VERSITA
Applied Computer Systems
doi: 10.2478/v10312-012-0004-4
2012 / 13

Ontology Merging in the Context of Concept Maps

Vita Graudina[1], Janis Grundspenkis[2], Sigita Milasevica[3], [1-3]*Riga Technical University*

***Abstract* – This paper proposes the approach to a concept map merging using methods and tools developed for the same task in the domain of ontologies. The developed method is based on ideas that concept maps and ontologies have structural similarities, and mutual transformations between them are possible therefore tools and methods suitable for ontologies can be applied to concept maps. Concept map merging is necessary to extend the functionality of intelligent concept map-based knowledge assessment system IKAS for reuse of captured concept maps.**

***Keywords* – Ontology merging, concept maps, similarity measures**

I. Introduction

Since 2005 the concept map based intelligent knowledge assessment system IKAS has been developed [1]-[6]. It is used for students' knowledge assessment and self-assessment. Using the system students solve different concept map-based tasks which are adapted to knowledge level of the particular student [5]. During the years concept maps for several study courses have been developed and stored in IKAS. After analysis of IKAS functionality, it is found that it would be useful to extend IKAS in direction of reuse of stored concept maps outside knowledge assessment, particularly to merge them for evaluation of study courses, their modules and a whole study programme. To solve this task it is needed to find or develop a method for concept map merging, and therefore this paper is dedicated exactly to this issue. Ontologies have some structural similarities with the concept maps, despite the fact that the ontology structure is much more expressive and more complex. During the research it has been stated that similarities between ontology and the concept map do not exclude mutual transformation, thus allowing the use of already existing ontology processing methods and tools.

The aim of this paper is to demonstrate how to apply already existing ontology merging tools to concept maps in order to reuse them outside their traditional applications – knowledge assessment.

The paper is organized as follows. Firstly, the paper gives an overview of solutions existing for ontology merging, i.e., ontology similarity measures and tools for their computation. Then main principles for concept maps transformation into ontologies and vice versa are described. Further, experimental results of concept map merging, applying ontology merging tools, are shown. Finally, some conclusions are drawn and future work is outlined.

II. Ontology Merging

Ontology merging uses alignment to make one ontology from two or more ontologies from related domains [7], [8]. Usually in the obtained ontology it is not possible to identify the source ontology of the particular element. The main part of ontology merging is ontology matching [8], which is related to correspondence or relationship determination between elements of different ontologies (Fig. 1). As a result of the matching, there is alignment, which contains the set of correspondences between elements of ontologies. Depending on the algorithm used for ontology matching for some of them, besides initial ontologies, additional parameters, resources (for example, lexical database WordNet [9]), or some alignment should be used as input.

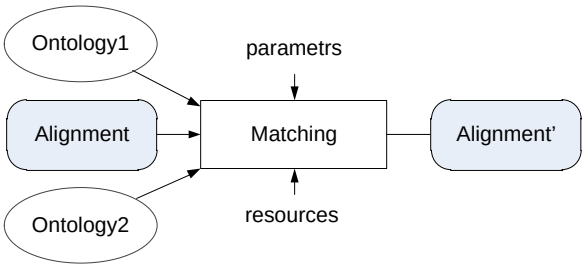

Fig .1. Ontology matching [8]

In order to achieve the alignment, a degree of similarity of elements of ontologies should be determined. Nowadays a large number of similarity measures have been developed. They can be divided based on similarity layers. Ehrig has subdivided the following layers [7]:

- Data or symbolic layer compares elements of ontologies as data values. Similarity measures used in data layer compare strings (for example, names of classes), and they are called symbolic measures while comparison of sets of elements (for example, sets of subclasses of two classes) is called object measures.
- Ontology or semantic layer compares elements of ontologies taking into account semantic relations between them.
- Context or pragmatic layer takes into account context of element usage, for example, similar elements have similar patterns of usage in the same context.

Ehrig's proposed division of similarity measures together with examples is shown in Fig. 2.

Today there are a large number of support tools for ontology mapping or merging [10]. These tools can be console-, web-based tools or tools with graphical user interface. Their functionality reaches from completely manual to fully automated ontology merging. Mostly ontology matching is done manually, although this is time- and effort-consuming work. Most of these tools are not available; some of them are research prototypes. Therefore these ontology merging tools are not suitable for a wide range of users,

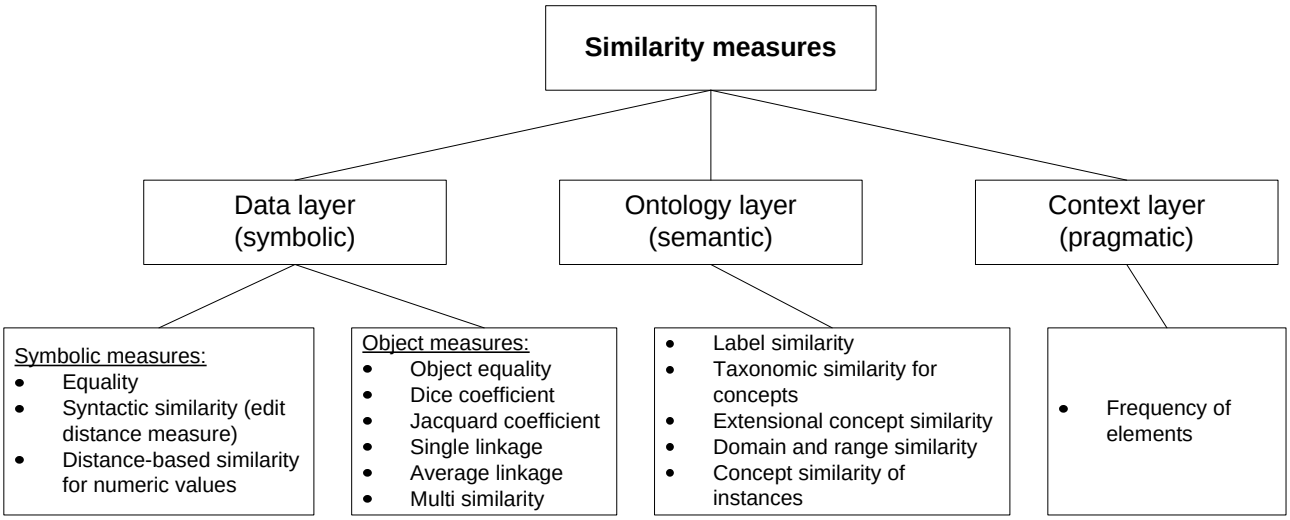

Fig. 2. Division of similarity measures, based on Ehrig [7]

because they require specific knowledge.

Euzenat & Shvaiko [8] have summed up 48 ontology matching tools. Despite this quite a large number, not all of them support exactly ontology merging. Only descriptions are available for part of tools not tools themselves. Some of them use specific data structures not only OWL, which is the most frequently used ontology description language, which has become the W3C standard. Thus, concept maps created with the IKAS system should be transformed. Some other solutions have specific requirements for an operating system. After experiments with several tools the authors of this paper have recognized plug-in PROMPT [11] for ontology editor Protégé (http://protege.stanford.edu/) as the most appropriate tool. During these experiments tools have been evaluated from the point of view of such aspects as availability of documentation and tool description, clearness of input and output format, as well as user friendly installation and application.

PROMPT is a semi-automatic ontology merging and alignment tool. It provides merging of two OWL ontologies, despite the fact that this plug-in is compatible with previous generation of Protégé tool and the new ontology has "redundant" data (related to the ontology storage format in that generation of the Protégé) which are included in tags:

```
<owl:Class rdf:ID="_DUMMY-FRAMES-METACLASS">;
<owl:Class rdf:ID="_DUMMY-FRAMES-METASLOT">;
<owl:Class rdf:ID="_TEMPORARY-ITEMS">;
<owl:ObjectProperty rdf:ID="_REFERENCES>.
```

When ontologies for merging have been chosen PROMPT makes initial suggestions for a user, i.e., a list of elements from source ontologies to merge or to copy to a new ontology. Then a user chooses action and the tool updates the list of suggestions, finds conflicts and generates new suggestions. This process is shown in Fig. 3.

During the mentioned actions in the new ontology several conflicts can arise [11]:

- name conflicts (several elements have the same name);
- reference to non-existing element;
- redundancy in a class hierarchy (there are more than one path from a class to its parent class which is not a root class);
- property restrictions, which violate class inheritance.

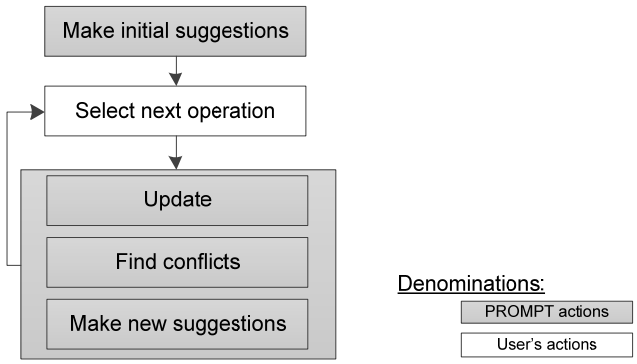

Fig. 3. Flow of the PROMPT algorithm [11]

For initial alignment PROMPT begins with the linguistic similarity measures, which are extended with ontology structure measures and user's actions. The set of ontology merging actions includes traditional ontology editing operations, as well as specific operations for ontology merging and aligning [11]:

- merge classes;
- merge properties;
- perform deep class coping, including coping of all superclasses (up to root class) and coping of all classes and properties related to the particular class;
- perform shallow class coping, coping only class without superclasses or related properties.

From methods and tools mentioned above it is obvious that in ontology engineering there are solutions for merging of two ontologies into one. If concept maps are transformed into ontologies then these solutions are applicable to concept map merging. Therefore it is needed to perform transformation from a concept map into an ontology which is described in the

Section 3. After ontology merging transformation from a resulting ontology to a concept map is needed to get back a concept map. The algorithm for this transformation is described in Section 4.

III. A Concept Map Transformation into Ontology

The algorithm for a concept map transformation into the ontology should determine the type of relation between concepts and based on the type it should be identified which ontology elements correspond to related concepts [12]. The mechanism build-in the algorithm determines the type of concepts, i.e., it has information how to determine type of relation and ontology elements from the name of link. In Fig. 4 schematically it is shown that exactly linking phrases determine the type of related concepts and only semantic (linguistic) and "part-whole" relations are directly transformable into object properties.

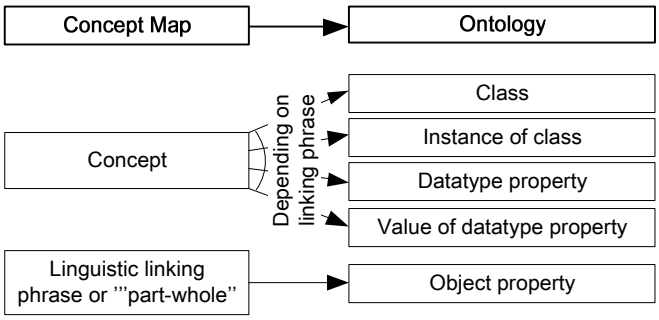

Fig. 4. Correspondence between elements of concept map and elements of ontology [12]

In a concept map it is possible to determine the following types of relations [12]:

- *the hierarchal relation*, where two classes are related with linking phrases "is a", "is a subclass of", "is a subset of";
- *the instance relation*, where a class is related to an instance with linking phrases "is an instance of", "is an example";
- *the whole-part relation* where two classes are related with the linking phrase "is a part of";
- *the hierarchal kind relation*, where two classes are related with the linking phrase "is a kind of";
- *the property relation*, where a class or an instance is related to a property with linking phrases "characterises", "has a property", "has a property (object-property)";
- *the value relation*, where a property is related to its value with the linking phrase "has a value";
- *the compliment relation* where two classes are related with the linking phrase "not";
- *the semantic or linguistic relation*, where two classes or instances are related with any other linking phrase.

The algorithm for a concept map transformation into an ontology consists of 7 steps [12] during which all elements of a concept map are handled to determine their correspondence to ontology elements and made appropriate ontology constructions. The first 6 steps analyse linking phrases included in a concept map. As a result, the type of related concepts is determined, and accordingly it is added to the ontology. Step 7 finds synonyms for concept map elements because they are not defined with links but stored differently in the structure of XML file.

Step 1: Find all concepts related with hierarchal relations.
Step 2: Find all concepts related with instance relations.
Step 3: Find all concepts related with property relations.
Step 4: Find all concepts related with value relations.
Step 5: Find all concept related with complement relations.
Step 6: Find all concepts related with part-whole or semantic relations.
Step 7: Find all synonyms defined for concepts and linking phrases. According to the determined pair of concepts in each step, write the appropriate OWL code.

IV. Ontology Transformation into a Concept Map

General correspondence between OWL ontology and a concept map is shown in Fig. 5, where it is illustrated which ontology elements directly correspond to concept map elements, i.e. all ontology classes, instances, data type properties and values of data type properties are concepts, and object properties correspond to links in concept maps [13]. Besides these elements, there are found and summarized also other OWL constructs, which describe different properties of these elements that influence element transformation into a concept map, for example, a construct defining that one class is a subclass of another class [14].

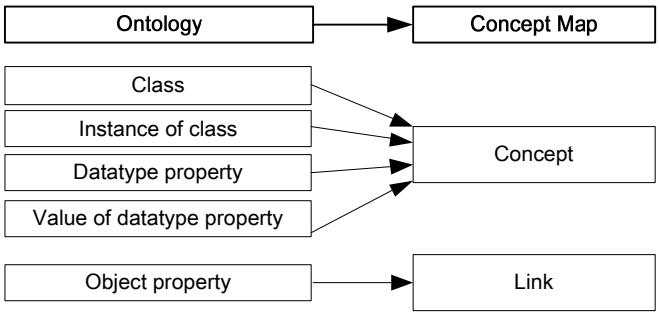

Fig. 5. Correspondence between elements of ontology and elements of concept map [13]

All cases of ontology transformations have been divided into 3 groups [13]:

- Hierarchal relations between classes and instances – include cases related to finding of ontology classes and their instances and establishing hierarchal relations, i.e. relations between a class and a subclass, between a class and its instances, Boolean relations between classes, synonyms of classes and instances, distinctions of classes and instances.
- Semantic relations between classes and instances – include cases related to finding of object properties which define semantic relations between classes/instances.
- Property relations for classes and instances – include cases related to finding of data type properties for classes/instances and values of them.

Examples of the identified mappings between OWL ontology and concept map are overviewed in [14]. The listings

of OWL code are followed by graphical representation of corresponding concept map elements.

During operation of the algorithm from the ontology saved in the text file an extended incidence matrix is obtained, where names of concepts and their interrelations are stored, showing the name of the link (linking phrase) and its direction. In addition, this matrix is extended with one more column where data about a type of concept or a label of concept type (a root class – class without superclass, subclass, instance, property or value) are stored. Basic steps for concept map generation from the ontology are the following [13]:

Step 1: Read an ontology file and check OWL syntax.

Step 2: Find all classes (begin creation of an incidence matrix).

Step 3: Find subclasses of each class (for particular class add the link "is a", which goes from a subclass to a superclass in the matrix, add labels to root classes).

Step 4: For each class check intersection, union and collection with other classes (add the link "is a" in the matrix between appropriate classes).

Step 5: For each class check complement relations to other classes (add the link "is not" in the matrix).

Step 6: Find instances of each class (add instances and links "is instance of" between appropriate classes and instances which go from an instance to a class in the matrix, add labels to instances).

Step 7: Find data type properties for each class and instance (add properties and links "has property" between appropriate class/instance and a property in the matrix, add labels to properties).

Step 8: Find values for each data type property (add values of properties and links between a data type property and its value "has value" in the matrix, add labels to values).

Step 9: For each class, instance and data type property check equivalence (add the link "is synonym of" in the matrix between appropriate elements).

Step 10: Find object properties for each class/instance (add appropriate links between classes or instances in the matrix).

Step 11: Check if an object property is inverse, symmetric or transitive (extend the matrix with appropriate links).

Step 12: Find hidden relations (relations, which can be inferred using reasoners and are not directly defined in the ontology).

Step 13: Perform corrections of concept and link names (replace understrike sign "_" with space). This is needed because spaces between words in the names of ontology elements are not allowed and usually they are replaced with "_".

Step 14: Display completed incidence matrix as a graph and save in XML accordingly to the format used in the IKAS.

V. Experiments with Concept Map Merging

With several concept maps created with IKAS, merging experiments have been performed. During these experiments the algorithm for concept map transformation into ontology has been applied, then obtained ontologies have merged into one using PROMPT, and finally one concept map has been acquired.

In this Section concept maps used for merging, problems emerged during transformation to ontology, results and problems of merging have been described.

A. Concept Maps Used for Merging

In order to perform experiments, 9 concept maps created with IKAS have been chosen. All concept maps belong to the same problem domain, i.e. they all are prepared for different study courses of the module "Artificial Intelligence" taught at Riga Technical University:

- Five concept maps for the course "Fundamentals of Artificial Intelligence", taught to the 3rd year bachelors of the academic programmes "Computer Systems" and "Intelligent Robotic Systems". These concept maps are for topics "State Space" (62 concepts), "Search Algorithms" (30 concepts), "Two Person Games" (46 concepts), "Knowledge Representation" (82 concepts) and "Logics" (56 concepts).
- One concept map for the course "Introduction to Artificial Intelligence", taught to the 2nd year bachelors of the academic programmes "Computer Systems" and "Intelligent Robotic Systems" as an elective course, and to the 3rd year bachelors of the professional programme "Computer Systems". The map contains 61 concepts.
- Three concept maps for the course "Artificial Intelligence" taught to the 1st year Master students of the academic study programmes "Computer systems" and "Intelligent Robotic Systems". The concept maps are for topics "Introduction to Agents" (47 concepts), "Logical Agents" (36 concepts) and "State Space Search Agents" (29 concepts).

B. Some Problems with Concept Map Transformation into the Ontology

The fact is that concept maps do not have formal rules for their constructions, and the user is completely free in relating concepts included in the concept map. It has led to a situation that the concept map transformation into the ontology is not fully automated. Creators of concept maps have used several constructions which are not allowed in the OWL DL language to keep the ontology computable.

During transformation the following cases are solved:

- Symbols forbidden in names of OWL ontology elements have been used as names of concepts. For example, spaces between words in element names are replaced with _ as it is traditionally performed in the ontology editor Protégé. Some other symbols as brackets, commas, asterisks are omitted or the names are reformulated.
- A number as the first character is used for a concept name. It is solved with replacing the number with letters describing it. For example, a concept name "4-tuple" is replaced with "Four-tuple".
- Object properties with identical names and different domains and ranges are not allowed in ontology; therefore indices for the names of object properties are added. Thus, for instance, object properties with names

"contains_1", "contains_2" exist in the ontology. An example of use of indices for linking phrase names is shown in Fig. 6. For the linkage between concepts "Variable CS" and "Current node", "List DE" and "Nodes whose successors do not contain a goal", "List NSL" and "Nodes waiting for opening", the indices are added because domains and ranges for each linking phrase are different. For a linking phrase "uses", indices are not added because a domain is the same, and only a range differs.

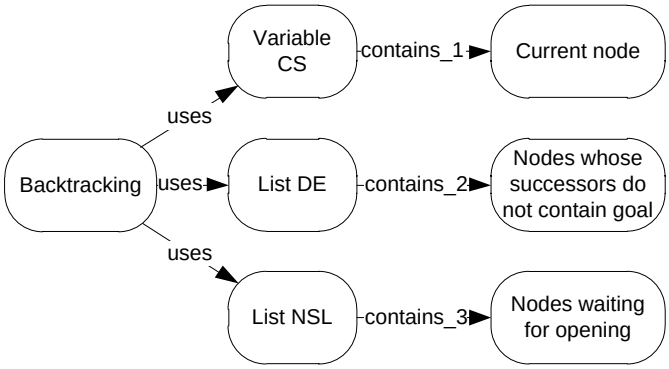

Fig. 6. An example of links with indices and without them

Some linkages between concepts are not allowed between corresponding elements in the ontology. Situations occurred are listed below:

- Two concepts, which are properties, are interconnected. It is not allowed in OWL DL; therefore, this link is deleted from the concept map. See Fig. 7 where "Average branching coefficient" and "Node branching factor" are related.

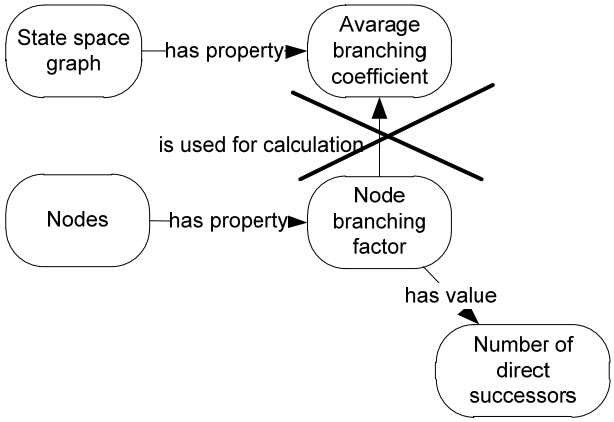

Fig. 7. An example of a link between properties

- An instance of one class is a subclass of another class. In this case instance relation is replaced with subclass relation. For example, see Fig. 8, where the concept "One step inference agent" is the instance of "Logical reasoning agent" and "Simple reflex agent" and the subclass of "Knowledge based agent".

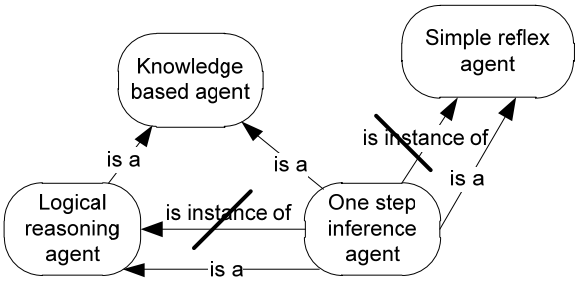

Fig. 8. An example of an instance as subclass

- A class has property and this property is further related with linguistic link to another class which is not its domain. To solve this case, a property link is replaced with a linguistic linking phrase, which is allowed in this case (see Fig. 9). This also changes the type of concept and it becomes a class. After the change of concept type and relation, the link is between two classes and a property is not related to some other class, whose property it does not belong to. The concept "Informed search algorithms" has property "Information about a state quality", which is further related to the concept "Heuristics".

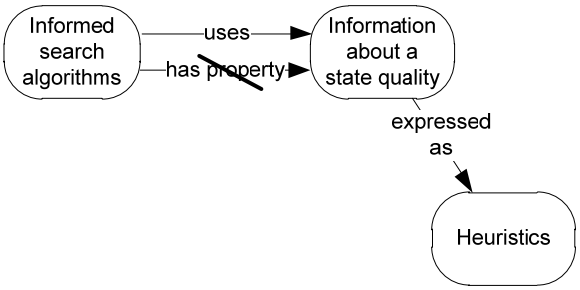

Fig. 9. An example of a property related to two classes

- A class is a part of some instances. To solve this problem instances are changed to subclasses and after that links between parts and subclasses are allowed. For example, see Fig. 10 where concepts "Propositional calculus" and "Predicate calculus" are instances of "Logical knowledge representation schemas" and at the same time the concept "Semantics" is their part.

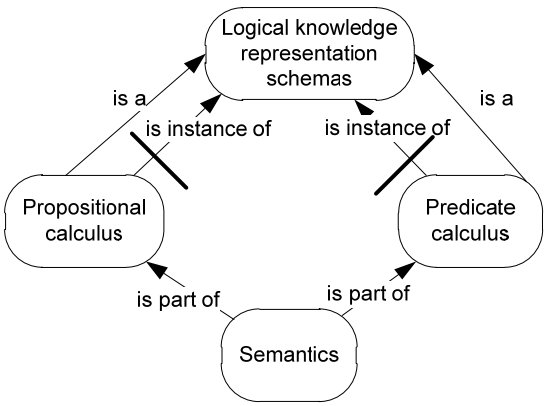

Fig. 10. An example of instance parts

To deal with problems mentioned above, several solutions exist. Firstly, some automation may be added to IKAS to check use of symbols in the concept names according to OWL principles. This solution can seriously affect use of concept maps for courses with a lot of formulas. Secondly, some automation may be added to solve problems with impropriate relations with particular types of concepts. Unfortunately, this solution influences all subjects and is in conflict with the principle of concept mapping that each person represents knowledge in different ways [16].

C. Merging with PROMPT

The concept map merging has been performed in two phases. Firstly, all five concept maps for the course "Fundamentals of Artificial Intelligence" have been merged into one concept map, also three concept maps for the course "Artificial Intelligence" have been merged into one. This has been done to obtain a single concept map for each study course. Secondly, the merge of the concept map for "Introduction to Artificial Intelligence" and the whole concept map of "Fundamentals of Artificial Intelligence" has been done, as well as the merge of the concept maps for "Fundamentals of Artificial Intelligence" and "Artificial Intelligence". These merges have been done to see the continuity and integrity of these particular subjects. In Fig. 11 the screenshot of the suggestions for merging within PROMPT is shown. The result of merge of 2 concept maps "Fundamentals of Artificial Intelligence" (MIP12) is merged with the third concept map for the same course (MIP3). The merge of 6 classes is recommended for initial suggestions. All other classes are recommended to copy into the resulting concept map.

D. Problems with Merging

As PROMPT is mainly based on syntactic similarities, some problems occur. For example, PROMPT suggests to merge the concept "Informed state search algorithms" with the "Uninformed state search algorithms", where the syntactical similarity is very close but semantically these concepts are completely different. Therefore, a user should be very careful in reading suggestions generated by PROMPT. The second problem is also related to the syntactic similarity and, in particularly, to names of links. As it is described in Section IV there could be several linking phrases with the same name in the concept maps, which is not allowed in ontologies. It has led to the fact that in ontologies there are links which differ only by indices. It means that they are syntactically quite similar and PROMT suggests merging those links. Therefore, a user should be even more careful than in case of classes.

VI. The Expected Use of Merged Concept Maps

As stated above, the proposed approach of merging concept maps has the objective to extend the IKAS functionality. We intend to use merged concept maps for several purposes

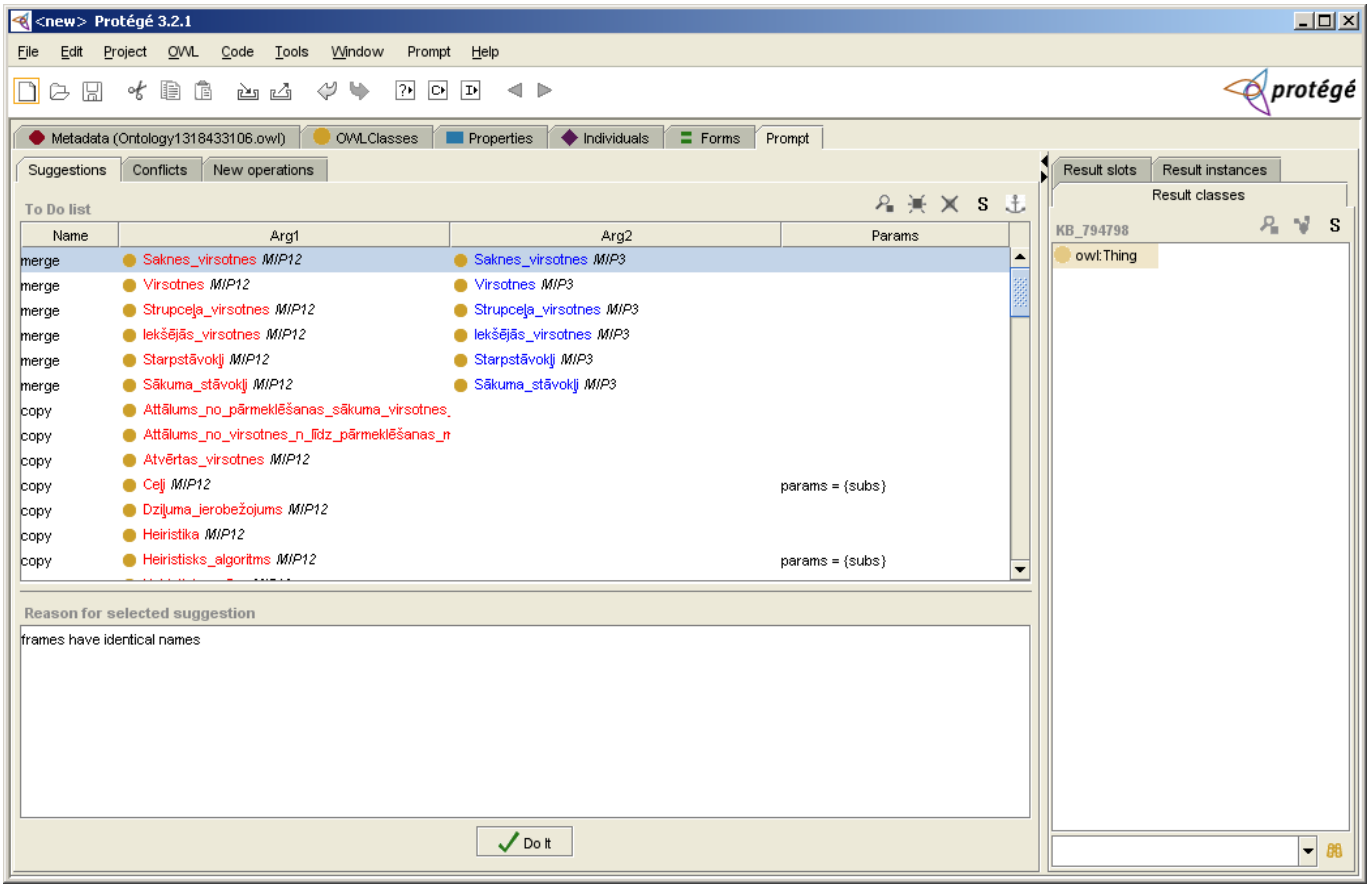

Fig. 11. Initial suggestions for ontology merge generated with PROMPT

depending on what the particular merged concept map represents, i.e. it may represent one study course, two or more related courses or a study programme as a whole.

The typical situation when the necessity of merging concept maps of one study course emerges is in case if several teachers are responsible for the same course. Suppose that each of them has a different view which concepts must be taught (the situation when these views are absolutely contradictory is not real). As a consequence, each teacher creates his/her own concept map, which to a certain extent differs from those constructed by other teachers. When merging is carried out, it is easy to find a set of concepts and linking phrases which are present in all initial concept maps, i.e. a set of those concepts and linking phrases on which all teachers agree. Besides, a merged concept map may be used as a basis for discussions about other concepts with the aim to align versions.

Considering two or more related study courses, there are two cases. First, one course is a prerequisite of another course. Usually these courses are taught by different teachers. Their merged concept maps may reveal that despite the description of courses where needed prerequisite knowledge and learning outcomes are clearly defined in fact there are not any common concept at all or at best there are only a few of them, and it is not enough to acquire the declared knowledge. Of course, there may also be merged concept maps that manifest that one course really contains all required prerequisite concepts for another course. Second, several related courses may compose a module, for example, as in case, which is analysed in this paper (there is a sequence of three courses: "Introduction into Artificial Intelligence", "Fundamentals of Artificial Intelligence" and "Artificial Intelligence", which are taught at different study years for Bachelor and Master studies). As a rule, one teacher is responsible for teaching all courses of this module. Since courses are related to a certain number of concepts and their linking phrases are repeatedly taught, a merged concept map shows these concepts, for example, in our case there are 16 concepts which are common in the courses "Introduction into Artificial Intelligence" and "Fundamentals of Artificial Intelligence", and 13 for courses "Fundamentals of Artificial Intelligence" and "Artificial Intelligence". This information is useful for a teacher who can consider at which level students must learn them in each course (to know, to be able to explain, to have skills of real life problem solving, etc.).

A merged concept map for a whole study programme allows checking whether the corresponding graph is connected or not. If it is connected then it indicates that study courses have logical sequence and the requirements for prerequisites are satisfied. Of course, if a graph is disconnected it means that a programme consists of isolated "knowledge islands" (groups of related courses), which lack relationships. It is worth pointing out that this fact cannot be used for decision making about the quality of a study programme because the latter in our case includes general engineering courses, computer science basic courses, courses from concentration areas, courses in economics, social sciences, etc. It is obvious that in case of merging concept maps of pairs of some abovementioned groups of courses, all concepts will be different. On the contrary, if a merged concept map of a whole programme shows that, for instance, computer science basic courses have not common concepts, it is evident that this part of programme does not satisfy the quality requirements.

And last, but not least, a merged concept map offers rather wide possibilities for knowledge remediation because if the IKAS discovers that some concept or a linking phrase is not mastered it can try to find the cause not only within "the boarders" of a course under consideration but can look for needed learning objects into a concept map of prerequisite course.

One way is to use broader learning objects, which include concepts and their relations according to the merged (full) concept map. The contents of learning objects are composed according to principles described in [17]. Depending on the subgraph type, several concepts could be added from the prerequisite concept map. An example of learning object extension, based on merged concept maps, is shown in Fig. 12. A student has made an error in the link "a" in the concept map A and therefore receives learning object LO1. Then using the merged concept map, which contains concepts from concept maps A and B, learning object LO1 is extended with material about the concept "d" from the concept map B and a student receives learning object LO2.

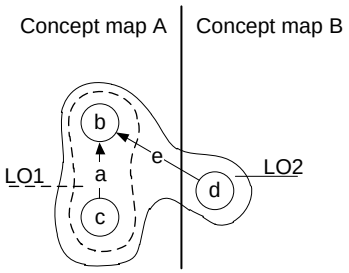

Fig. 12. An example of the use of merged concept map in IKAS

Another way to extend IKAS functionality with merged concept maps is to use them for additional testing to find more precisely which non-mastered concepts from the prerequisite concept maps are the reasons for errors made in the particular one.

VII. Conclusions and Future Work

The process of concept map merge using an ontology processing tool has been described in this paper. Nine concept maps from three different study courses of the module "Artificial Intelligence" have been merged using the ontology editor Protégé with the plug-in PROMPT.

Before merging concept maps they have been transformed into ontologies using the algorithm previously developed. During transformation several situations occurred where due to complete freedom in concept map building involvement an expert is needed. It happens because concept maps allow constructions, which are forbidden in ontologies, to keep them machine understandable and computable. The algorithms are applicable also to concept maps built with other tools, not only

IKAS. Only some changes for detecting concept/relation types more precisely may be added in the algorithm for concept map transformation into ontology, due to the fact that different linking phrases could be used.

Ontology merge is based on linguistic similarities in PROMPT; therefore, a user should be careful accepting suggestions for merge generated by PROMPT. Specially, attention should be paid to object property merge.

Future work is related to development of methods for interpretation numerical data obtained during merge, because now it is still unclear if it is good or not if, for example, 10% of classes could be merged. Also more representative experiments with a larger set of concept maps are needed to evaluate obtained results and to perform more general conclusions. Moreover, IKAS will be extended with additional features for student's knowledge remediation based on merged concept map analysis accompanied with studies related to pedagogy.

ACKNOWLEDGEMENT

The research is partly supported by internal research project of Riga Technical University FLPP-2011/8 "Development of Ontology-Based Methods and Algorithms for Comparison and Merging of Concept Maps of Different Related Study Courses" and project "New Ontology and Model Transformation Driven Information Technologies and Their Applications" supported by National Research program of Latvia in Materialsciences "Development of Novel Multifunctional Materials, Signal Processing and Information Technologies for Competitive Knowledge-Based Products".

REFERENCES

[1] Anohina A., Grundspenkis J. Prototype of Multiagent Knowledge Assessment System for Support of Process Oriented Learning. In Proceedings of the 7th International Baltic Conference on Databases and Information Systems (Baltic DB&IS 2006), July 3-6, 2006, Vilnius, Lithuania, pp. 211-219.

[2] Anohina A., Grundspenkis J. Learner's Support in the Concept Map Based Knowledge Assessment System. In Proceedings of the 7th European Conference on e-Learning, November 6-7, 2008, Agia Napa, Cyprus, Vol.1, pp. 38-45.

[3] Anohina A., Stale G., Pozdnakovs D. Intelligent System for Student Knowledge Assessment. In Scientific Proceedings of Riga Technical University, 5th Series, Computer Science. Applied Computer Systems, Vol. 26, 2006, pp.132-143.

[4] Anohina A., Vilkelis M., Lukasenko R. Incremental Improvement of the Evaluation Algorithm in the Concept Map Based Knowledge Assessment System. In International Journal of Computers, Communication and Control, 4(1), 2009, pp. 6-16.

[5] Grundspenkis J., Anohina A. Evolution of the Concept Map Based Adaptive Knowledge Assessment System: Implementation and Evaluation Results. In Scientific Proceedings of Riga Technical University, 5th Series, Computer Science. Applied Computer Systems, Vol. 38, 2009, pp. 13-24.

[6] Vilkelis M., Anohina A., Lukashenko R. Architecture and Working Principles of the Concept Map Based Knowledge Assessment System. In Proceedings of the 3rd International Conference on Virtual Learning (ICVL 2008), October 31 – November 2, 2008, Constanta, Romania, pp. 81-90.

[7] Ehrig M. Ontology Alignment: Bringing the Semantic Gap. Berlin-Heidelberg: Springer-Verlag, 2007, 247 p.

[8] Euzenat J., Shvaiko P. Ontology Matching. Berlin-Heidelberg: Springer-Verlag, 2007, 333 p.

[9] Fellbaum C. (Ed.). WordNet: An Electronic Lexical Database. Cambridge, MA: MIT Press, 1998, 422 p.

[10] Moser T., Schimper K., Mordinyi R., Anjomshoaa A. SAMOA - A Semi-automated Ontology Alignment Method for Systems Integration in Safety-critical Environments. International Conference on Complex, Intelligent and Software Intensive Systems, Fukuoka, Japan, 16-19 March 2009, pp. 724-729.

[11] Fridman Noy N., Musen M.A. PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment, In Proceedings of the 17th National Conference on Artificial Intelligence and 12th Conference on Innovative Applications of Artificial Intelligence, July 30 - August 3, 2000, Austin, USA, pp. 450-455.

[12] Graudina V., Grundspenkis J. Algorithm of Concept Map Transformation to Ontology for Usage in Intelligent Knowledge Assessment System. International Conference on Computer Systems and Technologies (CompSysTech'11), June, 16-17, 2011, Vienna, Austria, pp. 109-114

[13] Graudina V., Grundspenkis J. Concept Map Generation from OWL Ontologies. The 3rd International Conference on Concept Mapping, September, 22-25, 2008, Tallinn, Estonia and Helsinki, Finland, 2008, pp. 173-180.

[14] Graudina V. OWL Ontology Transformation into Concept Map. In Scientific Proceedings of Riga Technical University, 5th Series, Computer Science. Applied Computer Systems, Vol. 34, 2008, pp.80-91.

[15] OWL Web Ontology Language Guide. Available at: http://www.w3.org/TR/owl-guide/ (accessed 27.09.2011)

[16] Novak J.D., Cañas A.J. The Theory Underlying Concept Maps and How to Construct Them, Technical Report IHMC CmapTools. Revised 01-2008, Florida Institute for Human and Machine Cognition, 2008, 36 p. Available at http://cmap.ihmc.us/Publications/ResearchPapers/TheoryUnderlyingConceptMapsHQ.pdf (accessed 10.09.2011).

[17] Graudina V. Algorithms for Knowledge Remediation in Concept Map Based Assessment System. 2nd International Workshop on Intelligent Educational Systems and Technology-enhanced Learning (INTEL-EDU 2011), October 6, 2011, Riga Latvia. Local Proceedings of the 10th International Conference BIR, Associated Workshops and Doctoral Consortium, pp. 281-288.

Vita Graudina is a Researcher at Riga Technical University. She received Mg.sc.ing. with distinction in computer systems in 2005 and Dr.sc.ing. in 2011, both from Riga Technical University. The topic of the doctoral thesis was related to integration of ontologies in the intelligent concept map based tutoring system IKAS. Her research interests are ontologies and their applications in different fields, especially, in computer-based tutoring systems.
E-mail: vita.graudina@rtu.lv

Janis Grundspenkis is a Professor at Riga Technical University. He is the Dean of the Faculty of Computer Science and Information Technology, the Director of the Institute of Applied Computer Systems, and the Head of the Department of System Theory and Design. He obtained Dr.sc.ing. in 1972 and Dr.habil.sc.ing. in 1993, both from Riga Technical University. His research interests are artificial intelligence, particularly agent technologies and computer-based tutoring systems.

He is a member of the Institute of Electrical and Electronics Engineers (IEEE) and Association for Computing Machinery (ACM). He is a full member of the Latvian Academy of Science.
E-mail: janis.grundspenkis@rtu.lv

Sigita Milasevica received Mg.sc.ing. in computer systems from Riga Technical University in 2011 and recently has started working in the industry. Her research interests are concept maps, their evaluation and use in knowledge assessment systems.