Selectivity Estimation for SPARQL Graph Pattern

Hai Huang
Faculty of ICT
Swinburne University of Technology
hhuang@swin.edu.au

Chengfei Liu
Faculty of ICT
Swinburne University of Technology
cliu@swin.edu.au

ABSTRACT

This paper focuses on selectivity estimation for SPARQL graph patterns, which is crucial to RDF query optimization. The previous work takes the join uniformity assumption, which would lead to high inaccurate estimation in the cases where properties in SPARQL graph patterns are correlated. We take into account the dependencies among properties in SPARQL graph patterns and propose a more accurate estimation model. We first focus on two common SPARQL graph patterns (star and chain patterns) and propose to use Bayesian network and chain histogram for estimating the selectivity of them. Then, for an arbitrary composite SPARQL graph pattern, we maximally combines the results of the star and chain patterns we have precomputed. The experiments show that our method outperforms existing approaches in accuracy.

Categories and Subject Descriptors

H.2.4 [**Database Management**]: Systems—*query processing*

General Terms

Algorithms, Performance

Keywords

Selectivity Estimation, RDF Query Processing

1. INTRODUCTION

Since the use of RDF to represent data has grown dramatically over the last few years, query processing on RDF data becomes an important issue. Selectivity estimation for SPARQL graph patterns is crucial to RDF query processing. As we know, RDF data is a set of triples with the form (*subject*, *property*, *object*). This fine-grained model leads to SPARQL queries on RDF data with a large number of joins. As such, precise estimation of the selectivity of joined triple patterns is very important. In [1, 2] the *join uniformity assumption* is made when estimating the selectivity of joined triple patterns, which assumes that each triple satisfying a triple pattern is equally likely to join with the triples satisfying the other triple pattern. However, this assumption does not hold in many cases.

WWW 2010, April 26–30, 2010, Raleigh, North Carolina, USA.
ACM 978-1-60558-799-8/10/04.

2. ESTIMATION FOR STAR PATTERNS USING BAYESIAN NETWORKS

The *star* graph pattern is common in SPARQL graph patterns. It has the form of a number of triple patterns with different properties sharing the same subject (an example is shown in Figure 1). For estimating the selectivity of frequent star patterns, we construct the cluster-property table R for each one.

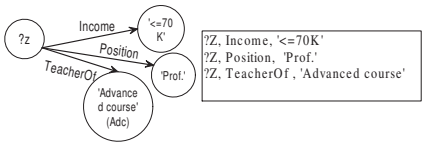

Figure 1: Star-style graph pattern

Given a frequent star pattern Q with predicates $prop_1$, $prop_2$, $\cdots$, $prop_n$, if we know the joint probability distribution over values of properties $\Pr(prop_1 = o_1,\ prop_2 = o_2,\ \cdots,\ prop_n = o_n)$ in R, we can easily obtain the selectivity $sel(Q)$ of Q as: $sel(Q) = \Pr(prop_1 = o_1, prop_2 = o_2, ..., prop_n = o_n) \cdot |R|$, where $|R|$ is the number of rows in R. However, it is impossible to explicitly store the joint probability distribution over property values in R, since the possible combinations of values of properties could be exponential. We employ Bayesian network to *approximately* store the joint probability distribution information. Bayesian networks make use of Bayes' Rule and conditional independence assumption to compactly represent the full joint probability distribution using a little space. Given a star pattern Q and Bayesian network β learned from table R, we have:

$$\begin{aligned} sel(Q) &= \Pr(prop_1 = o_1, prop_2 = o_2, ..., prop_n = o_n) \cdot |R| \\ &\approx Pr_\beta(prop_1 = o_1, prop_2 = o_2, ..., prop_n = o_n) \cdot |R| \\ &= \prod_{i=1}^{n} \Pr(prop_i = o_i \mid Parents(prop_i) = \vec{o_k}) \cdot |R| \end{aligned}$$

where $Parents(prop_i)$ denotes the set of immediate predecessors of $prop_i$ in the Bayesian network; $\vec{o_k}$ denotes the set of values of $Parents(prop_i)$. Note that for computing $\Pr(prop_i \mid parents(prop_i) = \vec{o_k})$, we only need to know the values of $prop_i$'s parent properties, which would save a lot of space in practice.

3. ESTIMATION FOR CHAIN PATTERNS

The *chain* graph pattern is another kind of common SPARQL query patterns, which consists a sequence of triple patterns

where the object of the previous triple pattern is also the subject of the next pattern. We construct the chain count table TC (shown in Figure 2) for frequent chain patterns, which has two attributes: *Head-Chain-Rear* and *Count*. Each row of TC indicates a chain pattern with their frequencies (selectivities). However, chain table TC could be too large.

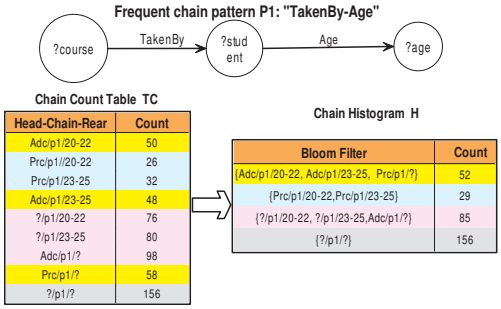

Figure 2: An example of the chain count table and the chain histogram. "?" indicates a variable.

Thus, we group the chain patterns in TC into several buckets according to their frequencies. And for each bucket we only need to save the average frequency and its chain pattern members. So given a chain pattern and which bucket it belongs to, we can easily get the frequency of this chain pattern. For efficient processing the membership queries, we use bloom filter, a space-efficient probabilistic data structure often used to test whether an element is a member of a set. Here, we use bloom filter to test whether a chain pattern is a member of a bucket.

4. ESTIMATION FOR COMPOSITE GRAPH PATTERNS

To estimate the selectivity of a composite SPARQL graph pattern Q, we propose to maximally use the statistics of the sub star and chain patterns we have precomputed to obtain the overall selectivity of Q. We wish to find the maximum precomputed pattern cover of Q and process the uncovered part of Q with independence assumption. Based on *dynamic programming*, we can get an optimal algorithm for finding the maximum pattern cover of Q.

For a composite graph pattern decomposed the precomputed patterns, we need to combine the selectivity of precomputed star and chain patterns. There are three basic cases: **Case 1 (star-chain join)**: The composite graph pattern Q can be decomposed into a precomputed star pattern S and a chain pattern C joined on a variable Y. **Case 2 (star-star join)**: Q can be decomposed into two precomputed star patterns joined on a variable Y. **Case 3 (chain-chain join)**: Q can be decomposed into two precomputed chain patterns joined on a variable Y.

For these cases, we go through all values of the join node Y. We can acquire the selectivity of star pattern S with different values on Y through inference on the Bayesian network. Similarly, we can obtain the selectivity of chain pattern C through the chain histogram. If two patterns have the same value on the join node Y, we combine the selectivity of two patterns in the product form. For the case where a graph pattern Q can be decomposed into multiple patterns, we select two joined patterns from Q and compute the selectivity of the joined patterns. Iterate this process until the overall selectivity of Q is obtained.

5. EXPERIMENTS

We run all algorithms on a windows XP system with 3G CPU and 2 GB RAM. We use the data set LUBM in our experiment and we generate 600k distinct triples; We compare our method with two other methods **PF** and **RDF-3X** proposed in [1] and [2] :

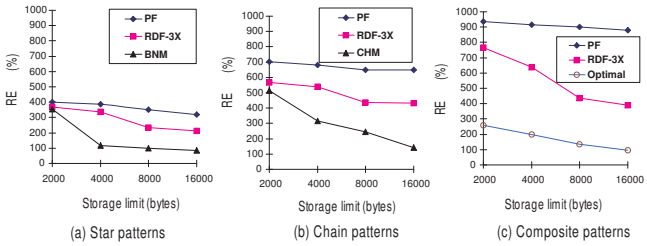

Figure 3: Performance of our method.

Figure 3 shows the accuracy of three methods for the queries on the LUBM dataset. In all figures, X-axis is the space limit and Y-axis is the average relative error RE($RE = \frac{|sel-\widetilde{sel}|}{max(1,sel)}$). We first develop 50 star queries and 50 chain queries respectively and vary the the space limit from 2K bytes to 16K bytes for storing CPTs and chain histogram for star and chain patterns. Figure 3(a), (b) show the performance of three methods. "BNM" and "CHM" indicate our Bayesian network and chain histogram based methods for star and chain patterns. Our method outperforms the other methods. Then we develop 50 composite query patterns. All these query patterns can be decomposed into star patterns and chain patterns we have precomputed. Figure 3(c) shows the results of three methods, where "Optimal" stands for the optimal decomposition algorithm. Our method obtains more accurate estimations since we construct the refined model when dealing with joined triple patterns and do not adopt join uniformity assumption.

6. CONCLUSION

In this paper, we construct the Bayesian networks and chain histogram for estimating the selectivity of star and chain patterns. For an composite graph pattern, we combine the results of precomputed chain patterns and star patterns to estimate the overall selectivity. Experiments demonstrate the effectiveness of our method.

7. ACKNOWLEDGMENTS

This work was supported partly by the Australian Research Council Discovery Project under the grant number DP0878405.

8. REFERENCES

[1] M. Stocker, A. Seaborne, A. Bernstein, C. Kiefer: SPARQL basic graph pattern optimization using selectivity estimation. In WWW, pages:595-604, 2008.

[2] T. Neumann, G. Weikum: RDF-3X: a RISC-style engine for RDF. PVLDB 1(1): 647-659, 2008.