User Preference and Optimization of Relational Queries

Nedbal, Radim
2008

User Preference and Optimization of Relational Queries

Post-Graduate Student:
RADIM NEDBAL
Institute of Computer Science of the ASCR, v. v. i.
Pod Vodárenskou věží 2
182 07 Prague, Czech Republic ,

Department of Mathematics
Faculty of Nuclear Science and Physical Engineering
Czech Technical University
Trojanova 13

120 00 Prague, Czech Republic

radned@seznam.cz

Supervisor:
ING. JÚLIUS ŠTULLER, CSC.

Institute of Computer Science of the ASCR, v. v. i.
Pod Vodárenskou věží 2

182 07 Prague, Czech Republic

stuller@cs.cas.cz

Field of Study:
Mathematical Engineering

This work was supported by the project 1ET100300419 of the Program Information Society (of the Thematic Program II of the National Research Program of the Czech Republic) “Intelligent Models, Algorithms, Methods and Tools for the Semantic Web Realization”, and by the Institutional Research Plan AV0Z10300504 “Computer Science for the Information Society: Models, Algorithms, Applications”.

Abstract

The notion of preference poses a new prospect of personalization of database queries. In addition, it can be exploited to optimize query execution. Indeed, a novel optimization technique involving preference is developed, and its algorithm presented.

1. Introduction

Preference provides a modular and declarative means for relaxing and optimizing database queries. It is a concept that needs a special framework for embedding in the relational data model: on the one hand, the framework should be rich enough to capture **various kinds of preference** to provide database users with an expressive language to formulate their wishes, and, on the other hand, robust enough to allow for possibly **conflicting preferences** as the assumption of consistency of complex preferences is hard to fulfill in practical applications.

To reach the above goal we consider sixteen kinds of preferences, some of them allowing for expressing uncertainty. Also, basic preference combiners (Pareto or lexicographic composition) are taken into account.

To embed the notion of preference into relational query languages, a **preference operator**, parameterized by user preference, is defined: it filters out not all the bad results, but only worse results than the best matching alternatives and returns the perfect match if present in the database, otherwise, it delivers best-matching alternatives, but nothing worse!

Optimization strategy of **pushing the preference specification** down the query execution tree is governed by both algebraic properties of the preference operator and logical properties of user preference that always is expressed over a set of possible states of the world. This strategy is based on the assumption that early application of the preference operator reduces intermediate results and thus minimizes the data flow during the query execution.

2. Embedding Preference in Relational Query Languages

2.1. Preference Operator

A new, preference operator is added to the relational algebra. Its expressive power depends on the expressivity of the language for expressing user preference – its single parameter.

Definition 1 (Preference operator) *Let U denote a universe and $W^{\mathcal{P}} \subseteq W$ a set of the most preferred worlds with regard to a preference specification $\mathcal{P}$ over a set W of possible worlds. The preference operator $\omega_{\mathcal{P}}$ is a mapping $\omega_{\mathcal{P}} \colon V \to 2^V$ from a set V of discourse into the powerset 2^V of V:*

$$\omega_{\mathcal{P}}(v) = \{v' \subseteq v | \exists u \in U \exists w \in W^{\mathcal{P}} \colon u \models w \wedge v'\} \ . \quad (1)$$

It is important to point out that the preference specification parameter $\mathcal{P}$ allows for complex preference compound from elementary preferences of various kinds. We take into account *locally optimistic*, *locally pessimistic*, *opportunistic*, and *careful* preferences, whose terminology and motivation has been introduced in [1]. Moreover, we consider another two binary choices: a preference can be strict or non-strict and can be evaluated without or with a ceteris paribus proviso, a concept introduced by von Wright [2]. Altogether, we get sixteen various kinds of preference.

On the one hand, this complex preference specification parameter yields a large expressivity, however, on the other hand, it makes the preference operator absent from algebraic properties fundamental for realizing the algebraic optimization strategy that is based on early application of the most selective operators of relational algebra. Thus a more general technique has to be developed.

2.2. Optimization

Algebraic optimization strategy involving the preference operator must provide a transformation (of a given database query) under which the preference operator, which is the last operator to be applied, is invariant.

Example 1 *Let $\mathcal{R}$ be a database schema and $\mathcal{I}$ its instance consisting of two relation instances I_1, I_2. Suppose a user expresses their requirements through a database query*

$$q(\mathcal{I}) = \pi_X(I_1 \cup I_2) \tag{2}$$

over $\mathcal{I}$ and their preferences (wishes) through a preference specification $\mathcal{P}$ over the set

$$W = 2^{q(\mathcal{I})} \tag{3}$$

of possible worlds. Then, the preference operator $\omega_{\mathcal{P}}\big(q(\mathcal{I})\big)$ evaluated over (2) *returns the best matching alternatives with regard to the user preferences.*

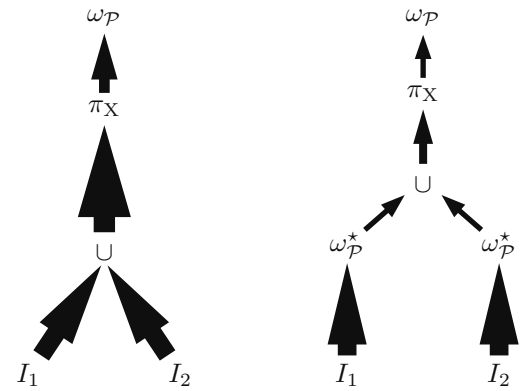
Suppose the preference operator is invariant under the following transformation of $q(\mathcal{I})$ to $q'(\mathcal{I})$:

$$q'(\mathcal{I}) = \pi_X\big(\omega^\star_{\mathcal{P}}(I_1) \cup \omega^\star_{\mathcal{P}}(I_2)\big) \ , \tag{4}$$

where $\omega^\star_{\mathcal{P}}(I_i)$ is a preference operator derivative filtering out "bad" tuples. Then, the preference operator $\omega_{\mathcal{P}}\big(q'(\mathcal{I})\big)$ evaluated over (4) *returns the best matching alternatives with regard to the user preferences:*

$$\omega_{\mathcal{P}}\big(q'(\mathcal{I})\big) = \omega_{\mathcal{P}}\big(q(\mathcal{I})\big) \ .$$

The query execution trees are depicted in Fig. 1, where data flow between the computer's main memory and secondary storage is represented by the drawing width.



(a) Before pushing *(b) After pushing*

Figure 1: *Improving the query plan by pushing the preference operator down the query execution tree*

Supposing that relation instances I_1 and I_2 are too big to fit into the main memory and using the number of the secondary storage I/O's as our measure of cost for an operation, it can be seen that the strategy of pushing the preference operator can improve the performance significantly.

Note that to push the preference specification $\mathcal{P}$ down the expression tree, a special derivative $\omega^\star_{\mathcal{P}}$ of the preference operator $\omega_{\mathcal{P}}$ realizing its filtering potential has been introduced. Unlike the preference operator (cf. 1), it is a mapping $\omega^\star_{\mathcal{P}} \colon V \to V$ from a set V of discourse – a set of all possible tuples over a given relation scheme – into itself. Most importantly, it fulfills the following property:

$$\omega_{\mathcal{P}}\big(\omega^\star_{\mathcal{P}}(I)\big) = \omega_{\mathcal{P}}(I) \ ,$$

i.e., it filters out bad tuples of a given relational instance I without affecting the value of the preference operator.

Furthermore, observe that $\omega_{\mathcal{P}}$ and $\omega^\star_{\mathcal{P}}$ have an identical value of the preference parameter. This value – a user preference $\mathcal{P}$ over W – however, is usually expressed over the result of a query (3). **Does it mean that we need to have computed** (3)**, and thus also** (2)**, before we are able to evaluate** (4)**?** The answer has to be searched for in the definition of the semantics of preference specification [3] and is provided by the following proposition 1.

In brief, a preference specification has the constructive semantics defined by means of a disjunctive logic program (DLP). In the following, $\mathfrak{W}$ stands for the

Herbrand universe for the DLP assigned to a preference specification $\mathcal{P}$, and $g_{\mathcal{P}}$ for a mapping that can be computed from models of the DLP. Note that models of the DLP can be computed using single exponential time on the cardinality of $\mathfrak{W}$, which , in turn, depends exponentially on the number of elementary preferences composing the preference specification $\mathcal{P}$. This number, however, is supposed to be small, usually between five and ten. Finally, $f_{\mathcal{P}}$ stands for a mapping that can be expressed as a first order query.

Lemma 1 *Let q denote a database query – a mapping $q\colon \mathrm{inst}(\mathcal{R}) \to \mathrm{inst}(S)$ from a set of database instances over a database schema $\mathcal{R}$ to a set of relation instances over a relation schema S. Given a preference specification $\mathcal{P}$ over a set W of possible worlds, there exist a finite set $\mathfrak{W}$ and a mapping $g_{\mathcal{P}}\colon 2^{\mathfrak{W}} \to 2^{\mathfrak{W}}$ such that the following properties hold for all subsets $\mathfrak{W}'$ of $\mathfrak{W}$ if $W = 2^{q(\mathcal{I})}$:*

$$g_{\mathcal{P}}(\mathfrak{W}') \subseteq \mathfrak{W}' \ , \tag{5}$$

$$g_{\mathcal{P}}(\mathfrak{W}') \subseteq f_{\mathcal{P}}^{-1}(\mathrm{supp}) \subseteq \mathfrak{W}' \Rightarrow W^{\mathcal{P}} = \langle {\mathfrak{W}'}^{\mathcal{P}} \rangle_W \ , \tag{6}$$

where $f_{\mathcal{P}}\colon \mathfrak{W} \to \{\mathrm{unsupp}, \mathrm{supp}\}$ is a function returning supp *for every $\mathfrak{w} \in \mathfrak{W}$ that, loosely speaking, is "supported" by $\mathcal{P}$ over W, and*

$$\langle {\mathfrak{W}'}^{\mathcal{P}} \rangle_W = \{w \in W |\ \exists \mathfrak{w} \in {\mathfrak{W}'}^{\mathcal{P}} \colon w \Rightarrow \mathfrak{w}\} \ . \tag{7}$$

Proposition 1 *Suppose $\mathcal{I}$, $\mathfrak{W}$, $f_{\mathcal{P}}$, and $g_{\mathcal{P}}$ are as in Lemma 1. Then, the mapping $h_{\mathcal{P}}\colon 2^{\mathfrak{W}} \to 2^{\mathfrak{W}}$:*

$$h_{\mathcal{P}}(\mathfrak{W}') = \big(\mathfrak{W}' - g_{\mathcal{P}}(\mathfrak{W}')\big) \cup \big(g_{\mathcal{P}}(\mathfrak{W}') \cap f_{\mathcal{P}}^{-1}(\mathrm{supp})\big) \tag{8}$$

has a fixpoint $\mathfrak{W}_{\mathrm{fix}}$ such that $\mathfrak{W}_{\mathrm{fix}} \supseteq f_{\mathcal{P}}^{-1}(\mathrm{supp})$.

Proof: It follows readily from (5) that $\forall \mathfrak{W}' \subseteq \mathfrak{W}\colon h_{\mathcal{P}}(\mathfrak{W}') \subseteq \mathfrak{W}'$. As $\mathfrak{W}$ is finite, it is clear that $\forall \mathfrak{W}' \subseteq \mathfrak{W}\ \exists n \in \mathbb{N}\ [i \geq n \Rightarrow h^i_{\mathcal{P}}(\mathfrak{W}') = h^n_{\mathcal{P}}(\mathfrak{W}')]$, i.e., $h^n_{\mathcal{P}}(\mathfrak{W}')$ is a fixpoint of $h_{\mathcal{P}}$. Now the observation: $\forall \mathfrak{W}' \subseteq \mathfrak{W}\ [\mathfrak{W}' \supseteq f_{\mathcal{P}}^{-1}(\mathrm{supp}) \Rightarrow h_{\mathcal{P}}(\mathfrak{W}') \supseteq f_{\mathcal{P}}^{-1}(\mathrm{supp})]$ completes the proof. ∎

The following corollary follows readily from (6) and from the observation

$$h_{\mathcal{P}}(\mathfrak{W}') = \mathfrak{W}' \iff g_{\mathcal{P}}(\mathfrak{W}') \subseteq f_{\mathcal{P}}^{-1}(\mathrm{supp}) \ .$$

Corollary 1 *Suppose q and $\mathcal{P}$ over W are as in Lemma 1. Then, $\mathfrak{W}_{\mathrm{fix}}$ being the fixpoint from Proposition 1 and $W = 2^{q(\mathcal{I})}$, the following equality holds:*

$$W^{\mathcal{P}} = \langle \mathfrak{W}^{\mathcal{P}}_{\mathrm{fix}} \rangle_W \ .$$

So the answer is: partially. To evaluate the preference operator derivative $\omega^{\star}_{\mathcal{P}}$, it suffices to find a relevant part of the query result. Intuitively speaking, this relevant part is subsumed by the fixpoint of (8) (Corollary 1) and computed by stepwise pruning the special set $\mathfrak{W}$ (Proposition 1).

3. An Algorithm

The above corollary is the key to effective computation of (4) in the above example:

Algorithm 5 Preference operator filtering tuples

Input: $q\colon \mathrm{inst}(\mathcal{R}) \to \mathrm{inst}(S), \mathcal{P}, \mathcal{I}$
Output: $\omega^{\star}_{\mathcal{P}}(I_1)$

```
1:  W_fix := W
2:  while change do
3:      compute g_P(W_fix)
4:      if ∃w ∈ g_P(W_fix): f_P(w) = unsupp then
5:          remove such w from W_fix
6:      end if
7:  end while
8:  compute W^P_fix
9:  ω*_P(I_1) := I_1
10: for all t ∈ I_1 do
11:     if ∀w ∈ W^P_fix: w ⇒ ¬t then
12:         remove t from ω*_P(I_1)
13:     end if
14: end for
```

On line 1, $\mathfrak{W}$ depends solely on preference specification $\mathcal{P}$. It is independent of the set W over which $\mathcal{P}$ is expressed, and thus it also is independent of the input database instance $\mathcal{I}$. The while block computes a fixpoint of (8): the function $g_{\mathcal{P}}$ can be computed in exponential time on input $\mathfrak{W}$, and the function $f_{\mathcal{P}}$ can be expressed as a first order query over $\mathcal{I}$. On line 8, $\mathfrak{W}^{\mathcal{P}}_{\mathrm{fix}}$ can be computed in exponential time on input $\mathfrak{W}$. In the for block, the input relation instance I_1 is filtered: on line 11, the logical condition follows from Corollary 1 and analysis of (1) and (7).

4. Related Work

The study of preference in the context of database queries has been originated by Lacroix and Lavency [4]. They, however, don't deal with algebraic optimization. Following their work, *preference datalog* was introduced in [5], where it was shown that concept of preference provides a modular and declarative means for formulating optimization and relaxation queries in deductive databases.

Nevertheless, only at the turn of the millennium this area attracted broader interest again. Kießling et al. [6, 7, 8, 9, 10, 11] and Chomicki et al. [12, 13, 14, 15] pursued independently a similar, *qualitative* approach within which preference between tuples is specified directly, using binary *preference relations*. They have laid the foundation for preference query optimization that extends established query optimization techniques: preference queries can be evaluated by extended – preference relational algebra. While some transformation laws for queries with preferences were presented in [11, 6], the results presented in [12] are mostly more general.

In brief, Chomicki et al. and Kießling et al. have embedded the concept of preference into relational query languages identically: they have defined an operator parameterized by user preference and returning only the best preference matches. This embedding is similar to ours. However, their operator differs from our preference operator by the parameter: Chomicki et al. and Kießling et al. consider such preference that the operator is partially antimonotonic with respect to its relational argument. By contrast, the preference parameter we consider is more complex and consequently, this property is not fulfilled by the preference operator. As a result, most algebraic properties presented by the above authors don't apply to the preference operator. Specifically, the commutativity and distributivity properties do not hold, and thus the optimization strategy presented in this paper has to rely on different techniques.

Moreover, Chomicki et al. and Kießling et al. are concerned only with one type of preference and don't consider preferences between sets of elements. In terms of logic of preference, they only take into account preferences between singleton worlds[1]. In this sense, their approach is subsumed by the approach presented in this paper, and, in particular, the introduced optimization technique can be applied to the their preference relational algebra.

A special case of the same embedding represents *skyline operator* introduced by Börzsönyi et al. [16]. Some examples of possible rewritings for skyline queries are given but no general rewriting rules are formulated.

[3] is preliminary contribution building on recent advances in logic of preference. Employing non-monotonic reasoning mechanisms, it takes into account various kinds of preferences. The embedding of preference in relational query languages is based on a single preference operator parameterized by a user preference. By contrast to the presented approach, it is assumed that user preference always is expressed over a fixed "universal" domain – a powerset of a universal relation[2]. Consequently, the preference operator has "nice" algebraic properties including conditional commutativity and distributivity. As a result, an optimization strategy of pushing the preference operator down the query expression tree could been developed [17].

A slightly different goal is pursued in [18], where the relational data model is extended to incorporate partial orderings into data domains. The *partially ordered relational algebra* (PORA) is defined by allowing the ordering predicate to be used in formulae of the selection operator. PORA provides users with the capability of capturing the semantics of ordered data. A similar approach to preference modelling in the context of web repositories is presented in [19]: a special algebra is developed for expressing complex *web queries*. The queries employ application-specific ranking and ordering relationships over pages and links to filter out and retrieve only the "best" query results. In addition, cost-based optimization is addressed. Also in [20], actual values of an arbitrary attribute are allowed to be partially ordered according to user preference. Accordingly, relational algebra operations, aggregation functions and arithmetic are redefined. However, some of their properties are lost, and the query optimization issues are not discussed.

A comprehensive work on partial order in databases, presenting the partially ordered sets as the basic construct for modelling data and proposing the embedding of the notion of partial order in relational data model by means of realizer, is [21]. Aiming at an effective representation of information representable by a partial order and proposing a suitable data structure, [22] builds on this framework. Other contributions aim at exploiting linear order inherent in many kinds of data, e.g., time series: in the context of statistical applications systems SEQUIN [23], SRQL [24], Aquery [25, 26]. Various kinds of ordering on power-domains have also been considered in context of modelling incomplete information: a very extensive and general study is provided in [27].

By contrast to the above qualitative approach, in the *quantitative* approach [28, 29, 30, 31, 32, 33, 34], preference is specified indirectly using *scoring functions* that associate a numeric score with every tuple. On the one hand, this approach enables expressing quantitative

[1] A singleton world is a world containing a single element.

[2] Here, the term *universal relation* denotes that unique relation instance over a relation schema that contains all possible tuples over that schema

aspects of preference, e.g., its strength, however, on the other hand, expressivity of the qualitative aspect of preference is restricted to the weak order – a special case of the partial order.

5. Conclusions

The paper deals with the optimization of relational queries using the concept of preference. It builds on the recent leading ideas that have contributed to remarkable advances in the field:

- Preferences are embedded into relational query languages by means of a single preference operator returning only the best tuples in the sense of user preferences. By considering the preference operator on its own, we can, on the one hand, focus on the abstract properties of user preference and, on the other hand, study special evaluation and optimization techniques for the preference operator itself.
- An optimization strategy is based on the assumption that early application of a selective operator reduces intermediate results and thus reduces data flow during the query execution. Pushing the preference operator, based on its algebraic properties, is a well known technique realizing this strategy.

Furthermore, to express a user preference, we employ the language introduced by Kaci and van der Tore [35], who have extended propositional language with sixteen kinds of preference. In their non-monotonic logic framework, we can capture complex preference, including preference between sets, yet the preference operator parameterized by such complex preference doesn't fulfil the commutativity and distributivity properties. For this reason, the optimization strategy needs to employ different technique: computing preference models over a stepwise pruned special set $\mathfrak{W}$ until the fixpoint is reached and then using a special preference operator derivative to filter out "bad" tuples.

In conclusion, the main contribution of the paper consists in presenting the optimization strategy of pushing the user preference down the expression tree and introducing the algorithm for its implementation.

References

[1] S. Kaci and L. W. N. van der Torre, "Algorithms for a nonmonotonic logic of preferences.," in *ECSQARU* (L. Godo, ed.), vol. 3571 of *Lecture Notes in Computer Science*, pp. 281–292, Springer, 2005.

[2] G. von Wright, *The logic of preference*. Edinburgh University Press, Edinburgh, 1963.

[3] R. Nedbal, "Non-monotonic reasoning with various kinds of preferences in the relational data model framework," in *ITAT 2007, Information Technologies – Applications and Theory* (P. Vojtáš, ed.), pp. 15–21, PONT, September 2007.

[4] M. Lacroix and P. Lavency, "Preferences; Putting More Knowledge into Queries.," in *VLDB* (P. M. Stocker, W. Kent, and P. Hammersley, eds.), pp. 217–225, Morgan Kaufmann, 1987.

[5] K. Govindarajan, B. Jayaraman, and S. Mantha, "Preference datalog," Tech. Rep. 95-50, 1, 1995.

[6] B. Hafenrichter and W. Kießling, "Optimization of relational preference queries," in *CRPIT '39: Proceedings of the sixteenth Australasian conference on Database technologies*, (Darlinghurst, Australia), pp. 175–184, Australian Computer Society, Inc., 2005.

[7] W. Kießling, "Foundations of Preferences in Database Systems," in *Proceedings of the 28th VLDB Conference*, (Hong Kong, China), pp. 311–322, 2002.

[8] W. Kießling, "Preference constructors for deeply personalized database queries," Tech. Rep. 2004-07, Institute of Computer Science, University of Augsburg, March 2004.

[9] W. Kießling, "Optimization of Relational Preference Queries," in *Conferences in Research and Practice in Information Technology* (H. Williams and G. Dobbie, eds.), vol. 39, (University of Newcastle, Newcastle, Australia), Australian Computer Society, 2005.

[10] W. Kießling, "Preference Queries with SV-Semantics.," in *COMAD* (J. Haritsa and T. Vijayaraman, eds.), pp. 15–26, Computer Society of India, 2005.

[11] W. Kießling and B. Hafenrichter, "Algebraic optimization of relational preference queries," Tech. Rep. 2003-01, Institute of Computer Science, University of Augsburg, February 2003.

[12] J. Chomicki, "Preference Formulas in Relational Queries," *ACM Trans. Database Syst.*, vol. 28, no. 4, pp. 427–466, 2003.

[13] J. Chomicki, "Semantic optimization of preference queries.," in *CDB* (B. Kuijpers and P. Z. Revesz, eds.), vol. 3074 of *Lecture Notes in Computer Science*, pp. 133–148, Springer, 2004.

[14] J. Chomicki and J. Song, "Monotonic and nonmonotonic preference revision," 2005.

[15] J. Chomicki, S. Staworko, and J. Marcinkowsk, "Preference-driven querying of inconsistent relational databases," in *Proc. International Workshop on Inconsistency and Incompleteness in Databases*, (Munich, Germany), March 2006.

[16] S. Börzsönyi, D. Kossmann, and K. Stocker, "The skyline operator," in *Proceedings of the 17th International Conference on Data Engineering*, (Washington, DC, USA), pp. 421–430, IEEE Computer Society, 2001.

[17] R. Nedbal, "Algebraic optimization of relational queries with various kinds of preferences," in *SOFSEM* (V. Geffert, J. Karhumäki, A. Bertoni, B. Preneel, P. Návrat, and M. Bieliková, eds.), vol. 4910 of *Lecture Notes in Computer Science*, pp. 388–399, Springer, 2008.

[18] W. Ng, "An Extension of the Relational Data Model to Incorporate Ordered Domains," *ACM Transactions on Database Systems*, vol. 26, pp. 344–383, September 2001.

[19] S. Raghavan and H. Garcia-Molina, "Complex queries over web repositories," tech. rep., Stanford University, February 2003.

[20] R. Nedbal, "Relational Databases with Ordered Relations," *Logic Journal of the IGPL*, vol. 13, no. 5, pp. 587–597, 2005.

[21] D. R. Raymond, *Partial-order databases*. PhD thesis, University of Waterloo, Waterloo, Ontario, Canada, 1996. Adviser-W. M. Tompa.

[22] R. Nedbal, "Model of preferences for the relational data model," in *Intelligent Models, Algorithms, Methods and Tools for the Semantic Web Realisation* (J. Štuller and Z. Linková, eds.), (Prague), pp. 70–77, Institute of Computer Science Academy of Sciences of the Czech Republic, October 2006.

[23] P. Seshadri, M. Livny, and R. Ramakrishnan, "The design and implementation of a sequence database system," in *VLDB '96: Proceedings of the 22th International Conference on Very Large Data Bases*, (San Francisco, CA, USA), pp. 99–110, Morgan Kaufmann Publishers Inc., 1996.

[24] R. Ramakrishnan, D. Donjerkovic, A. Ranganathan, K. S. Beyer, and M. Krishnaprasad, "Srql: Sorted relational query language," in *SSDBM '98: Proceedings of the 10th International Conference on Scientific and Statistical Database Management*, (Washington, DC, USA), pp. 84–95, IEEE Computer Society, 1998.

[25] A. Lerner, *Querying Ordered Databases with AQuery*. PhD thesis, ENST-Paris, France, 2003.

[26] A. Lerner and D. Shasha, "Aquery: Query language for ordered data, optimization techniques, and experiments," in *29th International Conference on Very Large Data Bases (VLDB'03)*, (Berlin, Germany), pp. 345–356, Morgan Kaufmann Publishers, September 2003.

[27] L. Libkin, *Aspects of partial information in databases*. PhD thesis, University of Pensylvania, Philadelphia, PA, USA, 1995.

[28] R. Agrawal and E. Wimmers, "A Framework for Expressing and Combining Preferences.," in *SIGMOD Conference* (W. Chen, J. F. Naughton, and P. A. Bernstein, eds.), pp. 297–306, ACM, 2000.

[29] A. Eckhardt, "Methods for finding best answer with different user preferences," Master's thesis, 2006. In Czech.

[30] A. Eckhardt and P. Vojtáš, "User preferences and searching in web resoursec," in *Znalosti 2007, Proceedings of the 6th annual conference*, pp. 179–190, Faculty of Electrical Engineering and Computer Science, VŠB-TU Ostrava, 2007. In Czech.

[31] R. Fagin, A. Lotem, and M. Naor, "Optimal aggregation algorithms for middleware," in *Symposium on Principles of Database Systems*, 2001.

[32] R. Fagin and E. L. Wimmers, "A formula for incorporating weights into scoring rules," *Theor. Comput. Sci.*, vol. 239, no. 2, pp. 309–338, 2000.

[33] P. Gurský, R. Lencses, and P. Vojtáš, "Algorithms for user dependent integration of ranked distributed information," in *Proceedings of TED Conference on e-Government (TCGOV 2005)* (M. Böhlen, J. Gamper, W. Polasek, and M. Wimmer, eds.), pp. 123–130, March 2005.

[34] S. Y. Jung, J.-H. Hong, and T.-S. Kim, "A statistical model for user preference," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 17, pp. 834–843, June 2005.

[35] S. Kaci and L. van der Torre, "Non-monotonic reasoning with various kinds of preferences," in *IJCAI-05 Multidisciplinary Workshop on Advances in Preference Handling* (R. Brafman and U. Junker, eds.), (Edinburgh, Scotland), pp. 112–117, 2005.