분산 노드 기반 LOD 변환 방법 연구

김인아, 이규철(교신저자)
충남대학교
dodary0214@gmail.com, kclee@cnu.ac.kr

LOD Translation based on Distributed Nodes

Kim InA, Lee Kyu-Chul(Corresponding Author)
Chungnam National Univ.

요 약

최근 국내에는 다양한 요소가 원인이 되어 발생하는 사회 현안들에 공공데이터를 활용 및 분석하여 사건의 발생을 사전에 예측하려는 시도들이 증가하고 있다. 여러 데이터를 사용한 복합 분석을 위해서는 현재 보유하고 있는 데이터를 서로 연계해야 하며, 이에 따라 국내에서는 공공데이터를 LOD(Linked Open Data)로 변환하여 이를 기반을 데이터를 연계하려는 시도를 지속해오고 있다. 그러나 LOD의 자원을 표현하는 URI(Uniform Resource Identifier)는 긴 문자 수를 가진 문자열 형태로 저장되며, 이는 높은 저장 비용과 연산 비용을 발생시킨다. 본 논문에서는 LOD를 압축하여 저장하기 위한 변환 방법을 제시하며, 빠른 변환을 위해 해시 테이블 기반의 ID 딕셔너리를 생성하여 LOD를 정수 값으로 변환함으로써, 저장 비용과 연산 비용을 줄인다. 또한, LOD의 크기가 증가할수록 ID 딕셔너리 크기도 증가하므로, 분산 캐시 저장소인 Memcached를 기반으로 ID 딕셔너리를 여러 노드에 분산 저장하여 LOD를 변환한다. 본 논문의 테스트 결과, 데이터셋의 크기가 원본 크기에서 약 10%~14%까지 감소하였다.

Ⅰ. 서론

최근 개방된 데이터를 활용한 서비스 개발로 인해, 공공데이터에 대한 민간에서의 이용 현황 수가 급격히 증가하고 있다. 국내 데이터 산업 시장은 2015년도부터 2018년도 사이 5.6%의 성장세를 기록하였으며, 2024년도에는 데이터 산업 시장이 23조원을 넘을 것으로 예상된다[1]. 공공데이터에 대한 수요가 증가함에 따라, 최근에는 다양한 요소가 원인이 되어 발생하는 사회 현안들에 대해 공공데이터를 활용 및 분석하고 예측하고자 하는 시도들이 증가하고 있다. 복합 재난과 같은 사회 현안의 경우 기존과 달리 단일 데이터 종류가 아닌, 여러 종류의 데이터에 대한 복합적인 분석이 요구된다.

이를 위해서는 국내에 존재하는 많은 데이터들이 서로 연계되어야 하며, 국내에서는 현존하는 공공데이터를 LOD(Linked Open Data)로 변환하여 이를 기반으로 데이터를 연계하려는 시도를 지속해오고 있다. LOD는 RDF(Resource Description Framework) 포맷을 기반으로 데이터를 저장한다[2]. RDF는 기본적으로 주어(Subject), 서술어(Predicate), 목적어(Object)를 가지는 트리플(Triple) 형태로 데이터를 저장하는데, LOD는 이러한 트리플들이 동일한 주어, 목적어를 기반으로 서로 연결된 그래프 구조를 가진다. LOD의 트리플을 구성하는 주어, 서술어, 목적어 자리에는 URI(Uniform Resource Identifier), 문자열, 빈 노드가 올 수 있다. 이 중 URI는 자원에 대한 고유 식별자로, LOD의 많은 자원들이 이와 같은 URI로 구성되어있다. URI는 긴 문자 수를 가진 문자열 형태로 표현될 수 있으며, 이는 높은 저장 비용과 데이터 연산 시 많은 문자의 비교로 인한 연산 비용을 발생시킨다.

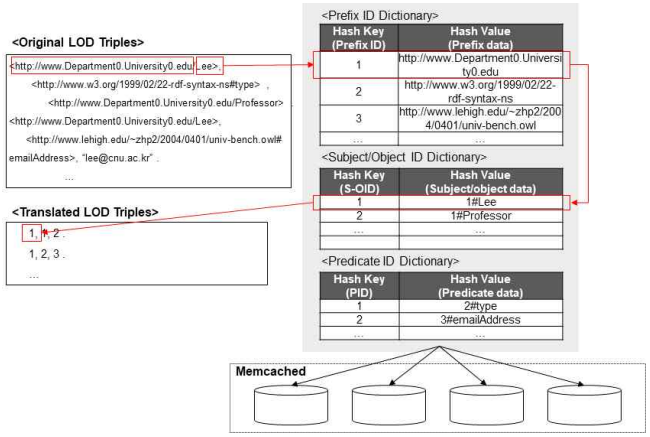
본 논문에서는 LOD를 압축하여 저장하기 위한 변환 방법을 제시한다. 본 논문은 해시 테이블 기반의 ID 딕셔너리를 생성하여 LOD를 정수 값으로 변환함으로써, LOD의 저장 및 연산 비용을 줄인다. 많은 연구들이 LOD를 다른 값으로 대체하여 저장하고 있으나, LOD 크기가 증가할 경우 단일 노드 기반의 변환 과정으로 인해 메모리 부족 문제가 발생하게 된다[3]. 따라서 본 논문에서는 많은 LOD를 변환하기 위해서, 인메모리 분산 저장소인 Memcached[4]를 사용하여 많은 원소가 포함된 ID 딕셔너리를 여러 노드에 분산하여 저장한다.

Ⅱ. 본론

LOD의 URI는 자원에 대한 고유 식별자로써, 자원을 구분하기 위한 Prefix와 이름이 결합된 형태를 가진다. LOD의 많은 요소를 구성하는 URI와 문자열은 결국 긴 문자 수를 가진 문자열로 저장되며, 이로 인한 저장 비용과 연산 비용을 야기한다. 본 논문에서는 해시(Hash) 테이블 기반의 ID 딕셔너리를 사용하여 트리플의 각 요소들을 정수 값으로 변환한다. 문자열에 비해 정수 값은 적은 저장 공간과 통신비용을 소비하며, 데이터 검색 시 데이터의 비교 비용을 줄일 수 있다.

본 논문은 데이터와 정수 ID를 매핑하는 Predicate, Subject/Object, Prefix 세 개의 ID 딕셔너리를 생성하였으며, 빠른 변환을 위해 세 개의 딕셔너리 모두 해시 테이블 구조를 가진다. 먼저 Predicate ID 딕셔너리는 중복되지 않은 트리플의 서술어 값에 매핑되는 ID를 저장하며, 정수 ID를 해시 Key로, ID에 매핑되는 서술어 값을 Value로 가진다. LOD의 주어와 목적어 자리에는 같은 자원이 위치할 수 있으므로, Subject/Object ID 딕셔너리는 중복되지 않은 트리플의 주어와 목적어 값에 매핑되는 ID를 저장한다. 마찬가지로 정수 ID를 해시 Key로, ID에 매핑되는 주어와 목적어 값을 Value로 가진다. 또한, URI는 자원을 구분하기 위한 Namespace를 Prefix로 가지며, 이로 인해 같은 클래스에 속한 많은 자원들은 동일한 Prefix를 가진다. 자원 간 Prefix 중복이 많을수록 Predicate, Subject/Object ID 딕셔너

리의 크기가 증가하게 되므로, 본 논문은 LOD의 Prefix 또한 정수 ID 값으로 변환하여 저장한다. Prefix ID 딕셔너리는 정수 ID를 해시 Key로, ID에 매핑되는 Prefix 값을 Value로 저장한다. 〔그림 1〕은 세 개의 해시 테이블 기반 ID 딕셔너리를 사용하여 LOD를 정수 값으로 변환하는 과정을 보여준다. 〔그림 1〕의 원본 LOD의 첫 번째 트리플의 주어 값에 대해서, 본 논문은 맨 마지막에 레이블된 'Lee'를 기준으로 앞부분을 Prefix로 분리한다. 분리된 Prefix는 Prefix ID 딕셔너리에 Value로 저장되며, '1'이라는 ID와 매핑된다. 따라서 변환 대상인 전체 주어 값은 Prefix에 해당하는 ID '1'과 뒤에 레이블된 이름 'Lee'가 결합된 '1#Lee'가 된다. 이러한 주어 값은 최종적으로 '1'이라는 ID와 매핑되어 변환 결과 값에 입력되며, Subject/Object ID 딕셔너리에 ID와 함께 저장된다.



〔그림 1〕 분산 노드 기반 LOD 변환 예시

그러나 LOD의 트리플 수가 증가하게 되면, 단일 노드의 메모리에 세 개의 ID 딕셔너리를 저장하기 어렵다는 문제점이 발생한다. 특히 주어와 목적어의 경우 전체 트리플 수보다 많은 수가 존재할 수 있으므로, Subject/Object ID 딕셔너리는 단일 노드에서 처리하기 어려운 원소의 크기를 가질 수 있다. 이를 해결하기 위해, 본 논문은 단일 노드 메모리 사이즈에 맞지 않는 ID 딕셔너리를 여러 노드에 분산 저장하여 데이터를 변환하였으며, ID 딕셔너리 저장을 위한 저장소로 Memcached를 사용하였다. Memcached는 분산 캐시 시스템으로써, 분산 노드에 독립적으로 존재하는 메모리를 논리적으로 하나의 메모리처럼 사용이 가능하게 한다. 본 논문에서 생성한 ID 딕셔너리는 해시 테이블 구조를 가지고 있으며, 여러 노드 간에 분산하여 저장된 해시 Key를 공유해야 한다. 따라서 본 논문은 Memcached를 기반으로 여러 노드의 메모리를 결합하여 하나의 메모리로 취급하고, ID 딕셔너리를 물리적으로 분산하여 저장함으로써, 대량의 LOD 트리플에 대한 변환 과정을 수행할 수 있다. 〔그림 1〕은 앞에서 생성한 세 개의 ID 딕셔너리를 Memcached로 결합된 분산 노드 기반 메모리에 분산 저장하는 것을 보여준다.

본 논문은 앞서 설명한 분산 노드를 기반으로 한 LOD 변환 과정을 통한 데이터 크기의 감소에 대해 테스트하였다. LOD 변환이 수행된 서버는 Xeon E5-2620 2.1GHz CPU, 64GB RAM, 1TB 7200RPM HDD 스펙을 가지고 있으며, 같은 스펙을 가진 16대의 서버에 ID 딕셔너리 저장을 위한 분산 캐시 시스템인 Memcached를 설치하였다. LOD 변환 테스트를 위해, RDF 데이터셋인 WatDiv〔5〕를 활용하였다. WatDiv는 RDF 데이터 관리 시스템의 성능 테스트를 위한 데이터셋으로, 사전 정의된 데이터 정보에 기반하여 다른 스케일의 데이터셋을 생성할 수 있다. 본 논문은 테스트를 위해 1M, 10M, 100M, 1B, 총 네 개의 다른 스케일을 가진 WatDiv 데이터셋을 생성하였다. 생성한 데이터셋에 대해 본 논문의 변환 과정을 수행한 결과, 데이터셋은 〔표 1〕에 나온 결과와 같이 크기가 감소하였다. 각 데이터셋의 경우 원본 크기에서 약 10%~14%의 크기로 줄어든 것을 알 수 있다.

〔표 1〕 LOD 변환 결과 데이터셋 크기 변화

데이터명	원본 크기	변환 크기
WatDiv-1M	0.14GB	0.02GB
WatDiv-10M	1.44GB	0.17GB
WatDiv-100M	14.64GB	1.90GB
WatDiv-1B	148.19GB	20.98GB

Ⅲ. 결론

본 논문에서는 LOD를 압축하여 저장하기 위한 변환 방법을 제시하였다. 본 논문은 빠르게 LOD를 변환하기 위해서 해시 테이블 기반의 세 개의 ID 딕셔너리를 생성하여 LOD를 정수 값으로 변환하였다. LOD를 정수 값으로 변환함으로써, 긴 문자열로 인한 저장 비용과 연산 비용을 줄일 수 있다. 또한, 변환하고자 하는 LOD의 크기가 증가할수록, LOD 값을 저장하고 있는 ID 딕셔너리 크기의 증가로 인해 메모리 부족 현상이 발생한다. 따라서 본 논문에서는 분산 캐시 저장소인 Memcached를 사용하여 분산 노드의 메모리를 하나의 메모리처럼 취급하고, ID 딕셔너리를 물리적으로 분산 저장하였다. 이처럼 분산 노드 기반의 변환 방법을 사용하여 대량의 LOD에 대한 변환 과정을 수행할 수 있으며, 테스트 결과 다른 스케일을 가진 데이터셋이 원본 크기보다 약 10%~14% 크기까지 감소되었다.

본 논문에서 제시한 LOD 변환은 ID 딕셔너리의 저장에 있어서 분산 노드를 기반으로 분산하여 저장하고 있다. 그러나 LOD 크기가 증가할 경우, ID 딕셔너리 뿐 아니라 변환한 LOD 또한 단일 노드에서 저장하는 데 문제가 발생하게 된다. 따라서 대량의 LOD 크기를 고려하여 변환한 LOD를 여러 노드에 분산 저장하는 것이 필요하다. 본 논문은 향후 연구로 변환한 LOD를 분산 파일 시스템인 HDFS에 저장하고, 분산 병렬 기반의 데이터 검색 과정에서 입력 데이터의 I/O 비용과 조인 연산 비용을 고려한 저장 구조를 연구할 예정이다.

ACKNOWLEDGMENT

본 연구는 국가과학기술연구회에서 시행한 개방형데이터솔루션(DDS) 융합연구단사업 "AI기술을 활용한 공공데이터 기반 지역현안 솔루션 개발 및 실용화-안전안심사회 실현을 위한 실증연구중심으로-"의 지원을 받아 수행된 연구임.

참 고 문 헌

[1] 한국데이터산업진흥원. "2018 데이터산업 현황 조사", 2018.

[2] Lassila, Ora, and Ralph R. Swick. "Resource description framework (RDF) model and syntax specification.", 1998.

[3] Neumann, Thomas, and Gerhard Weikum. "RDF-3X: a RISC-style engine for RDF.", Proceedings of the VLDB Endowment 1.1, pp. 647-659, 2008.

[4] Memcached, (https://memcached.org/)

[5] ALUÇ, Güneş, et al. "Diversified stress testing of RDF data management systems.", In: International Semantic Web Conference. Springer, Cham, pp. 197-212, 2014.