Preference Handling in Relational Query Languages

Nedbal, Radim
2009

Preference Handling in Relational Query Languages

Post-Graduate Student:

RADIM NEDBAL

Institute of Computer Science of the ASCR, v. v. i.
Pod Vodárenskou věží 2
182 07 Prague 8, CZ ,

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

120 00 Prague 2, CZ

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 8, CZ

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 paper outlines an approach to preference handling in relational query languages. The approach is based on the assumption that the information on possible outcomes is represented in the relational data model.

1. Introduction

Being one of the basic paradigms of human decision making, preferences are inherently a multi-disciplinary topic, of interest to philosophers, psychologists, political scientists, economists, mathematicians and other people coming from different human-centered disciplines, but facing similar questions. Recently, preferences have been studied in operations research, game theory, and several other areas related to computer science.

The main added value computer science has brought into the research on user preferences is an attempt to automate the whole process of preference handling. The goal of such automation is to make logical and mathematical foundations usable in systems that act on behalf of users or simply support their decisions. These could be (a) decision-support systems dealing with the situation where both the number of choice alternatives is huge, and no professional analyst is available to help a user, e.g., information search and retrieval engines that attempt to provide users with the most preferred pieces of information or web-based recommender systems such as shopping sites that attempt to help users identify the most preferred items, (b) automated problem solvers such as configurators, (c) sophisticated autonomous systems such as personal assistants, robots (e.g., Mars rovers), etc. Consequently, the preference handling has become a flourishing topic in many fields related to computer science (see Fig. 1 on the following page) such as database systems, electronic commerce, human-computer interaction, and numerous areas of artificial inteligence dealing with “choice situations”, e.g., knowledge representation, planning and scheduling, configuration and design, multiagent systems, algorithmic decision theory, computational social choice, and other tasks concerning intelligent decision support or autonomous decision making. In brief, preference-based systems allow finer-grained control over decision making automation and new ways of interactivity, and therefore provide more satisfactory results. In particular, explicit preference modeling provides a declarative way to choose among alternatives, whether these are answers to database queries, solutions of problems to solve, decisions of an autonomous agent, plans of a robot, and so on. Moreover, preference models may provide a clean understanding, analysis, and validation of heuristic knowledge used in existing systems such as heuristic orderings, dominance rules, heuristic rules, etc.

2. Preference Handling Meta-Model

The meta-model of preference handling provides a conceptualization consisting of six basic concepts capturing the most important aspects of preference handling:

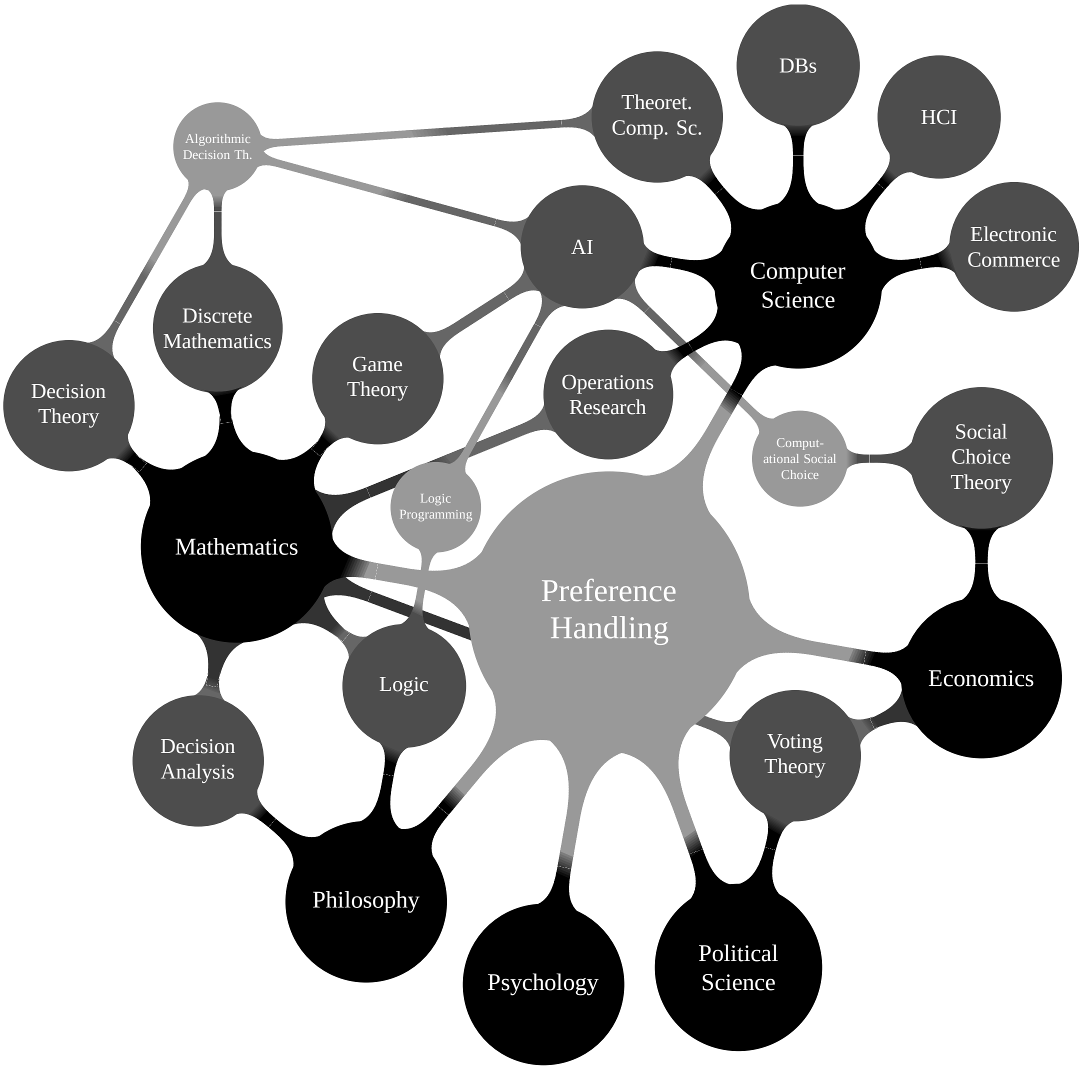
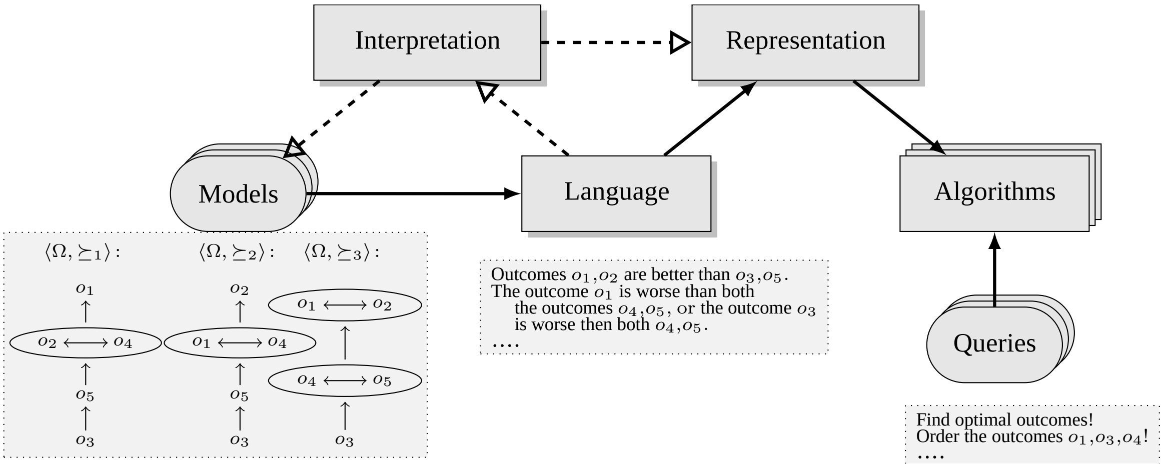


Figure 1: Preference handling mindmap

1. *Preference model* – a suitable mathematical (algebraical) structure that captures properties of specified preferences. (It is the structure we really care about.)
2. *Language* to specify models (ideally in an intuitive, concise manner).
3. *Interpretation* to give the exact meaning to language expressions. (It provides the mapping of the language expressions into a preference model.)
4. *Representation* to capture language expressions in a framework suitable for efficient query-answering algorithms.
5. *Queries* – questions about the models (the questions of interest).
6. *Algorithms* to evaluate answers to queries.

These concepts are depicted and interconnected graphically in Fig. 2 on the next page (adapted from [1]), in which the semantics of directed edges is “choice dependence”, and the dashed directed edges picture the interpretation mapping language expressions to preference models and to instances of representation structure.

To explain the “choice dependance,” note the two key questions that arise when modeling preference handling: What is the model? What queries do we want to ask about this model? Once we have a model and queries, we need algorithms to evaluate these queries about the model. However, algorithms for handling queries about preferences are typically tailored down to the specifics of the representation structure, which captures the language expressions specifying the model. The choice


Interpretation
Representation
Models
Language
Algorithms
Queries
$\langle\Omega,\succeq_1\rangle$:
$\langle\Omega,\succeq_2\rangle$:
$\langle\Omega,\succeq_3\rangle$:
o_1
$o_2 \longleftrightarrow o_4$
o_5
o_3
o_2
$o_1 \longleftrightarrow o_4$
o_5
o_3
$o_1 \longleftrightarrow o_2$
$o_4 \longleftrightarrow o_5$
o_3

Outcomes o_1,o_2 are better than o_3,o_5.
The outcome o_1 is worse than both
the outcomes o_4,o_5, or the outcome o_3
is worse then both o_4,o_5.
....

Find optimal outcomes!
Order the outcomes o_1,o_3,o_4!
....

Figure 2: The meta-model of preference handling

of a language, in turn, depends on the assumptions about the preference models.

Observe that the language, its interpretation, and representation are closely related because an interpretation gives a meaning to expressions in a given language, which can be possibly compactly represented. However, a compact representation is possible only when our preferences can be communicated to the system at hand in terms of concise expressions of the language.

3. The Goal, the Objective, Addressed Questions, and Targeted Activities

Our **goal** is to embed the concept of preference into relational query languages (RQLs).

Accordingly, the **objective** is to provide database users with a *language* that:

1. can express *heterogenous preferences* in an easy *declarative* manner,
2. *compactly* specifies the *preference model*,
3. is based on information that is
   (a) *cognitively easy to express and reflect upon* and
   (b) reasonably *easy to interpret*,
4. has *intuitive*, well defined *semantics* allowing for *conflicting preferences*,
5. allows *representation* that supports *efficient* query-answering *algorithms* for finding optimal matches with respect to *preference models*.

Primarily, the following **questions** have to be **addressed**:

I. Haw can all the capabilities of such a language be embedded into RQLs?
   A) What are the suitable *algebraic operators*[1]?
   B) What are the *algebraic properties* of such operators to lay foundation for algebraic optimization of database queries?
II. What *kinds of preferences* can be expressed by such a language?
III. How can *semantics*
   A) of possibly conflicting preferences be *defined*?
   B) be *computed* effectively?

Consequently, the following **activities** have also to be **targeted** to bring the results into a practice:

⋆ Development of efficient *algorithms for evaluating new algebraic operators*.
⋆ Proposal and analysis of novel *optimization strategies* and their integration with the existing ones.

All these steps are necessary to make the notion of preference a practical concept in RQLs.

4. The Proposed Preference Handling Meta-Model and its Key Concepts

4.1. Models

In general, preferences are expressed over a particular set W of possible worlds. In the relational data model (RDM) context, a possible world can be viewed as a tuple over a finite set $\boldsymbol{A}$ of attributes. Consequently, W can be abstracted to the Cartesian product of the domains of attributes from $\boldsymbol{A}$.

[1] We base ourselves on the algebraic paradigm

We propose to define the *preference model* as a single *preference relation* $\langle W, \succeq \rangle$ – a *partial pre-order* $\succeq$ over the set W of possible worlds (outcomes). In fact, the partial pre-order is introduced in order to capture possible conflicts in preferences in terms of incomparability among worlds.

4.2. Language

As the quantitative type of information is usually cognitively difficult to express and reflect upon, we propose to introduce a declarative language that is based on the qualitative type of information. That is to say, we suggest applying the *qualitative approach* to preference handling. Moreover, the language should enable an easy way to express *various kinds of preferences*.

To lift the propositional approach developed by [2] to the first-order case required by the RDM context, we propose to substitute propositional formulae in the language by *first order queries*. Accordingly, a user preference will be expressed by an appropriate *preference formula* of the form $\varphi \rhd \psi$, where φ, ψ are first order queries and $\rhd$ denotes a distinct kind of preference. These preference formulae constitute a simple declarative language that allows to capture complex, heterogenous preferences.

4.3. Interpretation

Interpretation of preferences (soft requirements) over a set W of possible worlds depends both on the information and mandatory requirements we have on W. This dependence is captured in terms of the so-called *forcing relation*, which represents relationships between individual possible worlds and preference formulae. Thus forcing relation is a parameter of interpretation, which ultimately is formalized by means of the *interpretation function* $\mathscr{I}(x, y)$ of two variables: x for forcing relation and y for a set of preference formulae.

We propose interpretation under ceteris paribus semantics in the sense of "all other things being *similar*", as formalized by [2] in terms of contextual equivalence relation. Moreover, we base ourselves on [3]'s proposal of a minimal logic of preference, in which *any* set of preferences is interpreted in a consistent way. We extend their approach so that *any* set of (possibly heterogenous) preferences, i.e., any set of preference formulae of our proposed language, can be represented by a first-order theory that is satisfiable.

In general, a set of preference formulae has no unique preference model under the proposed interpretation. Therefore, it is necessary to apply non-monotonic reasoning (NMR) mechanisms to identify the *distinguished models* with desired properties. Specifically, we suggest that the distinguished models are those that are maximal with respect to the set inclusion of the preference relation.

4.4. Representation

We want to prove that each set of preference formulae is logically equivalent to a set of disjunctive logic programs (DLPs) that are isomorphic: these DLPs are identical up to a renaming of constants from their Herbrand universes. Most importantly, it can be shown that the cardinality of these Herbrand universes is bounded by a function exponential in the cardinality of the set of preference formulae.

As isomorphic first order formulae have isomorphic models [4], it can be proved that a set of preference formulae is logically equivalent to a set of preference models, each of which is isomorphic to a particular model of a single DLP. Finally, these models are to be used to determine the most preferred possible worlds.

4.5. Queries and Algorithms

The most fundamental type of queries over preference models with the view of embedding the notion of preference in the RQLs is to find the most preferred matches with respect to user preferences.

It can be shown that the proposed distinguished model semantics (refer to Subsect. 4.3) and minimal model semantics of DLP agree. Consequently, the machinery of logic programming can be employed to compute the suggested declarative semantics of a set of preference formulae.

The overall concretization of the meta-model of the proposed approach to preference handling in the database context is depicted in Fig. 3 on the next page.

5. Embedding Preference into Relational Query Languages

5.1. Preference Operator

To filter out bad tuples, database users express a *selection condition*, which is embedded by a *selection* operator of the relational algebra (RA). This selection operator is parameterized by a logical condition that serves as a *hard constraint*. The user gets a perfect match if it is fulfilled. However, not every wish can become true.

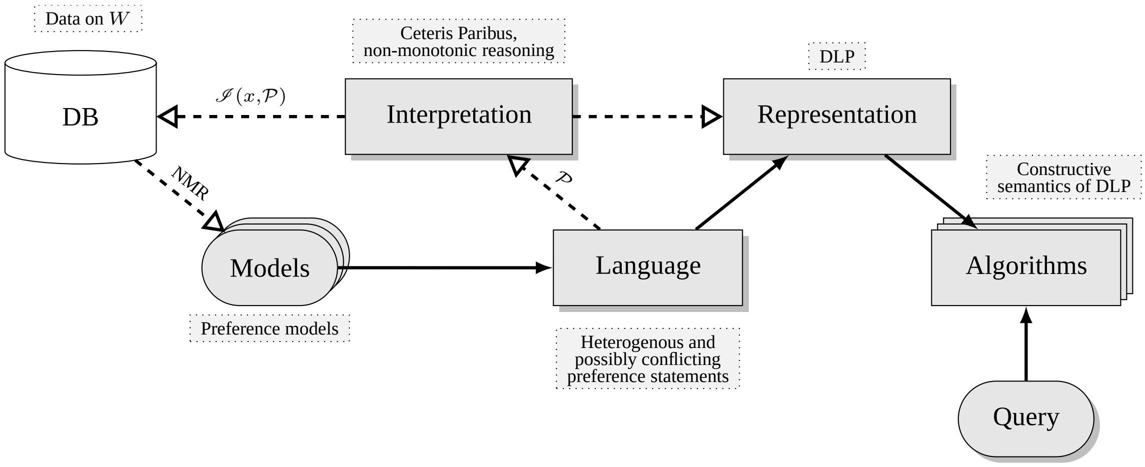

Data on W
DB
$\mathscr{I}(x,\mathcal{P})$
Ceteris Paribus, non-monotonic reasoning
Interpretation
DLP
Representation
NMR
$\mathcal{P}$
Constructive semantics of DLP
Models
Language
Algorithms
Preference models
Heterogenous and possibly conflicting preference statements
Query

Figure 3: The meta-model of the proposed approach

To filter out not all the bad tuples, but only worse tuples than the best matching alternatives, we will introduce a new, *preference operator*, parameterized by user preferences. It selects from its argument relation the most preferred tuples according to its parameter – a set of preference formulae.

5.2. Algebraic Optimization

In general, the algebraic optimization aims at minimizing the data flow during the query execution. Basically, it utilizes various optimization strategies such as pushing *selection* and *projection* operators down the *query execution tree*. These strategies, in turn, are based on the assumption that early application of the selection or projection operator reduces intermediate results. As input relations are usually too big to fit into *main memory*, using the number of the *secondary storage I/O's* as our measure of cost for an operation, it is easily seen that this reduction of intermediate results has a remarkable positive impact on the performance of query processing.

To provide a formal foundation for algebraic optimization, the focus should be on abstract properties of the preference operator. These abstract properties include algebraic rules that describe the interaction of the preference operator with other RA operators. By considering the preference operator on its own, we should be able, on one hand, to focus on the abstract properties of user preferences and, on the other hand, to study special evaluation and optimization techniques for the preference operator itself.

We propose a new, analogical optimization strategy of *pushing the preference operator* down the query execution tree. Most importantly, sufficient conditions under which the preference operator commutes with selection or projection, or can be distributed over *cartesian product* or *union* are identified.

6. Related Work – Preference in Database Systems

The study of preference in the context of database queries has been originated by [5]. They, however, don't deal with algebraic optimization. Following their work, *preference datalog* was introduced in [6], where it was shown that the 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 has attracted broader interest again. [7, 8, 9, 10] and [11, 12, 13, 14] pursued independently a similar (*qualitative*) approach within which preferences between tuples are specified directly, using binary *preference relations*. The embedding into RQL they have used is similar to ours: they have defined an operator returning only the best preference matches. In particular, they provided rewriting rules for the operator to lay foundation for algebraic optimization of database queries with preferences. Their optimization framework extends established query optimization techniques: preference queries can be evaluated by extended – preference RA. While some transformation laws for queries with preferences have been presented in [15, 16], the results presented in [11] are mostly more general.

A special case of the same embedding represents *skyline operator* introduced by [17]. Some examples of possible rewritings for skyline queries were given, but no general rewriting rules were formulated.

Building on the recent advances in logic of preference, [18] suggested a framework within which preferences between tuples are specified indirectly, using a declarative language based on the qualitative type of information. His language captures various kinds of preferences and allows for comfortable specification of preferences. The embedding of the concept of preference into RQLs is similar to that of [7] and [11]: it is realized by means of the *preference operator* returning only the best preference matches. By contrast, the best preference matches, in general, are sets of tuples. Basing himself on this framework, [19] aims at algebraic optimization of RQLs with preferences. In particular, he identifies the algebraic properties governing the interaction of the preference operator with the other operators RA. However, the semantics of the preference operator is unnatural in the sense that it is not based on the *closed world assumption* (CWA) – an implicit hypothesis standardly used in the realm of database systems.[2]

[20] addressed the issue of extending the RDM to incorporate partial orderings into data domains. Partially ordered data domains, in turn, are the leitmotiv of the approach to preference queries over web repositories [21]. Also in [22], actual values of an arbitrary attribute domain are allowed to be partially ordered according to user preferences. Accordingly, RA operations, aggregation functions and arithmetic are redefined. However, some of their properties are lost, and the query optimization issues are not discussed. Finally, [23] proposed a data structure for an effective representation of information representable by a partial order.

A comprehensive work on partial order in databases, presenting the partially ordered sets as the basic construct for modeling data, is [24]. 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 [25], SRQL [26], Aquery [27, 28]. Various kinds of ordering on power-domains have also been considered in the context of modeling incomplete information: an extensive and general study is provided in [29].

By contrast, preference is specified indirectly using *scoring functions* within the *quantitative* approach [30, 31, 32, 33, 34, 35, 36]. A scoring function associates a numeric score with every tuple.

7. Conclusions

We propose a framework for embedding preferences into RQLs. The framework relaxes assumptions that are inherent in traditional approaches to preference handling in the database systems. Specifically, various kinds of preferences are taken into account. Most importantly, the proposed approach ensures that any set of user preferences (preference specification) specified in our language can be interpreted in a consistent way. Another distinctive feature of the framework is the utilization of logic programming machinery to efficiently compute preference models. Building on recent leading ideas that have contributed to remarkable advances in the field, the framework also deals with the optimization of relational queries:

- 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.
- 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.

Consequently, we propose “pushing the preference operator strategy”, which is based on its algebraic properties.

References

[1] R. I. Brafman and C. Domshlak, “Preference handling – an introductory tutorial,” Tech. Rep. 08-04, Computer Science Department, Ben-Gurion University, Negev Beer-Sheva, Israel 84105, December 2007.

[2] J. Doyle and M. P. Wellman, “Representing preferences as ceteris paribus comparatives,” in *Decision-Theoretic Planning: Papers from the 1994 Spring AAAI Symposium*, pp. 69–75, AAAI Press, Menlo Park, California, 1994.

[3] G. Boella and L. W. N. van der Torre, “A non-monotonic logic for specifying and querying preferences.,” in *IJCAI* (L. P. Kaelbling and A. Saffiotti, eds.), pp. 1549–1550, Professional Book Center, 2005.

[4] V. Švejdar, *Logika: neúplnost, složitost a nutnost (Logic: Incompleteness, Complexity, and Necessity).* Praha: Academia, 2002. In Czech.

[2] CWA basically states that all the facts not in the database are false.

464 pages. With a section on Gödel-Dummett logic written by Petr Hájek.

[5] 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.

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

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

[12] 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.

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

[14] 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.

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

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

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

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

[23] 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.

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

[25] 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.

[26] 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.

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

[28] 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.

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

[30] 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.

[31] A. Eckhardt, “Methods for finding best answer with different user preferences,” Master’s thesis, 2006. In Czech.

[32] 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.

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

[34] 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.

[35] 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.

[36] 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.