Journal of The Korea Society of Computer and Information
Vol. 18, No. 12, December 2013
www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2013.18.12.091

관계형 데이터베이스에서의 시맨틱 기반 키워드 탐색 시스템

양 영 휴*

Semantic-based Keyword Search System over Relational Database

YoungHyoo Yang*

요 약

키워드의 모호성은 효율적인 키워드 탐색에 있어서 일반적인 이슈가 되어왔는데, 이 모호성은 탐색결과의 신뢰성에 큰 영향을 줄 수 있으며, 기본적으로 질의에 사용된 용어 자체가 가지는 문맥상 의미의 모호함에 기인한다. 질의 자체의 모호함뿐만 아니라, 사용자들이 그 탐색 결과를 적절하게 해석하기 위해 결과에 나타나는 키워드간의 관계도 중요하므로 명확하게 명시 되어야 한다. 이 논문에서는 기존의 질의 용어와 스키마 용어/인스턴스간의 키워드 매핑기법을 적용하여 키워드 탐색의 모호성을 해결한다. 용어간의 매핑에서는 질의 키워드와 스키마 용어간의 구문적 유사성은 물론 시맨틱 유사성까지 고려하기 때문에 기존의 시스템에 비해 매핑과 정밀도가 50% 이상 상승하는 결과를 얻을 수 있다. 탐색결과에 나타나는 용어간의 불분명한 관계를 점 더 명확하게 나타내기 위하여 시맨틱 웹 기술을 적용하여 키워드간의 의미 있는 관계를 더 많이 지식베이스 내에서 찾을 수 있도록 하였다.

▸ Keywords : 시맨틱 웹, 관계형 데이터베이스/SQL, 키워드 탐색, RDF, SPARQL

Abstract

One issue with keyword search in general is its ambiguity which can ultimately impact the effectiveness of the search in terms of the quality of the search results. This ambiguity is primarily due to the ambiguity of the contextual meaning of each term in the query. In addition to the query ambiguity itself, the relationships between the keywords in the search results are crucial for the proper interpretation of the search results by the user and should be clearly presented in the search results. We address the keyword search ambiguity issue by adapting some of the existing approaches for keyword mapping from the query terms to the schema terms/instances. The

•제1저자 : 양영휴
•투고일 : 2013. 12. 3, 심사일 : 2013. 12. 16, 게재확정일 : 2013. 12. 29.
* 한양여자대학 정보경영과(Dept. of Information Management, Hanyang Women's University)
*본 연구는 한양여자대학교 2012학년도 교내연구비 지원에 의해 수행되었음.

approaches we have adapted for term mapping capture both the syntactic similarity between the query keywords and the schema terms as well as the semantic similarity of the two and give better mappings and ultimately 50% raised accurate results. Finally, to address the last issue of lacking clear relationships among the terms appearing in the search results, our system has leveraged semantic web technologies in order to enrich the knowledgebase and to discover the relationships between the keywords.

▸ Keywords : Semantic Web,Relational Database/SQL, Keyword Search, RDF, SPARQL.

I. 서 론

시맨틱 웹 온톨로지 언어인 OWL, RDF/S는 웹에 존재하는 동영상, 이미지, 텍스트 등과 같은 웹 자원을 기술하는 웹 서술 언어로서 널리 사용되고 있다. 이와 더불어 XML 호환성으로 인한 온톨로지 공유의 표준을 제공할 수 있기 때문에, OWL, RDF/S의 사용은 웹뿐만 아니라 다른 온톨로지 영역으로 확장되고 있다. 시맨틱 웹 온톨로지 사용의 증가로 인한 온톨로지 데이터베이스의 용량이 방대해짐에 따라 효율적인 시맨틱 질의 처리가 요구된다. 이에, 관계형 데이터베이스에서의 키워드 탐색의 효율성에 관한 다양한 연구가 진행 되어 왔다 [1][2][3][4].

관계형 데이터베이스가 복잡해지는 한편 사용자는 전통적 SQL 질의보다는 좀 더 간편한 키워드 탐색을 선호하여 주로 사용하는 경향이 있다. 그러나 이 간편함 속에는 모호성이 내재 되어있다. 관계형 데이터베이스에서 키워드 질의에 대한 답을 구하기 위해서는, 이 키워드들이 포함되어 있는 데이터베이스의 구조를 발견하고 그 구조들의 상호 연관관계를 찾아내어 답을 만들어 내야 하는 것이다. 찾아낸 구조들은 그들의 상호관계와 함께 사실상 키워드 질의의 시맨틱 해석을 관계형 용어로 나타내는 것이라 할 수 있다.

기존의 키워드 질의 처리 접근방식들은 키워드와 스키마 용어를 매핑 할 때 시맨틱 유사성과 구문적 유사성은 고려하지 않으면서 다만 정확한 매치만을 하려 하였고, 더욱 중요한 것은 탐색 결과 속 용어들 간의 관계를 표현하지 않았으므로 사용자들은 원하는 탐색결과를 얻지 못했었다.

이 논문에서는 탐색결과의 질을 높이고 탐색의 효율을 강화하기 위해서 새로운 시스템 프로토타입을 제안한다. 이 시스템에서는 시맨틱 웹 기술들에 관한 이슈들과 질의 용어를 스키마 용어들로 키워드 매핑하는 기법들을 사용하였다. 이 매핑 기법들은 질의 용어와 스키마 용어들 사이의 구문적 유사성과 시맨틱 유사성 양쪽을 다 고려하여 적용한다. 즉, 관계형 데이터베이스의 스키마와 인스턴스 들이 주어졌을 때, 클래스를 나타내는 테이블들과 그 클래스간의 관계를 표현하는 스키마로부터 온톨로지를 추출해 낸다. 그 다음에 추출된 온톨로지와 관계형 데이터베이스의 인스턴스들로부터 RDF와 같은 지식베이스를 생성해낸다. 클래스, 클래스간의 관계 그리고 RDF내의 클래스 인스턴스들은 리소스를 나타낸다. 사용자의 키워드 질의가 입력되면, 질의 용어와 지식베이스 내에 있는 자원간의 유사성을 계산하여 질의용어와 가장 매핑이 잘 되는 지식베이스 용어를 찾아낸다. RDF 지식베이스에서 후보 자원을 찾은 다음, 대응되는 SPARQL 질의를 생성하고 실행하여, 명확하게 정의된 용어들 간의 관계까지도 포함하는 질의에 대한 탐색 결과를 사용자에게 제공한다.

II. 관련 연구

키워드 탐색의 인기는 꾸준히 증가하는 추세이다. 20년이 넘도록 텍스트 데이터베이스와 웹에서 IR의 검색 모델로서 가장 성공적인 방법이 키워드 탐색이다[5][6]. XML과 같은 데이터 관리 커뮤니티에서도 환영받는 모델이 되었다[7]. XML의 경우 기본 정보 모델이 IR과 같은 '문서'라는 이점을 갖게 되는데, 그럼으로써 다양한 IR 기법들이 XML에도 적용될 수 있다. 반면에 관계형 데이터베이스에서의 키워드 탐색은 좀 더 도전적인 면이 있으며, 이미 여러 연구에서 흥미로운 시스템들이 제안 되고 있다. 대표적인 시스템으로는 DISCOVER와 DBXplorer와 같은 시스템이 있다. 전형적

기법으로는 데이터베이스 내용에 특별 색인을 만든 다음, 그 색인으로 속성 값 내에 존재하는 질의 키워드를 식별하는 방법이 있다. 이때에는 주로 역 색인을 사용하는 한편, DBXplorer 시스템은 심볼 테이블을 사용한다. 심볼 테이블은 (value, attribute, relation)의 집합으로 구성되는 색인으로, 각 값을 스키마 정보와 매핑 하는데 사용된다. 키워드의 위치를 찾아내면 연관된 튜플들을 찾아내는 다양한 방법을 통해서 튜플들의 조이닝 네트워크 혹은 튜플 트리가 형성되어, 이 네트워크나 트리가 질의의 결과로서 사용자에게 전달되는 정보 단위가 되기도 한다. 조이닝 네트워크는 인스턴스들로 부터 직접 만들어지거나 혹은 질의를 수행한 결과로 만들어 진다. DISCOVER 시스템은 완전하면서도 최소인 조이닝 네트워크를 찾는데 중점을 둔다. 조이닝 네트워크는, 적어도 하나의 튜플에 각각의 키워드 질의가 포함 되어 있으면 완전하다고 하며, 만일 하나의 튜플을 삭제 했을 때 그 네트워크가 더 이상 완전하지 않으면 최소의 네트워크라 한다.

이 논문에서의 목표는 최근 발표된 각기 다른 기법에서처럼 관계형 데이터베이스에서 키워드 탐색의 문제를 해결하려는 것은 아니다. 기본 목표는 탐색의 효과를 강화하기 위해서 관계형 데이터베이스의 시맨틱 탐색 기술을 재정비하는데 있다. 관계형 데이터베이스에서의 시맨틱 기반 키워드 탐색 분야의 연구는 활발하지 않은 편인데, SPARK 시스템은 질의 용어를 지식베이스 용어로 매핑 함으로써 사용자의 키워드 질의를 SPARQL로 변형한 다음 질의 형식을 생성한다. KEYMANTIC 시스템은 데이터베이스 내의 인스턴스 들에 대한 사전 정보가 없어도 질의 용어에서 스키마 용어로의 최적의 매핑을 찾는 문제를 해결하려고 한다. 대부분의 관계형 데이터베이스에 대한 키워드 탐색 기법들은 SPARK, KEYMANTIC 그리고 XSEEK 시스템들의 영향을 받은 것들이다[7][8][9].

III. 키워드 질의결과의 생성

관계형 데이터베이스에서 키워드 탐색을 할 때에는 데이터베이스내의 모든 용어들에 대해 만들어지는 마스터 색인을 생성하여 수행하는 방법이 주로 사용되어 왔다. 키워드 질의가 도착하면, 속성 값들 중에 그 키워드를 최소한 한번은 포함하고 있는 테이블과 튜플(레코드)들을 찾아내기 위해 이 마스터 색인을 먼저 검색한다. 검색된 튜플 집합과 관계형 데이터베이스의 스키마 그래프를 함께 검색하여 조인 트리를 찾아낸다. 조인 트리는 기본 키와 외래 키 관계로 조인되어 있어, 데이터베이스 내에 상호 연결되어 있는 튜플들의 집합을 의미한다. 마지막으로, 조인 트리내에 상호 연결 되어 있는 레코드들은 모든 질의 키워드를 포함하고 있으므로, 탐색 결과로 나타낸다. 다음의 예제를 통해 질의 결과 생성 기법을 볼 수 있다. 〈보기 1〉 다음 그림 1.과 표 1. 은 인터넷 영화 데이터베이스(IMDB)의 스키마와 인스턴스이다.

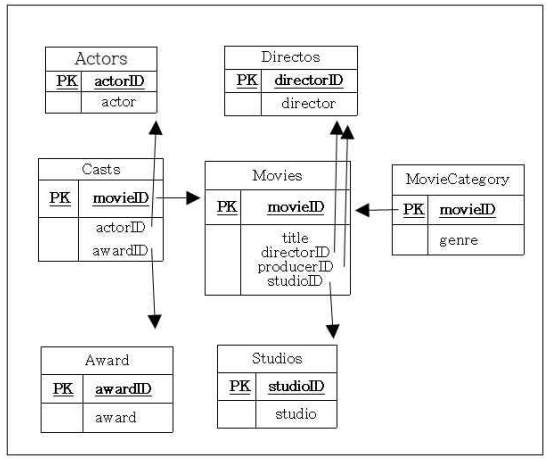

그림 1. IMDB 스키마 일부
Fig. 1. Portion of IMDB Schema

만일 사용자가 George Clooney가 감독하고 출연도 한 영화를 검색하기를 원한다면, Q = {movies directed starred George Clooney}과 같은 형식의 키워드 질의를 만들 것이다. 전통적인 관계형 데이터베이스에서의 키워드 검색에서는 두 가지 경우를 고려한다. 1) 논리 연산자 OR이 사용 된다면 1,2,4번의 열이 검색 결과가 될 것이고, 2) 논리 연산자 AND가 사용 된다면, 모든 질의 용어가 탐색 결과내에 모두 들어 있어야 하므로 검색 결과는 없음으로 될 것이다. 1)의 경우에는 심지어 데이터베이스 안에서 찾은 용어는 "George Clooney" 밖에 없으며, 질의에 있는 다른 키워드들은 찾을 수 없으므로 무시하게 된다. 그럼으로 사용자는 본래의 질의와는 무관하면서 양만 많은 의미 없는 결과를 돌려받게 되는 것이다. 또한 어떤 랭킹 스킴을 사용하느냐에 따라 실제 원하는 탐색 결과는 상위 k(k: 사용자의 의도에 부합되는 임계값)안에도 못 들어올 수도 있다.

이 논문에서는 이런 문제점을 보완하기 위하여 키워드 매핑 기술에 시맨틱 웹 기술을 접목하여, 키워드의 구문적 유사성외에 시맨틱 유사성까지 포함, 적용하여 탐색함으로써 좀 더 의미 있는 탐색 결과를 얻을 수 있도록 하였다.

표 1. IMDB의 인스턴스
Table 1. An Instance of IMDB

No.	Actor	Title	Studio	Director	Category
1	George Clooney	Ocean's Twelve	Village Roadshow Pictures	Soderbergh	Crime
2	George Clooney	Ocean's Eleven	Village Roadshow Pictures	Soderbergh	Crime
3	Zoe Saldana	Star Trek	Spyglass Entertainment	J. Abrams	Science Fiction
4	George Clooney	Leatherheads	Smokehouse Pictures	George Clooney	Comedy
5	Arnold Schwarzeneg ger	The Terminator	Hamdale Film	James Cameron	Action
6	R. Redford	Spy Game	Beacon Pics	T. Scott	Action

예를 들면, 앞의 키워드 질의에서 "George Clooney"라는 이름이 각기 다른 "acting"과 "directing"이라는 속성을 가지고, actor와 director 두 곳에 다 나타나야 한다는 사실을 이용하여 전반적으로 사용자의 키워드 질의와 상관있는 결과를 찾게 된다. 실제로 키워드 "directed" 혹은 "starred in"과 같은 키워드는 데이터베이스 인스턴스에 존재하지 않는다. 이를 해결하기 위하여 키워드 매핑 기술과 시맨틱 웹 기술을 함께 사용하는 것이다.

보기에서 주어진 키워드 질의에 대한 바람직한 결과는 "George Clooney"라는 키워드를 포함한 결과보다는 "George Clooney" 라는 키워드에 대한 각기 다른 성질(예: 감독한 영화, 배우로 출연한 영화 등)을 함께 돌려줌으로써, 이로부터 사용자가 의도하는 탐색 결과를 선택 할 수 있도록 하는 것이다.

IV. 시맨틱 기반 키워드 검색 시스템

1. 시스템 아키텍쳐

시맨틱 키워드 검색 시스템은 세 가지 주요 구조로 이루어진다. : 관계형 데이터베이스를 RDF로 바꾸는 변환기, 질의 키워드를 지식베이스 리소스로 매핑 해 주는 매퍼, 그리고 SPARQL 질의어 구축기로 이루어지며 그림 2.와 같은 구조를 가진다.

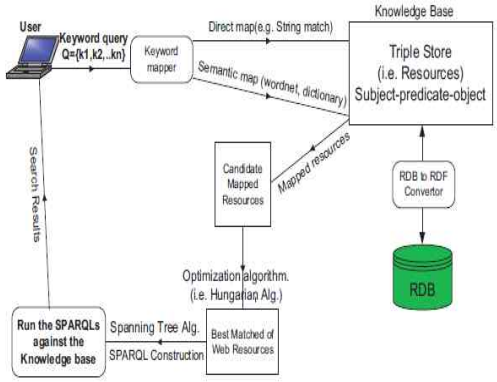

그림2. 시스템 구조
Fig. 2. System Architecture

1.1 RDB -〉 RDF 변환기

이 모듈의 입력은 관계형 데이터베이스 스키마와 인스턴스들이다. 변환기는 관계형 데이터베이스의 스키마와 인스턴스들을 지식베이스로 변환 시키는데, 릴레이션은 클래스로, 릴레이션의 속성들은 클래스의 속성들로 변환한다. 지식베이스는 기본적으로 'subject-predicate-object' 형태를 가진 RDF 이며, 이를 대상으로 SPARQL 질의가 수행된다.

1.2 질의 키워드 -〉 지식베이스 리소스 매퍼

이 모듈의 입력은 사용자의 키워드 질의이다. 매퍼는 각각의 키워드를 구문적으로나 시맨틱이 가장 유사한 지식베이스 리소스와 매핑하기 위하여 문자열 비교 기법을 사용한다.

1.3 SPARQL 질의 구축기

이 모듈의 입력은 RDF-그래프와 같이 트리플로 이루어진 데이터 집합과 매치에서 얻어낸 리소스들이다. SPARQL 질의 구축기는 서로 연결 되어 있는 매치된 리소스-노드들을 포함하고 있는 최소한의 부 그래프를 계산하기 위해서 그래프 이론 알고리즘을 적용한다. 이 부 그래프로부터 SPARQL 질의가 형성되며, 이 질의를 트리플 저장소에서 수행한 다음 사용자에게 질의에 대한 결과로 돌려준다.

2. RDB에서 지식베이스의 생성

2.1 기본 정의와 개념

이 절에서는 새로운 시스템에서 사용하는 기법을 이해하기 쉽도록 시맨틱 웹에서 사용되는 몇 가지 정의와 개념을 정리한다.

- 리소스 정의 프레임워크 (RDF) : RDF는 웹 자원에 대

한 정보를 표현하기 위해서 설계된 표준 프레임워크이다. 이 모델은 〈subject, predicate, object〉 형태의 트리플로 표기되는데, 이것이 한 개의 문장이다. 문장의 주어인 subject는 이 문장이 무엇에 관한 것인지를 나타내는 리소스이며, predicate는 subject의 시맨틱을 묘사하는 속성 혹은 특성을 나타낸다. object는 그 속성의 값을 나타낸다. 다시 말하면, 모든 웹 리소스는 트리플 문장들의 집합으로 시맨틱을 묘사할 수 있게 되는 것이다. 웹 리소스는 URI를 사용하여 고유하게 식별 할 수 있다. 트리플의 subject와 predicate는 항상 URI 웹 리소스인 반면, object는 URI 웹 리소스이거나 리터럴이다. 아래는 영화 'Terminator'를 표현하는 간단한 RDF 정보이다.

```
@prefix rdf: 〈http://www.w3.org/1999/02/22-rdf-syntax-ns#〉
@prefix dir: 〈http://localhost:8080/IMDB/Directors#〉
@prefix mov: 〈http://localhost:8080/IMDB/Movies#〉
@prefix movcat:
〈http://localhost:8080/IMDB/MovieCategory#〉

mov:The_Terminator rdf:type mov:movies.
mov:The_Terminator mov:name "The Terminator".
mov:The_Terminator mov:directedBy dir:James_Cameron.
mov:The_Terminator mov:category movcat:Action.
```

위의 RDF에는 4개의 문장이 있다. 두 번째 문장을 보면, "The_Terminator"를 subject로, "mov:name"을 predicate로 그리고 "The Terminator"를 object로 가지고 있다. 이중에서 subject와 predicate는 "mov" 접두사를 가진 웹 리소스이며, object는 문자열이다. 세 번째 문장을 보면, subject와 predicate가 웹 리소스이고, 두 번째 문장과는 달리 object 역시 웹 리소스로 이루어져 있다.

• 리소스 서술 프레임워크 스키마 (RDFS) : RDF는 단순한 문장을 통하여 리소스를 서술할 수 있는 어휘/용어를 지원하고 있다. RDFS는 이러한 RDF의 시맨틱을 확장하여 더 많은 양의 어휘/용어를 제공함으로써, 지식베이스의 시맨틱을 더 풍부하게 표현할 수 있도록 한다. 즉, RDFS는 리소스와 리소스간의 관계를 시맨틱하게 표현하는 방법을 지원해 준다. 다음은 리소스 클래스와 관계를 서술하기 위해 RDFS 어휘들을 추가한 간단한 RDF의 일부이다.

```
@prefix rdfs: 〈http://www.w3.org/2000/01/rdf-schema#〉
@prefix rdf:
〈http://www.w3.org/1999/02/22-rdf-syntax-ns#〉
@prefix dir: 〈http://localhost:8080/IMDB/Directors#〉
@prefix mov: 〈http://localhost:8080/IMDB/Movies#〉

mov:Movie rdf:type rdfs:class.
mov:The_Terminator rdf:type mov:Movie.
dir:Director rdf:type rdfs:class.
dir:James_Cameron rdf:type dir:Director.
mov:directedBy rdf:type rdf:Property.

mov:directedBy rdfs:domain mov:Movie.
mov:directedBy rdfs:range dir:Director.
```

ontology: 온톨로지는 모든 리소스 클래스, 그들과 연관된 속성들 그리고 그 클래스간의 관계들의 집합으로서, 이들을 총괄하여 개념을 정의하고 상호 관계를 서술한 것이다.

knowledgebase: 지식베이스는 트리플 형태로 표현되는 모든 데이터 인스턴스를 모아 놓은 것을 가리키며 RDF/RDFS 시맨틱 어휘를 사용하여 개념과 관계를 서술한다. 그러므로 지식베이스와 트리플 저장소는 같은 의미로 사용된다.

RDF 그래프: RDF 그래프는 방향 그래프로써 트리플 저장소를 표현하며, 각 점은 리소스 혹은 문자열을 나타내고 선은 두 개의 리소스 혹은 리소스와 문자열 사이의 관계를 나타내는 것이다. 위의 IMDB 예제를 RDF 그래프로 표현하면 그림. 3과 같이 나타난다.

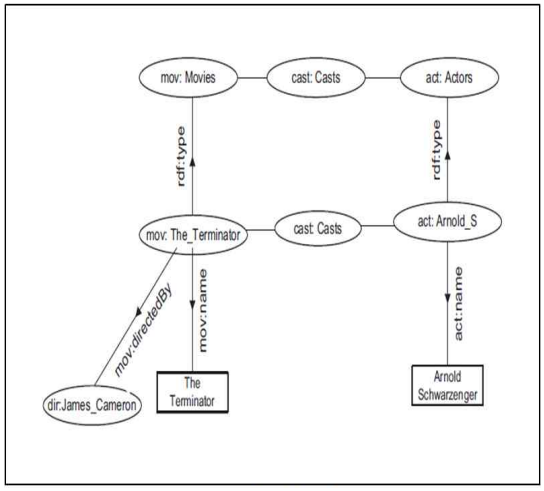

그림3. IMDB의 RDF 그래프
Fig. 3. RDF Graph ofr IMDB

2.2 데이터베이스 스키마에서 온톨로지 추출

주어진 관계형 데이터베이스 스키마에서, 그 스키마에 나타나는 모든 개념들과 그들의 속성, 그리고 관계들까지 모두

표현할 수 있는 온톨로지를 추출 해 낼 수 있다. 다행히도 RDB와 같은 구조적 데이터는 텍스트와 같은 비구조적 데이터나 반구조적 데이터에 비해서 스키마로부터 온톨로지를 끌어내는 작업이 훨씬 수월하다. 여기서 사용되는 기법은 상용화된 RDB2RDF (relational database to RDF)와 다른 연구[10]에서 사용하는 기법과 유사하다.

2.2.1 RDB 스키마를 RDB 모델로 변환

관계형 모델을 사용하여 영화 데이터가 모델링 되는 변환 과정을 보도록 하자. 이 예는 RDB 모델과 RDF 모델이 얼마나 유사한지를 잘 보여준다.

관계형 데이터베이스 전문가가 분산 되어 있는 영화 데이터들을 모아 개체-관계 모델로 통합하는 과정을 고려해 보자. 먼저 데이터에서 개념들을 추출한 다음 각 개념들에 연관된 속성들을 찾아내고 마지막으로 그 개념들 간의 관계를 연결하는 식으로 진행 할 것이다. 예를 들면, 먼저 두 개의 개념에 관한 개체(릴레이션)가 생성 될 것이다. "Director"와 "Movie"가 두 개의 개체이다. "Director" 개체에는 "firstName", "lastName" 그리고 "DOB"의 속성이 연결 되며, "Movie" 개체는 "movieTitle", "year", "genre" 그리고 "director" 속성이 연결 되어질 것이다. "Movie" 개체의 속성 중에서 "director"는 이미 독립적으로 존재하는 개체임에도 불구하고, "Movie"의 속성으로 포함 되어 있음을 알 수 있다. E-R 모델에서는 이러한 현상을 외래 키로 정의하고 "Movie" 개체에서 "Director" 개체로 참조를 함으로써 관계를 표시하고 있다. 이렇게 추출된 E-R 모델에 따라서 데이터베이스의 인스턴스들을 채우는 것이다. 각 릴레이션(테이블)내의 인스턴스(레코드)들은 기본 키에 의해서 고유하게 식별 되어야 함은 물론이다.

영화 데이터를 RDF 모델로 모델링하는 과정은 위와 같은 과정을 반복하는 것으로, RDB를 RDF로 변환하는 과정이 자연스럽게 진행 될 수 있다. RDB에서 개체를 나타내는 각 테이블은 RDF 모델에서는 개념을 표현하는 클래스로 변환되고, 테이블의 속성들은 클래스 속성으로 변환 된다. RDB에 저장된 실제 데이터를 RDF 모델의 트리플(subject-predicate-object) 형태로 변환하기 위해서는, 테이블의 열을 하나씩 스캔하는 것으로 시작한다. 기본 키에 의해서 고유하게 식별되는 테이블내의 각 열은 트리플의 subject로 변환되는데, 이는 URI에 의해 고유하게 식별되는 웹 리소스이다. 그럼으로, subject를 식별하기 위해서는 URI 이름으로 테이블 이름에 그 열의 기본 키를 덧붙여서 사용한다. (즉, http://localhost:8080/IMDB/Directors#1. 마지막 #1은 director의 기본 키이다.) 각 테이블의 행은 모두 predicate가 되는데 이 predicate는 웹 리소스이다. (즉, http://localhost:8080/IMDB/Moviess#title. title은 영화의 제목이다.) 마지막으로 행의 값은 object가 되며 이때 object는 문자열이거나,, 혹은 그 값이 외래 키인 경우에는 또 다른 웹 리소스를 가리키는 웹 리소스가 된다. 다음은 세 개의 트리플을 보여주는 예제이다. 첫 번째 트리플에서 영화 5의 제목은 "The Termiantor"이며 이는 문자열이다. 두 번째 트리플에서 영화 5의 감독은 "dir:4"이며 이는 웹 리소스임을 나타낸다. 세 번째 트리플은 "dir:4" 가 "dir:name"이라는 속성을 가지며 속성의 값은 "James Cameron" 임을 나타내고 있다.

```
@prefix dir: <http://localhost:8080/IMDB/Directors#>
@prefix mov: <http://localhost:8080/IMDB/Movies#>

mov:5 mov:title "The Terminator".
mov:5 mov:director dir:4.
dir:4 dir:name "James Cameron".
```

RDB에서 RDF로 변환하는 알고리즘은 그림 4.와 같다.

Algorithm: RDB to RDF Transformation
Input: RDB schema, set of tables and the data stored in them
Output: an RDF knowledgebase (triple store)

1: S // Relational database Schema
2: $table_i$ //A table in Schema
3: $attr_k$ // A column/attribute in a table
4: $table_{i\_PK}$ // The primary key for a *$table_i$*
5: $table_{i\_URI}$ // *$table_i$* namespace
6: KB // Knowledgebase
7: For each $table_i \in S$
8: For each $attr_j \in table_i$
9: if ($attr_j$.value is Literal)
10: SELECT ("$table_{i\_URI}$"+$table_{i\_PK}$), ("$table_{i\_URI}$"+"$attr_j$"), ($attr_j$)
11: else if ($attr_j$.value is Foreign_Key)
12: SELECT ("$table_{i\_URI}$"+$table_{i\_PK}$), ("$table_{i\_URI}$"+"$attr_j$"), ("$table_{r\_URI}$"+$attr_j$)
13://$table_{r\_URI}$ is the URI for the referenced table.
14: end for
15: end for

그림 4. RDB에서 RDF 변환 알고리즘
Fig. 4. RDB to RDF Algorithm

RDF 그래프가 생성되기만 하면, class, properties, instance등의 웹 리소스에 대해 그들의 문자열과 함께 색인을 만들어 매핑 할 때 빠른 액세스가 가능하게 된다.

3. 질의어를 지식베이스 용어로 변환

3.1 기본 정의와 개념

RDB를 트리플 저장소로 변환한 다음에는 질의 키워드를 지식베이스의 용어 중에서 구문적으로나 의미상 가장 유사한 용어로 매핑 하여야 하는데, 키워드 비교 기법으로는 스테밍, N-grams, 시맨틱 비교 그리고 Levenshtein 거리등의 기법들이 여러 응용분야에서 사용되어져 왔다. 이들 중 우리 시스템에서는 Levenshtein 거리와, SPARK와 KEMANTIC에서의 시맨틱 비교 기법을 사용하였다[7][8].

• Levenshtein 거리: Levenshtein 거리는 문자열 편집 거리(string edit distance)라고도 하는데, 두 문자열의 차이 혹은 거리를 측정하는 문자열 비교 기법이다. 두 개의 문자열이 주어지면, 하나의 문자열을 다른 문자열로 변환하는데 필요한 삽입, 삭제, 치환과 같은 편집 연산의 수를 계산한다. 두 문자열의 구문적 유사성을 계산하기 위해서 다음의 공식을 사용한다.

similarity(str1,str2) =
1-LevenshteinDist(str1,str2)/maxLength(str1,str2)

• 시맨틱 매핑 : 시맨틱 매핑을 할 때는 WordNet과 같은 단어사전을 활용하여 지식베이스 용어들에서 질의 키워드를 찾는다. 질의 문자열이 N개의 용어로 이루어졌다면, 구문적 유사성을 계산하기 위해서 질의 문자열을 파싱할 때 두 가지를 고려한다.

1) 질의 문자열을 N개의 용어로 나누어 토큰화 한 후, 각각의 토큰과 지식베이스 내에 있는 리소스들의 구문적 유사성을 계산하기 위해 위의 공식을 사용한다.
2) 다음과 같은 방식으로 질의 문자열로부터 두 개의 용어로 구성되는 구/명사 세트를 만들어간다. 첫 번째 구는 처음과 두 번째 용어로, 두 번째 구는 두 번째와 세 번째 용어로, 그리고 마지막으로는 N-1번째 용어와 N번째 용어로 쌍을 이루어 모두 N-1개의 세트를 구성한다. 이 각각의 구와 지식베이스 리소스간의 구문적 유사성을 계산한다.

질의 query Q = {scary movie stars}를 고려해 보자. 사용자의 의도는 가) "scary movie"라는 이름의 영화에 출연하는 "stars"를 찾거나 나) 단지 scary movie 즉, 공포영화에 출연하는 "stars"를 찾으려는 것일 것이다. 질의 문자열을 파싱할 때 사용자의 의도를 고려하지 않는다면, 사용자의 질의 의도를 놓칠 수 있다. 이에 두 가지 경우를 다 고려한다면, 가)와 나)로 구분되는 두 가지 질의 의도를 놓치지 않고 커버할 수 있다. 두 개 용어로 구를 구성하는 이유는, 통계적으로 볼 때 사용자의 공식화 된 키워드 질의들은 대부분 단일어이거나, 혹은 각 용어의 혼합 그리고 2개의 구 혹은 명사로 이루어지는 문자열이므로 3-용어, 4-용어로 세트를 구성할 필요가 없기 때문이다.

유사성 점수가 0.4 이상인 리소스는 대응되는 키워드의 매칭 세트 Q{}k에 유사성 점수와 함께 추가된다. 예를 들면, Q{}played={(played in, 0.9), (acted, 0.5), (performed, 0.5)}는 키워드 "Played"와 매치된 리소스 쌍들과 유사성 점수의 집합을 나타낸다. 구문적 유사성이 0.4 이하인 리소스는 질의 키워드에서 의도 하는 바가 아님을, 약 80%의 테스트 결과들에서 보여주었으므로 임계값은 0.4로 정하였다. 그 다음에 각각의 질의 키워드와 시맨틱이 유사한 키워드를 사전에서 찾아 성공하면, Q{}k에 유사성 점수 0.5와 함께 추가한다. 이는 질의 용어에 대응되는, 시맨틱이 유사한 리소스에 주어지는 유사성 점수이다.

3.2 키워드 매핑 기법

질의 키워드 kj의 매칭 세트 Q{}ki는 유사한 리소스 Rj와 유사성 점수 si,j를 포함한다. 키워드와 리소스의 매핑은 K와 R 집합 두개로 분리된 노들들을 연결하는 이분된 그래프로 쉽게 나타낼 수 있다

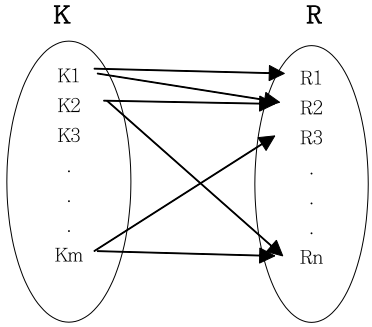

그림 5. 키워드-리소스 매핑을 모델링하는 이분 그래프.
Fig. 5. Bipartite graph modeling Keywords to Resources

여기에서 목표는 키워드-리소스 쌍 전체에 대한 유사성 점수를 최대화하기 위해 키워드와 리소스(즉, (Ki,Rj)쌍)의 최상의 매치를 찾는 것이다. 자세한 방법은 헝가리언 알고리즘을 참조한다[11]. 열은 리소스를 행은 키워드를 나타내는 유사성 행렬을 생성해서 각 셀은 해당하는 키워드와 리소스의

유사성 점수를 나타낸다. 이 유사성 행렬에 최적화 알고리즘을 실행하여 총합 유사성 점수를 최대화 하는 매치를 찾아내면 된다.

다음은 할당 문제를 설명해주는 예이다.

IMDB 데이터베이스에서 사용자가 "The Terminator" 영화의 주연 배우를 알고 싶어 한다고 가정하자. 사용자는 Q={terminator stars} 형식의 키워드 질의를 통해 검색을 하게 되고, 키워드와 그 키워드에 대응되는 지식베이스 용어(즉, 구문적, 시맨틱상의 유사한 용어)로 만들어지는 유사성 행렬은 다음과 같다.

	K_1 =Terminator	K_2 =Stars
R_1 =The Terminator	0.71	X
R_2 =Star Track	X	0.44
R_3 =Actors	X	0.5

여기에서의 최적화된 매치는 {(Terminator, The Terminator), (Stars, Actors)} 이며 이중 첫 번째 쌍은 구문적으로 가장 유사한 쌍이고 두 번째 쌍은 시맨틱이 가장 유사한 쌍이다.

4. SPARQL 질의의 생성

4.1 SPARQL 질의

SPARQL은 RDF 데이터베이스를 위한 그래프-매칭 질의어인데, 변수와 트리플 패턴으로 구성되는 질의를 구축 가능하게 해준다. 구문적 측면에서 SPARQL은 SELECT와 WHERE 절로 이루어지는 SQL 질의어와 비슷하다. 앞의 예제에서 본 키워드 질의를 고려해 보자. "The Terminator" 영화의 주인공을 찾는 SPARQL 질의는 다음과 같다.

```
SELECT ?actor
WHERE {
    ?movie rdf:type mov:Movies.
    ?movie rdf:name "The Terminator".
    ?movie cast:Casts ?actor.
    ?actor rdf:type act:Actors.
    }
```

SELECT문은 어떤 변수와 그의 값을 돌려받아야 하는지를 나타내는데, 여기서는 그 변수가 "?actor"이다. WHERE 절은 RDF 그래프와 매치 되어야 하는 트리플 패턴(부그래프)을 나타낸다. 이 질의에서는 두 개의 변수가 사용되는데, "?actor"와 "?movie" 변수 두 개는 "?actor"와 같이 돌려받아야 하는 리소스를 나타내든지, 아니면 WHERE 절 내의 RDF 그래프와 "?actor"와 "?movie" 둘 다 매치되는 웹 리소스를 나타낸다. 이 예제에서 WHERE 절의 트리플 패턴은 그림 6.에 나타난다.

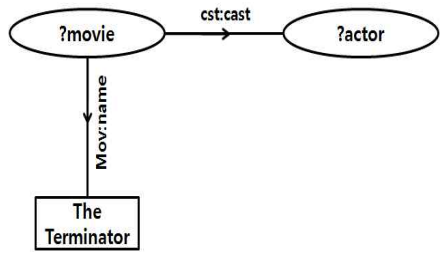

그림 6. 트리플 패턴
Fig. 6. The Triple Pattern

아래 표처럼 SPARQL 질의는 Actor 리소스 "atc:Arnold_S", "act:Linda_H", "act:Michael_B" 그리고 "act:Paul_W"의 결과를 돌려준다. 공간이 부족하여 모든 Actor를 나타내지는 못하였다.

Actors
act:Arnold_S
act:Linda_H
act:Michael_B
act:Paul_W
…

4.2 매핑 결과에 의한 SPARQL 질의의 생성

가장 잘 매치되는 리소스를 찾은 다음에는, 사용자의 의도에 맞는 키워드 탐색을 수행하기 위해 동일한 SPARQL 질의를 생성해야 한다. 질의 생성은 다음의 두 단계를 통해서 이루어진다.

1) 매치가 이루어진 모든 리소스를 한번 씩은 방문하는 부그래프를 만들어야한다. 이는 최소 Steiner 스패닝 트리 등으로 해결할 수 있는데, RDF상의 정점의 집합이 V이고 이와 매치된 리소스 정점의 집합을 R이라 할 때, R의 정점들을 가장 짧은 거리로 연결한 부그래프를,, 생성하는 것이 목표이다.

예제에서 최적화된 매치 리소스는 {The Terminator, Actors} 이며 "The Terminator"는 문자열이고 "Actors"는 웹 리소스 클래스이다. 최소 Steiner 스패닝 트리는 리소스 정점들이 제공되는 RDF에 알고리즘을 실행하여 찾는다.

2) 스패닝 트리를 찾은 다음에는 다음의 순서로 SPARQL 질의를 생성한다. 가) 예제처럼 매치한 리소스가 "Actors"와 같이 클래스이면, 변수 (?actors)를 할당하여 WHERE 절의 조건을 만족하는 클래스 타입 리소스 (Actors)를 갖게 한다. 나) Steiner 트리 경로에는 있으나 매칭과정에서 매핑 되지 않는 리소스 클래스 역시 (?movies)와 같이 변수로 처리 된다. 다) "The Terminator" 와 같이 매치된 리소스가 문자열이면, 대응되는 트리플 패턴의 WHERE 절 내 조건으로 사용한다.

4.3 랭킹 스킴

질의 용어를 스키마 용어로 매핑 할 때, 질의 용어 세트에 매치되는 리소스 세트가 하나 이상일 수도 있으므로 SPARQL 질의도 하나 이상일 수 있다. 그러므로 구축된 SPARQL을 실행할 때 나오는 결과의 수에 따라 각기 랭크 되어야 한다. 단, SPARQL의 결과에 직접적으로 랭크를 하지 않고 매치되어, 결과로 나온 리소스와 상관된 요소들에 따라 다음과 같이 SPARQL을 랭크한다.

1) 질의 용어에 좀 더 유사한 리소스 세트로 만들어진 SPARQL은 상위에 랭크 되며, 유사성 행렬에 최적화 알고리즘을 실행할 때 함께 수행한다.

2) 실험 결과에 의하면, 매치된 리소스 세트가 문자열을 담고 있는 터미널 노드들일 때, 이로부터 구축되는 SPARQL은 변수 노드들로 구축되는 SPARQL에 비해 높이 랭크 되어야 한다. Q = {scrary movie stars} 라는 질의를 고려해 보면, 웹 리소스 "scary movie"를 문자열로, "stars"를 변수로 처리하여 구축한 SPARQL은 "scary", "movie" 그리고 "actors"를 변수로 처리한 SPARQL 질의 보다 상위에 랭크 되어야 한다. 왜냐하면, 사용자가 {Movies Arnold Schwarzenegger}, {Action Movies Travolta}와 같이 하나 이상의 상수 개체 값을 가진 질의를 검색 엔진에 제출하면, SPARQL 구축 과정에서는 문자열이 매치되기 때문이다.

V. 실험 결과

일반적으로 키워드 검색 시스템을 평가하는 것은 힘든 과제이다. 관계형 데이터베이스의 범주에서 봤을 때, 테이블, 테이블의 속성 그리고 조건들이 명확하게 명시 되는 SQL 질의와는 달리, 키워드 질의는 불명확한 용어들을 통하여 사용자의 의도를 표현한다. 키워드 질의 결과의 정밀성과 정확성은 사용자 주관에 따라 달라지므로, 키워드 검색을 평가하는 기준을 정하는 것도 결과의 정확성을 공식적으로 증명하는 것도 불가능한 일이다. 우리 시스템의 공정한 평가를 위해서 여러 사용자들로 하여금 원하는 정보를 얻기 위한 키워드 질의를 만들도록 하였다. 이 사용자들은 우리 시스템에서 사용하는 기법에 대해 전혀 모르는 사용자들이다. 영화 데이터베이스 IMDB로부터 다양한 정보를 얻기 위해 각기 다양한 키워드 질의 20개를 만들도록 요청 하였다.

표 2.는 키워드 질의와 사용자가 의도하는 검색결과가 함께 표현된 리스트이다. 질의에는 두 가지 유형의 키워드가 포함 될 수 있다.

키워드 유형1 : namei로 표시된 키워드는 actor, actresses, studios등의 이름을 나타낸다. 키워드 유형2: actor, director, starred in과 같이 실제 개념이나 관계를 나타내는 키워드이다. 키워드 유형1 만 포함하는 질의는 테이

표 2. 샘플 키워드 질의
Table 2. Sample Keyword Queries

Keyword Query	Intended Search Results
moviei actors	list of the actors in moviei
moviei director	director of moviei
moviei director actors	list of the director and the actors in moviei
* moviei	list of the info about moviei (actors,director,genre,etc)
* actori genrei	list of the genrei movies where actori starred in
moviei stars	list of actors in moviei
moviei year	the year in which moviei was made
genrei movies directori	genrei movies directed by directori
movies directed starred namei	movies directed and starred in by namei
* actori yeari	movies made in yeari featuring actori
*actori directori	movies directed by directori featuring actori
genrei movies directori actori yeari	genrei movies made in yeari directed by directori featured actori

블에서 *와 함께 표기되고 질의 유형1로 간주되며, 키워드유형1과 키워드 유형2를 함께 포함하는 질의는 질의 유형2 로 간주된다.

질의 결과에 대한 사용자의 만족도를 비교 평가하기 위해 다음과 같이 실험을 하였다. 우리 시스템과 기존의 시스템(KSRDB [7])에 각각의 질의를 실행 한 다음, 상위 10개의 결과를 사용자에게 알려주고 그들 중 사용자의 의도에 부합하는 결과를 직접 평가하여 선택 하도록 하였다. 사용자가 각각의 시스템에서 선택한 결과를 통합하여 각 질의에 대한 총괄 검색 결과로 채택하였다.

시스템의 평가와 비교를 위해 다음 두 가지를 측정하였다.

1) 평균 정밀도/소환 (예: 각 소환 레벨 0.1에서의 평균 정밀도) : 각 시스템마다 교대로 유형1, 유형2 질의를 소환 레벨 0.1로 실행한 결과에 대해 평균 정밀도를 계산 하였다. 그림 8.은 두 시스템의 정밀도/소환 그래프이다.

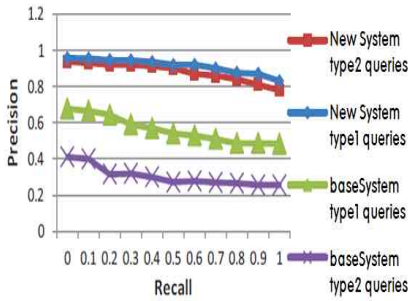

그림 8. 정밀도 / 소환 커브
Fig. 8. Precision / Recall Curve

그림에서처럼, 유형1, 유형2 질의가 우리 시스템에서 기존 시스템보다 정밀한 결과를 보였다. 질의 유형1이 질의 유형2보다 두 시스템 모두에서 더 정밀한 결과를 보였는데, 이는 관계, 개념 등을 키워드로 사용하는 질의 유형2보다 상수 값을 가진 질의 유형1을 사용할 때, 사용자의 의도에 근접한 결과를 찾을 수 있음을 나타낸다. 우리의 시스템에서 질의 유형1이 더 나은 결과를 보인 것은, 사용자들이 본래의 질의가 키워드 유형1로 시작했음에도 불구하고, 더 의미 있는 결과를 얻기 위해서는, 결국 개념과 그들 간의 관계를 이용하려는 의도가 있음을 나타낸다.

2) 역 랭크 중위(Mean Reciprocal Rank (MRR)):

질의의 역 랭크는 질의에 대한 첫 번째 정답 결과의 랭크를 거꾸로 한 것이다. 각각의 시스템에서 질의 유형1, 질의 유형2에 대한 평균 역 랭크를 계산하였다. MRR은 검색 결과 세트에서 얼마나 빨리 유효한 결과가 찾아지는지를 나타낸다. 그림 9.는 두 시스템에서 질의의 길이(즉, 사용된 키워드의 수)에 따른 MRR의 변화를 나타내고 있다. 우리 시스템이 기존의 시스템보다 나은 수행 결과인 높은 MRR을 보이며 실험 결과, 일반적으로 질의 유형2 보다 질의 유형1의 MRR이 높게 나왔다. 또한 평균적으로 키워드의 수가 2개 이상 4개 이하일 때 MRR이 높게 나오는데, 이는 이 범위내의 키워드를 사용할 때 원하는 결과를 빠르게 얻을 수 있음을 나타낸다.

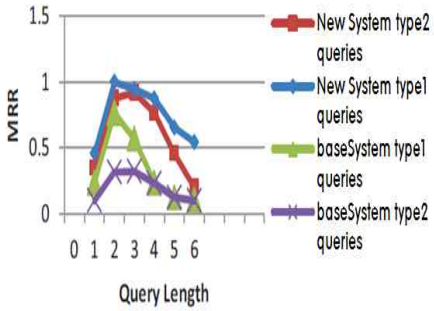

그림 9. MRR-질의 길이 플롯
Fig. 9. MRR-Query Length Plot

VI. 결론 및 향후 연구 방향

이 연구는 관계형 데이터베이스에서의 키워드 탐색에 대해 개략적으로 살펴보았으며, 사용자의 키워드 질의와 연관있는 결과를 얻는 기법과 랭킹 기법에 대한 것이다. 또한 RDF, RDFS 그리고 SPARQL과 같은 시맨틱 웹 기술에 대해 살펴보았다. 이 결과로 새로운 시스템을 제안하였는데, 이 시스템은 지식베이스내의 리소스 중에서 주어진 질의 용어와 구문적으로나 시맨틱이 가장 유사한 리소스와 매핑하고, 매핑 된 용어들로 이루어진 SPARQL을 구축하여 키워드 질의와 동일한 질의를 사용, RDB를 RDF 지식베이스로 변환한 상태에서 키워드 탐색의 결과를 돌려준다. 앞장의 실험 결과에서 나타나듯이, 새로운 시스템은 기존의 시스템에 비해 키워드 질의에 있어서 사용자의 의도에 좀 더 부합하는 결과를 돌려주는 것으로 나타났다. 특히 질의 유형에 상관없이 사용자가 원하는 결과를 얻기 위해서 개념과 그 개념간의 관계 레버리지를 사용할 때 성능 효과가 50% 이상 상승함을 보였다.

현재까지는 actor's name, genre, studio's name 등과 같은 짧은 문구로 이루어지는 속성만 고려했지만, 향후에는 영화 설명, 영화 요약 그리고 영화 리뷰등과 같은 긴 텍스트로 이루어지는 속성을 가진 테이블에 대한 추가 연구도 이루어져야한다. 이를 위해서는 이런 속성 값을 RDF로 바꾼 다

음 RDB에서 변환 된 나머지 RDF와 융합시켜야 한다. 용어 매핑과 SPARQL 질의의 구축 기술 또한 긴 문자 텍스트 속성을 새로운 리소스 클래스와 관계들로서 포함하여, 새롭게 생성되는 지식베이스를 수용할 수 있도록 확장 되어야 할 것이다.

참고문헌

〔1〕 Hak Soo Kim, Gun-Woo Kim, Hyemyung Seo, Jin Hyun Son, " An Efficient Semantic Query Processing Method based on SQL", Database Researcch, Vol. 24, No. 3, pp. 14-29, Dec. 2008.

〔2〕 Kim Youn Hee, Shin Hye Yeon, Lim Haechull, Chong Kyun Park, "Indexing and Storage Schemes for Keyword-based Query Processing over Semantic Web Data", Journal of The Korea Society of Computer and Information, Vol. 12, No. 5, pp. 93-102, Nov. 2007.

〔3〕 Youn-Hee Kim, Jee-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.

〔4〕 Sung Wan Kim, "Suffix Array Based Path Query Processing Scheme for Semantic Web Data", Journal of The Korea Society of Computer and Information, Vol. 17, No. 10, pp. 107-116, October 2012.

〔5〕 V. Hristidis and Y. Papakonstantinou, "Discover: Keyword search in relational databases", In VLDB, pages 670.681, 2002.

〔6〕 Sanjay Agrawal, Surajit Chaudhuri, Gautam Das, "DBXplorer: A System for Keyword-Based Search over Relational Databases," icde, pp.0005, 18th International
Conference on Data Engineering (ICDE'02), 2002.

〔7〕 Zhou, Q., Wang, C., Xiong, M., Wang, H., Yu, Y., "Spark: Adapting keyword query to semantic search. In: A berer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudre-Mauroux, P. (eds.) ISWC 2007. LNCS, vol. 4825, pp. 694-707. Springer, Heidelberg (2007)

〔8〕 Bergamaschi S, Domnori E, Guerra F, OrsiniM, Lado RT, Velegrakis Y., "Keymantic: Semantic keyword based searching in data integration systems", Proceedings of VLDB, vol 3(2), pp. 1637-1640, 2010.

〔9〕 Ziyang Liu , Jeffrey Walker , Yi Chen, "XSeek: a semantic XML search engine using keywords", Proceedings of the 33rd international conference on Very large data bases, September 23-27, Vienna, Austria, 2007.

〔10〕 Shufeng Zhou, "Exposing Relational Database as RDF", 2nd International Conference on Industrial and Information Systems, 2010.

〔11〕 http://en.wikipedia.org/wiki/Hungarian_algorithm

저 자 소 개

양 영 휴

1981: 이화여자대학교 문리대학 영어영문학과 문학사.

1985: Univ. of Pittsburgh Dept. of Computer Science 학사수료.

1987: Northeastern University School of Computer Science MS.

1992: 서강대학교 공과대학 컴퓨터공학과 박사수료.

현 재: 한양여자대학교 정보경영과 교수

관심분야: 데이터베이스

Email : yhueyang@gmail.com