Ranking RDF with Provenance via Preference Aggregation

Renata Dividino, Gerd Gröner, Stefan Scheglmann, and Matthias Thimm

Institute for Web Science and Technologies (WeST), University of Koblenz-Landau, Germany
{dividino,groener,schegi,thimm}@uni-koblenz.de

Abstract. Information retrieval on RDF data benefits greatly from additional provenance information attached to the individual pieces of information. Provenance information such as origin of data, certainty, and temporal information on RDF statements can be used to rank search results according to one of those dimensions. In this paper, we consider the problem of aggregating provenance information from different dimensions in order to obtain a joint ranking over all dimensions. We relate this to the problem of preference aggregation in social choice theory and translate different solutions for preference aggregation to the problem of aggregating provenance rankings. By exploiting the ranking orderings on the provenance dimensions, we characterize three different approaches for aggregating preferences, namely the lexicographical rule, the Borda rule and the plurality rule, in our framework of provenance aggregation.

1 Introduction

When querying the Web we are faced with highly varying quality of information. Techniques to find the relevant information out of the Web data should include ways to investigate the value of information. Provenance provides knowledge that can be used to quantify this value, and may come in different forms, e. g., trustworthiness of a source, time of validity, certainty and vagueness. Determining the top-k answers of a query that includes provenance information is an emerging problem.

Recent works [7, 22, 8, 19] have proposed frameworks for representing provenance structures. Provenance is associated to RDF statements in form of *annotations* and the different forms of provenance, denoted as *annotation dimensions*, are captured by an algebraic structure [11], which enables annotation propagation through query evaluation. Given a set of query results where each tuple is associated with annotations, we consider the problem of ranking the tuples according to the annotation dimensions. Usually, determining the top-k results when considering only one annotation dimension, for instance the temporal dimension, is done by sorting the results according to *all* the time values in order of importance (increasing or decreasing order). Ranking query results when an aggregation of many annotation dimensions is necessary, e. g., temporal and fuzzy, poses a variety of further challenges.

In this paper, we relate the problem of ranking stemming from an aggregation of different annotation dimensions to the problem of *preference aggregation* (or *judgement*

ID	Statement	Time
#1	[Cinemark Palace plays The Grey]	05.05.12
#2	[Cinemark Palace plays Man on a Ledge]	05.05.12
#3	[Prado Cinema Cafe plays Underworld: Awakening]	06.06.12
#4	[Deerfield Cinema plays Man on a Ledge]	02.03.12
#5	[The Grey hasGenre Action]	05.05.12
#6	[Man on a Ledge hasGenre Action]	02.03.12
#7	[Underworld: Awakening hasGenre Action]	02.03.12

Table 1. The set of annotated RDF statements in Ex. 1

aggregation) in social choice theory [15]. By adopting methods from preference aggregation, we formulate a general framework for annotation dimension aggregation that is based on solid formal grounds. We translate different solutions for preferences aggregation to the problem of aggregating provenance rankings. By exploiting the ranking orderings on the provenance dimensions, we characterize three different approaches for aggregating preferences, namely the lexicographical rule, the Borda rule, and the plurality rule, in our framework of provenance aggregation.

The combination of provenance has been considered before in e. g. [17] where the focus is on the representation of the combined annotation dimensions based on their possible dependency interactions. In [24], the aggregation of results from two specific dimensions is presented, but, to our knowledge, the problem of a general aggregation approach of multiple dimensions for top-k ranking has not been considered yet.

This paper is organized as follows. Section 2 describes foundations of representing and querying RDF with provenance. Section 3 presents the formal framework for ranking taking into consideration all annotation dimensions. In Section 4, we discuss how the aggregation is applied in an offline setting and for streaming data. We compare our work with related ones in Section 5 and conclude with a summary in Section 6.

2 Foundations

We follow the representation for annotated RDF used in [8]. Let p be some annotation dimension (such as *time*) and Ω_p its set of possible values, e. g., the set of all valid dates. Then, an annotated RDF statement is a quadruple $\alpha : \theta$ where α is an RDF statement and $\theta \in \Omega_p$ the attached annotation.

Example 1. Table 1 shows some RDF statements about movie theaters and films. Each statement is annotated with an element from the temporal annotation dimension. In the first column, we represent the statements in a simplified textual syntax. The second column shows the annotation value $\theta \in \Omega_p$ assigned to the statement. For example, the statement #1 states that the movie theater *Cinemark Palace* plays the film *The Grey*. The annotation value assigned to this statement is the timestamp 05.05.12. saying that this information has been published on this date.

We assume that annotation dimensions are represented in form of commutative *semirings* [11]. Let $K_p = (\Omega_p, \otimes_p, \oplus_p, \top_p, \bot_p)$ be a commutative and idempotent semiring

of an annotation dimension p where Ω_p is the value domain of p, $\otimes_p$, $\oplus_p$ are custom user-defined binary functions on Ω_p, and $\top_p$ and $\bot_p$ are top and bottom elements. On K_p we can define a *partial order* $\preceq_p$ on the values in Ω_p. For all $\theta_1, \theta_2 \in \Omega_p$, $\theta_1 \preceq_p \theta_2$ (θ_1 precedes θ_2) if and only if there is $\theta_3 \in \Omega_p$ such that $\theta_1 \oplus \theta_3 = \theta_2$. We write $\theta_1 \equiv \theta_2$ if both $\theta_1 \preceq \theta_2$ and $\theta_2 \preceq \theta_1$. For technical convenience, we only consider annotation dimensions p such that the order $\preceq_p$ is a *total preorder.*

Example 2. Let Ω_p be the set of all possible dates, $\otimes_p$ be the minimum function and $\oplus_p$ the maximum function, then we obtain the natural linear order $\preceq_p$ for dates, i. e., for $\theta_1, \theta_2 \in \Omega_p$ it holds that $\theta_1 \preceq_p \theta_2$ if and only if the date represented by θ_1 is not later than the date represented by θ_2.

There are various query languages [8, 24] for annotated RDF graphs that have been developed on top of SPARQL [21]. Those languages extend the algebra operators of SPARQL to compute annotation values of results, i. e., enable annotation propagation through queries using the algebraic operators of the underlying annotation structure. Basically, annotation values for some annotation dimension p can be propagated through query evaluation along *how-provenance* [11], i. e., annotation values derived from the individual derivation trees used to assign the variables. For a query q and an RDF graph G, we define $\Phi(r) \in \Omega_p$ to be the annotation value for a query result r in the result set $res(G, q)$. We refer to [8] for a complete formalization of how $\Phi(r)$ is determined.

Example 3. Let q be a query that asks for all movies theaters that play some action movie[1]. Querying the (annotated) RDF graph G from Ex. 1 with query q yields the result set $res(G, q) = \{r_1, r_2, r_3\}$ where $r_1 = \langle$Cinemark Palace$\rangle$, $r_2 = \langle$Prado Cinema Cafe$\rangle$, and $r_3 = \langle$Deerfield Cinema$\rangle$. Given the semantics of the semirings operators $\oplus$ and $\otimes$ in Ex. 2 and since r_1 is derived by joining the statements #1 and #5 or #1 and #6, its the provenance value is be obtained by evaluating the respective annotation values $(05.05.12 \otimes 05.05.12) \oplus (05.05.12 \otimes 02.03.12) = 05.05.12$, which corresponds to the most cautious but valid timestamp r_1 bases upon.

Finally, the order $\preceq_p$ on the annotation values of the annotation dimension p can be exploited to obtain a (partial) ranking on the results $res(G, q)$ for some query q. To do so, we extend the order $\preceq_p$ on $res(G, q)$ by defining

$$r \preceq_p r' \qquad \text{if and only if} \qquad \Phi(r) \preceq_p \Phi(r')$$

for all $r, r' \in res(G, q)$.

Example 4. In Ex. 3 $\{r_1, r_2, r_3\}$ are returned in the order $\langle r_3, r_1, r_2\rangle$, which reflects the recency of the statements those results are based upon.

3 Multiple Annotation Dimensions and the Aggregation Problem

Now we extend the concept of annotated RDF to allow for more than a single annotation dimension. As ranking according to various annotation dimensions may lead to different ranking orderings of answers, we formulate the problem of ranking aggregation.

[1] In SPARQL syntax:"SELECT ?*x* WHERE {?*x plays* ?*y* . ?*y hasGenre 'Action'*.}"

ID	Statement	Time	Source	Certainty
#1	[Cinemark Palace plays The Grey]	05.05.12	City Guide	0.8
#2	[Cinemark Palace plays Man on a Ledge]	05.05.12	City Guide	0.8
#3	[Prado Cinema Cafe plays Underworld: Awakening]	06.06.12	Movie Today	0.6
#4	[Deerfield Cinema plays Man on a Ledge]	02.03.12	Cinema On	0.7
#5	[The Grey hasGenre Action]	05.05.12	City Guide	0.7
#6	[Man on a Ledge hasGenre Action]	02.03.12	Cinema On	0.7
#7	[Underworld: Awakening hasGenre Action]	02.03.12	Movie Today	0.7

Table 2. The set of annotated RDF statements in Ex. 5

3.1 Annotated RDF with Multiple Dimensions

Until now, we only consider a single annotation for each RDF statement. However, statements may be annotated along multiple dimensions such as source, time, and certainty. Assume $\Gamma = \{p_1, \ldots, p_\gamma\}$ is a fixed set of *independent annotation dimensions* with $|\Gamma| = \gamma$ and let $K_{p_i} = (\Omega_{p_i}, \otimes_{p_i}, \oplus_{p_i}, \top_{p_i}, \bot_{p_i})$, for $i = 1, \ldots, \gamma$, be the corresponding annotation structures. We extend the definition of an annotated RDF statement accordingly. An annotated RDF statement is a tuple $S = \alpha : \theta_1, \ldots, \theta_\gamma$ with α being an RDF statement and $\theta_i \in \Omega_{p_i}$ an annotation. Similarly, the function $res(G, q)$ is extended to return tuples annotated with multiple annotations, where the annotation value in each dimension is determined separately using the approach from the previous section.

Example 5. We extend the annotations of the set of RDF statements from Table 1 along the independent dimensions $\Gamma = \{source, time, certainty\}$, cf. Table 2. The first row states that the movie theater *Cinemark Palace* plays the film *The Grey*, and that this statement was received from 'City Guide', has been published on '05.05.12' and the certainty value of the statement is 0.8.

Each single annotation dimension of an annotated RDF graph can be exploited for ranking results as discussed above. In general, those rankings do not coincide and depend on the chosen dimension. We are interested in a joint ranking taking *all* annotation dimensions into account.

Example 6. We assume that the RDF statements presented in Table 2 represent all information available on the Web. The query and query results presented in Ex. 3 correspond to the search request and its results respectively. We can exploit the orders $\preceq_p$ to obtain (partial) ranking ordering on the results. Suppose $\preceq_{\text{time}}$ corresponds to the natural linear order on dates, $\preceq_{\text{certainty}}$ corresponds to the order on (fuzzy) certainty values, and for the source dimension, *Movie Today*, *Cinema On*, and *City Guide* represents the set of all the possible values and we assume that *Movie Today* $\preceq_{source}$ *City Guide*, *Cinema On* $\preceq_{source}$ *City Guide* and both *Movie Today*, *Cinema On* are equally reliable. The ranking ordering of the results considering the temporal dimension is $\langle r_3, r_1, r_2 \rangle$, which reflects the recency of the statements those results are based upon. For the certainty dimension we have $\langle r_3, r_2, r_1 \rangle$, which reflects the vagueness of the statements, and for the source dimension we have $\langle (r_2, r_3), r_1 \rangle$, which reflects the reliability of the sources.

3.2 Aggregating Annotation Rankings

In the following, we consider the problem of how results of a query should be ranked according to an aggregation of independent annotation dimensions. Therefore, we wish to aggregate the annotation rankings into a single ranking ordering. This is a well-known problem in the field of *judgement aggregation* (or *preference aggregation*) [2, 15]. There, the problem under consideration is to aggregate interests or votes in order to come up with a decision that is favorable in the light of the individual interests. A well-known application for social choice theory is the problem of *voting*, i. e., of constructing a voting mechanism that is, in some sense, fair. In order to relate our approach to those works, we borrow some properties that were originally stated for social choice theory and investigate them in our framework of aggregating annotation rankings.

Let G be an annotated RDF graph, $\Gamma = \{p_1, \ldots, p_\gamma\}$ a set of annotation dimensions, and q some query for G. As discussed above, querying G with q yields a set of tuples $res(G, q)$, and each tuple is annotated with some annotation value for every annotation dimension in Γ. Using the rankings $\preceq_{p_1}, \ldots, \preceq_{p_\gamma}$ the set $res(G, q)$ can be ordered in different ways. We call $P = \{\preceq_{p_1}, \ldots, \preceq_{p_\gamma}\}$ the *annotation profile* of $res(G, q)$.

Definition 1 (Profile aggregator). *Let $P = \{\preceq_{p_1}, \ldots, \preceq_{p_\gamma}\}$ be an annotation profile of $res(G, q)$. A mapping $\preceq_{p_1}, \ldots, \preceq_{p_\gamma} \mapsto \preceq$, such that $\preceq$ is a partial order on $res(G, q)$, is called an* annotation aggregator.

In the field of judgement aggregation, desirable properties and different mechanisms for defining the aggregation of multiple orderings have been investigated for more than 50 years. One of the most important technical results is *Arrow's impossibility theorem* [2], which states that there is no such thing as a fair voting mechanism or "every voting mechanism is flawed". More precisely, there is no mapping $\mapsto$ satisfying the following three properties (let $\preceq$ be defined via the mapping $\preceq_{p_1}, \ldots, \preceq_{p_\gamma} \mapsto \preceq$):

Pareto-efficiency For every $r, r' \in res(G, q)$, if $r \preceq_{p_i} r'$ for all $i = 1, \ldots, \gamma$ then $r \preceq r'$.
Non-dictatorship There is no $i \in \{1, \ldots, \gamma\}$ such that $\preceq_{p_i} = \preceq$ for *every* profile.
Independence of irrelevant alternatives If for two sets $\{\preceq_{p_1}, \ldots, \preceq_{p_\gamma}\}$ and $\{\preceq'_{p_1}, \ldots, \preceq'_{p_\gamma}\}$ and every $i = 1, \ldots, \gamma$ it holds $r \preceq_{p_i} r'$ whenever $r \preceq'_{p_i} r'$ then $r \preceq r'$ whenever $r \preceq' r'$ (with $\preceq_{p_1}, \ldots, \preceq_{p_\gamma} \mapsto \preceq$ and $\preceq'_{p_1}, \ldots, \preceq'_{p_\gamma} \mapsto \preceq'$).

In the following, we consider three examples of profile aggregators, which are inspired by approaches to solve the problem of judgement aggregation in social choice theory. A simple profile aggregator (that fails to satisfy *non-dictatorship*) is the *lexicographical aggregator*. The lexicographical aggregator assumes some given total order on the annotation dimensions, e. g., the temporal information may be more important than the source information. Given this, a result item r' is preferred to r if r' is preferred to r wrt. the given timestamps ($r \preceq_{time} r'$) or r and r' have equal timestamps ($r \equiv_{time} r'$) but r' is preferred to r wrt. the source information ($r \preceq_{source} r'$). Note, that this kind of annotation aggregator order is heavily biased by the given ordering of dimensions.

Definition 2 (Lexicographical aggregator). *Let $\{\preceq_{p_1}, \ldots, \preceq_{p_\gamma}\}$ be an annotation profile on $res(G, q)$, and assume w.l.o.g. that $\langle \preceq_{p_1}, \ldots, \preceq_{p_\gamma} \rangle$ is the order of importance of*

the annotation profile ($\preceq_{p_1}$ being the most important ranking). Then the lexicographical aggregation $\preceq_l$ *of* $\preceq_{p_1}, \ldots, \preceq_{p_\gamma}$ *is defined via*

$$r \preceq_l r' \;\; \textit{iff} \;\; \neg\exists k \in \{1, \ldots, \gamma\} : \Big(r \not\preceq_{p_k} r' \wedge \forall i \in \{1, \ldots, \gamma\} : i < k \Rightarrow r \preceq_{p_i} r' \wedge r' \preceq_{p_i} r\Big)$$

for all $r, r' \in res$.

Example 7. Given the annotation profiles presented in Ex. 6, and we assume that $\langle \preceq_{certainty}, \preceq_{time}, \preceq_{source} \rangle$ is the order of importance of the annotation profiles. Then the lexicographic ordering of r_1, r_2, and r_3 corresponds to $r_3 \preceq_l r_2 \preceq_l r_1$. This represents exactly the ranking ordering when considering just the certainty dimension. Now, we assume that $\langle \preceq_{source}, \preceq_{time}, \preceq_{certainty} \rangle$ is the order of importance of the annotation profiles, then r_1, r_2, and r_3 are ordered in $r_2 \preceq_l r_3 \preceq_l r_1$.

Another popular aggregation method from voting theory is the *Borda rule*. The Borda rule defines preferences by assigning a value to each tuple according to its positions in the domain order. The sum of all the values of a tuple represents its score.

Definition 3 (Borda aggregator). *Let* $\{\preceq_{p_1}, \ldots, \preceq_{p_\gamma}\}$ *be an annotation profile on* $res(G, q)$. *For each* $r \in res(G, q)$ *define* $sc(r)$ *via*[2]

$$sc(r) = \sum_j \sum_{r' \in res(G,q) \setminus \{r\}} [r' \preceq_{p_j} r]$$

for all $r \in res(G, q)$. *Then the* Borda aggregation $\preceq_b$ *of* $\preceq_{p_1}, \ldots, \preceq_{p_\gamma}$ *is defined via*

$$r \preceq_b r' \quad \textit{iff} \quad sc(r) \leq sc(r')$$

for all $r, r' \in res$.

Example 8. Given the annotation profiles presented in Ex. 6, then the Borda ordering corresponds to $r_2 \preceq_b r_3 \preceq_b r_1$. The Borda score of r_1 is 5 (at the temporal dimension score 1, at the certainty dimension score 2, and at the source dimension 2, since r_1 is derived from the most reliable source). The Borda score of r_3 is 3 (2, 0, and 1 at temporal, certainty, and source dimension respectively) and of r_2 is 2 (0, 1, and 1 at temporal, certainty, and source dimension).

At last, we consider the *plurality* aggregation method, which defines the preferred element to be the tuple in the result set with the most votes (plurality), i. e., the tuple r which is the most preferred considering all annotation profiles.

Definition 4 (Plurality aggregator). *Let* $\preceq_{p_1}, \ldots, \preceq_{p_\gamma}$ *be an annotation profile on* $res(G, q)$. *Then the* plurality aggregation $\preceq_m$ *of* $\preceq_{p_1}, \ldots, \preceq_{p_\gamma}$ *is defined via*

$$r \preceq_m r' \quad \textit{iff} \quad |\{i \mid r \preceq_{p_i} r'\}| \geq |\{i \mid r' \preceq_{p_i} r\}|$$

for all $r, r' \in res$.

[2] Note that $[A]$ is the Iverson bracket defined via $[A] = 1$ if A is true and $[A] = 0$ otherwise.

Note that the plurality aggregator defined above suffers from the *Condorcet paradox* if more than two rankings are to be aggregated [10]. That is, even if $\preceq_1, \ldots, \preceq_\gamma$ are partial orders, the relation $\preceq_m$ may contain cycles.

Example 9. Let $res = \{r_1, r_2, r_3\}$ and consider $\preceq_1$, $\preceq_2$, and $\preceq_3$ defined via

$$r_1 \preceq_1 r_2 \preceq_1 r_3 \qquad r_2 \preceq_2 r_3 \preceq_2 r_1 \qquad r_3 \preceq_3 r_1 \preceq_3 r_2$$

and observe that $r_1 \preceq_m r_2$, $r_2 \preceq_m r_3$, and $r_3 \preceq_m r_1$.

Example 10. Given the annotation profiles of the certainty and source dimensions presented in Ex. 6, then the plurality ordering for r_1, r_2, and r_3 corresponds to $r_2 \preceq_b r_3 \preceq_b r_1$ since $r_2 \preceq_p r_1$ and $r_3 \preceq_p r_1$ for all the dimensions and $r_2 \preceq_{certainty} r_3$, and $r_2 \equiv_{source} r_3$.

4 Ranking of Stream Data

The proposed profile aggregators can be applied in offline and online settings. Offline setting means that the whole set of results is completely available, while online setting refers to the aggregation of streaming data. In the following, both settings are extensively discussed.

4.1 Offline Setting

First, we perform top-k querying on locally stored annotated RDF graphs rather than on-the-fly, i. e., the whole result set $res(G, q)$ of a query q is known a priori. As one or multiple annotation dimensions can be used for ranking, in the following, we shortly describe how ranking on annotated RDF can be done:

One-dimensional Approach. Providing top-k results when considering only one annotation dimension, for instance the temporal dimension, is done straightforward by sorting the results in order of importance (increasing or decreasing order).

Multi-dimensional Approach. A top-k ranking over multiple annotation dimensions requires an aggregation of the top-k rankings from each dimension. According to Sec. 3.2, the profile aggregators offer a variety of aggregation means with respect to individual preferences among the different annotation dimensions. Therefore, to obtain a joint ranking taking *all* annotation domains into account, we use one of the described annotation aggregators (e. g., the lexicographical rule, the Borda rule, or the plurality rule) to aggregate results from multiple annotation dimensions into a single ordering.

4.2 Online Setting

In contrast to offline ranking, in stream ranking $res(G, q)$ is not available at once, we rather receive the result tuples $r \in res(G, q)$ continuously whereas r_t is the tuple received at time t. We want to start presenting results up the first available tuple. Such a stream ranking mechanism could not decide upon receipt if a result tuple r_t is part of final the top-k results. It could only decide if a tuple r_t is part of the top-k results at time t because there could be better results not received at time t.

One-dimensional Approach. A simple approach for ranking over streams starts with an empty top-k result set res_k. New result tuples r_t are added to res_t until the result set is full (k tuples in res_k). We suppose that the k tuples are sorted wrt. a given criteria, for instance, in decreasing order. Let r_b, denoted as the border-tuple, be the lowest/smallest (the k-th) tuple. All tuples r_t received from this point have to be compared with the border-tuple r_b. If $r_t \preceq r_b$, the tuple r_t could be ignored otherwise r_b is removed and r_t is added to res_k. In this case, a new border-tuple has to be computed.
Multi-dimensional Approach. For the aggregation over multiple dimensions, the streaming algorithm remains the same if the ordering of the elements in res_k does not change when additional elements are received, i.e., if the *independence of irrelevant alternatives* holds. This property holds for several aggregators, e.g., for the lexicographic aggregator. The Borda aggregator, however, is an example where is is not possible to determine an aggregation without considering the whole set of results.

Example 11. In Ex. 8 $\langle r_1, r_3, r_2 \rangle$ corresponds to the top-3 result with respect to the Borda aggregator. As already mentioned, pairwise element comparison is not applicable to the Borda aggregator. To illustrate that, let us assume that we are looking for the top-2 results and that the elements are received in the following order, r_1, r_2, r_3. The top-2 results should be $r_3 \preceq_b r_1$. Like in the one-dimensional case, we fill our result set with the first two elements, i.e., $\{r_1, r_2\}$. Then we compare these elements to determine the border element to be r_2. Next, we compare this border element to our last element r_3. Since, it ranks r_2 and r_3 as equal, we are not able to determine if we have to replace the border element r_2 with r_3 or keep it. It is not possible to determine our top-2 result precisely, and only the top-2 result sets $\{r_1, r_2\}$ and $\{r_1, r_3\}$.

5 Related Work

Ranking of query results has been considered from different perspectives in the database research. In Agrawal et al. [1] the ranking criteria is based on a similarity measure between terms in the query condition and terms of tuple attributes in tables. A similar principle is applied by Fuhr [9], where ranking techniques from information retrieval are used to rank query results on databases. This approach relies on a relevance feedback from the user. Ilyas et al. [14, 13] give an overview of rank-aware join operations. While in the top-k query result ranking the results are ranked according to attribute values of result tuples, e.g,. by using order-by construct on attributes, the top-k join ranking specifies ranking scores based on multiple relations. Natsev et al. [20]. propose a a heuristic algorithm over multiple ranked data sources to efficient combining ranked results from single dimensions using. None of these approaches rank over RDF data and they do not consider the particular problem of aggregation of different independent ranking dimensions.

From the ranking perspective, several approaches consider also the problem of combining several dimensions of ranking criteria [6, 16, 23, 12]. Preferences are specified in terms of partial orders on attributes and they can be accumulated and aggregated to build complex preference orders. In [6], the general idea is to rank query results when there is no exact match, but some results are similar to the query. They compute the

distance of attribute values of the relation with respect to the query attributes. In [12], linear sums of attributes are used to rank preferences (assigned to attributes). Likewise, Li et al. [18] presents top-k ranking for different dimensions for relational databases. Compared to our work, none of them considers the ranking of semi-structured data like RDF and their focus is not on ranking w.r.t. annotations of data.

In the realm of the Semantic Web, we compare our work to annotated RDF data in general, and to aggregation principles in particular. Based on semirings [11], Buneman and Kostylev [7] and Straccia et al. [22] present an algebra for RDF annotations. Their approaches are for annotations in general, but they do not consider multiple dimensions simultaneously. Zimmermann et al. [24] present a combination of multiple annotation dimensions. They combine two dimensions by a composition into one dimension, modeled as a compounded annotation dimension. An aggregation function maps annotation dimensions into a set of pairs of annotations. Kostylev et al. [17] also consider the problem of combining various forms of provenance that, analogous to the previous, map annotation dimensions into a set of vectors. They introduce restrictions to semirings in order to define containment and equivalence of combined annotation relations. The latter ones are different from our work since we aggregate annotation dimensions considering aggregation functions, which does not rely on the structure of the annotation dimensions and can be generalized to the aggregation of every ordered set.

Ranking for streaming data is an emerging problem when querying large data collections as on the Web, and it is therefore considered for RDF querying and reasoning. For instance, SPARQL extensions allows ranking query results on streams [4, 5], as well as general reasoning frameworks incorporate the management of streaming data [3]. However, ranking with aggregation of multiple annotation dimensions is not studied on streaming RDF data so far.

6 Summary

We have presented a novel approach for top-k querying of RDF data with multiple provenance dimensions. RDF data is annotated with values providing knowledge on the origin, truthworthiness or validity of data and this knowledge should be taken into account when answering queries. Top-k ranking becomes complicated if data have multiple provenance dimensions, and the ranking has to incorporate a holistic ordering over all these dimensions. We have presented an aggregation approach over multiple provenance dimensions, which is based on preference aggregation in social choice theory. We have formalized three different aggregation methods, namely the lexicographical, the Borda and the plurality rule.

Additionally, we consider these aggregations in offline and online settings. For online settings, we first elaborate how the aggregators deal with a continuously enlargement of the available result tuples that are considered in the aggregation. Secondly, we explain which aggregators can be applied in online settings and which not. Further investigations and implementations for efficient ranking on streaming data are planned.

Acknowledgments The research reported here was partially supported by the SocialSensor FP7 project (EC under contract number 287975).

References

1. S. Agrawal, S. Chaudhuri, G. Das, and A. Gionis. Automated Ranking of Database Query Results. In *CIDR*, 2003.
2. K. J. Arrow. A Difficulty in the Concept of Social Welfare. *Journal of Political Economy*, 58(4):328–346, 1950.
3. D. F. Barbieri, D. Braga, S. Ceri, E. Della Valle, and M. Grossniklaus. Incremental Reasoning on Streams and Rich Background Knowledge. In *ESWC*, pages 1–15, 2010.
4. D.F. Barbieri, D. Braga, S. Ceri, and M. Grossniklaus. An Execution Environment for C-SPARQL Queries. In *EDBT*, pages 441–452. ACM, 2010.
5. A. Bolles, M. Grawunder, and J. Jacobi. Streaming SPARQL-extending SPARQL to process data streams. *The Semantic Web: Research and Applications*, pages 448–462, 2008.
6. N. Bruno, S. Chaudhuri, and L. Gravano. Top-k Selection Queries over Relational Databases: Mapping Strategies and Performance Evaluation. *ACM Trans. Database Syst.*, 27(2):153–187, 2002.
7. P. Buneman and E. Kostylev. Annotation algebras for RDFS. In *SWPM*. CEUR Workshop Proceedings, 2010.
8. R. Dividino, S. Sizov, S. Staab, and B. Schueler. Querying for provenance, trust, uncertainty and other meta knowledge in rdf. *JWS*, 7(3):204–219, September 2009.
9. N. Fuhr. A Probabilistic Framework for Vague Queries and Imprecise Information in Databases. In *VLDB*, pages 696–707, 1990.
10. W. V. Gehrlein. *Condorcet's Paradox*. Theory and Decision Library C , Vol. 40. Springer, Berlin, Heidelberg, 2006.
11. T. J. Green, G. Karvounarakis, and V. Tannen. Provenance Semirings. In *PODS*, pages 31–40, 2007.
12. V. Hristidis, N. Koudas, and Y. Papakonstantinou. PREFER: A System for the Efficient Execution of Multi-parametric Ranked Queries. In *SIGMOD*, pages 259–270, 2001.
13. I. F. Ilyas, G. Beskales, and M. A. Soliman. A Survey of Top-k Query Processing Techniques in Relational Database Systems. *ACM Comput. Surv.*, 40(4):11:1–11:58, October 2008.
14. I. F. Ilyas, R. Shah, W. G. Aref, J. Scott Vitter, and A. K. Elmagarmid. Rank-aware Query Optimization. In *SIGMOD*, pages 203–214, 2004.
15. J. Kelly. *Social Choice Theory: An Introduction*. Springer-Verlag, 1988.
16. W. Kießling. Foundations of Preferences in Database Systems. In *VLDB*, pages 311–322, 2002.
17. E. V. Kostylev and P. Buneman. Combining dependent annotations for relational algebra. In *ICDT*, 2012.
18. C. Li, K. Chen-Chuan Chang, I. F. Ilyas, and S. Song. RankSQL: Query Algebra and Optimization for Relational Top-k Queries. In *SIGMOD*, pages 131–142, 2005.
19. N. Lopes, A. Polleres, U. Straccia, and A. Zimmermann. Anql: Sparqling up annotated rdfs. In *ISWC*, pages 518–533, Berlin, Heidelberg, 2010. Springer-Verlag.
20. Apostol N., Y.-C. Chang, J. R. Smith, C.-S. Li, and J. S. Vitter. Supporting Incremental Join Queries on Ranked Inputs. In *VLDB*, pages 281–290, 2001.
21. E. Prud'hommeaux and A. Seaborne. Sparql query language for rdf. W3c recommendation, W3C, January 2008.
22. U. Straccia, N. Lopes, G. Lukácsy, and A. Polleres. A General Framework for Representing and Reasoning with Annotated Semantic Web Data. In *AAAI*, pages 1437–1442, 2010.
23. D. Xin, J. Han, H. Cheng, and X. Li. Answering Top-k Queries with Multi-Dimensional Selections: The Ranking Cube Approach. In *VLDB*, pages 463–475, 2006.
24. A. Zimmermann, N. Lopes, A. Polleres, and U. Straccia. A general framework for representing, reasoning and querying with annotated semantic web data. *JWS*, 11(0):72 – 95, 2012.