SnoopyDB: Narrowing the Gap between Structured and Unstructured Information using Recommendations

Wolfgang Gassler, Eva Zangerle, Michael Tschuggnall, Günther Specht
Databases and Information Systems, Institute of Computer Science
University of Innsbruck, Austria
{firstname.lastname}@uibk.ac.at

ABSTRACT

Knowledge is structured – until it is stored to a wiki-like information system. In this paper we present the multi-user system *SnoopyDB*, which preserves the structure of knowledge without restricting the type or schema of inserted information. A self-learning schema system and recommendation engine support the user during the process of inserting information. These dynamically calculated recommendations develop an implicit schema, which is used by the majority of stored information. Further recommendation measures enhance the content both semantically and syntactically and motivate the user to insert more information than he intended to.

Categories and Subject Descriptors

H.3.5 [**Storage and Retrieval**]: Online Information Services - Web-based services; H.4.m [**Information Systems**]: Miscellaneous

General Terms

Algorithms, Design, Experimentation, Human Factors

Keywords

Semistructured Data, Recommendations, Human Interaction, Ranking, Semantic Web, RDF

1. INTRODUCTION

Wikis are a major paradigm for storing information online in an easy, efficient and collaborative way. However, information is stored as plain text and therefore lacks any structure. Thus, complex queries – like "Which Austrian cities have more than 10.000 inhabitants and have a female mayor?" – are not feasible, as fulltext search is not powerful enough. Weikum *et al.* [3] pointed out that there is a need for rich information repositories, which support both structured and unstructured information to fully exploit the advantages of both worlds. Semistructured information systems provide such advantages, but do not place any restriction on the schema of knowledge. Therefore, multi-user systems naturally lead to a proliferation of schemas and substructures. According to Boulain *et al.* [1], Wikipedia has to cope with the same problem as only 35% of all edits in Wikipedia are changes of content. All other edits are related to structure (e.g avoiding proliferation) and do not concern the content itself. Even in the case of template-based schemas of infoboxes at Wikipedia, which are supervised and enforced by the community, Wu and Weld [4] showed that schemas are divergent, noisy and contain many untyped attributes.

We present SnoopyDB, a novel information system, which is able to cope with these problems by facilitating structure and developing a common schema by providing recommendations.

HT'10, June 13–16, 2010, Toronto, Ontario, Canada.
ACM 978-1-4503-0041-4/10/06.

2. SNOOPYDB

SnoopyDB stores information as collections, which are similar to wiki pages and consist of an arbitrary number of key-value pairs. Therefore, all information can be represented as RDF-triples consisting of collection, key and value, which are very well suited for the description of any (real-world) resource [2]. As already pointed out, such a semi-structured storage method would lead to the proliferation of substructures. SnoopyDB copes with this problem by providing intelligent recommendations, which is described in the following section.

2.1 Schema Alignment by Recommendations

SnoopyDB aims at creating a homogenous structure within the information system. Hence the user is guided by recommendations during the insertion process in order to align the entered information to a commonly used schema. The self-adapting schema is implicit and dynamically calculated based on all already stored information within the system.

In contrast to common wiki systems, the process of inserting information into SnoopyDB does not only consist of one rigid step. SnoopyDB is based on a dynamic, cyclic and guided process, as shown in Figure 1: At first the user starts to enter a key *(1)*. Subsequently SnoopyDB analyses the entered key and suggests possible enhancements *(2)*: While the user types, SnoopyDB computes context-sensitive recommendations, which suggests keys containing the newly entered string and are already present in the information system. To rank the potentially large set of suggested keys, the context is taken into account. Consider the example of a user who has just entered "continent: Europe" and is defining the next key "number". In this case, SnoopyDB ranks the recommendation of the key "numberOfInhabitants" higher than the key "numberOfChildren", as "numberOfInhabitants" and the key "continent" occur together in many already stored collections.

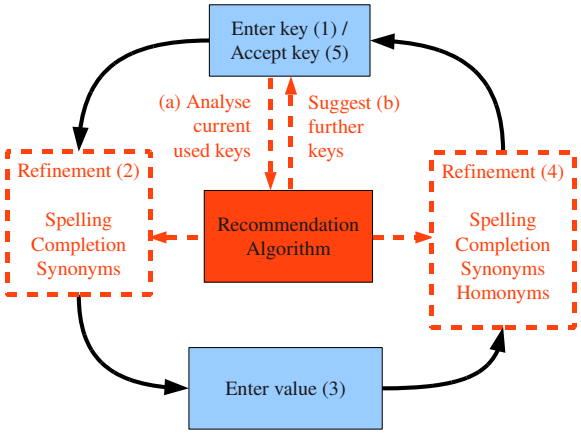

Figure 1: Guided insertion process cycle.

These recommendations also include spelling issues and complex enhancements in order to avoid the (extensive) use of synonyms – e.g. the commonly used key "numberOfInhabitants" is suggested to be used instead of the user's key "population". In step *(3)*, the user enters a value corresponding to the key and is supported by suggestions as well *(4)*. Furthermore, SnoopyDB "snoops" as much information as possible from the user, e.g. about the semantics of the newly inserted value or the datatype of the entered information. For instance, in the case of homonyms SnoopyDB provides links to already present, applicable collections which allows the user to clarify the semantics of the entered key. For example, the specification of the key-value pair "twinCity: Freiburg" leads to recommendations of links to "Freiburg (Germany)" or "Freiburg (Switzerland)" and therefore a semantic enhancement.

Based on the newly entered keys *(a)*, SnoopyDB computes possible further keys, which are contextually connected to the current ones entered by the user *(b)*. These keys, which are thematically compatible with the type of information the user wants to insert, are recommended and the user is encouraged to enter more information than he intended to (additional information is "snooped"). By accepting some of these additional recommended keys *(5)*, the user starts entering information at step *(1)* again. The recommendations are further refined with each cycle as there is more information available for the calculation of recommendations.

The recommendations are based on a data mining process which determines the common schema by extracting association rules and appliying a context-sensitive filter on the resulting suggestions. This self-learning recommendation algorithm maintains a commonly used, self-adapting schema, avoids synonyms, enhances the semantics of homonyms and facilitates the usage of a common vocabulary in the information system.

3. PRELIMINARY RESULTS

Experiments were conducted by human test users as declining or accepting recommendations cannot be simulated artificially. Therefore, two different prototype implementations were used: (I) SnoopyDB as described above and (II) a reduced SnoopyDB version without any recommendations, suggestions or any further guidance mechanisms. Fifteen test users chose arbitrary cities or musicians and entered relevant information firstly in system II and subsequently in system I. The results convincingly showed that SnoopyDB's guidance mechanisms are able to create a common schema (see Figure 2). In the SnoopyDB-system we observed a 33% smaller set of distinct property names used within the system (154 vs. 229 different property names on 50 collections) while at the same time, users indeed entered more information in the guided interface as can be seen from the number of entered properties (840 properties in I vs. 678 properties in II lead to additional 24% of data).

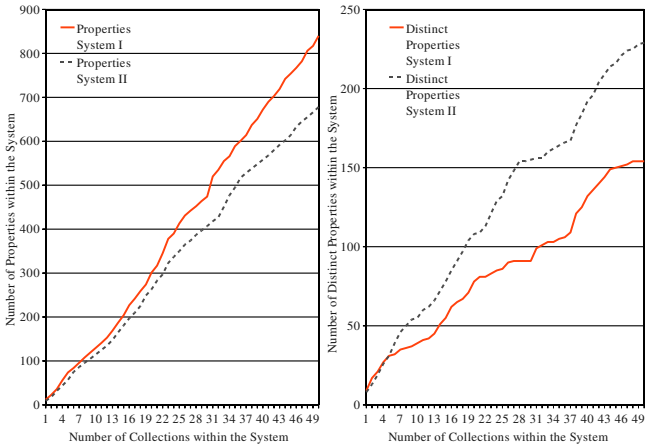

Figure 2: Total properties vs. distinct properties.

4. CONCLUSION

We presented SnoopyDB, a flexible information system for semistructured data which aims at aligning stored data to a common structure by providing intelligent recommendations during the insertion process. These recommendations enable the system to "snoop" as much information as possible, since most inserting users have extensive knowledge about the inserted collection. Without these recommendations, the additional knowledge – in our experiments 24% more data – would be lost. We showed that SnoopyDB significantly contributes to a common, implicit schema by reducing the proliferation of keys by 33%.

5. REFERENCES

[1] P. Boulain, N. Shadbolt, and N. Gibbins. Hyperstructure maintenance costs in large-scale wikis. In *Social Web and Knowledge Management 2008*, February 2008.

[2] B. McBride. The Resource Description Framework (RDF) and its Vocabulary Description Language RDFS. *Handbook on Ontologies*, pages 51–66, 2004.

[3] G. Weikum, G. Kasneci, M. Ramanath, and F. Suchanek. Database and information-retrieval methods for knowledge discovery. *Commun. ACM*, 52(4):56–64, 2009.

[4] F. Wu and D.S. Weld. Automatically refining the wikipedia infobox ontology. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 635–644, New York, NY, USA, 2008. ACM.