웹크롤러의 서버 오버헤드 최적화 시스템 설계

이종원 · 김민지 · 김아용 · 반태학 · 정회경

배재대학교 컴퓨터공학과

Web crawler designed utilizing server overhead optimization system

Jong-Won Lee · Min-Ji Kim · A-Yong Kim · Tae-Hak Ban · Hoe-Kyung Jung

Department of Computer Engineering, PaiChai University

E-mail : starjwon@naver.com, wnstjd697@naver.com, janlssary@naver.com, banth@hanmail.net, hkjung@pcu.ac.kr

요 약

기존의 웹크롤러들은 서버의 오버헤드 부담을 줄이면서 데이터의 무결성을 보장하기 위해 최적화 방안에 대해 지속적으로 발전해왔다. 기하급수적으로 빠르게 늘어가는 데이터의 양과 그 데이터들 중에서 필요한 데이터를 수집해서 사용해야 하는 현대인들에게 웹크롤러는 필수불가결의 존재이다.

본 논문에서는 기존의 웹크롤러 방식과 제안된 웹크롤러 방식의 효율성을 비교 및 분석하였다. 또한, 비교된 결과를 바탕으로 최적화된 기법을 제안하고, 웹크롤러의 데이터 수집 주기를 동적으로 조절하여 서버 오버헤드를 감소시키는 시스템에 대해 설계하였다. 이는 웹크롤러 방식을 사용하는 검색 시스템 분야에 활용될 것이다.

ABSTRACT

Conventional Web crawlers are reducing overhead burden on the server to ensure the integrity of data optimization measures have been continuously developed. The amount of data growing exponentially faster among those data, then the data needs to be collected should be used to the modern web crawler is the indispensable presence.

In this paper, suggested that the existing Web crawler and Web crawler approach efficiency comparison and analysis. In addition, based on the results, compared to suggest an optimized technique, Web crawlers, data collection cycle dynamically reduces the overhead of the server system was designed for. This is a Web crawler approach will be utilized in the field of the search system.

키워드

크롤러, 웹 사이트, 동적 주기, 오버헤드

Ⅰ. 서 론

인터넷을 통해 다양한 웹 페이지에서는 실시간으로 대량의 데이터들을 수집 및 기재하고 있다. 그로 인해 자신에게 필요한 데이터를 수집하기 위한 노력들이 곳곳에서 계속되고 있다. 이러한 노력들은 웹 크롤러의 개발로 이어졌고, 다양한 웹 크롤러들로 인해 데이터를 수집하는 일은 그리 어려운 일이 아니게 되었다. 하지만, 수집된 데이터가 사용자가 원하는 데이터가 아닌 경우가 빈번히 일어나며, 이러한 문제점을 해결하기 위해 데이터들을 수집과 정제, 데이터베이스로 통합하는 ETL(Extraction, Transformation, Loading) 도구가 필요하다. ETL 도구는 다양하며 Open Source로도 제공되고 있다. ETL 도구 중에서 데이터를 수집하는 기능을 하는 도구를 웹 크롤러라고 불리며, 웹 크롤러의 수집 대상 범위에 따라 범용 크롤러, 포커싱 크롤러, 토피컬 크롤러, 래퍼 기반 크롤러로 분류된다[1,2].

정적 수집 주기의 크롤러는 많은 데이터들을 중복 수집하기 때문에 서버 오버헤드가 발생하여 이러한 문제점으로 인해 사용자들은 수집 주기의 최적화가 필요하다.

본 논문에서는 기존의 정적 수집 주기의 문제점을 개선하기 위해 해당 웹 사이트에 수정 정보를 수집하고, 분석하여 동적 수집 주기를 생성한

뒤 할당한다.

Ⅱ. 관련 연구

웹 크롤러는 단순히 데이터를 수집만 하는 도구가 아니며, 목적에 따라 다른 크롤러들을 사용한다.

2.1 범용 크롤러와 포커싱 크롤러

범용 크롤러와 포커싱 크롤러[4]는 여러 파일 형태의 데이터들을 수집한다. 확실한 분야 설정이 이루어지지 않는 상황에서 데이터를 수집해야 한다면 범용 크롤러나 포커싱 크롤러를 통해 많은 양의 데이터를 수집할 수 있지만, 수집된 데이터의 최신성이나 정확성, 그리고 서버 오버헤드가 높다. 일반적인 검색이나 데이터 수집에 사용된다.

2.2 토피컬 크롤러

주제나 분야를 설정한 뒤 데이터를 수집하는 토피컬 크롤러는 범용 크롤러와 포커싱 크롤러보다 효율적으로 사용이 가능하다. 사용자 중심 설계가 가능하며, 원치 않는 주제를 설정하여 수집을 회피할 수 있으며, 수집된 데이터의 최신성과 정확성이 비교적 높다. 의학 분야에서의 데이터 수집이 적합하다.

2.3 래퍼 기반 크롤러

래퍼 기반 크롤러는 해당 웹 사이트를 미리 분석한 뒤 필요한 웹 페이지를 설정한 후 데이터를 수집하는 것이며, 수집 공간을 정해놓기 때문에 서버 오버헤드가 적다. 기업의 고객센터 페이지에 적용하는 것이 적합하다.

Ⅲ. 제안하는 방법

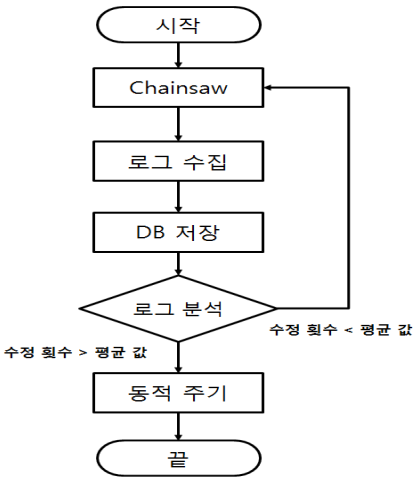

그림 1. 제안하는 시스템의 흐름도

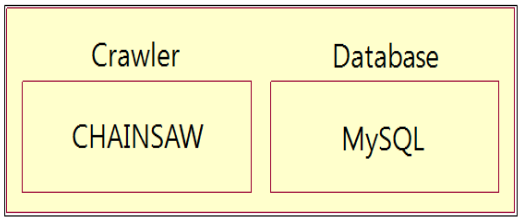

그림 2. 시스템의 구성도

그림 2는 시스템의 구성도이다. 크롤러는 자바로 구현한 CHAINSAW와 수집 주기로 이루어진다. 데이터베이스는 수집한 로그 기록과 로그 분석을 위한 테이블로 구성된다.

그림 1과 같이 시스템의 흐름은 CHAINSAW가 로그 기록을 수집한 뒤 데이터베이스에 저장하고, 저장된 로그를 분석한 뒤 각 URL별로 적합한 수집 주기를 적용하는 시스템이다.

3.1. 웹 사이트의 수집 주기 기록

그림 3과 같이 CHAINSAW를 이용하여 해당 웹 사이트의 수정된 데이터의 크기와 수정 횟수를 멀티쓰레딩을 통해 빠른 속도로 수집한다.

```
[main] DEBUG org.apache.log4j.chainsaw.MyTableModel - Total time [ms]: 4 in update, size: 120
[Thread-4] INFO org.apache.log4j.chainsaw.LoggingReceiver - Thread started
[Thread-4] DEBUG org.apache.log4j.chainsaw.LoggingReceiver - Waiting for a connection
[AWT-EventQueue-0] INFO org.apache.log4j.chainsaw.ExitAction - shutting down
```

그림 3. CHAINSAW를 이용한 로그 기록 수집

이 수치들은 데이터베이스에 수정된 시간과 수정된 데이터의 크기, URL 명으로 생성된 데이터베이스 테이블에 기록된다. 데이터베이스의 필드 항목으로는 표 1과 같다. 저장된 로그 기록들은 다음 단계인 로그 분석을 통해 각 URL별 수집 횟수를 정리한다.

표 1. 데이터베이스의 필드항목 정의

필드명	의미
URL_Name	검색대상 URL 주소
Update_Time	해당 웹 사이트가 수정된 시각
Take_Size	해당 웹 사이트에서 가져온 데이터 크기
Compare_Size	1시간 전 데이터 크기와 비교
Boolean	변경여부 확인(0,1)
Group	그룹 (1, 2)

3.2 로그 분석

로그 분석은 저장되어있는 로그 기록을 표 2와 같이 URL_Name과 Update_Time, 두 개의 필드 항목으로 정리한다.

표 2. URL별 데이터 수정 시각

URL_Name	Update_Time
U1	2014-03-31 13:41;00
U3	2014-03-31 15:02:00
U2	2014-04-01 15:28:00
U1	2014-04-02 13:50:00
U4	2014-04-03 14:12:00
U2	2014-04-03 15:36:00
U1	2014-04-04 13:48:00

U1은 일주일간 수정 횟수가 3번, U2는 2번, U3와 U4는 각각 1번씩이다. 그리고 수정 시간대를 보면 모든 URL들이 16시 전에 수정이 완료되었으며, 이를 기반으로 수집 주기의 시각을 16시로 정한다. 또한, 동적 주기를 측정하기 위해서 다음 단계인 URL별 그룹화를 진행한다.

3.3 동적 주기

URL들의 로그 기록을 보면 총 4개의 URL들이 있고, 전체 URL의 평균 수정 횟수는 1.75번이다. 평균값보다 많이 수정된 U1과 U2는 그룹 1이 되며, U3와 U4는 그룹 2가 된다.

그룹 1의 URL들은 2.5회의 평균 수정 횟수를 가지게 되며, 전체 평균보다 0.75가 높다. 그룹 1의 주기는 그룹 1의 평균에서 전체 평균을 뺀 뒤 그 값 X를 이용해서 정한다.

```
X= AVG1 - AVG;
for(i=1; i<3; i++)
{
if(Ui>AVG1)
N = Math.round(AVG1 + X);
else if(Ui==AVG1)
N = Math.round(AVG1 + X);
else
N = Math.round(AVG1 - X);
}
```

그림 4. 그룹 1의 동적 주기 가상 코드

그림 4와 같이 가상 코드를 적용시키면 수정된 데이터에 대한 수집 횟수를 최소한으로 줄일 수 있고, 데이터의 중복 수집을 예방할 수 있다. 그룹 1의 U1은 일주일에 3번씩 수집하고, U2는 2번 수집을 한다. 또한, 데이터 수집 시각은 16시이며, 요일은 해당 URL의 웹 사이트가 데이터를 수정한 요일이다.

그룹 2의 경우는 일주일간 로그 기록을 더 수집하여 그룹화를 한 뒤 수집 여부를 결정하게 된다. 수정 횟수가 평균값 보다 적으면 수집 주기를 2주에 1번으로 변경함으로써 불필요한 서버 오버헤드와 데이터 중복 수집을 줄일 수 있다.

Ⅳ. 결 론

본 논문에서 제안하는 시스템은 기존의 크롤러들이 가지고 있던 문제점들을 보완할 수 있는 클롤러 시스템 설계이다. 제안한 시스템은 수집 주기를 동적으로 할당하여 수집 횟수를 최소화한다. 또한, 서버의 오버헤드를 감소하여 데이터의 최신성을 유지할 수 있다. 정확한 검색을 위해서는 토피컬 크롤러를 사용하고, 데이터의 수정 횟수가 자주 발생하는 웹 사이트는 래퍼 크롤러를 사용한다.

향후 연구로는 다양한 웹 사이트에서 실험을 계속 진행하여 제시하고 있는 시스템의 유용성과 효율성 방향에 대한 개선 연구가 필요하다.

참고 문헌

[1] Hanhoon Kang, Seong Joon Yoo, Dongil Han, “Design and Implementation of Web Crawler Wrappers to Collect User Reviews on Shopping Mall with Various Hierarchical Tree Structure”, Korean Institute of Intelligent Systems, Vol.20, No.3, pp.318-325, 2010.6

[2] Chenghao Quan, Youngtak Lee, Youngjun Kim, Yongdoo Lee, “Design and Implementation of a High Performance Web Crawler“, Journal of the Korea Industrial Information Systems Research, Vol.8, No.4, pp.64-72, 2003.12

[3] Wansup Cho, Jeongeun Lee, Chihwan Choi, “Refresh Cycle Optimization for Web Crawlers“, The Korea Contents Association, Vol.13, No.6, pp.30-39, 2013.6

[4] Harry T Yani Achsan, Wahyu Catur Wibowo, “A Fast Distributed Focused-Web Crawling“, Procedia Engineering, Vol.69, pp.492-499, 2014.3

[5] Songhua Xu, Hongjun Yoon, Georgia Tourassi, “A user-oriented web crawler for selectively acquiring online content in e-health research“, Oxford Journals, Vol.30, No.1, pp.104-114, 2014.3

[6] Yaling Liu, Arvin Agah, “Topical Crawling on the web through local site-searches“, Journal of Web Engineering, Vol.12, No.3, pp.203-214, 2013.7