Journal of Korea Multimedia Society Vol. 17, No. 2, February 2014(pp. 240-247)
http://dx.doi.org/10.9717/kmms.2014.17.2.240

온톨로지를 이용한 변화하는 데이터의 효과적인 인덱싱 방법

김종욱†, 배명수††

요 약

웹상에서 생성·공유되는 데이터는 다양한 분야에서 대용량으로 생성되고, 콘텐츠가 사회적 관심에 따라 지속적으로 변화 한다는 특징이 있다. 이로 인하여, 웹 데이터를 분석하여 유용한 정보를 얻기 위해서는 (a) 대용량의 데이터를 빠르게 처리하고, (b) 사용자가 쉽게 정보를 찾을 수 있도록 데이터를 구성하는 것이 필수적이다. 이러한 두 가지 측면 중에서, 본 논문은 사용자의 정보 검색 부담을 덜어주기 위해 온톨로지를 활용한 데이터 구성 방법을 제시한다. 특히, 본 논문에서는 콘텐츠가 사회적 관심에 따라 지속적으로 변화하는 웹 데이터의 특성을 고려하여, 데이터 콘텐츠를 인덱싱하기에 가장 적합한 온톨로지를 기존에 존재하는 범용 온톨로지로부터 추출한다. 또한, 사례 연구를 통하여 제시한 알고리즘의 유용성을 보인다.

Effective Indexing for Evolving Data Collection by Using Ontology

Jong Wook Kim†, Myung Soo, Bae††

ABSTRACT

Data which is created and shared on the Web is characterized by the massive amount of user generated content on various applications and dynamically evolving content on the basis of user interests. Thus, in order to benefit from Web data, it is essential to provide (a) the mechanisms which enable scalable processing of large data collections and (b) the organization schemes which reduce the navigational overhead within complex and dynamically growing content. Between these two impending needs, in this paper, we are interested in developing an indexing scheme which aims to reduce the time and effort needed to access the relevant piece of information by leveraging ontologies. In particular, considering evolving nature of Web contents, the proposed technique in this paper computes the sub-ontology, which best matches a given data collection, from the existing large size of ontology. Case studies show that the proposed indexing scheme in this paper indeed helps organize dynamically evolving content.

Key words: Evolving Content(콘텐츠), Ontology(온톨로지), Navigation(정보검색)

1. 서 론

Web 2.0 기술의 발달로 인하여 웹상에서 생성·공유되는 데이터는 기존의 정적인 데이터(static data)와는 다른 특성을 가지고 있다. 웹 데이터는 (a) 일반 사용자에 의해 다양한 분야에서 대용량으로 생성되고, (b) 콘텐츠가 사회적 관심에 따라 지속적으로 변화 한다는 특징이 있다. 이로 인하여, 웹 데이터를

※ 교신저자(Corresponding Author) : 배명수, 주소 : 서울시 송파구 풍납2동 서울아산병원 아산생명과학연구원 의료영상로봇연구실(138-736), 전화 : 02) 3010-6346, FAX : 02) 3010-6196, E-mail : msbae21@gmail.com
접수일 : 2013년 11월 4일, 수정일 : 없음
완료일 : 2013년 12월 18일

† 상명대학교 미디어소프트웨어학과
(E-mail : jkim@smu.ac.kr)
†† 서울아산병원 영상의학과
※ 본 연구는 2013학년도 상명대학교 교내연구비를 지원받아 수행하였음.

활용하여 유용한 정보를 얻기 위해서는 효율성(efficiency)과 효과성(effectiveness)이라는 두 가지 측면을 동시에 고려해야 한다. 먼저, 효율성이라는 측면에서는 대용량의 데이터를 빠르게 처리하기 위한 기법이 요구되며, 이로 인하여 현재 다양한 방법이 활발하게 연구 진행되고 있다. 그 예로, 맵리듀스(MapReduce) 기법은 배치지향 시스템 상에서 대용량의 데이터를 동시에 처리하기 위한 방법을 제공해 주고 있다[1,2]. 또한, 병렬데이터베이스 시스템(PDBMS)은 비공유시스템(shared nothing architecture)상에서 SQL 질의를 효율적으로 처리하기 위해 사용되고 있다[3,4].

두 번째로, 효과성이라는 측면은 사용자의 정보검색 부담을 덜어줄 수 있는 방식을 제공해 주는 것을 의미한다. 현재 일반적으로 쓰이는 기법은 온톨로지(ontology)를 기반으로 하여 데이터 콜렉션(data collection)을 인덱싱(indexing)하는 방법이 있다. 온톨로지를 사용하여 데이터를 계층적으로 조직하는 기법은 사용자들이 쉽고 빠르게 대용량의 데이터를 탐색(navigation)해 나갈 수 있기 때문에, 사용자의 정보검색 부담을 덜어줄 수 있다는 장점을 가지고 있다. 그러나 콘텐츠가 지속적으로 변화하는 웹 데이터의 특성 때문에, 이 기법은 유지비용이 많이 든다는 단점을 가지고 있다. 예를 들어, 데이터 콘텐츠가 지속적으로 변화함에 따라 현재 사용하고 있는 온톨로지가 데이터를 인덱싱하기에 적합하지 않게 될 수 있다. 이 경우 전문가로 하여금 온톨로지를 수정하게 하여야 하며, 이는 많은 시간을 필요로 한다. 그 대안으로 전문가의 개입 없이 기존에 존재하는 온톨로지를 재사용 하는 방식이 다양한 분야에서 널리 사용되고 있다[5,6]. 그러나 이 방법 역시 기존에 존재하는 온톨로지가 데이터 콘텐츠보다 광범위 한 경우, 혹은 반대로 온톨로지가 데이터 콘텐츠보다 한정적인 경우, 효과적인 인덱싱을 할 수 없다는 문제점을 가지고 있다.

1.1 논문의 초점과 공헌

본 논문에서는 WordNet [7], Wikipedia [8], ODP (Open Directory Project) [9] 같은 전문가에 의해 생성/ 유지되는 범용의 대형 온톨로지(large-scale ontology)로부터, 현재 데이터 콜렉션을 인덱싱하기에 가장 적합한 서브 온톨로지를 구하는 기법을 제시한다. 본 논문에서 제시한 기법은 다음과 같은 응용 프로그램 환경에서 사용되어 진다. 웹 데이터처럼 사회적 관심에 따라 콘텐츠가 지속적으로 진화하는 데이터는 콘텐츠의 변화에 따라 인덱싱에 사용되는 온톨로지도 변화되어야 한다. 그림 1에서 보여 지듯이, WordNet [7], Wikipedia [8], ODP [9] 같은 범용의 대형 온톨로지 A가 존재 할 때, 특정 시점의 데이터 콜렉션을 인덱싱하기에 적합한 서브 온톨로지를 B라고 가정 하자. 데이터 콘텐츠가 진화함에 따라 서브 온톨로지 B가 더 이상 데이터 콜렉션을 인덱싱하

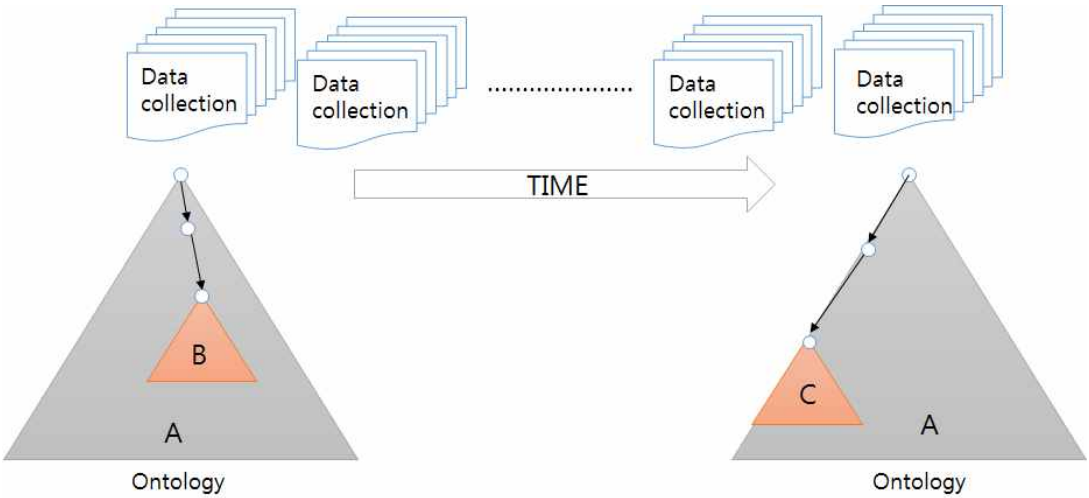

그림 3. 특정 시점에서 데이터 콜렉션을 인덱싱하기에 가장 적합한 서브 온톨로지 B는 콘텐츠의 변화에 따라 데이터를 인덱싱하기에 부적합 하게 된다. 이 경우, 현재 데이터 콘텐츠를 표현하기에 가장 적합한 다른 서브 온톨로지 C를 구하여 데이터를 인덱싱할 수 있다.

기에 부적합하게 되었다고 가정하자. 그 경우 범용의 대형 온톨로지 A에 속하는 다른 서브 온톨로지 C를 이용하여 데이터 콜렉션을 인덱싱할 수 있다. 본 논문에서 제시한 방법은 이러한 응용 프로그램 환경 내에서 특정 시점의 데이터 콜렉션을 인덱싱하기에 가장 적합한 서브 온톨로지를 구하는데 활용되어 진다.

본 논문은 다음과 같은 순서로 구성되어 있다. 2장에서는 논문에서 다루는 문제를 정의한다. 3장에서는 논문에서 제시하는 알고리즘을 설명한 후, 4장에서 사례 연구를 통하여 제시한 알고리즘의 유용성을 보인다. 마지막으로, 5장에서는 결론과 향후 연구 과제들을 논하다.

2. 문제 정의

본 장에서는 논문에서 다루는 문제를 정의 한다. 범용의 대형 온톨로지를 O(C, E)이라 정의 하자. 이때 C = {c_1, c_2, c_3, ..., c_m}는 온톨로지에 존재하는 컨셉 노드(concept node)들의 집합이고, E는 두 컨셉 노드를 연결하는 방향성 있는 에지(directed edge)들의 집합이다. 만일 컨셉 노드 c_i가 컨셉 노드 c_j의 부모 노드(parent node)에 해당되면, 방향성 있는 에지 $e_{i,j}$가 존재한다. 특정 시점의 데이터 콜렉션 DB = {d_1, d_2, ..., d_n}가 존재할 때, 본 논문에서는 DB를 인덱싱하기에 가장 적합한 서브 온톨로지 O_{db}(C_{db}, E_{db})를 범용의 대형 온톨로지 O(C, E)로부터 구한다. 여기서 C_{DB}와 E_{DB}는 각각 C와 E의 서브셋(subset)에 해당한다.

3. 제안하는 알고리즘

본 논문에서 제안한 기법은 컨셉 노드 $c_i \in C$와 데이터 $d_j \in DB$ 사이의 유사성을 이용하여 데이터 콜렉션을 인덱싱 하기에 가장 적합한 서브 온톨로지를 범용의 대형 온톨로지로부터 구한다. 제안하는 알고리즘은 다음과 같이 요약할 수 있다(그림 2).

- 데이터 $d_j \in DB$와 컨셉 노드 $c_i \in C$를 키워드 공간상에 맵핑(mapping) 한다. 키워드 공간상에 맵핑된 데이터와 컨셉노드 사이의 유사 행렬(similarity matrix)를 구한다 (3.1장).
- 유사행렬에 co-clustering 기법을 적용하여, 컨셉 노드들을 클러스터링 한다 (3.2장).
- 클러스터링된 컨셉 노드들을 이용하여 서브 온톨로지를 구하고, 데이터 콜렉션을 인덱싱한다(3.3장).

3.1 데이터와 컨셉 노드 사이의 유사성

3.1.1 데이터-키워드 행렬 (P)

정보 검색(information retrieval) 연구 분야에서 키워드 벡터를 이용하여 문서, 데이터를 나타내는 방식은 가장 보편적인 기법이다. 데이터 $d_i \in DB$에 해당하는 데이터 벡터를 $\overrightarrow{v_{di}}$ = {$w_{i,1}$, $w_{i,2}$, $w_{i,3}$, ... $w_{i,r}$}라 가정하자. 여기서 $w_{i,o}$는 데이터 d_i상에 존재하는 키워드 k_o의 전체 개수에 해당되며, r은 사전(dictionary)에 존재하는 모든 키워드들의 개수에 해당한다. 데이터 콜렉션 DB는 n개의 열과 r개의 행으로 구성된 데이터-키워드 행렬 P로 나타내어진다. 이때, n은 데이터 콜렉션 DB에 존재하는 전체 데이터의 개수 (즉, n=|DB|)에 해당되며, i-번째 열 P[i,-]는 키워드 벡터 $\overrightarrow{v_{di}}$ 에 해당 된다 (즉, P[i,-] = $\overrightarrow{v_{di}}$).

3.2 컨셉 노드-키워드 행렬(M)

키워드 벡터를 이용하여 나타낸 데이터와 온톨로지상에 존재하는 컨셉 노드사이의 효과적인 매칭을 수행하기 위해, 본 논문에서는 컨셉 노드를 키워드 공간상에 맵핑(mapping) 한다. 온톨로지에 존재하는 컨셉 노드를 키워드 공간상에 맵핑하는 방식은 데이터 마이닝과 (data mining)과 정보 검색분야에서 활발하게 연구 되어 왔다 [10, 11, 12]. 본 논문에서는 CP/CV [10] 기법을 사용하여, 온톨로지에 있는 각각의 컨셉 노드를 키워드 공간상에 맵핑한다. 온톨로지 O(C, E)가 주어졌을 때, CP/CV 기법은 온톨로지의 구조적 정보(structural information)를 이용하여, 컨셉 노드 $c_i \in C$를 키워드 공간상의 컨셉 벡터 $\overrightarrow{v_{ci}}$ ={$w_{i,1}$, $w_{i,2}$, $w_{i,3}$, ... $w_{i,r}$}로 맵핑한다 [10]. 컨셉 벡터를 이용하여 온톨로지 O(C, E)는 m개의 열과 r개의 행으로 구성된 컨셉-키워드 행렬 M으로 표현된다. 여기서 m은 전체 컨셉 노드들의 개수에 해당 하며 (즉, m=|C|), i-번째 열 M[i,-]는 컨셉 벡터 $\overrightarrow{v_{ci}}$ 에 해당 된다 (즉, M[i,-] = $\overrightarrow{v_{ci}}$).

3.1.3 데이터와 컨셉 노드 사이의 유사도(Similarity)

3.1.1장과 3.1.2장에서 데이터 콜렉션 DB와 온톨로지 O(C, E)를 데이터-키워드 행렬(P)와 컨셉-키워드 행렬(M)으로 각각 나타내었다. 이 두 개의 행렬을 이용하여, 본 장에서는 데이터와 컨셉 노드 사이의 유사성을 나타내는 유사행렬(Similarity Matrix)를 계산한다. 유사행렬 S는 n개의 열과 m개의 행으로 구성된 행렬로서, 유사행렬 S의 열(row)는 데이터 콜렉션 (DB = {d_1, d_2, d_3, ..., d_n})을 의미하며, 행(column)은 컨셉 노드들의 집합(C = {c_1, c_2, c_3, ..., c_m})을 각각 의미한다. 이때 유사행렬 S의 i-번째 열과 j-번째 행의 값 S[i,j]는 다음과 같이 계산한다.

S[i,j] = Similarity($\overrightarrow{v_{di}}$, $\overrightarrow{v_{cj}}$) = Similarity(P[i,-], M[j,-])

S[i,j]의 값은 데이터 $d_i \in$DB와 컨셉 노드 $c_j \in$C 사이의 유사도를 의미하며, 일반적으로 코사인 유사도(cosine similarity)를 이용하여 계산한다. 데이터와 컨셉 노드는 동일한 키워드 공간상에 존재하기 때문에, Similarity(P[i,-], M[j,-]) = cos(P[i,-], M[j,-])는 다음과 같이 계산한다.

$$\cos(P[i,-], M[j,-]) = \frac{\sum_{t=1}^{r}(P[i,t] \times M[j,t])}{\sqrt{\sum_{t=1}^{r} P[i,t]^2} \times \sqrt{\sum_{t=1}^{r} M[j,t]^2}}$$

유사행렬 S는 위 공식에서 의미 하듯이, 데이터 콜렉션에 존재하는 데이터와 온톨로지에 존재하는 컨셉 노드사이의 유사성을 나타낸다.

3.2 Co-clustering을 이용한 컨셉 노드 클러스트링

Co-clustering 기법은 2-차원 행렬로 표현된 데이터 (예, 유사행렬)의 행과 열을 동시에 클러스트링 하는 기법으로, biclustering 혹은 two-mode clustering이라고 불린다[13,14]. 일반적으로 co-clustering 기법은 두 데이터 집합 X = {x_1, x_2, ..., x_z}와 Y = {y_1, y_2, ..., y_w}, 두 데이터 집합 사이의 관계를 나타내는 행렬이 주어졌을 때, X를 k (단, k≤z)개, Y를 h (단, h≤w)개의 분리 클러스터(disjoint cluster)로 각각 다음과 같이 나눈다.

$$X = \{x_1, x_2, \ldots., x_z\} \rightarrow \hat{X} = \{\hat{x_1}, \hat{x_2}, ..., \hat{x_k}\}$$
$$Y = \{y_1, y_2, \ldots., y_w\} \rightarrow \hat{Y} = \{\hat{y_1}, \hat{y_2}, ..., \hat{y_h}\}$$

여기서, $\hat{x_i}$와 $\hat{y_i}$는 각각 X와 Y의 부분집합(subset)에 해당하여 다음을 만족한다.

$$X = \hat{x_1} \cup \hat{x_2} \cup .... \cup \hat{x_k}$$
$$Y = \hat{y_1} \cup \hat{y_2} \cup .... \cup \hat{y_h}$$

본 논문에서 co-clustering 기법을 사용하는 주된 이유는 온톨로지상에 존재하는 컨셉 노드들 중에, 데이터 콜렉션 DB와 유사도가 높은 컨셉 노드들을 선별하기 위함이다. 이러한 목적을 염두에 두고, 본 논문에서는 다음과 같이 co-clustering 기법을 유사행렬 S에 적용하여, 다음과 같은 분리 클러스터를 구한다.

$$DB = \{d_1, d_2, \ldots., d_n\} \rightarrow \{\hat{d_1}\}$$
$$C = \{c_1, c_2, \ldots., c_m\} \rightarrow \{\hat{c_1}, \hat{c_2}\}$$

즉, n개의 열과 m개의 행으로 구성된 유사행렬 S에 co-clustering기법을 적용하여, 컨셉 노드에 해당되는 m개의 행을 두 개의 분리 클러스터 $\hat{c_1}$과 $\hat{c_2}$로 나눈다. 유사행렬(S)를 기반으로 하여, $\hat{c_1}$는 데이터 콜렉션 DB와 유사도가 상대적으로 높은 컨셉 노드들의 집합에 해당하며, $\hat{c_2}$는 유사도가 상대적으로 낮은 컨셉 노드들의 집합에 해당된다. 또한 데이터 콜렉션 DB는 한 개의 클러스터가 형성되도록 co-clustering 기법을 적용한다. 본 논문에서 일반적인 클러스터링 기법[15-17]을 사용하지 않고, co-clustering기법을 이용하는 이유는 데이터 콜렉션상에 존재하는 이상치(outlier)와 유사도가 높은 컨셉 노드가 $\hat{c_1}$에 포함되는 것을 방지하기 위함이다. 이상치(outlier)와 유사도가 높은 컨셉 노드는 데이터 콜렉션에 존재하는 다른 데이터들과의 유사도가 상대적으로 낮기 때문에, co-clustering기법을 사용하면 이러한 컨셉 노드들이 $\hat{c_1}$에 포함되는 것을 방지할 수 있다.

3.3 온톨로지를 이용한 데이터 콜렉션 인덱싱 방법

3.2장에서 co-clustering 기법을 활용하여 온톨로지상에 존재하는 컨셉 노드들을 두 분리 클러스터 $\hat{c_1}$과 $\hat{c_2}$로 나누었다. 3.2장에서 설명하였듯이 $\hat{c_1}$에 존재하는 컨셉 노드들은 데이터 콜렉션과 상대적으로 높은 유사성을 가지고 있는 노드들의 집합이기 때문에, 본 논문에서는 $\hat{c_1}$에 존재하는 컨셉 노드들을 이용

하여 데이터 콜렉션을 인덱싱한다. 먼저, $\hat{c_1}$에 존재하는 컨셉 노드들중에서, 데이터 $d_i \in DB$와 가장 유사도가 높은 컨셉 노드 c_{di}를 다음과 같이 구한다.

$$c_{di} = MAX_{c_t \in \hat{c_1}} (\cos(d_i, c_t)) = MAX_{c_t \in \hat{c_1}} (S[i,t])$$

DB상에 존재하는 각각의 데이터와 가장 유사도가 높은 $\hat{c_1}$상에 존재하는 컨셉 노드 (즉, 위의 공식을 이용하여 구한 컨셉 노드)들의 집합을 $\widehat{c_{DB}}$라 가정하자. 그러면, 데이터 콜렉션을 인덱싱하기 위해 적합한 서브 온톨로지 $O_{DB}(C_{DB}, E_{DB})$는 $\widehat{c_{DB}}$상에 존재하는 모든 컨셉 노드들을 포함하는 최소 트리(minimum tree)에 해당되며, 이는 그림 3에 제시한 알고리즘을 이용하여 O(C, E)로부터 구한다. 1번과 2번에서 C_{DB}와 E_{DB}를 각각 공집합(Φ)으로 초기화 한다. 3번에서는 온톨로지 O(C, E)의 루트 노드(root_node)를 초기화 한다. 4번에서는 함수 GetRootNode()를 이용하여, 서브 온톨로지 $O_{DB}(C_{DB}, E_{DB})$의 루트 노드 (root_$node_{DB}$)를 구한다. 5번에서 함수 GetNodeAnd-Edge()를 이용하여 $\widehat{c_{DB}}$상에 존재하는 모든 컨셉 노드들을 포함하고, 루트 노드가 $root_node_{DB}$에 해당하는 최소 트리를 구한다.

마지막으로, 그림 3에 제시한 알고리즘을 이용하여 구한 최소 트리에 해당하는 서브 온톨로지 O_{DB} (C_{DB}, E_{DB})를 이용하여, 각각의 데이터 $d_i \in DB$가 컨셉 노드 $c_{di} \in C_{DB}$에 할당되도록 데이터 콜렉션을 구성한다. 이러한 온톨로지를 활용한 데이터 인덱싱 방법은 사용자가 쉽게 원하는 정보를 탐색해 나갈 수 있는 수단을 제공해 줌으로써, 사용자의 정보 검색 부담을 덜어줄 수 있다.

Pseudo-code for computing minimum tree which contains all concept nodes in $\widehat{c_{DB}}$ from O(C, E)
Input : O(C, E), $\widehat{c_{DB}}$
output: $O_{DB}(C_{DB}, E_{DB})$

```
1:  C_DB ← Φ
2:  E_DB ← Φ
3:  root_node ← root node of O(C, E)
4:  root_node_DB ← GetRootNode(root_node, C, E, ĉ_DB)
5:  GetNodeAndEdge(root_node_DB, C, E, C_DB, E_DB, ĉ_DB)
6:  return O_DB(C_DB, E_DB)

GetRootNode(root_node, C, E, ĉ_DB)
7:  for each child_node of root_node
8:      if ( sub-tree rooted at child_node subsumes all concept nodes in ĉ_DB)
9:          return GetRootNode(child_node, C, E, ĉ_DB)
10: return root_node;

GetNodeAndEdge(root_node, C, E, C_DB, E_DB, ĉ_DB)
11: for each child_node of root_node
12:     if ( sub-tree rooted at child_node contains any concept node in ĉ_DB)
13:         C_DB ← C_DB ∪ child_node
14:         E_DB ← E_DB ∪ e_root_node,child_node
15:         ĉ_DB ← ĉ_DB - child_node
16:         if ( ĉ_DB == Φ)
17:             return
18:         else
19:             GetNodeAndEdge(child_node, C, E, C_DB, E_DB, ĉ_DB)
20: return
```

그림 3. $\widehat{c_{DB}}$상에 존재하는 모든 컨셉 노드들을 포함하는 최소 트리 (minimum tree)를 구하는 알고리즘

4. 사례 연구

본 장에서는 사례연구 분석(case study)를 통하여 논문에서 제안하는 온톨로지를 활용한 데이터 인덱싱 방법의 타당성을 검증한다. 특히, 사례연구 분석을 통해 논문의 3장에서 제시한 알고리즘 (즉, 범용의 대형 온톨로지로부터 데이터 콜렉션을 인덱싱하기에 적합한 서브 온톨로지를 구하는 것)의 유용성을 보인다. 실험에서 사용한 데이터 콜렉션은 다음과 같다. ACM Digital Library [17]로부터 컴퓨터 과학(Computer Science)와 관련된 논문들의 요약문 100개를 수집했다. 실험에서 사용한 온톨로지는 ODP [9]로부터 1880개의 컨셉 노드들로 구성된 온톨로지를 추출 하였다. 실험에서 사용한 온톨로지는 Computers, Math, Business, Arts와 같은 다양한 분야를 포함 하고 있다.

논문에서 제안하는 기법의 타당성을 검증하기 위해 서브 온톨로지를 3장에서 제시한 알고리즘을 이용하여 구했다. 그리고 각각의 논문 요약문을 서브 온톨로지의 컨셉 노드에게 3.3장에서 설명한 방식을 이용하여 맵핑 하였다. 표 1은 컨셉 노드에 맵핑된 데이터의 개수에 따라 컨셉 노드들을 내림차순으로 정렬했을 때, 가장 순위가 높은 5개의 컨셉 노드들에 해당한다.

그림 4는 제안한 알고리즘을 이용하여 구한 서브 온톨로지의 일부에 해당한다. 그림 4에서는 서브 온톨로지에 속하는 전체 컨셉 노드 141개 중 설명 목적으로 일부분만을 보여 준다. 그림에서 보여 지듯이 우리가 구한 서브 온톨로지의 루트 노드는 Computer에 해당하며, 서브 온톨로지에 속하는 모든 컨셉 노드들은 Computer의 후손 노드(descendant node)에 해당한다. 이러한 실험 결과는 본 논문에서 제시

표 1. 컨셉 노드에 맵핑된 데이터의 수에 따라 내림차순으로 정렬했을 때, Top-5에 해당하는 컨셉 노드들

순위	컨셉 노드 (Concept Node)
1	Computers: Algorithms: Conferences
2	Computers: Algorithms: Conferences: Past
3	Computers: Computer Science: Database Theory
4	Computers: Human-Computer Interaction: Conference
5	Computers: Software: Software Engineering

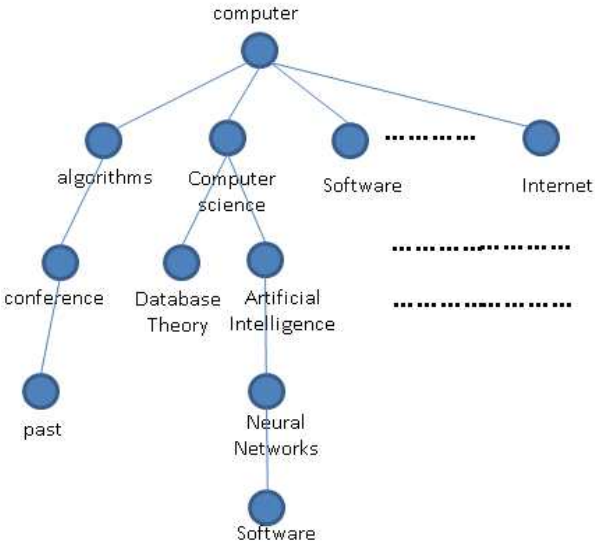

그림 4. 논문에서 제안한 알고리즘을 이용하여 구한 서브 온톨로지의 일부

표 2. 실험에서 사용한 전체 온톨로지와 제안한 알고리즘을 이용하여 구한 서브 온톨로지에서 Computer의 후손 노드의 개수 비교

	실험에서 사용한 전체 온톨로지	제안한 알고리즘을 이용하여 구한 서브 온톨로지
Computer의 후손 노드의 개수	526	140

안 (1) co-clustering 기법을 사용하여 먼저 데이터 콜렉션과 유사도가 낮은 컨셉노드를 제거한 후, (2) 유사도가 높은 컨셉 노드들만을 이용하여 데이터를 인덱싱하는 방법이 이상치(outlier)를 제거하는데 효과적인 방법임을 보여준다.

마지막으로 표 2에서는 Computer를 루트 노드로 한 서브 온톨로지와 논문에서 제시한 알고리즘을 이용하여 구한 서브 온톨로지를 비교한다. 먼저, 실험에서 사용한 온톨로지에서 Computer의 전체 후손 노드의 개수와 논문에서 제안한 알고리즘을 이용하여 구한 서브 온톨로지에서 Computer의 전체 후손 노드의 개수를 비교한다. 표에서 보여 지듯이, 제안한 알고리즘을 이용하여 구한 서브 온톨로지는 Computer의 후손 노드들 중 일부분만을 사용함을 알 수 있다. 이는 $\hat{c_1}$에 속하는 노드들중에서 데이터 콜렉션과 유사도가 상대적으로 낮은 컨셉 노드들은 서브 온톨로지 생성 단계에서 배제하기 때문이다. 이

러한 방식은 인덱싱에 사용될 온톨로지의 크기를 작게 유지할 수 있게 한다. 그러므로, 사용자가 불필요한 컨셉 노드를 탐색하는 것을 방지할 수 있으며, 이는 정보 검색 부담을 덜어 줄 수 있다는 효과를 가져온다.

본장에서 수행한 사례분석 연구는 논문에서 제시한 알고리즘이 데이터 콜렉션을 인덱싱하기에 적합한 온톨로지를 효과적으로 구할 수 있음을 보인다. 또한, 이러한 온톨로지를 사용한 데이터 인덱싱 기법은 사용자가 불필요한 컨셉 노드를 탐색하지 않게 함으로써, 정보 검색을 부담을 덜어 줄 수 있다는 효과가 있다.

5. 결 론

본 논문에서는 데이터 콘텐츠가 지속적으로 변화는 응용프로그램 환경 내에서 온톨로지를 활용 데이터 콜렉션을 인덱싱하는 방법을 연구 했다. 특히, 기존에 존재하는 전체 온톨로지를 사용하기 보다는, 데이터와 온톨로지상에 존재하는 컨셉 노드들간의 유사성을 이용하여, 데이터 콜렉션과 밀접한 연관성이 있는 컨셉 노들로 구성된 서브 온톨로지를 구하는 방법을 제안한다. 사례 연구에서는 실 데이터를 사용하여 본 논문에서 제안하는 기법의 유용성을 보였다.

참 고 문 헌

[1] J. Dean and S. Ghemawat. “MapReduce: Simplified Data Processing on Large Clusters,” *Symposium on Opearting Systems Design and Implementation*, pp. 137-150, 2004.

[2] J.W. Kim, “Data Partitioning on MapReduce by Leveraging Data Utility,” *Journal of Korea Multimedia Society*, Vol. 16, No. 5, pp. 657-666, 2013.

[3] Teradata, http://www.teradata.com. 1979.

[4] IBM Netezza Data Warehouse Appliances, http://www-01.ibm.com/software/data/netezza/, 2000.

[5] M. Cataldi, K.S. Candan, and M.L. Sapino, “Narrative-based Taxonomy Distillation for Effective Indexing of Text Collections,” *Data and Knowledge Engineering*, Vol. 72, No 2, pp. 103-125, 2012.

[6] O. Zamir and O. Etzioni, "Web Document Clustering: A Feasibility Demonstration," *Proc. of the International ACM SIGIR Conference*, pp. 46-54, 1998.

[7] WordNet, A lexical database for English, http://wordnet.princeton.edu/, 2013.

[8] Wikipedia, http://www.wikipedia.org/, 2001.

[9] Open Directory Project, http://www.dmoz.org/, 1998.

[10] J.W. Kim and K.S. Candna, "CP/CV: Concept Similarity Mining without Frequency Information from Domain Describing Taxonomy," *Proc. of the International ACM CIKM Conference*, pp. 483-492, 2006.

[11] M. Cataldi, C. Schifanella, K.S. Candan, M.L. Sapino, and L.D. Caro, "CoSeNa: A Context-based Search and Navigation System," *Proc. of the International Conference on Management of Emergent Digital EcoSystems*, pp. 218-225, 2009.

[12] L.D. Caro, K.S. Candan, and M.L. Sapino, "Using tagFlake for Condensing Navigable Tag Hierarchies from Tag Clouds," *Proc. of the International ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1069-1072, 2008.

[13] I.S. Dhillon, S. Mallela, and D.S. Modha, "Information-Theoretic Co-clustering," *Proc. of the International ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 89-98, 2003.

[14] I.S. Dhillon, "Co-clustering Documents and Words using Bipartite Spectral Graph Partitioning," *Proc. of the International ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 269-274, 2001.

[15] J. Zhao and G. Karypis, "Evaluation of Hierarchical Clustering Algorithms for Docu-

ment Datasets," *Proc. of the International ACM CIKM Conference*, pp. 515-524, 2002.
[16] R.T. Ng and J. Han, "Efficient and Effective Clustering Methods for Spatial Data Mining," *Proc. of the International Conference on Very Large Data Bases*, pp. 144-155, 1994.
[17] ACM Digital Library. http://portal.acm.org, 2014.

김 종 욱

1994년 3월~2000년 8월 고려대학교 전산과학과 학사
2000년 9월~2002년 8월 한국과학기술원 전산학과 석사
2004년 1월~2009년 12월 Arizona State University Computer Science 박사
2009년 10월~2010년 8월 Technicolor Member Research Staff
2010년 9월~2013년 8월 Teradata, Software Engineer
2013년 9월~현재 상명대학교 미디어소프트웨어학과 조교수

배 명 수

1997년 4월 Eastern Michigan University, Computer Science 학사
1999년 12월 Arizona State University, Computer Science 석사
2008년 8월 Arizona State University, Computer Science 박사
2008년~2010년 Arizona State University(Poly.), Research Scientist
2010년~2013년 이화여자대학교 컴퓨터그래픽스/가상현실연구센터 연구교수
2013년~현재 서울아산병원 영상의학과 연구교수
관심분야 : 컴퓨터그래픽스, 기하학적 모델링, 가상현실, 패턴인식