SPARQL-to-SQL 변환 알고리즘의 저장소 독립적 활용을 위한 시스템 모델

손지성[1] 정동원[O2] 김진형[1] 백두권[1]
고려대학교 컴퓨터학과[1] 국립군산대학교 정보통계학과[2]
{redfunky07, koolmania, baikdk}@korea.ac.kr[1], djeong@kunsan.ac.kr[O2]

System Model for Storage-Independent Use of SPARQL-to-SQL Translation Algorithm

Jiseong Son[1] Dongwon Jeong[O2] Jinhyung Kim[1] Doo-Kwon Baik[1]
Dept. of Computer Science and Engineering, Korea University[1]
Dept. of Informatics & Statistics, Kunsan National University[2]

1. 서론

시맨틱 웹[1]과 웹 온톨로지에 대한 연구가 활발해지면서 온톨로지 생성, 저장 및 활용을 위한 많은 시스템 및 도구들이 개발되고 있다 [2,3,4,5,6,7]. 특히 대부분의 웹 온톨로지 저장소들이 관계형 데이터베이스를 이용하고 있고 온톨로지 질의 언어인 SPARQL[8]의 활용성이 높아짐에 따라 SPARQL을 SQL로 변환하는 알고리즘 개발의 필요성이 대두되었다. 지금까지 Chebotko 알고리즘[9], Jena의 sparql2sql[10], Harris 알고리즘[11] 등과 같은 다수의 변환 알고리즘이 제안되었으나 SPARQL 일부만을 SQL로 변환하는 문제점을 지닌다. 또한 제안 알고리즘들이 특정한 저장 모델을 전제로 개발되었기 때문에 변환 알고리즘이 저장소 구조에 종속적이다. 이는 변환 알고리즘의 활용성을 저하시키고 다른 구조를 지니는 저장소에 이용하고자 할 때 추가적인 개발 비용을 요구한다. 이 논문에서는 앞서 언급한 문제점들 중에서, 저장소의 구조에 따라 변환 알고리즘을 수정해야 하는 문제점을 해결하고 특정 변환 알고리즘을 다양한 저장소에 활용할 수 있는 모델을 제안한다. 제안 모델을 위한 프로토타입 시스템 구조 설계 및 구현 결과에 대하여 기술하고 마지막으로 실험 결과 및 비교 평가 결과에 대하여 기술한다.

2. 본론

이 논문에서는 저장소와 변환 알고리즘 간의 독립성을 유지시키는 접근 방법을 통해 변환 알고리즘을 다양한 저장소에 활용할 수 있는 모델을 제안한다. 이를 위해 알고리즘과 저장소 사이에 가상테이블인 뷰를 생성한다. 뷰는 각각 다른 형태를 가지는 저장소와 동일한 물리적 공간에 위치하며 뷰의 구조는 변환 알고리즘이 사용하는 저장 구조 형태로 생성된다. 뷰의 저장 구조는 알고리즘이 사용하는 저장 구조에 따라서 변경해야 하나 뷰를 이용함으로써 변환 알고리즘과 저장소와의 독립성을 유지할 수 있다.

지금까지의 제안된 기존 변환 알고리즘들 중에서 Chebotko 알고리즘이 가장 높은 기능성을 제공하는 것으로 판단되며, 이 논문에서는 제안 모델을 위한 프로토타입 구현 및 실험을 위해 이 변환 알고리즘을 이용한다. Chebotko 알고리즘은 트리플 형태 (Subject, Predicate, Object)의 단일 테이블을 지니는 저장 구조에 데이터를 저장한다는 전제 조건을 지닌다. 따라서 다른 구조를 지니는 저장소의 활용을 위해 생성되는 뷰 역시 단일 트리플 구조로 정의되어야 함을 의미한다. 물리적 저장소와 변환 알고리즘을 독립적으로 유지시키기 위해 물리적 계층과 SQL 계층 사이에 뷰를 생성함으로써 변환 알고리즘의 수정 없이 관계형 데이터베이스 정보 검색이 가능하다.

앞서 언급한 개념과 접근 방법에 따라, 이 논문에서는 SPARQL 계층, Algorithm 계층, SQL 계층, View 계층, Physical 계층 등 총 5개의 계층으로 구성된 프레임워크를 정의한다. SPARQL 계층에서, 사용자는 검색을 위해 SPARQL 질의문을 입력한다. Algorithm 계층에는 Chebotko 알고리즘과 같은 변환 알고리즘들로 구성된다. 이 알고리즘들은 사용자의 입력으로 주어진 SPARQL을 SQL로 변환하는 역할을 담당한다. 이 계층에 임의의 알고리즘이 주어졌을 때, 변환된 SQL 질의문이 다수의 저장소에 전달된다. 특정 저장소는 주어진 알고리즘에서 전제한 저장 구조와 동일할 수 있으나 대부분은 다른 저장 구조를 지닌다. SQL 계층은 변환 및 전달된 SQL 질의문을 각 저장소에 전달하는 역할을 수행한다. 알고리즘에서 전제한 저장 구조와 동일한 저장 구조를 지닌 저장소인 경우에는 테이블을 직접 액세스하는 SQL 질의문을 전달하게 되며, 동일하지 않은 저장 구조를 지닌 저장소에는 뷰를 대상으로 하는 질의문을 전달하게 된다. View 계층은 알고리즘과 다른 저장 구조를 지니는 저장소를 위해 생성된 뷰들의 집합이다. 따라서 알고리즘과 상이한 저장 구조를 지닌 저장소에 대한 검색은 View 계층을 대상으로 연산이 수행된다. 마지막으로, Physical 계층에서는 실제 웹 온톨로지 데이터 값들이 저장되어 있는 물리적 테이블의 집합이 존재하며 알고리즘과 동일한 저장 구조를 지니는 저장소에 대한 검색은 이 계층을 대상으로 연산이 수행된다.

이미 기술하였듯이, View 계층에 생성된 뷰는 각 저장 시스템의 저장 구조가 다를지라도 모두 알고리즘과 동일한 구조로 생성된다. 이 논문에서는 구현 및 실험을 위해 트리플 형태의 단일 테이블 구조를 지니는 Triples 모델과 Jena 저장 모델을 이용한다. Chebotko 알고리즘은 단일 테이블 구조를 전제로 개발되었다. 따라서 Triples 모델의 경우 저장 구조가 동일하므로 뷰를 생성해 줄 필요가 없다. 그러나 Jena의 경우 트리플 구조가 아니기 때문에 트리플 구조의 뷰를 생성해야 한다. 즉, Jena는 총 7개의 테이블로 구성되어 있는 저장소를 제공하며 그 중에서

* 이 연구에 참여한 연구자는 'BK 21 2단계 사업'의 지원을 받았음

jena_gntn_stmt 테이블에는 기본적인 데이터가 저장된다. 데이터 길이가 256바이트 넘을 경우에는 다른 세 개의 테이블에 데이터를 저장한다. 질의 결과로 추출해야 할 컬럼들은 jena_gntn_stmt 테이블의 SUBJ, PROP, OBJ와 jena_long_lit, jena_long_uri, jena_long_prefix 테이블의 checksum이다. 단, jena_long_lit, jena_long_uri, jena_long_prefix 테이블에는 데이터 값이 항상 저장되지 않으므로 데이터를 추출할 때의 조건으로 데이터의 길이가 256바이트가 넘을 경우에만 jena_long_lit, jena_long_uri, jena_long_prefix 테이블에서 데이터를 추출한다.

제안 모델에 대한 실험 및 평가를 위해 프로토타입을 개발하였으며, 이를 통해 제안 모델과 주요 알고리즘의 정확성 검증을 위한 실험을 실시하였다. 즉, 각각 다른 구조의 저장소를 지니는 시스템이 제안 모델을 바탕으로 동일한 데이터 셋과 질의문에 대해 동일한 질의 결과를 반환하는지를 측정하였다. 실험 대상 저장소는 Triples 모델과 트리플 구조의 뷰를 생성한 Jena 모델을 이용하였으며, 총 7개의 질의 패턴을 정의하였다. 실험한 결과에서, Triples 모델과 Jena 모델의 질의 결과가 동일함을 알 수 있다.

실험을 통해, 두 저장소가 서로 다른 구조를 지니지만 Jena 모델에 트리플 구조의 뷰를 생성함으로써 저장소에 영향을 받지 않고 특정 변환 알고리즘을 동일하게 이용할 수 있음을 알 수 있다. 실험은 Triples와 Jena 저장 모델만을 대상으로 수행되었지만, 다른 모델의 저장소에서도 트리플 구조의 뷰를 생성하면 동일한 변환 알고리즘에 의해 생성된 단일 SQL 질의문으로 동일한 질의 결과 검색이 가능하다.

결론적으로, 이 논문에서 제안한 모델을 이용한 접근 방법은 각 저장소에 적합하게 알고리즘을 변경하거나 물리적인 테이블 구조를 변경해야 하는 종속적 접근 방법에 비해 실용성 및 효율성 측면에서 보다 나은 성능을 보인다. 즉, 알고리즘을 변경해야 하는 접근 방법 (저장소에 종속)의 경우, 알고리즘 변경 및 수정된 알고리즘에 대한 정확성 검사 등을 위해 많은 비용이 요구된다. 또한 물리적인 테이블 구조를 변경하는 접근 방법 (알고리즘에 종속)의 경우, 각 저장소의 특징 (장점)을 잃게 되고 파서, 추론 엔진, 에디터 등과 같은 관련 모듈과의 연동이 불가능하게 되는 문제점을 야기한다. 따라서 이 논문에서 제안한 뷰를 이용한 저장소에 독립적인 활용 모델은 그 활용성 및 실용성, 각 저장소의 특성 및 관련 모듈과의 연동성 유지 등의 관점에서 많은 장점을 제공한다. 반면, 제안 모델의 경우 뷰의 생성에 따른 저장소의 크기가 증가된다는 단점을 지닌다.

3. 결론

이 논문에서는 저장소의 구조에 따라 변환 알고리즘을 수정해야 하는 문제점을 해결하고 변환 알고리즘을 다양한 저장소에 활용할 수 있는 모델을 제안하고 이를 위해 프로토타입을 구현하였다. 제안 모델은 저장소와 변환 알고리즘 간 독립성을 유지하기 위해 뷰를 중간 계층에 생성하는 접근 방법이다. 프로토타입 구현을 위해 가장 기능성이 높은 Chebotko 알고리즘을 변환 알고리즘으로 선택하였다. 또한 구현된 프로토타입을 이용하여 제안 모델의 정확성 검증을 위한 실험 결과 및 비교 평가 결과에 대하여 기술하였다. 결과적으로, 제안 모델은 저장소와 변환 알고리즘 간의 독립성을 위한 뷰를 생성함으로써 종속적인 구조의 문제점을 해결할 수 있다. 이는 각 저장소 구조에 적합하도록 변환 알고리즘 변경하거나 혹은 알고리즘에서 이용하는 저장 구조에 맞게 각 저장소의 테이블 구조를 변경하는 접근 방법에 비해 효율성, 활용성 및 실용성 측면에서 보다 나은 접근 방법임을 알 수 있다.

향후 연구로서, 먼저 각 저장소를 위한 보다 편리한 뷰 생성 방법에 대한 연구가 요구된다. 이는 이 논문에서 제안하는 모델의 활용성 및 사용자 관리의 용이성을 보다 향상시킬 수 있다. 추가적으로, 기존의 변환 알고리즘들이 지원하지 않는 SPARQL의 UNION과 FILTER 구문을 SQL로 변환시킬 수 있는 알고리즘 개발이 요구된다.

참고문헌

[1] Tim Berners-Lee, James Hendler, and Ora Lassila, "The Semantic Web." Scientific American, Vol. 284, No. 5, pp. 34-43, May 2001.

[2] Sesame: RDF schema querying and storage, http://www.openrdf.org/.

[3] Zhengxiang Pan, Jeff Heflin, "DLDB: Extending relational databases to support Semantic Web queries," Workshop on Practical and Scalable Semantic Web Systems, ISWC 2003, pp. 109-113, 2003.

[4] Steve Harris, "SPARQL query processing with conventional relational database systems," WISE 2005 Workshops, Vol. LNCS 3807, pp. 235-244, 2005.

[5] OWLJessKB : A Semantic Web Reasoning Tool, http://edge.cs.drexel.edu/assemblies/software/owljesskb/.

[6] Jena Semantic Web Framework, http://jena.sourceforge.net/.

[7] Dongwon Jeong, Myounghoi Choi, Yang-Seung Jeon, Youn-Hee Han, Laurence T. Yang, Young-Sik Jeong, and Sung-Kook Han, "Persistent Storage System for Efficient Management of OWL Web Ontology," Springer-Verlag, Lecture Notes in Computer Science (LNCS), Vol. LNCS 4611, pp. 1089-1097, July 2007

[8] SPARQL Query Language for RDF, W3C Working Draft, http://www.w3.org/TR/2006/WD-rdf-sparql-query-20061004/, 4 October 2006.

[9] Artem Chebotko, Shiyong Lu, Hasan M. Jamil, and Farshad Fotouhi, "Semantics Preserving SPARQL-to-SQL Query Translation for Optional Graph Patterns," Technical Report TR-DB-052006-CLJF, May 2006, Revised November 2006.

[10] sparql2sql - a query engine for SPARQL over Jena triple stores, http://jena.sourceforge.net/

[11] Steve Harris, "SPARQL query processing with conventional relational database systems," WISE 2005 Workshops, Vol. LNCS 3807, pp. 235-244, 2005.