SCALABLE COMPUTING: PRACTICE AND EXPERIENCE
Volume 9, Number 1, pp. 51–59. http://www.scpe.org
ISSN 1895-1767

KNOWLEDGE PROCESSING FOR WEB SEARCH—AN INTEGRATED MODEL AND EXPERIMENTS

P. GURSKÝ* AND T. HORVÁTH*, J. JIRÁSEK*, R. NOVOTNÝ*, J. PRIBOLOVÁ*, V. VANEKOVÁ* AND P. VOJTÁŠ†

Abstract. We propose a model of a middleware system enabling personalized web search for users with different preferences. We integrate both inductive and deductive tasks to find user preferences and consequently best objects. The model is based on modeling preferences by fuzzy sets and fuzzy logic. We present the model-theoretic semantics for fuzzy description logic f-$\mathcal{EL}$ which is the motivation of creating a model for fuzzy RDF. Our model was experimentally implemented and the integration was tested.

Key words. middleware, fuzzy DL, fuzzy RDF, relevant objects, user preferences

1. Introduction and Motivation. One of the main goals of semantic web is to enable easy and automatic access to web resources and services by middleware engines or agents. Our research leads to the model of a middleware system which will help users in searching for objects from heterogenous sources but a single domain. In this paper we present different approaches to the most important aspect of such system—retrieving the best objects according to user's preferences.

Let us consider the following example: imagine a user looking for a hotel which has *good price*, *good distance from an airport* and has *good equipment in rooms*.

Each user (or group of users) has his own sense of quality (i. e. the preference). For the price of hotels, one user could prefer cheap hotels (student), second prefers expensive hotels (manager) and the other one prefers middle price (professor). The underlying meaning of this ordering is that the user determines the relations "better" or "worse" between two values of a given property. For each such property, user has a notion about the ordering of objects based on real value of property from an attribute domain. We call these orderings of particular attribute domains the *user local preferences*.

Nevertheless, these local preferences usually lead to incomparable objects, e. g. one hotel that is cheaper than another (which means that the latter is better for a student), but has inferior room equipment. The combination of local preferences gives global preference and will be modeled by fuzzy aggregation operators.

The main contributions of this paper are integration of several methods for local and global preferences into one framework, and practical introductory experiment with our system.

The paper is organized as follows: section 2 introduces methods for detecting user preferences and searching for relevant objects and it provides a theoretical model of these preferences based on description logic. Section 3 describes the system and experiments. Finally, section 4 concludes and provides some plans for future research.

2. Models and Methods. In this chapter we describe models and methods we use for a solution of the problem of web search. Later these methods are implemented and tested.

2.1. Detecting Local Preferences. To learn local preferences, we have several possibilities. First is, we present to user a representative sample of hotels (see Table 2.1) with several attributes, e. g. distance from the airport, price of the accommodation and equipment of rooms. The user classifies hotels into categories *poor*, *good* and *excellent* according to the relevance of the hotel to him. In practice we have chosen the seven classes of Likert scale.

We distinguish four basic types of local preferences, according to user's most preferable attribute values. We call these basic types *higher-best*, *lower-best*, *middle-best* and *marginal-best.*

The local preferences can be detected by statistical methods, e. g. regression or QUIN [4]. Using the linear regression we can detect just two basic types (*higher-best* and *lower-best*). In contrast to QUIN, the regression is resistant against statistically irrelevant values. Thus for the purposes of detecting the aforementioned four basic types, the polynomial regression is the most appropriate approach. In case of the Table 2.1, we checked that the higher distance (*higher-best* type of ordering) and the lower price (*lower-best* type of ordering) are appropriate for the user. Albeit this method is not yet used in the experimental implementation, it is useful for the motivation purposes.

The second possibility is to learn local preferences explicitly from the user. The user has possibility to specify his preferences by explicit choice of fuzzy functions. We have used this method in our experiments (see Section 3).

2.2. Learning Global Preferences. We learn user's global preferences by the method of ordinal classification with monotonicity constraints [9] based on Inductive Logic Programming (ILP) system ALEPH [15].

*Institute of Computer Science, Šafárik University, Košice, Slovakia, ({name.surname}@upjs.sk).
†Department of Software Engineering, Charles University, Prague, Czech Republic, (Peter.Vojtas@mff.cuni.cz).

TABLE 2.1
Representative sample of hotels evaluated by the user

Hotel	Distance	Price	Equipment	Evaluation
Apple	100 m	$ 99	nothing	poor
Danube	1300 m	$ 120	TV	good
Cherry	500 m	$ 99	internet	good
Iris	1100 m	$ 35	internet, TV	excellent
Lemon	500 m	$ 149	nothing	poor
Linden	1200 m	$ 60	internet, TV	excellent
Oak	500 m	$149	internet, TV	good
Pear	500 m	$ 99	TV	good
Poplar	100 m	$ 99	internet, TV	good
Rhine	500 m	$ 99	nothing	poor
Rose	500 m	$ 99	internet, TV	excellent
Spruce	300 m	$ 40	Internet	good
Themse	100 m	$ 149	internet, TV	poor
Tulip	800 m	$ 45	internet, TV	excellent

The user's global preferences are computed by using his local preferences which can be well represented in ILP. For illustration, consider that we have discretized values of distance to three classes: *near*, *middle* and *far*. The different orderings of these classes for different users can be:

$$\begin{array}{rcccl} \text{near} & \leq & \text{middle} & \leq & \text{far} \\ \text{near} & \leq & \text{far} & \leq & \text{middle} \\ \text{far} & \leq & \text{middle} & \leq & \text{near} \\ \text{far} & \leq & \text{near} & \leq & \text{middle} \\ \text{middle} & \leq & \text{near} & \leq & \text{far} \\ \text{middle} & \leq & \text{far} & \leq & \text{near} \end{array}$$

An usual aggregation function can be easily simulated by monotone classification rules in the sense of many valued logic (on Figure 2.1 the computed user's global preferences from our illustrative data are presented):

$$\begin{array}{lcl} \text{evaluation} & = & \text{excellent IF distance } \geq \text{ 500 AND price } \leq \text{ 99} \\ & & \text{AND services } = \{\text{TV, internet}\} \\ \text{evaluation} & = & \text{good IF (distance } \geq \text{ 500 AND services } = \{\text{TV}\}) \\ & & \text{OR (price } \leq \text{ 99 AND services } = \{\text{internet}\}) \end{array}$$

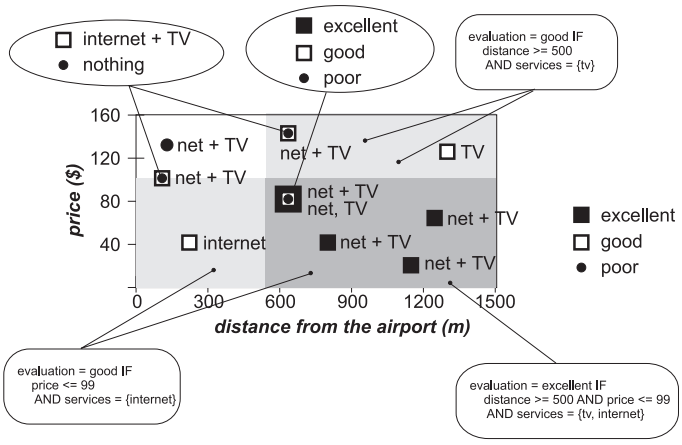

FIG. 2.1. *The results of our approach in the case of our illustrative example*

We can see that the ordered meaning of the classification is preserved in our results (this is proved in [10] as the *igap-consistency* of our approach): hotels classified in the grade "excellent" by the user also fulfills requirements for "good"

and "poor" hotels, and "good" hotels fulfills requirements for "poor" ones (e. g. the hotel with 800 m distance from airport and $45 price equipped with internet and TV is "at least as appropriate as" the hotel 300 m far from the airport and $40 price equipped just with internet) according to the local preferences (more far and more cheap is the better).

From our results we can obtain also additional information about crisp attributes (e. g. equipment). The meaning of any classification rule is as follows: If the attributes of object x fulfill expressions on the right side of the rule (body) then the overall value of x is at least the same as on the left side of the rule (head). We can, of course, assign the explicit values to vague concepts like excellent or good. During the simulation of computation of aggregation function we can simply test the validity of requirements of the rules from the strongest rule to weaker ones. When we find the rule that holds, we can say that the overall value of the object is the value on the left side of the rule. Since we test the sorted rules, we always rank the object with the highest possible value.

2.3. Fuzzy RDF Based on Fuzzy Description Logic. In this section we analyze the model of fuzzy RDF/OWL based on a model of fuzzy description logic. The terms like "cheap", "expensive" or "near" represent fuzzy sets. Our model of fuzzy RDF includes such fuzzy sets stored as RDF triples.

One important feature of our model is that we can prepare fuzzy RDF independently from user global preference. This is because we can adapt to user by adjusting his aggregation function @. This makes the processing of data more effective, because we do not need to order data for every user query. We already have the data ordered and we just combine the relevancies from fuzzy RDF into one result.

We introduce the model of building fuzzy RDF based on fuzzy description logic f-$\mathcal{EL}$ proposed in [17]. This logic removes some features of both classical and fuzzy description logic (like negation, universal restriction and fuzzy roles). On the other hand it adds an aggregation operator @. It should be also noted, that we lose the ability to describe fuzziness in roles. However, our data from the domain ontology are crisp (we do not consider uncertainty in values). User preferences are represented as fuzzy concepts and they are the source of fuzziness in results. Thus, we gain combination of particular user preferences to a global score by his @ function. An advantage of this description logic is lower complexity of querying. Expressivity is lower than that of full fuzzy description logics (DL) but still sufficient for our task and embedability into web languages and tools (see [16]).

Concepts and roles are the basic building blocks of every description logic. Here, roles express properties of resources, in our case hotels. Although the basic model of expressing RDF triples $\langle$*subject*, *predicate*, object$\rangle$ corresponds to oriented graphs, we use here the language of logic: *predicate*(*subject*, *object*).

The alphabet consists of sets $\mathcal{N}_C$ of concepts names, $\mathcal{N}_R$ role names and $\mathcal{N}_I$ instance names. The roles in f-$\mathcal{EL}$ are crisp and the concepts are fuzzy. Our language of description logic further contains a constructor $\exists$ and a finite set of aggregation functions symbols $@_U$ for each user and/or for each group of users. Concept descriptions in f-$\mathcal{EL}$ are formed according to the following syntax rules

$$C \to \top \mid A \mid @(C_1, \dots C_n) \mid \exists r.C$$

In order to give this syntax a meaning, we have to define interpretations of our language. In f-$\mathcal{EL}$ we have interpretations parameterized by a (possibly partially, usually linearly) ordered set of truth values with aggregations. For a preference structure (a set of truth values $P = [0, 1]$), a P-interpretation is a pair $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \bullet^{\mathcal{I}} \rangle$ with nonempty domain $\Delta^{\mathcal{I}}$ and fuzzy interpretation of language elements:

- $A^{\mathcal{I}} : \Delta^{\mathcal{I}} \to P$, for $A \in \mathcal{N}_C$
- $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, for $r \in \mathcal{N}_R$
- $(\exists r.C)^{\mathcal{I}} = \sup\{C^{\mathcal{I}}(y) : (x, y) \in r^{\mathcal{I}}\}$
- $(@(C_1, \dots, C_n))^{\mathcal{I}}(x) = @^{\bullet}(C_1^{\mathcal{I}}(x), \dots, C_n^{\mathcal{I}}(x))$

Suppose that we have $\mathcal{N}_R = \{$*price*, *distanceFromAirport*$\}$ where both elements of this set represent the RDF predicates from the domain ontology.

$\mathcal{N}_C = \{cheap_U, close_U\}$ where $cheap_U$ and $close_U$ are fuzzy functions explicitly figured by user preference ontology.

$\mathcal{N}_I = \{$*Apple*, *Danube*$, 99, 120, \dots\}$.

In Herbrand-like interpretation $\mathcal{H}$ we have:

$$Apple^{\mathcal{H}} = \text{Apple}, Danube^{\mathcal{H}} = \text{Danube}, 99^{\mathcal{H}} = 99, 120^{\mathcal{H}} = 120$$
$$cheap_U^{\mathcal{H}}(99) = 0.53, cheap_U^{\mathcal{H}}(120) = 0.42$$
$$price^{\mathcal{H}} = \{(\text{Apple}, 99), (\text{Danube}, 120)\}$$

For the sake of simplicity we overload our concept $cheap_U$ also for hotels (usually we must create new concepts in this case e. g. $cheapHotel_U$):

$$\begin{aligned} cheap_U^{\mathcal{H}}(x) &= (\exists price.cheap)^{\mathcal{H}}(x) \\ &\text{then} \\ cheap_U^{\mathcal{H}}(\text{Apple}) &= \sup\{cheap_U^{\mathcal{H}}(y) : (\text{Apple}, y) \in price^{\mathcal{H}}\} = 0.53 \\ cheap_U^{\mathcal{H}}(\text{Danube}) &= \sup\{cheap_U^{\mathcal{H}}(y) : (\text{Danube}, y) \in price^{\mathcal{H}}\} = 0.42 \end{aligned}$$

The supremum affects the case when we have more different prices for one hotel. Then we take the highest result.

$$\begin{aligned} (@(cheap_U, close_U))^{\mathcal{H}}(\text{Apple}) &= @^{\bullet}(cheap_U^{\mathcal{H}}(\text{Apple}), close_U^{\mathcal{H}}(\text{Apple})) \\ &= \frac{(3 \times close_U^{\mathcal{H}}(\text{Apple}) + 2 \times cheap_U^{\mathcal{H}}(\text{Apple}))}{5} \end{aligned}$$

Fuzzy description logic f-$\mathcal{EL}$ is the motivation for creating a model for fuzzy RDF. For example, a fuzzy instance $cheap_U^{\mathcal{H}}(\text{Apple}) = 0.53$ can be modelled by the RDF triple: $\langle \text{Apple}, cheap_U^{\mathcal{H}}, 0.53 \rangle$.

This is an embedding of a fuzzy logic construct into classical RDF, which needs to translate also constructions of DL, namely $\exists price.cheap$ in fuzzy DL turns to composition of roles, where $\exists price.(\exists cheap.\ \top)$ needs an aggregate max (or top-k) extending DL with a concrete domain (see [1]). It is out of the scope of this paper to describe such DL with a concrete domain and embedding of our fuzzy DL in more detail. What we claim here is an experimental implementation of both our fuzzy DL, special crisp DL with a concrete domain and a related model of RDF with extended syntax and semantics of `owl:someValuesFrom`).

Now, in our model, we can specify user profiles and represent these profiles in Fuzzy RDF triples based on data from domain ontology. Our next task is to find relevant objects for individual users.

Consider the type of user preferring cheap hotels. There can be many users that prefer cheap hotels. We want to share the same instance of the class *cheap* for them. However, their notion of "cheapness" can be different. For example one user can say that cheap hotels have price lower than \$30, while for some other user cheap hotel ends at \$50. Fortunately we can easily change the overall evaluation of objects to reflect these individual requirements. In the following theorem $f_1, \ldots, f_n$ are functions that represent local preferences expressed in user preference ontology. Function values $f_1(x_1), \ldots, f_n(x_n)$ are literals from fuzzy RDF expressing the relevance of respective values of an object. We show that we can use the same functions for different users with the same preference ordering and adjust the aggregation function only.

DEFINITION 2.1. *Let D be a subinterval of real line and f be a bijective (either strictly increasing or strictly decreasing) fuzzy function $f = ax + b$, $a \neq 0$, $f : D \to [0,1]$ and $g : D \to [0,1]$. We say, that g preserves the ordering of f over D, if for all $x, y \in D$ $f(x) < f(y)$ implies $g(x) \leq g(y)$.*

THEOREM 2.2. *Let $f_1, \ldots, f_n$ be bijective fuzzy functions such that for each i : $f_i = a_i x + b_i$, $a_i \neq 0$, $f_i : D_i \to [0,1]$. Let $g_1, \ldots, g_n$ be partially linear functions such that they preserve the ordering of $f_1, \ldots, f_n$ over D. Let @ be an n-ary aggregation function. Then there exists n-ary aggregation function $@'$ such that*

$$(\forall x_1, \ldots, x_n) @(g_1(x_1), \ldots, g_n(x_n)) = @'(f_1(x_1), \ldots f_n(x_n)).$$

Proof. The theorem above says about existence of an aggregation function $@'$. We will show how to find this function.

We can assume that f_i and g_i are defined on the same unit interval $D = [0,1]$. Function g_i is linear over certain subintervals of $[0,1]$. We take one such subinterval and name it K_j. The following holds for f_i and g_i over K_j:

$$(\forall x_i \in K_j) f_i(x_i) = a_i(x_i) + b_i \text{ and } g_i(x_i) = c_i^j(x_i) + d_i^j$$

Now we define new function h_i for each $i \in [1, n]$ as:

$$(\forall i, j) h_i(y) = c_i^j \frac{(y - b_i)}{a_i} + d_i^j.$$

The aggregation function $@'$ is $@'(y_1, \ldots, y_n) = @(h_1(y_1), \ldots, h_n(y_n))$. If we substitute $f_i(x)$ for every y_i:

$$@'(f_1(x), \ldots, f_n(x)) = @(h_1(f_1(x)), \ldots, h_n(f_n(x_n)))$$

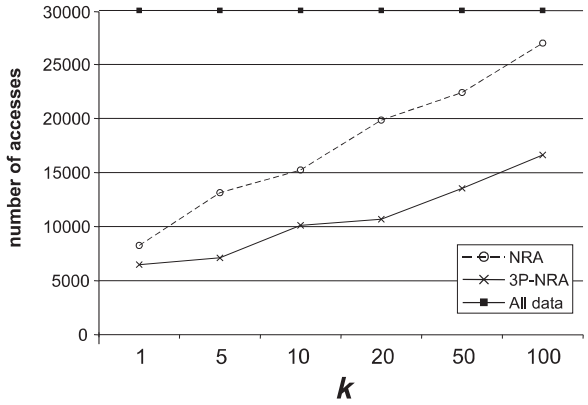

FIG. 2.2. *Number of accesses needed to retrieval of top-k objects*

For arbitrary subinterval $K_j \subseteq [0,1]$ where every g_i is linear we get

$$h_i(f_i(x_i)) = c_i^j \frac{f_i(x_i) - b_i}{a_i} + d_i^j = c_i^j \frac{a_i x_i + b_i - b_i}{a_i} + d_i^j = c_i^j x_i + d_i^j = g_i(x_i)$$

Therefore it holds

$$\begin{aligned} @'(f_1(x_1), \ldots, f_n(x_n)) &= @(h_1(f_1(x_1)), \ldots, h_n(f_n(x_n))) \\ &= @(g_1(x_1), \ldots, g_n(x_n)) \end{aligned}$$

□

This theorem allows users to specify their own meaning of "cheapness" exactly by fuzzy function and similarly for other attributes.

2.4. Relevant Object Search. Now we have orderings of properties from learning of local preferences, rules from learning of global preferences and prepared ordered data stored in fuzzy RDF. Our last task is to find top k objects which are suitable for particular user. We use the extension of middleware search of Ronald Fagin [6]. The main idea is to browse only necessary data until the system is sure that it has top-k objects already. Thus we do not need to calculate with whole data from domain ontology. Limiting the retrieval to the k best objects is often sufficient for the user and it saves time. The time efficiency grows with the number of objects stored in domain ontology and also with the number of properties we consider.

The model in [6] works with data stored in possibly distributed lists that are ordered from the best to the worst in particular property. Fagin considered two kinds of accesses to lists: sorted and random access. The sorted access gets the next best object from the list after each access, so we can retrieve data ordered from the best to the worst. The random access asks for the value of a particular property it includes for a particular object. We want to minimize the number of accesses to lists and of course the time of searching. The random access has one a large drawback. Each random access requires searching of the value of specific object. In the case of sorted access the results are prepared immediately (values can be preordered according to various criteria). Moreover, data can be sent in blocks. These important differences are the reason that we prefer the sorted access algorithms to the random access in our implementation.

In our application we use 3P-NRA (3 Phased No Random Access) algorithm [8], which is an improvement of NRA [7] algorithm.

As we found in our experiments (see Figure 2.2), using the techniques of top-k search can significantly reduce the number of accesses and correspondingly decreases the search time, especially in the case of large set of objects.

In the experiments, the data were generated with various distributions of values. We have used 2 exponential and 2 logarithmic distributions of properties with 10000 objects and 6 types of aggregation functions. For all experiments we have considered 3 properties. Various combinations of properties and aggregation functions were used for composition of 25 inputs for algorithms. The final results show the averages of particular results.

3. Web Search and Experiments. Our models and methods were implemented, integrated and experimentally tested in the project NAZOU, atop the application domain of job offers.

To identify suitable objects for user we need to obtain his local and global preferences. The complete user dependent searching process is illustrated on Figure 3.1.

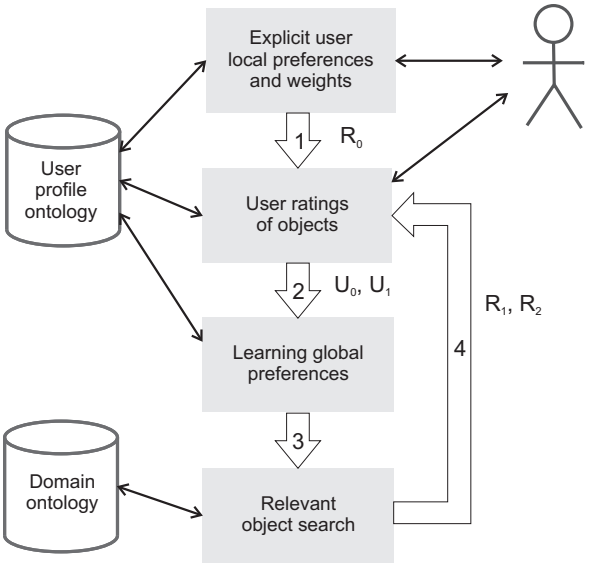

FIG. 3.1. *Flow diagram of user profile dependent part*

In our system and experiments, different users can explicitly specify local preferences. We consider job offer attributes like required education and experience level of the applicant, offered salary, manager position and the amount of traveling involved in the job (EducationLevel, ExperienceLevel, MinSalary, ManagementLevel, TravelingInvolved). The testing implementation first asks the user to specify his explicit preferences to these attributes. The user can choose if he prefers higher, lower, middle or marginal values of these attributes and he can even define how much he prefers every value (see Figure 3.2 and compare to Figure 2.1). Thus we can recognize *higher-best*, *lower-best*, *middle-best* and *marginal-best* preference types. Additionally a weight assigned to each attribute can be specified. This is used for combination function being weighted average. These parameters are used to retrieve (arrow 1) top-k objects to user. We denote this list of results as R_0.

User preferences can be changed or stated more precisely during several search cycles. We follow the idea that everybody can easily say, for a concrete object, how suitable it is. Thus we will require from user to evaluate the objects in scale from the worst to the best.

User evaluation uses 5 possible values ("worst", "bad", "neutral", "good", "best"). The k-tuple of initially retrieved objects R_0 is transformed to new set U_0 (a fuzzy set with different order induced by user ratings). This is sent (arrow 2) to our ILP tool learning global preferences. These are used (sent via arrow 3) to relevant object search to retrieve top-k objects from the whole set of objects, let us denote this by R_1. After new objects are given to user, he can evaluate them (creating U_1) and start the whole process again. We created a database log for user preferences, result sets and user ratings.

The testing involved 82 users who performed 333 cycles and rated 3547 results altogether. However, only 62 users passed at least one complete cycle successfully. The remaining users were "just browsing" or testing the system availability. In the following analysis we will consider only those users who performed at least one complete cycle, i. e. those who viewed and rated first 10 results and more precise global preferences were (possibly) found from their ratings.

Therefore we consider 62 users with 158 cycles. We find that global preferences were found from user ratings in 112 cycles, which is 71%. Successive cycles usually improve global preferences even further; the number of rules was higher or equal in 75% of successive cycles. Sometimes the number of rules oscillates from one cycle to another. This happens for 30% of users. We suppose that these users do not know exactly what they are looking for or their preferences do not remain the same throughout the testing. Although user's ratings are used for finding global preferences, they also provide us an important feedback. They define some ordering of results. If this ordering is similar to the ordering returned by top-k searching algorithm, it indicates that our preference model resembles real preferences well. We compare two orderings (the ordering R_i given by top-k algorithm and the ordering U_i given by user ratings) with Kendall tau rank correlation coefficient [11]. This coefficient determines a degree of correspondence between two orderings of the same objects.

FIG. 3.2. *Graphical user interface for explicit preference specification*

Kendall tau coefficient is defined as $\tau = \frac{2P}{\frac{1}{2}\cdot n\cdot(n-1)} - 1$ where P is the number of concordant pairs in both ratings. If two orderings are the same, they have tau coefficient 1. If one ordering is the reverse of the other, they have tau coefficient -1.

R_0 and U_0 have average tau coefficient 0.44 representing a strong correlation. If we analyze second cycles (R_1 and U_1), we have an average tau coefficient 0.6; third cycles (R_2 and U_2) have 0.58. Note that some users stopped after first or second cycle, so the number of results is smaller for the third cycle. The correlation coefficients show that the inductive methods really improve global preferences. The decrease of tau coefficient in the third cycle is very small; it can be due to inconsistent user behavior.

We present a complete record of one typical user. Table 3.1 shows that this user rated the first 10 results and this ordering by ratings was different from the ordering given by top-k algorithm (tau correlation of R_0 and U_0 is only 0.1556). However, two rules were found from user's ratings and the global preferences were refined. The second result set R_1 has tau correlation 0.5111 compared to U_1. In the third cycle, the number of rules increased to 3 and tau correlation increased to 0.9111. The user was content with his results and he stopped searching.

We can also analyze the local preference types and find which type is the most common for every attribute (see Table 3.2). It is easy to see that *higher-best* is generally the most common type and marginal-best is the least common. Lower-best type is the second most frequent for ManagementLevel and TravelingInvolved, while middle-best is the second most frequent for EducationLevel, MinSalary and ExperienceLevel. These results reflect some intuitive ideas, e.g. that most people are not satisfied with low salary, seek a job for some specific required level of education or experience and that many of them are not willing to travel.

4. Conclusions. In this paper we have described a model of system enabling users to search objects from the same domain and heterogeneous sources. Data are collected from various sources are processed to vector index and to a domain ontology. The system implements both user independent search and personalized search for users with different preferences. Our theoretical model integrates all parts of the system from collected data in classical RDF form to user query answering. We do not have a model for web resource downloading and ontology annotation part.

The model of user dependent search is based on modeling preferences by fuzzy sets and fuzzy logic. We present the semantics of fuzzy description logic f-$\mathcal{EL}$. User dependent search integrates both inductive and deductive approach. By induction we can learn local and global preferences (although currently we have implemented only global preferences). Results of induction as well as hand-filled orderings can be easily modeled in user ontology and fuzzy RDF. Deductive part of the system uses the preferences to find suitable objects for user, who can express his preferences precisely by fuzzy functions and aggregation function. The easiest way is to evaluate several objects in a scale and let the system to learn preferences. Repeating this process (evaluating a set of objects and finding new objects) leads to refining the user profile. User, who is satisfied with his profile and with the presented search results, does not have to evaluate objects again. When the domain ontology is actualized, user can just get the best objects according to his profile. Effective identification of suitable user type for a new user is the aim of our future research. It can be done by collecting more data from real users. We want to collect some personal information like age, gender, job position, etc. and the explicit specification of preferences from each user and then analyze the data in search for dependency.

Presented model was experimentally implemented and integration was tested using Cocoon and Spring Framework [14] and Corporate Memory of [5]. Both searching methods (user dependent and user independent search) are compound of tools which can be further improved separately. After the phase of gathering data about users, it will be possible to compare the efficiency, speed and complexity of these methods and decide about their practical usage. In future we plan to experiment with infrastructure of [18] and [3], which could possibly replace [14].

Our approach is used also in the Slovak project *NAZOU—Tools for acquisition, organization and maintenance of knowledge in an environment of heterogeneous information resources* [13] to find relevant job offers for the user according to his preferences.

TABLE 3.1
Testing Records for One User and Three Cycles

Offer ID	R_0	U_0	R_1	U_1	R_2	U_2	R_3
01096	2,837	1					
0f1b9	2,802	1					
01004	2,757	4					
01082	2,729	5					
01059	2,723	2					
01011	2,681	4					
9fe41	2,588	5	4	3			
8bc3a	2,588	4	4	4			
01090	2,54	5					
01095	2,54	2			3	2	
e4464			4	5	5	5	5
4c537			4	5	5	5	4
1d41d			4	1			
c56aa			4	5	5	4	4
01007			4	5	5	4	3
8f22d			4	5	5	3	3
01063			4	4	4	2	2
ef534			4	1			2
01069					3	2	
01068					3	1	
01100					3	1	
31a44							2
a8bbc							2
01036							2
τ	0,1556		0,5111		0,9111		
Rules		2		3		4	

TABLE 3.2
A Summary of Local Preference Types

Attribute	**Higher**	**Lower**	**Middle**	**Marginal**
EducationLevel	41	12	23	4
MinSalary	67	1	10	2
ExperienceLevel	50	10	15	5
ManagementLevel	43	24	8	5
TravelingInvolved	42	25	12	1

Similar strategy of communication with users (evaluation of sample objects) was used in [12], where the learning part of the system was covered by a neural network. This approach does not permit to model user preferences. We did not found any other similar approach to the whole process.

Acknowledgements. Partially supported by Czech projects 1ET100300419 and Slovak projects VEGA 1/3129/06 and NAZOU.

REFERENCES

[1] F. BAADER, R. KUESTERS, F. WOLTER, *Extensions to Description Logics*, in Description Logic Handbook, Cambridge University Press, 2003, pp. 219–261.

[2] P. BARTALOS, M. BARLA, G. FRIVOLT, M. TVAROŽEK, A. ANDREJKO, M. BIELIKOVÁ, P. NÁVRAT, *Building an Ontological Base for Experimental Evaluation of Semantic Web Applications*, in Proc. of SOFSEM 2007, Lecture Notes in Computer Science, Vol. 4362, Springer-Verlag, 2007, ISSN 0302-9743, pp. 682–692.

[3] D. BEDNÁREK, D. OBDRŽÁLEK, J. YAGHOB, F. ZAVORAL, *Data Integration Using DataPile Structure*, in Advances in Databases and Information Systems, Springer-Verlag, 2005, ISBN 3-540-42555-1, pp. 178–188.
[4] I. BRATKO, D. ŠUC, *Learning qualitative models*, AI Magazine 24 (2003), pp. 107-119.
[5] M. CIGLAN, M. BABIK, M. LACLAVÍK, I. BUDINSKÁ, L. HLUCHÝ, *Corporate memory: A framework for supporting tools for acquisition, organization and maintenance of information and knowledge*, in ISIM'06, Czech Republic, 2006, pp. 185–192.
[6] R. FAGIN, *Combining fuzzy information from multiple systems*, in J. Comput. System Sci., 58 (1999), pp. 83–99.
[7] R. FAGIN, A. LOTEM, M. NAOR, *Optimal Aggregation Algorithms for Middleware*, inProc. 20th ACM Symposium on Principles of Database Systems, 2001, pp. 102–113.
[8] P. GURSKÝ, *Towards better semantics in the multifeature querying*, in Proceedings of Dateso 2006, ISBN 80-248-1025-5, 2006, pp. 63–73.
[9] P. GURSKÝ, T. HORVÁTH, R. NOVOTNÝ, V. VANEKOVÁ, P. VOJTÁŠ, *UPRE: User preference based search system*, in Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI '06), Hong Kong, 2006, ISBN 0-7695-2747-7, pp. 841–844.
[10] T. HORVÁTH, P. VOJTÁŠ, *Induction of Fuzzy and Annotated Logic Programs*, in ILP 2006, LNAI 4455, Springer-Verlag, 2007, pp. 260–274.
[11] M. G. KENDALL, *Rank Correlation Methods*, Hafner Publishing Co., New York, 1955.
[12] E. NAITO, J. OZAWA, I. HAYASHI, N. WAKAMI, *A proposal of a fuzzy connective with learning function and query networks for fuzzy retrieval systems*, in Fuzziness in database management systems, P. Bosc and J. Kacprzyk, eds., Physica Verlag, 1995, pp. 345–364.
[13] *NAZOU. Tools for acquisition, organization and maintenance of knowledge in an environment of heterogeneous information resources*, [online, cited 11 November 2007], available from `http://nazou.fiit.stuba.sk`
[14] *Spring Framework. System for assembling components via configuration files*, [online, cited 11 November 2007], available from `http://www.springframework.org`
[15] A. SRINAVASAN, *The Aleph Manual*, Technical Report, Comp. Lab., Oxford University.
[16] P. VOJTÁŠ, *Fuzzy logic aggregation for Semantic Web search for the best (top-k) answers*, in Fuzzy logic and the semantic web, E. Sanchez, ed., Elsevier, 2006, pp. 341–360.
[17] P. VOJTÁŠ, *A fuzzy $\mathcal{EL}$ description logic with crisp roles and fuzzy aggregation for web consulting*, in Proc. IPMU'2006, B. Bouchon-Meunier et al., eds., EDK 2006, pp. 1834–1841.
[18] J. YAGHOB, F. ZAVORAL, *Semantic Web Infrastructure using DataPile*, in WI-IATW '06, Los Alamitos, California, ISBN 0-7695-2749-3, 2006, pp. 630–633.

Edited by: Maria Ganzha, Marcin Paprzycki
Received: Feb 4, 2008
Accepted: Feb 9, 2008