

Uncertainty Issues in Automating Process Connecting Web and User

Alan Eckhardt¹, Tomáš Horváth², Dušan Maruščák¹, Róbert Novotný²,
Peter Vojtáš¹

¹ Charles University, Prague,

² P. J. Šafárik University Košice

{alan.eckhardt, dusan.maruscak, peter.vojtas}@mff.cuni.cz,
{tomas.horvath, robert.novotny}@upjs.sk

Abstract. We are interested in replacing human processing of web resources by automated processing. Based on an experimental system we identify uncertainty issues which make this process difficult for automated processing. We show these uncertainty issues are connected with Web content mining and user preference mining. We conclude with a discussion of possible future development heading to an extension of web modeling standards with uncertainty features.

Keywords: Uncertainty modeling, Uncertain reasoning, World Wide Web, Web content mining, User profile mining,

1 Introduction

The amount of data accessible on Web is a great challenge for web search systems. Using these data (and information and knowledge hidden in them) can be a competitive advantage both for companies and individuals. Hence Web search systems form a part of different systems ranging from marketing systems, competitors and/or price tracking systems to private decision support systems.

The main vision of Semantic Web [3] is to automate some web search activities that a human is able to do personally, but they are time-consuming or tedious. Using this automation of human search will speed up the process of searching, find a wider range of resources and when necessary soften and optimize our search criteria.

We quote the Uncertainty Reasoning for the World Wide Web (URW3) Incubator Group charter [22]: “...as work with semantics and services (on the Web) grows more ambitious, there is increasing appreciation of the need for principled approaches to representing and reasoning under uncertainty. In this Charter, the term «uncertainty» is intended to encompass a variety of forms of incomplete knowledge, including incompleteness, inconclusiveness, vagueness, ambiguity, and others. The term «uncertainty reasoning» is meant to denote the full range of methods designed for representing and reasoning with knowledge when Boolean truth values are unknown, unknowable, or inapplicable. Commonly applied approaches to uncertainty reasoning

include probability theory, Dempster-Shafer theory, fuzzy logic, and numerous other methodologies.” In this paper we are using term “uncertainty” in this wider (generic) understanding and we would like to contribute to these efforts (for related discussion see [23]).

In this paper we concentrate especially to issues connected with replacing human abilities on the web by software. From this point of view, some sorts of uncertainty are not “human_to_machine_web” specific, like faulty sensors, input errors, data recorded statistically, medical diagnosis, weather prediction, gambling etc. These are difficult for human alone and also outside the web.

According to Turtle and Croft [18], uncertainty in information retrieval can be found especially in three areas: “*Firstly, there is the problem of the representation and annotation of a resource (service). Difficulties arise also in case when attempting to represent the degree to which a resource is relevant to the task. The second problem is the representation of the kind of information, action, that a user needs to retrieve, perform (this need is especially difficult since it typically changes during the session). Thirdly, it is necessary to match user needs to resource concepts.*”

In our opinion, these areas of uncertainty apply also to our case, when replacing human activities on the web by software. Specific tasks connected to these three problems are depicted in Figure 1 and we will discuss them in this paper.

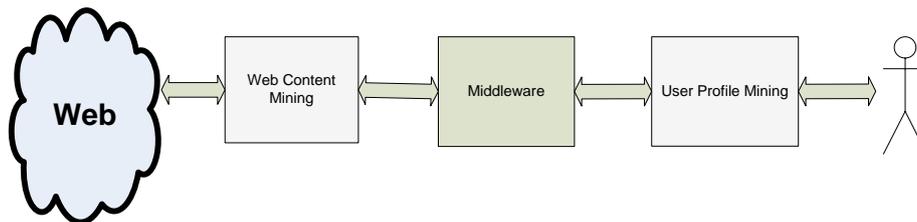


Fig. 1. Schema of an automated process connecting Web and User

Our goal is to discuss uncertainty issues based on a system integrating the whole chain of tools from the Web to the user. The uncertainty problem here appears as a problem of two inductive procedures. Two types of data mining that appear in these systems will be discussed here. One is Web content mining and second is user profile (preference) mining. Middleware will do the matching part and query evaluation optimization.

1.1 Motivating example

As a motivating example, assume that we have users looking for a hotel in a certain region. The amount of data is huge and they are distributed over several sites. Moreover users have different preferences which are soft and difficult to express in a standard query language.

From the middleware point of view, there is no chance to evaluate user’s query over all data. For middleware we have decided to use Fagin threshold algorithm [10],

which can find best (top- k) answers without looking to all objects. Fagin algorithm works under following assumptions. First, we have the access to objects (in our case hotels) in different lists ordered by user particular attribute ordering, equipped by a numerical score ranging from 0 to 1, e.g. $f_1(x) = \text{cheap}(x)$, $f_2(x) = \text{close}(x)$,... Second, we have a combination function computing total fuzzy preference value of an object based on preference values of attributes, e.g. $@(x) = ((3*\text{cheap}(x) + \text{close}(x))/4)$.

In the practical application we have to consider different users with possible different attribute orderings f_1^u, f_2^u and combination functions $@^u$. These represent the overall user preference $@^u(f_1^u, f_2^u)$ and the user profile for this task. The task for the user profile mining part is to find these particular attribute orderings and the combination function (using user's ranking of a sample of hotels).

On the web side of our system, the information of vendors, companies or advertisement is very often presented using Web pages in a structured layout containing data records. These serve for company presentation and are assumed to be mainly visited by a potential customer personally.

Structured data objects belong to very important type of information on the Web for systems dealing with competitor tracking, market intelligence or tracking of pricing information from sources like vendors.

We need to bring this data to our middleware. Due to the size of Web, the bottleneck is the degree of automation of data extraction. We have to balance the tradeoff between the degree of automation of Web data extraction and the amount of user (administrator) effort which is needed to train data extractor for a special type of pages (increasing precision).

First restriction we make is that we consider Web pages containing several structured data records. This is usually the case of Web pages of companies and vendors containing information about products and services and, in our case, hotels. Main problem is to extract data and especially attribute values to middleware.

Although we use a system which has the modules in experimental implementation, we do not present this system here. Our main contributions are

- Identification of some uncertainty issues in web content mining system and extracting attribute values from structured pages with several records
- Identification of some uncertainty issues in user profile model and using profile mining methods
- Discussion of coupling of these systems via a middleware based on Fagin threshold algorithm complemented by various storage and querying methods

We point to uncertainty issues by inserting (UNC) in the appropriate place in the text.

2 Uncertainty in Web Content Mining

In this section we describe our experience with a system for information extraction from certain types of web pages and try to point out places where uncertainty occurred.

Using our motivation as a running example, imagine a user looking for a hotel in a certain location. A relevant page for a user searching for hotels can look as on Figure 2. Comparing more similar pages would increase the chance of finding the best hotel. An automated tool would enhance this search.



Fig.2. A typical Web page containing several records

For structured Web data extraction it is possible to use semiautomatic systems like Lixto [1], Stalker [16] or WIEN [15]. These require user preannotated pages, which are used in the training process. Moreover, they are most suitable for pages, which have dynamic content, but relatively fixed structure.

Our solution is based on different approach. Instead of training techniques we use the automatic discovery of data regions which encompass multiple similar data records on the page. This is supported by an extraction ontology [9], which is used to extract the values from data records. There are many ways how to search for similar records in source tree. The system IEPAD [4] uses the features of Patricia tree (radix tree) to find the repeating sequences. This system is outperformed by the MDR system [5] which operates directly on the DOM tree of input in which it searches for repeating node sequences with same parent. However, both methods search for objects of interest in the whole web document. This can be time consuming and, as

we have experienced, it surprisingly decreases precision. Furthermore, these systems do not extract attribute values from data records.

In this paper we consider a system as a sequence of both data record extraction and attribute value selection, with possibility of ontology starting almost from scratch (e.g. user search key words).

The system will be described in several phases, which are described in the following sections.

2.1 Data Regions and Data Records Discovery

The first step in the extraction process is the retrieval of relevant web pages. For automatic localization of such resources we use the system Egothor (see [11], [12] and [24]), which is an open-source, high-performance, full-featured text search engine. This system is used for downloading the HTML source codes of relevant pages.

In the next step we build a DOM model [21] of the web page under considerations. This model is used for both data region and data records extraction. Figure 2 shows an example of relevant web page. This page contains summary information about three hotels, i. e. three data records. All of them form a single data region. Our goal is to automatically discover this data region and records within. (It should be noted that the discovery process is not limited to the single-region pages).

To reduce the search space and to increase precision, we prune the input DOM tree, omitting elements which do not contain any textual information in their subtrees. An example of such tree is shown on the Figure 3 – the numbers in black circles represent the relevance of the particular node. Zero-weighted nodes are omitted from the data record search. (UNC1) To identify nodes with relevant information in the sub-tree is the first uncertainty problem we point out in our system.

Next, we use breadth first partial tree alignment to detect data regions and records by taking element tuples, triples etc. and comparing their corresponding subtrees by various metrics (e. g. the tree Levenshtein distance) (UNC2) To tune the similarity measures for discovery of similar tags is another uncertainty problem in our system.

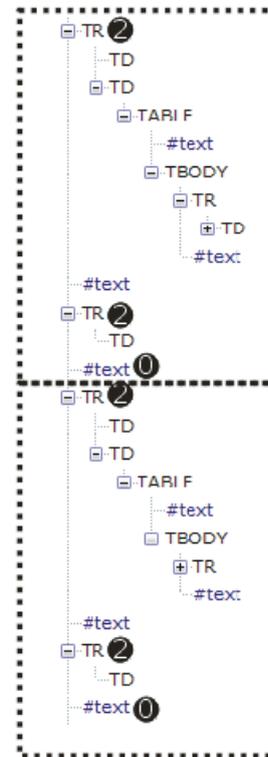


Fig.3. DOM subtree

Most often every repeated sequence of tags discovered in section 2.1 makes up a real data record (a single hotel). All attributes of this record can be found in one subtree and we can proceed to the attribute extraction using the ontology. However, the non-contiguous data records can pose a problem in the region discovery phase.

Typically a data record constitutes a single visual region, nevertheless in the HTML code can two or more records occur in a single table, which means that attributes of these records have a common subtree. It is therefore necessary to identify non-contiguous data records and separate attributes of these records (**UNC3**).

2.2 Attribute Values Extraction

As we have mentioned before, we use an ontology to extract the actual attribute values of product in the page. This ontology is dynamic – it starts from the scratch, containing user search keywords, and subsequently it evolves with new key words and typical values (using standard vocabularies). It is represented in OWL syntax with additional annotation properties and allows the specification of values extraction parameters: e. g. a regular expression which can be used to match the attribute values, an explicit enumeration of possible attribute values, or the tuning parameters (such as maximum or minimum attribute value length). It is evident, that the richer ontology leads to better results in the extraction process. An example of ontology specification can be seen on Figure 4:

```
<owl:DatatypeProperty rdf:ID="hasPrice">
  <rdfs:domain rdf:resource="#Hotel"/>
  <pl:maxLength
    rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
    10
  </pl:maxLength>
  <pl:pattern rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
    (\\$)? ?[\\d]{1,10} ?(\\.){1,3}
  </pl:pattern>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
    PRICE
  </rdfs:label>
</owl:DatatypeProperty>
```

Fig.4. An example of ontology

The extraction process can be improved in various ways. We have experimented with data extraction from the detail page (which is usually linked to the summary page), including OCR usage and a special technique based on text difference algorithm and style sheet analysis for better attribute value extraction. Additionally it is possible to employ the approximate regular expression matching, which allows to detect and repair mistyped or mismatched attribute values.

3 Middleware

3.1 Semantic Web infrastructure

User preference mining is done locally and assumes the extracted data are stored in middleware. Extracted data have to be modeled on an OWA (open world assumption) model, and hence traditional database models are not appropriate. We are compatible with a semantic web infrastructure described in [20]. The storage is based on the ideas of Data Pile described in [2]. A typical schema of record resembles a RDF statement with some statements about this statement (nevertheless we do not need reification here).

<i>resource</i>	<i>attribute</i>	<i>value</i>	<i>Extracted_from</i>	<i>Extracted_by</i>	<i>Using_Ontology</i>
Hotel1	Price	V1	URL1.html	Tool1	O1
Hotel1	Distance	D1	URL1.html	Tool1	O1

If a value of an attribute is missing, for our middleware system it means that a record is missing (thus implementing OWA). Note that we have records without any uncertainty degree attached. Any application can evaluate it according to the remaining values (e. g. it can be known that Tool1 is highly reliable on extracting price, but less on distance).

To know what we are looking for and which attribute values to extract we need to know user interest. For middleware we moreover need to know the ordering of particular attributes and the combination function.

3.2 Usage of user profiles as the user preference model

One possibility to model user preferences is to use user profiles. We work with the assumption that we have a set of user profiles P_1, \dots, P_k and we know the ideal hotel for each profile. These profiles may be created as the clusters of users or manually by an expert in the field (a hotel-keeper in our example). Manual creation is more suitable because we will know more details about user, but it is often impossible. Independently of the way profiles are created, we have ratings of hotels associated with each profile, thus knowing the best and worst hotels for that profile.

We propose computing the distance d_i of user User1 profile U_1 from each profile P_i in following way

$$d_i = \frac{\sum_{j=1, \dots, n} |Rating(User1, o_j) - Rating(P_i, o_j)|}{n} \quad (1)$$

Equation (1) represents the average difference between the user's rating of an object o_j and profile's P_i 's rating.

The ideal hotel for the user can be computed as an average of ideal hotels for each profile P_i , weighted by the inverse of distance d_i (see (2)). The average is computed on attributes of hotels. Formally,

$$\text{IdealHotel}(\text{User1}) = \frac{\sum_{i=1, \dots, k} \text{IdealHotel}(P_i) / d_i}{\sum_{i=1, \dots, k} 1/d_i} \quad (2)$$

Then, $\text{IdealHotel}(\text{User1})$ is the weighted centroid of profiles' best hotels. An example of data, user profiles' best hotel and user's best hotel is on Figure 5. User's best hotel is clearly closest to Profile 3.

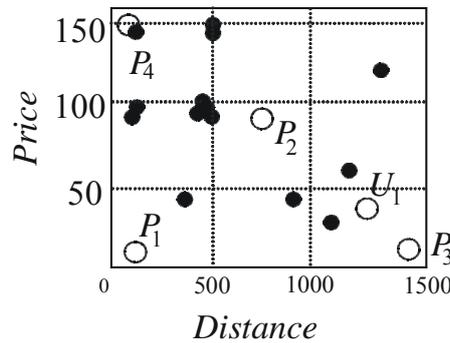


Fig. 5. Positions of best hotels for the user profiles and for the user

After the computation of the ideal hotel for the user, we will use it for computing ratings of remaining hotels. Disadvantage of this user model is that it cannot be used in the Fagin threshold algorithm.

4 Uncertainty in User Preference Mining

In our meaning, user preferences are expressed in the form of classification rules, where the values of attributes are assigned their *grades* corresponding to the orderings of the domains of these attributes. The higher the grade is, the more appropriate (preferable) the value of an attribute is for the given user. This form of grading corresponds to *truth values* well-known in fuzzy community and thus the orderings correspond to *fuzzy functions*.

The combination function can be represented by a fuzzy aggregation function (see [10]). Fuzzy aggregation functions are monotone functions of n variables, with the range of the unit interval $[0, 1]$ of real numbers (in practical applications we use only a finite part of it).

Main assumption of our learning of the user preferences is that we have a (relatively small) sample of objects (hotels) evaluated by the user. We would like to learn his/her preferences from this sample evaluation. The point is to use this learned user preference to retrieve top- k objects from a much larger amount of data. Moreover, using the user sample evaluation, we do not have to deal with the problem of matching the query language and document language. These ratings are a form of

QBE – querying by example.

There are many approaches to user modeling, one of the most used is collaborative filtering method [28]. Our method is content based filtering – it uses information about attributes of objects.

4.1 Learning Local Preferences

In [7] and [8] we have described several techniques of learning user’s preferences of particular attributes (**UNC5**) represented by fuzzy functions f_1, f_2, \dots on attribute domains. These techniques use regression methods. A problem occurs here. There can be potentially a big number of hotels of one sort (e.g. cheap ones) but the detection of user preference (cheap, medium or expensive) should not be influenced by the number of such hotels. Regression typically counts number of objects. We have introduced a special technique of discretization to get the user’s true local preference (for details see [7] and [8]).

Another approach not using regression is the following. The view of the whole domain of attribute *Price* is in Figure 6. We can see that with increasing price, the rating is decreasing. This can be formalized (details are out of the scope of this paper)

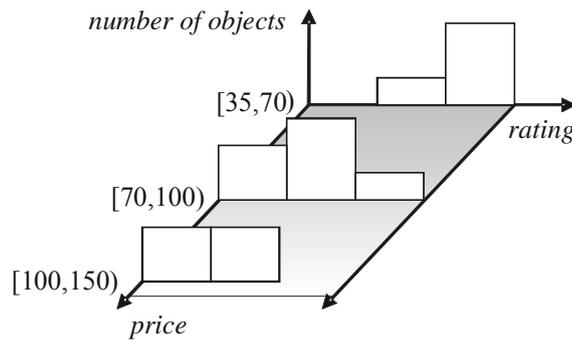


Fig. 6. Ratings for whole attribute domain

and we have experimented also with this possibility. These methods also give local preference in the form of a fuzzy function (here *small, cheap,...*) and hence are usable for Fagin Threshold algorithm.

4.2 Learning Combination Function

Second assumption of the Fagin’s model [10] is to have a combination function $@$, which combines the particular attribute preference degrees f_1, f_2, \dots (local preferences) to an overall score – $@(f_1, f_2, \dots)$ - according to which the top- k answers will be computed.

There are several ways to learn (**UNC6**) the combination functions and several models. It is an instance of classification trees with monotonicity constraints (see [17], more references to ordinal classification are presented).

We learn the aggregation function by the method of Inductive Generalized Annotated Programming (IGAP) described in [13, 14]. The result of IGAP is a set of Generalized Annotated Program rules in which the combination function has a form of a function annotating the head of the rule – here the quality of hotel:

```
User1_hotel(H) good in degree at least @( f1(x), f1(y), ...)  
IF User1_hotel_price(x) good in degree at least f1(x) AND  
    User1_hotel_distance(y) good in degree at least f2(y)
```

Note that these are rules of generalized annotated programs [25].

5 The Implementation and Experiments

Our Web content mining system has a modular implementation which allows additional modules to be incorporated (e. g. querying with preference-based querying). Communication between modules is based on the traditional Observer/Listener design pattern. All modules, which require communication with other ones, have to implement a Listener interface. All listeners are bound to the central Bus, which manages the communication between them. Each listener can specify a range of broadcasted and received events, which will be supported by it.

We proposed and implemented the middleware system for performing top-k queries over RDF data. As a Java library, our system can be used either on the server side, for example in a Web service, or on the client side. In both cases, it gathers information from local or Web data sources and combines them into one ordered list. To avoid reordering each time a user comes with different ordering, we have designed a general method using B⁺ trees to simulate arbitrary fuzzy ordering of a domain ([6]). There are several implemented classes for standard user scoring functions, and Fagin TA and NRA algorithms.

Detailed description of experiments is out of the scope of this paper. We can conclude that experiments have shown this solution is viable.

6 Conclusions and Future Work

Using an experimental implementation, in this paper we have identified several uncertainty challenges, when

- (UNC1) identifying HTML nodes with relevant information in the sub-tree,
- (UNC2) tuning similarity measures for discovery of similar tag subtrees,
- (UNC3) identifying single data records in non-contiguous html source,
- (UNC4) extracting attribute values
- (UNC5) learning user's preferences of particular attributes
- (UNC6) learn the user preference combination function.

We have experimented with some candidate solutions.

Models and methods in these experiments can be based on models of fuzzy description logic (FDL).

One possibility is to use a FDL with both concepts and roles fuzzified (see e. g. [26]). One problem of embedding FDL with fuzzy roles into OWL is that they consist of subject, predicate, object and the fuzzy value. This cannot be directly modeled by RDF data.

Second possibility is to use a FDL where only concepts are fuzzified and roles remain crisp (and hence both roles and fuzzy concepts can be modeled by RDF data). One such example is *fEL@* introduced in [27].

Acknowledgement. This work was supported in part by Czech projects IET 100300517 and IET 100300419 and Slovak projects VEGA 1/3129/06 and NAZOU.

References

1. Baumgartner, R., Flesca, S., Gottlob, G.: Visual Web Information Extraction. VLDB Conference, 2001
2. Bednárek, D., Obdržálek, D., Yaghob, J., Zavoral, F.: Data Integration Using DataPile Structure, In: Proceedings of the 9th East-European Conference on Advances in Databases and Information Systems, ADBIS 2005, Tallinn, ISBN 9985-59-545-9, 2005, 178-188
3. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. In: Scientific American Magazine, May 2001
4. Chang, C-H., Lui, S-L.: IEPAD: Information extraction based on pattern discovery. WWW-10, 2001.
5. Liu, B., Grossman, R., Zhai, Y.: Mining Data Records in Web Pages. In: Proc S IGKDD.03, August 24-27, 2003, Washington, DC, USA.
6. Eckhardt, A., Pokorny, J., Vojtas, P.: A system recommending top-*k* objects for multiple users preferences. In: 2007 IEEE Conference on Fuzzy Systems, IEEE 2007, 1101 – 1106
7. Eckhardt, A., Horváth, T., Vojtáš, P.: PHASES: A User Profile Learning Approach for Web Search. Accepted as short paper for WI'07 Web Intelligence Conference, November 2007, Fremont CA
8. Eckhardt, A., Horváth, T., Vojtáš, P.: Learning different user profile annotated rules for fuzzy preference top-*k* querying. Accepted for SUM'07 Scalable Uncertainty Management Conference, October 2007, Washington DC Area
9. Embley, D. W., Campbell, D. M., Smith, R. D., Liddle, S. W.: Ontology-Based Extraction and Structuring of Information from Data-Rich Unstructured Documents. CIKM 1998, 52-59
10. Fagin R., Lotem A., Naor M.: Optimal Aggregation Algorithms for Middleware. In Proc. 20th ACM Symposium on Principles of Database Systems, 102-113, 2001
11. Galamboš L.: Dynamization in IR Systems. In: Proc. IIPWM '04 - Intelligent Information Processing And Web Mining, ed. M. A. Klopotek, Springer 2004, 297-310
12. Galamboš, L.: Semi-automatic stemmer evaluation, *ibid.* 209-218.
13. Gurský, P., Horváth, T., Novotný, R., Vaneková, V., Vojtáš, P.: UPRE: User preference based search system, 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI'06), IEEE 2006, pp.841-844

14. Horváth, T., Vojtáš, P.: Ordinal Classification with Monotonicity Constraints. In: Proceedings of the 6th Industrial Conference on Data Mining (ICDM '06), Leipzig, Germany, 2006: LNAI 4065, Springer, 2006, ISBN 3-540-36036-0, p: 217-225
15. Kushmerick, N.: Wrapper induction: efficiency and expressiveness. *Artificial Intelligence*, 118:15-68, 2000
16. Muslea, I., Minton, S., Knoblock C.: A hierarchical approach to wrapper induction. *Conf. on Autonomous Agents*, 1999
17. Potharst, R., Feelders, A. J.: Classification trees for problems with monotonicity constraints. In: *ACM SIGKDD Explorations Newsletter archive Volume 4 , Issue 1 (June 2002)*: ACM Press, 2002, p: 1-10
18. Turtle, H. R., Croft, W. B.: Uncertainty in Information Retrieval Systems. In: *Proc. Second Workshop Uncertainty Management and Information Systems: From Needs to Solutions*, Catalina, Calif., 1993 as quoted in S. Parsons. *Current Approaches to Handling Imperfect Information in Data and Knowledge Bases. IEEE TKDE* 8,3 (1996) 353-372
19. Vojtáš, P.: EL description logic with aggregation of user preference concepts. In: Duží, M. et al. Eds. *Information modeling and Knowledge Bases XVIII*, IOS Press, Amsterdam, 2007, 154-165
20. Yaghob, J., Zavoral, F.: Semantic Web Infrastructure using DataPile, In: *Proc. 2006 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, Butz, C. J. et al. (eds.), IEEE 2006, 630-633
21. W3C Document Object Model. <http://www.w3.org/DOM/>
22. Charter of W3C Uncertainty Reasoning for the World Wide Web Incubator Group, <http://www.w3.org/2005/Incubator/urw3/charter>
23. Wiki of W3C Uncertainty Reasoning for the World Wide Web XG Search: <http://www.w3.org/2005/Incubator/urw3/wiki/FrontPage>
24. <http://www.egothor.org/>
25. Kifer, M., Subrahmanian, V. S.: Theory of generalized annotated logic programming and its applications, *J. Logic Programing*, 12 (1992) pp 335–367
26. Stoilos, G., Stamou, G., Tzouvaras, V., Pan, J. Z., Horrocks, I.: The fuzzy description logic f-shin. *Proceedings of URSW*, 2005.
27. Vojtáš, P.: EL description logic with aggregation of user preference concepts. M. Duží et al. Eds. *Information modelling and Knowledge Bases XVIII*, IOS Press, Amsterdam, 2007, 154-165
28. Aggarwal, C.: *Collaborative Crawling: Mining User Experiences for Topical Resource Discovery*, IBM Research Report, 2002.