http://dx.doi.org/10.5392/JKCA.2016.16.06.205

그래프 데이터에 대한 비-중복적 키워드 검색 방법

A Method for Non-redundant Keyword Search over Graph Data

박창섭
동덕여자대학교 컴퓨터학과

Chang-Sup Park(cspark@dongduk.ac.kr)

요약

최근 소셜 네트워크, 시맨틱 웹, 바이오 인포매틱스 등 여러 응용 분야에서 그래프 구조를 갖는 대용량 데이터들에 활용됨에 따라 이런 데이터들에 대한 키워드 기반 검색 방법이 많은 관심을 받고 있다. 본 논문에서는 그래프 구조 데이터에 대한 키워드 질의에 대해 질의와 연관성이 높으면서 구조적인 중복성을 갖지 않는 top-k 결과 집합을 효율적으로 검색하는 방법을 제안한다. 키워드 질의에 대한 비-중복적인 결과 트리 구조와 그것의 연관도 척도를 정의하고, 그래프 내에 포함된 유용한 경로 정보들에 대한 효과적인 인덱싱 방법을 제안한다. 그리고 기 생성된 인덱스를 활용하여 주어진 키워드 질의에 대해 비-중복적이면서 연관도가 큰 top-k 결과 집합을 생성하는 효율적인 질의 처리 알고리즘을 제시한다. 실 데이터를 이용한 실험을 통해 제안한 방법의 효과와 성능을 기존 방법과 비교 분석한다.

■ 중심어 : | 그래프 데이터 | 키워드 검색 | Top-k 질의 처리 |

Abstract

As a large amount of graph-structured data is widely used in various applications such as social networks, semantic web, and bio-informatics, keyword-based search over graph data has been getting a lot of attention. In this paper, we propose an efficient method for keyword search over graph data to find a set of top-k answers that are relevant as well as non-redundant in structure. We define a non-redundant answer structure for a keyword query and a relevance measure for the answer. We suggest a new indexing scheme on the relevant paths between nodes and keyword terms in the graph, and also propose a query processing algorithm to find top-k non-redundant answers efficiently by exploiting the pre-calculated indexes. We present effectiveness and efficiency of the proposed approach compared to the previous method by conducting an experiment using a real dataset.

■ keyword : | Graph Data | Keyword Search | Top-k Query Processing |

I. 서 론

최근 그래프 구조를 갖는 데이터들이 소셜 네트워크, 온라인 커뮤니티, 시맨틱 웹, 온톨로지, 바이오인포매틱스 등 여러 응용 분야에서 사용되고 있다. 노드와 간선들로 구성되는 그래프 데이터는 객체에 대한 정보뿐만

* 본 연구는 2015년도 동덕여자대학교 연구년제도에 의하여 수행되었습니다.
접수일자 : 2016년 01월 15일
수정일자 : 2016년 02월 12일
심사완료일 : 2016년 02월 12일
교신저자 : 박창섭, e-mail : cspark@dongduk.ac.kr

아니라 객체들 사이의 관계를 효과적으로 나타낸다. 그래프 데이터의 양이 크게 증가함에 따라 효과적이고 효율적인 검색 방법의 필요성이 커지고 있다. 특히 사용자의 편의성 향상을 위해 정보 검색(IR) 분야에서 널리 사용되는 키워드 검색 방식이 주목받고 있다[1-10].

그래프 데이터에 대한 키워드 검색은 일반적으로 질의 키워드가 포함된 노드들로 이루어 진 서브-트리(subtree) 또는 서브-그래프(subgraph)들을 찾아 검색 결과로 반환한다. 이런 결과 구조는 질의 키워드를 포함하는 노드들이 그래프 내에서 어떻게 연결되어 있는지를 나타낸다. 대규모 그래프 데이터에는 질의와 관련된 결과 구조들이 매우 많이 존재할 수 있으므로 일반적으로 연관도(relevance) 척도를 사용하여 질의와의 연관성이 가장 큰 top-k 개의 결과 구조들을 찾는다.

대용량 데이터에 대한 효율적인 질의 처리를 위해 많은 방법들이 서브-트리를 질의 결과로 사용한다[1-6][8-10]. 특히 개별 루트 시맨틱(distinct root semantic)을 따르는 방법들은 각 노드에 대해 그것을 루트로 갖는 서브-트리들 중 연관도 척도의 값이 가장 높은 것 하나만을 고려하여, 서로 다른 루트 노드를 갖는 서브-트리들의 집합을 결과로 생성한다[2][3][5][10]. 그러나 이 방법들의 공통적인 문제점은 루트 노드가 질의 키워드를 하나도 포함하지 않으면서 자식 노드를 하나만 갖는 비효과적인 서브-트리가 질의 결과로 생성될 수 있다는 점이다. 본 논문에서는 이러한 트리를 중복적 답 트리(redundant answer tree)라 부른다. 중복적 답 트리는 질의에 대한 답이 될 수 있으나, 그것이 내부적으로 포함하고 있는 비-중복적인 답 트리와 매우 유사한 정보를 나타낸다. 중복적 답 트리들을 포함하는 검색 결과는 다음과 같은 단점을 가진다. 첫째, top-k 답 트리 집합에 유사한 답 트리들이 포함되어 질의 결과의 다양성(diversity)이 감소한다. 이것은 질의 결과에서 다양한 답들을 얻기를 희망하는 사용자의 만족도를 저하시킬 수 있다. 둘째, 만일 어떤 비-중복적인 답 트리가 사용자의 검색 의도와 관련이 없거나 적다면 그것을 서브-트리로 포함하는 중복적인 답 트리들도 모두 그러하다. 이런 트리들 대신에 다른 답 트리들을 구한다면 검색 효과를 더 높일 수 있다.

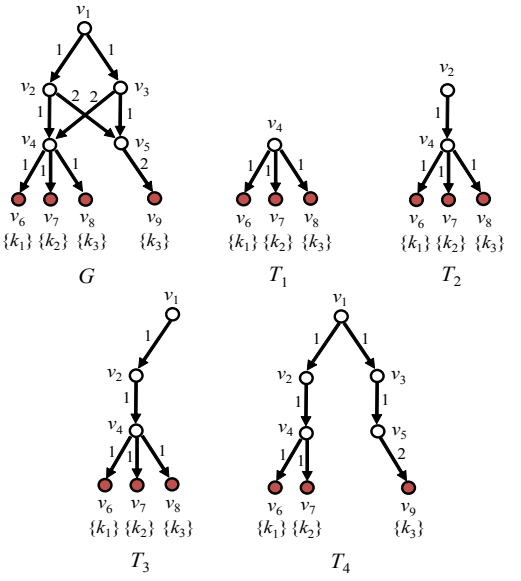

그림 1. 중복적/비-중복적 답 트리의 예

예를 들어, [그림 1]에 제시된 유향 가중 그래프 G에서 키워드 k_1 ~ k_3가 노드 v_6 ~ v_9에 포함되어 있다고 가정하자. 각 간선에 부여된 숫자는 인접한 두 노드 사이의 거리를 뜻한다. 키워드 질의 $q = \{k_1, k_2, k_3\}$가 주어질 때, 서브-트리 T_1 ~ T_4는 모두 q에 속한 키워드들을 포함하는 답 트리이다. 이 중 T_3와 T_4는 노드 v_1을 루트로 갖고 키워드 k_3에 대해 각각 v_8과 v_9, 그리고 그것들에 대한 최단 경로를 포함하는 서브-트리이다. v_1에서 v_8로 가는 최단 경로의 길이가 v_1에서 v_9로의 최단 경로의 길이보다 짧으므로, 개별 루트 시맨틱을 따르는 기존 방법들은 T_3의 질의 연관도가 T_4의 그것보다 더 높다고 간주하여 v_1을 루트로 갖는 답 트리로 T_3 하나만 선택한다. 그러나 T_3는 v_4를 루트로 갖고 크기가 더 작은 비-중복적인 답 트리 T_1을 자신의 서브-트리로 포함하므로 중복적인 답 트리이다. 반면, 노드 v_9과 그것에 대한 경로를 포함하는 서브-트리 T_4는 다른 답 트리를 서브-트리로 포함하지 않는 비-중복적인 트리이다. 이 경우, 중복적 트리인 T_3보다 연관도는 더 작으나 비-중복적 트리인 T_4를 v_1에 관한 답 트리로 선택하는 것이 질의 결과의 다양성 면에서 더 바람직하다고 할 수 있다. 한편, v_2를 루트로 갖는 서브-트리 T_2도 T_1을

포함하는 중복적인 트리임을 고려할 때, 기존 방법에서는 많은 중복적인 트리들이 top-k 결과에 포함될 가능성이 크다.

본 논문에서는 그래프 데이터에 대한 키워드 검색 결과의 다양성을 위해 중복적이지 않으면서 연관도가 큰 서브-트리들을 구하는 방법을 제안한다. 먼저 비-중복적인 구조를 갖는 답 트리와 그것의 연관도 척도를 정의한다. 효율적인 질의 처리를 위해 개별 루트 시맨틱을 기반으로 하되 각 루트 노드에 대한 답 트리의 조건을 완화하여 비-중복적이면 연관도가 가장 큰 top-k 트리 집합을 질의 결과로 정의한다. 질의 처리 성능을 높이기 위해 유용한 경로 정보들을 그래프로부터 미리 계산하여 인덱스를 생성하고 질의 처리 시 이를 활용한다. 인덱스 검색을 통해 발견되는 답 트리에 대해 중복성 여부를 판별하고 이에 대한 비-중복적 대안 트리를 효율적으로 생성할 수 있는 알고리즘을 제시한다.

II. 관련 연구

그래프 데이터에 대한 키워드 검색 결과로 질의 키워드들을 모두 포함하는 서브-트리 집합을 구하는 기존 방법들은 서브-트리의 연관도 계산 방식과 질의 결과로 선택되는 답 트리의 조건에 따라 스테이너 트리 시맨틱(Steiner tree semantic)에 기반한 방법[1][4][6]과 개별 루트 시맨틱을 따르는 방법[2][3][5][10]으로 크게 구분될 수 있다. 이 중 후자의 방법들은 각 서브-트리의 질의 연관도를 루트에서부터 각 키워드를 포함하는 노드까지의 최단 경로들에 대한 함수로 계산하고, 그래프의 각 노드마다 그것을 루트로 갖는 서브-트리들 중 연관도 값이 가장 큰 것 하나만을 질의 결과의 후보로 고려함으로써 전자의 방법들보다 더 효율적이고 실용적이다[11]. 질의에 대한 답 트리는 각 질의 키워드를 포함하는 노드들 중 루트에서부터의 최단 경로의 길이가 가장 짧은 것과 그 경로를 포함한다. 답 트리의 가중치는 각 키워드에 대해 선택된 경로들의 거리의 합으로 정의되고 서로 다른 루트를 갖는 답 트리들 중 가중치가 가장 작은 것들을 질의 결과로 선택한다.

기존 연구 중 양방향 탐색[2] 방법은 BANKS[1]의 역방향 탐색 방법을 개선한 것으로, 답 트리를 찾기 위해 그래프의 유향 간선에 대한 역방향 탐색과 순방향 탐색을 동시에 실행한다. 그러나 탐색 방향과 탐색 대상 노드를 선택하기 위해 휴리스틱(heuristic)에 의존함으로써 일반적인 그래프에 대한 탐색 성능을 보장할 수 없다. BLINKS 방법[3]에서는 그래프 내의 경로들에 대한 인덱스를 정의하여 질의와 관련된 서브-트리들을 효율적으로 검색한다. 인덱스는 그래프 내의 각 노드와 키워드 쌍에 대해 키워드를 포함하는 노드들에 대한 최단 경로를 미리 계산하고 그것에 대한 정보를 역 리스트와 맵 구조로 저장한다. 그래프 대신 인덱스를 탐색하여 최소 비용 답 트리들을 효율적으로 생성하는 top-k 알고리즘을 제안하고 인덱스의 읽기 회수에 대해 최적임을 증명하였다. 본 논문에서 제안하는 방법은 기본적으로 BLINKS와 유사한 인덱싱 방법과 알고리즘을 이용한다. 한편, [5]에서는 크기가 매우 큰 그래프 데이터에 대한 검색 시 I/O 횟수를 줄이기 위해 다중-입자(multi-granular) 그래프 표현을 제안하고 BANKS와 양방향 탐색을 확장한 알고리즘을 제시하였다. [10]에서는 질의에 대한 다양한 결과를 생성하기 위해 확장된 답 트리 구조와 순위 척도를 정의하고 인덱스를 활용한 효율적인 top-k 질의 처리 알고리즘을 제안하였다.

상기 방법들은 1장에서 기술한 바와 같이 중복적인 구조의 답 트리들을 질의 결과로 생성할 수 있다. BANKS나 양방향 탐색에서는 그래프 탐색 중에 중복적인 서브-트리를 탐지하여 제외시킬 수 있으나, 일반적으로 그런 서브-트리들이 기하급수적으로 많이 존재할 수 있으므로 실행 성능이 크게 저하된다. BLINKS는 질의 처리 시 인덱스에서 발견되는 답 트리의 중복성 여부를 고려하지 않으므로 많은 중복적 답 트리들이 생성될 수 있다. 이 방법은 개별 루트 시맨틱에 따라 루트 노드에서 도달가능한 키워드에 대해 최적의 경로 하나만을 계산하여 인덱스에 저장하므로, 만일 중복적인 답 트리를 탐지하여 제외시킨다 하더라도 인덱스 정보의 제한 때문에 그것과 같은 루트를 공유하는 다른 답 트리들이 모두 고려 대상에서 제외된다. 예를 들면, [그림 1]에서 중복적 답 트리 T_3에 대한 대안으로 T_4와 같은

트리를 검색할 수 없다. 이런 단점 때문에 top-k 질의 결과의 품질이 양방향 탐색보다 크게 저하될 수 있다.

III. 문제 정의

G(V, E)는 유향 가중 그래프로서, V에 속한 각 노드는 키워드들로 구성된 텍스트를 포함한다. 키워드 k를 포함하는 노드들의 집합을 V(k)라 표기한다. E에 속한 각 간선은 그것에 인접한 두 노드 사이의 거리 또는 근접도(proximity)를 의미하는 가중치를 갖는다. 인접하지 않은 두 노드 사이의 유향 경로의 길이는 경로에 속한 간선들의 가중치의 합으로 정의된다. 그래프 G에 대한 키워드 질의의 답은 다음과 같이 정의된다.

정의 1 (질의의 답). 데이터 그래프 G와 l개의 키워드들로 이루어진 질의 $q = \{k_1, k_2, \cdots, k_l\}$가 주어질 때, q에 대한 답은 l 개의 노드들의 멀티셋(multiset) $C = \{v_1, v_2, \cdots, v_l\}$를 포함하는 G의 서브-트리 T로, v_i는 각 키워드 k_i를 포함하며(즉, $v_i \in V(k)$) 또한 다음 조건들을 만족해야 한다($1 \le i \le l$).

(1) T는 루트에서부터 v_i까지의 최단 경로를 포함한다.
(2) T의 단말 노드(leaf node)들은 모두 C에 속한다.
(3) T의 루트 노드가 만일 하나의 자식 노드를 가지면 루트 노드도 C에 속한다. □

위 정의에서 C에 속한 각 노드 v_i를 키워드 k_i에 대한 *키워드 노드* 또는 *콘텐트 노드*라 부르고, 위와 같은 서브-트리 T를 질의에 대한 *비-중복적 답 트리*라 부른다. T의 루트 노드가 n이라 할 때, 조건 (1)은 그래프 G에서 n과 v_i 사이의 최단 거리 경로만이 답 트리의 구성요소가 될 수 있음을 뜻한다. 그 경로를 $n \rightarrow v_i$로 나타내고 간단히 *키워드 k_i로의 경로*라 부른다. 그리고 루트 노드 n과 키워드 노드들의 멀티셋 C에 의해 정의되는 답 트리를 $T(n, C)$라 표기한다.

위 정의의 조건들은 콘텐트 노드들 간의 관계를 나타내는데 꼭 필요한 노드들만 답 트리에 포함되도록 제한한다. 특히 조건 (3)은 루트 노드가 키워드 노드들을 연결하는데 필수적인 노드여야 함을 의미한다. 만일 어떤 답 트리 T가 이 조건을 만족하지 않으면, 그것의 루트 노드 n이 어떤 질의 키워드도 포함하지 않으면서 하나의 자식 노드만 가지므로 T와 같은 콘텐트 노드 집합 C를 포함하면서 조건 (3)을 만족하는 서브-트리 T'이 T 안에 존재한다. T'은 T보다 크기가 작으므로 질의 처리 과정에서 더 큰 연관도 점수를 부여받고 T보다 먼저 고려된다. 따라서 T는 불필요한 답 트리라 할 수 있다.

본 논문에서는 질의와 연관성이 높은 답 트리들을 찾기 위해 답 트리에 포함된 콘텐트 노드들과 그것들의 구조적 위치를 고려하여 트리의 연관도를 계산한다. 우선 정보 검색에서 널리 사용되는 TF-IDF 방법[13]을 이용하여 각 키워드를 포함하는 콘텐트 노드에 대해 키워드에 대한 연관도를 정량적으로 평가한다. 예를 들어, 노드 v 에 나타난 키워드 k의 출현 횟수를 tf(k, v)라 하고 그래프 내의 노드들의 총 개수와 k를 포함하는 노드들의 개수를 각각 |V|, df(k, V)라 하면, k에 대한 콘텐트 노드 v의 연관도는 다음과 같이 정의될 수 있다[16].

$$S(k,v) = \sqrt{tf(k,v)} \cdot \left(1 + \log\left(\frac{|V|}{df(k,V)+1}\right)\right)^2 \quad (1)$$

주어진 그래프에 속한 모든 키워드와 노드 쌍들에 대한 위 함수의 최대값을 S_{max}라 표기한다.

한편, 답 트리 T(n, C)의 구조적인 연관도를 계산하기 위해 루트 노드 n에서 C에 속한 각 노드 v_i 까지의 최단 경로의 길이 dist(n, v_i)를 고려한다. 루트 노드에서 콘텐트 노드까지의 최단 거리가 짧을수록 그것을 포함하는 답 트리의 질의 연관도는 높아진다고 간주한다. 주어진 질의에 대한 답 트리의 연관도 함수는 콘텐트 노드의 내용과 경로를 반영하여 다음과 같이 정의된다.

정의 2 (답 트리의 연관도). 질의 q에 대해, 노드 n을 루트로 갖고 각 질의 키워드 k_i에 대한 콘텐트 노드 v_i를 포함하는 답 트리 T가 주어질 때, q에 대한 T의 연관도 함수 S(T, q)는 다음과 같다.

$$S(T,q) = \sum_{1 \le i \le l} f_r(n, k_i, v_i)$$

$$f_r(n, k_i, v_i) = \frac{S(k_i, v_i)}{S_{\max}} \cdot \frac{1}{dist(n, v_i)+1} \quad \square$$

즉, q에 대한 T의 연관도는 각 키워드 k_i에 대한 키워드 노드 v_i와, 루트 노드 n에서 v_i까지의 최단 경로에 따라 결정되는 $f_r(n, k_i, v_i)$ 값들의 합으로 정의된다. 이 값을 각 키워드 경로에 대한 연관도 점수라 부른다.

IV. 질의 처리 방법

1. 인덱스 구조

본 논문에서 제안하는 인덱스 구조는 BLINKS와 유사하나 비-중복적 답 트리를 효율적으로 구하기 위해 확장된 구조를 갖는다. 정의 2의 연관도 함수를 고려하여, 그래프 내에서 각 노드와 키워드를 포함하는 다른 노드들 간의 경로를 미리 추출하고 그것들의 연관도 값을 계산하여 인덱스로 저장한 후 질의 처리 시 활용함으로써 복잡한 그래프에 대한 실시간 탐색을 피하고 top-k 질의 결과를 효율적으로 생성할 수 있다. 제안하는 인덱스 구조는 다음과 같은 요소들로 구성된다.

(1) **키워드-노드 리스트 집합 KNList:** 그래프 내에 포함된 각 키워드에 대해 하나씩 정의되는 역 리스트(inverted list)들의 집합으로, 키워드 k에 대한 리스트 KNList(k)는 k와 관련된 키워드 노드들에 대한 경로 정보를 연관도에 따라 정렬하여 저장한다. 그래프에 속한 노드 n 에서 k 를 포함하는 각 노드까지의 최단 경로들의 집합을 P(n, k)라 하고, P(n, k)에 속한 경로들 중 정의 2에 기술된 연관도 함수 $f_r(n, k, v_i)$ 값이 가장 큰 최적 경로를 $p_m(n, k)$라 하자. KNList(k)는 k를 직접 포함하거나 또는 k를 포함한 노드까지 경로가 존재하는 노드 n에 대해 경로 $p_m(n, k)$의 정보를 저장한다. 각 항목(entry)은 네 개의 필드 (n, kn, fn, rel)로 구성된다. kn 은 경로의 마지막 노드인 키워드 노드를, fn 은 루트 *n을 제외한* 첫번째 노드(즉, n의 다음 노드)를 나타내고, rel은 이 경로의 연관도 $f_r(n, k, kn)$ 값을 뜻한다. 이 항목들은 rel 값의 내림차순으로 정렬되어 저장된다. 이 인덱스는 질의 처리 시 그래프의 역방향 탐색을 효율적으로 수행할 수 있도록 지원한다.

(2) **최적 경로를 위한 노드-키워드 맵 NKMap:** 그래프 내의 각 노드 n과 키워드 k 쌍에 대한 최적 경로 $p_m(n, k)$의 정보를 저장하는 해시 테이블로, (n, k)를 키로 하여 관련된 (kn, fn, rel) 정보를 저장한다. 이 인덱스는 주어진 각 질의 키워드에 대한 최적 경로들을 효율적으로 찾음으로써 그래프에 대한 순방향 탐색을 효율적으로 지원한다.

(3) **대안 경로를 위한 노드-키워드 맵 $NKMap_s$:** 이 인덱스에서는 각 노드 n과 키워드 k 쌍에 대해 최적 경로 $p_m(n, k)$외의 다른 키워드 노드에 대한 경로 정보를 추가적으로 저장한다. n에서 k에 대한 키워드 노드들까지의 경로들 중 $p_m(n, k)$와 첫번째 노드 fn이 다르면서 연관도가 가장 큰 경로를 계산하여 그것에 관한 (kn, fn, rel) 정보를 저장한다. 이 정보는 질의 처리 시 비-중복적 답 트리를 효율적으로 구하기 위해 사용된다.

2. 질의 처리 알고리즘

본 논문의 키워드 검색 알고리즘은 [12]에서 제안된 임계 알고리즘을 기반으로 한다. 이 알고리즘은 다차원 속성을 갖는 객체들에 대한 유사도 기반 top-k 질의 처리에 널리 사용되며, 각 속성에 대해 정렬된 객체 리스트들을 이용한다. 속성 값들의 합이나 평균과 같은 단조(monotone) 집계 함수를 사용할 경우 이 알고리즘이 리스트 항목의 읽기 회수에 대해 최적임이 증명되었다.

본 논문에서 제안하는 알고리즘을 의사-코드로 기술하면 [그림 2]와 같다. 질의 $q = \{k_1, k_2, \cdots, k_l\}$가 주어질 때 각 키워드에 대응되는 키워드-노드 리스트들의 집합을 L(q)라 하자. 우선순위 큐 Q_t는 질의 처리 과정에서 발견된 답 트리들 중 연관도 점수, 즉 *l*개의 키워드 경로들의 연관도 값의 합이 가장 큰 k개의 답 트리들의 정보를 저장한다. 이 알고리즘은 L(q)에 속한 리스트들을 병렬적으로 순차 스캔(scan)하면서 항목들을 하나씩 읽어 들인다(line 3~4). 리스트에서 읽은 항목이 노드 n에 대해 처음 발견된 항목이면 *visitNode* 프로시저를 호출하여 n에 대한 처리를 수행한다(line 7). *visitNode*에서는 먼저 NKMap을 조회하여 n으로부터 각 질의 키워드에 대해 연관도가 최대인 경로 $p_m(n, k)$의 정보를 구한다(line 14~18). 모든 키워드에 대해 최적 경로 정보가 검색되면 n을 루트로 갖는 새로운 답 트리가 생성될 수 있다. 만일 이것이 중복적인 답 트리일 경우 다음

Input: $q=\{k_1, k_2, \cdots, k_l\}$, $L(q)=\{L_i | L_i=KNList(k_i)$ for $k_i \in q(1 \le i \le l)\}$, $k \in Z^+$
Output: a set of top-k reduced answer trees for q
Variables: a priority queue Q_t of size k, a set C of nodes visited, an array $curRel[1..l]$ of real numbers
1 $Q_t:=\varnothing$, $C:=\varnothing$, $curRel[i]:=0.0$ (for all $1 \le i \le l$)
2 **while** (an entry exists in a list in $L(q)$) {
3 Select a list L_i from $L(q)$ in a RR manner.
4 Read an entry (n, kn, fn, rel) at the current scan position in L_i.
5 $curRel[i] := rel$
6 **if** ($n \notin C$) {
7 **visitNode**(i, n, kn, fn, rel)
8 $C := C \cup \{n\}$
9 }
10 **if** $((|Q_t|=k) \wedge (rel_k \ge \sum_{1 \le i \le l} curRel[i]))$
break
11 }
12 Build top-k answer trees using the items in Q_t.
End

Procedure visitNode(i, n, kn, fn, rel)
Variables: arrays $V[1..l]$ and $A[1..l]$ of tuples ($nodeID$, $nodeID$, $relevance$)
13 $V[i]$:= (kn, fn, rel)
14 **for-each** ($k_j \in q$ such that $j \ne i$) {
15 Look up an entry (kn_j, fn_j, rel_j) with key (n, k_j) in $NKMap$ and store it into $V[j]$.
16 **if** (no entry is found) **return**
17 }
18 **if** ($V[j].fn=V[k].fn$ for all j, $k \in [1,l]$ and $V[j].kn \ne n$ for some $j \in [1,l]$) {
19 $A[j]:=\varnothing$ for all $j \in [1,l]$
20 **for-each** ($k_j \in q$)
21 Look up an entry (kn_j, fn_j, rel_j) with key (n, k_j) in $NKMap_s$ and store it into $A[j]$.
22 **if** ($A[j]=\varnothing$ for all $j \in [1,l]$) **return**
23 **else** {
24 Find j such that $A[j] \ne \varnothing$ and ($V[j].rel - A[j].rel$) is minimal($1 \le j \le l$).
25 $V[j] := A[j]$
26 }
27 }
28 Insert (n, V) into Q_t using $\Sigma_{1 \le i \le l} V[i].rel$ as its priority.
End

그림 2. 질의 처리 알고리즘

절에서 기술되는 방법을 이용하여 대안 트리를 구한다(line 18~27). 생성된 답 트리 정보를 Q_t에 저장한다(line 28).

이와 같은 과정에서 리스트들로부터 k개 이상의 답 트리들이 발견되고 그것들의 연관도 점수가 리스트에서 아직 발견되지 않은 모든 답 트리들의 연관도 점수보다 크거나 같다면 Q_t에 저장된 후보 트리들이 질의에 대한 top-k 결과라고 볼 수 있다. 위 알고리즘에서는 각 KNList(k_i)에서 항목을 읽을 때마다 배열 curRel[i]에 그것의 연관도 값을 저장한다(line 5). 각 리스트의 항목들은 연관도의 내림차순으로 정렬되어 있으므로, 이 배열에 저장된 값들의 합은 모든 리스트에서 아직 발견되지 않은 노드를 루트로 갖는 답 트리 T의 연관도 점수 S(T, q)의 상한(upper bound)이 된다. 따라서 Q_t에 저장된 k개의 후보 답 트리들 중 연관도 점수가 가장 작은 것의 점수를 rel_k라 하면, 다음 조건이 만족될 때 리스트 스캔을 종료하고 결과를 출력한다(line 10, 12).

$$(|Q_t|=k) \wedge (rel_k \ge \sum_{1 \le i \le l} curRel[i]) \quad (2)$$

3. 비-중복적 결과 생성

질의 $q = \{k_1, k_2, \cdots, k_l\}$가 주어질 때, 앞에서 기술한 질의 처리 알고리즘은 KNList와 NKMap 인덱스를 이용하여 그래프 내의 노드 n을 루트로 갖고 q에 속한 각 키워드 k_i에 대해 최적 경로 $p_m(n, k_i)(1 \le i \le l)$들을 포함하는 답 트리 T(n)을 생성한다. 이 트리가 정의 1의 조건 (3)을 만족하지 않는 중복적인 트리인 경우, 즉, 루트가 자식 노드를 하나만 가지면서 질의에 대한 키워드 노드로 선택되지 않은 경우, l개의 키워드 노드 경로들의 첫번째 노드는 모두 루트의 자식 노드가 된다. 따라서, NKMap에서 수집한 각 키워드 경로 정보의 fn 값들이 모두 동일한지 검사함으로써 답 트리의 중복성 여부를 알 수 있다. 단, 루트가 모든 키워드를 포함하고 키워드 노드로 선택된 경우에는 루트 노드 자체가 하나의 비-중복적 답 트리가 될 수 있음을 고려해야 한다(line 18).

T(n)이 중복적인 트리인 경우 n을 루트로 갖되 T(n)과 다른 키워드 노드 집합을 포함하는 비-중복적 답 트리를 찾아야 한다. 만일 이런 트리가 여러 개 존재하면 그 중 연관도 점수가 가장 큰 것을 선택해야 한다. 질의 키워드 k_i에 대해, n에서 $v \in V(k_i)$까지의 경로들 중

$p_m(n, k_i)$와 첫번째 노드가 같지 않으면서 연관도 $f_r(n, k_i, v)$가 가장 큰 경로를 $p_a(n, k_i)$라 하자. 그리고 T(n)에서 k_i에 관한 키워드 경로 $p_m(n, k_i)$를 제외하고 대신 $p_a(n, k_i)$를 포함하는 트리를 $T_i(n)$이라 하자. 이 트리는 k_i에 대한 키워드 경로가 다른 키워드 경로 $p_m(n, k_j)$들과 첫번째 노드가 일치하지 않으므로 비-중복적인 답 트리이다. 이제, 각 키워드에 대한 l개의 대안 트리 $T_i(n)(1 \le i \le l)$들 중 연관도 점수 $S(T_i, q)$가 가장 큰 것이 노드 n을 루트로 갖는 최적의 비-중복적 답 트리가 된다.

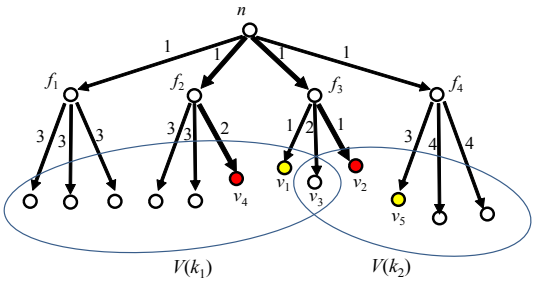

그림 3. 최적의 비-중복적 답 트리 탐색

예를 들어, [그림 3]의 그래프에서 키워드 k_1과 k_2를 포함하는 노드들의 집합 $V(k_1)$, $V(k_2)$가 각각 그림과 같고, 질의 $q = \{k_1, k_2\}$가 주어졌다고 가정하자. 각 키워드 k에 대한 키워드 노드 v의 연관도 S(k, v)가 모두 동일하다고 가정하면, 노드 n과 k_1에 대해 연관도가 가장 큰 경로 $p_m(n, k_1)$은 $n \rightarrow v_1$이고, n과 k_2에 대해 연관도가 가장 큰 경로 $p_m(n, k_2)$는 $n \rightarrow v_2$이다. 이 두 경로는 첫번째 노드로 f_3를 공유하므로 두 경로를 포함한 최적 서브-트리 T(n)은 중복적인 답 트리가 된다. 이 그래프에서 $V(k_1)$에 속한 노드들에 대한 경로들 중 f_3를 첫번째 노드로 갖지 않으면서 연관도가 가장 큰 것은 $n \rightarrow v_4$이고, $V(k_2)$에 속한 노드들에 대해서 f_3를 첫번째 노드로 갖지 않고 연관도가 가장 큰 경로는 $n \rightarrow v_5$이다. 이 대안 경로들을 이용하면 n을 루트로 갖는 비-중복적인 답 트리를 두 개 구할 수 있다. 즉, $T_1(n) = p_a(n, k_1) \cup p_m(n, k_2) = n \rightarrow v_4 \cup n \rightarrow v_2$과 $T_2(n) = p_m(n, k_1) \cup p_a(n, k_2) = n \rightarrow v_1 \cup n \rightarrow v_5$이다. 이 중 연관도 점수가 더 큰 $T_1(n)$이 비-중복적이면서 연관도가 가장 큰 답 트리가 된다. 만일 k_1이나 k_2에 대한 키워드 경로로 $n \rightarrow v_3$를 선택할 경우 그 결과는 여전히 중복적인 트리가 된다. 참고로 이 그래프에서 n을 루트로 갖는 비-중복적인 답 트리는 모두 36개가 존재한다.

[그림 2]의 알고리즘은 최적의 비-중복적 답 트리를 효율적으로 찾기 위해 기 생성된 인덱스 $NKMap_s$를 이용한다(line 18~27). $NKMap_s$는 노드 n과 키워드 k 쌍에 대해 최적 경로와 첫번째 노드가 다르면서 연관도가 가장 큰 경로인 $p_a(n, k)$를 저장하고 있으므로 각 질의 키워드에 대해 $NKMap_s$를 조회하여 대안 경로를 찾는다(line 20~21). 만일 모든 키워드에 대해 대안 경로가 존재하지 않으면 n을 루트로 갖는 비-중복적 답 트리는 존재하지 않는다(line 22). 앞에서 기술한 바와 같이 각 질의 키워드 k_i와 관련된 비-중복적 서브-트리 $T_i(n)$은 T(n)에서 $p_m(n, k_i)$ 대신 $p_a(n, k_i)$를 대체한 것이다. 모든 질의 키워드들 중 $p_m(n, k_i)$와 $p_a(n, k_i)$의 연관도 값의 차가 가장 작은 키워드 k_j를 찾아 $p_m(n, k_j)$ 대신 $p_a(n, k_j)$를 이용하면 n을 루트로 갖는 최적의 비-중복적 답 트리 $T_j(n)$을 구할 수 있다(line 24~25).

각 KNList 인덱스에 저장된 항목의 수는 최대 $|V|$개이므로 [그림 2]의 알고리즘은 $O(l \cdot |V|)$개의 항목을 읽는다. *visitNode* 프로시저에서 NKMap과 $NKMap_s$에 대한 조회수는 최대 $2l-1$이다. 따라서, 우선순위 큐 Q_t를 피보나치 힙으로 구현한다고 가정하면, 제안한 알고리즘의 이론적인 시간 복잡도는 $O(l \cdot |V| \cdot (l + logk))$이다.

V. 성능 평가

제안한 키워드 검색 방법의 효과와 성능을 평가하기 위해, 제안한 방법과 기존 BLINKS 방법을 Java 언어로 구현하고 실 데이터를 이용한 실험을 실시하였다. 실험을 위한 데이터는 세계 지리 정보 데이터베이스인 Mondial[1]을 활용하여, 대륙, 국가, 주, 도시, 정부, 종교, 산, 호수, 강, 바다, 섬, 사막 등의 객체들에 관한 데이터와 객체들 간의 관계들을 그래프로 모델링하였다. 이 그래프는 6,431개의 노드와 19,951개의 간선들로 구성되고, 15,815개의 키워드들을 포함한다. 그래프 데이터

1 http://www.dbis.informatik.uni-goettingen.de/Mondial/

구조를 메모리에 표현하고 노드들 간의 최단 경로를 계산하여 인덱스를 생성하기 위해 JGraphT[2] 라이브러리를 이용하였다. 또 식 (1)에 제시된 노드의 연관도를 계산하기 위해 Lucene[3] 라이브러리를 활용하였다. 본 실험에서는 편의상 간선의 가중치가 모두 1이라고 가정하고, 인덱스 공간과 질의 처리의 효율성을 위해 노드들 사이의 거리가 5이하인 경로들만 하고 답 트리의 구성요소로 사용한다. 본 실험은 2개의 2.0GHz Quad Core CPU와 16GB RAM이 설치된 Windows Server에서 실행되었다.

표 1. 테스트 질의

질의 ID	키워드 리스트
Q1	{caldera, lake, america}
Q2	{cape, gulf, africa}
Q3	{vienna, donau, alps}
Q4	{lake, quebec, canada}
Q5	{himalaya, india, pakistan}
Q6	{river, minnesota, louisiana}
Q7	{city, desert, california}
Q8	{lake, michigan, ontario}
Q9	{island, vancouver, seattle}
Q10	{alaska, arctic, sea}

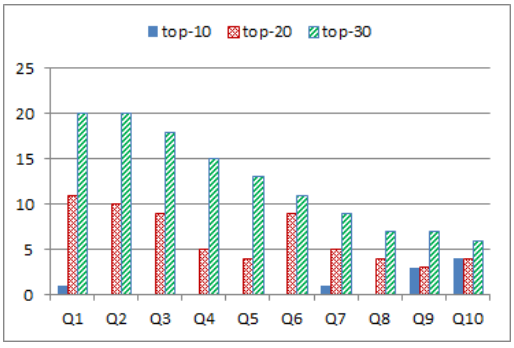

그림 4. BLINKS 결과의 중복적 답 트리 개수

[표 1]은 실험에 사용된 테스트 질의들 중 일부를 나타낸다. 이 질의들에 대해 본 논문에서 제안한 방법과 기존 BLINKS 방법, 그리고 BLINKS에서 중복적 답 트리를 식별하여 단순히 제외시키는 방법("BLINKS-N")을 각각 실행하였다. [그림 4]는 각 질의에 대해 BLINKS에 의해 생성된 top-10, top-20, top-30 결과의 중복적 답 트리의 개수를 나타낸다. 질의들에 대해 평균적으로 각각 1개, 6개, 13개(결과 크기 대비 10%, 30%, 43%)의 중복적 답 트리가 생성된 반면, 제안한 방법과 BLINKS-N은 모두 비-중복적인 답 트리들을 결과로 생성하였다.

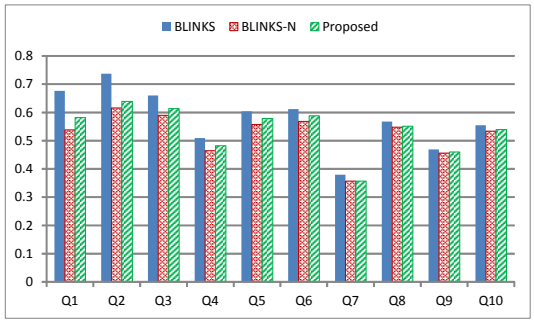

그림 5. 검색 결과의 연관도

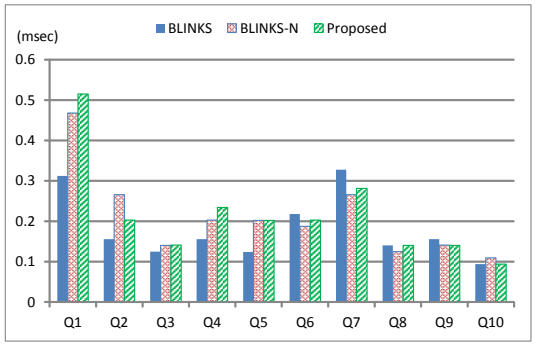

그림 6. 질의 처리 실행 시간

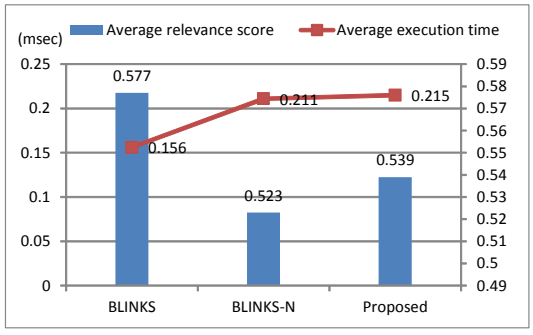

그림 7. 평균 연관도 및 실행 시간

2 http://jgrapht.org/
3 http://lucene.apache.org/

[그림 5]는 각 테스트 질의에 대해 세 가지 방법으로 구한 top-30 결과의 연관도 점수를 나타낸다. 본 논문에서 제안한 방법으로 생성된 결과의 연관도 점수가 BLINKS에 의해 생성된 결과보다 낮게 나타났으나 BLINKS-N의 결과에 비해서는 모두 높게 나타났다. [그림 7]에 도시된 전체 결과들의 평균 연관도 점수는 제안한 방법이 BLINKS에 비해 약 6.6% 낮았다. 이것은 중복적인 답 트리들을 제외하고 그 대신 연관도 점수가 낮은 비-중복적인 대안 트리들을 질의 결과에 포함했기 때문이다. 그러나 BLINKS-N에 비해서는 평균 연관도가 약 3.1% 높게 나타났다.

한편 [그림 6]은 각 테스트 질의에 대해 세 방법으로 top-30 질의 처리에 소요된 실행 시간을 나타낸다. [그림 7]의 평균 실행 시간을 비교해 보면 제안한 방법의 실행 시간이 BLINKS에 비해 약 37.8% 증가했으나 BLINKS-N과는 거의 차이가 없음을 알 수 있다. 제안한 방법의 실행 시간 증가는 질의 처리 시 비-중복적인 답 트리들을 추가로 탐색하기 위해 발생한 오버헤드라고 할 수 있다.

VI. 결 론

본 논문에서는 그래프 구조를 갖는 데이터에 대한 키워드 기반 질의에 대해 구조적인 중복성을 줄이면서 연관성이 높은 결과들을 효율적으로 검색하는 방법을 제안하였다. 비-중복적인 결과 트리 구조와 연관도 척도를 정의하고 그래프 내에 포함된 경로 정보들에 대한 효과적인 인덱싱 방법과 이를 활용한 질의 처리 알고리즘을 제시하였다. 제안한 방법은 주어진 질의에 대해 비-중복적이면서 연관도가 가장 큰 서브-트리 집합을 구함으로써 사용자에게 다양하고 효과적인 검색 결과를 제공할 수 있다. 향후 대규모 그래프 데이터를 이용한 성능 분석과 질의 결과에 대한 사용자 만족도 평가, 그리고 질의 결과에 포함된 콘텐트 노드들의 의미적 중복성을 줄이기 위한 후속 연구를 수행할 계획이다.

참 고 문 헌

[1] G. Bhalotia, A. Hulgeri, C. Nakhe, S. Chakrabarti, and S. Sudarshan, “Keyword searching and browsing in databases using BANKS,” Proc. of ICDE, pp.431-440, 2002.

[2] V. Kacholia, S. Pandit, S. Chakrabarti, S. Sudarshan, R. Desai, and H. Karambelkar, “Bidirectional expansion for keyword search on graph databases,” Proc. of the 31st Int. Conf. on VLDB, pp.505-516, 2005.

[3] H. He, H. Wang, J. Yang, and P. S. Yu, “BLINKS: ranked keyword searches on graphs,” ACM SIGMOD Conference, pp.305-316, 2007.

[4] B. Ding, J. X. Yu, S. Wang, L. Qin, X. Zhang, and X. Lin, “Finding top-k min-cost connected trees in databases,” Proc. of ICDE, pp.836-845, 2007.

[5] B. B. Dalvi, M. Kshirsagar, and S. Sudarshan, “Keyword search on external memory data graphs,” Proc. of the VLDB Endowment, Vol.1, No.1, pp.1189-1204, 2008.

[6] K. Golenberg, B. Kimelfeld, and Y. Sagiv, “Keyword proximity search in complex data graphs,” Proc. of ACM SIGMOD Conference, pp.927-940, 2008.

[7] L. Qin, J. X. Yu, L. Chang, and Y. Tao, “Querying communities in relational databases,” Proc. of the 25th ICDE, pp.724-735, 2009.

[8] T. Tran, S. Rudolph, P. Cimiano, and H. Wang, “Top-k exploration of query candidates for efficient keyword search on graph-shaped (RDF) data,” Proc. of the 25th ICDE, pp.405-416, 2009.

[9] M. Kargar and A. An, “Keyword search in graphs: finding r-cliques,” Proc. of the VLDB Endowment, Vol.4, No.10, pp.681-692, 2011.

[10] C. Park and S. Lim, “Efficient processing of keyword queries over graph databases for

finding effective answers," Information Proc. and Mgmt, Vol.51, No.1, pp.42-57, 2015.

[11] J. X. Yu, L. Qin, and L. Chang, "Keyword search in relational databases: a survey," Bulletin of the IEEE CS Technical Committee on Data Engineering, Vol.33, No.1, pp.67-78, 2010.

[12] R. Fagin, A. Lotem, and M. Naor, "Optimal aggregation algorithms for middleware," Journal of Computer and System Sciences, Vol.66, No.4, pp.614-656, 2003.

[13] S. Buttcher, C. Clarke, and G. Cormack, *Information retrieval: implementing and evaluating search engine*, MIT Press, 2010.

저 자 소 개

박 창 섭(Chang-Sup Park) 정회원

- 1995년 2월 : KAIST 전산학과 (공학사)
- 1997년 2월 : KAIST 전자전산학과(공학석사)
- 2002년 2월 : KAIST 전자전산학과(공학박사)
- 2002년 3월 ~ 2005년 2월 : (주)KT 책임연구원
- 2005년 3월 ~ 2009년 2월 : 수원대학교 교수
- 2009년 3월 ~ 현재 : 동덕여자대학교 컴퓨터학과 교수

<관심분야> : 데이터베이스, 정보 검색, 데이터 마이닝