RSM : Mediator 시스템 한울에서의 사용자 기반의 검색결과 관리자

박철현, 서울대학교
이상구, 서울대학교

RSM : User-based Result Set Manager of Mediator System Hanul

Park Chol-Hyun, Seoul National University
Sang-goo Lee, Seoul National University

요약

mediator 시스템을 통한 질의는 일반적으로 대량의 검색 결과를 제공하는데, 이 검색 결과 내에서 사용자가 원하는 정보를 효과적으로 찾을 수 있도록 하는 인터페이스가 필요하다. 질의한 결과를 지역적으로 저장하고 관리하는 검색결과 관리자를 통해 기존의 검색 결과를 보존하면서 검색 결과에 대한 각종 연산을 가능하게 하도록 하는 결과 재배열에 필요한 기능을 정리하고, 이를 제공하기 위한 모델을 제안한다. 사용자의 질의 히스토리는 계층구조로 표현되며, 검색결과 관리자는 각 질의에 대한 검색 결과 집합에 대한 질의를 제공하고 물리적 저장을 관리한다.

1 서론

사용자가 필요한 정보를 찾는 형태는 크게1)일반 도서관에서 서지 번호나 저자 이름으로 책을 찾는 경우, 2)웹에서 원하는 정보를 찾을 때 까지 링크를 따라 이동하며 검색하는 경우, 3) 넓은 의미의 주제에 관련된 검색에서 시작하여 차츰 범위를 한정시켜 나가면서 원하는 특정 문서를 찾는 정보 탐색(exploration), 세 가지로 분류할 수 있다[1]. 근래의 웹은 한 가지 주제에 관련된 정보원(information source)이 대폭 늘어나면서 정확히 사용자의 세부 요구사항에 맞는 문서를 찾는 일이 어려워졌다. 이런 상황에서 정보 검색에 대한 사용자의 요구사항은 기존의 방식과는 달리 3)의 필요성이 절실해지고 있으며, 이 논문에서 대상으로 생각하는 정보 검색도 3)의 경우에 해당한다. 이런 환경에서는 대체로 사용자의 정보 필요성이 일정하지 않으며 사용자의 관심사가 기존에 검색한 결과에 영향을 받아 변화하는 특성을 보인다[1]. 이 논문에서는 사용자가 넓은 추상적인 범위의 주제에서 구체적인 정보를 찾아가는 과정에서 필요로 하는 연산과 그 연산을 지원하기 위한 모델을 제시하고자 한다. 이런 환경에서의 정보 검색 과정은 임밀한 의미에서는 한 번 질의한 결과에 대한 재배열을 뜻한다. 사용자는 이미 질의한 결과에 대해서 필요한 기준으로 결과를 분류하는 작업이 수반되는데, 그 분류의 기준 제공과 분류의 실행은 전적으로 사용자의 필요에 의해 결정된다. 이 논문에서 제시할 모델은 사용자 기반의 결과 재배열을 지원하기 위한 것이며, 이를 위한 저장과 제질의 기능을 제공한다

2 결과집합(Result Set)

사용자가 질의한 결과에 대한 재배열을 위한 연산 대상을 결과집합(Result Set)이라고 정의하는데, 이를 관리하는 것이 결과집합 관리자(RSM, Result Set Manager)의 역할이다. 결과 집합을 사용하여 정보 탐색을 행하는 과정으로 다음과 같은 대략적인 시나리오가 가능하다. 사용자는 원하는 주제를 바탕으로 시험 삼아 몇 가지 질의를 할 수 있다. 기본적으로 mediator 환경을 배경으로 하고 있으므로 그 결과는 정보원에 따라 서로 다른 기준과 서로 다른 관련도에 의해 검색된 결과이며 양적으로도 방대하다. 이 방대한 결과 집합 속에서 요구에 맞는 적확한 정보를 찾는 일은 어려우므로, 그에 대한 여러 가지 재질의를 통해서 원하는 정보의 집합으로 범위를 좁혀 나간다. 처음의 주제에 관련된 질의의 결과와 그에 대한 재질의 결과는 모두 RSM에서 처리하게 된다

원격의 데이타를 지역(local)에 저장하여 관리한다는 의미에서 캐쉬의 역할과 유사하지만, RSM은 캐쉬와는 다른 목적으

로 존재한다 한 번 질의한 결과에 대해 재질의를 통한 결과 재배열이 목적이므로, 캐쉬를 안정적으로 유지하기 위해서 필요한 여러 가지 부가 작업이 따로 필요하지 않다 RSM은 사용자가 접속해서 작업하는 동안은 논문에서 제안하는 모델에 근거하여 결과를 관리하면서 사용자 주도의 결과 재배열을 효과적으로 지원하고, 사용자의 작업이 끝나면 그 동안의 결과(Result Set)를 모두 버리게 된다

3 관련연구

현재 웹에서 사용되고 있는 검색엔진의 검색된 결과는 양이 방대한 경우가 많으므로 검색되어 온 결과에 대한 관리를 목적으로 하는 인터페이스에 관한 연구가 다양하다. [2]는 aggregation 연산을 위주로 하는 새로운 질의 인터페이스를 이용하여 화면에 출력되는 데이타의 양을 동적으로 조절할 수 있다. [3]은 검색된 대량의 문서를 정해진 개수의 슬롯에 배치하는데, 의미적으로 가까운 문서끼리 같은 클러스터를 이루게 된다. 그 중 관심 있는 슬롯 한 두 개를 대상으로 다시 정해진 개수 만큼 클러스터링을 하는 작업을 반복하여 원하는 문서를 찾아가도록 하는 시스템이다. 검색된 결과에 대한 의미 있는 재배열이라는 입장에서 보면, [2]는 사용자의 의도에 따라 분류가 되고 [3]은 알고리즘에 의해 분류되는 형태이다 한편으로, 정해진 분류구조를 가지고 결과를 분류하는 시스템도 있는데 , [4]나 [5]가 이 경우이다. 특히 [6]은 정보 탐색을 목적으로 개발한 사용자 중심의 결과 관리 인터페이스로서, 정보 탐색을 제공하기 위한 인터페이스가 지녀야 할 연산을 정의한다.

서로 방식은 달라도 이 시스템들은 공통적으로 사용자가 한번에 다루기 힘들 정도의 대량의 검색 결과를 대상으로 사용자의 정확한 요구에 맞는 결과를 찾아내는 일을 보조하는 것이 목적이다. 이 논문에서 제안하는 RSM도 같은 목적으로 제안된 것이고, 추가로 폭 넓은 정보 탐색 작업을 이루기 위해서 이전 질의 결과를 보존하여 필요할 때 사용할 수 있도록 한다. 이것은 사용자가 접속기간 동안 사용했던 질의식과 그에 대한 검색결과를 유지함으로써 가능한데, 사용자의 정보 관심사의 변화에 유동적으로 대처할 수 있다는데 이점이 있다. 한 가지 검색 결과에 대한 추가 질의만을 지원하는 시스템과는 달리, RSM은 새로운 검색결과를 지역적으로 저장한 후에도 이전의 재배열된 결과를 보존하여 사용자가 가능한 많은 검색 결과를 대상으로 재배열 연산을 하도록 한다

4. 결과 재배열 연산

RSM은 사용자의 질의 하나에 대해서 해당 결과를 보여주는데, 윈도우 하나에 표시되는 결과 집합을 collection 이라 정의한다. collection 은 결과 집합을 나타내므로 사용자의 결과 재배열과 재질의는 이 collection에 대한 연산의 형태로 나타난다 사용자가 취할 수 있는 연산은 그 연산의 결과가 새로운 collection을 생성하는지 여부에 따라 크게 내수연산 (inner collection operation)과 대외연산 (inter collection operation)으로 구분한다

내수 연산은 collection의 결과 크기의 관점에서 크기를 제한하는 연산과 크기를 증가시키는 연산으로 나눌 수 있다. 일반적으로 collection 하나에 대한 재질의는 selection, projection을 포함하므로 질의 결과의 크기를 줄이는 역할을 하게 된다 중복 제거 (duplication elimination)는 collection 간의 병합 후의 결과에 대해서 사용된다 한편, 절대적인 크기는 줄지 않지만 결과를 attribute별로 정렬하는 연산도 포함된다. 그리고, 정렬과는 별도로 focus 라는 연산을 정의하였다. 일반적으로 동일한 값을 많이 갖게 되는 attribute 에 대한 정열을 뜻하는 연산인데, 화면에 보여주는 형식이 정열과는 다르다. 저자나 제목 같은 경우에는 동일한 값이 많지 않아서 일반적인 정열이 필요하지만, 정보원(information source) 의 경우처럼 여러 결과가 동일한 값을 갖는 경우에는 단순한 정렬을 통해 결과를 보여주는 것이 효과적이지 못할 것이기 때문이다

한편, 결과 collection의 크기가 커지는 경우는 지각 결과(late result)에 대한 처리를 위해 필요하다. 네트워크 부하나 정보원 시스템의 다운 등의 이유로 특정 정보원에서 결과가 늦게 도착하거나 결과가 나오지 않을 경우 사용자가 체감하는 속도는 참아내기 힘들 정도일 것이기 때문이다. 따라서, 일정한 시간 내에 도달한 결과는 미리 사용자 인터페이스의 collection 에 보여주고 늦는 결과는 사용자의 요청에 따라 나중에 추가시킬 수 있게 한다.

대외 연산은 collection 과 collection 사이의 연산이다. 사용자의 입장에서는 서로 다른 두 collection에 있는 결과를 같은 대상으로 병합하여 질의를 하는 경우가 발생할 수 있는데, 이 연산은 사용자가 원하는 구체적인 결과를 찾아가는데 필요한 연산이다.

mediator 환경에서 사용되는 검색결과 관리자는 질의를 기원질의(original-query)와 재질의(sub-query)로 구분한다. 직접 원격 정보원으로부터 검색하는 것이 기원 질의이고, 지역적으로 저장된 결과에 대한 질의가 재질의이다. 이런 두 가지 질의를 통한 검색 결과는 계층구조를 이루게 되는데, 계층구조는 사용자가 질의했던 내용에 관한 구조를 나타내며 의미상으로는 각 collection이 계층구조의 노드를 형성한다. 기원질의는 계층 구조의 가장 상위를 구성하고, 사용자는 위에서 정의한 연산을 필요에 따라 적용시켜 기원질의에 대한 재질의를 함으로써 기존 검색 결과에 대한 재배열을 할 수 있다.

5 모델

위에서 설명한 결과 재배열에 관련하여 실제 결과를 관리하기 위해서 고안한 모델은 사이클 없는 방향성 그래프 구조이다.

```
RSM Graph = ( N, E )
N = { (RS, RA, State) }
E = { <Ni, Nj> }
State = ( Full-condition, etc)
<Ni, Nj> = (FA, C)
```

RSM 그래프는 노드(N)과 간선(E)로 구성된다. 노드의 구조는 질의 결과를 뜻하는 결과집합(RS)과 결과 집합을 구성

하는 속성(attribute, RA), 그 외 결과에 대한 상태 정보를 나타내기 위한 state의 쌍으로 구성된다. state는 노드의 결과가 어떤 질의식에 의해 구성되는지를 나타내는 Full-condition과 node의 중복제거 상태 등을 표시하는 기타 정보(etc)로 이루어져 있다. 간선은 노드 사이의 관계를 나타낸다 i-노드(Ni)에 포함된 결과에 대해서 질의식 C로 질의를 하여 j-노드(Nj)의 결과가 생성되었다면, 두 노드 Ni, Nj 사이에 방향이 존재하는 간선 <Ni, Nj> 가 생성된다 간선은 질의를 뜻하므로, 질의 결과로 보여줄 attribute의 집합 (FA)와 constraint 조건 (C, WHERE-Clause)으로 구성된다 간선을 구성하는 condition은 두 노드간의 관계를 뜻한다는 점에서 full-condition과는 다르다 Ni의 full-condition이 FCi이고 연산 <Ni, Nj> 즉, (RA, C)를 통해서 Nj 라는 결과 집합이 만들어 졌다면, Nj의 full-condition FCj는 FCi ∧ C 가 된다

RSM 그래프에는 특수한 노드로서 초기 노드(init node)가 존재하는데, 개념상으로 실세계에 존재하는 모든 정보를 나타내며 사용자의 질의 결과로 생긴 노드는 모두 초기 노드의 하위 노드에 해당한다.

RSM 모델은 결과 재배열에서 필요로 하는 연산[6]을 모두 지원해야 한다. 노드 N_{parent}에 대해 재질의를 하여 노드 N_{child}를 생성하는 경우를 생각해 보자

$N_{parent} = (RS_p, RA_p, (FC_p, etc_p))$
$<N_{parent}, N_{child}> = (FA, C)$
$N_{child} = (RS_c, RA_c, (FC_c, etc_c))$
$RS_c \subseteq RS_p, FC_c = FC_p \wedge C, RA_c = FA \subseteq RA_p$

selection, projection 연산은 두 노드 (연산 대상 노드, 연산 결과 노드) 사이의 간선으로 간단하게 표현할 수 있다. projection은 C를 무시하고 RA의 단순한 포함관계로 나타낼 수 있으며, selection은 C를 통해서 표현 할 수 있기 때문이다 그러나, 병합 연산은 그리 간단한 문제가 아니다 계층구조는 트리 구조로도 충분히 표현할 수 있지만, 병합 연산을 지원함으로써 한 노드가 여러 개의 부모 노드를 갖게 함으로써 복잡한 그래프 구조가 된다. 두 노드(N1, N2)의 병합으로 생긴 새로운 노드(Nm)는 N1, N2의 super-set 이면서 N1, N2의 공통 조상인 A의 자식 노드가 된다. Nm의 부모 노드 A는 의미상으로 Nm에 가장 가까운 노드가 되는데, 이에 관한 알고리즘과 자료 구조를 제공한다.

6. 구현

RSM 은 현재 mediator인 확장-한울 시스템의 일부로서 초기 버전이 구현되어 있고, 개발환경은 UNIX 운영체제와 프로그래밍 언어로 자바2를 사용하였다. 현 시스템은 자바 언어의 문제점인 느린 인터프리트 속도를 개선하기 위해 개발된 각종 부가 서비스를 사용하여 현재 자바 시리즈에서 가능한 최상의 실행환경을 갖춘 상태이다

RSM은 mediator 시스템 내에서 검색한 결과 집합을 저장하고 관리하기 위한 독립된 모듈이다. 원격 정보를 대상으로 한 질의를 처리하는 meta-search 엔진이 결과를 지역(local)에 저장하고 관리하는 RSM의 기능을 사용하기 위해서는, RSM과 meta-search 엔진 사이의 통신을 위해 지킬 필요가 있는 간단한 정보교환 프로토콜을 제공하기만 하면 된다

7. 결론 및 향후연구

mediator처럼 대량의 질의 결과가 검색되는 상황에서는 사용자가 전부 확인하기 힘든 검색 결과 속에서 필요한 정보를 찾을 수 있어야 한다. 이처럼 대규모의 정보가 제공되는 상황에서, RSM은 이전의 질의를 지역적으로 관리함으로써, 기존 질의에 새로운 질의 조건을 추가하여 다시 원격에서 질의하는 비효율성을 제거할 수 있다. 또한, 기존의 질의 결과를 보존하여 그 속에서 필요한 정보를 찾아 낼 수 있는 연산을 제공하여 정보탐색에 필요한 풍부한 정보를 유지 관리 할 수 있다.

현재의 모델은 단순히 몇 가지 연산을 제공하기 위한 구현의 수준인데 이 모델에 대한 이론적 기반을 구축하는 일이 필요하다 성능에 관련해서는 많은 양의 정보를 지역적으로 관리하므로, 시간과 저장공간의 관점에서 재질의가 얼마나 성능을 발휘하는지에 대한 평가 기준의 마련이 추가로 연구되어야 할 과제이다

< 참고 문헌 >

[1] michelle Q Wang Baldonado, *"An interactive, structured-mediated approach to exploring information in a heterogeneous, distributed environment"*, Ph.D. Dissertation, Stanford University, 1997

[2] Jade GoldStein, Steven F. Roth, *"Using Aggregation and Dynamic Queries for Exploring Large Data Sets"*, SIGCHI, 1994

[3] Marti A. Hearst, David R.Karger, Jan O. Pederden, *"Scatter/Gather as a Tool for the Navigation of Retrieval Results"*, AAAI, 1995

[4] Wanda Pratt, *"Dynamic Organization of Search Results Using the UMLS"*, AMIA, 1997

[5] Marti A Hearst, Chandu Karadi, *"Cat-a-Cone: An Interactive Interface for Specifying Searches and Viewing Retrieval Results using a Large Category Hierarchy"*, SIGIR Conference. 1997

[6] michelle Q Wang Baldonado, Terry Winograd, *"SenseMaker: An Information-Exploration Interface Supporting the Contextual Evolution of a Users Interests"*, SIGCHI, 1997