韓國컴퓨터情報學會 論文誌
第16卷 第7號, 2011. 7.

2011-16-7-1-10

RDF 데이터에서 접미사 배열을 이용한 ρ-intersect 연산의 처리

김 성 완* 김 연 희**

Processing of ρ-intersect Operation on RDF Data Using Suffix Array

Sung Wan Kim* Youn-Hee Kim**

요 약

보다 신속하고 정확한 정보 검색에 대한 지능적이고 자동화 된 서비스 제공을 지향하는 시맨틱 웹 기술의 실제적 활용이 점점 구체화 되고 있다. 이에 시맨틱 웹상에서 존재하는 방대한 양의 데이터를 관리하기 위한 표준 포맷 중 하나로 널리 사용되는 RDF로 표현된 데이터에 대한 효율적인 질의 처리는 계속적인 중요한 연구 주제가 되고 있다. RDF 데이터에 대한 전형적인 질의 처리 유형은 임의의 리소스로부터 특정한 관계성을 갖는 리소스들을 검색하는 것으로 이에 대한 많은 연구들이 진행되어 왔다. 그러나, 기존의 연구들에서는 리소스간의 복잡한 관계성들의 발견(discovery) 즉, 질의 처리의 결과로 리소스간의 연관성을 반환하는 유형의 질의 처리에 대해서는 충분히 고려하지 않고 있다. 본 논문에서는 시맨틱 연관성 검색 유형의 하나인 ρ-intersect 연산의 처리를 위한 인덱싱 및 질의 처리 방안을 소개한다. 이를 위해 접미사 배열을 이용한 인덱싱과 ρ-intersect 연산의 특징을 고려한 최적화 처리 방안을 제안한다. 실험적 성능 평가는 기존 기법에 비해 제안 기법의 평균 실행 시간이 3~7배의 빠른 질의 처리 성능을 보인다.

▸ Keyword : RDF 데이터, 시맨틱 연관성, 접미사 배열, 질의 처리

Abstract

The actual utilization of Semantic Web technology which aims to provide more intelligent and automated service for information retrieval over the Web becomes gradually reality. RDF is widely used as the one of standard formats to present and manage the voluminous data on the Web. Efficient query processing on RDF data, therefore, is one of the ongoing research topics. Retrieving

• 제1저자 및 교신저자 : 김성완
• 투고일 : 2011.04.12, 심사일 : 2011.05.06, 게재확정일 : 2011.05.16.
* 삼육대학교 컴퓨터공학부(Divsion of Computer, Sahmyook University)
** 부천대학교 e-비즈니스과(Dept. of e-Business, Bucheon University)
※ 이 논문은 2011년 한국컴퓨터정보학회 제43차 동계학술대회에서 발표한 논문(“시맨틱 연관성 검색을 위한 ρ-intersect 연산의 처리”)을 확장한 것임

resources having a specific association from a given resource is the typical query processing type and several researches for this have done. However the most of previous researches have not fully considered discovering the complex relationship among resources such as returning the association between resources as the query processing result. This paper introduces the indexing and query processing for ρ-intersect operation which is one of the semantic association retrieval types. It includes an indexing scheme using suffix array and optimal processing approaches for handling ρ-intersect operation. The experimental evaluations shows that the average execution times for the proposed approach is 3~7 times faster than the previous approach.

▸ Keyword : RDF Data, Semantic Association, Suffix Array, Query Processing

I. 서 론

시맨틱 웹(Semantic Web)은 웹 상에 존재하는 데이터 혹은 리소스의 의미를 컴퓨터가 이해하고 처리할 수 있도록 구성된 데이터의 웹(Web of Data)의 구현을 궁극적인 목적으로 한다. 이러한 데이터의 웹을 실현하기 위해서는 방대한 양의 데이터를 표준화된 포맷으로 표현하고 접근하도록 하는 것이 중요하다. 또한 이들 데이터들 간의 관련성도 관리되어져야 한다. 이러한 목적을 위해 RDF(Resource Description Framework)가 표준 포맷 중 하나로 사용되어 왔으며, RDF로 표현된 방대한 양의 RDF 데이터에 대한 효율적인 관리 및 처리를 위해 저장 기법, 인덱싱 및 질의 처리 등에 대한 계속적인 연구가 진행되어 왔다[1][2][11][12].

한편, RDF로 표현된 데이터의 웹으로부터 정보를 검색하기 위해 SPAQRL 등의 RDF 질의어가 제안되었다. 대부분의 RDF 질의어에서 지원하는 전형적인 질의 형태는 임의의 리소스 A로부터 특정한 관계성 R을 갖는 모든 리소스들을 검색하는 것이다. 이러한 전형적인 질의는 관계성 R이 조인 조건 혹은 경로식 형태로 질의에 명세되어 진다.

한편, Anyanwu 등의 연구에서는 이러한 전형적인 질의 유형 외에도 질의 처리 결과로 관계성 R을 반환하는 새로운 질의 유형을 소개하고 이에 대한 중요성을 언급하였다[3][4]. 또한, 리소스들 간의 이러한 관계성 R을 시맨틱 연관성(semantic association)으로 정의하고 이에 대한 프레임워크를 제안하였다. 두 리소스 사이의 연관된 관계를 발견하는 질의는 항공 보안, 테러 방지 및 국가 보안 등의 업무에서 매우 중요한 유형의 질의 중 하나이다.

Matono 등은 전형적인 RDF 질의 처리를 위해 접미사 배열을 이용한 인덱싱 및 질의 처리 기법을 제안하였으나 시맨틱 연관성에 대한 처리는 고려하지 않고 있다[5][6]. 이에 본 논문에서는 기존의 Motono 등의 연구에서 제안된 접미사 배열을 이용한 인덱싱 기법을 기반으로 시맨틱 연관성 검색 유형 중의 하나인 ρ-intersect 연산의 처리 기법을 소개한다. 이를 통해 전형적인 RDF 질의 유형뿐만 아니라 시맨틱 연관성 질의 유형도 지원할 수 있도록 한다.

본 논문은 다음과 같이 구성된다. 2장은 관련 연구로서 시맨틱 연관성에 대한 개념과 접미사 배열을 이용한 RDF 데이터의 인덱싱 및 질의 처리 기법에 대해 서술한다. 3장에서는 시맨틱 연관성의 한 가지 유형인 ρ-intersect 연산의 처리를 위한 인덱싱 및 이에 대한 최적화된 질의 처리 방안을 제안한다. 4장에서는 제안 기법에 대한 실험적 성능평가에 대해 서술하며, 5장에서 결론 및 향후 연구에 대해 언급한다.

II. 관련 연구

1. 시맨틱 연관성

RDF에서는 웹상의 데이터들 간의 연관성을 표현하기 위해 <subject, property, object> 형태로 구성된 트리플을 최소 표현 단위로 사용한다[1]. 여기서 subject와 object는 데이터를 의미하는 리소스(resource)라 하며, property는 두 리소스간의 관련성을 표현한다. 여기서 object는 리소스 혹은 리터럴 값을 갖을 수 있다. 예를 들어 리소스 r1과 r4 사이에 'knows'라는 관련성이 있을 경우 <r1, knows, r4> 형태의 트리플로 표현할 수 있으며, r1의 이름이 'Hong'일 경우 <r1, name, 'Hong'> 형태의 트리플로 표현할 수 있다.

한편, RDF로 기술된 데이터는 <그림 1>과 같이 subject와 object를 노드로 하고 property를 간선으로 갖는 방향성 그래프 형태로도 표현될 수 있다. 예제를 단순화하기 위해 <그림 1>에서는 각 리소스에 대해 object를 리터럴로 갖는 경우는 생략하였다.

시맨틱 웹의 활성화에 따라 RDF로 표현된 데이터에 대한 질의 처리를 위해 SPARQL 등의 RDF 질의어가 제안되었

다. 초기에 제안된 대부분의 RDF 질의어에서는 리소스간의 복잡한 관계성들의 발견(discovery)을 위한 충분한 지원을 하지 못하고 있다[7][8]. RDF 질의의 전형적인 유형은 임의의 리소스 A로부터 특정한 관계성 R을 갖는 모든 리소스들을 질의 결과로 검색하는 것이다. 이러한 질의 유형은 반환되어야 할 리소스에 대한 조건 즉, 관계성 R에 대한 명세가 질의 조건으로 주어져야 하며, 관계성 R은 조인 조건 혹은 경로식으로 표현될 수 있다.

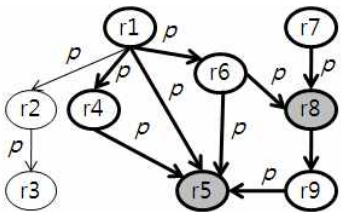

그림 1. 예제 RDF 그래프
Fig. 1. Example RDF Graph

한편, 또 다른 유형의 질의는 리소스들 간의 관계성 R을 검색 대상으로 하는 것이다[3][8]. 예를 들어 리소스 A와 리소스 B 사이에 어떠한 관계가 있는지 검색하는 경우, 두 리소스 사이의 연관성 R이 질의 결과로 반환되어져야 한다.

[3][4][7]의 연구에서는 리소스 사이에 발생할 수 있는 이와 같은 복잡한 관계성(complex relationships)을 시맨틱 연관성(semantic association)으로 정의하였다. 결국, 시맨틱 연관성의 발견(discovery)은 두 리소스 개체를 연결하는 특정한 의미를 지니는 경로를 검색하는 것이라고 할 수 있다.

[7]에서는 만일 RDF 그래프 상의 임의의 한 시퀀스 'e1, p1, e2, ..., en' 가 있다면 두 개체 e1과 en 사이에는 시맨틱 연관성이 있다고 정의하고 (여기서 ei와 pi는 리소스 개체와 프로퍼티를 각각 의미함), 리소스 개체와 프로퍼티가 교대로 구성된 이러한 시퀀스를 시맨틱 경로(semantic path)로 정의하였다. 또한, 시맨틱 연관성의 유형으로 ρ-intersect 연관성 등을 정의하였다. ρ-intersect 연관성은 두 개의 시맨틱 경로가 특정 리소스 상에 교차 되는 경우를 의미한다[3][8]. 예를 들어 <그림 1>의 RDF 그래프 상에서 리소스 'r1'과 'r7' 사이의 ρ-intersect 연관성은 {(r1.r4.r5, r7.r8.r9.r5), (r1.r5, r7.r8.r9.r5), (r1.r6.r5, r7.r8.r9.r5), (r1.r6.r8, r7.r8)} 등 4개가 존재한다 (위 <그림 1>에서 교차되는 노드는 음영으로 표시하였다). 이러한 유형의 질의는 항공 보안, 국가 보안, 범죄자 관리 등 같은 업무에서 매우 중요한 유형의 질의이다. 예를 들어, 탑승객 A와 B간 혹은 탑승객 A와 요주의 목록에 있는 기관 C와의 사이에 어떠한 공통적인 관계가 있는지 확인하고 그 관계가 보안 측면에서 문제가 있을 경우 즉각적인 조취를 취하여야 한다.

2. 접미사 배열을 활용한 인덱싱

Matono 등은 경로식으로 표현될 수 있는 RDF 데이터에 대한 효과적인 질의 처리를 위해 접미사 배열을 이용한 인덱싱 기법을 최초로 제안하였다[5]. 접미사 배열(suffix array)은 전통적으로 텍스트 데이터에서 주어진 문자열의 포함여부를 탐색하기 위해 정보검색분야에서 사용되는 자료구조로, 이진 탐색을 활용하여 특정 패턴을 효율적으로 검색할 수 있는 특징이 있다[9].

Matono가 제안한 기법에서는 RDF 데이터를 DAG(Directed Acyclic Graph)로 간주하고 경로 패턴들을 추출하였으며, 추출된 경로 패턴들은 변형된 접미사 배열을 이용하여 인덱스를 구축하는데 활용된다. 그러나 이 연구에서는 고정된 길이를 갖으며 경로를 구성하는 요소들이 빠짐없이 순차적으로 구성된 단순 경로식 만에 대한 질의 처리만을 고려하였다. 즉, 주어진 경로 패턴으로부터 도달 가능한 리소스를 검색만을 지원하도록 연구하였으며 시맨틱 연관성 질의 처리에 대해서는 고려하지 않고 있다. [6]에서는 접미사 배열의 이진 탐색 범위 축소를 통한 질의 처리 성능을 개선하기 위해 한 개의 인덱스 구조 대신 여러 개의 접미사 배열 기반의 인덱스를 사용하는 인덱싱 기법 및 질의 처리 방안을 제안하였으나 시맨틱 연관성 질의는 역시 고려하지 않았다.

III. 제안 기법

본 장에서는 효과적인 경로 기반의 질의 처리를 위해 [5][6]에서 제안한 것과 같이 접미사 배열을 사용한 인덱싱 기법을 활용하며, 시맨틱 연관성 특히, ρ-intersect 연산을 위한 최적화된 질의 처리 방법을 제안한다. 본 논문에서 RDF 데이터는 DAG 형태로 가정하며, RDF 그래프상에서 추출된 시맨틱 경로는 노드(즉, 리소스)들 만으로 구성된 것으로 정의한다. 예를 들어 <그림 1>과 같은 RDF 그래프로 부터 'r1.r2.r3'과 같은 경로 패턴들이 추출될 수 있다.

1. RDF 데이터의 인덱싱 방법

<그림 2>은 인덱스 생성 과정을 나타낸 것이다. 인덱스를 구성하기 위해서는 첫째, RDF 데이터로부터 모든 경로들을 추출하고 각 경로에 대해 경로 식별자(pid)를 할당한다. 경로 식별자는 추출된 각 경로를 유일하게 구별하기 위한 것으로 정수 값이 사용된다.

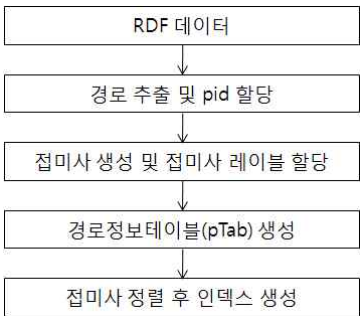

그림 2. 인덱스 생성 과정
Fig. 2. Index Generation Steps

둘째, 추출된 각 경로로 부터 접미사들을 생성한다. 예를 들어 <그림 1>로부터 추출된 경로 패턴 'r1.r2.r3' (pid는 1로 가정)로 부터 3개의 접미사 즉, 'r1.r2.r3', 'r2.r3', 'r3'가 생성된다. 이렇게 생성된 접미사들에 대해서는 경로 식별자(pid)와 인덱스 포인트(idx) 쌍으로 구성된 '접미사 레이블'을 할당한다. 인덱스 포인트는 한 경로 내에서 해당 접미사의 위치 값을 의미한다. 예를 들어, 경로 'r1.r2.r3'로부터 생성된 접미사 'r2.r3'는 접미사 레이블 (1, 2)가 할당 된다. 셋째, 위와 같이 구해진 경로와 접미사 그리고 할당된 접미사 레이블들을 테이블 형태로 표현하는데 이를 경로 정보 테이블(pTab)이라 한다. <그림 3>은 <그림 1>로부터 구해진 경로 정보 테이블이다.

idx / PID	1	2	3	4	5
1	r1	r2	r3		
2	r1	r4	r5		
3	r1	r5			
4	r1	r6	r5		
5	r1	r6	r8	r9	r5
6	r7	r8	r9	r5	

그림 3. 경로 정보 테이블 (pTab)
Fig. 3. Path Information Table (pTab)

마지막 단계는 인덱스를 생성하는 과정으로 추출된 접미사와 접미사에 할당된 접미사 레이블들을 접미사 패턴에 대해 사전 순으로 정렬한 후, 정렬된 접미사 레이블 값을 배열 요소의 값으로 갖는 인덱스 즉, 접미사 배열을 생성한다. <그림 4>는 추출된 접미사들을 사전 순으로 정렬하는 과정을 나타낸 것이며 최종적으로 얻어진 인덱스의 내용은 [(1, 1)(2, 1)(3, 1)(4, 1). . . . . (6, 2)(5, 4)(6, 3)]와 같다. 접미사 정렬 과정을 통해 유사한 패턴을 가지는 접미사들은 인덱스 상에서 물리적으로 서로 인접하는 특징을 가지게 된다.

접미사 (정렬 전)	접미사 레이블		접미사 (정렬 후)	접미사 레이블
r1.r2.r3	(1, 1)		r1.r2.r3	(1, 1)
r2.r3	(1, 2)		r1.r4.r5	(2, 1)
r3	(1, 3)		r1.r5	(3, 1)
r1.r4.r5	(2, 1)		r1.r6.r5	(4, 1)
r4.r5	(2, 2)		r1.r6.r8.r9.r5	(5, 1)
r5	(2, 3)		r2.r3	(1, 2)
r1.r5	(3, 1)		r3	(1, 3)
r5	(3, 2)		r4.r5	(2, 2)
r1.r6.r5	(4, 1)		r5	(2, 3)
r6.r5	(4, 2)	정렬 >	r5	(3, 2)
r5	(4, 3)		r5	(4, 3)
r1.r6.r8.r9.r5	(5, 1)		r5	(5, 5)
r6.r8.r9.r5	(5, 2)		r5	(6, 4)
r8.r9.r5	(5, 3)		r6.r5	(4, 2)
r9.r5	(5, 4)		r6.r8.r9.r5	(5, 2)
r5	(5, 5)		r7.r8.r9.r5	(6, 1)
r7.r8.r9.r5	(6, 1)		r8.r9.r5	(5, 3)
r8.r9.r5	(6, 2)		r8.r9.r5	(6, 2)
r9.r5	(6, 3)		r9.r5	(5, 4)
r5	(6, 4)		r9.r5	(6, 3)

그림 4. 접미사 정렬
Fig. 4. Sorting Suffixes

본 논문에서는 접미사 배열을 이용한 질의 처리 시 수반되는 이진 탐색 범위를 축소하여 질의 처리 성능을 높이기 위해 단일 접미사 배열을 사용하는 대신 <그림 5>와 같이 동일한 리소스로부터 시작하는 접미사 패턴들을 그룹핑 하여 각각의 독립적인 접미사 배열에 유지한다.

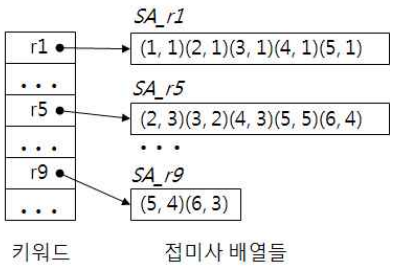

그림 5. 인덱스 구조
Fig. 5. Index Architecture Overview

<그림 5>에서 SA_ri는 리소스 ri로 시작하는 접미사 패턴들에 대한 접미사 레이블들을 유지하는 접미사 배열을 의미한다. 이를 통해 ri로 시작하는 접미사 패턴을 검색 시 탐색 범위를 전체 인덱스가 아닌 SA_ri만으로 헌정시킬 수 있다. 키워드 인덱스는 각 접미사 패턴들의 첫 번째 구성요소들로 이루어진 인덱스로 이를 통해 해당 SA_ri인덱스로 접근하게 된다.

2. ρ-intersect 연산의 처리

ρ-intersect 연산이 포함된 시맨틱 연관성 검색을 위한 질의는 두 개의 부분 경로 패턴이 주어지며, 각각의 부분 경로 패턴부터 시작하는 경로 상에 교차되는 리소스가 있는지 검색하는 것이다. 이 때, 가장 단순한 부분 경로 패턴은 단일 리소스만으로 구성된 경우이다. 예를 들어, <그림 1>로부터 두 개의 부분 경로 패턴 'r1.r6'와 'r7'에 대한 ρ-intersect 연관성을 검색하는 경우 {(r1.r6.r5, r7.r8.r9.r5), (r1.r6.r8, r7.r8)}을 검색하게 된다.

ρ-intersect 연관성 검색을 위한 가장 직관적인 처리 방법은 질의에 주어진 각 부분 경로 패턴으로 부터 시작되는 완전한 경로들을 모두 추출한 후, 이 경로들 간에 교차되는 노드가 있는지 상호 검색하는 것이며, 이 때 경로를 구성하는 모든 노드들을 대상으로 교차 여부를 확인하는 것이다.

질의 처리 성능 향상을 위해 본 논문에서 제안하는 기본 아이디어는 진입 차수가 1인 이하인 노드는 절대로 교차(intersect)될 수 없다는 사실을 활용하여 질의 처리 범위와 대상을 축소시키는 것이다. 이러한 기본 아이디어의 적용을 위한 선행 작업으로 진입 차수가 2이상인 노드들을 별도로 식별해 둔다. 즉, <그림 3>의 경로 정보 테이블(pTab)에 진입 차수가 2이상인 노드에 대해서는 별도로 표기 한다(편의상 그림에서는 r5와 r8과 같이 볼드체로 표기). 진입 차수가 2이상인 노드의 식별은 트리플 집합으로 구성된 RDF 데이터에서 object 필드 값이 동일한 트리플들이 두 개 이상인지 검색하면 쉽게 찾아 낼 수 있다.

<그림 6>은 본 논문에서 제안한 질의 처리 알고리즘을 나타낸 것으로 ProcessingRhoIntOperation 함수부터 질의 처리 단계가 시작된다. 이 함수의 처음 실행 단계는 사용자 질의에 주어진 두 개의 부분 경로 패턴을 인수로 하여 GetSuffixLabel 함수를 각각 호출한다. GetSuffixLabel 함수는 인수로 주어진 부분 경로 패턴과 일치하는 접미사들에 대한 접미사 레이블 집합을 구한다.

GetSuffixLabel 함수의 첫 번째 단계에서는 접미사 배열 SA_ri에 대한 이진 탐색과 경로 정보 테이블(pTab)을 활용하여 인수로 주어진 부분 경로 패턴과 최초로 일치되는 접미사를 찾아 그 접미사 레이블을 결과 집합에 포함 시킨다 (이 때 찾아진 배열 요소의 위치 값을 p라 하자). 두 번째 단계는 위치 값 p의 좌우에 인접한 배열 요소들을 대상으로 부분 경로 패턴과 일치되는 나머지 접미사 패턴들을 찾아 그 접미사 레이블들을 결과 집합에 추가한다. 예를 들어, 인수로 주어진 부분 경로 패턴이 'r1.r6'과 'r7'일 경우 접미사 레이블 집합 {(4, 1), (5, 1)}와 {(6, 1)}을 결과 집합으로 각각 얻게 된다.

ProcessingRhoIntOperation 함수의 그 이후 실행 내용은 GetSuffixLabel 함수의 실행 결과로 부터 얻어진 두 개의 접미사 레이블 집합 A와 B에 포함된 각각의 접미사 레이블을 대상으로 해당 접미사 레이블로 부터 시작하는 접미사 패턴 상에서 교차되는 노드의 유무를 반복적으로 상호 확인 및 평가하는 과정으로 접미사 패턴을 구성하는 각 구성 요소를 차례로 비교한다. 예를 들어, 접미사 레이블 (4, 1)과 (6, 1)로부터 시작하는 접미사 패턴 r1.r6.r5와 r7.r8.r9.r5에서 r5를 교차되는 노드로 찾게 된다.

본 논문에서 제안하는 ρ-intersect 연산 처리를 위한 최적화 방안은 다음과 같이 2 가지이다. 첫째, GetSuffixLabel 함수에서 인수로 주어진 부분 경로 패턴과 일치되는 접미사에 대한 접미사 레이블을 찾을 때, 이 접미사 레이블 이후부터 시작하는 경로 상에 진입 차수가 2개 이상인 노드를 포함하지 않는 경우 이 경로는 교차 가능성이 없으므로 해당 접미사 레이블은 결과 집합에 포함시키지 않도록 필터링 하는 것이다. 이 때, 주어진 부분 경로 패턴 이후의 경로 상의 진입 차수가 2개 이상인 노드의 포함 여부는 pTab을 이용하여 확인할 수 있다. 예를 들어, 주어진 부분 경로 패턴이 'r1'일 경우, 접미사 레이블 (1, 1)은 그 이후부터 시작하는 경로 패턴 즉, 'r2.r3'에는 진입 차수가 2이상인 노드가 없으므로 결과 집합에 포함되지 않게 되며, 최종적으로 {(2, 1), (3, 1), (4, 1), (5, 1)}을 결과 집합으로 반환하게 된다. 만일 최적화 방안을 적용하지 않을 경우 {(1, 1), (2, 1), (3, 1), (4, 1), (5, 1)}을 결과 집합으로 반환한다.

둘째, <그림 6>의 ProcessingRhoIntOperation 함수의 처리 부분 중 교차 노드 확인을 위한 반복적 처리 및 평가 과정에서 접미사 패턴을 구성하는 각 구성 요소들 간의 상호 교차 여부를 평가하게 되는데, 이때 진입 차수가 2이상이 아닌 노드는 평가 대상에서 제외하여 질의 처리 대상을 축소하도록 한다. 접미사 패턴을 구성하는 각 노드의 진입 차수가 2이상인지의 확인은 pTab 테이블에서 쉽게 확인할 수 있다.

예를 들어, 접미사 레이블 (2, 1)로 시작되는 경로를 pTab으로 부터 추출하면 'r1.r4.r5'이 되며, r1과 r4는 진입차수가 2미만 이므로 교차 여부 확인을 위한 반복 처리 과정에서 평가 대상에 포함되지 않고 r5만이 평가된다. 접미사 레이블 (2, 1)과 (6, 1)로부터 추출되는 경로에 대해 제안 방법에 따라 진입 차수가 2이상인 노드만을 대상으로 평가를 계속 진행하면 두 번의 교차 여부 확인만을 통해 r5를 교차 노드로 구할 수 있다.

```
Function GetSuffixLabel(QueryPattern)
  // 입력 : 질의 패턴 QueryPattern
  // 출력 : 접미사 레이블의 집합 tempSet

  1단계) 접미사 배열 SA_i 상에서 QueryPattern에 첫 번째로 일치하는 접미사에 대한 배열 요소 위치 값 p를
         검색 후 SA_i[p]의 내용(즉, 접미사 레이블 값)을 결과 집합 tempSet에 추가
  2단계) QueryPattern에 일치되는 접미사들에 대한 추가 검색
         2-1) 위치 p를 기준으로 SA_i 의 좌측 배열 요소 중에 QueryPattern에 일치되는 접미사들에 대한
              접미사 레이블들을 추가 검색하여 tempSet에 추가
         2-2) 위치 p를 기준으로 SA_i 의 우측 배열 요소 중에 QueryPattern에 일치되는 접미사들에 대한
              접미사 레이블들을 추가 검색하여 tempSet에 추가
End  Function

Function ProcessingRhoIntOperation(usrQueryPattern)
// 입력 : 사용자 질의 패턴 usrQueryPatttern
// 출력 : 교차하는 노드 집합

Call Function GetSuffixLabel(usrQueryPatternA)  // 접미사 레이블 집합 A를 구함
Call Function GetSuffixLabel(usrQueryPatternB)  // 접미사 레이블 집합 B를 구함

// 집합 B에 포함된 접미사 레이블 개수가 집합 A에 포함된 접미사 레이블 개수보다 적을 경우 A와 B를 상호 교환

Foreach S in A  //  S는 접미사 레이블 (spid, sidx)을 의미함
  While ( R ≠ Null AND Indegree(R) >= 2 )  //  R은 pTab[spid][sidx]의 값인 리소스를 의미함
      Foreach T in B  //  T는 접미사 레이블 (tpid, tidx)를 의미함
         While ( Q ≠ Null AND Indegree(Q) >= 2 )  // Q는 pTab[tpid][tidx]의 값인 리소스를 의미함
              If R = Q  Then // 교차 여부
                  최종 결과 집합에 해당 리소스를 추가
                  Exit While
              End If
              tidx ← tidx+1
          End While
      End For
      sidx ← sidx+1
  End While
End For
End Function
```

그림 6. 질의 처리 알고리즘
Fig. 6. Query Processing Algorithm

IV. 실험 및 성능평가

본 장에서는 3장에서 제안한 p-intersect 시맨틱 연관성 질의 처리 방법에 대한 실험적 성능평가 내용에 대해 기술한다. 이를 위해 2가지 방법을 비교하였다. 첫째, 2.2절에서 소개한 Matono의 인덱싱 기법을 기반으로 최적화 방안을 적용하지 않은 직관적 질의 처리 방법과 둘째, 3.1절에서 언급한 개선된 인덱싱 기법을 기반으로 본 논문에서 제안한 최적화된 방법을 적용한 방법을 직접 구현하여 실험 및 평가하였다. <그림 7>은 구현된 시스템의 대략적인 구성도를 나타낸 것이다. 데이터 해석기 및 인덱스 생성기는 3.1절에서 설명한 바와 같이 데이터 셋으로부터 경로 및 접미사 추출, 접미사 레이블 할당, 경로 정보 테이블 pTab구성 그리고 접미사 배열을 기반으로 한 인덱스를 생성한다.

질의 처리기는 사용자로부터 입력된 p-intersect 시맨틱 연관성 질의를 처리하는 모듈이다. 이때, 데이터베이스에 저장된 인덱스 정보와, 경로 정보 테이블을 활용하여 3.2절에 설명한 질의 처리 알고리즘을 수행한다.

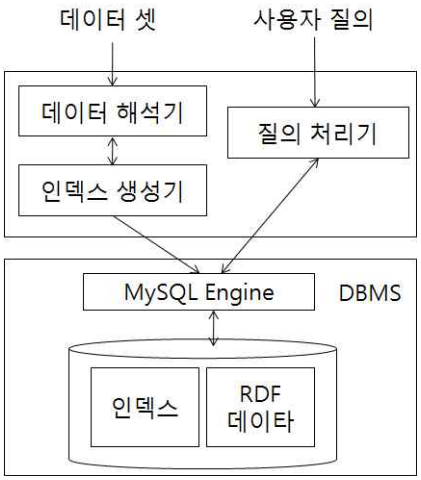

그림 7. 시스템 구성도
Fig. 7. System Organization

실험에 사용된 실험 데이터는 FOAF project에서 제공하는 FOAF 온톨로지 기반의 RDF 데이터[10]를 DAG 형태로 변형하여 사용하였으며, 크기가 다른 두 개의 데이터 셋을 사용하였다. 다음 <표 1>은 두 개의 실험용 데이터 집합으로부터 추출된 트리플 개수, 경로 개수, 접미사 개수를 나타낸 것이다.

표 1. 실험용 데이타셋
Table 1. Experimental Data Set

	D1	D2
데이터 크기(KB)	2,000	10,000
경로 수	1,314	6,570
접미사 수	14,874	74,370

시스템 구현 및 실험은Visual C++ 6.0과 MySQL 5.0을 사용하였으며, 1GB 메모리, 300GB HDD가 설치되고 Window XP Professional이 설치되어 있는 Intel Core2 Duo 2.20GHz CPU 기계에서 실험을 수행하였다. MySQL 데이터베이스의 캐시는 기본 설정 크기인 16M를 사용하였다.

성능평가를 위해 다음 <표 2>와 같은 4가지 실험용 질의 유형을 사용하였다. 여기서, p-intersect 길이는 질의에 주어진 두 개의 부분 경로 패턴으로부터 상호 교차되는 노드들까지의 경로의 길이를 의미한다. ,결과 개수는 교차되는 노드의 개수를 의미하며, p-intersect 값이 클수록 교차 노드의 평가를 위한 경로들의 개수가 많게 되어 그 만큼 질의 처리 복잡성이 커지게 된다.

표 2. 실험용 질의
Table 1. Experimental queries

유형	질의 특징		
	p-intersect 길이	결과 개수	복잡성
Q1	short	1	단순
Q2	mid	2	보통
Q3	mid	3	보통
Q4	long	12	높음

다음 <표 3>은 ProcessingRhoOperation 함수의 처음 부분에서 GetSuffixLabel 함수를 2회 호출 후 결과로 반환된 접미사 레이블 집합A와 B에 각각 포함된 접미사 레이블의 개수를 각 질의 유형 별로 나타낸 것이다. 전 장에서 언급한 것과 같이 제안기법에서는 첫 번째 질의 처리 최적화 방안 적용한 결과 교차 가능성이 없는 접미사 레이블들을 필터링하여 교차 평가 대상이 되는 접미사 레이블 집합의 크기를 축소시켰음을 알 수 있다.

표 3. 교차 여부 평가를 위한 접미사 레이블 개수
Table 1. Number of Suffix labels for Evaluating Intersect

	Matono의 기법		제안기법	
	집합A	집합B	집합A	집합B
Q1	17	72	7	28
Q2	41	84	7	80
Q3	40	160	25	136
Q4	142	161	87	98

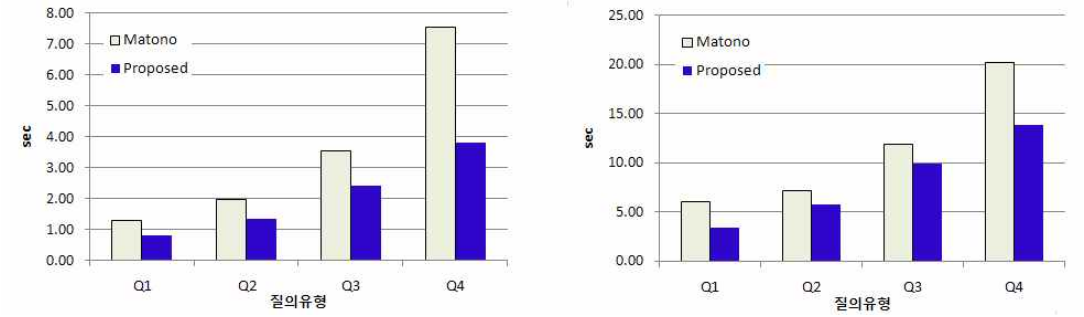

그림 8. 최초실행시간(좌측:D1, 우측:D2)
Fig. 8. Initial Execution Time (Left:D1, Right:D2)

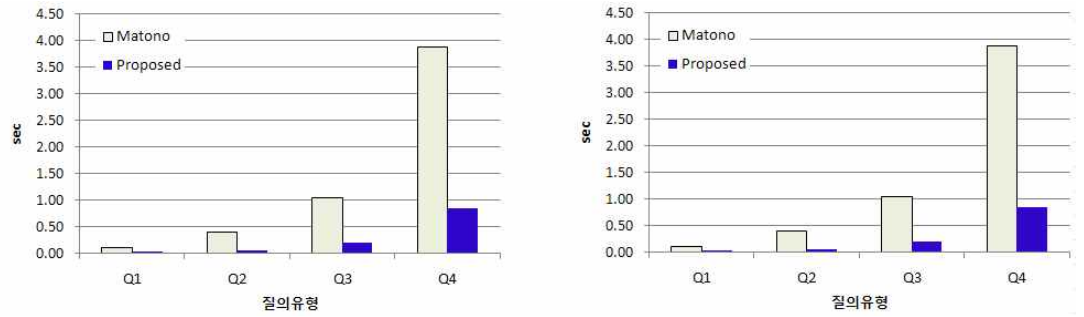

그림 9. 평균실행시간(좌측:D1, 우측:D2)
Fig. 9. Average Execution Time (Left:D1, Right:D2)

실험 질의 유형들에 대한 질의 수행 시간은 최초 실행 시간과 최초 실행 이후의 평균 실행 시간으로 나누어 측정하였다. 최초 실행 시간은 질의 처리 성능의 향상을 위해 데이터베이스가 자체적으로 활용하는 캐시 기능이 아직 사용되지 않은 경우를 의미한다. <그림 8>은 실험 데이터셋 D1과 D2에 대한 최초 실행 시간을 각각 보여주고 있다. 4가지 실험 질의 유형 모두에 대해 최적화 방안을 적용한 제안 기법이 약 1.2~2배 정도의 우수한 질의 처리 성능을 보이고 있다.

<그림 9>는 실험 데이터셋 D1과 D2에 대한 평균 실행 시간을 각각 보여주고 있다. 각 평균 실행 시간은 최초 실행 이후의 10회의 실행 시간을 평균 낸 값으로 데이터베이스의 캐시 기능의 효과의 영향을 받고 있다. 이로 인해 각 질의 처리 유형에 대해 최초 실행 시간보다 질의 처리 시간이 크게 단축됨을 볼 수 있다. 또한, Matono의 방법과 제안 방법과의 성능 차이가 최초 실행 결과에서 보다 큰 차이가 나고 있으며, 최적화 방안을 적용한 제안 기법이 4가지 실험 질의 유형 모두에 대해 Matono의 방법에 비해 3~7배 정도의 우수한 질의 처리 성능을 보였다.

한편, 데이터베이스 캐쉬 효과로 인해 데이터 셋의 크기가 큰 D2에 대한 실행시간이 D1에 대한 실행시간보다 근소하게 더 소요되었으나 큰 차이를 보이지 않았다.

V. 결 론

본 논문에서는 ρ-intersect 연산을 기반으로 하는 시맨틱 연관성 검색에 대한 처리 기법에 대해 제안하였다. 제안 기법

에서는 접미사 배열을 이용한 인덱싱과 p-intersect 연산의 특징을 고려한 최적화 방법을 활용하여 질의 처리 성능을 향상하도록 하였다. 또한, 4가지의 질의 유형에 대한 실험적 성능 평가를 통해 제안된 질의 처리 최적화 방안을 적용한 방법이 기존의 방법보다 최초 실행 시간의 경우 1.2~2배의 우수한 질의 처리 성능을 보였으며, 평균 실행 시간의 경우 3~7배의 우수한 질의 처리 성능을 보였다.

본 논문에서 제안된 기법은 전형적인 RDF 질의 유형 뿐만 아니라 다양한 시맨틱 연관성 검색을 위한 기반 요소 기술로 활용될 수 있을 것이다.

참고문헌

[1] W3C, RDF Primer, http://www.w3.org/TR/rdf-pr imer

[2] T. Tran. and G. Ladwig, "Structure Index for RDF Data," Proc. of the Workshop on Semantic Data Management(SemData@VLDB), Sept. 2010.

[3] K. Anyanwu, A. Sheth, "ρ-Queries: Enabling Query ing for Semantic Associations on the Semantic Web," Proc. of Int'l Conf. on WWW, pp.690-699. 2003,

[4] B. Aleman-Meza et al. "Ranking Complex Relationshi ps on the Semantic Web," IEEE Internet Computing, Vol. 9, No. 3, pp. 37-44, 2005.

[5] A. Matono, et al., "An Indexing Scheme for RDF and RDF Schema based on Suffix Arrays," First Int'l Workshop on SWDB, pp.151-168, Sept. 2003.

[6] S. Kim, "Improved Processing of Path Query on RDF Data Using Suffix Array," Journal of Convergence Information Technology, Vol. 4, No. 3, pp.45-52. 2009.

[7] A. Sheth et al, "Semantic Association Identification and Knowledge Discovery for National Security Applications," Journal of Database Management, Vol 16, pp.33-53, 2005.

[8] K. Kochut and M. Janik, "SPARQLeR: Extended Sp arql for Semantic Association Discovery," LNCS, Vol. 4519, Proc. of the 4th European Conf. on The Semantic Web, pp. 145-159, 2007.

[9] Willlam B. Frakes and Richard Baeza-Yates, "Infor mation Retrieval : data sructures & alogorthms," Sigma Press, 1995.

[10] The Friend of a Friend (FOAF) project, http://ww w. fo af-pr oject.com

[11] Soonmi Lee, "Design of Relational Storage Schema and Query Processing for Semantic Web Documents," Journal of the Korea Society of Computer and Information, Vol.14, No.1, pp.35-45, January 2009.

[12] Youn-Hee Kim, and Ji-Hyun Kim, "The Scheme for Path-based Query Processing on the Semantic Data," Journal of the Korea Society of Computer and Information, Vol.14, No.10, pp.31-41, October 2009.

저 자 소 개

김 성 완

1998: 홍익대학교 전자계산학과 이학석사.
2003: 홍익대학교 전자계산학과 이학박사
1999~2005: 삼육의명대학 컴퓨터정보과 조교수
2006~현재: 삼육대학교 컴퓨터학부 부교수
관심분야: XML 및 웹 데이터베이스, 정보처리, 시맨틱 웹
Email : swkim@syu.ac.kr

김 연 희

2000: 홍익대학교 컴퓨터공학과 공학사.
2002: 홍익대학교 컴퓨터공학과 공학석사.
2006: 홍익대학교 컴퓨터공학과 공학박사
현 재: 부천대학교 e-비즈니스과 강의전담교수
관심분야: 시맨티 웹, XML, 분산 데이터베이스
Email : yhkim@bc.ac.kr