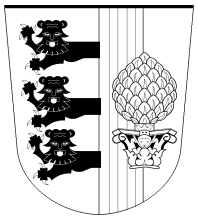
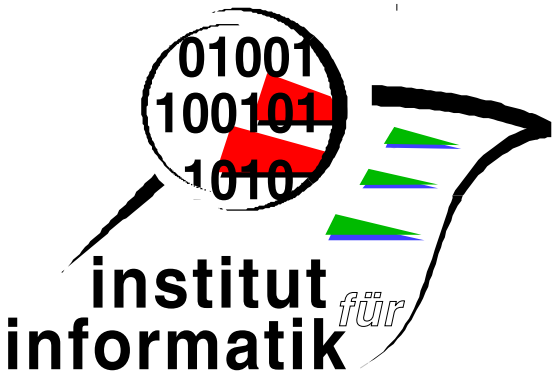
Universität Augsburg



Personalized Nonlinear Ranking Using Full-text Preferences

Achim Leubner, Werner Kießling

Report 1999-05 September 1999



Institut für Informatik
D-86135 Augsburg

Personalized Nonlinear Ranking Using Full-text Preferences

Achim Leubner, Werner Kießling

Institut für Informatik
Universität Augsburg

{leubner, kiessling}@informatik.uni-augsburg.de

Abstract

Today Internet information systems commonly use a total ranking to present search results. These rankings are typically cut off at arbitrary points which are hard to understand. In this paper we present a new approach for rankings based on partial orders, which model personal preferences. It naturally groups large result sets according to the quality of results and presents only the top ones. It is possible for the user to expand these result sets selectively along chains of the partial order. We expect a considerable gain in comprehensibility, clarity and user friendliness. A pilot application is being implemented and first encouraging evaluation results are reported.

Keywords: Personalized Information Systems, Full-text Preferences, Nonlinear Ranking, Partial Orders, Preference SQL.

1 Introduction

Today the Internet provides various personalized information services adopted to a single user's needs. Sources like [myCNN], [InfoBeat] or [My-Newspaper] deliver personal newspapers, i.e. you can choose between several rubrics and services offered by them. With 'meta newspapers' like the German [Paperball] you can even compose your personal newspaper by choosing specific sections of different online newspapers. They also offer personal rubrics by searching for user-given keywords in user-selected repositories. These and other personalized information services provide the opportunity to save considerable amounts of time otherwise necessary for browsing through piles of irrelevant

information. Therefore, they undoubtedly provide advantages compared to conventional non-personalized information sources. But even Paperball's personal rubrics are only very basic tools. Although there are technologies to extract keywords from sample documents and to adjust them using relevance feedback, people still have to choose keywords and are responsible for adapting them to their changing interests on their own. Moreover, query results are always presented as linear ranked lists.

In this paper we propose a new approach based on system-derived user preferences, which are modeled as partial orders of keywords. We limit the initially presented information to the top-scored documents. The user may then decide if she wants to see more and thus expand the presentation selectively according to her information need. In the next section we will concentrate on the relation between our approach and conventional information retrieval/filtering. In section 3 we model preferences more theoretically defining base preferences and multiset preferences. We also introduce how to combine preferences into complex ones. Section 4 focuses on our approach of a personalized information service with nonlinear ranking and presents a prototype implementation. We conclude the paper with our preliminary results and a short outlook.

2 Relation to Information Retrieval and Filtering

All WWW information systems we are aware of, while using different retrieval techniques, present their results as linear ranked lists. The ranks typically are based on the correspondence between a set of keywords and the searched documents, and are adjusted according to term frequency, etc. These techniques often assign non-zero scores even to irrelevant documents. To relieve the user from browsing through all (including irrelevant) documents, a cutoff point has to be chosen. This means limiting the result set by either presenting the N top results or using a threshold. It's a general and important problem how to choose an optimal N or a suitable threshold, as a small N worsens recall and a too large one precision. It is known that 'top N' approaches often don't deliver optimal precision/recall ([FoDu92]).

In our opinion the user can decide better about an optimal cutoff point. To aid the user in doing so, the system has to aggregate the information content in an intelligible way. Therefore, we don't rely on total orders but use *partial orders* on keywords. In this way the user gets the best-matching documents first. These matches imply a high precision (but possibly a bad recall). The matches are grouped with respect to the keywords contained. It is also possible for the user to enlarge the result explicitly, thus tendentiously improving recall at the expense of precision. In contrast to total ranking solutions this can be done selectively along chains of the partial order.

Of course, there are other approaches to improve the clearness of the presentation, for example clustered representations as used in [ZaEt99]. The basic difference between these approaches and our approach is the use of a total ordering. Whereas clustered representations typically display the clusters as ranked lists, our approach would lead

to a presentation of partial order of clusters delivered by the retrieval component below our system.

3 A Model for Full-Text Preferences

In this section we briefly describe the concept of preferences and the theoretical foundations of our solution. Importantly, as shown in [KoKi95], computational models using partial orders are compatible with relational database technology. We first focus on base preferences and then introduce multiset preferences.

3.1 Base Preferences

Definition 1
A base preference is a partial order $\langle V, > \rangle$ *a set* V *of values of a particular data type.*

Let's assume we want to buy an iMac computer. As everybody knows, the most important decision in this case is which color to choose. Today the iMac is offered in the colors blueberry, strawberry, tangerine, grape and lime, ([iMac]). Since it's such an important decision we are a little bit hesitant, but we develop some clear preferences as shown in figure 1.

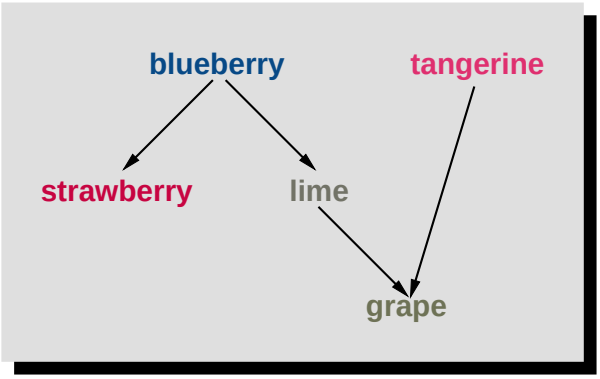


Figure 1: The iMac example

If we enter this preference into an ordering system at our local vendor, it should offer us the best available iMac's according to our preference. I.e. it would offer us a blueberry and/or tangerine version as best selections. Otherwise if neither blueberry nor tangerine versions are on stock, it should offer us a strawberry and/or a lime version. And only if none of these can be delivered it should offer a grape version.

3.2 Multiset Preferences

Consider a more complicated example: Let's say we are interested in jazz and want to known what's going on. We pose our query on a database containing all recent articles of an international music magazine. So our preference on a set of keywords V may be the one shown in figure 2.

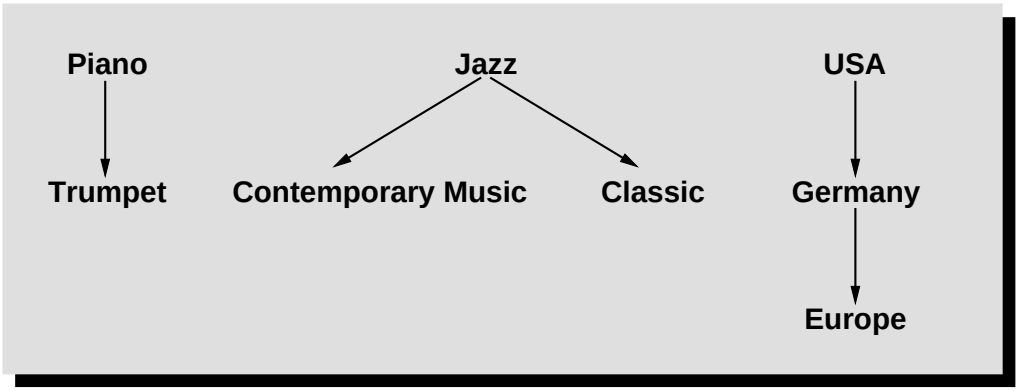


Figure 2: A query against an international music magazine

This time we cannot simply apply this preference to the column containing the articles, since in each article various keywords may occur simultaneously. Consider documents containing 'Classic' and 'Piano' respectively 'Jazz' and 'Trumpet'. Since 'Piano' is better than 'Trumpet' according to the given preference, the first document should be preferred. But this would be inconsistent with the preference of 'Jazz' over 'Classic'.

To solve this problem we represent each article by the multiset of matching keywords from V. These multisets are elements of the set of finite multisets $M(V)$ over V. We derive our order on $M(V)$ by constructing a multiset order.

Definition 2
For the given partially ordered set $(V, >)$*, the multiset preference* $(M(V), \gg)$ *is defined for* $A, B \in M(V)$ *as follows:*

$$A \gg B \iff \exists\ \emptyset \neq X \subseteq A, Y \in M(V) : \underbrace{(A \backslash X) \cup Y \supseteq B}_{I} \wedge \underbrace{\forall\ y \in Y\ \exists\ x \in X : x > y}_{II}$$

We omit the proof that $\langle M(V), \gg \rangle$ is a partial order since it's only a simple variation of the multiset order known from term rewriting (see [DeMa79]). Obviously, we get our base order $\langle V, > \rangle$ if all compared sets contain only one element.

Informally, to show that A is better than B, we have to choose a non-empty submultiset X of A. We then replace this multiset by new elements in Y. For each new element there has to be a better element in the replaced multiset (condition II). If it is possible

to worsen A this way such that B or a supermultiset of it is created (condition I), then A is better than B.

For example, to show that {USA, Piano, Classic} is better than {USA, Trumpet}, we may choose to replace {Piano} with {Trumpet}. Since 'Trumpet' is worse than 'Piano' according to the base preference, the substitution is permitted. This way we get a superset of {USA, Trumpet}, so {USA, Piano, Classic} is better than {USA, Trumpet}.

Now we continue our example: For articles represented by the sets {USA, Trumpet, Jazz}, {Europe, Piano, Jazz}, {Germany, Trumpet, Jazz}, {Europe, Classic} and {Europe} we get the partial order $\gg$ in figure 3.

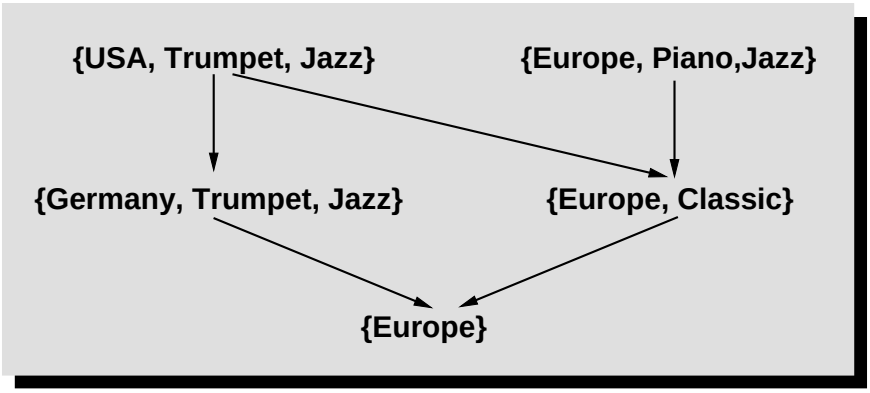


Figure 3: The result of above query against an international music magazine

An arrow from A to B means that A is preferred over B. E.g. {Europe, Piano, Jazz} is more interesting than {Europe, Classic} according to the given preference.

3.3 Combinations of Preferences

Both types of preferences, base preferences and multiset preferences can be combined by either *prioritization* or *cumulation* to form a more complex preference:

- Priorization prefers a preference over another.
- Cumulation treats preferences as equally important.

Combined preferences can be nested the same way.

3.3.1 Prioritized Preferences

Reconsidering the result of the multiset example (see figure 3), we observe that the documents containing only the keyword 'Europe' are probably not relevant for us, since we are specifically interested in music here. A solution is splitting the preference from

figure 2 on page 4 into two multiset preferences combined by prioritization. This way the first preference is stressed: An object is better, if it's better according to the first base preference. If it's not better and not worse according to the first base preference, the second preference decides.

Definition 3
Formally, for the preferences $\langle X_1, > \rangle$ *and* $\langle X_2, > \rangle$*, the prioritized preference* $\langle X_1 \cdot X_2, > \rangle$ *is defined by:*

$$\forall\, (a_1, a_2), (b_1, b_2) \in X_1 \times X_2 : (a_1, a_2) > (b_1, b_2) \Longleftrightarrow a_1 > b_1 \vee (a_1 \not> b_1 \wedge a_1 \not< b_1 \wedge a_2 > b_2)$$

We omit the proof that $\langle X_1 \cdot X_2, > \rangle$ is a partial order since it's only a slight variation of the lexicographic order on the cartesian product as described in [DaPr90].

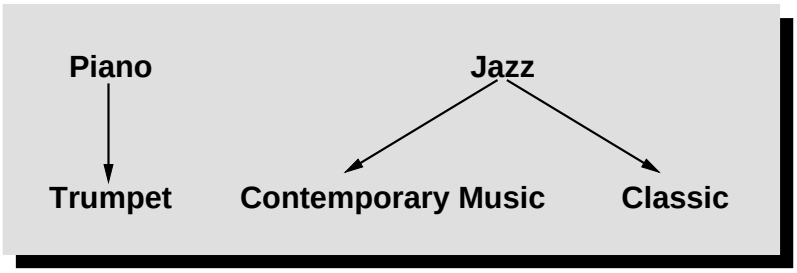


Figure 4: The first part of the split music preference

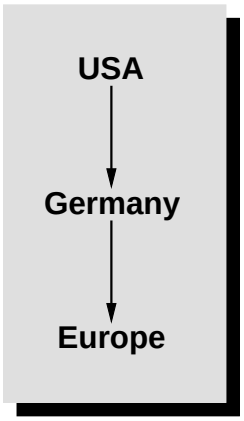


Figure 5: The second part of the split music preference

In our example, we split the preference from figure 2 on page 4 into the preferences in figure 4 and 5. The preference in figure 4 is our first preference to consult and the one in figure 5 the second.

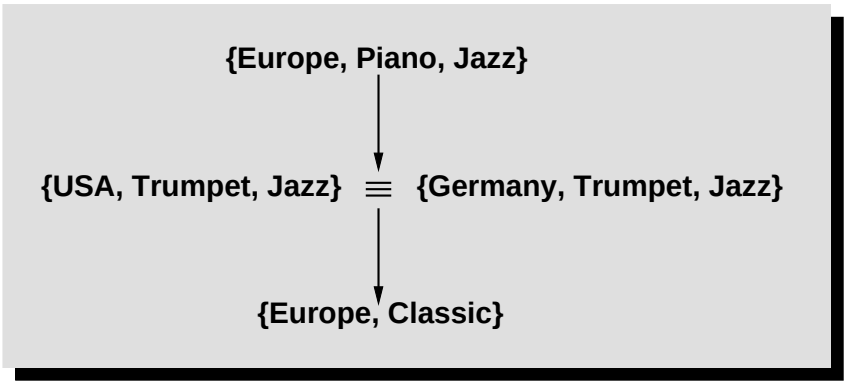


Figure 6: The preliminary result

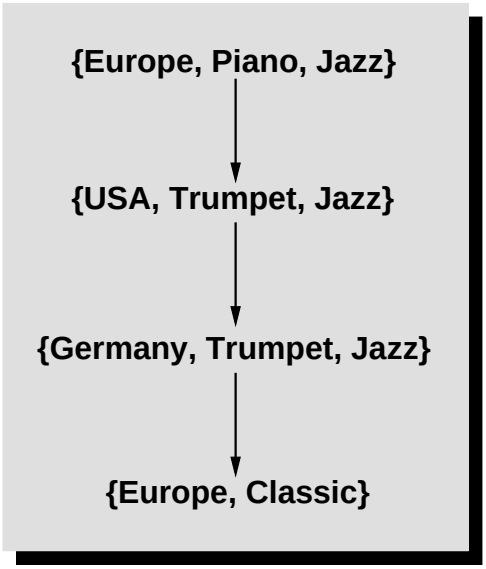


Figure 7: The final result

Now let's consider the result for the document set of the multiset example on page 5. The simple base preference from figure 4 on the page before applied to the document set leads to the order in figure 6. The sets {USA, Trumpet, Jazz} and {Germany, Trumpet, Jazz} are equivalent at this stage, since the first preference ignores the terms USA and Germany.

At the next stage all keyword sets that are equivalent or incomparable according to the first preference are compared with respect to the second preference. This leads to the order in figure 7.

3.3.2 Cumulated Preferences

Reconsidering the iMac example on page 3, we recognize that we additionally prefer an offer with 64 MB RAM over the standard offer with 32 MB. We may combine both preferences using a *cumulated preference*.

Definition 4
Formally, the cumulated preference $\langle X_1 + X_2, > \rangle$ *for the preferences* $\langle X_1, > \rangle$ *and* $\langle X_2, > \rangle$ *is defined by:*

$$\forall\, (a_1, a_2), (b_1, b_2) \in X_1 \times X_2 : (a_1, a_2) > (b_1, b_2) \Longleftrightarrow (a_1 > b_1 \wedge a_2 \geq b_2) \vee (a_2 > b_2 \wedge a_1 \geq b_1)$$

This means a document is better if it's really better with respect to one of the base preferences and at least as good with respect to all other base preferences. It also implies that both documents have to be comparable with respect to all base preferences. For the example it means we prefer an iMac over another, if it has a better color or more RAM and is at least equally good with respect to the other criterion. Cumulation is a version of the coordinatewise partial order on the cartesian product as described in [DaPr90].

4 Prototype Implementation

In this section we will present our vision of a personalized information service based on preferences. We also describe which parts of preference technology already have been implemented.

4.1 A Personalized Information Service with Full-text Preferences

A personalized information service for our approach would require the user to select predesigned facets of preferences based on natural-language descriptions associated with the facets. The system would combine them into a set of preferences. We do not expect user's to design preferences on their own. Each time the user queries the system the top documents that have not already been delivered are presented with a short summary, description or a snippet and grouped according to the partial order. We believe partial orders are a natural way to express personal preferences. In contrast to numerical calculation or boolean logic, most people are familiar with preferences from early youth on. Therefore we expect this presentation to improve the comprehensibility of the results. The user can then select relevant articles or extend the result set. Since preferences generated from predesigned facets are only rough approximations of the user's interests and since the user's interests continually change, the system has to use the implicit relevance feedback given by the user (e.g. which articles the user really selected) to adapt the preferences continuously. Figure 8 on the next page shows the gross architecture of such a personalized information service.

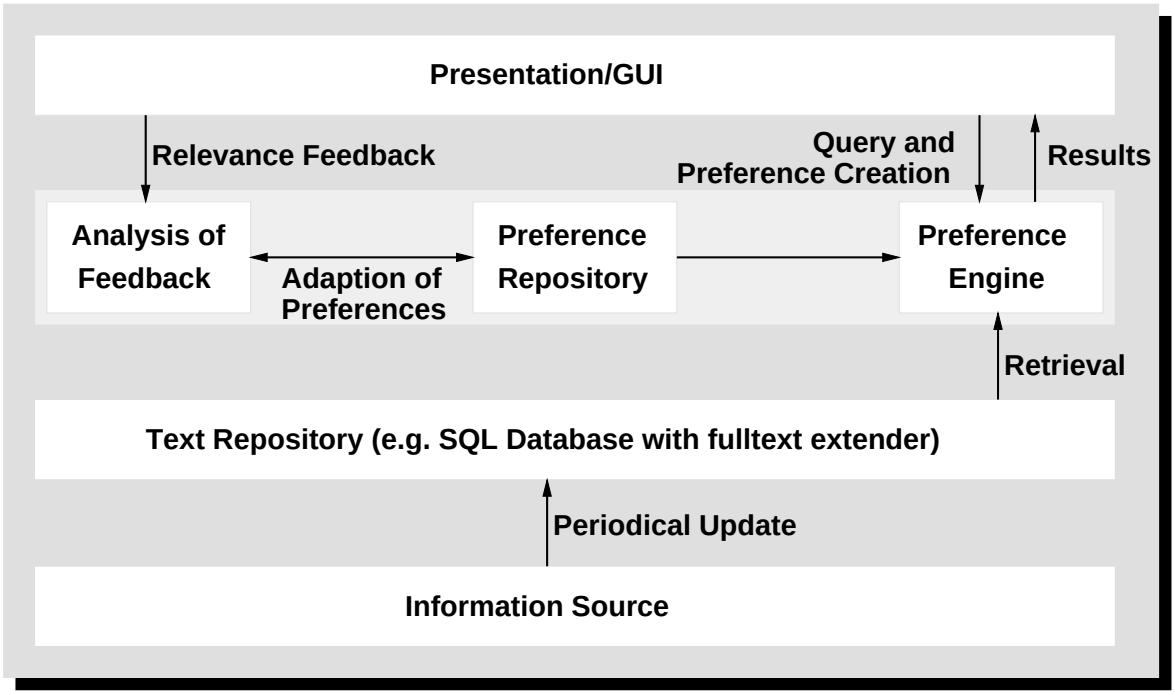


Figure 8: Gross architecture of a personalized information service

The underlying conventional information retrieval or filtering system gets its information from an arbitrary information source. The preference engine then retrieves information from the text repository according to the preferences stored in the preference repository. The results are presented by a GUI that returns the user's feedback to an analysis component. This component adjusts the preferences, if necessary.

4.2 What's already Implemented

So far there is one commercial implementation of base preferences including combined preferences together with commercially important basic preference types and features not mentioned here ([PrefSQL]). Preference SQL is basically an additional layer between any SQL-compliant database and an arbitrary application. This layer translates the Preference SQL extensions to standard SQL. The application can interact with Preference SQL using JDBC or ODBC.

In Preference SQL extended by multiset preferences, the previous example for prioritization in figure 5 on page 6 can be formulated as:

```
CREATE PREFERENCE DOMAIN jazzDomain AS CHAR(*)
   (CHECK VALUE IN ('Piano', 'Trumpet', 'Jazz',
                    'Contemporary Music', 'Classic',
                    'USA', 'Germany', 'Europe'));
```

```
CREATE PREFERENCE jazzPref1 (
   MULTISET PREFERENCE IN (VALUES
      ( 'Piano', 'Trumpet' ),
      ( 'Jazz', 'Contemporary Music' ),
      ( 'Jazz', 'Classic' )),
   PREFERENCE DOMAIN jazzDomain);

CREATE PREFERENCE jazzPref2 (
   MULTISET PREFERENCE IN (VALUES
      ( 'USA', 'Germany' ),
      ( 'Germany', 'Europe')),
   PREFERENCE DOMAIN jazzDomain);

    SELECT Title, Date, Description
      FROM Article
     WHERE Date > DATETIME(1999-06-01)
PREFERRING jazzPref1(Text) PRIOR TO jazzPref2(Text);
```

For multiset preferences there is a first research prototype implemented at the University of Augsburg in Java with JDBC on top of the relational database system DB2 5.2. This prototype is currently restricted to sets instead of multisets. The text repository currently contains about 59000 articles (more than 800 MB) from the online version of the German newspaper 'Die Welt' ([Welt]). Every night the new online edition is inserted into the database. The prototype evaluates multiset preferences containing arbitrary full-text expressions. It is possible to filter potential results for hard constraints by using a normal SQL where-clause (as shown above).

Obviously, the worst-case complexity of algorithms evaluating multiset preferences is very high. But our preliminary experiences show that for reasonable numbers of documents satisfying the hard selection condition (e.g. 900) and keywords (e.g. 20) even our Java-based prototype calculates the complete partial order in about 11 seconds on a Linux system with a 300 MHz Intel Pentium II CPU using the JDK with a JIT-Compiler (tya). Therefore further optimizations along with a modification of the prototype algorithms to calculate preferences incrementally should lead to practicable average performance even for large scale applications.

5 Summary and Outlook

In this paper we proposed a new approach to rankings based on partial orders. Compared to total rankings, typically used for Internet information services, we expect our approach to lead to a more comprehensible presentation of query results. Our technique empowers the user to enlarge the presented result set selectively along chains of the partial order.

This is a fundamental advantage over conventional total rankings where as a matter of principle also uninteresting chains are expanded by lowering the threshold or enlarging the number of shown results. Further on, our approach only returns empty result sets if no keyword matches at all. In all other cases only the top results are presented, even if they don't match perfectly. This often relieves users from refining their queries, when the result set is too large or empty.

Our preliminary experiences with a first prototype have confirmed these expectations. They also are encouraging with respect to the performance. Although the theoretical worst-case complexities of our algorithms are very high, optimized algorithms are practicable for many real world applications. Along with implementing a more complete prototype with respect to the evaluation of combined preferences, we plan to implement implicit relevance feedback to adjust user preferences in the future. An online version of the prototype is supposed to be available soon.

Acknowledgements

The authors thank Gerhard Köstler, Ebenezer Ntienjem and Tilo Balke for helpful discussions.

References

[DaPr90] B. A. Davey, H. A. Priestley.
Introduction to Lattices and Order.
Cambridge University Press, 1990, ISBN 0-521-36766-2.

[DeMa79] Nachum Dershowitz, Zohar Manna.
Proving Termination with Multiset Orderings.
Communication of the ACM, August 1979, Volume 22, Number 8.

[FoDu92] Peter W. Foltz, Susan T. Dumais.
Personalized Information Delivery: An Analysis of Information-Filtering Methods.
Communications of th ACM, December 1992, Volume 35, Number 12.

[iMac] *iMac Page*
http://www.apple.com/imac/

[InfoBeat] *InfoBeat*
http://www.infobeat.com/

[KoKi95] Gerhard Köstler, Werner Kießling, Helmut Thöne, Ulrich Güntzer.
Fixpoint Iteration with Subsumption in Deductive Databases

Journal of Intelligent Information Systems:123-148, Volume 4, Kluwer Academic Publishers, 1995.
(http://www.Informatik.Uni-Augsburg.DE/info2/literature/Papers/jiis95.html)

[myCNN] *myCNN.com*
http://customnews.cnn.com/

[My-Newspaper] *My-Newspaper*
http://www.my-newspaper.com/

[Paperball] *Paperball*
http://www.paperball.de/

[PrefSQL] *Preference SQL 1.2 Reference Manual*
Database Preference Software GmbH, Augsburg, May 1999.
(http://www.preference.de)

[Welt] *Die Welt online*
http://www.welt.de/

[ZaEt99] Oren Zamir, Oren Etzioni.
Grouper: A Dynamic Clustering Interface to Web Search Results
World Wide Web Conference 99,
(http://www8.org/w8-papers/3a-search-query/dynamic/dynamic.html).