www.calsec.or.kr
http://dx.doi.org/10.7838/jsebs.2014.19.1.131

트리플 데이터베이스 단축 경로 이득 함수와 구성 인자 실험 분석

Empirical Analysis on the Shortcut Benefit Function and its Factors for Triple Database

강승석(Seungseok Kang)*, 심준호(Junho Shim)**

초 록

3-컬럼의 트리플 테이블로 구성되는 트리플 데이터베이스의 질의 처리는 고비용이 드는데, 단축 경로는 그 비용을 감소시키는 방법으로 알려졌다. 어떠한 단축 경로를 선택 구성할지는 주요한 문제이며, 질의 빈도를 기반으로 단축 경로 이득을 계산하는 방식이 주로 사용된다. 하지만 이러한 방식은 트리플 데이터의 추가 혹은 변경을 적절히 반영하지 못한다. 본 논문에서는 질의 처리 시간 단축 측면뿐 아니라 경로 구축 및 유지 비용도 고려하는 이득 모델을 다룬다. 이득 모델은 이득 함수로 설계되어 단축 경로 선택 기법에 적용된다. 이득 함수 구성 인자가 미치는 영향을 실세계 트리플 데이터를 사용해 실험 분석한다.

ABSTRACT

A triple database consisting of a number of three-column tables require high cost of query processing, whereby building a shortcut is known as an effective way to reduce the cost. It is important to figure out what shortcuts needs to be selectively built. Most shortcut selection algorithms make use of a benefit model that considers the query frequency. However they work poor to reflect the database update. In this paper, we consider a benefit model for triple databases. The model considers not only the profit of query response times but also the building and maintenance costs of the shortcuts. We apply the model to design a benefit function which can be plugged in a greedy-based shortcut selection algorithm. We perform the empirical experiments on a real-world dataset and analyze the effect of each factor employed in the benefit function.

키워드 : 단축 경로, 시맨틱웹, 트리플, 이득 함수
shortcut, semantic web, triple, benefit function

본 연구는 숙명여자대학교 2012학년도 교내연구비 지원에 의해 수행되었음.
* First author, Samsung Electronics(sseok.kang@samsung.com)
** Corresponding author, Sookmyung Women's University(jshim@sookmyung.ac.kr)
2014년 01월 24일 접수, 2014년 02월 14일 심사완료 후 2014년 02월 17일 게재확정.

1. 서 론

트리플(triple)은 시맨틱 웹(Semantic Web)에서 사용되는 데이터 표준 표현 방식으로, 지식(knowledge)을 <Subject, Predicate, Object>의 3개의 구성 요소로 표현한다. 이를 통해 시맨틱 웹은 컴퓨터가 이해할 수 있는 의미 정보를 데이터 안에 포함함으로써, 지능적인 컴퓨팅 환경을 추구할 수 있게 한다. 예를 들어 전자상거래의 상품 정보는 트리플 형식의 데이터 표현에 의해 다양한 의미 전달을 가능케 한다[11, 12]. 트리플 기반의 트리플 데이터베이스(triple database)는 잘 구성된 데이터 표현 형식과 풍부한 데이터 표현 범위에도 불구하고, 트리플 데이터 관리 및 질의에서 발생하는 성능 문제로 인하여 실용적으로 사용되는 데 어려움이 있다.

자기 조인(self-join) 연산이란 그래프 형태로 나타내어지는 트리플 데이터베이스 상에서 질의 처리를 위해 트리플을 저장하고 있는 트리플 테이블을 반복적으로 조인하는 연산이다. 자기 조인 연산은 3개의 칼럼으로 이루어진 단일 트리플 테이블을 기반으로 하는 트리플 데이터베이스를 사용할 때 흔히 나타나는 성능 저하의 주요한 요인이다[5, 9, 8]. 트리플 데이터가 급격하게 증가하여 수억 개의 트리플을 포함하고 있는 경우, 한 번의 트리플 테이블의 자기 조인만으로도 전체 시스템의 성능을 급격하게 떨어뜨릴 수 있다[2]. 이를 해결하기 위하여 기존 연구에서는 트리플을 분산하여 저장하거나, 3-컬럼 트리플 테이블을 여러 개의 2-컬럼 테이블로 나누어서 표현하는 등의 저장 구조의 변환을 이용하거나, 특정한 노드에서 이웃하지 않은 다른 노드까지의 직접 경로를 인덱스 형태로 추가하여 자기 조인을 줄이는 단축 경로 선택 기법을 이용하여 문제를 해결하고 있다[1, 3, 7, 9].

저장 구조의 변환을 통해 문제를 해결하는 방법은 자기 조인이 발생되는 요인을 그대로 두고, 트리플 테이블을 분산하거나 분열하여 해결하는 방법으로, 자기 조인 문제의 해결보다는 트리플의 효율적인 저장 및 보편적 질의 처리에 초점을 맞추고 있는 연구이므로 본 논문의 연구 범위에는 포함되지 않는다. 두 번째 해결 방식인 단축 경로 선택 기법은 자기 조인을 요구하는 노드 간의 그래프 탐색을, 가상 경로(virtual path)를 추가하여 자기 조인의 발생 자체를 줄이는 것을 목표로 한다. 이때 어떠한 단축 경로를 우선적으로 선택해야 하는가가 단축 경로 선택 기법의 주요한 연구 주제가 되는데, 기존 연구는 대부분 단축 경로의 효율성을 계산하기 위하여 질의에 기반을 둔 단축 경로 효율 계산 모델을 활용하고 있다. 즉, 기존의 질의 형태 및 패턴을 파악하여, 가장 자주 사용되는 질의를 포함하고 있는 단축 경로를 우선적으로 생성하는 것이다. 이 방법은 가장 많이 발생하는 질의를 빠르게 처리할 수 있는 방법으로 일반화할 수 있으므로 대부분의 서비스에 적용 가능하나, 서비스의 변경이나 추가로 인하여 새로운 질의 형태를 요구하는 과정이 시스템에 추가될 경우 콜드 스타트 문제(cold-start problem)이 발생할 가능성이 있다. 이전에 한 번도 사용되지 않았던 트리플이나 새롭게 추가된 트리플에 대해서는 어느 사용자에게도 질의에 이용되지 않았기 때문에, 상호작용이 일어나지 않는 초기 아이템 문제(first-item problem)이 발생할 가능성 또한 존재한다[8].

특히 기존의 트리플 데이터베이스 관련 연구들은 생성된 트리플 데이터베이스에 대하여 가용 한도 및 질의 처리 한도에 대한 연구에 집중하는 반면, 트리플 데이터의 변경(update)으로 인한 데이터 관리 문제는 상대적으로 연구가 많이 이루어져 있지 않기 때문에, 이러한 초기 아이템 문제는 트리플 데이터베이스가 기업 프로세스에 적극 활용될수록 더 자주 발생할 수 있는 문제이다. 본 연구는 바로 단축 경로의 구성에 있어서, 트리플 데이터의 변경 측면을 고려해야 한다는 점에 주목한다.

본 논문에서는 트리플 데이터베이스에서 적절한 단축 경로 선택을 위한 단축 경로 이득(benefit) 모델을 설명한다. 제시하는 이득 모델은 단축 경로 선택에 따른 질의 처리 비용의 감소에 따른 이득뿐만 아니라, 트리플 데이터베이스의 변경 및 갱신 비용도 반영하도록 통합 설계되어 있다. 이러한 이득 모델은 이득 함수(benefit function)로 설계되어 단축 경로 선택 알고리즘에 사용된다. 제시한 이득 함수가 그리디 기반의 단축 경로 선택 알고리즘에 적용될 때, 이득 함수의 각 구성 요소가 트리플 데이터베이스 질의 처리 환경 및 데이터베이스 유지에 어떠한 영향을 주는지를 분석한다.

본 논문의 내용은 다음과 같이 구성된다. 제 2장에서는 트리플 단축 경로 문제와 이득 모델에 대해 기술한다. 제 3장에서는 확장된 이득 모델과 이득 함수에 반영된 구성 인자를 설명한다. 제시된 이득 함수의 성능은 제 4장에서 실험 되고 그 결과가 분석 기술된다. 제 5장에서는 결론을 제시한다.

2. 단축 경로와 이득 함수

2.1 트리플 그래프와 단축 경로

트리플 데이터베이스는 일반적으로 노드(node)와 노드 사이의 간선 연결로 표현되는 그래프로 나타낼 수 있기 때문에, 본 논문에서는 전통적인 그래프 표현 스킴(scheme)에 기반한 트리플 그래프(triple graph)를 정의하여 단축 경로 선택의 기반 데이터 구조로 사용한다[8]. 트리플 데이터베이스로부터 구성된 트리플 그래프는 노드(node) 집합 V와 간선(edge) 집합 E를 포함한 방향성 그래프 G = (V, E)이다. 여기에서 V는 트리플 데이터베이스의 모든 Subject와 Object의 값으로 구성된 유한한 개수의 노드의 집합이며, E는 Subject와 Object 간의 모든 Predicate로 이루어진 유한한 개수의 간선의 집합이다.[1] 단축 경로(shortcut)는 트리플 그래프 내에서 서로 인접하지 않은 두 개의 노드 사이의 가능한 모든 가상 경로(virtual path)를 의미한다. 단축 경로 집합 SC = {sc_1, sc_2, …, sc_n}은 단축 경로 sc들의 집합이다.

<Figure 1>은 노래와 가수 정보를 표현하는 트리플 그래프에서의 단축 경로의 예를 보여준다. 그래프에서 Musician 노드와 Title 노드 간에는 간선이 실제로는 존재하지 않는다. 이러한 트리플 그래프에서 만일 Musician 노드와 Title노드 두 정보로부터 대답을 구해야 하는 질의 처리가 필요하다면, Musician으로부터 Album, Album으로부터 Song, Song

1) 본 논문에서는 설명의 단순화를 위하여 스키마와 인스턴스를 그래프를 구분하지 않는다는 점이 [8]과 다르다.

으로부터 Title 노드로 실제 존재하는 간선들을 따라가야 한다. 이때 Musician으로부터 Title 노드로 직접 연결된 *sc1* 가상 간선을 구축해 놓는다면 Musician에서부터 Title 정보의 연결이 단축된다. 이러한 *sc1*를 단축 경로라고 하고, 이 예에서 단축경로 집합은 SC = {*sc1, sc2, sc3, sc4, sc5*}이다.

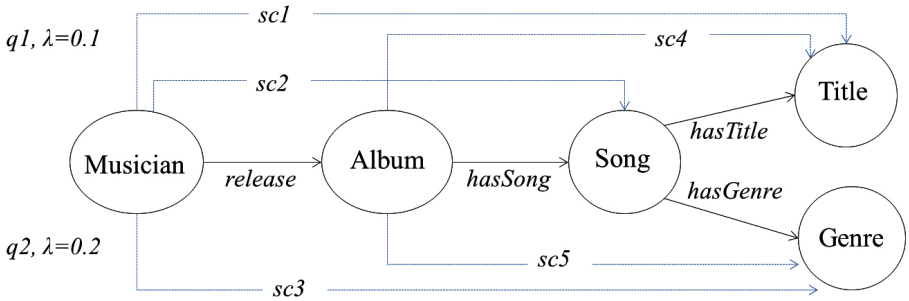
단축 경로 선택 기법에서 중요한 다른 구성 요소는 질의(query)이다. 질의 집합(query workload)은 미리 구성된 질의들로 구성되며, 질의는 사용자가 필요에 따라 시스템에 요구하는 결과를 가져오기 위한 그래프에서의 특정 경로를 뜻한다. 각 질의에는 질의가 요청되는 빈도를 뜻하는 질의 빈도(query frequency)가 주어지며, 과거에 자주 요청되었던 질의 또는 질의 패턴일수록 더 높은 질의 빈도값을 가진다.

<Figure 1>에서, 노래와 가수 정보를 표현하는 트리플 그래프의 예에서 *Let it go*라는 제목을 가진 노래를 부르는 가수의 이름을 구하는 질의를 생각해보자. 이 질의는 트리플 그래프에서 Musician과 Title을 연결하여 그 정보를 얻게 되는데, 이렇게 Musician로부터 Title로 따라가야 질의가 *q1*이고 그 빈도는 0.1이다. 또 다른 질의 *q2*는 Musician으로부터 Genre을 연결하여 그 정보를 얻게 되는 처리가 필요한데, 그 빈도는 0.2로 주어져 있다.

2.2 단순 이득 모델

주어진 트리플 그래프 G와 질의 집합 Q에 대한 단축 경로 선택의 문제는 선택된 경로로부터 얻을 수 있는 이득(profit)을 최대화 시키도록 경로를 선택하는 것이다. 선택된 단축 경로의 수가 많아짐에 따라 단축 경로를 통해 얻을 수 있는 이득이 많아지는 것은 당연하다. 단축 경로 문제에 대한 해결이 아무런 제한이 없는 것은 아니다. 무한정으로 주어진 비용(cost) 한도에서 문제를 풀기보다는 적당한 문제 해결 비용이 미리 주어진 경우가 대부분이고, 이때 제일 중요시 되는 비용은 시간(time)이다. 또한 가능한 모든 단축 경로를 모두 구성하기보다는 구축할 수 있는 단축 경로의 총 공간(space)이 제한되는 것도 일반적이다.

단축 경로 선택 문제는 관계 데이터베이스 분야에서 뷰(view) 구성 혹은 선택 문제와 개념적인 연관성을 가진다[15, 10]. 트리플 데이터베이스나 관계 데이터베이스에 존재하지는 않는 단축 경로나 뷰와 같은 추가적인 구



〈Figure 1〉 Shortcuts and Queries in a Music Triple Graph

조를 생성하고 유지함에 의해 이러한 구조로부터 이득을 얻으려는 공통점이 있다. 단축 경로는 다중의 트리플로부터 구성되고, 뷰는 다중의 베이스 테이블로부터 구성되는 차이점이 있지만, 두 문제 모두 기본적으로 NP-complete문제이다[10]. 트리플 데이터베이스 단축 경로 선택 문제 해결은 이와 같은 종래 관계데이터베이스 뷰 구성 문제 해결을 참조할 수 있는데, 실제로 본 연구의 이득함수 모델은 이득 함수에 질의 수행에 대한 단순 이득뿐 아니라 유지 비용까지 통합하는 [15]의 모델을 참조 발전시킨다.

하나의 단축 경로를 구성함으로써 얻을 수 있는 이득은 단축 경로 없는 경우 자기조인 연산에 들어가는 질의 처리비용에서 단축 경로를 사용한 질의 처리비용의 차이이다. 예를 들어 앞 절의 <Figure 1>에서 단축 경로 *sc1*의 이득은 아래와 같이 계산될 수 있다.

$$queryCost(Musician \xrightarrow{release} Album \xrightarrow{hassong} song \xrightarrow{hasTitle} Title) - queryCost(Musician \xrightarrow{sc1} Title)$$

단축 경로로부터 얻을 수 있는 이득은 이득을 얻을 수 있는 질의가 처리될 때마다 발생하므로, 각 질의 (q_k)에 대한 빈도(λ_k)는 단축 경로로부터 얻을 수 있는 이득의 계산에 중요한 구성 요소가 된다. 트리플 데이타베이스 환경에서 질의 빈도는 미리 고정된 값으로 주어지기보다는 적당한 방법으로 예측 계산하는 것이 일반적이다. 본 논문에서는 질의 빈도의 예측은 주된 관심이 아니며, 과거의 질의 수행 로그로부터 주어진 시간 동안 질의 발생 빈도수를 계산해 질의 빈도를 예측한다고 가정한다. 이 가정을 사용하여 단축 경로 sc_i의 이득 $profit(sc_i)$을 기술하면 다음과 같다.

$$profit(sc_i) = \sum_{q \in RQ_i} (QueryCost(q, NULL) - QueryCost(q, sc_i))$$

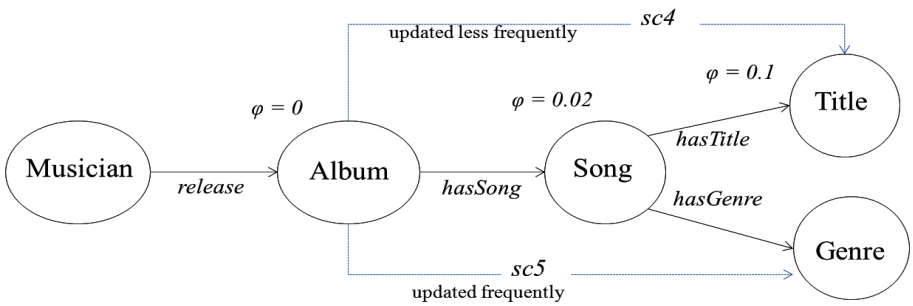
여기서 RQ_i는 단축 경로 sc_i를 사용해 질의 처리를 할 수 있는 질의 집합이고, $QueryCost(q, sc_i)$는 질의 q를 단축 경로 sc_i를 사용해 처리하는 질의 비용이다. 이에 비해 $QueryCost(q, NULL)$는 질의 q를 단축 경로 사용 없이 처리하는데 드는 질의 비용이다. 앞서 설명하였듯이 질의 q의 처리비용 및 이득에는 빈도가 고려되어야 하고, 예를 들어 위의 $QueryCost(q, sc_i)$는 아래와 같다.

$$QueryCost(q, sc_i) = \sum_{q_k \in q} \lambda_k \times queryCost(q_k, sc_i)$$

3. 확장된 이득 모델과 단축 경로 문제

3.1 확장된 이득 모델

트리플 데이터베이스에서의 단축 경로 구성에는 관계 데이터베이스나 데이터 웨어하우스에서의 뷰(view) 유지 문제와 같은 단축 경로 유지(maintenance) 비용이 든다. 트리플 데이터베이스 스키마에 업데이트가 일어나게 되면 단축 경로 역시 변화가 필요하다. 앞 절의 음악 트리플 데이터베이스 예를 살펴보기로 하자. <Figure 2>에는 Album과 Title에 *sc4*, Album과 Genre에 *sc5*라는 단축 경로가 구성



〈Figure 2〉 Shortcut Maintenance Cost and Update Frequency

되어 있다. 데이터베이스의 각 노드는 업데이트 비율(update frequency)이 계산되어 있다고 가정하고, Title과 Genre의 다른 요소들이 동일한 상태에서 Genre의 업데이트 비율이 Title의 업데이트 비율보다 높다고 하자. 이 경우, *sc4*와 *sc5*는 Title이나 Genre의 업데이트에 따라 재구성 (rebuild)되어야 하므로 *sc5*의 재구성 비용이 *sc4*의 재구성 비용보다 더 들게 된다. 다시 말하면 *sc5* 단축 경로의 이득은 상대적으로 높은 재구성 비용이 들기 때문에 이득 계산 모델은 이러한 재구성 비용을 감안해야 한다.

단축 경로의 비용은 단축 경로 구성비용(building cost)과 구성된 단축 경로를 적절하게 유지하는데 필요한 유지 비용(maintenance cost)을 모두 반영해야 한다. 유지 비용을 계산하는 방법과 이를 전체 단축 경로의 비용에 반영할 수 있는 방법은 다양할 수 있다. 우리 모델에서는 단순화 측면에서 단축 경로의 비용은 구성비용과 유지 비용의 단순 합으로 모델링 한다.

$$Cost(sc_i) = buildCost(sc_i) + maintenanceCost(sc_i)$$

단축 경로의 구성비용(building cost, 위에서 $buildCost$)은 트리플 그래프의 스키마에서 존재하는 개별 인스턴스들의 개수에 비례 증가하게 된다. 스키마에 따른 인스턴스의 개수가 많을수록 구성비용은 증가하기 때문이다. 예를 들어, <Figure 2>에서 Album으로부터 Title로의 단축 경로의 실제 인스턴스, 즉 개별 앨범과 노래의 제목의 개수가 1,000개라면, Album 으로부터 Title 사이의 *sc4* 단축 경로의 구성 비용은 1,000개 단축 경로를 구성하는 것처럼 증가한다. 즉, 단축 경로의 구성비용은 아래와 같이 나타낼 수 있다. 여기서 $\mu(sc_i)$는 트리플 그래프에서 단축 경로 sc_i의 실제 인스턴스 개별 개수를, $\delta(sc_i)$는 단축 경로 sc_i의 스키마 차원의 구성비용을 나타낸다.

$$\begin{aligned} buildCost(sc_i) &= \mu(sc_i) \times \delta(sc_i) \\ &= \sum_{sc_i \in sc} \mu(sc_i) \times \delta(sc_i) \end{aligned}$$

단축 경로 유지 비용(maintenance cost, 위에서 $maintenanceCost$)은 구성 비용과는 다르다. 트리플 데이타베이스의 스키마가 변경되는 빈도를 업데이트 비율 *update frequency* φ이라고 하자. 즉, φ는 트리플 그래프의 특정 노드가 변경되는 확률을 나타낸다. 예를 들어, <Figure 2>에서 Title의 업데이트 비율은 0.1인데, 이는 트리플 그래프에서 Title

노드가 10번의 질의 처리 과정 동안 1번 변경되는 것을 의미한다. <Figure 2>에서 Album의 업데이트 비율은 0으로서, 이는 트리플 그래프에서 Album 노드는 업데이트가 발생하지 않음을 의미한다.

단축 경로의 전체 업데이트 비율은 단축 경로에 속한 노드들의 업데이트 비율의 합이라 볼 수 있다. 예를 들어, <Figure 2>에서 단축 경로 *sA*(*Album*→*Title*)의 업데이트 비율은 0+0.02+0.1 = 0.12가 된다. 단축 경로 sc_i와 sc_i에 나타나는 노드를 v_j라 하고, $\rho(sc_i)$를 sc_i의 유지 비용이라 하자. 그러면 단축 경로 집합의 전체 유지 비용은 아래와 같이 된다.

$$maintenance Cost(sc_i) = \varphi(sc_i) \times \mu(sc_i) \times \rho(sc_i),$$

$$where \ \ \varphi(sc_i) = \sum_{v_i \in RN_i^{sc}} \varphi(v_i)$$

단축 경로의 유지 비용을 어떻게 모델링 할지는 다양한 방법이 존재한다[6]. 본 연구에서는 문제의 단순화를 위해 단축 경로의 유지는 단축 경로의 재구성을 통해서 이루어진다고 가정한다. 즉, 유지 비용을 재구성 비용(rebuilding cost)으로 가정하여 문제를 단순화 시킨다.

이상을 종합하면 트리플 그래프의 유지 비용을 고려한 확장된 이득 모델(benefit model)은 아래와 같이 정리된다.

$$benefit(sc_i) = profit(sc_i) - totacl Cost(sc_i)$$

$$= \sum_{q \in RQ_i} \sum_{q_k \in q} \lambda_k (queryCost(q_k, NULL) - queryCost(q_k, sc_i)) - \mu(sc_i)(\delta(sc_i) + (\varphi(sc_i) \times \rho(sc_i)))$$

3.2 이득 모델의 경로 선택 문제 적용

앞 절의 이득 모델을 사용해 경로 선택 문제의 목적 함수(objective function)를 기술해보면, '$\sum_{i=1}^{m} size(sc_i)$ < 주어진 단축 경로 저장 공간 크기'와 같은 주어진 제한 조건하에서, $\sum_{i=1}^{m} benefit(sc_i)$의 값을 최대화(maximize) 시키는 단축 경로 집합{sc_1, sc_2, …, sc_m}을 찾는 문제가 된다. 이러한 단축 경로 문제 자체는 제 2절에서 설명한 것 같이 NP-complete의 문제이고, 최적의 부분 해결 방안을 구하려는 많은 방안이 존재할 수 있다.

본 논문에서는 PageRank[14]를 이용한 단축 경로의 프루닝(pruning) 단계와, 프루닝을 거쳐 제시된 후보 단축 경로(candidate shortcuts)로부터 목적함수를 최대화 하도록 그리디(greedy) 기법을 사용해 최종적인 단축 경로 집합을 구하는 단계적 해결 방안을 사용한다. 자세한 해결 방안과 구체적인 알고리즘은 Kang[8]을 참조할 수 있는데, 본 논문에서는 그리디 기법을 사용한 단축 경로 선택 과정이 하나의 과정으로 이루어지고 있는데 비해, Kang[8]에서는 그 과정이 질의 빈도를 고려한 단축 경로 후보 선택 과정과 그 이후 유지 비용까지 고려한 단축 경로 선택과정의 두 단계가 구분되어서 이루어지는 차이점이 있다. 또한 Kang[8]연구에서는 후보 단축 경로 선택에 PageRank 이외의 휴리스틱(heuristic)을 사용한 프루닝 과정도 고려하는데 비하여, 본 논문에서는 단축 경로 유지 비용의 이득 모델 반영이 그 주안점이므로 PageRank에 한정한다.

4. 실험 및 결과 분석

4.1 실험 환경

본 절에서는 앞에서 제시된 이득 모델의 사용한 단축 경로 성능을 실 세계에서 사용되는 데이터 셋에서 실험을 하고 그 결과를 분석한다. 실험 성능은 구체적으로 단축 경로 구성을 통한 질의 성능 실험, 단축 경로 구성 비용 실험, 단축 경로 유지 비용 실험의 세 가지를 하여 제시된 이득 모델에서의 단축 경로 유지 비용 요소와 구성 비용 등에 대한 요소가 질의 성능 향상과 더불어 어떠한 결과를 보이는지를 보인다. 실험 환경은 Intel CPU 쿼드코어 2.4GHz, 16GB RAM, Windows 7 환경을 사용하였다.

실험의 데이터 셋은 실세계의 DBLP[13] 데이터를 사용한다. DBLP 데이터에 나타나는 학술논문이 실린 프로시딩 정보, 저자 정보, 프로시딩 시리즈 정보, 출판사 정보 등을 담고 있고, 데이터는 트리플로 변환되었다. 실험 데이터는 약 1.8M 튜플을 가지고 있고, PageRank기법에 따른 후보 단축 경로 프루닝 등을 위한 단축 경로 총 수는 10개를 사용하였다.

트리플 데이터베이스에서 처리될 질의는 [4, 5, 7] 등에서 제시된 바와 같이 Subject-Subject, Subject-Object, Object-Object 타입과 자기-조인 수 등을 골고루 고려한 총 10개의 질의를 임의로 생성하였고, 각 질의의 빈도는 [5, 6, 8] 등에서 제시된 값의 범위에서 임의 값(random value)을 생성해 사용하였다.

4.2 질의 성능

이 절에서는 확장된 이득 모델을 단축 경로 선택 문제에 그리디 기법을 사용해 적용한 기법의 성능을 분석한다. 질의 성능 시험에서 비교되는 알고리즘은 트리플 데이터베이스에서 단축 경로를 구성하지 않고 질의를 처리하는 방법이다. 본 논문에서는 이를 NS (No shortcuts) 알고리즘으로 기술한다. NS는 우리 알고리즘과 같이 단축 경로를 구성하는 방법의 성능을 비교할 수 있는 기준 알고리즘으로서 역할을 한다.

본 논문에서 제시한 방법으로 단축 경로를 구성하여 질의를 처리하는 알고리즘은 RS (Reduced Shortcuts)로 기술한다. 참고로 실험에서 주어진 질의 처리에 사용할 단축 경로가 없는 경우라면, 시스템은 기저(baseline) 트리플 데이터베이스로부터 전통적인 방법 그대로 자기-조인 연산을 수행하도록 하였다.

<Table 1>은 DBLP 데이터베이스에서 수행된 임의의 10개의 질의 평균 반응 시간(average query response time) 결과를 보여주고 있다. 결과에서 알 수 있듯이 단축 경로를 구성하여 질의 처리를 하는 것은 그렇지 않는 기법에 비해 평균 60.7% 이상 질의 처리 성능을 향상 시킨다고 볼 수 있다. 질의의 형태는 질의 처리 향상율에 영향을 미치게 되며 실험 결과에서는 q3 질의에서는 평균 40.2%의 질의 처리 성능 향상을 보이고 있는데 이는 다른 질의들에 비교할 때 최소의 성능 향상율이다. 반대로 q10 질의에서는 평균 69.4% 이상의 질의 처리 성능 향상을 보이고 있는데 이는 다른 질의들에 비교할 때 최대의 성능 향상율이다.

〈Table 1〉 Average Query Response Time Improvement

	q1	*q2*	*q3*	*q4*	*q5*	*q6*	*q7*	*q8*	*q9*	*q10*
NS	3.2	4.0	3.9	10.7	17.8	20.1	24.9	40.2	58.0	78.4
RS	1.8	1.4	2.3	4.0	8.2	6.8	10.6	17.0	26.5	24.0
Improvement(%)	43.8	63.9	40.2	62.7	54.3	66.1	57.6	57.7	54.3	69.4

본 연구의 중점 분석 사항은 단축 경로 선택에 따른 질의 성능 향상 자체라기보다는 이득 모델에 유지 비용 등을 고려한 사항이므로, 질의 형태와 RS에 선택적 기법을 추가하여 발생하는 세부적 성능 변화는 Kang[8]를 참조하기 바란다.

4.3 단축 경로 구성 비용 분석

이 절에서는 단축 경로 구성 비용을 확장된 이득 모델에 적용한 그리디 단축 경로 선택 문제로 해결한 RS 기법에 대해서 단축 경로 구성 비용 실험 및 결과를 보여준다.

단축 경로 구성 문제에서, 가능한 단축 경로를 많이 구성하여 놓게 되면, 질의가 이용할 수 있는 단축 경로의 존재 가능성도 높아지게 되고, 따라서 평균 질의 처리 시간을 단축할 수 있음은 자명하다. 하지만 많은 단축 경로의 구성은 그에 따라 더 많은 구성 비용과 유지 비용이 들게 되고, 현실적으로 제한이 되는 단축 경로의 수는 제한되게 된다. 제한의 기준은 다양할 수 있는데, 실 세계에서 종종 사용되는 기준은 단축 경로 집합을 저장할 공간 크기이거나 혹은 단축 경로들을 생성하는데 드는 구성 시간이다. 본 논문에서는 단축 경로 집합을 저장할 공간을 제한 기준으로 실험하였으며, 그 기준은 기저 데이터베이스의 크기의 퍼센트 비율로 하였는데, 구체적으로는 DBLP 데이터베이스의 크기의 20%, 40%, 60%, 80%, 100%를 기준 값으로 실험하였다.

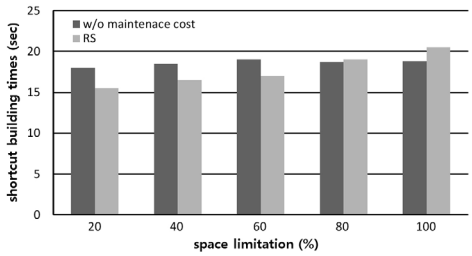
본 논문에서 RS와 단축 경로 구성 비용을 비교하는 대상은 Dritsou et al.[6]에서 제시된 방법이다. Dritsou et al.[6]에서는 질의 빈도를 고려하되 모든 단축 경로의 이득을 계산하여 상위 단축 경로만을 선택한다. RS는 단축 경로 계산에 질의 빈도를 고려하되, 단축 경로 집합 저장 공간 제한을 고려한다. 또한 이득 계산에 유비 비용의 고려 여부가 차이점이다.

<Figure 3>은 두 개의 단축 경로 구성 알고리즘의 단축 경로 집합 구성 비용(Y-축)을 단축 경로 집합의 저장 공간 크기(X-축)에 따라 실험한 결과를 보여주고 있다. 그림에서 RS 비교 대상 기법은 w/o maintenance cost로 표기되었다. 결과에서 보여 주듯이 w/o maintenance cost 기법은 단축 경로의 저장 공간 크기에 관계없이 생성 가능한 모든 단축 경로의 이득을 계산하여야 하기 때문에, 계산량을 감소시키고자 중요한 단축 경로에 대한 우선 고려를 그래프 분석 기법을 응용하여 적용한 RS와는 차이가 있다.

RS 기법은 80% 이하의 저장 공간 크기에서 w/o maintenance cost에 비해 모두 그 구성 비용이 적었다. w/o maintenance cost 기법은 저장 공간 크기에 상관없이 모두 일정한

패턴을 보였는데, 그 이유는 저장 공간의 크기와 상관없이 구성 가능한 모든 단축 경로의 경우의 수를 모두 고려한 이후에 저장 공간에 포함될 단축 경로를 선택하기 때문이다.

상대적으로 RS 기법은 저장 공간 크기가 늘어남에 따라 더 많은 수의 단축 경로를 구성하게 되고, 그에 따라 단축 경로의 구성 비용이 늘어나게 된다. 마지막으로 저장공간 비율을 20%~100%를 모두 고려한다면, RS의 구성비용이 평균적으로 w/o maintenance cost에 비해 역 5% 정도 낮은 것으로 나타난다.



〈Figure 3〉 Shortcut Building Time Comparison

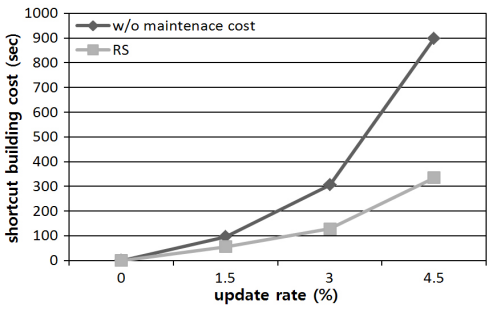
4.4 단축 경로 유지 비용 분석

단축 경로 구성은 평균 질의 처리 시간을 단축할 수 있는 장점이 있는 반면, 질의 처리 결과가 바르게 하기 위해서는 데이터베이스 변화에 따른 단축 경로를 반영하는 유지 비용의 오버헤드가 발생하게 된다. 이 절에서는 우리 RS 기법과 같은 단축 경로 선택 구성 알고리즘의 유지 비용 실험 및 결과를 보여준다. 본 실험에서 RS 기법은 앞 절에서와 마찬가지로 [6] 기법과 그 비용을 비교 분석한다.

단축 경로의 유지 비용은 트리플 데이터베이스 변경율(update rate)에 따라 영향을 받음은 자명하다. 높은 변경율은 상대적으로 높은 단축 경로 유지 비용을 발생시킬 것이다. 본 실험에서는 변경율(φ)을 1.5%, 3%, 4.5%로 설정했을 때 그 결과를 측정하였는데, 여기서 변경율이란 하나의 질의 요청되어 처리되는 동안 평균적으로 변경율(φ)%의 확률로 노드가 그 값을 변경하는 것을 의미한다. 참고로 노드의 값이 변경되면 해당 노드를 포함하는 단축 경로의 이득 값도 재계산된다. 본 실험에서는 DBLP 데이터 100% 저장공간 기준으로 실험하였다.

<Figure 4>는 두 개 비교 알고리즘을 사용한 단축경로의 변경율(X-축)에 따른 유지 비용(Y-축)을 보여주고 있다. 그림에서 알 수 있듯이 우리의 RS 기법은 w/o maintenance cost 에 비해 1.5, 3, 4.5를 모두 고려하면 평균적으로 54.1% 적은 유지 비용을 보여준다. 추세 그래프가 나타내듯이 두 가지 기법의 유지 비용의 차이는 변경율이 높아질수록 그 차이가 더 크다.

예를 들어 1.5%일 때는 그 차이가 41.8%였지만, 3%에서는 그 차이가 57.8%이고, 제일 큰 4.5%에서는 유지 비용의 차이가 더 커져 62.7%에 이른다. 이는 RS 기법이 단축 경로를 선택할 때 w/o maintenance cost처럼 단축 경로로 인한 단순 질의 처리 이득만 고려하지 않고 유지 비용까지 고려하여 이득을 계산하였기 때문이다. 이 실험은 유지 비용을 고려한 비용 모델을 채용한 단축 경로 선택 기법의 장점을 분명하게 보여주고 있다고 볼 수 있다.



〈Figure 4〉 Shortcut Maintenance Cost with Respect to Update Rate

5. 결 론

본 논문에서는 트리플 데이터베이스의 자기 조인을 수반하는 질의 처리 성능을 위한 단축 경로 선택 문제에 사용되는 이득 함수 모델을 제시하였다. 이득 함수 모델은 질의 처리 비용의 감소에 따른 이득뿐만 아니라 데이터베이스의 변경에 따른 비용도 반영해야 한다. 이를 위하여 본 논문에서는 이득 함수가 질의 처리 비용과 질의 빈도뿐 아니라 단축 결로 구성 비용 및 데이터베이스 변경율 등의 요소를 고려하도록 설계하였다. 설계 제시된 이득 함수는 그리디 기반의 단축 경로 선택 알고리즘에 적용 구현될 수 있고, 이득 함수의 각 구성 인자 요소가 트리플 데이터베이스 질의 처리 성능 및 단축 경로 유지에 어떠한 영향을 주는지를 분석하였다.

트리플 데이터베이스의 단축 경로는 질의 처리를 신속히 할 수 있다는 측면에서 데이터베이스 분야의 뷰나 인덱스와 같은 역할을 한다. 본 연구에서는 DBLP 데이터베이스에 한정하여 이득 함수 모델과 단축 경로 성능을 실험적으로 분석하였다. 실세계 전자상거래 상품정보 데이타베이스와 같은 다양한 도메인에 대한 적용은, 본 연구 모델의 확장성을 살펴보는 추후 연구 과제가 될 것이다.

References

[1] Abadi, D. J., Marcus, A., Madden, S. R., and Hollenbach, K., Scalable semantic web data management using vertical partitioning. In Proceedings of the 33rd international conference on Very large data bases (VLDB '07), pp. 411-422. VLDB Endowment, 2007.

[2] Abadi, D. J., Marcus, A., Madden, S. R., and Hollenbach, K., SW-Store : a vertically partitioned DBMS for Semantic Web data management, The VLDB Journal, Vol. 18, No. 2, pp. 385-406, 2009.

[3] Agrawal, S., Chaudhuri, S., and Narasayya, V. R., Automated Selection of Materialized Views and Indexes in SQL Databases. In Proceedings of the 26th International Conference on Very Large Data Bases (VLDB '00), pp. 496-505. Morgan Kaufmann Publishers Inc., 2000.

[4] Arias, M., Fernández, J. D., Martínez-Prieto, M. A. and de la Fuente, P., An Empirical Study of Real-World SPARQL Queries, In proceedings of the 1st International Workshop on Usage Analysis and the Web of Data (USEWOD2011) in the 20th International World Wide Web

Conference (WWW2011). 2011.

[5] Constantopoulos, P., Dritsou, V., and Foustoucos, E., Developing query patterns. In Proceedings of the 13th European conference on Research and advanced technology for digital libraries (ECDL'09), pp. 119-124. Springer-Verlag, 2009.

[6] Dritsou, V., Constantopoulos, P., Deligiannakis, A., and Kotidis, Y., Optimizing query shortcuts in RDF databases. In proceedings of the 8th extended semantic web conference on the semantic web : research and applications-Volume Part II (ESWC'11), pp. 77-92. Springer-Verlag, 2011.

[7] Huang, J., Abadi, D. J., and Ren, K., Scalable SPARQL Querying of Large RDF Graphs. In Proceedings of the VLDB Endowment, Vol. 4, No. 11, pp. 1123-1134, 2011.

[8] Kang, S., An Indexing Framework for Improving Data Consistency of Triple Database, Ph. D. Thesis, Seoul National University. 2013.

[9] Kang, S., Shim, J., and Lee, S.-g., Tridex : A Lightweight Triple Index for Relational Database-Based Semantic Web Data Management, Expert Systems with Applications, Vol. 40, No. 9, pp. 3421-3431, Elsevier, 2013.

[10] Karloff, H. and Mihail, M., On the complexity of the view-selection problem. In Proceedings of the eighteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (PODS '99), pp. 167-173. ACM, 1999.

[11] Lee, H., Shim, J., and Kim, D., Ontological modeling of e-catalogs using EER and description logics. In Proceedings of International Workshop on Data Engineering Issues in E-Commerce (DEEC'05), IEEE, 2005.

[12] Lee, M., Lee, H., and Shim, J., Analysis and Modeling of Semantic Relationships in e-Catalog Domain. The Journal of Society for e-Business Studies, Society for e-Business Studies, Vol. 9, No. 3, pp. 243-258, 2004.

[13] Ley, M. The DBLP computer science bibliography. http://www.informatik.uni-trier.de/~ley/db/. Nov 15, 2012.

[14] Page, L., Brin, S., Motwani, R., Winograd, T., The PageRank citation ranking : bringing order to the Web. In proceedings of the 7th International World Wide Web Conference, pp. 161-172. 1998.

[15] Scheuermann, P., Shim, J., and Vingralek, R. WATCHMAN : A data warehouse intelligent cache manager. In Proceedings of the 22th International Conference on Very Large Data Bases (VLDB '96), pp. 51-62. Morgan Kaufmann Publishers Inc., 1996.

저 자 소 개

강승석 (E-mail : sseok.kang@samsung.com)
2005년 서울대학교 컴퓨터공학부 (학사)
2007년 서울대학교 컴퓨터공학부 (석사)
2013년 서울대학교 컴퓨터공학부 (박사)
2013년~현재 삼성전자 무선사업부
관심분야 데이타베이스, 모바일, 시맨틱 웹

심준호 (E-mail : jshim@sookmyung.ac.kr)
1990년 서울대학교 계산통계학과 졸업 (학사)
1994년 서울대학교 계산통계학과 전산과학전공 (석사)
1998년 Northwestern University, Electrical and Computer Engineering (박사)
2001년~현재 숙명여자대학교 컴퓨터과학부 교수
관심분야 데이타베이스, 전자상거래, 상품정보, 온톨로지