PubMed 미러링 시스템, PubMedIF의 개발

이유진° 차재혁
한양대학교 정보통신대학원
eu98°@ihanyang.ac.kr, chajh@hanyang.ac.kr

Implementation of PubMed Mirroring System, PubMedIF

Yujin Lee° , Jaehyuk Cha
The Graduate School of Informations & Communications, Hanyang University

요 약

본 논문은 PubMed를 통한 데이터 미러링 방법 및 시스템 개발에 대해 기술한다. 기존의 PubMed 데이터베이스 검색 시스템은 원격의 사용자와 network로 연결되었기 때문에 대역폭의 제한을 받으며, Web query interface를 이용하기 때문에 검색 방법이 제한적이다. 또한 Text mining과 같은 local application의 이용에는 효과적이지 못하다. 본 논문에서는 효과적인 미러링을 위한 검색어를 제안하고, network 연결 상황에서 발생할 수 있는 비정상적 종료를 극복할 수 있는 PubMedIF를 개발하였다.

1. 서 론

PubMed는 미국 국립의학도서관(National Library of Medicine, NLM)[1]의 서지 데이터베이스 검색 시스템으로서 MEDLINE, Nucleotide, Protein Sequence 등의 데이터베이스를 검색할 수 있는 Web 기반 문헌 검색 시스템이다. PubMed는 의학, 생명의학과 관련된 저널 초록과 서지 정보를 포함하고 있으며 관련 연구자들의 이용이 활발하다[2][3][4]. 그러나 PubMed 사용자들은 Network를 통하여 검색함으로 작은 대역폭 때문에 많은 질의를 한번에 할 수 없고, Web query interface의 제한으로 인해 데이터를 사용자 목적에 따라 조작하기 어렵다. PubMed에서는 한번에 검색하는 레코드 수에 제한을 주거나 서버에 부하를 줄 경우 해당 사용자의 IP를 차단하고 있으며, PubMed의 Web query interface인 Boolean operation과 Search Field Description and Tag[3]로는 사용자 목적에 맞는 조작이 어렵다. 또한 Text mining과 같은 local application을 사용하기 위해 local 데이터를 가지고 있는 것이 더 효율적이다.

본 연구에서는 관련 연구자의 효율적인 정보 검색 활동 및 목적에 맞는 조작을 할 수 있도록 PubMed 미러링 시스템을 개발하고자 한다. PubMed 데이터베이스의 데이터를 미러링 하여 local에 유지는 것은 기존의 PubMed에서 제공하는 문헌 검색을 자유롭게 할 뿐 아니라 Text mining을 이용한 관련 문서간의 새로운 정보 추출 등 자유로운 데이터 조작에 효율적이다. PubMed 미러링 시스템을 개발하기 위해서는 원본 데이터베이스의 내용 그대로 복사하여 local에 유지하는 것은 물론이고, 원본에 수정사항이 있으면 복사본에도 수정하여 항상 원본과 같게 하기 위한 전략이 필요하다. 원격지의 PubMed와 network로 연결하여 미러링 하기 때문에 network상 접속이 불안정한 상황과 비정상적으로 연결이 끊기는 것에 대한 문제 등을 해결해야 한다. 이러한 문제를 해결하고 PubMed 미러링에 대한 전 과정을 자동화할 수 있도록 구현하고자 한다.

2. 기존 PubMed 사용의 어려운 점 및 제약사항

첫 번째는 PubMed의 Search System에서의 검색상의 문제점으로, Web query interface에서 Search Field Description and Tag과 Boolean operation만을 제공하기 때문에 검색에 제한점을 가지고 있다. Search Field Description and Tag는 Author는 [AU]로 Journal Title은 [TA]로 Publication Date는 [DP]로 나타낼 수 있는 것으로, "o'brien j[AU]"와 "1998[DP]"처럼 각 검색어 뒤에 대괄호([])를 붙여서 검색어를 명확하게 하는 방법을 사용하고 Boolean operation[3]은 다른 Web 기반 검색에서처럼 AND, OR, NOT을 말한다. 그러나 이것만으로는 사용자 목적에 맞는 조작을 하기는 어렵다. 예를 들면, 1990년부터 1993년까지의 저널을 찾는다든지, 개별적인 단백질 기능에 대한 연구문헌을 이용하여 단백질간의 상호작용에 대해 알아보고자 하는 등 기존 자료로부터 새로운 정보를 추출하는 것은 불가능하다.

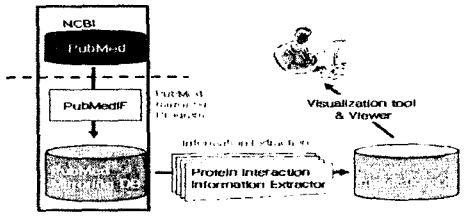
두 번째는 PubMed와 사용자는 network로 연결 되어 있기 때문에 network 상 문제가 발생하거나 한쪽 시스템의 비정상적인 종료 시에는 처음부터 다시 검색을 해야 하는 어려움이 있다.

세 번째는 PubMed 자체에서 사용자에게 준수하기를 요구하는 제약 사항이다. 이는 일부 사용자에 의해 전체 시스템에 부하를 주지 못하게 하기 위한 사항들이지만 이를 어겼을 때에는 사용된 IP로는 접속이 불가능하게 된다.

①, 미국 동부 표준시로 17시부터 익일 오전 5시까지를 제외한 시간에는 100번 이상의 쿼리를 할 수 없다.

②, 연속해서 3초 이내에 쿼리를 하지마라.

③, 하루에 5000개 이상의 레코드를 쿼리 할 수 없다.[5]



[그림 1] PubMed 미러링의 구성도

3. PubMed 미러링 시스템 개발

PubMed를 미러링을 통해 기존 PubMed 사용에 어려움을 극복해야 하며, 제약사항을 준수하기 위한 방법 및 network의 비정상적인 종료를 극복하는 방법이 필요하다.

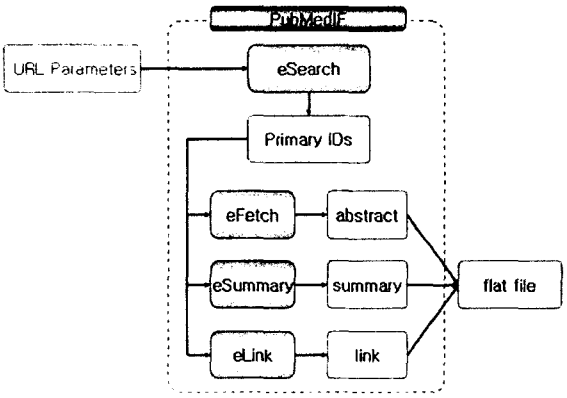
원본과 같은 데이터베이스를 만드는 것은 물론 원본에 추가 데이터가 생겼을 때, 그 증분에 대해 중복 없이 점증적으로 추가 할 수 있는 검색어를 만드는 것이 필요하다.

3.1절에서는 PubMed 미러링을 위해 만들어진 PubMedIF라는 미러링

자동화 시스템의 내부 구조에 대해 소개하고, 3.2절에서는 PubMedIF의 미러링에 필요한 검색어 전략에 대해 설명한다. 3.3절은 네트워크 상황 극복하는 방법에 대해 설명하며, 3.4절은 미러링 된 데이터를 관리하기 위해 파일 저장하는 방법에 대해 간략하게 소개한다.

3.1 PubMedIF의 내부 구조

PubMed 서지 데이터베이스의 미러링 데이터베이스 구축을 위해 PubMedIF라는 프로그램을 구현하였다. PubMed에서 제공하는 eSearch, eFetch, eSummary, eLink로 되어 있는 Entrez Programming Utilities[5][6]는 각기 개별적으로 혹은 두개의 Utilities가 함께 이용하도록 되어 있다. 이것을 이용하여 PubMedIF에서는 URL의 HttpConnection을 열어서 쿼리를 실행할 때 각 Utilities가 연속이 일어나도록 하고 이를 자동화 한 것이다. 즉 eSearch에서 고유한 PubMed ID(PMID)를 모두 얻어서 그에 대한 논문 초록을 eFetch를 통해 얻고, 요약정보를 eSummary, 서지 정보와 관련된 링크 정보를 eLink를 통해 얻는다. 3.2절에서는 각 Utilities를 모듈화 하여 통합한 시스템에 대해서 설명하겠다.



[그림 2]PubMedIF의 내부 구조(모듈화 된 query interface)

3.2 PubMedIF의 PubMedID(PMID)를 통해 자료를 얻기 위한 검색어 전략

eSearch에서는 고유키인 PMID를 얻기 위한 검색어 전략이 필요하다. 검색어의 요구 조건 다음과 같다.

첫째. 전체 데이터를 빠짐없이 얻어 올 수 있어야 한다.

둘째. 추가 데이터에 대해 점진적으로 중복 없이 쿼리 할 수 있는 것이어야 한다.

셋째. 쿼리의 횟수가 적을수록 좋다.

여기서는 5가지 방안에 대해 설명하고 적절한 검색어를 설명하겠다.

방안 1) PubMed ID(PMID) - PMID가 고유한 키이므로 중복 없이 전체 데이터를 증가되도록 얻을 수 있다. 그러나 한번에 하나의 PMID를 쿼리 하기 때문에 세 번째 요구 조건은 만족하지 않는다.

방안 2) 저널 타이틀 리스트를 이용[7] - PubMed에서는 논문이 실린 저널 타이틀 리스트를 제공하며 이는 첫째 조건과 셋째 조건을 만족한다. 그러나 추가 데이터만 구분하여 쿼리 할 수 없으므로 둘째 조건을 만족하지 않으므로 적당하지 않다.

방안 3) 주제어를 통한 검색어[3] - cancer나 AIDS와 같은 주제어로 검색어를 만드는 것이다. 이것은 셋째 조건은 만족하지만 전체 데이터를 가져올 수 있다는 보장이 없으며 추가 데이터만 구분하여 쿼리 할 수 없으므로 첫째, 둘째 조건을 만족하지 않는다.

방안 4) 출판날짜를 통한 검색어[3] - 출판날짜를 검색어로 함으로써 데이터를 점증적으로 얻을 수 있다. 또한 추가 데이터만 구분하여 쿼리 하는 것이 가능함으로 첫째, 둘째, 셋째 조건 모두를 만족한다. 그러나 테스트 결과 PubMed DB에 입력한 날짜가 출판 날짜 순서대로 되어 있지 않기 때문에 오래전에 출판된 논문이 근래에 데이터베이스에 추가되었다면 미러링 되지 않을 수 있는 문제가 있다.

방안 5) 데이터베이스에 입력된 날짜 이용[3] - 데이터베이스에 입력된 날짜로 검색어를 만드는 방법은 첫째, 둘째, 셋째 조건 모두를 만족하는 가장 적당한 방법이다. 구체적인 방법은 3.2.1절에서 설명하도록 한다.

3.2.1 PubMedIF 자동화에 대한 구체적인 전략(1) - 데이터베이스에 입력된 날짜로 검색어 만들기

PubMed 데이터베이스는 1960년대 중반부터 구축되었기 때문에 그때부터 현재날짜까지를 검색어로 하면 된다.

만약 1960년 1월 1일부터 2003년 3월 31일까지를 검색어로 만든다면 YYYY/MM/DD:YYYY/MM/DD[edat]의 형식대로 1960/01/01:2003/03/31[edat]라고 사용하면 된다. [edat]는 Search Field Description and Tag로 데이터가 입력된 날짜를 의미하는 태그이다. 즉, PubMed는 Entrez에 속한 데이터베이스 중 하나 이므로 [edat]는 PubMed 데이터베이스에 입력된 날짜를 일반적으로 일컫는 것이라 할 수 있다.

추가 데이터에 대해서도 미러링 이후의 날짜부터 현재 날짜까지를 검색어로 하면 증가분에 대해 미러링 할 수 있다. 만약 1960년 1월 1일부터 2003년 3월 31일까지 미러링 한 이후에 5월 31일까지의 증가분을 미러링 하려한다면 2003/04/01:2003/05/31[edat]으로 검색어를 만들면 된다. 원본 데이터베이스에 새로운 데이터가 추가되더라도 입력된 날짜를 기준으로 하기 때문에 데이터베이스에 입력된 날짜를 이용하는 것은 적당한 검색어다.

3.2.2 PubMedIF 자동화에 대한 구체적인 전략(2) - 제약사항 해결

2.2절의 세 가지 제약 사항을 지키면서 미러링이 자동으로 진행되고, 진행상태를 로그 파일을 기록하여 미러링이 자동으로 실행되더라도 현재 상태를 파악할 수 있도록 한다.

2.2절의 제약 사항을 지키기 위해 유효시간에 5000개의 레코드를 가져오도록 한다. 즉, 이 시간 동안 5000개의 레코드를 가져오고 멈추었다가 다음날 같은 시간에 지속적으로 반복하도록 해야 한다. 이를 위해 Entrez 데이터베이스에 입력 날짜인 검색어에 대해 eSearch로 검색된 전체 PMID 수를 5000개씩 나눠서 유효한 시간에만 eFetch, eSummary, eFetch가 실행되도록 한다.

[그림 3]을 통해 자세히 소개하면 다음과 같다.

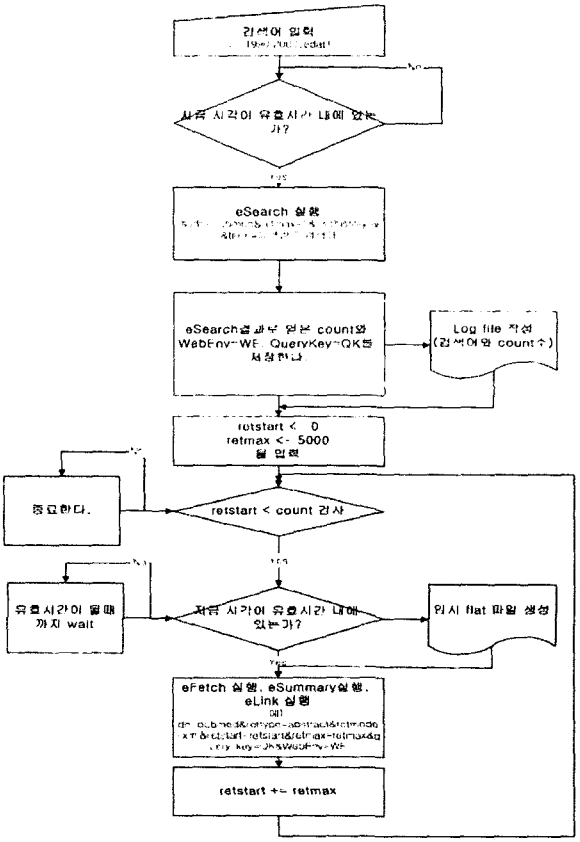
① 사용자로부터 날짜 형식의 검색어를 입력 받은 후 현재 시간을 검사하여 현재 시각이 첫 번째 제약 사항에서의 유효한 시간 내에 포함되면 실행을 하고, 그렇지 않으면 대기 한다. 대기 시간은 유효 시간과 현재 시간의 차이 만큼이며, 대기하면서 계속 유효시간인지 확인 한다. 이것은 Timer를 두어 시간을 계속 체크 하도록 한다.

② eSearch 단계에서 검색어에 대한 전체 레코드 수 및 PMID를 얻고 로그파일을 생성한다. 로그파일은 검색어와 전체 레코드 수를 기록한다.

또한 검색어에 대한 QueryKey와 WebEnv를 얻는다. 이것은 Web query interface의 URL parameters이다. 이것을 통해 한번 입력한 검색어에 대해 같은 검색어 입력을 계속 하지 않도록 사용자 편의를 위한 것으로 eFetch나 eSummary의 검색어에서도 이용된다.

③ 또한 eSearch 단계에서 retstart = 0, retmax=5000을 입력한다. 이것 도 URL parameters로써 retstart는 몇 번째부터 결과를 보여주는지 정하는 부분이고, retmax는 몇 개의 결과를 보여주는지 정해주는 부분이다. 각각의 값을 위와 같이 정하면 0부터 5000개의 검색된 레코드를 보여준다. eFetch, eSummary, eLink 실행 후 retstart는 retmax만큼 증가한다. 이렇게 계속 retstart의 값을 증가하여 5000개씩 쿼리 하도록 하며 PMID값을 배열에 저장해 둔다.

④ 각 PMID에 대해 각 모듈을 실행하고, 쿼리 결과를 파일에 저장하고 로그파일을 다시 기록한다.



[그림 3] PubMedIF의 흐름도

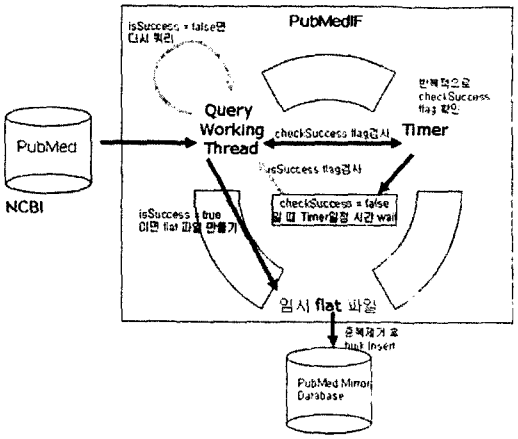
3.3 네트워크 상황 체크 및 자동 시작하기

PubMedIF는 URL의 HttpConnection을 열어서 쿼리에 따른 데이터를 PubMed로부터 가져온다. 네트워크 상황을 체크하기 위해 쿼리에 대한 결과 성공여부와 지연시간을 기록한다. 이를 위해 Query Working Thread와 Timer를 만들었다.

Query Working Thread는 eSummary, eFetch, eLink로 구성된 것으로, eSearch에 의해 검색된 결과인 PMID를 배열로 저장하여 각각의 PMID에 대해 Query Working Thread를 실행하도록 한다. 즉 하나의 PMID에 대해 하나의 Query Working Thread가 작동하며, 그에 대한 결과 상태를 Boolean 값으로 저장한다. 쿼리에 성공하여 데이터를 얻어오면 checkSuccess라는 값을 true, 그렇지 않으면 false를 가진다.

Timer는 일정시간 간격으로 성공여부를 검사하는데, 만약 Timer가 checkSuccess값을 검사했을 때 checkSuccess값이 false이면, Query Working Thread는 일정시간 대기 한다. 잠시 대기후에도 Query Working Thread의 check Success가 바뀌지 않으면 isSuccess라는 flag를 false로 한다. 즉 현재 단계의 검색이 실패했음을 기록하고, 자료를 얻어오는 것은 실패했지만 아직 살아있는Thread를 없앤다. 이렇게 함으로써 실패한 Query Working Thread가 계속해서 HttpConnection을 하고 있지 않도록 한다. 실패한 쿼리에 대해 다시 쿼리 하도록 하는 것은 이미 PMID값이 배열에 저장되어 있기 때문에 다시 실행하도록 할 수 있다. 위와 같이 Timer와 Query Working Thread를 이용하여 Network 접속이 불안정 하거나 비정상적 연결로 PubMedIF가 종료하지 않도록 하였다. 즉, 이것은 PubMedIF에서 실패한 쿼리에 대해 지속적으로 재시도가 가능하기 때문이다

3.4 파싱 및 임시 flat 파일 데이터베이스에 넣기



[그림 4]PubMedIF의 Query Working Thread와 Timer

Query Working Thread에 의해 가져온 결과는 필요한 데이터 부분만 파싱할 필요가 있다. eSummary의 결과는 출판날짜, 저자이름, 제목, 저널 타이틀, 페이지 수, 권 수, 사용된 언어, 저자 주소, ISSN등으로 eFetch의 정보와 겹친다. 그러므로 eFetch에서 eSummary에 없는 논문 초록만 파싱하여 저장한다. 이렇게 중복된 내용이 없도록 파싱한 결과는 구분자를 두어 flat파일(text파일)로 만든다. 각 flat파일은 5000개씩 구분되어 저장되도록 하며, "data검색어_블록 번호.txt"의 형식으로 파일을 만든다. 임시 flat파일은 PubMed미러링 데이터베이스에 다시 저장 한다.

4. 결 론

PubMedIF라는 미러링 시스템의 개발을 통해 지속적이고 점증적으로 PubMed 데이터를 얻어 올 수 있게 하였다.

PubMedIF는 미러링에 적당한 검색어를 선택하며, 기존 PubMed 검색의 제약사항을 극복하기 위해서 유효시간을 체크하고 쿼리 수를 제안하여 실행하도록 하였다. 또한 네트워크 상황을 극복하기 위해 Timer와 쿼리 상태를 Boolean값으로 표현하고 비정상적인 종료를 하지 않도록 자동으로 재시도가 가능한 시스템을 구현하였다. 그리고 입력된 검색어와 검색어에 대한 전체 검색 레코드 수 및 네트워크의 상태를 로그로 기록함으로써 진행상황을 알 수 있게 하였다.

5. 참 고 문 헌

[1]http://www.ncbi.nlm.nih.gov NCBI

[2]http://www.ncbi.nlm.nih.gov/entrez/query/static/overview.html PubMed Overview. NCBI

[3]http://www.ncbi.nlm.nih.gov/entrez/query/static/help/pmhelp.html PubMed Help. NCBI

[4]Kathi Canese, Jennifer Jentsch, and Carol Myers. PubMed: The Bibliographic Database. NCBI pp.9

[5]http://eutils.ncbi.nlm.nih.gov/entrez/query/static/eutils_help.html Entrez Programming Utilities. NCBI

[6] Jim Ostell. The Entrez Search and Retrieval System. NCBI. pp.6

[7]List Of Journals Indexed. National Library of Medicine. pp.301.