Chapter

Semantic Web and Interactive Knowledge Graphs as an Educational Technology

Victor Telnov and Yuri Korovin

Abstract

Technologies of knowledge representation, inductive reasoning, and semantic annotation methods are considered in relation to knowledge graphs that are focused on the domain of nuclear physics and nuclear power engineering. Interactive visual navigation and inductive reasoning in knowledge graphs are performed using special search widgets and an intelligent RDF browser. As a toolkit for ontologies refinement and enrichment, a software agent for the context-sensitive searching for new knowledge in the WWW is presented. In order to evaluate the measure of compliance of the found content with respect to a specific domain, the binary Pareto relation and Levenshtein metrics are used. The proposed semantic annotation methods allow the knowledge engineer to calculate the measure of the proximity of an arbitrary network resource in relation to classes and objects of specific knowledge graphs. Operations with remote semantic repositories are implemented on cloud platforms using SPARQL queries and RESTful services. The proposed software solutions are based on cloud computing using DBaaS and PaaS service models to ensure scalability of data warehouses and network services. Examples of using the proposed technologies and software are given.

Keywords: knowledge database, ontology, inductive reasoning, knowledge graph, semantic annotation, cloud computing, education

1. Introduction

Nowadays, the ontology description languages RDF, OWL [1], description logics [2], and knowledge graphs provide a modern theoretical basis for the creation of systems and methods of acquisition, presentation, processing, and integration of knowledge in computer systems of artificial intelligence.

There are substantial considerations in favor of the predominant use of inductive reasoning in modern knowledge graphs instead of traditional deduction. Inductive reasoning rules based on consideration of possible alternatives (precedents) allow generating and verifying cognitive hypotheses (fuzzy knowledge) that cannot be obtained directly by deductive reasoning in the graph. Inductive inference is one of the basic technologies of semantic annotation of the WWW content, when it is necessary to refine, expand, and update existing graphs with new knowledge. With the help of the inductive inference, the problems of

classification and clustering of new entities in the semantic database of nuclear knowledge are solved [3].

The aim of the work presented in the chapter is to create a working prototype first and then a semantic web portal of knowledge in the domain of nuclear physics and nuclear power engineering based on ontologies and using databases deployed on cloud platforms [3]. The task of the study was to create the following graphs of nuclear knowledge:

- World nuclear data centers
- Nuclear research centers
- Events and publications from CERN
- IAEA databases and network services
- Nuclear physics at MSU and MEPhI
- Nuclear physics journals
- Integrated nuclear knowledge graph

To ensure the effective use of the nuclear knowledge database in educational activities, additional software agents have been created for reconnaissance context-sensitive search for adequate network content and its semantic annotation based on existing knowledge graphs (for example, with the aim of authoring training materials), as well as public endpoints for easy navigation on international knowledge databases DBpedia and Wikidata.

The potential beneficiaries of information solutions and technologies that are proposed in the chapter are students, professors, experts, engineers, managers, and specialists in the domain of nuclear physics and nuclear power engineering (target audience).

2. Knowledge representation: ontology design

Ontologies are often regarded as special knowledge repositories that can be read and understood both by people and computers, alienated from the developer and reused. Ontology in the context of information technology is a formal specification with a hierarchical structure, which is designed to represent knowledge. Typically, ontology includes descriptions of classes of entities (concepts) and their properties (roles) in relation to a certain subject domain of knowledge, as well as relationships between entities and restrictions on how these relationships can be used. Ontologies, which additionally include objects (instances of entity classes) and particular statements about these objects, are also called knowledge graphs. The formal ontology model O is understood as an ordered triple of the form

$$\mathrm{O} = <X, R, F>, \text{ where}$$

X is a finite set of entity classes (concepts) for the domain represented by the ontology O; R is a finite set of properties (roles) that establish relationships between entities for some domain; and F is a finite set of interpretation functions defined on

entities and/or properties for ontology O. It can be said that interpretation functions map formal ontologies to certain domains.

As an illustration of the ontology creation process, **Figure 1** below shows a design pattern for an ontology "Nuclear Training Center" type, which is used in the project [4]. This model was created on the basis of an analysis of the educational programs of the following Russian and international training centers: National Research Nuclear University MEPhI, Physics Department of Moscow State University, IAEA. The ontology design pattern is represented in the UML notation according to the international standard [5]. The actual ontology in serialized form for the knowledge graph titled "Nuclear Physics at MSU and MEPhI" is available in Ref. [6]. Another approach to the development and refinement of the structure of ontologies is based on Terminological Decision Trees (TDT) [7].

One of the attractive features of the semantic web is that it becomes possible to extract (infer) new knowledge from the facts which already exist in the knowledge graph. For this purpose, intelligent software agents are used, which are called reasoners. The way inference is carried out algorithmically is not specified in the ontology itself or in the corresponding OWL document, since OWL is a declarative language for ontologies describing. The correct answer to any question is determined by the semantics of the description logic that sets the language standard.

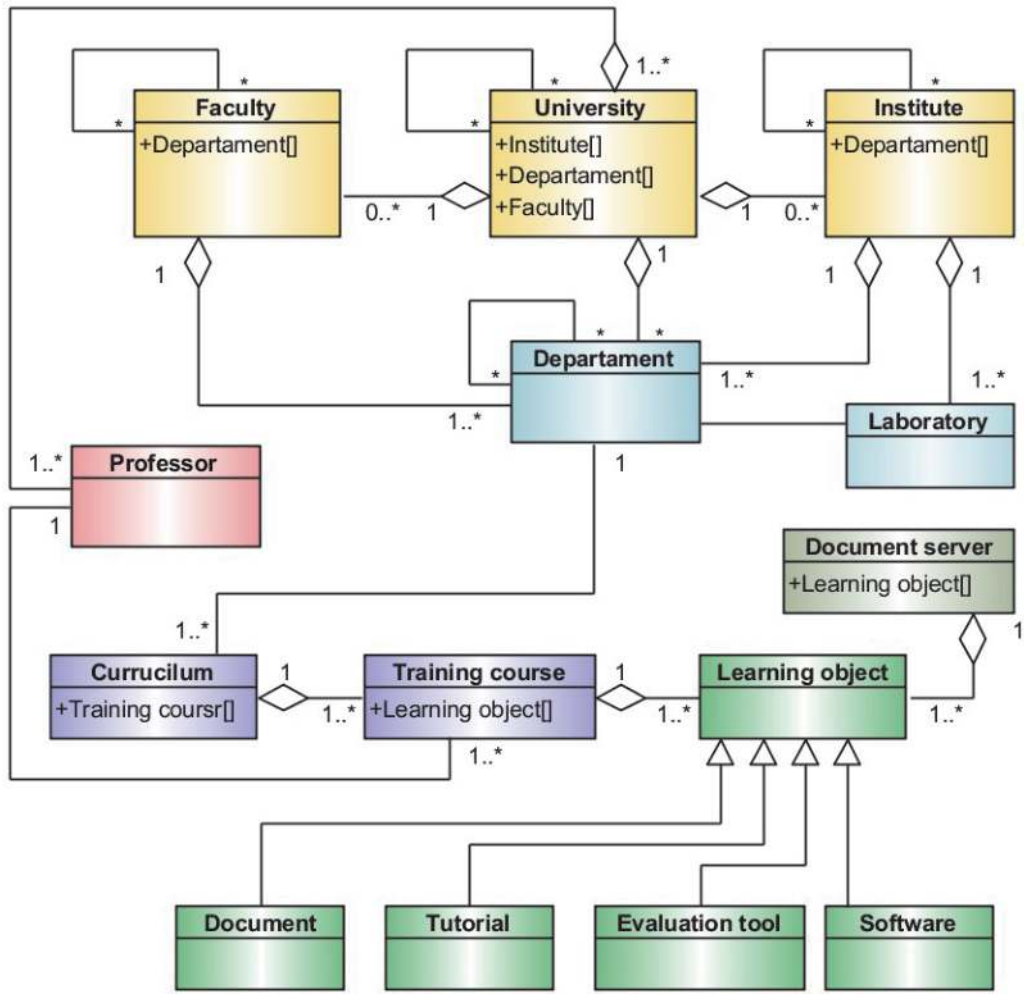

Figure 1.
Design pattern for an ontology "nuclear training center" type in UML notation.

In particular, the project under discussion is based on the description logic with the signature SROIQ (D), see [3].

The RDF browser is another significant attribute of the project [4], which distinguishes it from other well-known solutions in the field of semantic web. An example of inductive reasoning using the RDF browser is given below. Clusters of entities that are related to each other by a particular property or group of properties are examples of deduced facts (samples of new knowledge) that were not originally explicitly presented in the graph. The deduced facts in the RDF browser have the form of petals grouped around the nodes of the graph, are opened with a mouse click, and are very convenient for subsequent visual navigation in the graph.

Once on the desired location of the desired knowledge graph using the search widget, then the user through the RDF browser can perform visual navigation on the graph, visiting its nodes in the desired order and extracting metadata, hypertext links, full-text, and media content associated with the node, wherein the neighborhood (environment, closure) of each node of the graph becomes visible and available for navigation. This neighborhood includes the nodes of the graph, through which the user initially entered the semantic web, as well as adjacent nodes of other graphs that are supported by the knowledge database [3].

The visual way of specifying the inference rules on the graph makes it stand out from the more traditional reasoner's interfaces, where inference rules are specified using SWRL language, logical predicates or a SPARQL-like syntax. It seems, that the intuitive, interactive visual way of specifying inference rules is more friendly for unsophisticated users of knowledge graphs.

3. Inductive reasoning in knowledge graphs

Knowledge graphs may contain various kinds of uncertainties. For this reason, the presentation of real domains of knowledge in the context of the semantic web may encounter difficulties if only classical logical formalisms are used. Alternative approaches sometimes assume the probabilistic nature of knowledge, which is hardly always appropriate and justified [8]. In addition, purely deductive exact logical reasoning may not be possible for knowledge databases on the WWW; such reasonings do not take into account statistical patterns in the data. In this regard, of particular interest is the ability of knowledge databases as artificial intelligence systems to evaluate cognitive hypotheses, using for this purpose, in addition to a deduction, other methods of reasoning, such as inductive reasoning, argumentation, and reasoning based on precedents.

As an example of inductive reasoning in the knowledge graph, consider the following situation [3]. Some students have to pass an exam in nuclear physics at the Physics Department of Moscow State University. Let the student know only the title of the training course: "Physics of the atomic nucleus and particles" and the name of the professor: "I.M. Kapitonov." Let us formulate the task.

Task 1. Using the semantic educational web portal [4], it is required to find and study all the video lectures for this training course.

Let us also assume that the student found a video lecture in the WWW titled "Lecture 1. Physics of the atomic nucleus and particles." He suggests that this video lecture may be relevant to the training course being studied. Let us formulate a hypothesis.

Hypothesis 1. "Lecture 1. Physics of the atomic nucleus and particles" is taught by professor "I.M. Kapitonov" at the Faculty of Physics of the Moscow State University, and it is part of the training course titled "Physics of the atomic nucleus and particles."

To solve Task 1 and to verify the validity of Hypothesis 1, the following obvious reasoning should be performed step by step on the knowledge graph.

Step 1. On the educational web portal [4], from the drop-down list, select the knowledge graph "Nuclear Physics at MSU, MEPhI" (the fourth from the top in the list of knowledge graphs). Further, to solve Task 1 and to verify the validity of Hypothesis 1, one can start reasoning either with the corresponding classes "Training course," "Training video," "Professor," etc., or with specific objects "Physics of the atomic nucleus and particles," "Lecture 1. Physics of the atomic nucleus and particles," "I.M. Kapitonov," etc. Let it be decided to start the reasoning with the class "Training course." One should type the first characters of the class name in the corresponding input field of the search widget, for example "Tr," and then in the drop-down list, select the line "Training course." To begin working with the knowledge graph, click the "Start" button, as shown in **Figure 2** below.

Step 2. Depending on the current setting of "Display of knowledge graphs" in a pop-up window, in a new tab of a web browser or in the same window, the workspace of the RDF browser and the corresponding graph node named "Training course" will be opened, see **Figure 3**. Each graph node has the form of a colored circle equipped with a button for displaying a local pop-up menu and a button for displaying the metadata. In addition, around each node there are petals of various sizes, shapes, and colors, with which it is possible to start step-by-step inductive

Figure 2.
Widgets for quick diving into the knowledge graphs: in the knowledge graph "Nuclear Physics at MSU, MEPhI" select the class "training course," then click the "start" button.

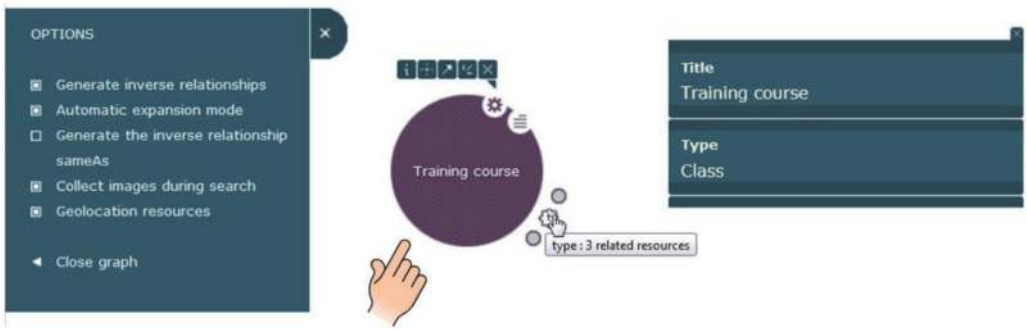

Figure 3.
RDF browser: the first node when diving into the knowledge graph "nuclear physics at MSU, MEPhI," class "training course."

reasoning and navigation in the graph. In the upper left part of the workspace of the RDF browser, there are options and help resources (legend, training videos, etc.). In the upper right part of the workspace of the RDF browser, the metadata associated with a particular node can be displayed if desired. Petals located around a node correspond to the single RDF triples in which this node is involved, or to the groups of such RDF triples (large petals). When user hovers over the petals, tooltips appear in which it is possible to see the names of the components of triplets. For large petals (triplet groups), the number of related resources is also displayed. Any group of triplets can be expanded or collapsed with a simple mouse click on the corresponding petal.

Step 3. We are interested in objects which have the type (i.e. belongs to the class) titled "Training course." There are three such objects and they are linked to our node by the "type" property, see **Figure 3**. Click to expand this resource group. Then go to the pop-up local menu of the "Training course" node and click the "View related resources" button (the second one on the right in the row of buttons). The RDF browser will display all the nodes associated with our node by any kind of properties, see **Figure 4**. Next, close the extra nodes and leave only those nodes that are associated with our node by the incoming "type" property, see **Figure 5**. In the course of practical work with the graph, it is advisable not to open the extra nodes by clicking only on the obviously necessary petals. The right side of **Figure 5** shows the metadata for the object named "Physics of the atomic nucleus and particles," which belongs to the class titled "Training Course." This object is an obvious candidate for further reasoning. However, **Figure 5** shows two other alternatives that can be left for further consideration in the inductive reasoning.

Step 4. The student is interested in the training course named "Physics of the atomic nucleus and particles," which is taught exactly at the Faculty of Physics of the Moscow State University. At this step, it is possible to narrow down the number of alternatives considered, taking an interest in the "teaches" property. **Figure 6** shows how this is done. Two alternative training courses taught at the National Research Nuclear University MEPhI, at this step, it is advisable to exclude from further considerations.

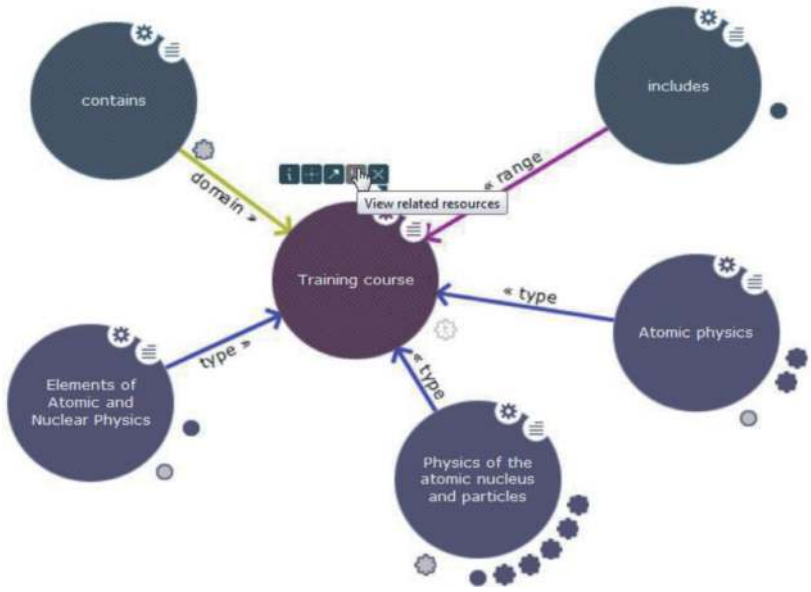

Figure 4.
RDF browser: displaying related resources for the class titled "training course."

DOI: http://dx.doi.org/10.5772/intechopen.92433

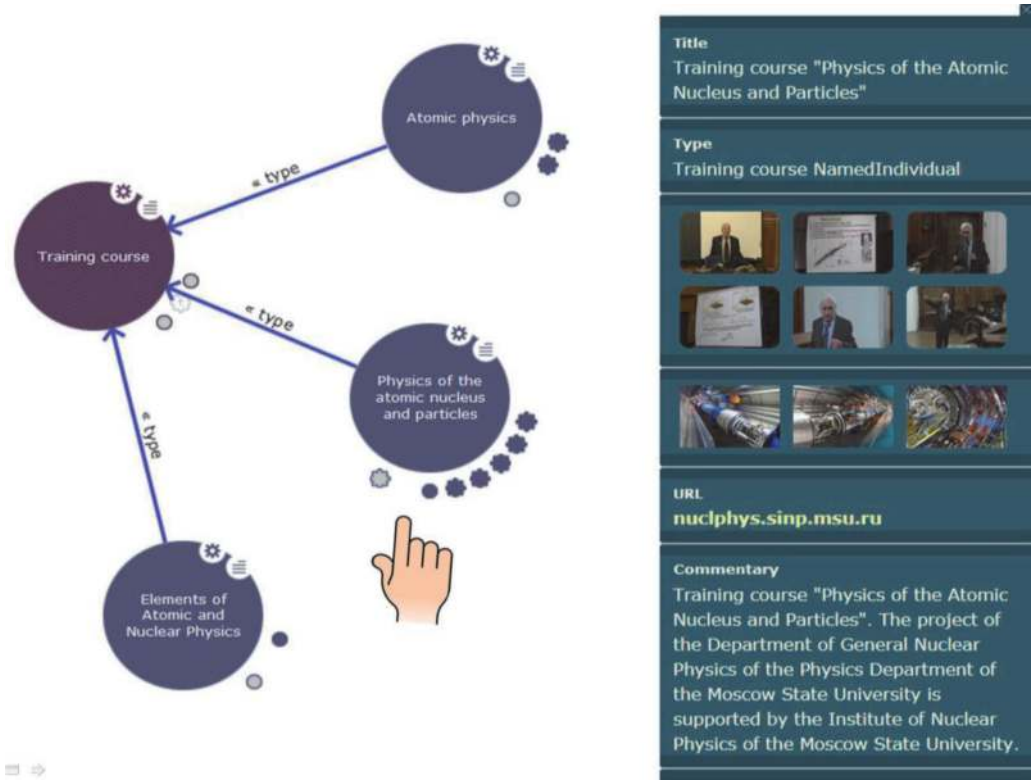

Figure 5.
RDF browser: Displaying the nodes of the graph, essential for continuing the reasoning, and the metadata for the object titled "physics of the atomic nucleus and particles."

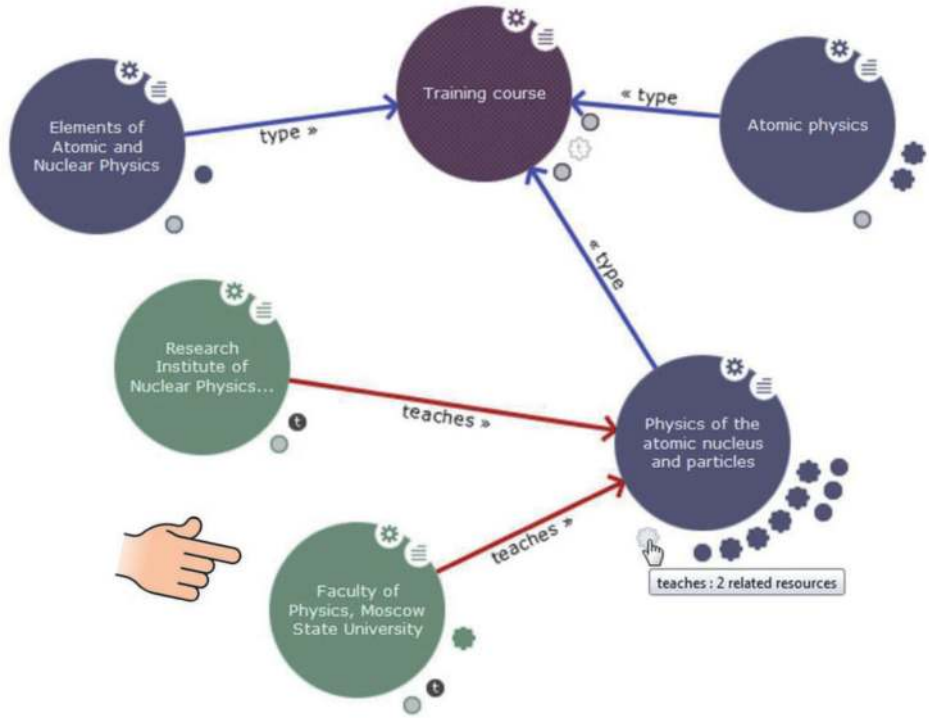

Figure 6.
RDF browser: Using the "teaches" property to reduce the number of alternatives under consideration.

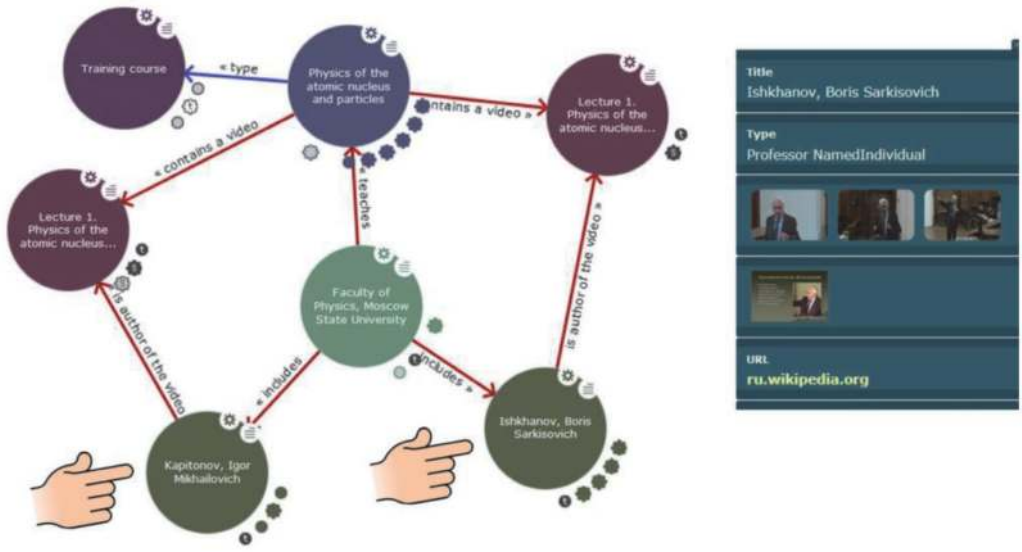

Figure 7.
RDF browser: Solving task 1 and confirming the validity of hypothesis 1.

Step 5. Continuing inductive reasoning for the object "Faculty of Physics, MSU" by the property "includes" and/or reasoning for the object "Physics of the atomic nucleus and particles" by the property "contains video," the student will be convinced of the validity of the Hypothesis 1 and will get the solution for Task 1, see **Figure 7** below.

It is possible to view detected video lectures without leaving the workspace of the RDF browser, simply by clicking on the corresponding icon in the metadata area for the object named "Lecture 1. Physics of the atomic nucleus and particles."

The result obtained in Step 5 could be achieved in the course of deductive reasoning, without considering possible alternatives. However, the use of inductive reasoning allows the user to naturally extract additional knowledge from the graph, which will not be easy to obtain with a simple deductive inference [3].

Using the above method, it is easy to discover that professor "B.S. Ishkhanov" also gives lectures on the training course "Physics of the atomic nucleus and particles" at the Faculty of Physics of the Moscow State University, see **Figure 7**. All video lectures and other learning objects of both professors for this training course are available. Through the graph of knowledge, the full content of any training course and all the existing relationships are clearly revealed.

As can be seen from the above example, the inductive reasoning process in knowledge graphs resembles a computer adventure game, does not require special skills, and is accessible to an inexperienced user. Knowledge graphs similar to those considered are used in the real educational activity at the National Research Nuclear University MEPhI. Practice shows that university students master the methods of interactive work with knowledge graphs within a few minutes.

4. Knowledge acquisition: context-sensitive search

As a toolkit that prepares data for ontology refinement and enrichment, a software agent (which is essentially a specialized meta-search engine) for the reconnaissance context-sensitive search for new knowledge in the WWW is provided. To begin with it, several characteristic features of popular search engines that are well known to most users should be noted.

- The documents found are ranked by the public search engine in accordance with its internal algorithm, which does not always meet the interests of a particular user.

- Users are not always comfortable to manage the context of the search query, refine, and direct the search.

- Links to the commercial sites usually have a higher rating than other search results. Such effect is achieved through the use of the so-called search engine optimization (SEO) to artificially raise the positions of commercial network resources on pages of popular search engines, in order to increase the flow of potential customers for the subsequent monetization of traffic.

It seems that the above circumstances and trends make public search engines an increasingly inadequate tool for extracting knowledge in the WWW for educational purposes. The context-sensitive search is based on a simple idea: to create such an intermediary (software agent) between the knowledge engineer and public search engines, which helps to systematize search results in accordance with his professional needs, by effectively filtering inappropriate content and information garbage. The goal is to involve the power of the modern search engines in the maximum level, including built-in query languages and other search controls.

When the "Context-sensitive search" software agent is working, the global document search, as well as the search for specialized web resources, is initially performed by the regular search engines (Google Ajax Search, Yandex, Yahoo, and Mail.ru), the interaction with which occurs asynchronously via the dynamic pool of the proxy servers, each of which is hosted on the Google Cloud Platform. The results of the work of the regular search engines are a kind of "raw material" for further processing. Specially designed proxy servers on the cloud platform parse these results and generate the feeds, which are then sent to the client computer, where from the feeds, snippets are formed. These snippets, which contain metadata, before they appear on the monitor of the client computer, undergo additional processing, screening, and sorting, as described below. In particular, for each snippet, its relevance, persistence, and a number of other indexes are calculated, which are further used to systematize and clustering search results obtained.

5. Search context

The query language of some search engines may include the so-called "search context." It is about the use directly in the text of the search query of special operators, which allow the user to specify the presence and relative location of specific tokens in the documents found. In this paper, a "search context" is understood slightly different way, namely, as a certain restricted on length text that characterizes the domain that is currently of interest to the knowledge engineer.

When setting the search context, the following data sources are available: taxonomies, thesauri, keywords, ontologies, textual files from the client computer, and arbitrary resources from the WWW. Any combination of the above methods for setting the search context is allowed. The resulting context is the union of the selected options. The context defined in this way allows to select, sort, and organize information that comes from the search engines through the proxy servers.

Figure 8 below shows the possible options for setting the search context.

Figure 8.
Setting the context for the reconnaissance context-sensitive search: 1 – setting the context using a file from the client computer; 2 – setting the context using an arbitrary site; 3 – widgets to show the established context.

6. Relevance, pertinence, and metrics

For the purposes of this paper, the relevance of the snippet is the measure of the similarity between the snippet and the search query text. Under the pertinence of the snippet is meant the measure of the similarity between the snippet and search context that was defined earlier. These and other measures are calculated by means a fuzzy comparison of the corresponding texts. To quantify these measures, "Context–sensitive search" software agent uses the Levenshtein metrics [9]. The algorithm for calculating the relevance of the one particular snippet is as follows.

Each lexical unit (token) from the snippet is sequentially compared with each token from the text of the search query. In the case of an exact match of tokens, the relevance of a snippet is increased by number 3. If a complete match of the lexemes requires the use of one of the Levenshtein operations (insertion, deletion, and substitution of one symbol), then the relevance of a snippet is increased by number 2 and not 3 (2 = 3–1). Here, number 1 is the price of one Levenshtein operation. If a complete match of the lexemes requires the use of two Levenshtein operations, then the relevance of a snippet is increased by number 1 (1 = 3–2). Here, number 2 is the price of the two Levenshtein operations. In the case that more than two Levenshtein operations are required to match the lexemes, the relevance of the snippet does not increase at all. It is possible to finetune the prices (weights) of Levenshtein operations of each kind, which initially (by default) are all equal to one.

The algorithm for calculating the snippet's pertinence looks similar, with the only difference that each token from the snippet is successively compared to each token from the search context. As can be seen from the above description of the algorithm, the process of calculating the relevance and pertinence of snippets is a formal one, without analyzing the possible connections of individual tokens and their environment. It is assumed that earlier such an analysis was implemented to some extent during the initial search of documents and their full-text indexing in databases of regular search engines.

Various options for sorting search results in the final output of the "Context-sensitive search" software agent are allowed. The sorting by aspect named "dominance index" deserves a special mention, which provides a joint account of the values of many metrics that characterize the adequacy of the snippets. For example,

the dominance index, in addition to the relevance and pertinence of the snippets, can also take into account the measure of the similarity between the snippet and the keywords, categories, and properties of the educational portal in total. For the practical calculation of the values of the dominance index, it seems reasonable to use the formalism of Pareto dominance relation [10], since Pareto's multicriteria ranking does not presuppose an a priori knowledge of the relative importance of aspects (for example, what is more important, relevance or pertinence?).

Let given the initial set of snippets, from which one should choose some optimal subset, the choice should be made on the basis of certain ideas about the adequacy of snippets (the principle of optimality). The selection task is a simple one, if there is only a single aspect by which it is possible to compare any two snippets and directly indicate which one is more adequate. The solution of the simple selection problems is obvious. In real situations, it is not possible to single out any one aspect. Moreover, it is often generally difficult to single out aspects. The selection and ranking of aspects that are essential for subsequent selection, in turn, is the task of choice. If some of the aspects are more important (priority) than other aspects, this circumstance should be taken into account in the mathematical model of choice.

The selection task is the algebra $<\Omega, \mathrm{O}>$ where Ω is a set of alternatives (in our case, a set of snippets) and O is the optimality principle. The task makes sense if the set of alternatives is known. Usually, the principle of optimality is unknown.

For further discussion, suppose that each snippet $x \in \Omega$ is characterized by a finite set of aspects $x = (x_1, x_2, \ldots, x_m)$. Let $\mathrm{A} = \{1, \ldots, m\}$ be the set of aspect numbers to consider when choosing; $\{\mathrm{A}\}$ is the set of all subsets A.

It can be assumed that choosing between any two snippets x and y with only one of any aspect taken into account is a simple task. If this is not the case, the corresponding aspect can be decomposed and presented as a group of simpler aspects. For each pair of snippets (x, y), we define a family of functions $\alpha_j(x, y)$ as follows:

$$\alpha_j(x,y) = \left\{ \begin{array}{ll} 1, & \textit{if } x \textit{ exceed } y \textit{ in aspect } j \\ 0, & \textit{if } y \textit{ exceed } x \textit{ in aspect } j \end{array} \right\} \textit{where } j \in \mathrm{A}; \quad x, y \in \Omega; \tag{1}$$

If x and y are equal or not comparable in some aspect with the number j, then for such number j, the function $\alpha_j(x, y)$ is not defined. Let us form a set J of numbers of such aspects that x and y differ in these aspects

$$J = \{\ j : j \in \mathrm{A};\ \ \alpha_j(x,y)\ \textit{is defined}\}, \quad J \in \{\mathrm{A}\}; \tag{2}$$

Next, we construct a metric that takes into account the number of aspects by which a particular snippet is inferior to all other snippets. Let there be two snippets $x, y \in \Omega$. Denote

$$d(y,x) = \sum_{j \in J} \alpha_j(y,x) \tag{3}$$

the number of aspects in which y is better than x. Then, the value

$$D_\Omega(x) = \max_{y \in \Omega} d(y,x) \tag{4}$$

is called the dominance index of x when presenting the Ω set. This value characterizes the number of aspects of the snippet x that are not the best in comparison with all other snippets available in the Ω set.

Let us define the function $C^D(\Omega)$ for selecting the best snippets as follows:

$$C^D(\Omega) = \left\{ x \in \Omega : D_\Omega(x) = \min_{z \in \Omega} D_\Omega(z) \right\} \quad (5)$$

Here, the value $D_\Omega = \min_{x \in \Omega} D_\Omega(x)$ is called the index of dominance of the whole Ω set. Snippets with a minimum value of the dominance index form the Pareto set. The Pareto set includes snippets that are the best with respect to all the considered aspects, including relevance and pertinence.

In the project [4], an intuitively more acceptable value is used as the index of dominance, equal to the difference between the number of aspects taken into account and the dominance index determined by the formula (Eq. (4)). Groups of snippets with the same value of the dominance index form clusters, which in the final output of the "Context–sensitive search" software agent are arranged in descending order of this index.

As an illustration of the previous computations in the next section **Figure 9** shows a variant of sorting snippets by dominance index. Snippets are sorted in descending order of the dominance index value when six metrics are taken into account, including snippets relevance and pertinence. When snippets are ordered by the value of the dominance index, within groups of elements with the same value of the dominance index (that is, within a cluster), the snippets are ordered by each of the metrics taken into account in the calculations. Other ways to organize and systematize the content found are available for any combination of metrics that characterize the adequacy of the snippets.

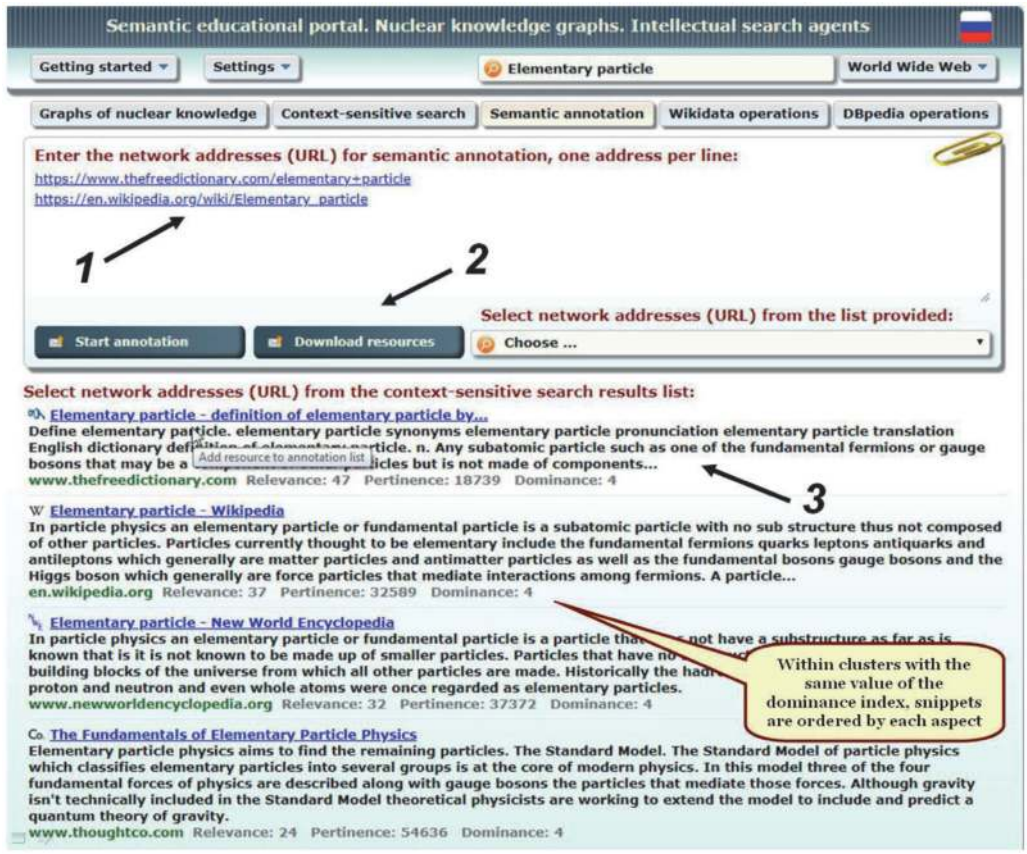

Figure 9.
Selecting network resources for semantic annotation: 1 – workspace for entering and editing network addresses (URLs) to be annotated; 2 – setting options and loading results of the context-sensitive search; 3 – the most relevant results of the context-sensitive search.

7. Knowledge graphs enrichment: semantic annotation

The database world is a place that is controlled by computers. Supercomputers have amazing computing capabilities, but they can be a struggle when it comes to acquiring new knowledge and experience or putting knowledge into practice. While it is easy for a human to decide whether two or more things are related based on cognitive associations, a computer often fails to do it. Unlike traditional lexical search where search engines look for literal matches of the query words and their variants, semantic annotation tries to interpret natural language close to how people do it. During semantic annotation, all references to cases related to entities in the ontology are recognized. Semantic annotation is the glue that ties ontologies into document spaces, via metadata.

The working panel for implementing the semantic annotation process is shown in **Figure 9** below. At the top of this panel is a workspace for entering and editing network resource addresses (URLs) to be annotated. The data in this workspace can be entered from any source, including manually. However, a more technologically advanced approach is to first find on the WWW those network resources that are most adequate to a given domain using the "Context–sensitive search" software agent. The found adequate content can then be easily loaded using the "Download resources" button and included in the list for annotation with a single mouse click.

The settings panel for the semantic annotation process is shown in **Figure 10** below. For annotation, you can select any of the knowledge graphs that are presented in the semantic repository, as well as any combination of them. To calculate measures of similarity between the annotated resource and entities from knowledge graphs, both text analysis methods and neural networks that are trained on existing knowledge graphs can be used.

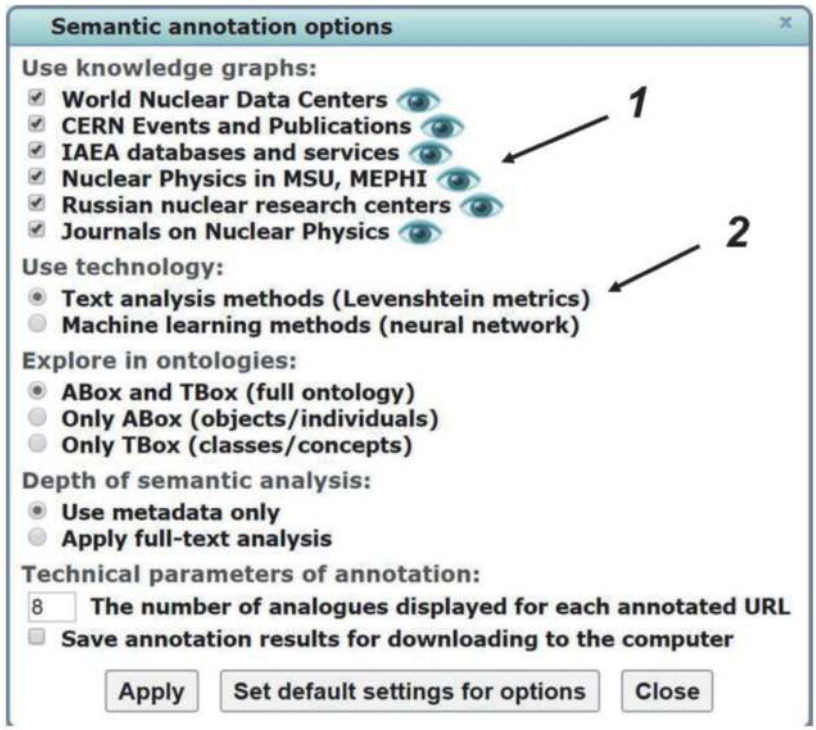

Figure 10.
Setting the options for the semantic annotation process: 1 – selecting and visualizing the knowledge graphs used; 2 – selecting of the technology and setting semantic annotation parameters.

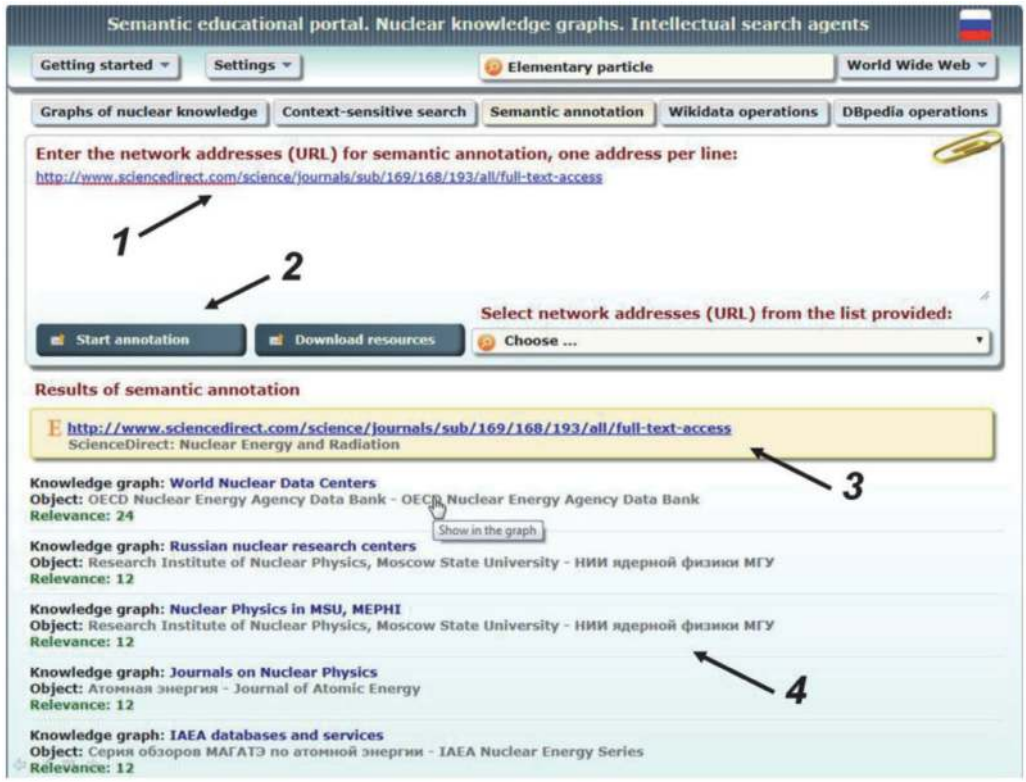

Figure 11.
Displaying semantic annotation results: 1 – addresses of the annotated network resources (URLs); 2 – setting options and starting the semantic annotation process; 3 – network resource for which semantic annotation is performed; 4 – knowledge graphs and entities corresponding to the annotated network resource.

It is possible to annotate network resources using classes (concepts) of the corresponding ontology (TBox – terminological components), using objects (individuals) of knowledge graphs (ABox – assertion components), or using both.

The depth of the carried out semantic analysis can be limited by considering only textual metadata inherent in network resources and entities in knowledge graphs. Full-text semantic analysis can be very expensive and, in many ways, redundant. Improving the accuracy of annotation in full-text analysis often does not justify the increased consumption of computing resources and time.

The number of displaying entities from knowledge graphs can be limited by the user. At the top of the output of the "Semantic annotation" software agent, the entities that are most adequate to the annotated resource appear. All the results of the work can be saved in files on the user's computer for later study.

As an example of using the "Semantic annotation" software agent, **Figure 11** below shows the results of the semantic annotation of one network resource. It can be seen that semantic annotations from five different knowledge graphs were discovered. With one click, the user can open the RDF browser and visualize the found annotations in any of the knowledge graphs, as well as anyone can see the surroundings of the entities found, for example, their classes and neighboring objects. This information is essential for a knowledge engineer who is engaged in knowledge graph refinement and enrichment.

8. Related work and conclusion

Groups of scientists from the University of Manchester, Stanford University, University of Bari and a number of other universities are focused on the issues of

theory development and technology's implementation for semantic web, description logics and incarnations of the ontologies description language OWL. A recent qualified review [11] gives a fairly complete picture of the progress made in this area and the directions for further research.

Special mention should be made on the project [12], where for the first time an attempt was made to put into practice the methods of inductive reasoning for the purpose of semantic annotation of content from the WWW. As for the issues of visualization linked data [13], here, one of the first successful projects was Lodlive [14], which provided a tool for easier surfing through the DBpedia knowledge database. It is important to continue to develop and improve tools for intuitive perception of linked data for non-professionals. VOWL [15] is one of the modern project for the user-oriented representation of ontologies; it proposes the visual language, which is based on a set of graphical primitives and an abstract color scheme. As noted in [3], LinkDaViz [16] proposes a web-based implementation of workflow that guides users through the process of creating visualizations by automatically categorizing and binding data to visualization parameters. The approach is based on a heuristic analysis of the structure of the input data and a visualization model facilitating the binding between data and visualization options. SynopsViz [17] is a tool for scalable multilevel charting and visual exploration of very large RDF & Linked Data datasets. The adopted hierarchical model provides effective information abstraction and summarization. Also, it allows to efficiently perform the statistic computations, using aggregations over the hierarchy levels.

In contrast to the above solutions, the project [4] is mainly focused on the implementation in educational activities of universities and is not limited to visualization of knowledge graphs and interactive navigation, but is aimed at the introduction of the latest semantic web technologies to the training process, taking into account the achievements in the field of uncertain reasoning. The results obtained and the software created are used in the real educational process at National Research Nuclear University MEPhI, and the project, as a whole, is focused on the practical mastering of semantic web technologies by students and professors.

Acknowledgements

The reported study was funded by the Russian Foundation for Basic Research and Government of the Kaluga Region according to the research projects 19-47-400002 and was funded by the Vladimir Potanin Foundation according to the project GC190001383.

Author details

Victor Telnov* and Yuri Korovin
National Research Nuclear University MEPhI, Obninsk, Russian Federation

*Address all correspondence to: telnov@bk.ru

IntechOpen

References

[1] W3C OWL 2 Web Ontology Language. 2012. Available from: http://www.w3.org/TR/owl2-overview/ [Accessed: 29 March 2020]

[2] Baader F, Calvanese D, McGuinness D, Nardi D, Patel-Schneider P. The Description Logic Handbook: Theory, Implementation and Applications. 2nd ed. New York: Cambridge University Press; 2010. p. 505

[3] Telnov V, Korovin Y. Semantic web and knowledge graphs as an educational technology of personnel training for nuclear power engineering. Nuclear Energy and Technology. 2019;**5**(3): 273-280. DOI: 10.3897/nucet.5.39226

[4] Telnov V. Semantic educational portal. Nuclear knowledge graphs. Intellectual search agents [Internet]. 2020. Available from: http://vt.obninsk.ru/x/ [Accessed: 29 March 2020]

[5] ISO 19505 UML Part 2 Superstructure. 2012. Available from: http://drive.google.com/file/d/0B0jk0QU2E5q9NVIwMFNieGxOZVU [Accessed: 29 March 2020]

[6] Ontology example "Nuclear Physics at MSU and MEPhI". 2020. Available from: http://drive.google.com/file/d/1AIXMsm3cfAxR6NX220R4ZeFeoSFp0mj5 [Accessed: 29 March 2020]

[7] Fanizzi N, d'Amato C, Esposito F. Induction of concepts in web ontologies through terminological decision trees. In: ECML/PKDD. Barcelona. Spain; 2010. pp. 442-457. DOI: 10.1007/978-3-642-15880-3_34

[8] Bobillo F, Carvalho R, Costa P, d'Amato C, Fanizzi N, Laskey K, et al. Uncertainty reasoning for the semantic web III. In: SWC International Workshops URSW. Revised Selected Papers; 21–25 October. Sydney, Australia; 2013. pp. 1-328. DOI: 10.1007/978-3-319-13413-0

[9] Levenshtein V. Binary codes capable of correcting deletions, insertions and reversals. Soviet Physics – Doklady. 1965;**10**(8):707-710

[10] Chen Y, Wang Z, Yang E, Li Y. Pareto-optimality solution recommendation using a multi–objective artificial wolf–pack algorithm. In: Proceedings of 10th International Conference on Software, Knowledge, Information Management & Applications (SKIMA); 15–17 December 2016. Chengdu, China; 2016. pp. 116-121. DOI: 10.1109/SKIMA.2016.7916207

[11] d'Amato C. Machine learning for the semantic web: Lessons learnt and next research directions. Semantic Web. 2020;**11**:195-203. DOI: 10.3233/sw-200388

[12] d'Amato C, Fanizzi N, Fazzinga B, Gottlob G, Lukasiewicz T. Combining Semantic Web Search with the Power of Inductive Reasoning. 2013. Available from: http://ceur-ws.org/Vol-527/paper2.pdf [Accessed: 29 March 2020]

[13] Bikakis N, Sellis T. Exploration and Visualization in the Web of Big Linked Data: A Survey of the State of the Art. 2016. Available from: http://arxiv.org/pdf/1601.08059.pdf [Accessed: 29 March 2020]

[14] Camarda D, Mazzini S, Antonuccio A. Lodlive, exploring the web of data. In: Proceedings of the 8th International Conference on Semantic Systems, I-SEMANTICS, ACM; September 2012. Graz, Austria; 2012. pp. 197-200

[15] Schlobach S, Janowicz K. Visualizing ontologies with VOWL. Semantic Web. 2016;**7**:399-419. DOI: 10.3233/SW-150200

[16] Thellmann K, Galkin M, Orlandi F, Auer S. LinkDaViz – Automatic binding of linked data to visualizations. In: Proceedings of the 15th International Semantic Web Conference; October 2015. Bethlehem, PA, USA; 2015. pp. 147-162

[17] Bikakis N, Skourla M, Papastefanatos G. Rdf:SynopsViz - a framework for hierarchical linked data visual exploration and analysis. In: Proceedings of the European Semantic Web Conference ESWC; May 2014. Heraklion, Crete, Greece; 2014. pp. 292-297