Web Semantization – design and principles

Jan Dědek, Alan Eckhardt, and Peter Vojtáš

Abstract Web Semantization is a concept we introduce in this paper. We understand Web Semantization as an automated process of increasing degree of semantic content on the web. Part the of content of the web is further usable, semantic content (usually annotated) is more suitable for machine processing. The idea is supported by models, methods, prototypes and experiments with a web repository, automated annotation tools producing third party semantic annotations, semantic repository serving as a sample of semantized web and a proposal of an intelligent software agent. We are working on a proof of concept that even today it is possible to develop a semantic search engine designed for software agents.

1 Introduction

In their Scientific American 2001 article [3], Tim Berners-Lee, James Hendler and Ora Lassila unveiled a nascent vision of the semantic web: a highly interconnected network of data that could be easily accessed and understood by a desktop or handheld machine. They painted a future of intelligent software agents that would "answer to a particular question without our having to search for information or pore through results" (quoted from [15]). Lee Feigenbaum, Ivan Herman, Tonya Hongsermeier, Eric Neumann and Susie Stephens in their Scientific American 2007 article [15] conclude that "Grand visions rarely progress exactly as planned, but the Semantic Web is indeed

Jan Dedek · Alan Eckhardt · Peter Vojtas
Department of software engineering
Charles University, Prague, Czech Republic,
e-mail: {dedek,eckhardt,vojtas}@ksi.mff.cuni.cz

Institute of Computer Science
Academy of Sciences of the Czech Republic

emerging and is making online information more useful as ever". L. Feigenbaum et al. support their claim with success of semantic web technology in drug discovery and health care (and several further applications). These are mainly corporate applications with data annotated by humans. Ben Adida when bridging clickable and Semantic Web with RDFa ([1]) assumes also human (assisted) activity by annotations of newly created web resources.

But what to do with the content of the web of today or of pages published without annotations? The content of the web of today is too valuable to be lost for emerging semantic web applications. We are looking for a solution how to make it accessible in semantic manner.

In this paper we would like to address the problem of semantization (enrichment) of current web content as an automated process of third party annotation for making at least a part of today's web more suitable for machine processing and hence enabling it intelligent tools for searching and recommending things on the web (see [2]).

Our main idea is to fill a semantic repository with information that is automatically extracted from the web and make it available to software agents. We give a proof of concept that this idea is realizable and we give results of several experiments in this direction.

Our web crawler (see Fig.1) downloads a part of the web to the web repository (Web Store). Resources with semantic content can be uploaded directly to the semantic repository (Semantic Store). Extractor 1 (classifier) extracts those parts of Web Store which are suitable for further semantic enrichment (we are able to enrich only a part of resources). More semantic content is created by several extractors and annotators in several phases. The emphasis of this paper is on the automation and the usage of such extracted/enriched data.

The paper is structured as follows. The next section describes our main ideas introduced in this paper. Then two special sections denoted to the domain independent annotation (section 3) and domain dependent annotation (section 4) follow. Section 5 describes our user agent. Our experiments are presented in the section 6 and at the end, just before conclusion, there is a section about related work.

2 Idea of Web semantization

2.1 Web repository

First idea is the idea of a web repository. It develops as follows in details. Semantic enrichment is in fact a data mining task (although a special one) - to add to web documents a piece of knowledge, which is obvious for human perception not for a machine. That means to annotate data by concepts from

an ontology which is the same as to map instances to an ontology. Such a data mining task will be easier to solve when there is a sort of a repetition (modulo some similarity).

We decided to enrich only textual content for the present, no multimedia (this substantially reduces the size of information we have to store). Especially we restrict to the pages with content consisting either dominantly of natural language sentences (let us call them textual pages) and those containing large number of table cells (let us call them tabular pages). Of course this division need not be disjoint, and will not cover the whole web.

The Web repository is a temporal repository of Web documents crawled by a crawler. The repository supports document's metadata, e.g. modification and creation dates, domain name, ratio HTML code/text, content type, language, grammatical sentences etc. It keeps track of all changes in a document and simplifies access to and further processing of Web documents. We are experimenting with the Web crawler Egothor[1] 2.x and it's Web repos-

[1] http://www.egothor.org/

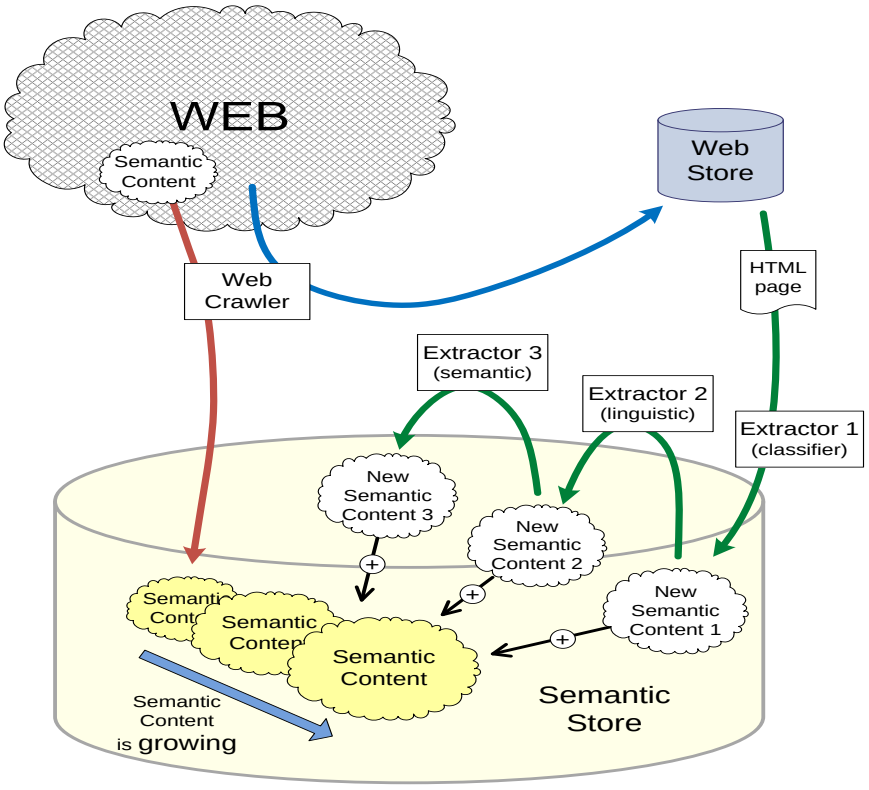

Fig. 1 The process of semantization of the Web

itory. We have been filled this repository with several terabytes of textual part of Czech web (domain *.cz) and it very simplified access to this data.

2.2 Domain (in)dependent annotation

Second idea is to split annotation process to two parts, one domain independent intermediate annotation and second domain dependent user directed annotation. Both should be automated, with some initial human assisted learning. This first part of learning could require assistance of a highly skilled expert; the second (probably faster part) should be doable by an user with average computer literacy.

The first domain independent annotation will serve as an intermediate annotation supporting further processing. This will run once on the whole suitable data. Hence initial necessary human assistance in training can be done by a highly skilled expert (indeed it can be a longer process).

The second domain dependent annotation part can consists of a large number of tasks with different domains and ontologies. This should be possible to be executed very fast and if possible with assistance of an average computer skilled user. We can afford this having intermediate annotation.

Domain independent intermediate annotation can be done with respect to general ontologies. First ontology is the general PDT tectogrammatical structure [19] (it is not exactly an ontology written in ontology language, but could be considered this way) which captures semantic meaning of a grammatical sentence. This is the task for computational linguistics; we make use of their achievements which were originally focused on machine translation. Of course tectogrammatical structure is not the only option for this purpose. English language for example can be parsed in many different ways (most often according to some kind of grammar). All the other possibilities are applicable, but in our current solution we make use of a tree structure of the annotations. In this paper we will present our experience with Czech language and tectogrammatical structure that we have used for domain independent intermediate annotation of pages dominantly consisting of grammatical sentences.

For structured survey or product summary pages (we call them "tabular pages") we assume that their structure is often similar and the common structure can help us to detect data regions and data records and possibly also attributes and values from detailed product pages. Here annotation tools will be also trained by humans – nevertheless only once for the annotation of the whole repository.

We can see an interesting inversion in the use of similarity and repetition depicted in Fig. 2.2. While for tabular pages we use similarities in the intermediate phase, for textual pages we use similarities in the domain dependent phase where similar sentences often occur.

Type of annotation	Tabular pages	Textual pages
Intermediate general	Uses similarities	Does not use similarities
Domain dependent	Does not use similarities	Uses similarities

Fig. 2 Use of similarity in annotation approaches

Domain (task) dependent (on demand) annotation is concerning only pages previously annotated by general ontologies. This makes this second annotation faster and easier. An assistance of a human is assumed here for each domain and new ontology. For textual pages repetitions make it possible to learn a mapping from structured tectogrammatical instances to an ontology. This domain (task) dependent task can be avoided by a collaborative filtering method, assuming there is enough users' acquaintance.

2.3 Semantic repository

Third important idea is to design a semantic repository. It should store all the semantic data extracted by extraction tools and accessed thorough a semantic search engine. Of course many different problems (e.g. querying and scalability) are connected with this but they are at least partially solved now days. Let us mention work of our colleges [11] that is available for use.

The semantic repository should also contain some sort of uncertainty annotation besides above mentioned ontologies. The main reason is that the annotation process is error prone and in the future we can have different alternative annotation tools so we can aggregate results. This aspect is not further described in the paper but can be found with many details in [7] and [12].

2.4 User software agent

Last, but also important idea is to design at least one software agent, which will give the evidence that the semantization really improved the usability of the Web. Besides using annotated data it should also contain some user preference dependent search capabilities. The process of a user agent searching and making use of the semantic search engine is represented in Figure 3. Our main focus in this paper is on the agent and on the extractors.

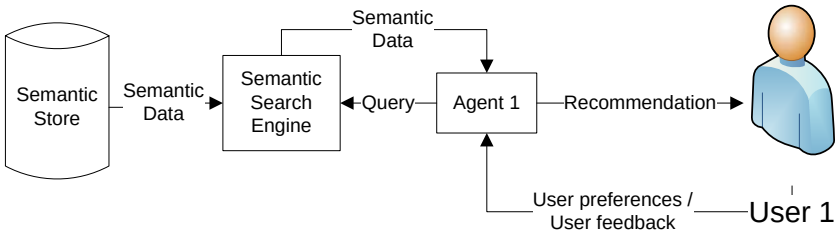

Fig. 3 Process of querying Semantic Search Engine by an user agent

3 Intermediate domain independent automated annotation

Web information extraction is often able to extract valuable data. We would like to couple the extraction process with the consecutive semantic annotation (initially human trained; later it should operate automatically). In this chapter we would like to describe details of our idea to split this process in two parts - domain independent and domain dependent.

3.1 Extraction based on structural similarity

First approach for domain independent intermediate information extraction and semantic annotation is to use the structural similarity in web pages containing large number of table cells and for each cell a link to detailed pages. This is often presented in web shops and on pages that presents more than one object (product offer). Each object is presented in a similar way and this fact can be exploited.

As web pages of web shops are intended for human usage creators have to make their comprehension easier. Acquaintance with several years of web shops has converged to a more or less similar design fashion. There are often cumulative pages with many products in a form of a table with cells containing a brief description and a link to a page with details about each particular product.

There exist many extraction tools that can process web pages like this. Most complete recent survey can be found in [4] and also in [18]. We come with similar but also different solution divided into domain dependent and domain independent phases. See below for details.

Our main idea (illustrated in the Figure 4) is to use a DOM tree representation of the summary web page and by breadth first search encounter similar subtrees. The similarity of these subtrees is used to determine the

data region - a place where all the objects are stored. It is represented as a node in the DOM tree, underneath it there are the similar sub-trees, which are called data records.

Our ontology engineering part is based on a bottom-up approach of building ontologies. The first classes are 'tabular page', 'data region' and 'data record' which are connected with properties 'hasDataRegion' and 'hasDataRecord'. Also, between data record and its detail *Data_region* and *Detailed_page* we have property 'hasDetailPage'. Note that these concepts and properties have soft computing nature which can be modeled by a fuzzy ontology. For instance, being an instance has a degree of membership depending on the precision of our algorithm (dependent on similarity measures used to detect data regions and/or records). Using some heuristics we can detect what are resources described in this page (Tabular_page describes rdfs:Resources). To detect possible attributes and even their values we explore similarities between the content of a data record and corresponding content of a detailed page. The main idea is to find same pieces of text in the data record and in the detail page. This occurs often, because a brief summary of the object (e.g. a product) is present in the data record. Somewhere near the attribute values are located the names of attributes in the detail page. These names of attributes can be extracted. The extraction of attribute names is easy because on every detail page the names will be the same. The names of attributes can be used in our low-level ontology as names of properties - each object will be represented by the URL of the detail page, and a set of properties that consists of the attribute values found on the detail page.

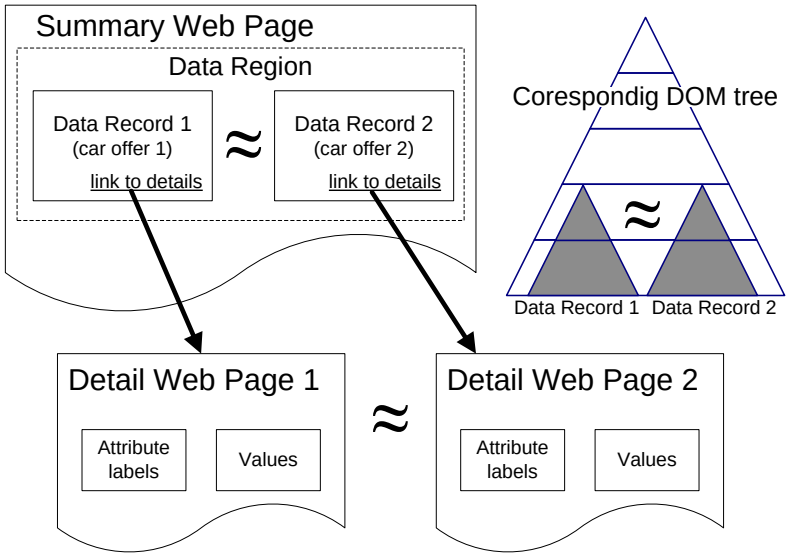

Fig. 4 Structural similarity in web pages

This idea is modification of our previous implementation from [12]. Here we decided to split the extraction process to the domain independent and the domain dependent part (see in section 4.1) so the generally applicable algorithm described here is separated form the domain dependent stuff that can be supported with a domain and an extraction ontology (see in section 4.1).

```
1 function BFSfindDR(LevelNodes)
2 begin
3   NextLevelNodes = {};
4   regions = {};
5   for each Node in LevelNodes do
6   begin
7     regions=identDataRegions(
              normalized(Node.children));
8     NextLevelNodes=NextLevelNodes  U
           (Node.Children not in regions);
9   end
10    if NextLevelNodes != {}
11      return regions U BFDfindDR(NextLevelNodes);
12    else return regions;
13 end
```

Fig. 5 Algorithm for finding data regions in DOM

In the Fig. 5 is described an algorithm for finding the data regions. The input of the function is the root element of the DOM tree of a web page. The function BFSfindDR is recursive; each run processes one level of the tree. The algorithm proceeds across each node of the input LevelNodes (5) and tries to identify if some of the node's children represents a data region (7). If so, those children are excluded from further processing (8). The nodes that are not in any data region are added to NextLevelNodes. Finally, if there are some nodes in NextLevelNodes, the function is called recursively. Data regions found on the page form the output of the function.

3.2 A method based on Czech linguistics

Our approach for Web information extraction form textual pages is based on Czech linguistics and NLP tools. We use a chain of linguistic analyzers ([16], [6], [17]) that processes the text presented on a web page and produces linguistic (syntactic) trees corresponding with particular sentences. These trees serve as a basis of our semantic extraction.

Unlike the usual approaches to the description of English syntax, the Czech syntactic descriptions are dependency-based, which means that every edge of a syntactic tree captures the relation of the dependency between a governor

and its dependent node. Especially the tectogrammatical (deep syntactic) level of representation [19] is closer to the meaning of the sentence. The tectogrammatical trees (Example of such a tree is on the Figure 6) have a very convenient property of containing just the type of information we need for our purpose (extraction of semantic information), namely the information about inner participants of verbs - actor, patient, addressee etc. So far this linguistic tool does not work with the context of the sentence and hence does not exploit a possible frequent occurrence of similar sentences.

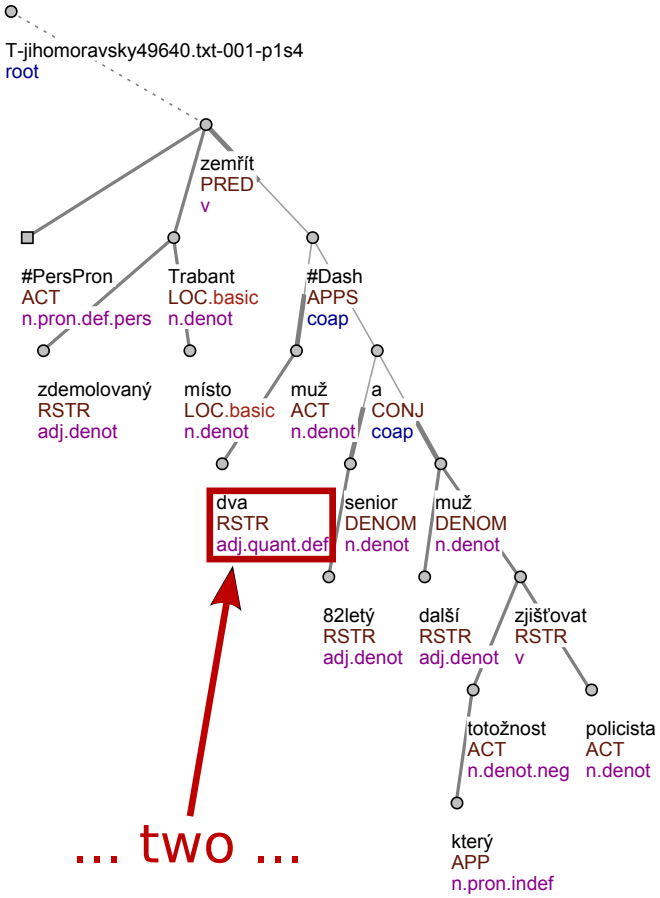

Fig. 6 A tectogrammatical tree of sentence: "Two men died on the spot in the demolished Trabant..."

4 Domain dependent automated annotation

The second phase of our extraction-annotation process is domain dependent. It makes use of the previous intermediate general (domain independent) annotation. Our goal is to make this process as fast and as easy as possible, e.g. to be trained very fast (possibly in query time) and precisely by any user with average computer skills.

4.1 Extraction and annotation based on extraction ontology

A domain ontology is the basis for an extraction ontology. The extraction ontology [14] extends the domain ontology by adding some additional information about how the instances of a class are presented on web pages. We used regular expressions to determine values of the properties. These regular expressions are matched against relevant pieces of text found on the web page in previous general annotation phase. These regular expressions are not dependent on an extraction tool, so the extraction ontology is general – it contains generally applicable information, which can be shared and imported to the particular tool from variety of locations.

So far our method of the creation of an extraction ontology has to be done by a very experienced user and has to be verified on a set of web pages. In the future we plan to invest more effort and soft computing techniques in automating this part. In this paper we do not deal with learning of extraction ontology.

4.2 Linguistic-based domain dependent annotation

Assume we have pages annotated by a linguistic annotator and we have a domain ontology. The extraction method we have used is based on extraction rules. An example of such an extraction rule is on Figure 7 (on the left side). These rules represent common structural patterns that occur in sentences (more precisely in corresponding trees) with the same or similar meaning. Mapping of the extraction rules to the concepts of the target ontology enables the semantic extraction. An example of such mapping is demonstrated in Figure 7. This method is not limited to the Czech language and can be used with any structured linguistic representation. We have published this method in [9], more details can be found there.

We experimented with obtaining extraction rules in two ways.

1. Rules and mappings were designed manually (like the rule on the Fig. 7).

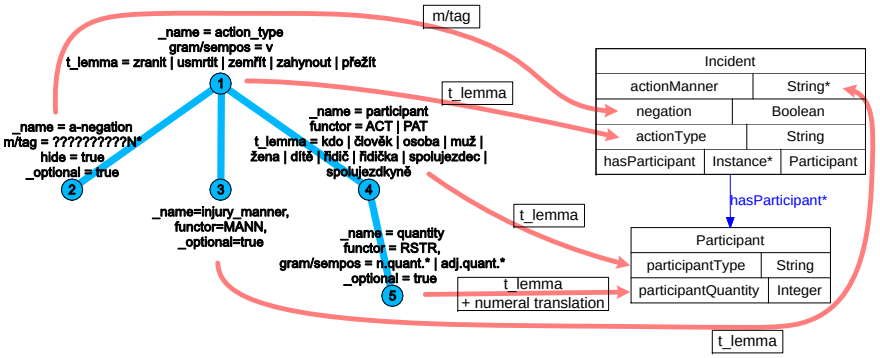

Fig. 7 An example of extraction rule and its mapping to ontology

2. Rules and mappings were learned using Inductive Logic Programming (ILP) methods (see in the following section).

Finally, having this mapping, we can extract instances of the target ontology from the answer corresponding to an extraction rule. This answer is given by the semantic repository, and the instances of the ontology are also stored there.

4.3 Using Inductive Logic Programming (ILP)

ILP [20] is a method for a generation of rules that describe some properties of data. ILP takes three kinds of input

- Positive examples E+ – objects that have the desired property.
- Negative examples E- – objects that do not have the desired property.
- Background knowledge – facts about the domain (so far we do not consider background knowledge in the form of rules).

ILP tries to generate such rules that all positive examples and no negative example satisfy them. It uses concepts from the background knowledge in the body of the rules. The main advantage of ILP is that it can be trained to mine arbitrary data described in predicate logic - the process of learning is dependent only on the positive and the negative examples and on the amount of information we have about them. The more information we have, the more precise the learning will be.

Thanks to the fact that ILP learns from positive and negative examples, we can propose an assisted learning procedure for learning and also tuning the extractor (described in previous section). The process is such that the extractor gets pure textual or semantically (pre)annotated data from a web page and presents it to the user. The user then annotates the data or only

corrects the data saying which are well and which are badly annotated. The extractor can be taught and tuned to the desired goal in this way. Of course the user has to understand the semantics of the annotations - so the user has to be an expert in that way of understanding the target ontology. Yet this ontology can be quite simple in the case we are interested only in a specific kind of information e.g. number of people injured during a car accident like in our experiments (see next section) and motivation.

We discovered that we can use a straightforward transformation of linguistic trees (see an example on the Figure 6) to predicates of ILP (example of the transformation is in the Figure 8) and the learning procedure responds with significant results, which are presented in next section.

On the Figure 6 we can see the relationship between particular words of a sentence and nodes of tectogrammatical tree - the highlighted node of the tree corresponds with the word 'two' in the sentence (also highlighted). This relationship allows a propagation of the information from the user (which annotates just the text of sentence) to the ILP learning procedure.

5 User agent

One of the aspects of our proposed agent is to compare different offers (e.g. used car offer) based on their attributes. We call this a "decathlon effect", or in economy terminology utilizing possibly "conflicting objectives". The user has some preferences on car offer's attributes; the user wants low price and low consumption. These preferences are called objectives. We can use

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Two men died on the spot in the % demolished trabant – a senior 82
% years old and another man, who’s
%%%%%%%% Nodes %%%%%%%%
tree_root(node0_0). node(node0_0).
id(node0_0, t_jihomoravsky49640_txt_001_p1s4).
node(node0_1).
t_lemma(node0_1, zemrit).
functor(node0_1, pred).
gram_sempos(node0_1, v).
node(node0_2).
t_lemma(node0_2, x_perspron).
functor(node0_2, act).
gram_sempos(node0_2, n_pron_def_pers).
...
%%%%%%%% Edges %%%%%%%%
edge(node0_0, node0_1). edge(node0_1, node0_2). edge(node0_1, node0_3).
...
edge(node0_34, node0_35).
%%%%%%%% Injury %%%%%%%%
injured(t_jihomoravsky49640_txt_001_p1s4).
```

Fig. 8 Sample from prolog representation of a sentence

fuzzy logic notation and to each attribute A_i associate its fuzzy function $f_i : D_{A_i} \rightarrow [0, 1]$, which normalizes the domain of the attribute (the higher the fuzzy value the higher the user preference).

Some of these objectives are conflicting e.g. wanting low price and low consumption. There is no clear winner, some of the car offers may be incomparable - Audi A4 has high price but also lower consumption, on the other hand Tatra T613 may have lower price but also higher consumption. Regarding two objectives "low price" and "low consumption", these two cars are incomparable. This undecidability is solved by an aggregation function (in economy called utility function), often denoted by @. This function takes preference on attributes and combines them into one global score of the car offer as a whole. In this way, the agent can order all car offers by their score. The aggregation function may be a weighted average, for example:

$$@(Price, Consumption) = \frac{3 * Price + 1 * Consumption}{4}$$

where $Price$ and $Consumption$ are the degrees to which the car offer satisfies the use criteria. We have spent much time in the research of user modelling, these ideas are discussed in detail for example in recent [13].

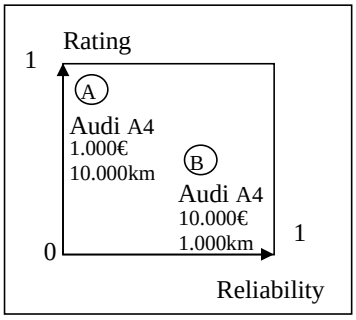

Fig. 9 An example of reliable and unreliable rating

Another phenomenon of our approach is that we take into account the (un)certainty of correct extraction. In our system we have for example two different pieces of information about the same offer of Audi A4. Let us denote "the first Audi A4" as A and the second as B (see in Fig. 9). A has much higher rating than the B. However, A has also much lower reliability. The reliability of the rating of A and B is determined by extractors that made the annotation or by a later user feedback, finding first extractor mistaken. This may be caused for example by an error of extractor that switched the price (10.000(Euro)) with the distance travelled (1.000(km)). Thus A has much better price than B and consequently also the rating. We use a second

order utility function to combine preference degree and reliability trained to specific user profile.

6 Experiments

Our experiment was to enable our agent to access semantic information from the web. We tried to gather information form car offers with information about car accidents. We wanted to find out the number of injured persons during car accidents and match them to the offers according the car make and model. We have used firemen reports; some of them were about car accidents, some were not. Each report was split into sentences and each sentence was linguistically analyzed and transformed into a tectogrammatical tree, as described above. These trees were transformed to a set of Prolog facts (see Figure 8).

Sentences, which talk about an injury during a car accident, were manually tagged by predicate `injured(X)`, where X is the ID of the sentence. Those sentences that do not talk about injured persons during a car accident were tagged as `:-injured(X)`, which represents a negative example. This tagging can be done even by a user unexperienced in linguistics and ILP, but the user has to understand the semantics of information he is tagging (it usually means that the user has to understand the target ontology). These tagged sentences were the input for ILP, we have used 22 sentences as positive examples and 13 sentences as negative examples. We used Progol [21] as ILP software. The rules ILP found are in following Figure 10.

The first four rules are overfitted to specific sentences. Only the last two represent generally applicable rules. But they do make sense - 'zranit' means 'to hurt' and 'nehoda' means 'an accident' in Czech. These two rules mean that the root element is connected either to a noun that means an accident or to a verb that means to hurt.

We tested these rules on the set of 15 positive and 23 negative examples. The result accuracy was 86.84%. Fig. 6 shows a 4-fold table where rows depict

```
injured(A) :- id(B,A), id(B,t_57770_txt_001_p5s2).
injured(A) :- id(B,A), id(B,t_60375_txt_001_p1s6).
injured(A) :- id(B,A), id(B,t_57870_txt_001_p8s2).
injured(A) :- id(B,A), id(B,t_57870_txt_001_p1s1).

injured(A) :- id(B,A), edge(B,C), edge(C,D),
              t_lemma(D,zranit).
injured(A) :- id(B,A), edge(B,C), edge(C,D),
              t_lemma(D,nehoda).
```

Fig. 10 Rules mined by ILP

	A	¬A
P	11	1
¬P	4	22

Fig. 11 Results on the test set

classification by the rule set and columns classification by the user. Results are promising, nevertheless tagging is done by unskilled user for whom the tagging is usually tedious.

In this case we learned a set of rules that identifies relevant sentences – roots of relevant tectogrammatical trees. We have made some additional experiments (some of them were published in [8]) that seem to lead to our goal. It is still a long way to go, but we understand these results as a proof of concept that ILP can be used for finding different kinds of information present in nodes and structure of linguistic trees. We for example extracted number of injured people from relevant trees when we have modified the training data. And still there is the possibility to design extraction rule manually if the learning method fails.

7 Related work

Of course the idea of exploitation of information extraction tools for the creation of the semantic web is not entirely novel. Above all we have to mention the very recognized work of IBM Almaden Research Center: the SemTag [10]. This work demonstrated that an automated semantic annotation can be applied in a large scale. In their experiment they annotated about 264 million web pages and generated about 434 millions of semantic tags. They also provided the annotations as a *Semantic Label Bureau* – a HTTP server providing annotations for web documents of 3rd parties. The architecture and component infrastructure of the Seeker system from [10] is a great inspiration to us. In this paper we are more concentrated on the richness and complexity of semantic annotations and an overview of the personalized software user agent making use of the annotations.

Another similar system is the KIM platform[2] (lately extended in [22]) – next good example of a large-scale automated semantic annotation system. Similarly to the SemTag, the KIM platform is oriented to recognition of named entities and limited amount of key-phrases. These annotations allow very interesting analysis of co-occurrence and ranking of entities together with basic useful semantic search and visualization of semantic data.

Interesting idea of "the self-annotating web" was proposed in [5]. This idea consists in that it uses globally available Web data and structures to seman-

[2] `http://www.ontotext.com/kim`

tically annotate–or at least facilitate annotation of–local resources. This idea can be seen as an inversion to ours (because we intend to make the semantic annotations globally available), but in fact the method proposed in [5] can be integrated to an automated semantic annotation system with many benefits on both sites.

To conclude this section we refer to two survey papers about Web Information Extraction Systems [4] and Semantic Annotation Platforms [23], which describe many other systems that are compatible with the idea of Web Semantization and can be also exploited in the process of gradual semantization of the web. As we mentioned above the domain dependent IE systems need initial learning for each new task. We take it into account and propose to use the learned systems only for suitable web pages selected by a prompt classifier.

8 Conclusions and further work

In this paper we have developed the idea of web semantization, which can help to arch over the gap between Web of today and Semantic Web. We have presented a proof of concept that even today it is possible to develop a semantic search engine designed for software agents. In particular contributions consists of two sorts of automated third party annotation of existing web resources, the idea of a semantic repository serving as a sample of semantized web with extension of the ontology model for additional annotation (e.g. uncertainty annotation) and a proposal of an intelligent user agent.

Future work goes in two directions. First is in the integration of specific parts of the system, in the automation of the whole process and in the extensive experiments with a larger number of web pages and different users. Second is in improving special parts of the system - either making it more precise or making it more automatic (able to train by a less qualified user).

Acknowledgment

This work was partially supported by Czech projects 1ET100300517, 201/09/H057 GAČR and MSM-0021620838.

References

1. Adida, B.: Bridging the clickable and semantic webs with rdfa. ERCIM News - Special: The Future Web **72**, 24–25 (2008). URL `http://ercim-news.ercim.org/content/`

view/334/528/
2. Berners-Lee, T.: The web of things. ERCIM News - Special: The Future Web **72**, 3 (2008). URL http://ercim-news.ercim.org/content/view/343/533/
3. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web, a new form of web content that is meaningful to computers will unleash a revolution of new possibilities. Scientific American **284**(5), 34–43 (2001)
4. Chang, C.H., Kayed, M., Girgis, M.R., Shaalan, K.F.: A survey of web information extraction systems. Knowledge and Data Engineering, IEEE Transactions on **18**(10), 1411–1428 (2006). URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1683775
5. Cimiano, P., Handschuh, S., Staab, S.: Towards the self-annotating web. In: WWW '04: Proceedings of the 13th international conference on World Wide Web, pp. 462–471. ACM, New York, NY, USA (2004). DOI http://doi.acm.org/10.1145/988672.988735
6. Collins, M., Hajič, J., Brill, E., Ramshaw, L., Tillmann, C.: A Statistical Parser of Czech. In: Proceedings of 37th ACL Conference, pp. 505–512. University of Maryland, College Park, USA (1999)
7. Dědek, J., Eckhardt, A., Galamboš, L., Vojtáš, P.: Discussion on uncertainty ontology for annotation and reasoning (a position paper). In: P.C.G. da Costa (ed.) URSW '08 Uncertainty Reasoning for the Semantic Web - Volume 4. The 7th International Semantic Web Conference (2008)
8. Dědek, J., Eckhardt, A., Vojtáš, P.: Experiments with czech linguistic data and ILP. In: F. Železný, N. Lavrač (eds.) ILP 2008 - Inductive Logic Programming (Late Breaking Papers), pp. 20–25. Action M, Prague, Czech Republic (2008)
9. Dědek, J., Vojtáš, P.: Computing aggregations from linguistic web resources: a case study in czech republic sector/traffic accidents. In: C. Dini (ed.) Second International Conference on Advanced Engineering Computing and Applications in Sciences, pp. 7–12. IEEE Computer Society (2008)
10. Dill, S., Eiron, N., Gibson, D., Gruhl, D., Guha, R., Jhingran, A., Kanungo, T., Rajagopalan, S., Tomkins, A., Tomlin, J.A., Zien, J.Y.: Semtag and seeker: bootstrapping the semantic web via automated semantic annotation. In: WWW '03: Proceedings of the 12th international conference on World Wide Web, pp. 178–186. ACM, New York, NY, USA (2003). DOI http://doi.acm.org/10.1145/775152.775178
11. Dokulil, J., Tykal, J., Yaghob, J., Zavoral, F.: Semantic web infrastructure. In: P. Kellenberger (ed.) First IEEE International Conference on Semantic Computing, pp. 209–215. IEEE Computer Society, Los Alamitos, California (2007)
12. Eckhardt, A., Horváth, T., Maruščák, D., Novotný, R., Vojtáš, P.: Uncertainty Issues and Algorithms in Automating Process Connecting Web and User, *Lecture Notes in Computer Science*, vol. 5327. Springer Verlag (2008)
13. Eckhardt, A., Horváth, T., Vojtáš, P.: Learning different user profile annotated rules for fuzzy preference top-k querying. In: H. Prade, V. Subrahmanian (eds.) International Conference on Scalable Uncertainty Management, *Lecture Notes In Computer Science*, vol. 4772, pp. 116–130. Springer Berlin / Heidelberg, Washington DC, USA (2007)
14. Embley, D.W., Tao, C., Liddle, S.W.: Automatically extracting ontologically specified data from html tables of unknown structure. In: ER '02: Proceedings of the 21st International Conference on Conceptual Modeling, pp. 322–337. Springer-Verlag, London, UK (2002)
15. Feigenbaum, L., Herman, I., Hongsermeier, T., Neumann, E., Stephens, S.: The semantic web in action. Scientific American **297**, 90–97 (2007). URL http://thefigtrees.net/lee/sw/sciam/semantic-web-in-action
16. Hajič, J.: Morphological Tagging: Data vs. Dictionaries. In: Proceedings of the 6th Applied Natural Language Processing and the 1st NAACL Conference, pp. 94–101. Seattle, Washington (2000)
17. Klimeš, V.: Transformation-based tectogrammatical analysis of czech. In: Proceedings of the 9th International Conference, TSD 2006, no. 4188 in Lecture Notes In Computer Science, pp. 135–142. Springer-Verlag Berlin Heidelberg (2006)

18. Liu, B.: Web Data Mining. Springer-Verlag (2007). URL http://dx.doi.org/10.1007/978-3-540-37882-2
19. Mikulová, M., Bémová, A., Hajič, J., Hajičová, E., Havelka, J., Kolářová, V., Kučová, L., Lopatková, M., Pajas, P., Panevová, J., Razímová, M., Sgall, P., Štěpánek, J., Urešová, Z., Veselá, K., Žabokrtský, Z.: Annotation on the tectogrammatical level in the prague dependency treebank. annotation manual. Tech. Rep. 30, ÚFAL MFF UK, Prague, Czech Rep. (2006)
20. Muggleton, S.: Inductive logic programming. New Generation Comput. **8**(4), 295– (1991)
21. Muggleton, S.: Learning from positive data. In: S. Muggleton (ed.) Inductive Logic Programming Workshop, *Lecture Notes in Computer Science*, vol. 1314, pp. 358–376. Springer (1996)
22. Popov, B., Kiryakov, A., Kitchukov, I., Angelov, K., Kozhuharov, D.: Co-occurrence and ranking of entities based on semantic annotation. Int. J. Metadata Semant. Ontologies **3**(1), 21–36 (2008). DOI http://dx.doi.org/10.1504/IJMSO.2008.021203
23. Reeve, L., Han, H.: Survey of semantic annotation platforms. In: SAC '05: Proceedings of the 2005 ACM symposium on Applied computing, pp. 1634–1638. ACM, New York, NY, USA (2005). DOI http://doi.acm.org/10.1145/1066677.1067049