관심 커뮤니티 추천을 위한 시스템 질의를 지원하는 온톨로지 기반 시맨틱 블로그 모델

(An Ontology-based Semantic Blog Model for Supporting System Queries to Recommend Interest Community)

양 경 아 † 양 재 동 †† 최 완 †††
(Kyung-Ah Yang) (Jae-Dong Yang) (Wan Choi)

요 약 본 논문에서는 계층화된 개념 지식인 온톨로지를 블로그 환경에 적용함으로써 시스템 관리자가 블로그 공간(Blogosphere)을 체계적으로 분석, 관리할 수 있는 온톨로지 기반 시맨틱 블로그 모델을 제안한다. 제안 모델에서 시스템 관리자는 온톨로지 용어 사이에 설정된 다양한 관계성 링크를 추적하고 분석함으로써, 블로그 사용자들이 다양한 관점에 따라 유기적으로 원하는 블로그 자원에 접근할 수 있도록 지원한다. 즉, 블로그들 간에 이루어지는 상호작용 활동을 파악하고 상호작용을 기반으로 동적으로 관심 커뮤니티를 형성해주며, 시스템은 이를 적절한 사용자들에게 추천 방식으로 제공해 주는 기능을 지원한다. 제안 모델의 지원 기능을 체계화하기 위해 본 논문에서는 1) 블로그 환경에 사용되는 자원과 관련된 정보를 객체와 관계성으로 표현한 뒤 2) 이들을 대상으로 구조적 연산을 지원하는 일련의 연산자 집합 및 이들간의 조합으로 구현되는 시스템 질의 처리 과정을 정형화한다.

키워드 : 블로그, 온톨로지, 시맨틱 블로그, 시스템 질의, 시맨틱 웹

Abstract This paper suggests an intelligent semantic blog model to systematically analyze and manage blogosphere with ontology as its conceptual knowledge base. In the model, the system managers may support users to easily find appropriate blog resources by tracking and analyzing various relationships between ontology – they may intelligently recommend interest blog communities to relevant users by monitoring interaction activities in blogoshpere, dynamically grouping the communities with the ontology. To systematically specify the functionality of our model, 1) we first express the structure of blog resources in terms of objects and relationships between them and then 2) we formalize a set of operators designed to be applied to the resources. System queries are implemented by the combination of the operators.

Key words : Ontology, Semantic Blog, System Query, Semantic Web

· 본 연구는 정보통신부 및 정보통신연구진흥원의 IT신성장동력핵심기술 개발사업의 일환으로 수행하였음. [2007-S-015-01, SaaS기반 이동형 개인맞춤 사무환경 구축 기술 개발]
· 이 논문은 2007 한국컴퓨터종합학술대회에서 '관심 커뮤니티 추천을 위한 시스템 질의를 지원하는 온톨로지 기반 시맨틱 블로그 모델'의 제목으로 발표된 논문을 확장한 것임

† 정 회 원 : 케이태크 멀티미디어 DB 연구원
escavan@gmail.com
†† 정 회 원 : 전북대학교 전자정보공학부 교수
jdyang@chonbuk.ac.kr
††† 종신회원 : ETRI 온디맨드서비스연구팀 팀장
wchoi@etri.re.kr

논문접수 : 2007년 10월 2일
심사완료 : 2008년 2월 26일

정보과학회논문지: 소프트웨어 및 응용 제35권 제4호(2008.4)

1. 서 론

최근 괄목할만한 성장을 이루고 있는 블로그는 온라인 상에서 활발한 정보 교류를 가능하게 하는 새로운 커뮤니티 방식으로 각광받고 있다. 블로그 사용자는 다른 사용자가 블로그에 작성한 글, 즉 포스트를 RSS[1]로 구독하고 이들을 브라우징 하는 방식으로 관심 커뮤니티를 확장하며 비슷한 성향의 사용자 네트워크를 형성한다. 현재 블로그 사용자는 여러 독립적인 블로그 지원 도구를 사용하여 웹 상에서 손쉽게 블로그 자원을 생산하고 유포하여 그 양은 갈수록 방대해지고 있는 추세이다.

다양한 형식으로 다양한 장소에 분산되어 있는 블로그 자원은 자체적으로 독립적인 속성을 지니며 여러 관점을 내포하고 있는 것이 특징이다[2]. 따라서, 기존의

디렉토리와 같은 고정적이고 제한적인 정보 분류 스키마 구조만으로는 특정 관점과 연관된 적절한 집합체를 형성하거나 사용자가 원하는 정확한 정보를 추출하는데 한계가 있다. 이 문제를 해결하기 위해 태그와 시맨틱 웹 기법을 사용하여 블로그 공간(Blogosphere)을 카테고리화하고 조직화하는 연구들이 수행되었다[3-8].

태그 접근 방식은 블로그 사용자가 직접 블로그 자원에 키워드인 태그를 여러 개 입력하여 다양한 관점을 반영할 수 있도록 한 블로그 분류방식으로 사용자가 별도의 훈련 과정 없이 태그를 손쉽게 삽입할 수 있다는 장점 때문에 널리 사용되고 있다. 그러나, 같은 의미를 뜻하면서 서로 다른 이름들을 가지고 있는 태그의 경우 이를 처리하는 메커니즘을 제시하지 못하며 태그 간 관계가 의미적으로 구조화 되어 있지 못하다는 단점을 지닌다. 시맨틱 웹 접근 방식은 RDF/OWL과 같은 시맨틱 웹 언어를 사용하여 블로그 자원에 대한 개념적 의미를 표현할 수 있도록 하였으나 전문가가 아닌 일반 사용자들이 사용하였을 때 이 언어를 이해하고 처리할 수 있어야 한다는 제약이 있다. 기존의 두 연구 방식의 또 다른 문제점은 사용자들이 동적으로 상호작용하는 블로그 공간 내부에서 발생하는 커뮤니티 활동을 모니터링하여 유동적으로 변화하는 상황을 블로그 사용자 서비스에 반영하는 체계적인 방법론은 지원하지 못하고 있다는 점이다.

블로그 사용자가 복잡한 전문지식 없이도 블로그 자원을 직관적으로 탐색하고 사용자의 다양한 관점을 반영하여 원하는 자원에 접근하기 위해서는, 시스템 관리자가 블로그 자원을 효율적이고 조직적으로 관리할 수 있는 환경이 지원되어야 한다. 더하여 보다 지능적인 블로그 서비스를 위해서는 외부에서 블로그 공간에 대한 상황을 동적으로 모니터링함으로써 사용자 상황을 정확히 인식한 뒤 사용자의 성향 및 관심사항을 고려하여 관련 블로그 커뮤니티들을 적합한 사용자에게 적절히 추천할 수 있는 기능이 반드시 지원되어야 한다.

본 논문에서는 계층화된 개념 정보인 온톨로지를 블로그 환경 내에 지식정보로 활용함으로써 블로그 공간을 체계적으로 분석, 관리하고, 블로그 사용자들이 관심 커뮤니티들을 효율적으로 접근할 수 있도록 지원하는 온톨로지 기반 시맨틱 블로그 모델을 제안한다. 제안 모델에서 시스템 관리자는 온톨로지 용어 사이에 설정된 다양한 관계성 링크를 추적하고 분석함으로써, 블로그 사용자들이 다양한 관점에 따라 유기적으로 원하는 블로그 자원에 접근할 수 있도록 지원한다. 즉, 블로그들 간에 이루어지는 상호작용 활동을 파악하고 상호작용을 기반으로 동적으로 관심 커뮤니티를 형성해주며, 시스템은 이를 적절한 사용자들에게 추천 방식으로 제공해 주는 기능을 지원한다. 제안 모델의 지원 기능을 체계화하기 위해 본 논문에서는 1) 블로그 환경에 사용되는 자원과 관련된 정보를 객체와 관계성으로 표현한 뒤 2) 이들을 대상으로 구조적 연산을 지원하는 일련의 연산자 집합 및 이들간의 조합으로 구현되는 시스템 질의 처리 과정을 정형화한다.

본 논문의 구성은 다음과 같다. 먼저 2장에서는 온톨로지 기반 시맨틱 블로그와 연관된 기존 연구에 대해 살펴본다. 3장에서는 온톨로지 기반 시맨틱 블로그 모델에서 온톨로지 개념을 나타내는 개념 객체와 포스트 객체, 블로그 객체를 정의하고 객체 간 설정 가능한 관계성을 정의한다. 또한 다양한 관점에 따라 객체들을 효과적으로 브라우징 할 수 있는 일련의 연산자 집합을 정의하고 이들의 조합으로 시스템 관리자가 의도하는 객체를 추출할 수 있는 시스템 질의 기법을 기술한다. 4장에서는 시스템 질의와 클러스터링 메소드를 사용하여 사용자 관심 커뮤니티를 추천하는 방식을 살펴보고, 마지막으로 5장에서는 결론 및 향후 연구 과제를 제시한다.

2. 관련연구

블로그는 지역적으로 분산된 장소에 설치되며 블로그 공간 내에서 그 자체로 독립적인 속성을 지닌다[9]. 블로그 공간을 조직화하고 카테고리화하기 위한 연구는 크게 태그 접근 방식(Tag Approach)과 시맨틱 웹 접근 방식(Semantic Web Approach)으로 나뉠 수 있다.

태그 접근방식에서 태그는 블로그 포스트의 특성을 나타내는 짧은 키워드로 일종의 메타 데이타 역할을 수행한다. [3-5]에서는 방대한 양의 정보를 생산해내는 분산된 블로그를 효율적으로 분류하는 방법으로 기존의 고정적 분류체계가 아닌 사용자가 직접 키워드를 연상하여 블로그 정보에 태그를 덧붙이는 방식의 폭소노미(Folksonomy)를 제안하고 있다. 폭소노미는 사용자가 생성하는 상향식(bottom-up) 분류방식[10]으로 다양한 관점을 반영할 뿐만 아니라 사이트 방문자가 제공하는 정보의 집합체 역할도 수행한다. [11]에서는 사용자 위주의 지역적이고 개별적인 태그 기반의 블로그 분류 방식에서 벗어나 글로벌한 토픽 위주의 블로그 분류 환경을 제공하고자 하였다. 이 연구에서는 태그를 이용한 내용기반 클러스터링을 수행하여 블로그를 조직화하고 블로그 정보에 메타 데이타를 삽입하여 사용자들이 정보를 카테고리화 하였다. 그러나, 태그 접근 방식에 기반을 둔 이들 연구에서 태그는 단순한 키워드들의 모음으로 태그 간의 의미적, 구조적인 관계성들을 관리하는 기능은 지원되지 않는다는 단점을 가진다.

시맨틱 웹 접근방식으로 블로그 공간 및 커뮤니티를 구조화하려는 연구들도 이루어졌는데, 대표적인 것들로

는 [6-8] 등을 들 수 있다. 특히 HP의 시맨틱 블로그[6]는 SWAD(Semantic Web Advanced Development) 프로젝트[12]의 일환으로 지능적인 블로그 서비스를 위해 온톨로지 기반의 메타데이타를 블로그에 적극 활용하는 연구를 수행하고 있다. 이를 위해 웹 사이트 간의 정보를 공유할 수 있는 표준인 RDF를 포스트를 기술하는 메타데이타로 사용하고 있다. 시맨틱 블로그의 자세한 기능은 별도로 운영중인 데모사이트[13]를 참조하기 바란다. SIOC프로젝트[7]는 시맨틱 웹 기술을 이용해 온라인 커뮤니티를 의미적으로 연결하기 위한 시도이다. SIOC는 RDF 기반의 단순한 온톨로지를 이용해 데이타의 의미를 표현하고 관련된 데이타를 연결해 준다. 기존 폐쇄된 커뮤니티 사이트에서 블로그 자원들은SIOC 인터페이스를 통해 RDF로 저장되고 SPARQL[14]를 통해 질의를 하게 된다. [8]에서는 RDF를 포스트를 기술하는 포맷으로 사용하여 블로그를 시맨틱 웹의 근간 플랫폼으로 확대하여 블로그 공간을 구조화하기 위한 연구를 수행하였다.

그러나, 이같은 시맨틱 웹 접근 방식은 전문가가 아닌 일반 사용자들이 사용하였을 때 시맨틱 웹 언어를 학습하여 이를 이해하고 처리할 수 있어야 한다는 한계점이 있다. 또한 아직RDF 기반의 단순한 온톨로지를 이용하여 간단한 트리플 구조의 블로그 자원을 검색하는 수준에 그치고 있어 세부적이고 정교한 블로그 검색 질의를 수행하기 어렵다는 단점을 가지고 있다.

한편, 태그 접근 방식과 시맨틱 웹 접근 방식 모두의 공통적인 또 다른 문제점으로는 블로그 사용자들이 동적으로 상호 작용하는 인적 구성 활동을 고려하고 있지 못하다는 점인데, 이를 보완하기 위한 연구들로 [15,16]에서는 웹 사이트에 존재하는 개인의 프로파일 정보를 추출하고 분석하여 지연이나 인맥 혹은 관심사 등을 기반으로 네트워크를 연결하는 인적 구성망 서비스(Social Network Service)를 지원하고자 하였다. 특히 [16]에서는 블로그 상에 구축된 FOAF[17]정보들을 이용하여 블로그 구성원들이 이루는 인적 구성망의 그래픽 시각화를 지원하는 시스템도 제시하였다. 그러나, 이 방식들 또한 온톨로지를 이용해 블로그 공간을 구조화시키고 블로그 사용자들의 인적 활동을 모니터링하여 유동적으로 변하는 관심 커뮤니티 정보를 사용자에게 추천해 주는 기능은 지원하지 못하고 있다.

3. 온톨로지 기반 시맨틱 블로그 모델(OSEM)

온톨로지 기반 시맨틱 블로그 모델(OSEM: **O**ntology-based **Sem**antic Blog Model)은 기존의 블로그 시스템에 온톨로지의 강력한 의미적 표현력을 적용함으로써 지금까지 언급한 제반 문제점들을 해결하고자 하는 새로운 시도로 볼 수 있다. OSEM에서 온톨로지는 블로그 공간 내 개념적인 정보 및 정보들 간의 세부적인 관계성들을 구조적으로 파악하기 위한 필수 요소로 사용된다.

OSEM에서 객체는 개념 객체, 포스트 객체, 블로그 객체 등이 있으며 객체는 수직/수평적인 연관 관계성을 가질 수 있는 계층화된 형태로 표현된다. 본 논문에서 카테고리, 키워드 그리고 온톨로지 용어 등은 모두 개념 객체와 동일한 의미를 가진다.

그림 1은 객체와 객체 간 설정된 관계성을 UML 표기 방법을 써서 표현한 시맨틱 블로그 모델의 한 예를 보이고 있다. 여기서, "computer"와 "business"는 각각 두 개념 객체 계층들의 최상위 객체 명들이며, 각 개념 객체 계층 내 한 개념 객체는 세부적인 의미를 지닌 하위 개념 객체들을 가질 수 있다. 개념 객체들은 수평적으로 다른 개념 객체들과 연관 관계성(associationOf)을 비롯한 다양한 관계성들을 맺을 수 있는데, 예를 들어, "business"와 "computer"는 수평적으로 associationOf 관계를 가지며, "business"계층의 "strategic planning"은 "computer" 계층의 "knowledge management"와 수평적으로 hasPlanFor 관계를 가지고 있다. 또한 "competition strategy"는 "semantic web"과 hasSpecificPlanFor 관계에 있고, "web design"은 "web plan"과 improvements 관계성을 가진다.

계층 내의 각 개념 객체는 관련된 내용을 갖는 포스트 객체들을 포함할 수 있으며 이는 contains 관계성으로 표현한다. 예를 들어, 포스트 객체 p03이 "시맨틱 웹 환경에서 블로그와 위키의 비교 분석"이라는 내용을 담고 있다고 가정했을 때, c08, c09 그리고 c010 개념 객체들은 p03을 contains 하는 관계에 있게 된다. 이 때, p03를 할당한 "wiki", "blog"와 "semantic web"은 각각 개념 객체의 이름들이며, 동시에 그 포스트 객체를 대표하는 키워드들로 볼 수 있다.

실제 블로그 환경에서 하나의 블로그는 여러 개의 포스트들을 가질 수 있는데 isPostOf 관계성은 이러한 의미를 명세한다. 예를 들어, 다섯 개의 포스트 객체 p01, p02, p03, p04, p05는 모두 URI가 "paul.net"인 블로그 객체 b01의 포스트들이며 isPostOf 관계성을 이용하여 블로그와 포스트들간의 관계를 표현한다. 블로그 내에서 포스트는 다른 포스트에 트랙백을 할 수 있으며 이는 trackback 관계성을 이용하여 나타낼 수 있다. 예를 들면, p010는 p03과 trackback 관계에 있다. 특정 블로그는 다른 포스트를 스크랩할 수 있으며 이러한 관계는 scraps를 통하여 표현된다. 예를 들어, URI= "ted.net"인 블로그 객체 b03는 p05, p06 그리고 p07과 scraps한 관계에 있다.

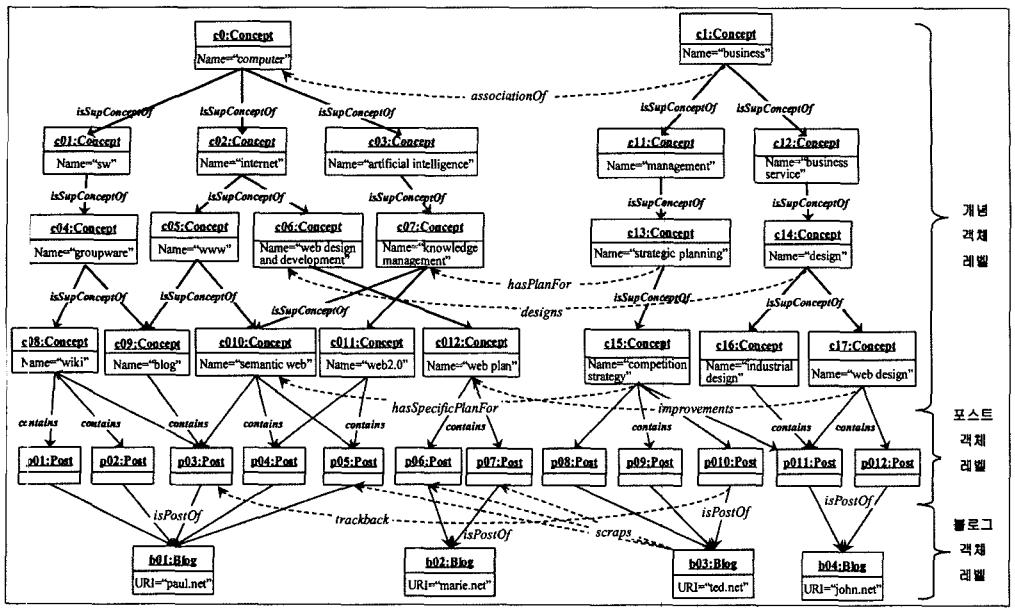

그림 1 온톨로지 기반 시맨틱 블로그 모델의 예

정의 1은 OSEM에서 블로그 환경을 조직적으로 구조화하기 위한 자료형 및 이들을 처리하기 위한 연산자 집합, 그리고 시스템 질의를 정의하고 있다.

정의 1. OSEM은 다음과 같이 3-투플로 정의된다.

$$\text{OSEM} = < S, OP, SQ > .$$

여기서 S는 OSEM에서 사용되는 모든 객체와 관계성으로 이루어진 구조체(Structure)이며, OP는 S에 적용되는 연산자들의 집합, SQ은 S에 적용되는 시스템 질의들의 집합이다. OSEM을 구체화하기 위해 다음에는 OSEM을 이루는 각 구성 요소들에 대해 보다 자세히 기술하기로 한다.

3.1 객체-관계성 구조체(S)

객체-관계성 구조체 S는 레이블화된 계층형 그래프 모양으로 표현되는 자료 구조로, 객체들의 집합 O와 관계성 집합 R로 구성된다. S를 정의하기 위하여, S를 구성하는 O와 R이 먼저 정의될 필요가 있다.

• 객체 집합

객체 집합 O는 개념 객체 집합 C, 포스트 객체 집합 P, 블로그 객체 집합 B 등 세 종류로 구성된다. 즉,

$$O = C \cup P \cup B.$$

포스트의 성격을 지정하는 하나의 개념 객체 $c \in C$는 구체적인 의미의 하위 개념 객체들을 가질 수 있다. 하위 개념 객체는 상위 개념 객체보다 세부적인 의미를 지니며 isSupConceptOf / isSubConceptOf를 통해 계층 구조를 이루게 된다. 개념 객체 집합 C에 부여되는 속성들은 다음과 같이 Property(C)={ID, Name}로 표현한다.

포스트 객체 $p \in P$는 개별적인 포스트를 나타내는 실제 예의 의미를 지니며 개념 객체와 달리 자신들의 하위 개념을 갖지 못한다. 하나의 포스트 객체는 포스트의 성격에 따라 하나 이상의 개념 객체에 할당될 수 있으며 개념 객체와 contains / isContained 관계를 가진다. Property(P)는 포스트 객체 집합 P에 부여되는 속성들로 Property(P)={ID, URI, Title, Body, Date, Tags}이다.

블로그 객체 $b \in B$는 개별적인 블로그를 나타내는 실제 예로서 블로그 객체 역시 자신들의 하위 개념을 갖지 못하며 포스트 객체와 isBlogOf / isPostOf 관계를 가질 수 있다. 블로그 객체는 개념 객체와 직접적인 관계성은 가지지 않는다. 블로그 객체 집합 B에 부여되는 속성 집합은 Property(B)={ID, URI, Title, Date, User}이다.

각 객체 $o \in O$에 대한 속성값을 참조하고자 할 때는 $o.A$, $A \in$Property(O)로 표기한다. 이 때, 변수 o의 타입이 결정되는 시점은 o가 특정 인스턴스로 바인딩되는 시점이다. 예를 들어, o.ID의 경우 변수 o에 어떠한 객체가 바인딩되느냐에 따라 속성 ID의 성격이 결정된다.

객체를 참조할 때 특정 객체를 지칭하는 객체 상수와 상황에 따라 바뀔 수 있는 객체를 나타내는 객체 변수를 사용할 수 있다. 객체 상수는 객체 각각을 구별할 수 있도록 부여되는 유일한 값이다. 예를 들어, 개념 객체의 경우 c01, 포스트 객체의 경우 p01, 블로그 객체의 경우 b01 등으로 각 객체는 고유한 식별자를 지닌다.

객체변수는 객체 상수를 값으로 가질 수 있는 변수로, 객체 상수와 구분하기 위해 이탤릭체 소문자로 표기한다. 객체변수 역시 객체 상수와 마찬가지로 객체 타입에 따라 각각 개념 객체 변수인 경우 c, 포스트 객체 변수인 경우 p, 블로그 객체 변수 b 등으로 표기할 수 있다.

• 관계성 집합

관계성은 두 개의 객체 사이의 연관된 의미를 표현하는 것으로 개념 객체, 블로그 객체, 포스트 객체 간의 관계를 설정할 수 있다. 개념 객체는 자신보다 일반적인 혹은 구체적인 의미를 가진 개념 객체와 일반화/구체화 관계를 가짐으로써 계층구조를 형성하며 또한 수평적으로 다른 계층의 객체들과 의미적 관계성을 맺는다.

정의 2. 관계성 $r \in R$의 표기법은 다음과 같이 정의한다.

$$r: O_i \rightarrow O_j.$$

관계성 r에 대한 domain과 range는 각각 domain(r), range(r)로 나타낼 수 있기 때문에, domain(r)= O_i, range(r)= O_j이다. r은 O_i와 O_j에 따라 방향성을 가지며 역방향에 따른 역관계성도 지닌다. 관계성 및 역관계성을 나타내는 관계성 집합 R과 R^{-1}은 다음과 같다: R = {isSupConceptOf, contains, associationOf, rssfeed, isPostOf, scraps, trackback, neighbor, userDefined}, R^{-1}={isSubConceptOf, isContained, inveseAssociationOf, inverseRssfeed, isBlogOf, isScrapped, inverseTrackback, inverseNeighbor, inverseUserDefined}.

이제, 객체 집합과 객체들 간의 관계성 집합을 이용하여 객체-관계성 구조체 S를 정의한다.

정의 3. 객체-관계성 구조체 S는 다음과 같이 투플들의 집합으로 정의할 수 있다.

$$S(O,R) = \{s \mid s = < r, o_1, o_2 > \in R \times O \times O\}.$$

여기서 O와 R은 각각 $S(O, R)$ 내 객체들의 집합, 관계성들의 집합이다.

예 1. 그림 1에서 $S(O, R)$는 정의3에서 정의된 투플들의 집합으로 표현이 가능하다. 예를 들어, <isSupConceptOf, c04, c08>, <contains, c08, p01>, <isPostOf, p01, b01>$\in S(O, R)$.

OSEM에서의 관계성은 수직적인 관계성 집합 R_1과 R_1^{-1}, 수평적인 관계성 집합 R_2 그리고 R_2^{-1}으로 구분할 수 있으며 이에 따라 $S(O, R)$도 정의 4와 같이 두 집합으로 분할될 수 있다.

정의 4. $R = R_1 \cup R_1^{-1} \cup R_2 \cup R_2^{-1}$ 이고 R_1 = {isSupConceptOf, contains}, $R^{-1}{}_1$ = {isSubConceptOf, isContained}, R_2 = {associationOf, rssfeed, isPostOf, scraps, trackback, neighbor, userDefined}, $R^{-1}{}_2$ = {inveseAssociationOf, inverseRssfeed, isBlogOf, isScrapped, inverseTrackback, inverseNeighbor, inverseUserDefined}라고 하자. 그러면 $S(O, R) = S_1(O, R \cup R_1^{-1}) \cup S_2(O, R_2 \cup R_2^{-1})$일 때, 모든 $s = < r, o_1, o_2 > \in S(O, R)$에 대해서 $r \in R_1$ 또는 $r \in R_1^{-1}$이면 $s \in S_1(O, R_1 \cup R_1^{-1})$이고, $r \in R_2$ 또는 $r \in R_2^{-1}$ 이면 $s \in S_2(O, R_2 \cup R_2^{-1})$로 정의한다.

이후 S_1과 S_2는 각각 객체-관계성 수직 구조체, 객체-관계성 수평 구조체로 명명하기로 한다. 또한, $S(O, R)$에서 R을 명세할 필요가 없을 때는 $S(O)$, O와 R 모두의 명세가 불필요할 경우에는 간단히 S로 표기하기로 한다.

다음 표 1은 정의 2에 따른 표기법에 의해 관계성 집합 R_1과 R_2에 대한 각각의 역관계성을 나타낸 것이다.

본 논문에서 클래스의 계층을 설정하는 isSupConceptOf 관계성의 경우(Parent Concept) isSupConceptOf(Children Concept)이므로 화살표 방향이 상위개념 객체에서 하위개념 객체로 향하게 된다.

상위 레벨의 개념 객체 간 관계성은 하위 레벨의 개념 객체의 관계성으로 상속이 된다. R_2에서 associaotionOf 관계성은 다른 모든 수평적인 관계성들을 일반화하는 최상위 관계성으로 정의되기 때문에, 다른 관계성들은 모두 associationOf 의 하위 관계성들이다. 그림 2는 관계성들을 구조화한 계층도이다.

그림 2에서 associaotionOf 관계성을 비롯하여 이것의 하위 관계성인 trackback, scraps, neighbor, rssfeed,

표 1 관계성과 역관계성

	Relationships(R)	Inverse Relationships(R^{-1})
수직적 관계집합 (R_1)	isSupConceptOf : $C_i \rightarrow C_j$ contains: $C \rightarrow P$	isSubConceptOf : $C_i \rightarrow C_j$ isContained : $P \rightarrow C$
수평적 관계집합 (R_2)	associationOf : $O_i \rightarrow C_j$ isPostOf : $P \rightarrow B$ trackback : $P_i \rightarrow P_j$ scraps : $B \rightarrow P$ neighbor : $B_i \rightarrow B_j$ rssfeed : $B_i \rightarrow B_j$ userDefined : $O_i \rightarrow C_j$	inverseAssociationOf: $O_i \rightarrow C_j$ isBlogOf : $B \rightarrow P$ inverseTrackback : $P_i \rightarrow P_j$ isScrapped : $P \rightarrow B$ inverseNeighbor : $B_i \rightarrow B_i$ inverseRssfeed : $B_i \rightarrow B_j$ inverseUserDefined : $O_i \rightarrow C_j$

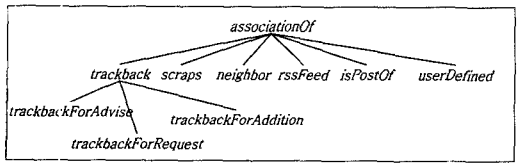

그림 2 관계성 R_2의 계층도

isPostOf는 시스템이 기본적으로 제공하는 시스템 정의 관계성이다. 시스템 관리자는 객체 간 다양하고 적절한 관계성을 부여하기 위해 시스템 정의 관계성을 확장하거나 새로운 관계성인 userDefined 관계성을 만들 수 있다. 예를 들어, 시스템 관리자는 trackback과 같은 시스템 정의 관계성을 확장하여 trackbackForAdvise, trackbackForAddition, 그리고 trackbackForRequest 등으로 세분화된 관계성을 작성할 수 있으며 그림 1에서 보인 hasPlanFor, hasSpecificPlanFor, designs, improvements 등과 같은 userDefined 관계성을 생성할 수 있다.

계층화된 관계성은 시맨틱 블로그 모델에서 확장 검색에 사용될 수 있다. 예를 들어, 그림 1에서 p010과 trackbackForRequest 관계에 있는 포스트들을 검색하였을 때 이에 대한 해당 결과가 없을 경우 상위 관계성인 trackback으로 관계성을 확장하여 p03 포스트를 결과로 반환할 수 있다. 본 논문에서는 온톨로지를 기반으로한 시맨틱 블로그 모델을 설계하고 질의어를 작성하는 것을 주안점으로 두었으므로 관계성의 상속을 이용한 세부적인 논리 추론방식은 본 논문에서는 언급하지 않기로 한다. 관계성의 상속성을 이용한 자세한 논리 추론 방식의 관심 사항은 [18]를 참고하기 바란다.

다음은 관계성들을 이용해, 객체들간의 상관 관계를 유기적으로 추적하기 위해 필요한 정의이다.

정의 5. 다음은 S_1에서 R_1에 대한 전이 폐쇄 집합 S_1^+에 대한 정의이다.

$S_1 \subseteq S$ 이고 $r_1 \in R_1$에 대해 s_1 = <isSupConceptOf, o_i, o_j>, s_2 = <r_1, o_j, o_k>, s_3 = <r_1, o_i, o_k>라고 하자. 그러면 S_1^+는 다음과 같이 정의 된다.

1) 만약 $s \in S_1$이면 $s \in S_1^+$이다.

2) 만약 $s_1 \in S_1^+$이고 $s_2 \in S_1^+$ 이면 $s_3 \in S_1^+$ 이다.

3) S_1^+ 의 원소는 1)과 2)로만 얻어진다.

예 2. 그림 1에서 <isSupConceptOf, c03, c07>이고 <isSupConceptOf, c07, c010>이므로 <isSupConceptOf, c03, c010>$\in S_1^+$이다. 또한 <isSupConceptOf, c07, c010>이고 <contains, c010, p05>이므로 <contains, c07, p05>$\in S_1^+$이다.

정의 6. 개념 객체 $c \in O$를 최상위로 하는 객체-관계성 수직 구조체 $S_1(O_c) \subseteq S_1(O)$에 대한 정의는 다음과 같다.

$$S_1(O_c) = \{s \mid s = < r, o_1, o_2 > \in R_1 \times O_c \times O_c\}.$$

이 때, O_c = {o' | o'=c ∨ <r, c, o'>$\in S_1^+(O)$}이다.

편의상, c.Name = 객체명일 때, O_C는 $O_{\text{객체명}}$과 병행해서 표기하기로 한다. 예를 들어, c0.Name = "computer" 이므로, O_{c0}는 O_{computer}로 표기할 수 있다.

3.2 연산자

OSEM은 객체들을 대상으로 구조적인 연산을 가능하게 하는 다음의 연산자 집합 OP를 제공한다 :

OP = {"[A=상수]", "*r", ∩, ∪, GLBS, LUBS}.

먼저, 각 $op \in OP$-{∩, ∪} 연산자는 하나의 객체 집합에 적용되어 그 결과값으로 객체 집합을 되돌려 주는 연산식을 구성하게 되는데, 다음은 이들 각 연산자를 한 번씩 적용하여 표현된 단위 연산식 E_{op}를 정의한 것이다.

정의 7. 단위 연산식 E_{op}: $2^{O_i} \rightarrow 2^{O_j}$, $op \in OP$-{∩, ∪}는 다음과 같이 정의된다.

1) $E_{[A=\text{상수}]}(O') = O'$ [A = 상수] = {o' | $o'.A$ = 상수, $o' \in O'$}.

2) $E_{*r}(O') = O'$ *r = {o| <r, o', o >$\in S$, $o' \in O'$}.

3) $E_{GLBS}(O')$ = GLBS(O'), $E_{LUBS}(O')$ = LUBS (O').

여기서 2^{O_i}, 2^{O_j}는 O_i, $O_j \subseteq O$의 멱집합, $O' \subseteq O_i$, $A \in$Property(O'), $r \in R$이다.

다음은 각 연산자와 해당 단위 연산식을 예를 통해 자세히 설명하도록 한다.

• "[A = 상수]" 연산자

객체 집합에 적용되어 특정 속성값을 가지는 객체들만을 추출할 경우 사용한다. 그림 3에서 포스트 객체 집합 P 중 Title 속성값으로 "wiki tutorial"를 가지고 있는 포스트들을 추출하기 위한 단위 연산식은 $E_{[\text{Title="wiki tutorial"}]}(P) = P[p.\text{Title="wiki tutorial"}]$ = {p01}이다.

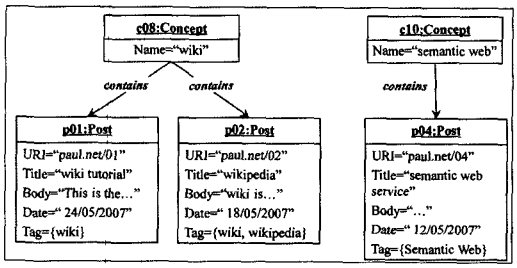

그림 3 개념 객체에 할당된 포스트 객체 집합의 예

• "*r" 연산자

$s \in S$의 관계성들에 따라 객체 간을 항해 할 수 있는 기능을 수행하며, 관계성의 종류를 표기하여 특정 관계성을 추적할 수 있다.

예 3. 그림 1에서 c07이 포함하는 포스트 객체들은

정의 5에서 정의한 S_1^+의 전이적 성질에 의해 <contains, c07, p03>, <contains, c07, p04>, <contains, c07, p05>$\in S_1^+$이므로 $E_{*contains}(\{c07\})=\{c07\}*contains=\{p03, p04, p05\}$이다.

• LUBS / GLBS연산자

연산자 의미를 설명하기 위해 먼저 S를 구성하는 객체 계층을 정의하고 객체들 사이의 상계 집합과 하계 집합을 정의한다. 임의의 객체 집합 $O' \subseteq O$의 UBS (Upper Bound Set)와 LBS(Least Bound Set)의 정의는 다음과 같다.

정의 8. $r \in R_1$과 $o' \in O' \subseteq O$에 대해, $sup(o')=\{o|<r, o, o'>\in S_1\}$이고, $sub(o')=\{o|<r, o', o>\in S_1\}$일 때, UBS($o'$)과 LBS($o'$)는 다음과 같이 정의된다.

$$UBS(O') = \bigcap_{\forall o' \in O} sup(o'), \quad LBS(O') = \bigcap_{\forall o' \in O} sub(o').$$

하나의 최상위 객체로부터 파생되는 객체 계층은 유일한 최소 상계와 최대 하계를 항상 가질 수는 없기 때문에 격자 구조로는 표현될 수 없다. 따라서 다음과 같이 최소 상계 집합(LUBS: Least Upper Bound Set)과 최대 하계 집합(GLBS: Greatest Lower Bound Set)을 정의할 필요가 있다. LUBS(O')는 객체 $o' \in O'$를 모두 하위 객체로 가지는 개념들 중에서 가장 구체적인 개념을 추출하며 GLBS(O')는 객체 $o' \in O'$가 공통으로 가지는 하위 객체들 중에서 가장 일반적인 객체들을 추출한다[19].

정의 9. S와 $O' \subseteq O$의 모든 객체 $o'_i \subseteq O'$, $i=1,2,\cdots,n$에 대한 LUBS(O')와 GLBS(O')는 $r \in R$일 때 다음과 같이 정의된다.

1) $LUBS(O') = UBS(O') - \{o \mid o, o' \in UBS(O'), < r, o, o' > \in S_1\}$.

2) $GLBS(O') = LBS(O') - \{o \mid o, o' \in LBS(O'), < r, o', o > \in S_1\}$.

예 4. 그림 4에서 $O'=\{c02, c03\}$에 대해 sup(c02)={c0, c02}이고 sup(c03)={c0, c03}이다. 따라서 UBS(O') = sup(c02) ∩ sup(c03)로부터 LUBS(O')={c0}를 구할 수 있다. 또한 sub(c02)={c02, c05, c06, c09, c010, c012, p03, p04, p05, p06, p07}이고 sub(c03)={c03, c07, c010, c011, p03, p04, p05}이다. 따라서 LBS(O')= sub(c02) ∩ sub(c03) = {c010, p03, p04, p05}로부터 GLBS(O')={c010}.

LUBS연산자는 이미 알고 있는 구체적인 의미의 객체들을 모두 포괄할 수 있는 일반적인 의미의 상위 객체들을 참조하기 위해 사용된다. 즉, 원래의 질의 영역의 조건을 완화하여 보다 일반적인 영역을 지정할 수 있도록 한다. 반면에, GLBS 연산자는 객체 사이의 상속 성질을 이용하여 상위 객체들이 공통으로 포함하는 구체적인 의미의 하위 객체들을 참조하기 위해 사용된다. GLBS 연산자는 질의 영역을 세부적으로 제약하여 보다 구체적인 질의 영역을 설정할 때 이용될 수 있다.

마지막으로 ∩와 ∪는 지금까지의 연산자들과는 달리 두 개의 서로 다른 객체 집합에 적용되는 연산자들이다.

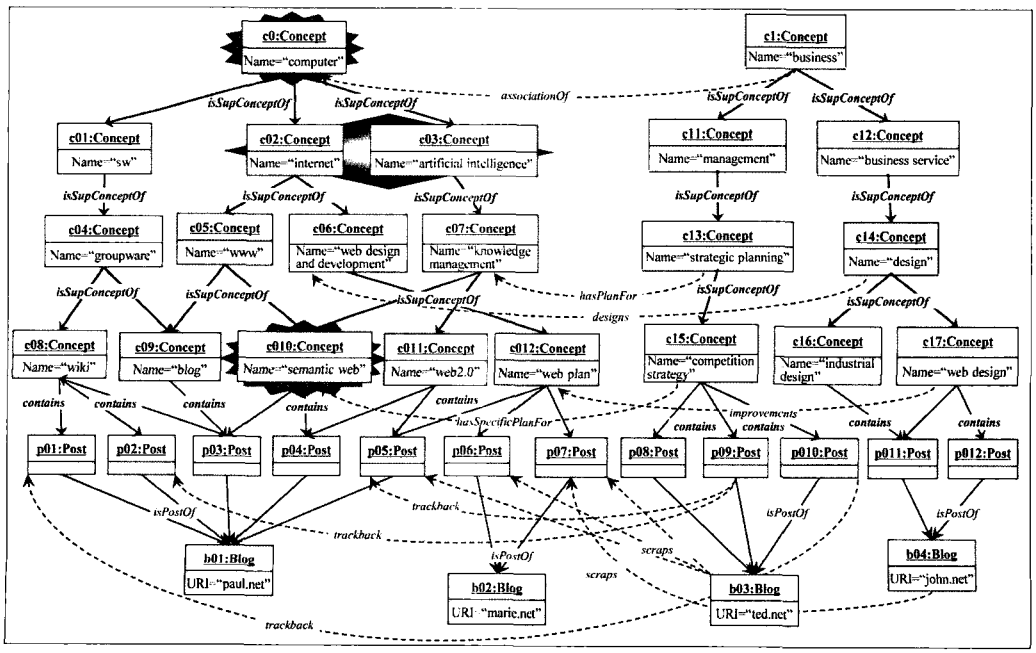

그림 4 LUBS({c02,c03})와 GLBS({c02, c03})

• Intersection(∩) / Union(∪) 연산자

Intersection(∩) 연산자는 두 개 이상의 객체 집합에 대한 교집합을 구하고자 할 때 사용하며, 두 개 이상의 객체 집합에 대한 합집합을 구하고자 할 때에는 Union (∪) 연산자를 사용한다.

OP 내의 각 연산자들은 서로 조합되어 복합 연산자 표현식을 구성할 수 있는데, 다음 정의는 이를 정형화한 것이다.

정의 10. $op \in$ {"[*A*=상수]", "**r*", GLBS, LUBS}, $op' \in \{\cap, \cup\}$, 그리고 $r \in R$일 때, 복합 연산자 표현식, 또는 간단히 복합 연산식 E_{op*}는 다음과 같이 정의된다.

1) 단위 연산식 E_{op}는 복합 연산식 E_{op*}이다.
2) E_{op1*}가 복합 연산식일 때, E_{op1*} [*A* = 상수]는 복합 연산식이다. 여기서, $A \in$ Property(E_{op1*}).
3) E_{op1*}가 복합 연산식일 때, $E_{op1*} * r$는 복합 연산식이다.
4) E_{op1*}가 복합 연산식일 때, GLBS(E_{op1*}), LUBS (E_{op1*})는 복합 연산식이다.
5) E_{op1*}, E_{op2*}가 복합 연산식일 때, E_{op1*} op' E_{op2*}는 복합 연산식이다.
6) 복합 연산식은 오직 1), 2), 3), 4) 그리고 5)로만 표현된다.

예 5. 가) 그림 4에서 E_{op1*} = B[*b*.URI = "ted.net"] * isBlogOf 일 때, E_{op1*} = {p08, p09, p010}이고, $E_{op*} = E_{op1*}$ *trackback는 {p08, p09, p010}*trackback = {p01, p02, p05}.

나) URI가 "ted.net"인 블로그가 스크랩한 포스트 집합 중 카테고리가 "web plan"에 포함되는 포스트는 다음과 같이 복합 연산식 E_{op3*}으로 표현될 수 있다. : E_{op1*} = B[*b*.URI = "ted.net"]*scraps, $E_{op2*} = C_{computer}$ [*c*.Name = "web plan"]*contains 일 때, $E_{op3*} = E_{op1*} \cap E_{op2*}$ = {p05, p06, p07}.

복합 연산식은 상당한 의미적 표현력을 지니고 있지만, 복합 연산식들이 추출해 주는 객체 집합의 객체 형을 고려하지 않고, 제약 없이 사용할 경우 질의 처리 시 심각한 문제를 야기할 수도 있다. 예를 들어, {b1, b2} $\subset B$일 때, E_{op1*} = LUBS({b1,b2}) = ø가 되고, 이는 $E_{op2*} = E_{op1*}$ [*b*.URI="ted.net"]을 ø로 만들어 버린다. 복합 연산식의 객체 형을 질의 처리 시 검사하는 문제는 질의의 정확성(soundness)을 시스템이 제약을 두어 보장하는 것과 관련이 있는데, 질의의 완벽성(completeness) 문제와 함께 자세한 고려 사항은 논문이 불필요하게 복잡해지는 것을 피하기 위해 다음 연구로 미루기로 한다.

OSEM에서는 앞서 정의한 연산자와 연산식 이외에 추가로 시스템 정의 함수(System-defined function)를 제공하여 객체들에 대한 연산을 지원한다. OSEM에서 제공하는 대표적인 시스템 정의 함수로 객체 집합에 적용되어 출현 빈도가 가장 높은 객체를 반환하는 함수인 maxFreqObject 함수가 있다. maxFreqObject 함수는 입력으로 받은 모든 객체들 중에 최대 출현빈도를 보인 객체를 결과로 추출하는 역할을 수행한다. 예를 들어, maxFreqObject(B*scraps)는 블로그에 스크랩된 포스트 중 가장 스크랩이 많이 된 포스트들을 결과로 반환한다.

3.3 시스템 질의 집합(*SQ*)

시스템 질의 또는 간단히 질의 *SQ*는 시스템 관리자가 다양하고 세부적인 표현을 통해 원하는 객체들을 내부적으로 참조하기 위해 사용한다. *SQ*는 [19]에서 제안한 객체 기반 시소러스 참조 질의를 시맨틱 블로그 모델에 적합한 방식으로 확장한 것이다. *SQ*는 다음과 같이 정의된다.

정의 11. 시스템 질의 *SQ*는 *SQ* = (영역질의(Scope_Query), 제약질의(Qualification_Query))와 같이 두 부분으로 표현되며 영역질의와 제약질의는 각각 다음과 같이 정의된다.

1) $\text{Scope\_Query} = \vee_{i=1}^{n} E_{op_i*}$, $\text{Scope\_Query} = \wedge_{i=1}^{n} E_{op_i*}$,
또는 *, n>1일 때
Scope_Query = E_{op*}, n=1일 때.

2) $\text{Qualification\_Query} = \text{OR}_{i=1}^{m}(\text{AND}_{j=1}^{n_i}\text{QUAL}_{i,j})$.

여기서 QUAL은 E_{op*} 또는 시스템 정의 함수.

영역질의는 GLBS, LUBS 연산자를 이용하여 개념 계층의 참조 범위를 특정 객체 계층의 일부분으로 한정하며, 제약질의는 영역질의의 참조 범위를 제한하기 위해 사용된다.

정의 12. 영역질의 Scope_Query의 평가 결과인 *O*(Scope_Query)는 다음과 같이 정의된다.

n >1 일 때,

Scope_Query = $\vee_{i=1}^{n} E_{op_i*}$ 이면,
$O(\text{Scope\_Query}) = \text{LUBS}(E_{op1*} \cup E_{op2*} \cup \cdots \cup E_{opn*})$,

Scope_Query = $\wedge_{i=1}^{n} E_{op_i*}$ 이면,
$O(\text{Scope\_Query}) = \text{GLBS}(E_{op1*} \cup E_{op2*} \cup \cdots \cup E_{opn*})$,

n = 1일 때,
Scope_Query = E_{op*}이고, O(Scope_Query) = E_{op*}.

정의 13. *SQ* = (Scope_Query, *)가 시스템 질의일 때, *SQ*의 평가 결과인 *O*(*SQ*)는 다음과 같이 정의된다.

$$O(SQ) = O(\text{Scope\_Query}).$$

예 6. 그림 5에서 SQ_1 = ($C_{computer}$[*c*.Name="wiki"] ∧ $C_{computer}$[*c*.Name="semantic web"], *)일 때, *O*(Scope_Query) = GLBS($C_{computer}$[*c*.Name="wiki"] ∪ $C_{computer}$

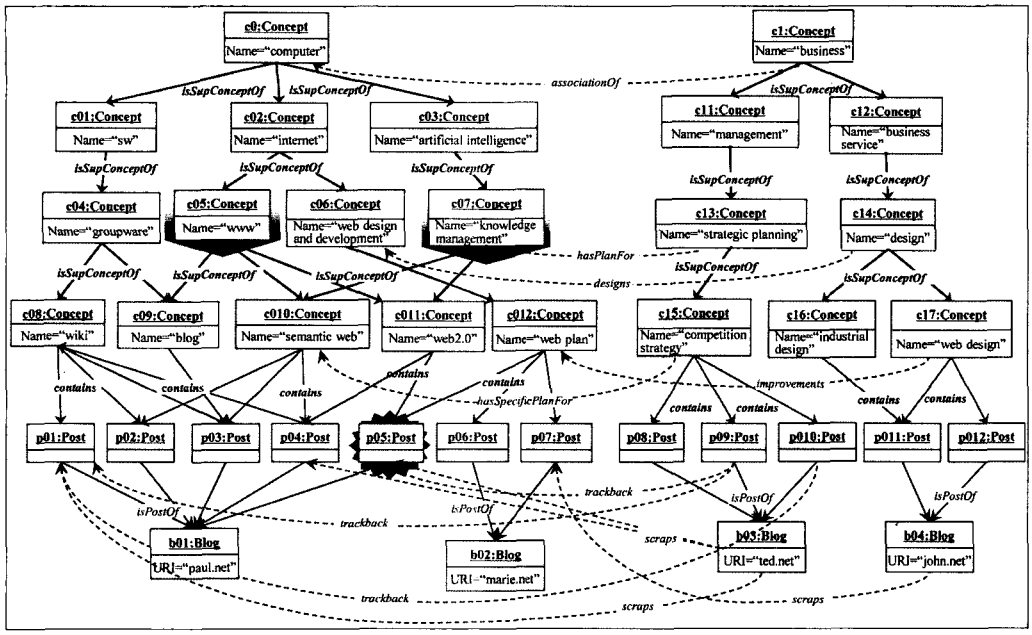

그림 5 SQ_1과 SQ_3의 질의 처리 과정

[c.Name="semantic web"]) = {p02, p03, p04}이며 $O(SQ_1)$ = O(Scope_Query)이다.

정의 14. $o_{i=1\cdots m} \in O$(Scope_Query)를 최상위 객체들로 하는 객체-관계성 수직 구조체에 포함되는 객체 집합 $O_{\text{Scope_Query}}$는 다음과 같이 정의된다.

$$O_{\text{Scope\_Query}} = \bigcup_{i=1}^{m} O_{o_i}.$$

예 7. SQ_2 = (C_{computer}[c.Name="www"] ∧ C_{computer}[c.Name="knowledge management"], *)일 때, O(Scope_Query) = GLBS(C_{computer}[c.Name="www"] ∪ C_{computer}[c.Name = "knowledge management"]) = {c010, c011}이다. 정의 14에 의해 $O_{\text{Scope\_Query}} = O_{c010} \cup O_{c011}$ ={c010, p02, p03, p04} ∪ {c011, p04, p05} = {c010, c011, p02, p03, p04, p05}이다.

정의 15. SQ = (Scope_Query, Qualification_Query)에서 Qualification_Query = $\text{OR}_{i=1}^{m}(\text{AND}_{j=1}^{n_i}\text{QUAL}_{i,j})$일 때, $O(SQ)$는 다음과 같이 정의된다.

$$O(SQ) = \bigcup_{i=1}^{m}\left(\bigcap_{j=1}^{n_i}\text{QUAL}_{i,j}\right), \quad O(SQ) \subseteq O_{\text{Scope\_Query}}.$$

정의 15에서 SQ = (*, Qualification_ Query)로 주어졌을 경우, $O(SQ) = \bigcup_{i=1}^{m}\left(\bigcap_{j=1}^{n_i}\text{QUAL}_{i,j}\right)$이다.

예 8. 그림 5에서 SQ_3 = (C_{computer}[c.Name="www"] ∧ C_{computer}[c.Name = "knowledge management"], B[b.URI="ted.net"]*scrap AND {p09}*trackback)를 고려해보자. 예 7에서 $O_{\text{Scope\_Query}}$ = {c010, c011, p02, p03, p04, p05}이고, 정의 15에 따라 $\text{QUAL}_1 = E_{op1*}$ = B[b.URI = "ted.net"]*scrap = {p01, p04, p05}, $\text{QUAL}_2 = E_{op2*}$ = {p09}*trackback = {p01, p05}, 그리고 $\text{QUAL}_1 \cap \text{QUAL}_2$={p01, p05}이고 p01∉$O_{\text{Scope\_Query}}$ 이므로 $O(SQ)$ = {p05}.

예 9. "전략적 계획이론과 관련된 컴퓨터 분야의 포스트를 찾아라"는 질의를 고려해보자. 이 질의는 다음과 같이 변환이 가능하다.

SQ_4 = (C_{computer}[c.Name="computer"], C[c.Name = "strategic planning"] *hasPlanFor *contains)

그림 6에서 $O_{\text{Scope\_Query}}$ = {c0, c01, c02, c03, c04, c05, c06, c07, c08, c09, c010, c011, c012, p01, p02, p03, p04, p05, p06, p07}이고, 정의 15에 따라 $\text{QUAL}_1 = E_{op1*}$ = C[c.Name="strategic planning"] * hasPlanFor * contains = {p03, p04, p05, p013, p014, p15}이다. {p03, p04, p05} ⊆ $O_{\text{Scope_Query}}$이므로 $O(SQ)$ = {p03, p04, p05}.

본 논문에서 시스템 질의는 편의상 사용자가 직접 자연어로 작성한다고 가정한다. 그러나, 보다 현실적인 질의처리를 위하여 사용자가 견고한 질의를 구성할 수 있도록 GUI 기반의 다이얼로그 메뉴의 도움을 받을 필요가 있다. 이를 위해, 차후 시맨틱 블로그 모델의 전체적인 구조 파악이 가능한 관리 시스템(가칭 시맨틱 블로

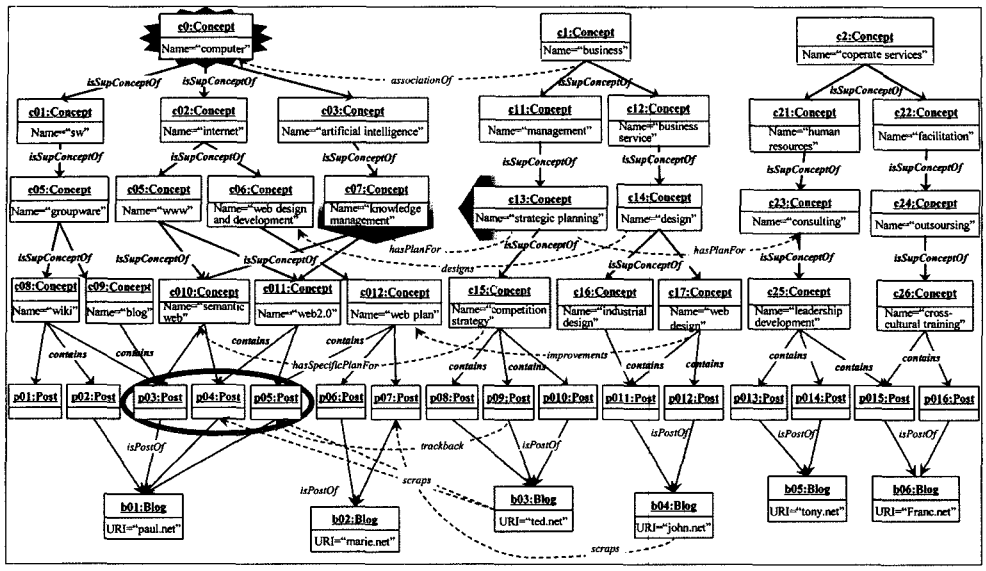

그림 6 SQ_4의 질의 처리 과정

그 관리 시스템)을 구현하고 시스템 질의 작성 메뉴를 이용하여 사용자에게 입력 가능한 연산자와 객체명 등을 자동으로 추천, 제시할 예정이다. 시스템 질의는 현재 사용되고 있는 온톨로지 질의어인 SPARQL만으로 그 의미를 충분히 표현하기에 어려움이 있으므로 시스템 질의를 처리하기 위한 별도의 질의 처리기가 필요하다. 이 모듈은 시맨틱 블로그 관리 시스템 내에 필수 모듈로 포함될 예정이다.

4. 질의와 클러스터링을 이용한 관심 커뮤니티 추출 방식

이 절에서는 구조화된 시스템 질의와 클러스터링 지원 메소드에 의해 객체 간 설정된 관계성 링크를 추적함으로써 관심 커뮤니티를 추출하는 방식에 대해 살펴보도록 한다.

4.1 시스템 질의를 이용한 관심 커뮤니티 추출 방식

시스템 질의 방식은 시스템 관리자가 원하는 관점으로 연관된 커뮤니티를 유동적으로 파악할 수 있도록 지원한다. 다음은 시스템 질의 SQ 내의 영역질의와 제약질의를 통해 특정 조건을 충족시키는 객체 집합, 즉 관심 커뮤니티를 추출하는 예제이다. 먼저, 예 10에서 인공지능을 이용한 인터넷 서비스는 비즈니스 전략과 밀접한 관련성을 지니고 있다. 이러한 관계를 이용하여 "internet"과 "artificial intelligence" 분야에 해당하는 포스트들 중 경영전략인 "competition strategy"과 관련성 있는 포스트들을 추출하기 위해 다음과 같은 시스템 질의를 사용할 수 있다.

예 10. 그림 7에서 '"internet"과 "artificial intelligence" 카테고리에 속하면서 "competition strategy" 카테고리에 속한 포스트들과 트랙백(trackback) 관계에 있는 포스트 집합을 보여달라'는 질의는 구체화 연산자를 이용해 다음과 같이 작성한다.

$$SQ_5 = (C_{computer}[c.Name="internet"] \wedge C_{computer}[c.Name="artificial\ intelligence"],\ C_{business}[c.Name="competition\ strategy"]*contains*trackback)$$

$O(Scope_Query) = \{c010\}$이므로, 시스템은 $C_{business}[c.Name = "competition\ strategy"]$ *contains*trackback $=\{p04, p05\} \subseteq O_{semantic\ web}$를 결과로 추출한다. 이와 같이 SQ는 객체 내에 설정되어 있는 세밀한 관계성을 이용하여 시스템 관리자의 의도에 적합한 객체들을 검색하는 질의에 효과적으로 이용될 수 있다.

이때 의도한 결과가 제시되지 않은 경우 관계성의 확장을 통해 SQ를 재생성 할 수 있다. 즉, $C_{business}[c.Name = "competition\ strategy"]$*contains*trackback 에 해당되는 검색결과가 없을 경우 관계성의 계층성을 이용해 trackback의 상위관계성인 associationOf로 관계성이 확장되어 $C_{business}[c.Name="competition\ strategy"]$ * contains*associationOf 로 변환이 가능하다. 이 같은 경우 "competition strategy"가 포함하는 p08과 associationOf 관계에 있는 p03 포스트가 결과로 반환된다. 검색결과가 없을 경우 관계성 확장 이외에 영역질의 내에서 검색 범위를 확장시킴으로써 사용자의 의도를 해치지 않는 범위 내에서 질의를 완화하여 검색 효율

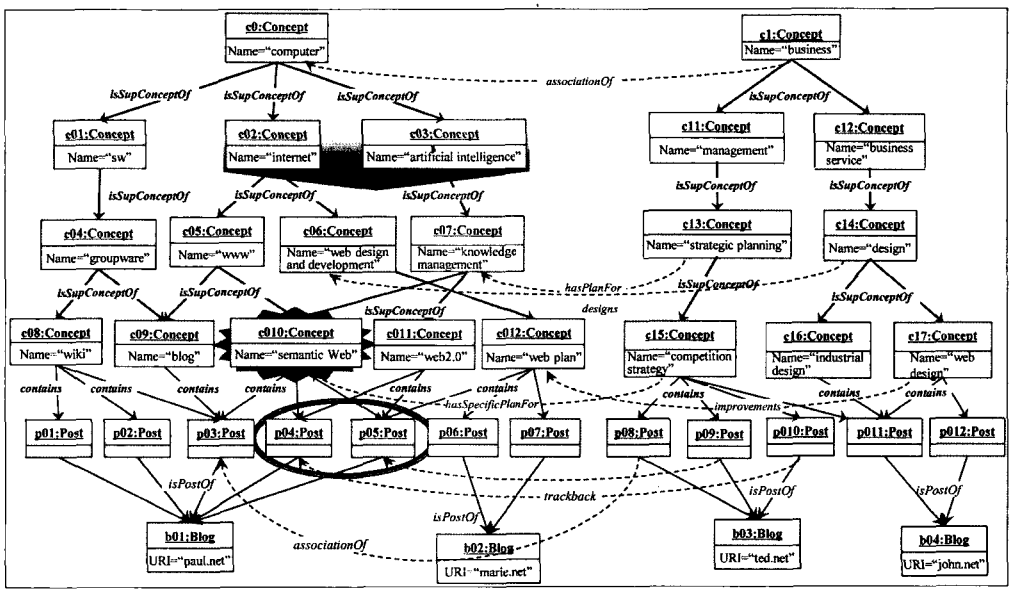

그림 7 SQ_5의 질의 처리 과정

을 높일 수 있다. 영역질의의 검색 범위 확장에 따른 질의 완화에 대한 자세한 사항은 [20]을 참고하기 바란다.

마지막 예로, 한 카테고리에 속한 포스트 집합 중 다른 포스트들에 비해 많은 횟수의 트랙백 링크가 걸려있거나 최다 횟수로 스크랩된 포스트들의 경우 해당 카테고리 내에 인기 있는 포스트일 가능성이 매우 높다. 다양한 관계성을 맺음으로써 다른 사용자들의 높은 호응을 얻는 인기 포스트는 유용한 내용이나 흥미 있는 내용을 포함할 가능성이 높으며 이러한 인기 포스트를 추출하여 아직 이를 아직 발견하지 못한 다른 블로그 사용자에게 제시한다면 인기포스트들에 대한 접근성을 획기적으로 높일 수 있다. 예 11은 질의 SQ_6 를 이용하여 가장 많이 스크랩된 인기 포스트들을 찾아 추출하는 예를 보인 것이다.

예 11. 그림 8에서 "web plan" 카테고리에 속한 포스트 중에 가장 스크랩이 많이 된 포스트들을 참조하는 질의 SQ_6는 특정 관계성에 대해 가장 참조 빈도 수가 높은 객체를 반환하는 함수인 maxFreqObject 함수를 이용하여 아래와 같이 표현된다. 다음 질의는 "web plan" 카테고리에 속한 포스트 중 최다 스크랩된 포스트 객체 집합을 결과로 반환한다.

$$SQ_6 = (C_{computer}[c.\text{Name} = \text{"web plan"}], \text{maxFreqObject}(B*\text{scraps}))$$

$O(C_{computer}[c.\text{Name}=\text{"web plan"}]) = \{c012\}$, $O_{c012} = \{c012, p05, p06, p07\}$이며 이 중 가장 많이 스크랩된 포스트로 $\{p07\} \subseteq O_{c012}$이 추출된다.

4.2 클러스터링 메소드를 이용한 관심 커뮤니티 추천

다음은 클러스터링을 지원하는 메소드인 DBMC(Density Based Metric Clustering)와 관련 링크를 추적하여 외부의 객체를 추출하는 메소드인 CBRA(Clustering Based Recommendation Algorithm)를 이용하여 관심 커뮤니티를 추천하는 방식에 대해 살펴보도록 한다.

• DBMC

객체들 사이에 다양한 상호작용이 일어날수록 즉, 객체들 사이에 서로 맺는 관계의 횟수가 잦아질수록 관계성의 밀집도는 높다고 볼 수 있으며 이는 강한 관계성을 가진다고 정의한다. 강한 관계성을 가진 포스트 집단은 하나의 클러스터가 되고 이는 공통된 관심사를 갖는 커뮤니티로 간주될 수 있다.

DBMC는 특정 포스트 객체 집합을 대상으로 강한 관계성을 기준으로 클러스터 집합을 형성하는 일종의 연산자이며 DBMC(P', r) 형식으로 사용된다. 입력인자 P'는 클러스터링 대상이 되는 포스트 객체 집합, 입력인자 $r \in R_2$은 P'에 설정된 관계성들 중 클러스터링의 기준이 되는 관계성을 나타낸다.

DBMC는 우선 두 객체 사이 관계성에 대한 관련 정도를 계산하는 함수 ϕ를 이용하여 P'가 맺고 있는 r 관계성들에 가중치를 할당한다: 즉, ϕ: $r \rightarrow W$, 여기서 W는 가중치로 사용되는 실수 집합. 각 관계성의 가중치는 $\phi(r) \geq 0$이고, 가중치의 범위는 [a, b]이다. $\phi(r)$의 가중치를 갖는 관계성 중 임계값이 δ 이하의 값을 갖는 약한 관계성은 제거된다. 약한 관계성을 제거하면 강한 관계성을 지닌 그룹화된 포스터 객체의 집합들이 생성되

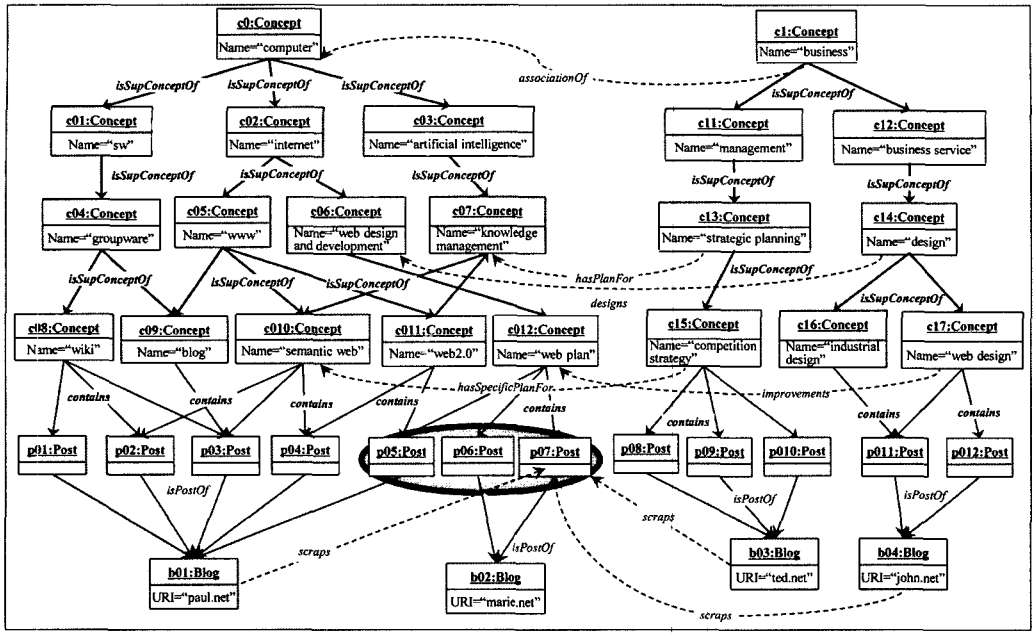

그림 8 SQ_6의 질의 처리 과정

며, 이들은 클러스터 집합을 의미하는 CS={cs_1, cs_2, …, cs_n}로 구분된다.

DBMC는 Metric-based Clustering Algorithm을 응용한 것으로 세부적인 알고리즘은 [21]을 참조하기 바란다.

예 12. 그림 9는 그림 1에서 보인 "computer" 객체를 최상위 객체로 가진 계층 중 "web plan" 개념 객체에 속한 포스트들 간의 trackback관계성에 대한 클러스터링 연산 전/후의 상태를 보이고 있다. 시스템 관리자는 DBMC를 이용하여 "web plan" 카테고리에 속하는 모든 포스트들에 대해 trackback 관계성을 기준으로 클러스터링 연산을 수행할 수 있다: SQ = (*, $C_{computer}$ [c.Name="web plan"]*contains) 질의를 사용하여 연산을 적용할 객체 집합 P` = $O(SQ)$를 구한 뒤 trackback 관계성에 대해 DBMC를 적용하면 다음의 클러스터링 집합을 얻을 수 있다:

DBMC(P`, trackback) = {cs_1, cs_2, cs_3}.

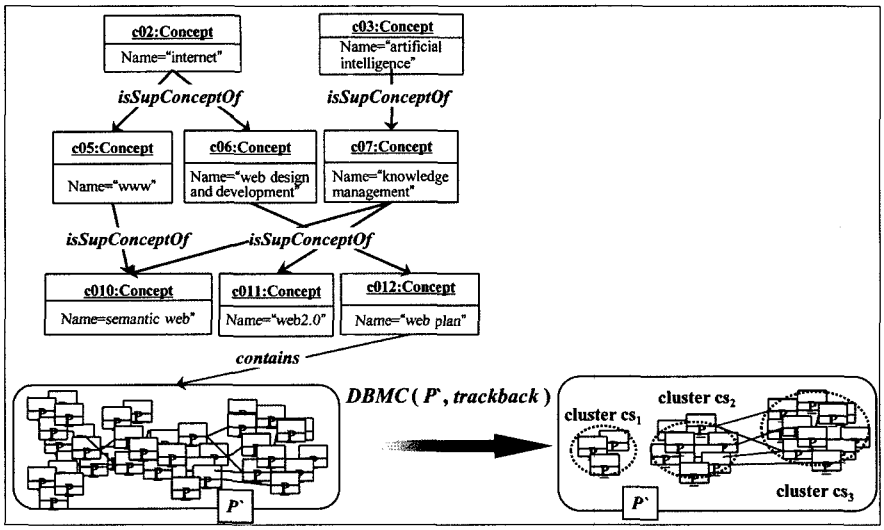

그림 9 DBMC의 연산 과정

```
// 입력1 : DBMC(P', r)
// 입력2 : r'
CBRA ( DBMC(P', r ), r' )
{
// GS : Cluster set
// C : Concept Object
초기화 :
    클러스터집합 초기화, GS = DBMC(P', r)
반복 :
    For each G ∈ GS
      - 클러스터 G와 r' 관계에 있는 외부 포스트 집합 P1 추출.
      - P1을 가장 많이 포함하는 객체 C 추출. C = maxFreqObject ( P1 * isContained )
      - C가 포함하고 있는 모든 포스트 집합 P2 추출.
      - P1 와 P2의 차집합인 P3 추출.
      - P3를 G 에 추천.
    End For
}
```

그림 10 클러스터링 기반의 객체추천 알고리즘

• CBRA

CBRA는 DBMC 수행 결과인 클러스터 집합을 입력값으로 받아 각 클러스터가 외부 카테고리의 객체들과 맺고 있는 관계성들을 분석하여 외부의 관심 포스트 객체들을 추출하는 연산자이다. CBRA는 CBRA(DBMC (P' r), r') 형식으로 사용되며 입력인자 r'는 외부 카테고리에 속한 객체 간 관계성 링크들 중 관심이 되는 특정 관계성을 선정하기 위한 기준이다. 다음은 CBRA의 Pseudo-code를 보인 것이다.

CBRA는 외부의 관심 포스트 객체 집합 P_1을 r'에 의해 추출한 후, maxFreqObject 함수를 사용하여 P_1을 부분적으로 가장 많이 포함하고 있는 상위 개념객체 c를 얻는다. c를 얻은 후 c가 포함하는 모든 포스트들을 검색하여 결과적으로 연관되는 외부 관심 포스트 객체 범위를 확장하게 된다.

예 13은 예 12에서 보인 클러스터 {cs_1, cs_2, cs_3}를 대상으로 trackback을 기준으로 CBRA를 이용하여 외부 카테고리에 속한 관련 커뮤니티를 추출하는 과정, 즉 CBRA (DBMC($O(SQ)$, trackback), trackback)을 보이고 있다. 설명의 편의상 한 클러스터 $cs_3 \in$ DBMC($O(SQ)$, trackback)에 대해서만 수행 과정을 보이도록 한다.

예 13. 그림 11은 cs_3를 대상으로 외부 객체들과 맺고 있는 trackback 관계성 링크를 추적하는 과정을 보인 것이다. ① CBRA({cs_3}, trackback)일 때 포스트 객체 집합 P_1을 형성한다. ② P_1을 포함하고 있는 모든 개념 객체들 중 최대 출현 빈도를 지닌 개념 객체 "web design(c017)"을 얻는다. 즉, maxFreqObject(P_1*isContained) = {c017}. ③ "web design" 카테고리에 포함된 포스트 객체 집합을 P_2라고 하고, P_1과 P_2의 차집합인 포스트 객체 집합을 P_3라고 했을 때 P_3는 CBRA의 연산결과가 된다. 즉, CBRA({cs_3}, trackback) = {P_3}.

CBRA 메소드는 cs_3와 직접적인 관계성을 가진 객체 집합 P_1을 추출할 뿐만 아니라 상위 개념객체 추적을 이용해 객체 추출의 범위를 확장함으로써 묵시적으로 cs_3과 연관 가능성이 높은 외부 포스트 객체 집합 P_3를 추출한다. 이와 같이, CBRA 메소드는 DBMC와 함께 사용되어 직간접적으로 연관되어 있는 관심 커뮤니티들을 인식하고 추천하는데 유용하게 활용될 수 있다.

4. 결론 및 향후 연구 과제

독립적인 속성을 지니는 블로그 자원이 다양한 장소에 분산되어 있는 블로그 공간에서 사용자가 원하는 정확한 정보를 추출하는데 한계가 있다. 또한, 현재 블로그 환경은 동적으로 상호작용하는 블로그 공간 내부에서 발생하는 커뮤니티 활동을 블로그 사용자 서비스에 반영하는 체계적인 방법론은 지원하지 못하고 있다. 본 논문에서는 이 문제를 해결하기 위해 계층화된 개념 정보인 온톨로지를 블로그 환경 내에 지식정보로 활용함으로써 블로그 공간을 체계적으로 분석, 관리하는 OSEM 모델을 제안하고 이를 정형화하였다. 제안 모델에서 시스템 관리자는 온톨로지 용어 사이에 설정된 다양한 관

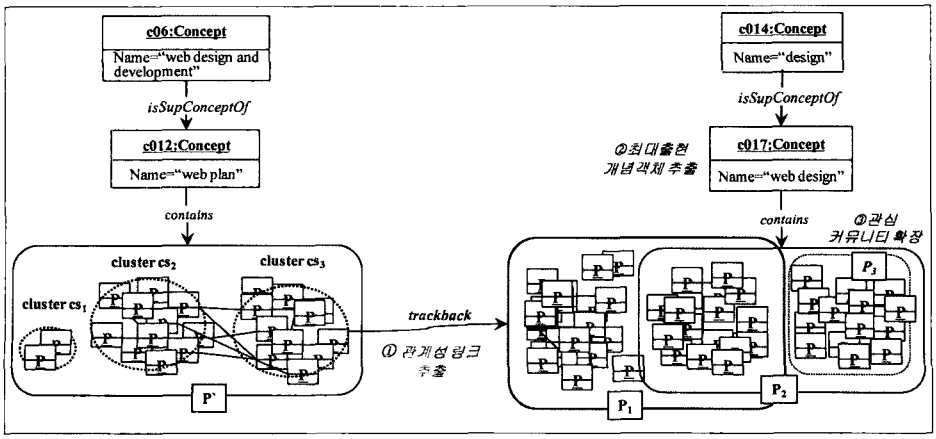

그림 11 CBRA의 연산 과정

계성 링크를 추적하고 분석함으로써, 블로그 사용자들이 다양한 관점에 따라 유기적으로 원하는 블로그 자원에 접근할 수 있도록 지원한다. 특히, OSEM에서 제공하는 객체와 관계성, 연산자들을 통해 시스템 관리자가 의도한 정확한 블로그 자원을 브라우징하고 참조할 수 있도록 하였으며 이를 기반으로 세부적이고 구체적인 시스템 질의를 생성하여 적절한 블로그 자원을 검색할 수 있도록 하였다. 또한, 블로그들 간에 이루어지는 상호작용 활동을 파악하고 상호작용을 기반으로 동적으로 관심 커뮤니티를 형성하여 사용자에게 추천하는 기능을 제공함으로써 블로그 사용자 간의 관심 커뮤니티 간의 효율적인 접근 방식을 지원하도록 하였다.

OSEM 은 현재 여러 포털 사이트에서 운영되고 있는 블로그 시스템들이나 인적 구성망 서비스 시스템들에서 사용자의 관심사항에 맞는 적합한 블로그 자원이나 커뮤니티들을 추천할 수 있는 방식으로 용이하게 적용될 수 있다. 이들 시스템의 기반 모델로서 제반 관련 데이타 마이닝이나 사용자 프로파일 분석 기법 등이 체계적으로 접목된다면, 관련 사용자 서비스를 한 차원 높일 수 있는 기회를 제공할 수 있을 것으로 기대한다.

향후 연구로 블로그 자원들의 인적 구성망간 상호작용 활동을 시스템 관리자가 일목 요연하게 시각적으로 살펴 볼 수 있는 그래픽 인터페이스를 정의하고 개발할 필요가 있다. 또한 블로그 사용자의 태그들을 커뮤니티별로 클러스터링하여 사용자 레벨의 상호작용에 따른 결과가 온톨로지 레벨에 동적으로 반영되게 함으로써 커뮤니티의 요구에 따라 블로그 자원을 유연성 있게 관리할 수 있도록 할 필요가 있다.

참 고 문 헌

[1] RSS(RDF Site Summary), <http://web.resource.org/rss/1.0>

[2] T. Finin, L. Ding, and L. Zou., "Social networking on the semantic web," The Learning Organization, 2005.

[3] Quintarelli, Emanuele., "Folksonomies: power to the people," UniMIB meeting, 2005.

[4] de.licio.us, <http://del.icio.us>

[5] Flickr, <http://www.flickr.com>

[6] Cayzer, Steve., "What next for semantic Blogging," Proceeding of the SEMANTICS 2006 conference, pp. 71-81, 2006.

[7] SIOC, <http://sioc-project.org>

[8] R.Karger, David. and Quan, Dennis., "What would it mean to blog on the semantic web?," Journal of Web Semantics, Vol.3, No.2, pp. 147-157, 2005.

[9] Ohmukai, Ikki., Numa, Kosuke. And Takeda, Hidaeki., "Egocentric Search Method for Authoring Support in Semantic Weblog," Workshop on Knowledge Markup and Semantic Annotation, 2003.

[10] Wright, Alex., "Folksonomy," <http://www.agwright.com/blog/archives/0000900.html>

[11] Hayes, Conor., Avesani, Paolo. and Veeramachaneni, Sriharsha., "An Analysis of Bloggers and Topics for a Blog Recommender System," 7th European Conference on Machine Learning and the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD), 2006.

[12] D. Brickley, S Buswell, B Matthews, L Miller, D Reynolds, M Wilson(2002) : SWAD-Europe: Semantic Web Advanced Development in Europe Presented at the International Semantic Web Conference

[13] Semantic Blogging (HP), <http://www.semantic-blogging.org>

[14] SPARQL(Query Language for RDF), < http://www.w3.org/TR/rdf-sparql-query>

[15] Friendster. http://www.friendster.com

[16] Mika, P., "Flink: Semantic web technology for the extraction and analysis of social networks," Journal of Web Semantics, Vol. 3, pp. 211-223, October 2005.

[17] Ding, Li., Zhou, Lina., Finin, T. and Joshi, A., "How the Semantic Web is Being Used: An Analysis of FOAF Documents," System Sciences, Proceedings of the 38th Annual Hawaii International Conference, 2005.

[18] Kim, Ki-Heon., Yang, Jae-Dong., Choi, Jae-Hun., Yang, Kyung-Ah. and Ha, Young-Guk., "A Semantic Inheritance/Inverse-Inheritance Mechanism for Systematic Bio-Ontology Construction," Proceedings of the 29th Annual International Conference of the IEEE EMBS, France, 2007.

[19] Choi, Jae Hun., Yang, Jae Dong. and Lee, Dong Gil., "An Object-Based Approach to Managing Domain Specific Thesauri: Semiautomatic Thesaurus Construction and Query-Based Browsing," Int'l Journal of Software Engineering & Knowledge Engineering, Vol.10, No.4, pp. 1-27, 2002.

[20] Li, Wen-Syan., Candan, K. SelcËuk., Vu, Quoc. and Agrawal, Divyakant., "Query Relaxation by Structure and Semantics for Retrieval of Logical Web Documents," IEEE Transactions on Knowledge and Data Engineering, Vol.14, No.4, pp. 768-791, 2002.

[21] Chricota, Yves., Jourdan, Fabien. and Melancon, Guy., "Software Components capture using Graph Clustering," Program Comprehension, 11th IEEE International Workshop, 2003.

양 경 아

2001년 한국교원대학교 컴퓨터교육학과(학사). 2003년 전북대학교 컴퓨터정보학과(석사). 2004년 전북대학교 컴퓨터통계정보학과 박사수료. 현재 케이테크 멀티미디어DB 연구소 연구원. 관심분야는 온톨로지, 시맨틱웹, 소셜 네트워크 서비스, 정보검색

양 재 동

1983년 서울대학교 컴퓨터공학과(학사) 1985년 한국과학기술원 전산학과(석사) 1991년 한국과학기술원 전산학과(박사) 1995년~1996년 Univer.of Florida, Visiting Scholar. 현재 전북대학교 전자정보공학부 교수. 관심분야는 OODBs, Expert System, CASE, 온톨로지

최 완

1981년 경북대학교 전자공학과(학사). 1983년 한국과학기술원 전산학과(석사). 1985년 한국과학기술원 전산학과 연구조교. 1988년 정보처리기술사(전자계산응용). 현재 한국전자통신연구원 팀장. 관심분야는 소프트웨어 공학, 미들웨어, 소프트웨어 스트리밍