

분산된 준구조적 데이터 검색을 위한 경로 질의 처리 기법

(A Path Query Processing Scheme for Distributed Semi-structured Data Retrieval)

이재형[†] 정연돈^{**} 김덕현^{***} 김명호^{****}
(Jaehyung Lee) (Yon Dohn Chung) (Duk Hyun Kim) (Myoung Ho Kim)

요약 본 논문에서는 분산된 준구조적 데이터에 대한 질의 처리 문제를 다룬다. 분산된 준구조적 데이터는 루트가 있고 간선에 레이블이 있는 그래프 모델로 표현될 수 있으며, 그래프의 노드들은 한 사이트 또는 여러 사이트들에 위치할 수 있다. 분산된 준구조적 데이터의 효율적인 검색을 위해 '질의 단축 및 확산' 방법에 기반을 둔 질의 처리 모델을 제안한다. 이 방법은 사용자 질의가 사이트 내부에서 단축되고 다른 사이트로 분산되는 과정을 통해 데이터를 검색한다. 또한, 제안된 모델에 필요한 알고리즘들을 제시하고 정확성을 증명한다.

Abstract In this paper, we address the problem of query processing on distributed semi-structured data. The distributed semi-structured data can be modeled as a rooted and edge labeled graph, where nodes are located in a single or a number of sites. For efficient retrieval of distributed semi-structured data, we propose a query processing model that is based on the 'query reduction and diffusion' method. In the method, a user query is reduced in a site and distributed to other sites for data retrieval. We also propose a set of algorithms for the proposed model and show their correctness.

1. 서론

준구조적 데이터 (semi-structured data)는 “구조가 전혀 없지 않으면서 또한 특정 구조를 잘 따르지 않는 데이터[1]”로 설명된다. 준구조적 데이터의 예로는 HTML 문서, BIBTEX 문서, 유전자 데이터 등을 들 수 있으며, 이러한 준구조적 데이터들은 구조가 불규칙적이며 변할 수 있다는 공통적인 특징을 갖고 있다.

준구조적 데이터에 대한 연구는 기존의 데이터베이스 관련 기술을 준구조적 데이터에 대한 영역으로 확장하

여 기존의 관계형 데이터베이스나 객체지향형 데이터베이스의 기능을 보완하는 방향을 펴고 있다. 세부 연구 분야로는 데이터 통합, 웹 사이트 관리, 범용의 준구조적 데이터 관리[2], 데이터 모델[3, 4], 질의 언어 디자인[4, 5], 질의 처리[6], 인덱스 기법[7] 등이 있다.

준구조적 데이터를 위해 제안된 대부분의 데이터 모델들은 루트가 있고 간선에 레이블이 있는 그래프를 기반으로 하고 있다[3, 4]. 그래프의 노드들은 각 객체를 의미하고 간선들의 레이블은 문자열이나 정수 또는 이 미지나 사운드 같은 값으로 구성될 수 있다.

여러 가지 언어들이 준구조적 데이터를 위해 제안되었으며 이들은 표현 스타일이나 표현 능력에 있어서 다양하다[4, 5, 8]. 그렇지만 이러한 언어들은 일반적으로 정규 경로 표현식에 기반을 두고 있으며, 간선들의 레이블로 구성되는 경로를 만족하는 노드들을 선언적으로 표현한다.

본 논문에서는 분산된 준구조적 데이터 검색을 위한 질의 처리 모델을 제안한다. 이 모델은 경로 질의 '단축 (reduction)'과 '확산 (diffusion)' 방법을 사용하며, 현

[†] 비 회 원 : 한국과학기술원 전자전산학과
jlee@dbserver.kaist.ac.kr

^{**} 비 회 원 : 한국과학기술원 정보전자연구소 연구원
ydcchung@dbserver.kaist.ac.kr

^{***} 종신회원 : 국방과학연구소 연구원
dhkimn@umtel.co.kr

^{****} 종신회원 : 한국과학기술원 전자전산학과 교수
mhkim@dbserver.kaist.ac.kr

논문접수 : 1999년 11월 17일
심사완료 : 2000년 9월 27일

제의 HTML 문서 또는 XML 문서로 구성된 웹 (Web) 환경에 적용 가능하다. 각 사이트에서는 정규식으로 주어지는 질의가 간선의 레이블을 따라 이동하면서 경로 식이 줄어드는 질의 단축 방법과 외부의 사이트로 향하는 간선에 대하여 그 사이트에 단축된 질의를 전달하는 질의 확산 방법으로 질의가 처리된다.

분산 질의 처리에 대한 기존 연구로는 질의 분해 방법[6]이 있다. 질의 분해 방법에서는 준구조적 데이터가 미리 알려진 고정된 사이트에 나뉘어져 있고 모든 사이트들은 그들의 입력 노드와 출력 노드를 알고 있다고 가정한다. 본 논문에서 제안하는 분산 질의 처리 모델은 이러한 가정 대신에 현재의 웹 환경에 적용 가능한 각 사이트가 자신의 출력 노드만을 알고 있다는 것만을 가정으로 한다.

분산 질의 처리 모델에서는 질의가 시작되었던 사이트가 질의 처리의 종료를 알 수 있는 방법이 필요하다. 본 논문에서는 모든 관련 사이트에서 질의 처리가 끝나는 정상적인 경우와 사용자의 취소 요구에 의해 질의 처리가 종료 경우에 있어서 분산 질의 처리의 종료를 탐지할 수 있는 알고리즘을 제안한다. 그리고, 사이트의 고장 또는 사이트간의 통신에 장애가 발생하여도 각 사이트에서 질의 처리를 종료할 수 있는 방법을 제시한다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 연구 배경으로 준구조적 데이터 모델, 준구조적 질의 언어, 분산 질의 처리에 대하여 알아본다. 3장에서는 본 논문에서 다룬 분산된 준구조적 데이터 검색을 위한 질의 처리 모델을 제안하고, 4장에서는 정상적인 질의 처리 종료와 사용자의 취소 요구에 의한 질의 처리 종료를 탐지하는 알고리즘과 시간 지정 종료 방법을 제안하고 정확성을 증명한다. 마지막으로 5장에서는 결론 및 추후 연구 방향을 제시한다.

2. 연구 배경

2.1 준구조적 데이터 모델 (Semi-structured Data Model)

준구조적 데이터는 다음과 같은 간단한 구조로 표기될 수 있다[3].

```
{name: {first: "Jaehyung", last: "Lee"},
tel: 3570,
email: "jlee@univ.ac.kr"}
```

이러한 표기법은 단순히 데이터를 레이블 (label)과 그것의 값 (value)의 짝 (pair)으로 표현한 것이며, 값은 또 다른 구조를 가질 수도 있다.

또한 준구조적 데이터는 루트가 있고 간선에 레이블

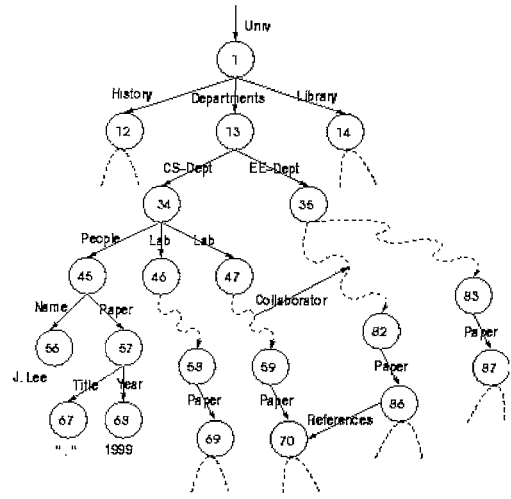


그림 1 준구조적 데이터 예 - 대학교 홈페이지

과 방향성이 있는 그래프 형태의 모델로 표현된다[3]. 그림 1은 대학교 홈페이지의 일부분을 준구조적 데이터의 그래프 형태로 표현한 예이다. 그래프의 노드들은 웹 페이지를 의미하며 간선들은 노드들 간의 하이퍼링크를 나타낸다. 노드 내부에 표시된 숫자는 각 노드의 식별자를 나타낸다.

2.2 준구조적 질의 언어

준구조적 질의 언어들 (UnQL[4], Lorel[5], StruQL[8])의 가장 큰 특징은 그래프의 임의의 부분을 검색한다는 점이다. 이러한 검색을 위해 경로식 (path expression)이라는 개념을 사용한다. 준구조적 데이터 모델 상의 경로식은 간선의 레이블들의 배열 (sequence)이다. 경로식은 준구조적 데이터에 대한 간단한 질의로 생각될 수 있으며, 이 질의의 결과는 루트 노드에서부터 시작하여 주어진 경로식을 만족하는 노드들의 집합이 된다.

준구조적 데이터에 대한 질의 언어로 경로식을 사용할 때 경로상의 모든 레이블을 기술하는 것 대신에 정규식 (regular expression)을 이용하면 질의를 보다 간결하게 표현할 수 있다. 준구조적 데이터의 질의 언어로 사용되는 정규식은 다음과 같은 문법으로 표현된다[9].

$$R ::= P \mid a \mid _ \mid R_1 R_2 \mid R \Rightarrow R \mid R^*$$

여기에서 P 는 레이블에 대한 사용자 정의의 조건문이거나 또는 이러한 조건문들의 부울 (bool) 조합이다. a 는 레이블 상수 (constant)이고, $_$ 는 임의의 레이블을 나타내며, $R_1 R_2$ 는 교대 (alternation), $R_1 \Rightarrow R_2$ 는 연결 (concatenation), 그리고 R^* 는 폐쇄 (closure)를 나타낸

다.

다음의 경로식 **Query1**은 그림 1의 대학교 홈페이지에서 전산과의 모든 논문들을 찾는다.

Query1 : $_ \wedge \Rightarrow CS-Dept \Rightarrow _ * \Rightarrow Paper$

그림 1에서 **Query1**으로 찾아지는 노드들은 {57, 69, 70, 86} 이다. 앞으로 정규식에서 혼동이 없는 한 \Rightarrow 를 생략해서 쓰겠다.

본 논문에서는 정규식 R 을 다음과 같은 질의로 정의한다.

$$Q(DB) = \text{select } t \\ \text{where } R \Rightarrow t \text{ in } DB$$

이때 이러한 질의를 정규 경로 질의 (regular path query) 또는 정규 질의 (regular query) 라고 부르며, 이 질의의 결과는 DB 의 루트에서부터 레이블의 배열이 R 과 일치하는 경로들을 통해 도달할 수 있는 모든 노드들을 의미한다.

2.3 분산 질의 처리

현재 많은 홈페이지들이 여러 사이트에 분산되어 존재하며 이들은 서로 하이퍼링크로 연결되어 있다. 이러한 홈페이지들에 대하여 앞에서 살펴보았던 준구조적 데이터 모델을 적용할 수 있다. 그림 2는 대학교 홈페이지가 세 개의 서로 다른 사이트에 분산되어 저장되어 있는 것을 준구조적 데이터 모델[3]로 표현한 예이다. 그림 2의 노드 1에 대해 **Query1**을 분산 질의 처리하는 경우에, 그 결과는 빗금으로 표시된 노드들이 된다.

[6]은 분산되어 있는 준구조적 데이터에 대한 질의 처리를 위해 질의 분해 (query decomposition) 방법을 제안하였다. 질의 분해 방법은 질의가 모든 사이트에 전

달되고 각 사이트에서 계산이 이루어진 후 그 결과가 종합되어 최종 질의 결과를 얻게되는 방식이다.

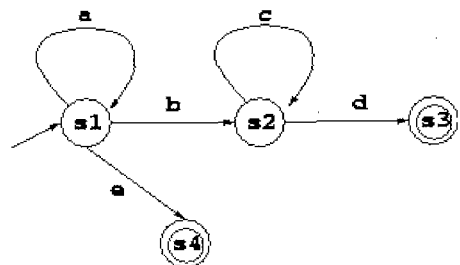
3. 질의 처리 모델

기존의 분산 질의 처리 방법[6]에서는 준구조적 데이터가 미리 알려진 고정된 사이트에 나뉘어져 있고 모든 사이트들은 그들의 입력 노드와 출력 노드를 알고 있다고 가정한다. (여기에서, 입력 노드란 외부 사이트에서 가리키는 간선이 있는 노드를 의미하고, 출력 노드란 외부 사이트에 있는 다른 노드를 가리키는 간선을 갖고 있는 노드를 의미한다.) 그러나, HTML 문서나 XML 문서로 구성된 웹 페이지들을 고려해 볼 때 하이퍼링크를 통해 서로 연결되어 있는 문서들의 위치를 미리 알기는 어렵다. 또한 일반적으로 HTML 문서나 XML 문서의 입력 노드를 알 수 없으므로 이러한 가정들은 현재의 웹 환경과는 맞지 않는다. 따라서 본 논문에서는 각 사이트가 자신의 출력 노드만을 알고 있다는 현실적인 가정 하에 분산 질의 처리를 모델화 한다. 이를 위해 3.1절에서는 준구조적 데이터 검색을 위해 하나의 사이트에서 행해지는 일반적인 질의 처리 모델을 정의하고, 3.2절에서는 분산 환경을 위한 질의 처리 모델을 제안한다.

3.1 질의 단축을 이용한 지역 질의 처리 모델

먼저 경로 질의의 단축에 사용되는 *Reduce* 함수를 정의하자. *Reduce* 함수는 정규식으로 주어지는 경로 질의와 오토마타의 상태 (state) 이동에 사용되는 레이블을 입력으로 한다.

$$R : a^*bc^*d \mid a^*a$$



$$\begin{aligned} Reduce(R,a) &= a^*bc^*d \mid a^*a \\ Reduce(R,b) &= c^*d \\ Reduce(R,c) &= Null \\ Reduce(R,d) &= Null \\ Reduce(R,e) &= \epsilon \end{aligned}$$

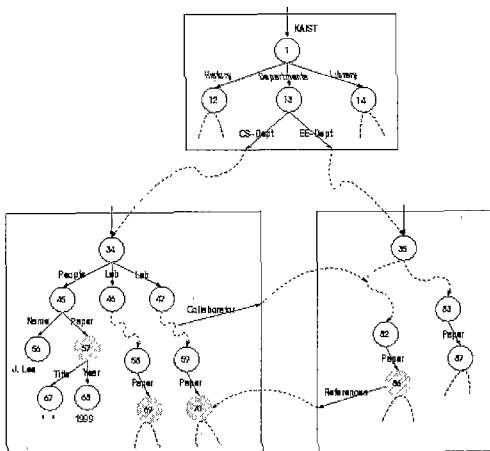


그림 2 분산되어 있는 준구조적 데이터 예

그림 3 경로 질의 단축 예

주어진 경로 질의에 해당하는 오토마타의 시작 상태에서로부터 주어진 레이블을 이용하여 이동이 가능하면, 이동 후에 위치하는 상태를 새로운 시작 상태로 하는 오토마타를 구성하여 반환한다. 그렇지 않고 주어진 레이블을 이용하여 이동이 불가능하면 Null 을 반환한다. 그림 4는 정규식 $R: a^*bc^*d \mid a^*e$ 에 대한 오토마타와 각 레이블에 Reduce 함수를 적용한 예이다.

각 지역 사이트에서 경로 질의를 처리하는 것으로, 알고리즘 1을 사용하는 모델을 가정한다. 알고리즘 1의 LQP (Local Query Processing) 함수는 정규식으로 주어지는 경로 질의와 지역 사이트의 노드 식별자에 대해, 해당 노드에서 출발하는 모든 간선들의 레이블에 Reduce 함수를 적용시킨다. Reduce 함수의 결과가 Null 이 아니면 단축된 질의를 사용하여 자식 노드에 대해 LQP 함수를 재귀적으로 호출한다. 그림 4는 한 사이트에서 경로 질의 a^*bc^*d 가 처리되는 과정을 보여주고 있으며, 질의 처리 결과는 {3, 4}이다. 이와 같이 지역 질의 처리 모델을 사용하면 주어진 경로 질의에 대한 지역 사이트 내의 질의 처리 결과를 얻을 수 있다.

알고리즘 1 지역 질의 처리

```

Visited ← ∅ /* nodes already visited */
Result ← ∅ /* query result */
LQP(R, Root(DB))
function LQP(R, u)
begin
  if start state of R ∈ final states of R then
    Result ← Result ∪ {u}
  end if
  if <R, u> ∈ Visited then return end if
  Visited ← Visited ∪ { <R, u> }
  for all  $u \xrightarrow{a} v$  do
    R2 ← Reduce(R, a)
    if R2 ≠ Null then LQP(R2, v) end if
  end for
end
    
```

정리 1 (알고리즘 1의 정확성) 알고리즘 1이 수행되고 나면 Result는 지역 질의 처리의 결과 (주어진 DB를 그래프로 표현할 때 그 그래프의 루트 노드에서부터 R을 이용하여 도달 가능한 노드들의 집합)가 된다.

[증명]

DB를 그래프로 표현한 것을 $G_1 = (V_1, E_1)$ 이라 하고, R의 오토마타로 구성되는 그래프를 $G_2 = (V_2, E_2)$ 라고 하자. 여기에서 V_1 은 G_1 의 노드의 집합이고 $E_1 =$

$\{(m, l, n) \mid m, n \in V_1, l \text{은 레이블}\}$ 이다. 마찬가지로 V_2 는 G_2 의 노드의 집합이고 $E_2 = \{(m, l, n) \mid m, n \in V_2, l \text{은 레이블}\}$ 이다. 여기에서 두 그래프 G_1 과 G_2 를 이용하여 다음과 같이 $G_1 \times G_2$ 를 정의하자.

$$G_1 \times G_2 = (V, E)$$

$$V = \{ (V_{1i} \times V_{1j}) \mid V_i \in V_1, V_j \in V_2 \}$$

$$E = \{ (V_{1i} \times V_{1j1}, l, V_{1i} \times V_{1j2}) \mid V_{1i} \times V_{1j1}, V_{1i} \times V_{1j2} \in V, (V_{1i}, l, V_{1j}) \in E_1, (V_{1j1}, l, V_{1j2}) \in E_2, l \text{은 레이블} \}$$

그러면 알고리즘 1은 그래프 $G_1 \times G_2$ 상에서 재귀적인 깊이 우선 탐색을 사용하여 도달 가능한 모든 노드들 $V_{1i} \times V_{1j}$ 중에서 V_{1j} 가 G_2 의 종료 상태 (final states)에 속하는 노드들을 찾는 문제와 같아진다. (여기에서 알고리즘 1의 첫 번째 if 문은 재귀 호출의 종료 조건이 되고, 두 번째 if 문은 중복을 방지해 준다. 그리고, for 문은 가능한 모든 노드에 대해 재귀 호출을 수행하게 된다.) 따라서 알고리즘 1을 사용하여 지역 질의 처리의 결과를 얻을 수 있게 된다. □

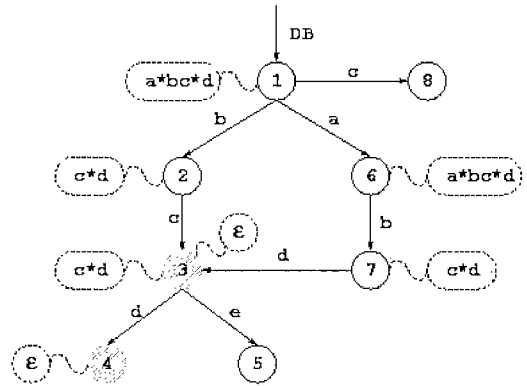


그림 4 지역 질의 처리 예

3.2 분산 질의 처리 모델

분산된 준구조적 데이터에 대한 질의 처리 모델은 알고리즘 2를 기반으로 한다. 알고리즘 2는 분산된 준구조적 데이터와 정규식으로 표현 가능한 경로 질의를 입력으로 하며, 모든 사이트가 자신의 출력 노드만을 알고 있다고 가정한다.

지역 질의 처리와는 달리 분산된 준구조적 데이터 검색에서는 질의가 여러 사이트들에 전달되어야 하며, 이러한 질의 전달은 질의 확산 (diffusion)으로 이루어진다. 여기에서 확산은 각 사이트에서 질의 단축에 따라 주어진 경로 질의를 처리하다가 외부 사이트의 노드로

향하는 간선을 만났을 시에 이 간선의 레이블에 대한 *Reduce* 함수가 *Null* 이 아니면 그 간선을 따라 질의가 전달되는 것을 의미한다.

알고리즘 2 분산 질의 처리

s : 질의가 시작된 사이트
O_p : 사이트 *p*에 있는 노드의 식별자
R : 정규 경로식
 {질의 메시지 <*s*, *O_p*, *R*>이 사이트 *p*에 도착했을때
 receive <*s*, *O_p*, *R*>
 evaluate <*s*, *O_p*, *R*>
 {외부 사이트로 향하는 <*s*, *O_r*, *R'*>이 계산되었을때
 send <*s*, *O_r*, *R'*> to site *r*
 {질의 처리 결과인 *result_p*가 계산되었을때
 send <*result_p*> to site *s*

한 사이트가 질의 메시지를 받으면 사이트 내에서 지역 질의 처리 모델에 따른 경로 질의 처리가 시작된다. 이때 *Reduce* 함수를 만족하는 외부 사이트로 향하는 링크가 발견되면 그 사이트로 질의 메시지를 보낸다. 이때 질의 단축 결과에 따라 단축된 정규식을 새로운 경로 질의로 하여 질의 메시지와 함께 전달한다. 한편, 주어진 경로 질의에 대한 모든 지역 사이트 내의 처리가 끝나면 그 결과를 경로 질의가 시작되었던 사이트로 전달한다. 분산 질의 처리 모델은 경로 질의의 단축과 확산을 기반으로 하고 있으며, 이러한 분산 질의 처리 모델을 이용한 질의 처리 예가 그림 5에 있다.

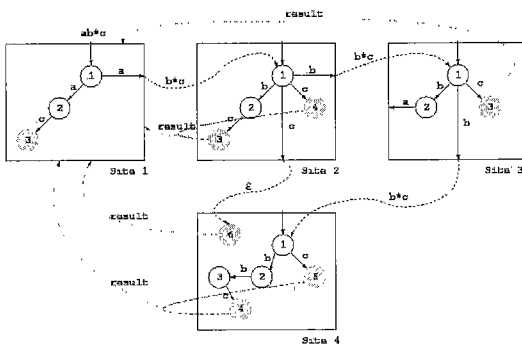


그림 5 분산 질의 처리 예

그림 5에서는 질의 *ab*c*가 사이트1에 전달되어 분산 처리되는 과정을 보여주고 있다. 사이트1에 전달된 질의는 질의 단축과 확산을 통하여 사이트2로 전달되고, 사이트2에서 사이트3과 사이트4로, 사이트3에서 사이트4로 질의가 전달된다. 사이트2, 사이트3, 사이트4에서는 각각

자신의 질의 처리 결과를 사이트1로 전달한다. 이러한 과정을 통하여 사이트1은 주어진 질의에 대한 처리 결과를 얻게 된다.

4. 질의 처리 종료 탐지

본 장에서는 분산 질의 처리 모델에서 질의의 처리가 종료되는 것을 질의 처리가 시작된 사이트에서 알 수 있는 알고리즘을 제안한다. 또한 사용자의 요구에 의해 질의 처리가 취소되는 경우에 있어서도 이러한 취소 요구가 모든 사이트에 올바르게 전달되어 모든 작업이 종료되었음을 질의 처리가 시작된 사이트에서 알 수 있는 확장된 알고리즘도 제안한다. 그리고, 사이트나 통신에 고장이 발생한 경우에도 각 사이트에서 질의 처리를 종료할 수 있는 시간 지정 종료 (time out) 방법을 제시한다. 질의 처리의 정상적인 종료 및 사용자의 취소 요구에 의한 질의 처리 종료를 탐지하기 위해 사용되는 변수들은 표 1과 같다.

표 1 필요한 변수들

변수명	설 명
$1, \dots, N$	사이트 (질의 시작 사이트는 <i>s</i>)
$state_p$	각 사이트의 상태 (active, passive) ($state_s=active, state_p=passive$ for $p \neq s$)
$parent_p$	사이트 <i>p</i> 에 질의를 보낸 사이트 ($parent_s=s, parent_p=\perp$ for $p \neq s$) ' \perp '는 변수에 아무 값도 할당되지 않음을 나타냄.
log_p	사이트 <i>p</i> 에 주어지는 질의 정보들에 대한 기록
$result_p$	주어진 질의에 대한 사이트 <i>p</i> 의 결과
$n(query_p)$	사이트 <i>p</i> 에서 받아야할 <query-ack> 메시지의 갯수
$n(result_p)$	사이트 <i>p</i> 에서 받아야할 <result-ack> 메시지의 갯수
$n(query_p)$	사이트 <i>p</i> 가 사이트 <i>r</i> 에 보낸 질의 메시지에 대해 받아야할 <query-ack> 메시지의 갯수

4.1 정상 종료

질의 처리가 처음 시작된 사이트에서 모든 관련된 사이트의 질의 처리가 끝났음을 알 수 있도록 하기 위하여 각 사이트 *p*에서 수행해야 할 일은 다음과 같이 다섯 가지 경우로 나뉜다.

- 질의 메시지 <*s*, *q*, *O_p*, *R*>이 사이트 *p*에 도착했을 때¹⁾

```

receive <s, q, Op, R>
if <Op, R> ∈ logp then
    send <query-ack> to q
    return
else
    logp ← logp ∪ { <Op, R> }
end if
if parentp = ⊥ then
    parentp ← q
else
    send <query-ack> to q
end if
statep ← active
evaluate <s, Op, R>
• <query-ack>이 사이트 p에 도착했을 때
n(queryp)--
if n(queryp)=0 and statep=passive then
    if parentp=p then
        Termination Detect!
    else
        send <query-ack> to parentp
        parentp ← ⊥
    end if
end if
• 질의 처리 결과인 resultp가 계산되었을 때
send <resultp> to site s
n(resultp)++
• <result-ack>이 사이트 p에 도착했을 때
receive <result-ack>
n(resultp)--
if n(resultp)=0 then
    statep ← passive
end if
if n(queryp)=0 and statep=passive then
    if parentp=p then
        Termination Detect!
    else
        send <query-ack> to parentp
        parentp ← ⊥
    end if
end if
• 외부 사이트로 향하는 <s, p, Or, R'>이 계산되었을 때
send <s, p, Or, R'> to site r
n(queryp)++
    
```

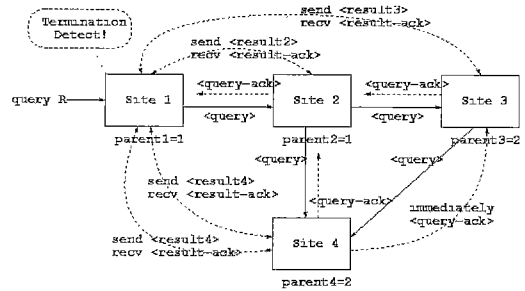


그림 6 정상 종료 예

그림 6은 경로 질의가 4개의 사이트에서 분산 처리되는 것과 이러한 분산 처리의 종료를 사이트1이 탐지하는 예를 보여준다. 사이트1에서 사이트2로, 사이트2에서 사이트3과 사이트4로, 사이트3에서 사이트4로 질의 전달이 이루어지고 각 질의에 대한 지역 사이트의 처리 결과가 사이트1로 전달된다. 사이트3과 사이트4는 사이트2로 질의 처리 종료를 알리고 사이트2는 사이트1로 질의 처리 종료를 알린다. 이러한 과정을 통해 사이트1은 분산 질의 처리의 종료를 탐지한다.

사이트3에서 사이트4로 보낸 질의에 대해서 사이트4는 이미 parent 변수가 사이트2로 설정되어 있으므로 사이트3에 곧바로 질의 처리 종료를 알린다. 이것은 사이트4에 주어진 모든 질의에 대한 실제 처리 종료를 사이트2에 알리도록 함으로써 질의 처리 종료 메시지가 전달되는 경로를 트리 형태로 유지하는 역할을 한다.

질의 순환 (cycle) 방지

사이트 A의 노드 a에 R이라는 질의가 주어진 경우를 가정하자. 만약 이를 처리하는 과정 중에 사이트 B의 노드 b로 같은 질의의 R이 확산되어 전달되고 이것이 다시 사이트 A의 노드 a로 전달된다면 이에 의해 질의 순환이 발생할 수 있다. 이러한 질의 순환을 방지하기 위해서는 이미 처리했거나 처리중인 같은 질의 메시지가 도착한 경우에 이를 다시 처리하지 않아야 한다.

질의 순환의 방지를 위해서 각 사이트는 전달되는 질의 메시지를 저장하는 log 변수를 사용한다. log 변수의 관리란 질의가 시작된 사이트 s에서 질의 결과인 result를 보낸 사이트들에 대한 정보를 유지하고, 사이트 s에서 질의 처리의 종료를 탐지한 후에 result를 보낸 사이트들에게 모든 질의 처리의 종료를 알리면 각 사이트는 log 변수의 정보를 초기화하는 방식으로 이루어진다.

4.2 사용자 취소 요구에 의한 종료

주어진 질의가 여러 사이트에서 분산 처리 중에 있을 때 사용자는 더 이상의 질의 처리를 원하지 않을 수 있

1. <s, q, O_p, R>
 - s : 질의가 시작된 사이트,
 - q : 질의를 전달한 부모 사이트,
 - O_p : 사이트 p에 있는 노드의 식별자,
 - R : 경로 질의

다. 이러한 사용자 요구에 의한 분산 질의 처리의 종료 를 지원해 주기 위해 표 1에서 설명한 $n(query_p)$ 변수 가 사용된다.

정상 종료를 탐지하는 알고리즘에 사이트 r 로 향하는 질의가 계산되었을 때 $n(query_p)$ 변수를 하나 증가시키는 것과 <query-ack>가 사이트 r 로부터 오는 경우에 $n(query_p)$ 변수를 하나 감소시키는 것을 추가한다. 그리고 다음과 같이 취소 (abort) 메시지에 대한 고려를 추가하면 사용자 요구에 의한 분산 질의 처리의 종료를 탐지할 수 있다.

```

    • <abort> 메시지가 사이트  $p$ 에 도착했을 때
    if  $parent_p = \perp$  then return end if
    stop its processing
     $state_p \leftarrow$  passive
    if  $n(query_p) = 0$  then
        if  $n(result_p) = 0$  then
            if  $parent_p = p$  then
                Termination Detect!
            else
                send <query-ack> to  $parent_p$ 
                 $parent_p \leftarrow \perp$ 
            end if
        end if
    end if
    else
        send <abort> to all sites  $r$  where  $n(query_r) \neq 0$ 
         $n(query_r) \leftarrow 0$  for all  $r$ 
    end if

```

4.3 시간 지정 종료 (time out)

지금까지 살펴본 경로 질의 처리는 각 사이트의 고장 또는 사이트간의 통신 (communication)에 장애가 없는 경우를 가정하고 있다. 만약 사이트 또는 사이트간의 통신에 장애가 발생하면 부모 사이트에 <query-ack>를 전달할 수 없는 경우에도 자식 사이트들에서는 계속 질의가 확산될 수 있다. 이러한 것을 방지하기 위해 다음과 같은 시간 지정 종료 (time out) 정책을 쓰기로 한다.

사이트 S 가 질의를 전달하고 사이트 R 이 질의를 전달받는 경우를 가정하자. 사이트 S 는 질의 전달 시에 '종료 시간' (예상 질의 처리 시간 + 메시지 왕복 시간)을 함께 전달한다. 사이트 R 은 '종료 시간 - 메시지 전달 시간' 내에 질의 처리가 끝나면 <query-ack>를 보낸다. 그렇지 않으면 종료 시간 재설정 (reset)을 요구하는 메시지를 사이트 S 에 보내고, 사이트 S 는 새로운

종료 시간을 설정하여 사이트 R 에 보낸다. 여기에서 메시지 왕복/전달 시간은 Cristian[10]의 방법을 따르기로 한다. 이와 같은 방법을 기반으로 다음과 같은 경우에도 각 사이트에서의 질의 처리를 종료할 수 있다.

- 사이트 S 가 사이트 R 로부터 종료 시간 내에 <query-ack> 또는 종료 시간 재 설정 요구 메시지를 받지 못한 경우
 - (원인) 사이트 R 고장 또는 사이트 S 와 사이트 R 간의 통신 고장
 - (종료 방법) 사이트 R 에 보낸 질의의 처리가 종료된 것으로 간주
- 사이트 R 이 사이트 S 에 종료 시간 재 설정 요구 메시지를 보냈으나 이에 대한 응답을 받지 못한 경우
 - (원인) 사이트 S 고장 또는 사이트 S 와 사이트 R 간의 통신 고장
 - (종료 방법) 사이트 S 로부터 전달받은 질의 q 에 대한 사이트 R 내부의 지역 질의 처리 중단 및 q 로 인해 다른 사이트로 확산된 질의의 종료 시간 재 설정 요구 거절

이러한 시간 지정 종료 방법은 앞 절에서 제안한 분산 질의 처리의 종료 탐지 알고리즘과 함께 사용될 수 있으며, 사이트의 고장 또는 사이트간의 통신에 장애가 발생하여도 각 사이트에서의 질의 처리 종료를 가능하게 한다.

4.4 알고리즘의 정확성

본 절에서는 분산 질의 처리가 정상적으로 종료되는 것을 탐지하는 '정상 종료 알고리즘'과 사용자 취소 요구에 의해 분산 질의 처리가 종료되는 것을 탐지하는 '사용자 취소 알고리즘'의 정확성을 증명한다. 이러한 두 알고리즘의 경우에 있어서, 분산 질의 처리가 종료된 후에 질의가 시작된 사이트가 수동 (passive) 상태로 되돌아온다는 것과 질의가 시작된 사이트가 수동 상태로 되돌아 온 경우에 실제 분산 처리가 종료되었음을 증명한다. 증명을 위하여 먼저 다음과 같이 분산 처리의 종료에 대한 정의가 필요하다.

정의 1 (분산 처리의 종료) 분산 처리에 관여된 모든 사이트가 수동 (passive) 상태에 있고 질의 (query) / 질의응답 (query-ack) / 결과 (result) / 결과응답 (result-ack) 등의 메시지를 보냈는데 받지 않은 것이 없을 때 경로 질의에 대한 분산 처리가 종료되었다고 한다.

사용자 취소 알고리즘의 경우에 네트워크에 <abort> 메시지가 남아 있을 수 있다. 그러나, 이 메시지는 더 이상 사이트들을 능동 (active) 상태로 만들지 않는다.

정리 2 경로 질의에 대한 분산 처리가 종료되고 한 정된 몇 스텝이 지난 후에, 질의가 시작된 사이트는 수동 상태로 돌아온다. 또한, 질의가 시작되었던 사이트가 수동 상태로 돌아왔을 때, 경로 질의에 대한 모든 분산 처리는 종료된다.

[증명]

i) 정상 종료 알고리즘

일반적인 확산 계산 (diffusion computing)에 대한 종료 탐지 방법이[11]에서 제시되었으며 이를 변형 (transform)하여 증명한다. 정상 종료 알고리즘에 사용되는 메시지들은 <query>, <query-ack>, <result>, <result-ack> 이며, 하나의 <query>가 사이트에 도착했을 때 가능한 메시지의 발생 순서는 다음과 같이 표현될 수 있다.

```
receive <query>→ (A)
(send <query>, receive (<query-ack>)*
send <result>, receive <result-ack>
(send <query>, receive (<query-ack>)*
send <query-ack>→ (B)
```

우선, 각 사이트의 수동 (passive) 상태를 각 노드의 중립 (neutral[11]) 상태로 생각하고, 질의가 시작된 사이트를 환경 (environment[11])으로 생각하자. 그리고, (A)를 메시지 (message[11])로, (B)를 시그널 (signal[11])로 생각하면 모든 조건이[11]과 같아진다. 이에 대한 자세한 증명은[11]을 참조한다.

ii) 사용자 취소 알고리즘

정상 종료 알고리즘과 사용자 취소 알고리즘의 차이점은 <abort> 메시지의 지원 여부이다. <abort> 메시지는 작업중인 사이트들에 임의의 순간에 전달될 수 있다. 이때 <abort> 메시지를 받은 사이트들은 우선 질의 처리 작업을 중단하고, <query>를 보냈지만 이에 대한 <query-ack>를 보내지 않은 자식 사이트들에게 <abort> 메시지를 보낸다. 부모 사이트가 자식 사이트에 <abort> 메시지를 보내고 나서 이에 대한 <query-ack>를 받을 때 이것이 <abort>에 대한 <query-ack>인지 아니면 실제 작업을 끝마친 것에 대한 <query-ack>인지에 대해 구분하지 않는다. 따라서 자식 사이트에 <abort>가 도착했는지에 대한 여부를 확인할 수 없다. 결국 질의가 시작되었던 사이트가 수동 (passive) 상태가 되어도 네트워크 상에는 <abort> 메시지가 남아 있을 수 있으며, <query-ack>를 이미 보낸 사이트가 <abort> 메시지를 받을 수 있다. 그렇지만,

각 사이트는 *parent* 변수가 1 이므로 <abort> 메시지를 받아도 더 이상의 작업을 하지 않는다. 따라서, 질의가 시작되었던 사이트가 수동 상태인 경우에 정상 종료의 경우와 마찬가지로 질의와 관련된 모든 사이트는 이미 수동 상태이다. □

5. 결론

본 논문에서는 준구조적 데이터에 대한 데이터 모델과 질의 언어, 그리고, 분산 질의 처리 방법을 살펴보았으며, 분산된 준구조적 데이터 검색을 위해 질의 단축 및 확산 방법에 기반을 둔 질의 처리 모델을 정의하였다. 지역 질의 처리 모델에서는 질의 단축 방법을 사용하였고 분산 질의 처리 모델에서는 질의 단축과 함께 외부 사이트에 대하여 단축된 질의가 전달되는 질의 확산 방법이 사용되었다. 제안된 질의 처리 모델은 기존의 질의 분해 방법[6]과는 달리 관련된 사이트들을 미리 알 필요가 없으며, 각 사이트가 자신의 출력노드만을 알고 있다는 현실적인 가정을 기반으로 하고 있다.

또한 본 논문에서는 질의 처리가 정상적으로 종료되거나 또는 사용자의 요구에 의해 질의 처리가 취소되는 경우에 있어서 분산 처리의 종료를 탐지할 수 있는 알고리즘을 제안하였고 이에 대한 정확성을 증명하였다. 그리고, 사이트의 고장 또는 사이트간의 통신에 장애가 발생하여도 질의 처리를 종료할 수 있는 시간 지정 종료 방법을 제시하였다.

향후 연구 과제로, 사용자의 다양한 요구 처리를 위해 지원 가능한 질의 언어의 영역을 넓히는 것과 각 지역 사이트 내에서의 질의 처리를 빠르게 하는 방법에 대한 연구가 필요하다.

참고 문헌

- [1] Serge Abiteboul, "Querying semi-structured data," *International Conference on Database Theory*, 1997.
- [2] J. McHugh, S. Abiteboul, R. Goldman, D. Quass, and J. Widom, "Lore: A database management system for semistructured data," *SIGMOD Record*, 26(3), September 1997.
- [3] Y. Papanikolaou, H. Garcia-Molina, and J. Widom, "Object exchange across heterogeneous information sources," *International Conference on Data Engineering*, March 1995.
- [4] Peter Buneman, Susan Davidson, Gerd Hillebrand, and Dan Suciu, "A query language and optimization techniques for unstructured data," *Proceedings of ACM SIGMOD Conference*, 1996.

- [5] S. Abiteboul, D. Quass, J. McHugh, J. Widom, and J. Wiener, "The Lorel query language for semistructured data," *International Journal on Digital Libraries*, 1(1), April 1997.
- [6] Dan Suciu, "Query Decomposition and View Maintenance for Query Languages for Unstructured Data," *Proceedings of Very Large Databases Conference*, 1996.
- [7] Tova Milo and Dan Suciu, "Index structures for path expressions," *International Conference on Database Theory*, 1999.
- [8] Mary Fernandez, Dana Florescu, Alon Levy, and Dan Suciu, "A Query Language for a Web-Site Management System," *SIGMOD Record*, 26(3), September 1997
- [9] Dan Suciu, "Distributed Query Evaluation on Semistructured Data," *Unpublished*, 1997.
- [10] George Coulouris, Jean Dollimore, and Tim Kindberg, *Distributed Systems*, Addison Wesley, 1998.
- [11] Edsger W. Dijkstra and C.S. Scholten, "Termination Detection For Diffusing Computations," *Information Processing Letters*, 11, 1980.



김 명 호

1982년 서울대학교 컴퓨터 공학과 학사.
 1984년 서울대학교 컴퓨터 공학과 석사.
 1989년 MICHIGAN 주립대 전산학과 박사.
 1989년 MICHIGAN 주립대 연구원.
 1989년 ~ 1993년 한국과학기술원 조교수.
 1993년 ~ 1999년 한국과학기술원 부교수.
 1999년 ~ 현재 한국과학기술원 교수.
 1993년 개방형 컴퓨터 통신 연구회(OSIA) 분산트랜잭션처리 분과위원(TG-TP)의장.
 1993년 ~ 1994년 한국통신기술협회(TTA) 분산 트랜잭션처리 실무위원회 의장.
 관심분야는 분산데이터베이스, 분산트랜잭션, 멀티미디어 데이터베이스



이 재 형

1995년 2월 한국과학기술원 전산학과 학사.
 1995년 3월 ~ 1998년 5월 LG소프트 연구원.
 2000년 2월 한국과학기술원 전산학과 석사.
 2000년 4월 ~ 현재 행정자치부 전산사무관시보.
 관심분야는 database systems, semi-structured data.



정 언 돈

1994년 2월 고려대학교 전산학과 학사.
 1996년 2월 한국과학기술원 전산학과 석사.
 2000년 8월 한국과학기술원 전자전산학과 전산학전공 박사.
 2000년 9월 ~ 현재 한국과학기술원 정보전자연구소 Post-Doc. 연구원.
 관심분야는 이동 컴퓨팅 시스템, OLAP, Data Warehouse, Semistructured Databases, 분산시스템 등



김 덕 현

1976년 서울대학교 산업공학과 학사.
 1988년 한국과학기술원 경영과학과 석사.
 1993년 한국과학기술원 경영과학과 박사.
 1976년 ~ 현재 국방과학연구소.
 관심분야는 기업 통합 모델링, 정보시스템 통합, 분산 객체 관리 등.