

Personalizing the Search for Knowledge

Minko Dudev
Max-Planck Institute for
Informatics
Saarbrücken, Germany

mdudev@mpi-inf.mpg.de

Shady Elbassuoni
Max-Planck Institute for
Informatics
Saarbrücken, Germany

elbass@mpi-inf.mpg.de

Julia Luxenburger
Max-Planck Institute for
Informatics
Saarbrücken, Germany

julialux@mpi-inf.mpg.de

Maya Ramanath
Max-Planck Institute for
Informatics
Saarbrücken, Germany

ramanath@mpi-
inf.mpg.de

Gerhard Weikum
Max-Planck Institute for
Informatics
Saarbrücken, Germany

weikum@mpi-inf.mpg.de

ABSTRACT

Recent work on building semantic search engines has given rise to large graph-based knowledge repositories and facilities for querying them and more importantly, ranking the results. While the ranking provided may prove to be acceptable in general, for a truly satisfactory search experience, it is necessary to tailor the results according to the user's interest. In this paper, we address the issue of personalizing query results in the specific setting of graph-based knowledge bases. In particular, we address two important issues: i) construction of the user profile based on the inference of the user's interest and ii) a formal model for personalized scoring which incorporates the user's interest. Preliminary experimental results show that our techniques are indeed promising.

1. INTRODUCTION

Personalization of search has been named as one of the next big challenges in information retrieval. Understanding the user and the context of her search is crucial in satisfying her information need and drastically reducing the time needed to find the "right" information. Personalization research has encompassed a wide variety of tasks, including, analyzing user click-stream data, generating user profiles, result re-ranking techniques based on user profiles, architectures for personalization, etc. [15, 13]. The main focus of this research is on the personalization of search results in the context of documents and keyword-based web search.

However, while the web is the largest repository of data, it is also, for the most part, unstructured. Several recent efforts have gone towards the building of large "knowledge" repositories – knowledge bases containing structured data extracted from the web in the form of entities and relationships (for example, Freebase¹, YAGO [14], etc.). Systems such as NAGA [7] and ExDBMS [3] provide query processing

¹<http://www.freebase.com>

facilities on such repositories. Semantic web standards such as RDF and SPARQL aim at supporting a consistent representation and querying of such semantic, structured data. With the growth of structured knowledge bases, search result ranking and personalization again become key issues in satisfying a user's information need.

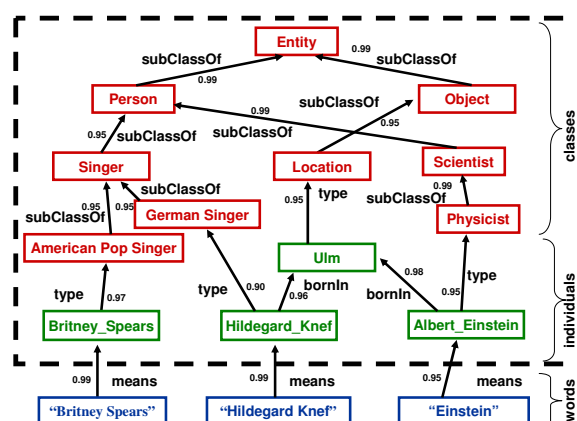


Figure 1: Portion of the YAGO knowledge graph.

In this paper, we are concerned with personalizing query results on graph-based knowledge bases. In particular, we develop techniques for personalization in the context of NAGA [7], a semantic search engine. In addition to providing query processing over a large knowledge base, NAGA also provides a scoring model which can be used as a basis to test our techniques. NAGA's knowledge base consists of several millions of facts (see YAGO [14] for details), provides a graph-based query language and returns graphs as results. A snippet of the knowledge base is shown in Figure 1. NAGA's data model is a graph, in which the nodes represent entities and the edges represent relationships between the entities. An edge in the graph with its two end-nodes forms a *fact*. As facts have been automatically extracted from the Web each one is associated with a confidence value to reflect the trust in the extraction process. It is possible to ask queries such as "When was Einstein born?" (`<Einstein BORNIN $x>`), "What do Britney Spears and Hildegard Knef have in com-

mon?” (`<Spears CONNECT Knef>` – they are both singers), etc. The results returned by NAGA are ranked according to a scoring model which takes into account the confidence of facts in the results (higher confidence facts are preferred), informativeness (“popular” facts preferred) and compactness (“tightly connected” answers preferred).

NAGA’s scoring model has been shown previously to be useful and effective (see [7] for details). Our aim is to further enhance its effectiveness by incorporating user interest into the scoring model. As a concrete example, suppose a user issues the query `<$x ISA singer>` in order to get a list of singers. The original NAGA scoring model ranks “Britney Spears” at the top. This would be acceptable if there were no prior information about the kind of music the user prefers. However, if it was known that the user had shown an interest in German music (possibly through a previous query or browsing session), then ranking “Hildegard Knef” and other German singers at the top might be of more interest to her.

Our work addresses two specific problems encountered in the personalization of structured search:

First, we propose methods to infer a user’s interest in various topics based on her interest in a limited number of facts and entities. We regard the user profile as a snippet of the knowledge graph and utilize the ontological facts present in the knowledge base to propagate scores from user-accessed entities and facts.

Second, we present a formal model for personalization which can be easily incorporated into any probabilistic scoring model. Our model is based on generative language models and incorporates personalized entity and relationship scores into the ranking. In particular, we show how to incorporate our personalized scoring model into NAGA’s ranking function.

Some previous work exists on the personalization of query results on structured data. For example, [8] develops techniques for personalizing results on relational databases and Piment [1] does the same for XML queries. We differ considerably from these works in both the setting (schemaless, graph-structured data) and approach (use of ontological facts to infer preferences). Techniques to use ontologies to derive profiles have been described in [6, 13]. While we too use ontological facts to derive user profiles, our methods for doing so and our setting (graph-based knowledge base as opposed to documents) are different.

Contributions and Outline. In a nutshell, we make the following contributions.

- a conceptualization of user profiles as snippets of the knowledge graph and methods to infer user interest across this graph.
- a probabilistic personalized ranking model, and
- a preliminary experimental evaluation as a proof-of-concept.

The remainder of the paper is organized as follows. In Section 2 we provide some background information on NAGA, the semantic search engine over which we have implemented and tested our techniques. Section 3 describes our method to generate the user profile. We then describe how to make use of the constructed user profile in order to perform personalization in Section 4. In Section 5 we present a preliminary

evaluation to assess the effectiveness of our approach. We discuss related work in Section 6 and conclude in Section 7.

2. BACKGROUND

Our work is motivated by large graph-based knowledge repositories such as [14]. While our techniques can be adapted and applied to any system where the underlying knowledge-base is a graph and the query results are graphs or trees, in this work, we incorporate our personalization strategies in the framework provided by NAGA [7], a new semantic search engine. In this section we provide a brief overview of NAGA and outline details of its ranking function.

NAGA is a semantic search engine with a large knowledge-graph of facts. It’s query language is based on SPARQL and the results are naturally regarded as graphs. As mentioned in the introduction, NAGA’s scoring model scores answer graphs based on confidence (higher confidence facts are preferred), informativeness (“popular” facts are preferred) and compactness (“tightly connected” answers are preferred), which are integrated into a unified framework. Its approach is inspired by existing work on language models for information retrieval on document collections, and is adapted and extended to the new domain of labeled and weighted graphs. Here we outline how the scoring model is derived and refer the reader to [7] for additional details.

NAGA’s scoring model assumes that a query q is generated by a probabilistic model of a result graph g . A query q is represented as $q = q_1 q_2 \dots q_k$, where q_i is a *fact template* (that is, if a fact is represented as `<x R y>`, a fact template has at least one of x , R or y unbound). Analogously, a result graph g is represented as $g = g_1 g_2 \dots g_k$, where g_i is a fact matching the fact template q_i . The ranking of a result graph is based on $P(g|q)$, which is the probability that the result graph g generated the (observed) query q . After applying Bayes formula and dropping a graph-independent constant, we have: $P(g|q) \sim P(q|g)P(g)$ where $P(g)$ is the prior and is assumed to be uniform. We now estimate $P(q|g)$ as: $P(q|g) = \prod_{i=1}^n P(q_i|g)$. The likelihood of a fact template given an answer graph is now modeled as a mixture of two distributions, $\tilde{P}(q_i|g)$ and $\hat{P}(q_i)$ as follows:

$$P(q_i|g) = \alpha \cdot \tilde{P}(q_i|g) + (1 - \alpha) \cdot \hat{P}(q_i), 0 \leq \alpha \leq 1 \quad (1)$$

$\tilde{P}(q_i|g)$ is the probability of drawing q_i randomly from an answer graph, $\hat{P}(q_i)$ is the probability of drawing q_i randomly from the total knowledge graph and α is either automatically learned (via EM iterations [4]) or set to an empirically calibrated global value. [4, 17] show the connection between this style of probabilistic models and the popular *tf · idf* heuristics.

In order to capture confidence, informativeness and compactness, $\tilde{P}(q_i|g)$ is modeled by a mixture model which puts different weights on confidence and informativeness. This is close in spirit to linear interpolation models used for smoothing [17].

$$\tilde{P}(q_i|g) = \beta \cdot P_{conf}(q_i|g) + (1 - \beta) \cdot P_{info}(q_i|g) \quad (2) \\ 0 \leq \beta \leq 1$$

For details of the estimation of each of the component probabilities as well as the background model $\hat{P}(q_i)$, we refer the reader to [7]. In summary, we use P_{NAGA} to refer

to the ranking provided by NAGA. We develop a new scoring model P_{user} which ranks results based only on the user preference. We then combine P_{user} with P_{NAGA} to get the personalized scoring model, $P_{personalized}$. In the next section, we describe the generation of user profiles and then develop the model for $P_{personalized}$ in Section 4.

3. USER PROFILE

A user profile corresponding to a single user is, conceptually, the knowledge graph with interest scores attached to entities and facts. Intuitively, we associate interest scores to entities and facts which have been accessed by the user². However, since the number of elements which the user accesses is a very small fraction of the actual knowledge base, we allow propagation of interest scores to the neighbors of the accessed entities, as well as to related facts in order to reason about the context of her interests and to incorporate them in future queries. We discuss how these scores are assigned and propagated in this section.

3.1 Entity Score Assignment and Propagation

We first consider the problem of assigning and propagating interest scores to entities.

Score Assignment. Initially, all entities have an interest score of $\epsilon > 0$, a very small default interest score. When the user accesses an entity k , the interest score of that entity $P_{in}(k)$ is updated to:

$$P_{in}(k) = \frac{\#accesses(k)}{\sum_n \#accesses(n)}$$

where the numerator is the total number of times k has been accessed and the denominator is the total number of entity accesses in the knowledge graph.

Score Propagation. The interest score described only accounts for the entities the user already knows. A personalization based only on these interest scores would limit the user in her desire of discovering new entities, and would miss the opportunity of generalizing more abstract user interests from the observed user access patterns. And so, in order to infer the user’s interest, we allow for the propagation of scores from accessed entities.

Given that an entity k was accessed, we first determine which other entities are candidates for score propagation. A natural candidate is the class of the entity accessed. Consider the example depicted in Figure 2. In the first iteration, the entity Britney_Spears is accessed. Thus, we infer in the second iteration that the user may be interested in other entities of the class American_Pop_Singer. And so, we propagate a portion of the updated interest score for Britney_Spears to the American_Pop_Singer. Similarly, we can also propagate a portion of the score to other entities of the class American_Pop_Singer (for example, Sheryl_Crow). To avoid cycles, each edge can be traversed at most once during propagation. Thus there is no interest score propagated back from American_Pop_Singer to Britney_Spears in this step. Further, we continue to propagate interest scores up

²We assume that we are able to determine user interest in facts and entities when she accesses them through an appropriate user interface.

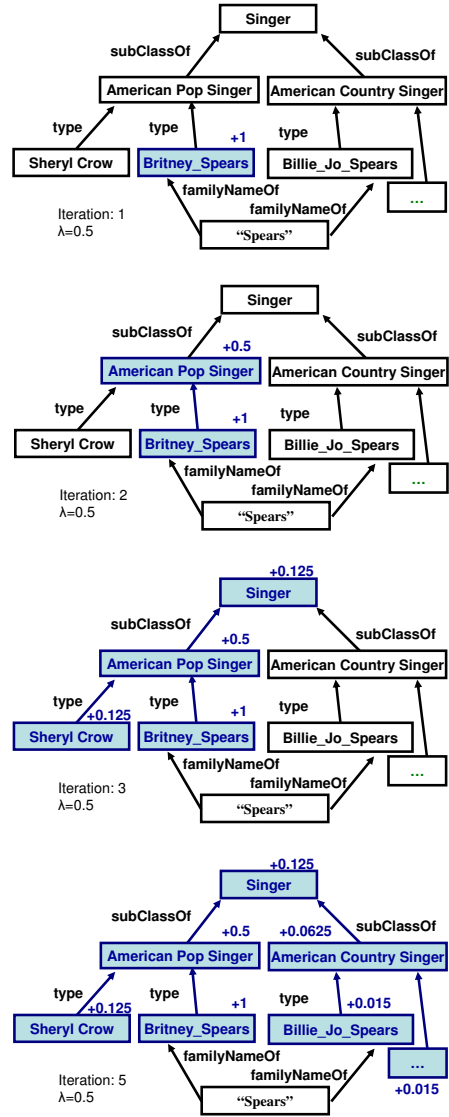


Figure 2: Entity score propagation

the class hierarchy until the root node entity. And so, the interest scores can flow from entity to class, from class to class and from class to entity, but not from entity to entity. Thus in our sample propagation, there is no flow of interest from Britney_Spears to Spears across the FAMILYNAMEOF relationship. This is to prevent interest scores from topic drifts to rather unrelated concepts. E.g., via the relationship FAMILYNAMEOF Britney_Spears is connected to Heather_Spears, a Canadian artist and poet.

More formally, our entity interest propagation scheme can be described as follows. Let k be the node whose interest score was updated. Let Q be the set of its qualifying neighbors which are eligible for score propagation (that is, nodes reachable through the edges labeled SUBCLASSOF and TYPE), but to which the score has not yet been propagated. For each $i \in Q$, we update the interest score using the formula:

$$P_{in}^*(i) = P_{in}(i) + \frac{P_{in}(k) \cdot \lambda}{|Q|}$$

where the left hand side denotes the updated interest score for entity i , $0 \leq \lambda \leq 1$ is a *damping factor* to control the amount of score to be propagated, and $|Q|$ is the number of qualifying nodes. The above formula propagates equal damped scores to each qualifying node. The propagation is repeated for each entity in Q until there are no more entities left, or the propagated score mass ($P_{in}(i) \cdot \lambda$) is less than a pre-defined threshold. Finally, a normalization of the obtained entity interest scores lets us cast $P_{in}^*(i)$ into the probability of user interest in entity i .

3.2 Fact Score Assignment and Propagation

The user may be interested in two different entities in the knowledge graph. If these entities are connected by a relationship, we may be tempted to infer that the user is interested in the fact denoted by these two nodes and the edge connecting them. However, since the user could have independently accessed either entity in different contexts, this inference may not be valid. And so, we assign interest scores to facts separately.

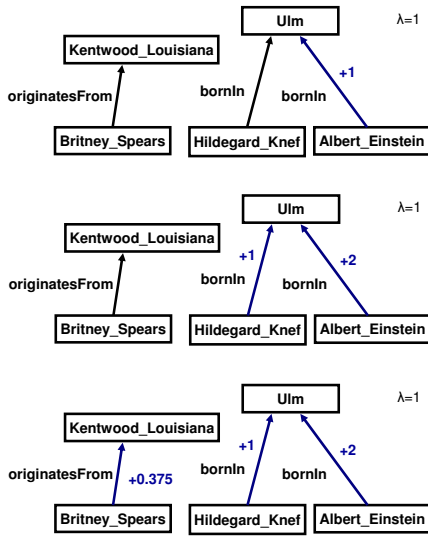


Figure 3: Fact score propagation

Score Assignment. Initially, all facts have an interest score of $\epsilon > 0$, a very small default score. Analogous to the case of entities, when the user accesses a fact f , the interest score of that fact is updated to:

$$P_{in}(f) = \frac{\#accesses(f)}{\sum_n \#accesses(n)}$$

where the numerator is the total number of times f has been accessed and the denominator is the total number of fact accesses in the knowledge graph.

Score Propagation. In order to infer the user’s interests, we need to propagate interest scores from specific facts to other “related” facts. We need to first identify candidates for score propagation. We have at least a couple of cases to consider here: i) facts with the same relation as the accessed fact, ii) facts with “similar” relationship types as the accessed facts.

For the first case, we propagate equal score to each fact with the same relation as the accessed fact. Let f be the edge accessed. Let F be the set of facts with the same relationship as f . For each fact $i \in F$, we update its score to:

$$P_{in}^*(i) = P_{in}(i) + \frac{P_{in}(f) \cdot \lambda}{|F|}$$

where the left hand side denotes the updated interest score for fact i , $0 \leq \lambda \leq 1$ is a damping factor to control the amount of score to be propagated. As with the case of entities, only a damped score is added to each of the other facts. In the example in Figure 3, interest is thus propagated from the fact $\langle \text{Albert_Einstein BORNIN Ulm} \rangle$ to $\langle \text{Hildegard_Knef BORNIN Ulm} \rangle$.

Next, we consider relationships which are similar to the relation of the accessed fact. For example, BORNIN and ORIGINATESFROM are similar relationships since the entities that can have these relationships have the same classes – **person** and **location**. However, the same entity may belong to different classes. And so, in order to more precisely quantify the similarity, we utilize a similarity metric. Let $\langle x R y \rangle$ denote a fact. Let $left$ denote the set of classes to which x belongs and let $right$ denote the set of classes to which y belongs. Note that all entities belong to the class entity. Let i and j be two sets of facts corresponding to relations R_i and R_j respectively. The similarity between R_i and R_j is computed as follows:

$$similarity_{ij} = \left(\frac{|left_i \cap left_j|}{\max(|left_i|, |left_j|)} + \frac{|right_i \cap right_j|}{\max(|right_i|, |right_j|)} \right) / 2$$

Let f be the edge accessed and R be the set of relationships in the knowledge graph which are different from the relation of f . For each $i \in R$, we first compute the similarity score $S(i)$ using the formula above. Let F_i be the set of facts with relationship i . For each fact $f_i \in F_i$, we update the interest score as follows:

$$P_{in}^*(f_i) = P_{in}(f_i) + \frac{P_{in}(f) \cdot \lambda}{|F_i|} \cdot S(i)$$

where λ is the damping factor. Note that the similarity $S(i)$ acts as a weighting factor to give higher weight to facts with highly similar relations as compared to facts with less similar relations. In our example in Figure 3, this means that interest is propagated from $\langle \text{Albert_Einstein BORNIN Ulm} \rangle$ to the fact $\langle \text{Britney_Spears ORIGINATESFROM Kentwood_Louisiana} \rangle$ as follows. Each of the entities involved in the two facts belong to the following classes.

<i>Albert_Einstein</i>	∈	Physicists, German_Americans, Pacifists, person
<i>Britney_Spears</i>	∈	Singers, English_Americans, Actors, person
<i>Ulm</i>	∈	Cities_in_Baden_Württemberg, city, municipality, location
<i>Kentwood_Louisiana</i>	∈	Towns_in_Louisiana, town, municipality, location

Then the similarity between the two facts amounts to $(0.25 + 0.5) / 2 = 0.375$. And so, a score of 0.375 is propagated from the first fact with Einstein to the second fact with Spears. Again, a final normalization step allows us to treat interest scores $P_{in}^*(f)$ as the probability of user interest in the fact f .

4. PERSONALIZATION STRATEGY

We cast our personalized ranking approach in a probabilistic model. To allow for a tuning of the strength of the employed personalization at any time, we define our personalized ranking function as a mixture of the original ranking function $P_{NAGA}(g|q)$ of NAGA and the user-biased ranking function $P_{user}(g|q)$ as follows:

$$P_{personalized}(g|q) = \gamma \cdot P_{NAGA}(g|q) + (1 - \gamma) \cdot P_{user}(g|q)$$

Thus by choosing γ appropriately, the influence of the user-specific ranking bias can be regulated.

In the following, we derive our proposal for estimating the user-specific probability that the answer graph g is relevant to the query q . Our derivation of a personalized ranking function follows similar lines as the derivation of the original NAGA ranking, thus transferring the reasoning from the complete knowledge graph to the user-specific graph snippet. We also adopt the notion of a background model weighting the fact templates comprising the query against each other, as well as, informativeness which measures the popularity of facts matching the query. While informativeness based on the complete knowledge graph would, e.g., give high weight to the entity "Britney Spears", informativeness on the personal knowledge graph snippet might deem "Hildegard Knef" more popular, and thus more interesting to the user.

More formally, the user-specific ranking function $P_{user}(g|q)$ can be re-written as $P_{user}(g|q) \approx P_{user}(q|g) \cdot P_{user}(g)$ using Bayes' rule. The denominator $P_{user}(q)$ can be dropped as it is the same for all answer graphs and thus has no influence on the ranking order. With $P_{user}(g)$ assumed to be uniform, we are left with $P_{user}(q|g)$, the probability that the answer graph g generated the query q given the user profile.

Recall that a query is expressed as $q = q_1 q_2 \dots q_k$ where q_i is a fact template. A fact template is of the form $\langle x R y \rangle$ where at least one of x , R or y is unbound. Assuming independence between fact templates, we have

$$P_{user}(q|g) = \prod_{i=1}^n P_{user}(q_i|g)$$

Again we apply Bayes' rule to obtain $P_{user}(q_i|g) \approx P_{user}(g|q_i) \cdot P_{user}(q_i)$. $P_{user}(g|q_i)$ is the probability that the answer graph g matches the fact template q_i . $P_{user}(g|q_i)$ assesses the informativeness of the answer graph given the query template and user profile, while $P_{user}(q_i)$ serves the purpose of a user-specific background model.

4.1 Definition of the user-specific background model

The background model weights the different fact templates in the query (this is similar in spirit to term weighting in standard IR). In order to weight the fact templates, we need to reason on the *bound* parts of each fact template q_i , and weight different query units against each other depending on the user interest. That is

$$P_{user}(q_i) = \begin{cases} P_{in}^*(x) & \text{if only } x \text{ is bound in } q_i \\ P_{in}^*(y) & \text{if only } y \text{ is bound in } q_i \\ P_{in}^*(x) \cdot P_{in}^*(y) & \text{if } x, y \text{ are bound in } q_i \\ \sum_{f'=(s,R,t)} P_{in}^*(f') & \text{if only } R \text{ is bound in } q_i \\ P_{in}^*(x) \cdot \sum_{f'=(s,R,t)} P_{in}^*(f') & \text{if } x, R \text{ are bound in } q_i \\ P_{in}^*(y) \cdot \sum_{f'=(s,R,t)} P_{in}^*(f') & \text{if } R, y \text{ are bound in } q_i \end{cases}$$

Clearly, when only one of the entities is bound, the user interest boils down to her interest in that particular entity. Our score propagation scheme ensures that both direct as well as indirect interest in the entity play a role in the final interest score. However, when both entities are bound, we have choice of either multiplying or adding the user interest in those entities. We chose multiplication based on the intuition that interest in two entities may have been expressed independently of each other.

The last three cases deal with fact templates when the relation is bound. When only the relation is bound, the user interest in that relation is the sum of interest in all facts containing that relation. Note that if the user has previously expressed interest in very few facts containing this relation, our propagation takes care that other facts with the same relation receive only a damped score. As in the previous case, if an entity as well as a relation are both bound, we then assume independence in the user interest in each of them and multiply the interest scores.

4.2 Definition of the user-specific informativeness

Next we estimate the user-specific informativeness $P_{user}(g|q_i)$ which serves a similar purpose as informativeness in the original NAGA ranking. Instead of assessing the overall popularity of facts, it measures popularity with respect to the user-specific snippet of the knowledge graph. Since the bound arguments of a query q , comprising of fact templates, are the same for each answer graph g , the user-specific informativeness $P_{user}(g|q_i)$, estimates the user interest in the *unbound* parts of the query. Since each q_i has a match f , $P_{user}(g|q_i)$ reduces to $P_{user}(f|q_i)$.

The probability of a fact $f = \langle x' R' y' \rangle$ matching $q_i = \langle x R y \rangle$ in the context of the user profile, $P_{user}(f|q_i)$, is defined as the probability of user interest in the unbound parts of the fact f , given that we just learnt from the user formulating the query q that she is interested in the bound parts of f .

We estimate this probability by reasoning on the interest in the unbound parts of f , i.e.,

$$P_{user}(f|q_i) = \begin{cases} P_{in}^*(x') & \text{if } x \text{ unbound in } q_i \\ P_{in}^*(y') & \text{if } y \text{ unbound in } q_i \\ P_{in}^*(x') \cdot P_{in}^*(y') & \text{if } x, y \text{ unbound in } q_i \\ \sum_{f'=(s,R',t)} P_{in}^*(f') & \text{if } R \text{ unbound in } q_i \\ P_{in}^*(x') \cdot \sum_{f'=(s,R',t)} P_{in}^*(f') & \text{if } x, R \text{ unbound in } q_i \\ \sum_{f'=(s,R',t)} P_{in}^*(f') \cdot P_{in}^*(y') & \text{if } R, y \text{ unbound in } q_i \\ P_{in}^*(x') \cdot \sum_{f'=(s,R',t)} P_{in}^*(f') \cdot P_{in}^*(y') & \text{else} \end{cases}$$

Here we assume independence, and thus consider the products of user interests in the unbound parts of f . The general user interest into the relation R is obtained by aggregating the user interest in facts containing a relation R .

Example. Suppose a user accesses the fact $\langle \text{Britney_Spears ISA singer} \rangle$. Some time later, she poses the query $q = \langle \$x \text{ BORNIN Ulm} \rangle \wedge \langle \$x \text{ ISA } \$y \rangle$ i.e., a query consisting of two templates $q_1 = \langle \$x \text{ BORNIN Ulm} \rangle$ and $q_2 = \langle \$x \text{ ISA } \$y \rangle$. Then the user-specific background model, $P_{user}(q_i)$ weights the two templates q_1 and q_2 against each other based on the probability of user interest in the bound query parts, i.e., the user interest in BORNIN Ulm with respect to ISA. For each template the facts matching the template are ranked based on their unbound parts. E.g., let the two candidate answers be $g_1 = \langle \text{Albert.Einstein BORNIN Ulm} \rangle \wedge \langle \text{Albert.Einstein ISA physicist} \rangle$ and $g_2 = \langle \text{Hildegard.Knef BORNIN Ulm} \rangle \wedge \langle \text{Hildegard.Knef ISA singer} \rangle$. Now, g_1 and g_2 are ranked based on $P_{user}(\text{Albert.Einstein}|q_1)$ and $P_{user}(\text{Albert.Einstein, physicist}|q_2)$, respectively $P_{user}(\text{Hildegard.Knef}|q_1)$ and $P_{user}(\text{Hildegard.Knef,singer}|q_2)$. In the original NAGA ranking, we would expect the informativeness of Albert.Einstein and physicist to be higher than the informativeness of Hildegard.Knef and singer due to the popularity of Albert Einstein. In the context of the user profile which documented a user interest in singers, we would, however, expect Hildegard Knef to have the higher user-specific informativeness.

5. EXPERIMENTAL EVALUATION

5.1 Experimental Setup

We use the NAGA semantic search engine to test our score propagation and personalization techniques. At the time of this evaluation, we did not have access to user logs³. Instead, we created our own queries and profiles to specifically test the various components of our techniques.

Profile	Topic	Query used to build profile
1.	OpenGL	What is OpenGL?
2.	Java (the PL)	All facts about Java
3.	Kevin Mitnick	All facts about Kevin Mitnick
4.	Britney Spears	All facts about Britney Spears
5.	Friedrich Nietzsche	All books written by Nietzsche
6.	Yao Ming	All facts about Yao Ming
7.	Pirates of the Caribbean	All actors of the Pirates of the Caribbean movie
8.	Levenshtein distance	All facts about Levenshtein distance
9.	Steven Seagal	Movies in which Seagal acted
10.	Picasso	All of Picasso's creations

Table 1: User profiles generated from topics

Creation of user profiles. In order to create user profiles, we chose topics from the INEX 2007 Adhoc Track⁴. The track makes use of a subset of Wikipedia and around 450 topics are proposed. Each topic is usually based on an entity. We chose 10 of those topics which were also available in NAGA's knowledge base as the basis for generating user profiles. Table 1 lists these topics. For each topic, the query

³Note that the collection of user logs is work in progress.

⁴inex.is.informatik.uni-duisburg.de/2007/

listed in the third column was fired and the top-10 results were taken as interesting to the user. The entities as well as facts corresponding to each of the results were deemed to be clicked by the user. These clicks were then translated into scores and propagated to other parts of the knowledge base. For example, for profile number 5, the query fired was: $\langle \text{Friedrich.Nietzsche WROTE } \$x \rangle$. The top-10 results included entity bindings for $\$x$ (for example, "The birth of tragedy"). Each of these entities were assumed to be clicked. In addition, the fact corresponding to the result ("Friedrich.Nietzsche wrote The.birth.of.tragedy") was also assumed to be clicked.

Profile	Topic	Example Query	Expected Results
Re-finding Queries			
1.	OpenGL	$\$x$ is Graphics_library	OpenGL on top
2.	Java (the PL)	$\$x$ \$r Sun_Microsystems	Java on top
3.	Kevin Mitnick	$\$x$ type Computer_Security_Specialist	Mitnick's name on top
4.	Britney Spears	$\$x$ isa American_dance_musician	Britney Spears on top
Queries testing entity interest propagation			
5.	Friedrich Nietzsche	$\$x$ isa book $\$x$ subClassOf book	Nietzsche's books, books on philosophy The class of Nietzsche's books, the category philosophical books on top
6.	Yao Ming	$\$x$ subClassOf Player	The classes Houston Rockets players, basketball players on top
7.	Pirates of the Caribbean	$\$x$ subClassOf movie	The classes Pirate films, action films on top
8.	Levenshtein distance	$\$x$ subClassOf algorithm	String algorithms on top
Queries testing fact interest propagation			
9.	Steven Seagal	$\$x$ subClassOf movie Woody.Allen \$r \$x	Action thrillers, American movies movies in which Allen was an actor
10.	Picasso	Vincent_van_Gogh \$r \$x	van Gogh's paintings

Table 2: Queries generated to test personalization

Queries. Once each of the above profiles was set up, we formulated queries to test our techniques. Table 2 lists some examples of queries we constructed and the results we expected to see. The queries were designed to specifically test the effects of the individual components of the user-specific informativeness. The first set of queries in Table 2 makes a case for remembering previous accesses to entities. For example, query 2 makes use of the clicks on the entity Java.PL (programming language) when the user searches for interesting facts about Sun Microsystems. The next set of queries we considered was conceived to test our entity score propagation. For example, on profile 5 the goal was to

test the entity score propagations from Nietzsche’s books to other books (e.g., `The_birch_of_tragedy` → category philosophy books → other instances of philosophy books), as well as, from Nietzsche himself to other philosophers. Finally, we constructed queries to witness the effects of our fact score propagation scheme. Consider for example query 10. Since the profile was generated based on Picasso’s creations as an artist, we expect the listing of van Gogh’s paintings to be ranked high, rather than his personal details (such as, birth date, birth place, etc.). Similarly, profile 9 is concerned with the movies in which Seagal acted in and so, for query 9, we would expect a list of movies in which Woody Allen *acted* in, as opposed to his personal details or movies that he directed.

5.2 Results

Rank	NAGA	Personalization
1.	I, Robot	The Antichrist
2.	I, Libertine	The Birth of Tragedy
3.	I, Claudius	On the Genealogy of Morality
4.	Little, Big	The Gay Science
5.	Contact	Ecce Homo
6.	V.	Nietzsche contra Wagner
7.	Night	Twilight of the Idols
8.	Magic, Inc	In a Glass Darkly
9.	Sex	The Book on Adler
10.	Girl, Interrupted	The Autobiography of Charles Darwin

Table 3: Results for the query $\langle \$x \text{ isa book} \rangle$ based on the profile generated by $\langle \text{Friedrich_Nietzsche wrote } \$x \rangle$

On our test queries, our personalization approach coincides with our expectations. In the following we discuss a couple of the more interesting examples testing our propagation scheme in more detail. We first consider the queries on profile 5. Table 3 shows the results and ranking returned by both NAGA as well as our personalization⁵ for the first query $\langle \$x \text{ isa book} \rangle$. As expected, the books by Nietzsche ranked very high – in fact, the first 7 results are books by Nietzsche. In addition, we also have a book written by Kierkegaard, another well-known philosopher, at rank 9 (The Book on Adler) showing that propagation and personalization are boosting the right results to the top. How the books reported at position 8 (In a Glass Darkly by Sheridan Le Fanu) and position 10 (The Autobiography of Charles Darwin by Charles Darwin) connect to Nietzsche is not as obvious since neither of them is a philosophical book nor written by a philosopher. A closer look at the propagation revealed that Nietzsche’s book “The Birth of Tragedy” was written in 1872, the same year that the above two books were written. Since “The Birth of Tragedy” also belongs to the category “1872 books”, the score was propagated to the other two books as well. Thus, our propagation inferred an interest in books written in the times of Nietzsche.

Similarly, our second query on this profile exhibits effects of our entity score propagation. Suppose a user had shown interest in Nietzsche’s work and was now interested in browsing other books in the knowledge base, he could issue the query $\langle \$x \text{ subclassOf book} \rangle$. The results for this query are reported in Table 4. Consider the personalized results.

⁵Note that the result ranking presented here are based only on the personalized ranking. We did not mix the original NAGA score and the personalized score since our goal was to test only the personalization techniques.

Rank	NAGA	Personalization
1.	formulary	Books_by_Nietzsche
2.	curiosa	Philosophy_books
3.	authority	1895_books
4.	booklet	1889_books
5.	tome	1908_books
6.	storybook	1887_books
7.	pop-up_book	1882_books
8.	bestiary	1872_books
9.	catechism	Books_by_Kierkegaard
10.	trade_book	Travel_books

Table 4: Results for the query $\langle \$x \text{ subclassOf book} \rangle$

Clearly, the top 9 results are a direct result of personalization. Results at rank 3 through 8 are the years in which Nietzsche’s works were published. The result at rank 10 was a “random” result – that is, there were no other results which could be preferred since all other results had the same score.

Rank	NAGA	Personalization
1.	type artist	created "The Potato Easters"
2.	type person	created Sunflowers
3.	type Dutch painter	created "The Starry Night"
4.	isCalled *	created Irises
5.	is Called Vincent van Qoq	created "Portrait of Dr. Gachet"
6.	isCalled *	type artist
7.	isCalled *	type person
8.	isCalled *	type Dutch painter
9.	isCalled *	isCalled *
10.	isCalled *	is Called Vincent van Qoq

Table 5: Results for the query “Vincent_van_Gogh \$r \$x”

Next, we discuss an example pointing out the merits of fact score propagation. Suppose a user showed interest in the works of Picasso (profile 10), and is now searching facts about Vincent van Gogh ($\langle \text{Vincent_van_Gogh } \$r \$x \rangle$). Without personalization we find answers such as Vincent van Gogh is an artist on top followed by the synonyms under which he was known in various languages. In contrast, the personalized results present a number of his paintings on top (see Table 5 for the result rankings). This is due to the interest propagation from $\langle \text{Picasso CREATED Chicago_Picasso} \rangle$ to, e.g., $\langle \text{Vincent_van_Gogh CREATED Sunflowers} \rangle$ via the relation `CREATED`.

In summary, we have shown in this section that not only is personalization highly essential (comparing the ranked result list of NAGA with the personalized list), but also that our techniques are the right way to proceed. We tested our techniques with both simple as well as complex queries. We have reported on a subset of the most interesting queries here.

6. RELATED WORK

Our present work on personalizing the search for knowledge finds related work on personalization in various fields:

in Web search, XML search, as well as in databases. However, to the best of our knowledge we are the first to endeavor to personalize search in a knowledge base. This bears commonalities with personalization in databases as both lack elaborate textual information. At the same time, the knowledge base we consider is schema-free as opposed to databases and XML where a successful personalization system always needs to be adapted to the schema at hand.

In [8], Koutrika et al. present a personalization framework for database users that relies on user profiles which comprise user preferences with respect to atomic query elements such as individual join conditions or selections. At query-time, the preferences most relevant to the query and user are selected from the profile, and integrated to form a new modified database query. This query-rewriting process is inherently hard as, e.g., conflicting constraints need to be handled. Koutrika et al. provide means for handling syntactical conflicts, however, semantical conflicts are schema-dependent and thus not generally addressable.

In [2], the XML personalization system, PIMENTO is presented. In their approach, a user profile is a set of rules of the form (*condition, action, conclusion*). The condition and conclusion parts are XQuery full text, and the action can be *add, remove* or *replace*. Whenever a query matches a rule condition, it is re-written accordingly. However, the generation of the rules in the user profile requires the user's active participation. Pan describes an interesting approach in [10] where he proposes query expansion for XML search based on ontological similarities. After initial feedback from the user a query specific ontology is constructed from parts of the global ontology and the query itself, and then used for expansion. In [11, 12] Schenkel and Theobald present a way for structural query expansion based on relevance feedback for XML. [5] is an approach along similar lines based on pseudo feedback and only a portion of the document. XML structure is also considered by Wang et al. in [16]. They propose an extension of Preference XPath for the purposes of personalized MPEG-7 digital libraries.

Jiang et al. [6] introduce a personalization approach for semantic search. Their methodology is similar to ours in that they employ a domain ontology, and reason on user interest in concepts and relations. They consider long-term interest scores based on previous accesses, and perform spreading activation on the concepts present in the initial query result set. By combining scores from the spreading activation with the long-term interest weights, results "close" to the result set that have been accessed before will be ranked highest. By performing spreading activation on the result set instead of the user profile, this approach, however, does not allow to infer interest in concepts related to the ones accessed by the user. Similarly, Sieg et al. [13] present an approach to personalized search that involves building models of user context as ontological profiles by assigning implicitly derived interest scores to existing concepts in a domain ontology. Still both approaches do not tackle the problem of personalization the search on a knowledge graph but consider personalizing document search.

Another differing, still related area is ontological user profiling in recommender systems. For example, Middleton et al. [9] present two experimental systems, Quickstep and Fox-trott, for recommending academic research papers.

7. CONCLUSIONS AND FUTURE WORK

In this paper we presented a personalization approach tailored to the specifics of searching a knowledge base. We developed techniques to infer user interests to create a user profile and then showed how user interest could be incorporated in our model for personalization. Our preliminary experiments examined both entity and fact score propagation and showed promising results.

In ongoing experiments, we are studying how mixing user interest with the original NAGA score affects results. We are also studying how the user-specific background model affects results.

8. ADDITIONAL AUTHORS

9. REFERENCES

- [1] S. Amer-Yahia, I. Fundulaki, P. Jain, and L. Lakshmanan. Personalizing xml text search in piment. In *Proc. of VLDB*, 2005.
- [2] S. Amer-Yahia, I. Fundulaki, and L. Lakshmanan. Personalizing xml search in pimento. In *Proc. of ICDE*, 2007.
- [3] M. Cafarella, C. Re, D. Suciu, and O. Etzioni. Structured querying of web text data: A technical challenge. In *Proc. of CIDR*, 2007.
- [4] D. Hiemstra. A probabilistic justification for using tf.idf term weighting in information retrieval. *Inf. Proc. and Management*, 3(2), 2006.
- [5] W. Hsu, M. L. Lee, and X. Wu. Path-augmented keyword search for xml documents. In *ICTAI*, 2004.
- [6] X. Jiang and A.-H. Tan. Learning and inferencing in user ontology for personalized semantic web services. In *Proc. of WWW*, 2006.
- [7] G. Kasneci, F. M. Suchanek, G. Ifrim, M. Ramanath, and G. Weikum. Naga: Searching and ranking knowledge. In *Proc. of ICDE*, 2008.
- [8] G. Koutrika and Y. Ioannidis. Personalization of queries in database systems. In *Proc. of ICDE*, 2004.
- [9] S. E. Middleton, N. R. Shadbolt, and D. C. D. Roure. Ontological user profiling in recommender systems. *ACM Trans. Inf. Syst.*, 22(1):54-88, 2004.
- [10] H. Pan. Relevance feedback in xml retrieval. In *EDBT Workshops*, 2004.
- [11] R. Schenkel and M. Theobald. Feedback-driven structural query expansion for ranked retrieval of xml data. In *Proc. of EDBT*, 2006.
- [12] R. Schenkel and M. Theobald. Structural feedback for keyword-based xml retrieval. In *Proc. of ECIR*, 2006.
- [13] A. Sieg, B. Mobasher, and R. Burke. Web search personalization with ontological user profiles. In *Proc. of CIKM*, 2007.
- [14] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A Core of Semantic Knowledge. In *Proc. of WWW*, 2007.
- [15] J. Teevan, S. T. Dumais, and E. Horvitz. Personalizing search via automated analysis of interests and activities. In *Proc. of SIGIR*, 2005.
- [16] Q. Wang, W.-T. Balke, W. Kiessling, and A. Huhn. P-news: Deeply personalized news dissemination for mpeg-7 based digital libraries. In *Proc. of ECDL*, 2004.
- [17] C. Zhai and J. Lafferty. A risk minimization framework for information retrieval. *Inf. Proc. and Management*, 42(1), 2006.