtering function takes the actual *in*Cloud , the target query t and a distance parameter $w \geq 0$ setting the extension of the filtering and produces the new *in*Cloud iC':

$$filtering(iC, t, w) \rightarrow iC'$$

The query t is processed using standard tokenization techniques for extracting constituent terms. Each constituent term $\overline{t}$ is matched against sets K_i and T_i of each concept $n_i \in N$. Each n_i containing $\overline{t}$ in K_i and/or T_i (looking at the type names) is selected and added to the set of filtered concepts $N' \in iC'$. N' is then expanded by adding, for each concept $n_i \in N'$, all the adjacent concepts $n_j \in N$ for which $e(n_i, n_j) \in E$. This last step is iterated w times. According to this procedure, the resulting *in*Cloud iC' is composed by all the concepts matching the target t and all the concepts at distance w in iC from them.

3.2 The Prominence Function

The prominence p_i of a concept $n_i \in iC'$ is a measure of the relevance of n_i in iC' according to the number of proximity relations holding between n_i and the other concepts in N', as follows:

$$prominence(iC') \rightarrow [0, 1]$$

Concept prominence is computed on the basis of the random walks procedure that has been proposed in [11]. This measure is calculated by counting how often a concept n_i is traversed by a random walk between two other concepts, using proximity relations between concepts as paths in the *in*Cloud. In particular, the probability of using a given proximity relation in a random walk for moving from a concept n_i to another concept n_j is proportional to the proximity degree x_{ij} between n_i and n_j. We first label each concept with a same initial small value

of prominence and then we start walking in the *in*Cloud moving from a concept to the subsequent one using their proximity relation as a path. Each time a concept is visited, we augment its prominence. When more than one concept is reachable from a starting concept, we randomly choose the proximity relation to follow. The probability of each proximity relation to be chosen as a step in the random walk is proportional to its proximity degree. For the sake of simplicity, we simply stop the procedure after a pre-defined number of times. According to this approach, the concepts with the highest levels of prominence at the end of the process are those which are more connected with others through proximity relation with high degrees of proximity.

3.3 Example

As an example of target configuration, we suppose that the user is interested in switching her view of the *in*Cloud shown in Figure 1 on the target "rome". This configuration produces two main cut-outs in the resulting *in*Cloud, one containing geographical and touristic information (Figure 3 (a)) and the other one containing movies (Figure 3 (b)). In the figure, concepts directly matching the target (i.e., $w = 0$) are highlighted, and adjacent concepts (i.e., $w = 1$) are shown as gray circles. Concepts directly matching the target are related to the city of Rome and to movies located in Rome. Adjacent concepts contain information about resources that are indirectly referred to the city of Rome, such as for example Italian villages surrounding the capital or movies similar to the one located in Rome. The new prominence values for the resulting *in*Cloud show that the five concepts directly matching the target are the most prominent, since the new cloud is built starting from them. However, looking at the

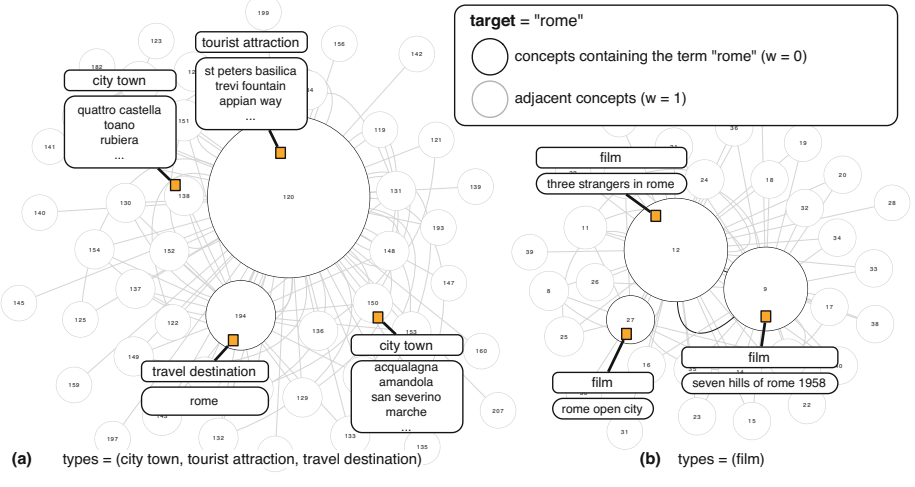

Fig. 3. An example of target-configuration of the *in*Cloud of Figure 1 with target "rome"

cut-out of Figure 3 (a), it is interesting to note how the most prominent concept is the one containing touristic attractions, since it is the one containing resources associated not only with Rome but also with the other towns surrounding the capital.

4 Cluster-Configuration Service

The cluster-configuration service is conceived to enforce a dynamic re-organization of the *in*Cloud through the use of a different classification method for linked data clustering. As an example, we consider the *in*Cloud fragment of Figure 4(a) taken from Figure 1 where clusters are produced using a hierarchical agglomerative clustering. In this example, numerous and very focused clusters characterize the *in*Cloud, thus enforcing a sort of "analytic" view of the underlying linked data. By invoking the cluster-configuration service, the user can switch from the *in*Cloud of Figure 4(a) to the *in*Cloud of Figure 4(b). The example of Figure 4(b) is built by employing clique percolation as classification method. We observe that the clusters of Figure 4(b) provide a sort of "thematic" view of the underlying linked data where a higher level of aggregation is enforced and a lower number of clusters is produced as a result.

The cluster-configuration service changes the cluster structure of the *in*Cloud and it is conceived to switch from one classification method to another according to the kind of view that the user aims to enforce (i.e., analytic vs. thematic). The cluster-configuration service works on a similarity matrix M and it generates a new set of clusters CL' with new proximity relations and associated degrees of proximity.

The core functions of cluster-configuration are the *clustering* function and the *proximity* function.

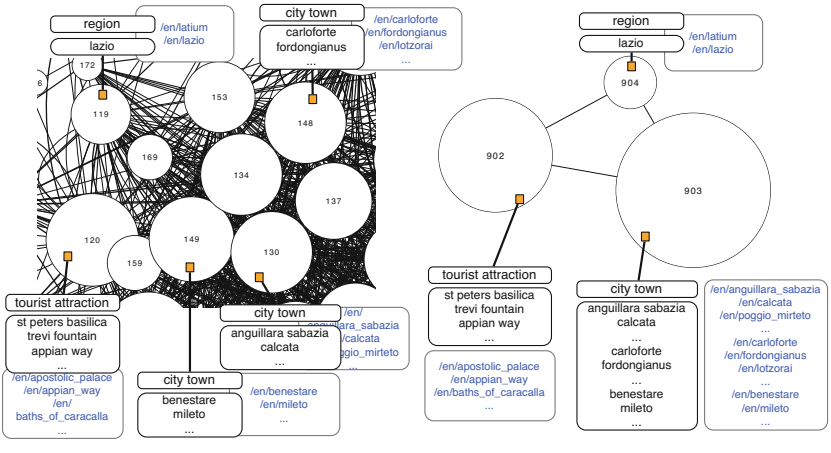

Fig. 4. An example of *in*Cloud generated with a) hierarchical clustering and b) clique percolation

4.1 The Clustering Function

The clustering function produces a set of clusters CL' calculated by relying on a similarity matrix M through the use of the classification method cm as follows:

$$clustering(M, cm) \rightarrow CL'$$

In the literature, a number of clustering algorithms are available to be employed as classification method cm. In [10], the main existing clustering algorithms are surveyed according to the partitioning solution used for segmenting the dataset to cluster. In the following, we distinguish the possible classification methods in two different families, namely *single-partitioning* and *multiple-partitioning*.

Single-Partitioning Classification Methods. They are based on the idea that a linked data resource is placed in only one cluster and thus cluster overlapping is not possible. An example of single-partitioning classification method is agglomerative hierarchical clustering [3]. Hierarchical clustering works on the similarity matrix M through a series of successive merging operations of linked data resources into groups. The algorithm follows a bottom-up approach to compose a tree where each leave corresponds to a linked data resource, while intermediate nodes represent virtual elements,(i.e., cluster "centroids") that group the child nodes of the tree. Initially, a singleton cluster cl_i is created for each linked data resource stored in the matrix M. Then, in each successive iteration, the closest pair of clusters (i.e., the clusters with the highest similarity value in M) are merged and inserted in the tree. The iteration terminates when no further merge operations are possible. By exploiting the cluster tree, clusters with a desired level of similarity can be selected by setting an appropriate similarity threshold th. In the example of Figure 4(a), hierarchical clustering with a similarity threshold $th = 0.7$ has been adopted, generating a total number of 210 clusters. More technical details about similarity-based hierarchical clustering of linked data can be found in [3].

As a general remark, we note that single-partitioning methods are usually characterized by quadratic/cubic computational complexity and they tend to produce a sort of "analytic" view of the linked data in $\mathcal{S}$ composed of small, homogeneous clusters. The readability of clusters is high due to the focused information therein contained. However, retrieving a resource within clusters becomes hard when some incorrect grouping operation has happened and a linked data resource can be misplaced into an inappropriate cluster.

Multiple-Partitioning Classification Methods. They allow the possible insertion of a linked data resource in more than one cluster. An example of multiple-partitioning classification method is the clique percolation method (CPM) [12]. The CPM is a clustering algorithm that works on a linked data graph $\mathcal{S}^+$ as input for generating the set of clusters CL as output. $\mathcal{S}^+$ corresponds to the RDF graph of the linked data in $\mathcal{S}$ "augmented" with additional edges to directly connect the pairs of similar linked data reosurces that have an entry in the similarity matrix M. The CPM recognizes as a cluster a region of nodes in $\mathcal{S}^+$ which are

more densely connected to each other than to the nodes outside the region. The CPM is based on the notion of *k-clique* which corresponds to a complete (fully-connected) sub-graph of k nodes within the graph $\mathcal{S}^+$. Two k-cliques are defined as *adjacent k-cliques* if they share $k-1$ nodes. The CPM determines clusters from k-cliques. In particular, a cluster, or more precisely, a k-clique-cluster, is defined as the union of all k-cliques that can be reached from each other through a series of adjacent k-cliques. In the example of Figure 4(b), the CPM has produced a total number of 26 clusters. More technical details about the CPM classification method can be found in [4].

As a general remark, we note that multiple-partitioning methods are usually characterized by exponential computational complexity and they tend to produce a sort of "thematic" view of the linked data in $\mathcal{S}$ composed of large, varied clusters. Each cluster provides a "bird-eye" of the contained linked data and a certain resource can belong to different clusters due to the fact that a linked data can match with more than one linked data resource.

4.2 The Proximity Function

The proximity function produces a set of edges E' calculated by relying on a set of clusters CL' and a similarity matrix M as follows:

$$proximity(CL', M) \rightarrow E'$$

The proximity function exploits the clusters in CL' and it calculates the proximity degree x_{ij} for each pair of clusters $cl_i, cl_j \in CL'$. An edge is inserted in E' when a proximity degree $x_{ij} > 0$ is found. The proximity degree x_{ij} measures the level of similarity between the contents of the clusters cl_i and cl_j. Thus, x_{ij} is proportional to the number of similar linked data resources between cl_i and cl_j according to the similarity matrix M. The proximity degree x_{ij} is calculated as follows:

$$x_{ij} = \frac{\mid s_{ij} \mid}{\mid cl_i \mid}$$

where $s_{ij} = \{M[k,z] > 0 : k \in cl_i, z \in cl_j\}$ is the set of similar linked data between cl_i and cl_j and $\mid cl_i \mid$ is the cardinality of the cluster cl_i, that is the number of linked data resources belonging to cl_i. The higher the number of similar linked data between cl_i and cl_j, the higher the corresponding proximity degree x_{ij}. We note that the proximity degree between two clusters cl_i and cl_j is asymmetric, denoting the fact that the cluster cl_i can be more specific than cl_j and somehow "contained" in it, or viceversa.

5 Similarity-Configuration Service

The similarity configuration service is invoked in order set the dimension of interest for similarity evaluation in terms of properties to be considered for matching

resources in $\mathcal{S}$. We call *dimension* $\mathcal{D}$ a subset of the set of properties $\mathcal{P}$ featuring linked data resources in $\mathcal{S}$. If, for example, a user is interested in the "geographical" dimension of data for similarity evaluation, she will choose only those properties representing geo-oriented attributes of resources, such as geo-location coordinates, regions, countries. The core functions of similarity configuration are the *dimensioning* function and the *similarity* function.

5.1 The Dimensioning Function

The dimensioning function defines the composition of the dimension $\mathcal{D}$ by selecting the constituent properties among those in set $\mathcal{P}$, as follows:

$$dimensioning(\mathcal{P}) \rightarrow \mathcal{D}$$

Dimensioning is performed by the end-user who interactively selects the properties of interest. The choice is assisted by giving the user the chance to select all the properties having one or more types of interest in $\mathcal{S}$ as domain, in order to support a faster process of property selection. In the bootstrapping phase, a general dimension $\mathcal{D} \equiv \mathcal{P}$ is applied to evaluate a comprehensive value of similarity. During the tailoring phase, dimensioning allows the modification of the similarity criterion by focusing on a specific subset of properties. Since the similarity matrix is calculated through matching techniques by taking into account only the properties in $\mathcal{D}$, it happens that the property choice affects the criterion used in order to consider two resources to be similar. For example, in case of geographical information, the notion of "similar" will be intended as "near", since similar resources will be the ones having similar values for their geographic properties.

5.2 The Similarity Function

The similarity function returns a similarity matrix M' starting from the linked data resources in $\mathcal{S}$ by applying the dimension $\mathcal{D}$, as follows:

$$similarity(\mathcal{S}, \mathcal{D}) :\rightarrow M'$$

M' is calculated through linked data matching techniques working on the set of linked data properties specified in $\mathcal{D}$. In order to implement *similarity*, we rely on the matching library of HMatch 2.0 matching system, where a wide set of matching functions are available to accommodate different matching requirements and cases. In particular, given two resources $r_i, r_j \in \mathcal{S}$, their similarity coefficient $M'[i,j]$ in M' is computed as follows:

$$M'[i,j] = \frac{\sum_{k=0}^{|\mathcal{D}|} match(p_k(r_i), p_k(r_j))}{\mid \mathcal{D} \mid}$$

where $p_k(r_i)$ denotes the value of the property $p_k \in \mathcal{D}$ for the resource r_i. The function *match* returns 0 if one of the two resources has no values for the

property at hand; otherwise, *match* returns a similarity degree between the two property values in form of a coefficient in the range [0,1]. The similarity degree between property values is calculated by exploiting different matching functions depending on the type of the property values at hand. In particular, if we are matching dates or numbers, we rely on specific matching functions which measure the distance between the values at hand, in such a way that a higher similarity degree corresponds to a smaller distance between the dates/numbers that are matched. Instead, in case of textual values, conventional, state-of-the-art string matching functions like I-Sub, Q-Gram, Edit-Distance, and Jaro-Winkler are available in HMatch 2.0.

5.3 Example

As an example of how the choice of a dimension may change the resulting similarity matrix and, as a consequence, the resulting concepts/clusters in the corresponding *in*Cloud, we take into account four resources, namely the URIs /en/italy, /en/france, /en/spain, and /en/united_kingdom representing the countries Italy, France, Spain, and UK in Freebase, respectively. A portion of the properties and corresponding values of these resources are shown in Figure 5.

At the bootstrapping, the degree of similarity between the four countries is quite low (i.e., not exceeding 0.42), since countries are matched according to the general dimension based on all their properties. In fact, in spite of the presence of some properties with similar or equal values, there are many country-specific properties and values that are different from one country to the other, such as for example the historical events and the local products (in the example we reported the case of beers produced in each country). The picture changes if we configure more specific dimensions such as the geographical dimension $\mathcal{D} = \{\text{area}, \text{containedby}, \text{time\_zones}\}$ and the political dimension $\mathcal{D}' = \{\text{org\_founded}, \text{form\_of\_government}, \text{currency\_used}, \text{official\_language}\}$. As we can see in Figure 6, the

property	/en/italy	/en/france	/en/spain	/en/united_kingdom
area →	301338.0	674843.0	504030.0	244820.0
containedby →	europe	europe	europe	europe
containedby →	southern_europe	western_europe	southern_europe	western_europe
time_zones →	CET	CET	CET	GMT
events →	social_war	july_revolution	first_carlist_war	battle_of_britain
events →	...	...	...	...
beers_from_here →	peroni	fischer_tradition	cerveza_reina	greens_discovery
beers_from_here →	...	...	...	...
form_of_government →	republic	republic	monarchy	monarchy
org_founded →	council_of_europe	council_of_europe	-	council_of_europe
currency_used →	euro	euro	euro	uk_pound
official_language →	italian_language	french_language	spanish_language	english_language
official_language →	-	-	catalan_language	-

Fig. 5. A portion of properties and corresponding values featuring the resources /en/italy, /en/france, /en/spain, and /en/united_kingdom in Freebase

Geographical dimension

	/en/italy	/en/france	/en/spain	/en/united_kingdom
/en/italy	1.0			
/en/france	0.81	1.0		
/en/spain	0.9	0.89	1.0	
/en/united_kingdom	0.74	0.67	0.66	1.0

Political dimension

	/en/italy	/en/france	/en/spain	/en/united_kingdom
/en/italy	1.0			
/en/france	0.91	1.0		
/en/spain	0.62	0.58	1.0	
/en/united_kingdom	0.57	0.6	0.7	1.0

Fig. 6. Example of similarity matrices for different dimensions

similarity between countries of the example changes if we adopt the geographical or the political dimension, respectively. From a geographic point of view, the most similar countries are Italy, Spain and France, and thus we can define two clusters of countries containing Italy, Spain and France on one side, and United Kingdom on the other side. This result is mainly caused by the fact that the three countries have a similar dimension and adopt the same time zone. According to the political dimension, we note that Italy and France are the most similar countries and this would result in two different clusters: one containing Italy and France and the other one containing Spain and United Kingdom, respectively.

6 Related Work

Work more strictly related to our approach is focused on improving retrieval, search, and exploration of data belonging to the Linked Data cloud [2,8]. In this context, some tools are recently being appearing, like for example Parallax (`http://www.freebase.com/labs/parallax/`), gFacet (`http://www.visualdataweb.org/gfacet.php`), Sig.ma (`http://sig.ma/`), and Microsoft Pivot (`http://www.microsoft.com/silverlight/pivotviewer/`). Mainly, the idea of these tools is to work on the presentation aspects of the Web of Data and to provide functionalities for smart browsing in form of visual interfaces based on graphs, mashups, and histograms. In a similar direction, structured and collaborative search engines are being emerging as a promising solution for presenting query results in a sort of structured form with the aim at focusing on understanding the user information need. Examples in this field are Wolfram Alpha (`http://www.wolframalpha.com`) and YAGO2 (`http://www.mpi-inf.mpg.de/yago-naga/yago`).

Other work related to *in*Clouds includes approaches for linked data organization and presentation. Examples of solutions in this respect are [6,9], where tools for exploration of DBpedia and Freebase are presented, not only via directed RDF links, but also via newly-discovered knowledge associations and visual navigation paths. Aggregation techniques are proposed to combine related topics in unified nodes, providing also a textual description of each node. In other

approaches, like Marbles (`http://www5.wiwiss.fu-berlin.de/marbles`) and LESS (`http://less.aksw.org`), information about resources of interest is presented exploiting HTML and RSS and by using different colors to distinguish sources. In comparison with recent approaches to graph summarization [14], we stress that *in*Clouds do not aim at providing efficient graph compression techniques for visualization of a potentially large dataset. Instead, the ultimate goal of *in*Clouds is similarity-based aggregating of linked data resources for exploration purposes. Summarization and compression are interesting side effects of our clustering methods that could be integrated in our clouding services for improving visualization in case of large linked data sets to consider.

Main Contribution of the Proposed Approach. The main contribution of our clouding services for linked data exploration regards the definition of techniques to move from the flat and static organization of linked data to a high-level and dynamic thematic view. This new organization of data makes it possible the definition of powerful services for linked data exploration as well as new paradigms for linked data presentation. We introduce an intuitive visualization of linked data in terms of concepts, which synthesize the contents of thematic clusters. Moreover, we define a dynamic environment where *in*Clouds are not seen as a static data structures, but where they can be dynamically (re-)configured during their life-cycle by the user choosing different techniques for matching, clustering and filtering of contents.

7 Concluding Remarks

In the paper, we defined clouding services for smart and effective exploration of linked data through interactive configuration of *in*Clouds. Besides aggregation-based visualization of linked data, clouding services aim at providing the essential functionalities for allowing the end-user to dynamically change the view over a certain linked data set of interest according to her/his personal preferences. In this sense, target-, cluster-, and similarity-configuration services are to be seen as a sort of *dashboard*, enabling the user to take control of *in*Clouds and thus of the underlying data through powerful and intuitive re-organization/manipulation tools. Ongoing work is devoted to the development of the proposed services in our prototype for *in*Cloud construction and management (`http://islab.dico.unimi.it/inCloud/`). Some experimental results have already been collected by focusing on evaluation of *in*Clouds with respect to matching/clustering accuracy and user-perceived quality of data cloud organization [3]. In particular, we presented an experiment run with a group of 18 students of the Databases course of the Master Degree in Computer Science held at the University of Milan. The students had a similar background on Linked Data and Semantic Web, mainly based on some classes delivered on these topics in the course. Students where required to work on three test cases corresponding to different kinds of initial seeds of interest and *in*Clouds involving different datasources, including Freebase and DBpedia. In particular, we asked each student to compare *in*Clouds with respect

to conventional web tools for accessing the linked data contents, such as the web interfaces of Freebase and Wikipedia. The main goal of the experiment was to collect a feedback concerning the effectiveness and advantages of our approach for linked data exploration. The answers were positive. For about the 75% of the users, *in*Clouds provide relevant and sufficient information about the data of interest and the perceived quality of the thematic organization is generally good. Moreover, the majority of the involved students reported that *in*Clouds provide an advantage in terms of effectiveness and usability with respect to conventional web tools for linked data exploration. A more specific evaluation of dimension-based configuration is being conducted using a method analogous to the one used for the *in*Cloud quality evaluation. However, the *in*Cloud quality evaluation results are promising also for the usefulness of dimension-based matching and re-configuration of *in*Clouds. In fact, from the experiments we got the suggestions of providing mechanisms to interactively focus the *in*Clouds organization according to different user needs (i.e., matching dimensions).

References

1. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data - The Story So Far. Int. Journal on Semantic Web and Information Systems 5(3) (2009)
2. Bozzon, A., Brambilla, M., Valle, E.D., Pasini, C.: A conceptual framework for linked data exploration. In: Proc. of the 1st ICWE Int. Workshop on Search, Exploration and Navigation of Web Data Sources (ExploreWeb 2011), Paphos, Cyprus (2011)
3. Castano, S., Ferrara, A., Montanelli, S.: Structured Data Clouding across Multiple Webs. Information Systems (2011) (to appear)
4. Castano, S., Ferrara, A., Montanelli, S.: Thematic Exploration of Linked Data. In: Proc. of the 1st VLDB Int. Workshop on Searching and Integrating New Web Data Sources (VLDS 2011), Seattle, USA (2011)
5. Davies, S., Hatfield, J., Donaher, C., Zeitz, J.: User Interface Design Considerations for Linked Data Authoring Environments. In: Proc. of the WWW Int. Workshop on Linked Data on the Web (LDOW 2010), Raleigh, NC, USA (2010)
6. Hirsch, C., et al.: Interactive Visualization Tools for Exploring the Semantic Graph of Large Knowledge Spaces. In: Proc. of the IUI Int. Workshop on Visual Interfaces to the Social and the Semantic Web, Sanibel Island, USA (2009)
7. Hogan, A., Harth, A., Passant, A., Decker, S., Polleres, A.: Weaving the Pedantic Web. In: Proc. of the WWW Int. Workshop on Linked Data on the Web (LDOW 2010), Raleigh, NC, USA (2010)
8. Marchionini, G.: Exploratory Search: from Finding to Understanding. Communications of the ACM 49(4) (2006)
9. Mirizzi, R., Ragone, A., Di Noia, T., Di Sciascio, E.: Semantic Wonder Cloud: Exploratory Search in DBpedia. In: Proc. of the ICWE 2nd Int. Workshop on Semantic Web Information Management (SWIM 2010), Vienna, Austria (2010)
10. Müller, E., Günnemann, S., Färber, I., Seidl, T.: Discovering Multiple Clustering Solutions: Grouping Objects in Different Views of the Data. In: Proc. of the 10th IEEE Int. Conference on Data Mining (ICDM 2010), Sydney, Australia (2010)
11. Newman, M.J.: A Measure of Betweenness Centrality based on Random Walks. Social Networks 27(1) (2005)

12. Palla, G., Derényi, I., Farkas, I., Vicsek, T.: Uncovering the Overlapping Community Structure of Complex Networks in Nature and Society. Nature 435 (2005)
13. Petrelli, D., Mazumdar, S., Dadzie, A.-S., Ciravegna, F.: Multi Visualization and Dynamic Query for Effective Exploration of Semantic Data. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 505–520. Springer, Heidelberg (2009)
14. Tian, Y., Hankins, R., Patel, J.: Efficient Aggregation for Graph Summarization. In: Proc. of the 2008 ACM SIGMOD Int. Conference on Management of Data, Vancouver, Canada (2008)