Fixing Comparative Preferences for SPARQL*

Peter F. Patel-Schneider[1], Axel Polleres[2,3], and David Martin[1]

[1] NAIL Laboratory, Nuance Communications, Sunnyvale, CA, USA
[2] Vienna Univ. of Economics & Business / Complexity Science Hub Vienna, Austria
[3] Stanford University, CA, USA**

Preferences have been part of the goal of the Semantic Web from its inception [1] but are not currently part of any Semantic Web standards, such as SPARQL.

Several proposals [2–4] have been made to add comparative preferences to SPARQL. Comparative preferences are based on comparing solutions to a query and eliminating ones that come out worse in the comparison, as in searching for gas stations and eliminating any for which there is a closer station serving the same brand of gasoline. The proposals each add an extra construct to SPARQL filtering out non-preferred query solutions. Their preference constructs are of different expressive power but they can each be thought of as providing a skyine operator, defined as follows:

Definition 1. *Given a set of (potential) solutions P, a preference relation $\succ$ is any relation over $P \times P$. A solution s_1 is* dominated by *a solution s_2 if $s_2 \succ s_1$.*

Definition 2. *(Adapted from Chominki [5].) If S is a finite set of (candidate) solutions and $\succ$ a preference relation over some superset of S then the* skyline *(the preferred solutions) of S with respect to $\succ$ is*

$$\omega_{\succ}(S) = \{s \in S | \neg \exists s' \in S . s' \succ s\}$$

That is, a solution is retained only if there is no solution that dominates it according to the preference relation. So to prefer the closest gas station for each brand, the preference relation would state that one gas station dominates a second if they both serve the same brand of gas and the first is closer than the second.

SPREFQL

The most general of the proposals to add preferences to SPARQL is SPREFQL [4], where the preference relation can be any SPARQL expression.

A query in SPREFQL preferring the closest gas station for each brand would be

```
SELECT ?X ?B ?D
WHERE { ?X a :GasStation ; :brand ?B ; :dist ?D . }
PREFER ?X1 ?B1 ?D1 TO ?X2 ?B2 ?D2
       IF ?B1 = ?B2 && ?D1 < ?D2
```

* For technical details, including formal statements of all the theorems and their proofs see the technical report available at http://polleres.net/publications/patel-schneider-etal-2018TR.pdf

** Axel Polleres' work was supported under the Distinguished Visiting Austrian Chair Professors program hosted by The Europe Center of Stanford University.

The SPARQL expression in the PREFER construct has access to two candidate solution mappings through the two lists of variables in the PREFER construct. A candidate solution mapping S_1 dominates candidate solution mapping S_2 if the expression in the PREFER construct evaluates to true when the first list of variables in the PREFER are bound using S_1 and the second are bound using S_2. The PREFER construct selects those solution mappings (hereafter just solution) that have no dominating solution, which in this example means that the preferred solutions are those for which there is no solution with the same brand and a smaller distance.

Special macros are provided in SPREFQL for preferences combining multiple preferences, including the conjunction of two preferences (e.g., prefer closer and cheaper gas stations) and the cascading of preferences (e.g., prefer cheaper gas stations but among those with the same price prefer the closer ones).

Troumpoukis et al. [4] provide a mapping from their extension back into SPARQL 1.1, mapping

SELECT V WHERE { P } PREFER V_1 TO V_2 IF C

to

SELECT V WHERE { P FILTER NOT EXISTS {$P_{(V/V_1)}$ FILTER $C_{(V_2/V)}$} }

Unfortunately this mapping is not correct because of problems in the definition of EXISTS in SPARQL [6]. Certain constructs occuring in C, such as BOUND, are handled incorrectly inside of EXISTS in SPARQL. Fortunately there is a simple fix for this problem, namely using a well-known replacement for EXISTS, already proposed by Gueroussova et al. [3] as a mapping from their preferences into SPARQL 1.0. The mapping below uses a variant of their mapping.

Mapping 1 (Simple Mapping to SPARQL)

```
SELECT V WHERE { P
      OPTIONAL { P_(V/V1) FILTER ( C_(V2/V) ) BIND 1 TO ?exists }
      FILTER (!BOUND(?exists)) }
```

The OPTIONAL *part only binds a value to* ?exists *when a dominator exists so the final filter only lets through solutions that are not dominated.*

Troumpoukis et al. [4] only consider the solutions in the skyline as defined above. It can be useful to define a second skyline as the skyline after the first skyline is removed, and so on. The rank of a solution can then be defined as the number of the skyline it is in. A simple extension to SPREFQL, adding LIMIT SKYLINE n [AS v_{rank}] and LIMIT SKYLINE ALL AS v_{rank} modifiers to the PREFER construct, could produce the first n skylines and optionally the rank of each solution or all solutions with their rank.

Preference Cycles

There is a very serious problem in SPREFQL, however. Any solution that is in a preference cycle, i.e., the solution is in a cycle of solutions where each one dominates the next, will not be a preferred solution, even if the cycle is just the solution dominating itself. Not only will such solutions not be in the first skyline, they will not be in *any* skyline, nor will any solution that they transitively dominate.

Some comparative preference languages, such as PrefSPARQL [3], are built up from metrics (like closest distance) and cannot have preference cycles. However, in SPREFQL it is easy to construct preference cycles. Simply preferring a particular brand, as in the following SPREFQL query

```
SELECT ?X ?B ?D
WHERE { ?X a :GasStation ; :brand ?B ; :dist ?D . }
PREFER ?X1 ?B1 ?D1 TO ?X2 ?B2 ?D2 IF ?B1 = :Mobil
```

results in any solution for brand Mobil being preferred to itself and if there are any solutions with brand Mobil there will be no preferred solutions at all.

It is easy for users to provide preferences that produce cycles for certain data,[4] For example having preferences about brands associated with gas stations, as in

```
SELECT ?X ?B ?F ?T
WHERE {?X a :GasStation; :brand ?B; :shop ?F; :antifreeze ?A.}
PREFER ?X1 ?B1 ?F1 ?A1 TO ?X2 ?B2 ?F2 ?A2
IF ( (?B1 = :Mobil && ?B2 = :Chevron) AND
     (?F1 = :KwikieMart && ?F2 = :711) AND
     (?A1 = :Xerex && ?A2 = :Prestone) )
```

results in a preference cycle if there is a Mobil station with a TigerMart that sells Prestone antifreeze; a Chevron station with a KwikieMart that sells Delo antifreeze; and a gas station with a 7-11 that sells Xerex antifreeze.

To handle preference cycles, it is necessary to change the definition of skyline so that a preferred solution is no longer just one with no dominating solution. Instead a preferred solution is one where every solution that transitively dominates it is also transitively dominated by it (i.e., is in a preference cycle with it):

Definition 3. *If S is a finite set of solutions and $\succ$ a preference relation over some superset of S then the* skyline *of S with $\succ$ is*

$$\omega^{l}_{\succ}(s) = \{s \in S | \neg \exists s' \in S.(s' \succ^{*} s \wedge \neg(s \succ^{*} s'))\}$$

where $\succ^{*}$ is the transitive closure of $\succ$ over candidate *solutions, i.e., there is a sequence of candidate solutions, each dominating the next.*

This simple change means that there are always preferred solutions in any non-empty, finite set of solutions and that every solution in a finite set of solutions has a finite rank. This repairs the problem in SPREFQL at the expense of complicating the definition of preferred solutions.

This complication matters, as there is no mapping to a single SPARQL query for the correct definition of skyline, as the query would have to examine all the candidate solutions looking for preference cycles. If, however, multiple queries are allowed it is possible to use two `CONSTRUCT` queries to create the preference relation in an RDF graph and then a `SELECT` query to return the preferred solutions from this graph.

[4] We do not argue here that such preferences are rational, but simply observe that people have preferences of these forms.

Constructing the domination relation in this way would be quite inefficient, but it is possible to implement this correct definition of skylines as an extension of SPARQL by using a simple algorithm that actually computes the rank of each solution. The core of the algorithm collapses preference cycles into a single representative using the union-find algorithm [7] resulting in cheap determination of whether a solution is in a preference cycle with another.

In the worst case this algorithm requires $O(n^2)$ time (to compute the domination relationships between each pair of candidate solutions), which is no worse than the algorithm required for SPREFQL. There is a difference in expected time, however, as the correct algorithm cannot stop considering a solution as soon as it finds a dominator of it but instead has to also check whether this dominator is in a cycle.

Transitive Preferences

If the preference relation is transitive, but still can have loops, then there is a simpler skyline definition that is correct:

Definition 4. *If S is a finite set of (candidate) solutions and $\succ$ a transitive preference relation over some superset of S then the* skyline *of S with $\succ$ is*

$$\omega^t_{\succ}(S) = \{s \in S | \neg \exists s' \in S.(s' \succ s \wedge \neg(s \succ s'))\}$$

This definition exploits the fact that the transitive closure of a transitive relation is the relation itself. There is a mapping to a single SPARQL query for this definition of skyline, that is only a slight modification of Mapping 1.

With these corrections SPREFQL is well behaved on any preference relation. As well, the useful notion of the rank of a solution under a preference relation can be defined as an extension to SPREFQL. The corrected version cannot always be translated to a single SPARQL query but there is still an effective algorithm for computing the rank of each solution in a solution set under a preference relation.

References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American **284**(5) (2001) 28–37
2. Siberski, W., Pan, J.Z., Thaden, U.: Querying the semantic web with preferences. In: The 5th International Semantic Web Conference (ISWC 2006). (2006) 612–624
3. Gueroussova, M., Polleres, A., McIlraith, S.A.: SPARQL with qualitative and quantitative preferences. In: 2nd International Workshop on Ordering and Reasoning (OrdRing 2013), at ISWC 2013. (2013) CEUR Workshop Proceedings, Volume 1059.
4. Troumpoukis, A., Konstantopoulos, S., Charalambidis, A.: An extension of SPARQL for expressing qualitative preferences. In: The 16th International Semantic Web Conference (ISWC 2017). (2017) 711–727
5. Chomicki, J.: Preference formulas in relational queries. ACM Transactions on Database Systems **28**(4) (2003) 427–466
6. Patel-Schneider, P.F., Martin, D.: EXISTStential aspects of SPARQL. In: The 15th International Semantic Web Conference (ISWC 2016). (October 2016)
7. Tarjan, R.E.: Efficiency of a good but not linear set union algorithm. Journal of the ACM **22**(2) (1975) 215–225