정보검색관리시스템 KRISTAL-2000 설계 및 구현

이민호[0] 진두석 김광영 주원균 서정현 류범종
한국과학기술정보연구원
(cokeman[0], dsjin, kykim, joo, jerry, ybj)@kisti.re.kr

Design & Implementation of Information Retrieval & Management System KRISTAL-2000

Min-Ho Lee[0], Du-Seok Jin, Kwang-Young Kim,
Won-Kyun Joo, Jerry Hyeon Seo, Beom-Jong You
Dept. Information System Research, Korea Institute of Science & Technology Information

요 약

데이터베이스를 구축하고 서비스하기 위한 대부분의 방식은 상용 DBMS를 사용하여 문서에 대한 관리를 수행하고, 사용자 검색을 수행하기 위해 정보검색 시스템을 사용하는 DBMS와 정보검색 시스템의 연동방식을 채택하고 있다. 이러한 환경은 DBMS와 정보검색 시스템을 이중으로 운영하여야 하는 불편함이 있고, 변동된 문서에 대한 데이터 일관성을 유지하기 어렵고, 데이터를 중복 저장하므로 저장 공간의 낭비를 가져온다. DBMS를 기반으로 정보 검색 시스템을 밀결합하는 방법도 있으나 검색 속도 저하문제를 야기하였다. 본 논문에서는 정보 검색 시스템을 바탕으로, DBMS의 필수 요소인 안정적인 문서 관리 부분을 접목한 정보검색 관리 시스템을 설계한다.

1. 서 론

인터넷과 인트라넷 사용의 확산과 전자문서에 대한 요구사항이 증대함에 따라 점차 처리해야 할 데이터는 기하급수적으로 증가하고 있고, 그 종류 또한 다양한 형태로 생성되고 있다. 텍스트 문서 외에 여러 멀티미디어 문서들에 대한 저장 및 관리에 대한 요구사항 또한 증가하고 있는 추세이며, 과거 정보검색에 대한 수요가 일부 전문가 집단에 의해 주도 되었던 때와 달리, 여러 계층의 다양한 요구사항을 수용하기 위하여 다양한 방식의 정보 검색 모델을 제시하고 지원하여야 할 필요성도 증대되고 있다. 이러한 요구사항을 지원하기 위한 대부분의 방식은 상용 DBMS를 사용하여 문서에 대한 관리를 수행하고, 사용자 검색을 수행하기 위해 정보검색시스템을 사용하는 DBMS과 정보검색 시스템의 연동 방식을 사용하여왔다. 이러한 환경은 DBMS와 정보검색시스템을 이중으로 운영하여야 하는 불편함이 있고, 문서가 변경됨에 따라 DBMS 시스템에 저장된 정보와 정보검색 시스템에 저장된 색인정보와의 일관성을 유지하기 어렵고, 데이터를 중복 저장하므로 저장 공간의 낭비를 가져온다.[1] 다른 방법으로 DBMS에 정보검색 시스템을 밀 결합하는 방식이 있으나 기존 DBMS가 가지는 한계로 인하여 대용량의 비정형 문서를 처리하기에는 많은 속도저하를 감수하여야 했다. KISTI에서는 정보 검색시스템의 기술을 기반으로 DBMS의 가장 기본적인 기능인 데이터의 추가, 삭제, 변경 등의 데이터 처리, 트랜잭션과 데이터 복구 등의 요소기술만을 추가하여 기존 정보검색 시스템에 비해 안정적인 데이터 관리를 가능하게 하는 KRISTAL-2000을 개발하였다. 본 논문에서는 데이터 추가, 삭제, 변경 등의 데이터 처리를 함과 동시에 이를 바로 검색할 수 있는 KRISTAL-2000의 구조를 2절에서 설명하고, 검색과 데이터 처리 요구시 KRISTAL-2000이 처리하는 방식을 설명한다. 4절에서는 기존 결과 셋과 데이터 불일치가 발생하였을 때 문제를 해결하는 방식을 설명하고, 끝으로 결론을 맺는다.

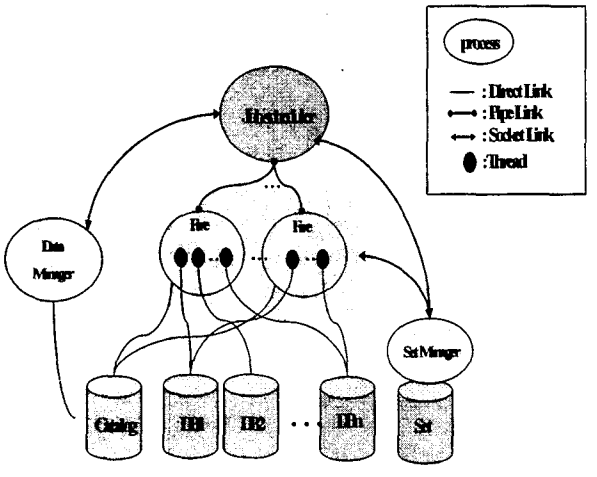
2. KRISTAL-2000 구조

KRISTAL-2000은 검색 성능을 떨어뜨리지 않으면서도 DBMS의 장점인 안정적 데이터 관리 기능을 구현하기 위하여, 검색을 위한 부분과 데이터 관리를 위한 부분이 별도의 데몬 방식 프로세스로 동작하도록 설계되었다. 또한, 보다 나은 검색 속도를 위하여 검색 결과를 임시 저장하는 캐쉬 서버를 데몬 방식 프로세스로 별도로 두었다.

검색을 담당하는 FIRE(Fast Information Retrieval Engine)는 서버

풀 방식으로 미리 정해진 개수만큼 띄워져 사용자의 요구를 각각 담당함으로써, 동시에 들어오는 다수의 사용자 요구를 처리할 뿐만 아니라, 데이터 처리와 검색 요구를 동시에 수용할 수 있다. 또한, 미리 정해진 개수보다 더 많은 요구가 들어오면 요구를 처리할 수 있는 수만큼 프로세스가 자동으로 더 생성되어 검색을 수행하고, 검색 수행이 끝나면 서버 풀에 원래 지정된 수만큼만 남기고 나머지 프로세스들은 종료된다. 그러므로, 빠른 검색 응답율과 서버 자원의 효율적인 이용이 가능하다. 또한 FIRE는 멀티 쓰레드를 발생시켜 동시에 여러 DB들을 엑세스할 수 있어 더 빠른 검색을 수행할 수 있다. 이러한 FIRE 검색 프로세스 생성과 종료, 현재 검색을 수행하지 않는 유휴 프로세스를 탐지하여 업무를 할당하는 등의 임무는 Job Scheduler라는 상위의 프로세스가 담당한다. Job Scheduler는 사용자의 요구 메시지를 분석하여 데이터관리 요구이면 Data Manager로 요구를 보내고 검색 요구이면 FIRE로 요구를 보내는 요구 분배 임무와 데이터가 추가, 삭제 혹은 변경되어 FIRE가 오픈한 데이터베이스와의 데이터 불일치가 발생하면 FIRE에게 데이터베이스를 재오픈하도록 명령하고 Set Manager에게는 캐쉬를 비울 것을 명령하는 일도 담당하고 있다. Data Manager는 온라인 데이터 관리 요구를 트랜잭션을 통해 처리하며, 로그를 통해 에러 복구를 할 수 있다. Set Manager는 검색 결과를 일시적으로 저장하는 캐쉬역할을 한다. 그러므로, 같은 질의가 들어왔을 때는 다시 검색을 수행하지 않고 바로 결과를 리턴해 줄 수 있다.

그림1은 KRSITAL-2000의 전체 프로세스 구조를 나타낸다. 각 데몬들은 TCP 방식의 socket을 이용하여 메시지를 전달하며, Job Scheduler와 FIRE 간에는 Pipe를 통하여 메시지를 전달한다. [2]



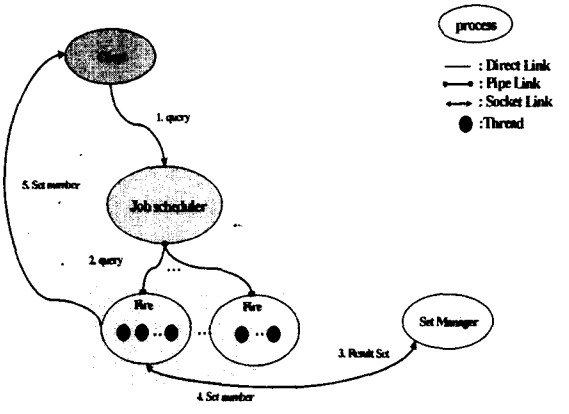
[그림 1] KRISTAL-2000의 전체 구조

3. 사용자 요구 처리 방법

사용자의 요구는 검색 수행 요구인지 데이터 관리 요구인지를 먼저 Job Scheduler에서 분석하게 된다.

검색 요구 입력시

Job Scheduler는 자신의 유휴 프로세스 테이블을 검색하여 유휴 상태인 Fire에게로 요구를 전달한다. Fire는 질의 처리를 거쳐 확장된 질의를 B-tree와 비교하고 발견된 단어들의 Posting 정보를 DB의 수만큼 thread를 발생시켜 각 DB에서 가져온다. 가져온 Posting 정보에서 DF,TF 정보를 추출, 가중치를 계산하여 검색 결과를 생성한다.[3] 최종 검색 결과는 Set Manager로 보내지고 Set Manager는 이를 Set DB (캐쉬)에 저장한다. 저장하여 얻어진 Set 번호는 다시 Fire에게로 리턴되며 Fire는 최종적으로 검색을 요구한 Client에게로 이 Set 번호를 넘겨준다.



[그림 2] 검색 요구시 처리 흐름

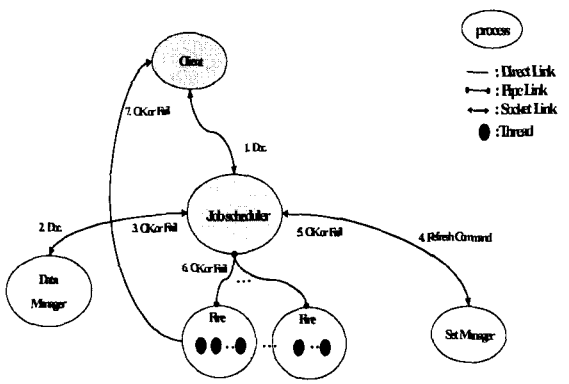
데이터 관리 요구 입력시

Job Scheduler는 이 요구를 Data Manager로 보내는데, 이때 Job Scheduler는 Data Manager로 보내진 요구 처리가 끝날 때까지 Blocking 상태로 있게 된다. 따라서 클라이언트로부터의 새로운 요구는 받아들일 수 없게 되어 DB가 갱신되고 있는 동안의 검색 일관성이 유지된다. 데이터 관리 처리가 Data Manager로부터 끝나게 되면 처리 결과를 Job Scheduler가 받아서 정상적으로 처리되었을

경우에만 Set Manager에게 데이터가 변경되었음을 알려 Set DB를 갱신하도록 한다. Set Manager는 자신의 Set DB에 DB가 변경된 시점을 표시하는 TimeStamp를 찍고 처리 결과를 리턴한다. Job Scheduler는 각 유휴 프로세스 테이블에 DB가 갱신되었음을 알리는 Dirty Bit를 세팅한다. 이 후에 Fire로 전달되는 모든 메시지는 이 Dirty Bit와 같이 전달되게 된다. Job Scheduler는 유휴 프로세스 테이블을 검색하여 유휴상태인 Fire에게 데이터 관리 요구의 처리 결과를 전달하며, Fire는 Dirty Bit가 셋 되었으므로 Open 되어 있는 DB를 닫은 후 재 Open하게 된다. 그 후 최종결과를 Client에게 보낸다.

4. 기존 결과 셋과의 데이터 불일치 처리 방법

KRISTAL-2000 은 온라인 데이터 변경을 하면서 질의에 대한 검색 결과를 일시적인 캐쉬에 저장하는 방식이기 때문에, 사용자가 질의 검색을 끝낸 후 데이터가 변경이 되면 캐쉬에 들어있는 기존 검색 결과와 실제 데이터간에 불일치가 발생하게 된다. 이를 그대로 두면 사용자는 기존 결과 셋 번호를 가지고 데이터를 엑세스하기 때문에 자신이 검색을 요구한 질의와 전혀 부합되지 않는 문서가 엑세스 되거나 혹은 이미 검색이 된 문서가 실제로는 없어진 상황이 발생할 수도 있다. 그러므로 KRISTAL-2000에서는 Set Manager에 Timestamp 역할을 하는 데이터가 변경된 시점의 셋 번호를 기록한다. 결과 셋 번호는 검색이 수행될 때마다 증가하기 때문에 Timestamp 번호 이전의 셋 번호에 대한 요청은 데이터 불일치가 있을 수 있는 결과 집합의 요청이다. 따라서 이러한 경우에는 검색을 다시 하라는 응답 메시지를 보낸다.



[그림 3] 데이터 관리 요구시 처리 흐름

5. 결론 및 향후 계획

기존에는 비정형 문서의 데이터 관리 및 검색을 위하여 DBMS와 정보검색 시스템의 연동방식을 사용하여 왔으나, 이러한 방법은 DBMS와 정보검색시스템을 이중으로 운영하여야 하는 불편함, 데이터 변경에 따른 DBMS 시스템에 저장된 정보와 정보검색 시스템에 저장된 정보의 불일치성, 저장 공간의 낭비등의 문제를 가져왔다. 다른 방법으로 DBMS에 정보검색 시스템을 밀 결합하는 방식이 있으나 대용량 비정형 문서를 처리하기에는 속도저하 문제가 발생하였다. 본 논문에서는 기존 관점과는 다르게 정보검색 시스템에 가장 필수적인 정보관리 기능만을 추가하는 방법으로 검색 성능을 떨어뜨리지 않고 온라인 데이터 처리가 가능하도록 시스템을 설계하였다.

향후 계획으로는, 우선 본 논문에서 제시하지 못한 DBMS 와 정보검색 시스템을 밀 결합한 방식과 KRISTAL-2000과의 검색 성능 비교 실험을 할 예정이다. 시스템 성능 향상을 위하여는 하부 저장엔진과 데이터베이스를 여러 대의 기계에 분산 배치하고, 이를 병렬 검색을 수행하여 검색 속도를 향상시키려고 한다. 또한, 내부 데몬들과 하부 저장엔진들 사이의 메시지 전달 방법을 RPC를 통하여 좀 더 안정적이며 확장이 용이하도록 변경할 계획이다. 또한, 현재 지원되고 있는 데이터 삽입, 삭제, 변경 외에 DBMS에서 지원하고 있는 조인 연산과 데이터 관리 편리성을 위해 정보검색관리 시스템을 위한 SQL인 IRSQL 을 개발하는 것도 과제로 남아있다.

[참고 문헌]

[1] 한국과학기술정보연구원, "과학기술정보유통시스템 개발 연구보고서", 2002.

[2] W.Richard Stevens, "UNIX Network Programming vol.1", Prentice Hall, 1998.

[3] Ricardo Baeza-Yates, Berthier Ribeiro-Neto, "Modern Information Retrieval", Addison Wesley, 1999.