ISSN 2234-4438
융합정보논문지 제7권 제2호 pp. 43-51, 2017 DOI : https://doi.org/10.22156/CS4SMB.2017.7.2.043

큐를 이용한 다중스레드 방식의 웹 크롤링 설계

김효종[1], 이준연[2], 신승수[1*]
[1]동명대학교 정보보호학과, [2]동명대학교 디지털미디어 공학부

Multi-threaded Web Crawling Design using Queues

Hyo-Jong Kim[1], Jun-Yun Lee[2], Seung-Soo Shin[1*]
[1]Department of Information Security, Tongmyong University
[2]School of Digital Media Engineering, Tongmyong University

요 약 연구목적 : 본 연구의 목적은 광역 네트워크로 연결된 다수의 봇을 활용하여 단일처리 방식의 시간 지연의 문제점과 병렬처리 방식의 비용증가, 인력낭비에 대한 문제점을 해결할 수 있는 큐를 이용한 다중스레드 방식의 웹 크롤링을 연구한다. 연구방법 : 본 연구는 큐를 이용한 다중스레드 방식의 시스템 구성을 바탕으로 독립된 시스템에서 실행하는 어플리케이션을 설계하고 분석한다. 연구결과 : 큐를 이용하여 다중 스레드 방식의 웹 크롤러 설계를 제안한다. 또한, 웹 문서의 처리량을 수식에 따라 클라이언트와 스레드 별로 나누어 분석하고, 각각 효율성 비교를 통해 최적의 클라이언트의 개수와 스레드의 개수를 확인 할 수 있다. 제안하는 시스템의 설계 방식은 분산처리를 기반으로 각각의 독립된 환경에서의 클라이언트는 큐와 스레드를 이용하여 빠르고 신뢰성이 높은 웹 문서를 제공한다. 향후연구 방향 : 특정 사이트를 대상으로 하는 웹 크롤러 설계가 아닌 범용 웹 크롤러에 큐와 다중 스레드를 적용하여 다양한 웹 사이트를 빠르고 효율적으로 탐색 및 수집하는 시스템이 필요하다.

키워드 : 웹 크롤링, ERD, 다중스레드, 큐, URL

Abstract Background/Objectives : The purpose of this study is to propose a multi-threaded web crawl using queues that can solve the problem of time delay of single processing method, cost increase of parallel processing method, and waste of manpower by utilizing multiple bots connected by wide area network Design and implement. Methods/Statistical analysis : This study designs and analyzes applications that run on independent systems based on multi-threaded system configuration using queues. Findings : We propose a multi-threaded web crawler design using queues. In addition, the throughput of web documents can be analyzed by dividing by client and thread according to the formula, and the efficiency and the number of optimal clients can be confirmed by checking efficiency of each thread. The proposed system is based on distributed processing. Clients in each independent environment provide fast and reliable web documents using queues and threads. Application/Improvements : There is a need for a system that quickly and efficiently navigates and collects various web sites by applying queues and multiple threads to a general purpose web crawler, rather than a web crawler design that targets a particular site.

Key Words : Web Crawling, Entity-Relationship Diagram, Multi-threaded, Bot, Uniform Resource Locator

Received 2017-03-28 Revised 2017-04-10 Accepted 2017-04-12 Published 2017-04-30
*Corresponding author : Seung-Soo Shin (shinss@tu.ac.kr)

1. 서론

웹 사용자들은 여러 기기 및 매체를 통하여 쇼핑, 게임, 스포츠, 뉴스 기사 등과 같이 자신이 필요한 정확한 정보를 얻기 위하여 검색 서비스를 사용한다. 이러한 정보가 정보화 시대에서 중요하기 때문에 사용자들은 자신이 필요한 정보를 더욱 쉽고, 빠르고, 편리하게 습득하려는 욕구로 웹 검색 기술이 발전하고 있다[1,2]. 정보 검색 향상을 위해 고성능 웹 크롤러의 중요성이 부각되고 있다. 인터넷에서 제공하는 수많은 웹 문서를 일정한 간격마다 자동으로 탐색하고 수집하는 기술을 웹 크롤러(Web Crawler)라 한다[3].

웹 크롤러는 구글봇(GoogleBot), 네이봇(NaBot), 폴리봇(PolyBot), 메르카토르(Mercator) 등에서 활용되고 있다. 구글 봇은 인터넷 웹 문서를 대상으로 30억 개 이상의 웹 문서 수정 주기를 분석한 후 동적과 병렬 방식으로 수집한다. 그리고 수집된 웹문서를 중복 검출 하고 저장 및 관리하는 웹 크롤러이다. 네이봇은 국내의 웹 문서들을 대상으로 URL이 지시하는 웹 문서만을 수집하고 웹 문서들로부터 발견된 새로운 URL을 추가한다. 추가된 URL로 웹을 탐색하여 웹 문서를 수집하고 저장 및 관리하는 웹 크롤러이다[4-7].

현재 상용중인 웹 크롤러는 주제별 웹 크롤러, 래퍼기반 웹 크롤러, 범용 웹 크롤러 등으로 분류하고 웹 크롤러들은 검색엔진의 상황에 따라 웹 크롤러를 선택한다. 검색엔진들은 사용자가 정보를 요청할 경우 웹 크롤러에서 가공된 데이터를 저장한 데이터베이스에서 색인과정을 통해 재구성되어 효율적으로 정보를 보여주게 된다. 이러한 정보들을 탐색·수집·가공을 하는 웹 크롤러들은 다양한 데이터 처리방식 중에서 대부분 분산처리를 기반으로 동작한다[8]. 일반적인 웹 크롤러는 분산처리 기반으로 서버와 클라이언트로 동작하고, 데이터를 임시로 저장하는 큐와 데이터를 저장 및 처리하는 다중 스레드로 구성된다.

본 논문에서 제안하는 시스템은 서버와 클라이언트로 구성되며, 서버에서의 URL 큐는 루트 URL에서 생성한 주제 URL을 저장하고 데이터 큐를 클라이언트로 받은 정보를 저장하는 큐로 분류된다. 그리고 클라이언트에서의 URL 큐는 서버로부터 받은 주제 URL을 탐색하여 선별한 URL을 저장하고, 데이터 큐는 선별한 URL을 분석한 데이터를 저장한다.

본 논문의 구성은 다음과 같다. 2장에서는 웹 크롤러와 관련 연구에 대해서 분석하고, 3장에서는 큐를 이용한 다중 스레드방식의 웹 크롤러를 설계하고 구현한다. 그리고 4장에서는 분석하고, 마지막으로 5장에서 결론을 맺는다.

2. 관련연구

웹 검색 시스템은 웹 크롤링 시스템의 기반으로 사용자가 필요한 정보를 요청할 경우 데이터베이스의 색인과정을 통해 사용자에게 정보를 제공한다. 그리고 웹 크롤링은 구성도, 구조, 처리방식으로 구성되며 웹을 자동으로 탐색·수집·가공·저장을 한다.

2.1 웹 검색 시스템

웹 검색 시스템은 사용자가 웹 서버에게 검색어를 요청할 경우 웹 서버는 검색어를 재가공하여 데이터베이스 서버에게 전송한다. 전송받은 데이터베이스서버는 분류 및 저장된 정보 중에서 색인과정을 통해 정보를 사용자에게 보여준다. 분류・저장・색인을 하는 구조는 Fig. 1와 같다[9].

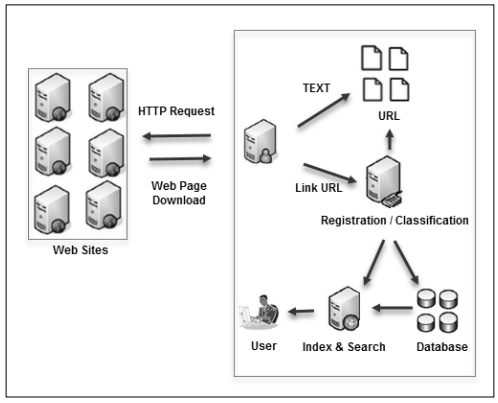

Fig. 1. Web Search System

2.2 웹 크롤러

웹 크롤러는 자동화된 방법으로 방대한량의 웹 문서를 일정한 간격으로 자동수집 및 탐색한다. 수집된 웹 문서들 중에서 사용자가 필요한 정보를 변환하여 데이터베이스에 저장한다.

2.2.1 웹 크롤러 구성도

웹 크롤러는 Monitor, Frontier, Agent, Database, Web 등으로 구성된다. 일반적인 웹 크롤러는 Fig. 2와 같은 순서로 동작한다. ①의 동작은 Frontier가 Agent에게 시드 URL을 분배한다. ②의 동작은 수신 받은 Agent는 시드 URL을 따라 웹을 탐색 및 수집한다. ③의 동작은 Agent가 수집된 결과물을 Frontier에게 전송하고, 전송받은 Frontier는 필터링 하고, Frontier에서 URL 목록을 추가한다. ④의 동작은 Frontier에서 URL 목록에 추가된 URL을 Agent에게 전송한다. 마지막으로 목록에 존재하는 URL이 사라질 때 까지 ④의 동작을 반복한다.

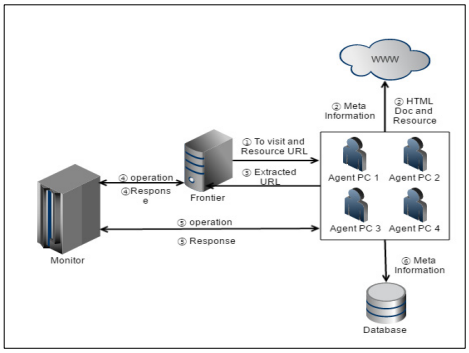

Fig. 2. Basic principles of operation of the Web crawler

2.2.2 웹 크롤러의 구조

웹 크롤러는 크롤러 매니저, 크롤러 시스템, 크롤러 어플리케이션, 하나 이상의 다운로더, DNS Resolver로 구성된다. 크롤러 매니저는 크롤러 어플리케이션으로부터 URL을 요청받아 활용 가능한 다운로더와 DNS Resolver에게 전달하여 많은 페이지를 다운로드 한다. 일반적인 웹 크롤러의 구조는 Fig. 3와 같다[11].

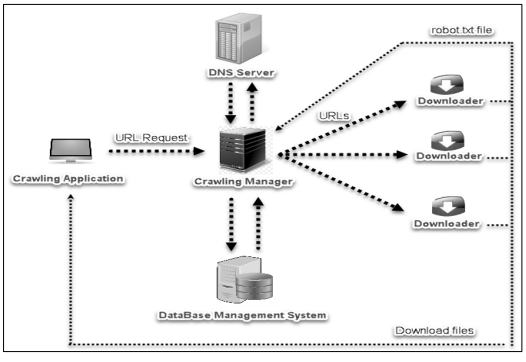

Fig. 3. General Web Crawler Structure

2.2.3 웹 크롤러 처리방식

웹 크롤러의 처리방식은 단일처리, 병렬처리, 분산처리 등이 있다. 단일처리 방식은 웹 크롤러의 초기에 설계된 방식으로 안정성이 높지만 데이터의 양이 증가함에 따라 시간의 지연도 비례하게 된다. 이러한 문제점을 보완하기 위해 병렬처리 방식이 제안되었고, 다수의 봇을 로컬에서 활용하여 시간 지연의 문제점도 점차 해결되었다. 그러나 로컬에서 다수 봇으로 인한 비용증가 및 인력낭비의 문제점이 제기된다[12].

분산처리방식은 네트워크로 연결된 에이전트와 에이전트를 관리하는 서버로 구성된다. 서버에서는 수집할 웹 페이지의 출발점인 시드(Seed URL)를 서버에 접속되어있는 에이전트에게 적절하게 분배한다. 에이전트는 서버로부터 전송받은 시드를 웹을 탐색하고 필요한 페이지의 데이터를 수집 및 가공하여 데이터베이스에 저장하게 된다. 이러한 기존의 분산처리 알고리즘의 문제점은 가능성, 중복성, 병목현상, 속도저하 등으로 제기되면서, 그에 따른 연구가 활발히 진행되어 왔다.

2.2.4 큐를 이용한 처리방식

기존의 원형 큐를 사용하는 방식은 하나의 큐에 여러 인덱스를 사용하여 데이터를 분류한다. 웹 크롤링이 완료된 데이터 중 신규로 추출된 URL은 FIndex를 사용하고 URL을 수정할 경우 RIndex를 사용한다[13]. 데이터가 방대할 경우 이러한 원형 큐를 적용하면 많은 량의 데이터로 인해 속도저하 문제가 발생할 수 있다.

3. 다중스레드 방식의 웹 크롤링 설계

본 장에서는 광역네트워크로 연결된 다수의 봇을 활용하여 단일처리방식의 시간 지연의 문제점과 병렬처리 방식의 비용증가, 인력낭비에 대한 문제점을 해결할 수 있는 다양한 분산처리방식 중에서 큐를 이용한 다중스레드방식의 웹 크롤링을 설계하고 구현한다.

3.1 시스템 설계

다중스레드 방식의 웹 크롤링 시스템 설계는 시스템 구성을 바탕으로 독립된 시스템에서 실행하는 어플리케이션을 설계한다. 다중스레드 방식은 메인 스레드에서 자식 스레드를 관리하는 방식으로 여러 개의 스레드가 각자의 독립된 공간으로 나뉘어 일련의 반복된 작업, 또

는 분산처리를 위한 기술이다. 이러한 기술을 바탕으로 JCA(Java Client Application)와 JSA(Java Server Application)에는 각자의 독립된 자식 스레드를 최소 3개에서 최대 n개로 나뉘어 작업을 실행한다. 이런 방식은 해당 PC의 사양에 따라 작업의 효율을 최대한으로 올릴 수 있는 장점이 있다.

3.1.1 시스템 구성

사용자의 관심이 높은 분야를 빠르게 검색하기 위해서 제안하는 웹 크롤러의 구성은 다음과 같다. 웹 크롤러는 JCA, JSA, Database server, Web server로 구성된다. JCA는 인터넷에서 URL 정보를 탐색, 수집한 데이터를 정형화된 정보로 가공 한 후, JSA에게 전송한다. 그리고 JSA는 JCA들을 관리하고, JCA로부터 받은 정보를 DB Server에 전송하는 역할을 한다. DB 서버는 JSA로 받은 정보를 저장하고 웹 서버로부터 요청받은 정보를 전송한다. 웹 서버는 사용자가 필요한 정보를 DB 서버에 요청하고 응답받은 정보를 가공하여 사용자에게 보여준다. 웹 크롤러의 시스템의 구성은 Fig. 4와 같다.

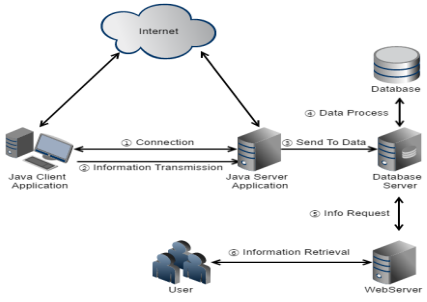

Fig. 4. System Configuration of Web crawler

3.1.2 웹 문서 탐색 알고리즘

Fig. 4에서 기존의 봇은 탐색·수집·분석 중 탐색 부분에서 마지막 페이지를 모르는 경우 무한 루프의 위험성이 높아지기 때문에 JCA에서 사용할 웹 문서 탐색 알고리즘을 제안한다. 제안한 웹 문서 탐색 알고리즘의 의사코드는 Fig. 5와 같다.

```
Int page_count;//페이지 수
Int list_data;//페이지 당 게시글 수
Int total_count;//total data count


if(total_count%list_data = 0){
 page_count = total_count / list_data;
}else{
 page_count = (total_count / list_data) + 1;
}
```

Fig. 5. Web Document Search Algorithm

JCA는 원하는 데이터를 탐색하기 위해 반복적으로 페이지를 방문해야한다. 반복적으로 페이지를 방문하기 위해서는 처음과 마지막 페이지의 수를 알아야 효율적인 알고리즘을 적용할 수 있다. 본 논문에서 제안하는 웹 문서 탐색 알고리즘은 두 가지 방식이 있다. 첫 번째, 식 ①에서 y는 전체 페이지의 데이터양이고, x는 현재 페이지의 데이터양을 의미한다. 마지막으로 나온 값을 올림한 결과 값이 z이다. 두 번째, 식 ②에서는 y와 x의 값을 계산 후, +1로 증감을 한다. 이후 버림을 통해 마지막 결과 값 z를 산출한다. 검색 알고리즘을 위해 다음 3가지의 조건을 만족해야 한다. 첫 번째, 처음 탐색할 페이지에 대한 주소, 두 번째, 탐색할 하나의 페이지에 대한 데이터양, 세 번째, 탐색할 전체 페이지의 데이터의 량이 있다. 이에 따라 3가지의 조건을 모두 만족할 경우 다음의 수식을 활용할 수 있다.

$$① \quad result: \lceil y/x \rceil = z$$

$$② \quad result: \lfloor (y/x)+1 \rfloor = z$$

3.1.3 JCA의 흐름도

Fig. 4에서 탐색·수집·분석을 하는 JCA의 흐름도는 Fig. 6와 같다.

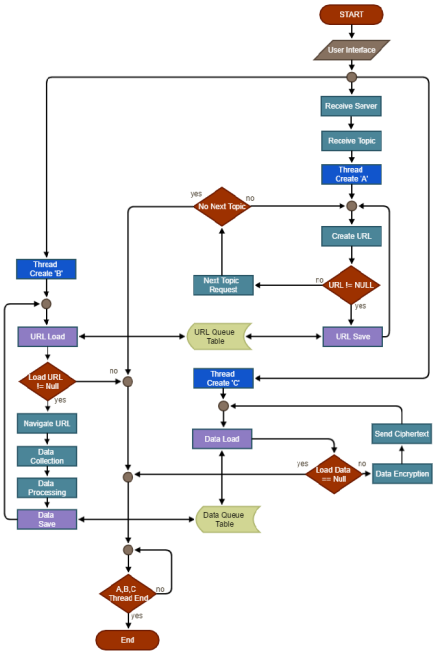

Fig. 6. Flow Chart of JCA

JCA는 스레드 B, C를 생성하고 JSA에게 연결을 요청한다. JSA로 연결이 될 경우 탐색 할 Topic URL을 전송받고 스레드 A를 생성한다. 스레드 A는 전송받은 Topic URL로 접속하여 수집할 데이터의 URL을 생성하고 URL Queue Table에 저장한다. 스레드 B는 URL Queue Table로부터 Data URL을 로드하여 탐색·수집·가공 후 Data Queue Table에 저장한다. 스레드 C는 Data Queue Table을 체크하여 null이 아닐 경우 암호화하여 JSA에게 전송한다.

3.1.4 JSA의 흐름도

Fig. 4에서 탐색·수집·분석·저장을 하는 JSA의 흐름도은 Fig. 7과 같다. JSA는 스레드 A, B, C를 생성하고 스레드 A는 Root URL을 탐색하여 Topic URL을 생성하고 URL Queue Table에 저장한다. 스레드 B는 Data Queue Table를 체크하여 null이 아닐 경우 Data를 로드 후, DB 서버에 전송한다. 스레드 C는 JCA의 접속을 기다리고, JCA가 접속할 경우 스레드 D를 생성한다. 스레드 D는 각각의 JCA들을 관리, 데이터 송/수신, JCA로부터 수신

받은 데이터 중 암호문일 경우 복호화를 한 뒤, Data Queue Table에 저장한다.

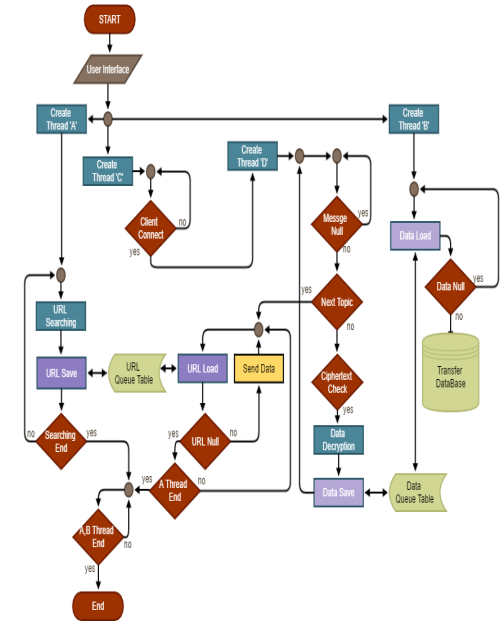

Fig. 7. Flow Chart of JSA

3.1.5 웹 크롤러의 DB 설계

본 논문에서는 개념적 설계 방식 중 하나인 ERD (Entity-Relationship Diagram)방식을 적용했다. DB는 사용자 계정 테이블 user와 웹 크롤러 데이터 테이블 data_table로 구성된다. user 테이블은 웹 크롤러 사용에 대한 JSA 검증을 통하여 보안성 및 신뢰성을 높일 수 있다. 웹 크롤러 서버는 DB의 user 테이블에서 검증을 마친 후, 봇과의 연결을 통하여 데이터를 수집한다. 이후, 수집된 데이터는 가공을 통해 정보로 변환 작업을 하고, data_table에 저장한다. user의 컬럼은 고유번호인 no, 사용자 이름의 name, 사용자 아이디의 id, 사용자 패스워드의 pw, 현재 날짜의 date로 구성되며, data_table의 컬럼은 고유번호의 no, 게시글 제목의 title, 게시글 내용의 desc, 게시글 시작 날짜의 start_date, 게시글 마지막 날짜의 end_date, 현재 날짜의 now_date, 게시글 주제의 topic로 구성된다. 데이터 다이어그램은 Fig. 8와 같다.

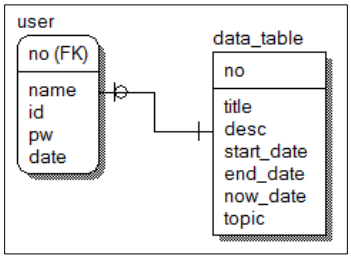

Fig. 8. Data Diagram

3.2 시스템 구현

기존의 큐를 이용한 방식은 한 개의 큐를 사용하여 데이터를 저장하고 관리한다. 본 논문에서 제안하는 웹 크롤링은 데이터 큐와 URL 큐로 분리한다. 데이터를 처리하기 위한 다중 스레드에서는 데이터 큐와 URL 큐를 공유하여 사용한다.

3.2.1 시스템 구현환경

시스템의 구현환경은 어플리케이션, OS, Basic Application으로 구성된다. 어플리케이션은 DB Server, Web Server, JSA(Server), JCA(Bot)으로 구성되고, Windows 10 Pro라는 OS를 사용하고, Basic Application으로는 Apache 2.2.1, PHP, MySQL, Java를 이용한다. 자세한 시스템의 구현환경은 Table 1과 같다.

Table 1. System Implementation Environment

System Performance by Program				
App	DB Server Web Server		CPU	Intel(R) Core(TM) i7-4790
			RAM	8.00GB
	JSA		CPU	Intel(R) Core(TM)2 Duo
			RAM	4.00GB
	JCA	1	CPU	Intel(R) Core(TM) i5-6300U
			RAM	8.00GB
		2	CPU	Intel(R) Core(TM) i5-2320
			RAM	4.00GB
OS	Windows 10 Pro			
Basic Application				
Apache		PHP	MySQL	Java

3.2.2 JSA UI 구현

JSA는 SeedURL을 기반으로 웹을 탐색하여 Topic URL을 선별한 후 URL 큐에 저장한다. JCA와 “Client Connection”을 성공적으로 한 경우, URL 큐에 저장되어 있는 Topic URL을 JCA에게 전송하고 대기한다. 이후 JCA로부터 가공된 정보를 전송받아 데이터 큐에 저장하고 데이터베이스 서버로 전송한다. JSA의 의사코드는 Fig. 9과 같다.

```
String Seed URL;
While () {
      Seed URL Search Function();
      set Topic URL to URL Queue;
     if Client Connect() {
              URL queue to Client send;
     }
     if Client Response() {
              set Client Data to Data Queue;
              Database Server send;
     }

}
```

Fig. 9. Pseudo code of JSA

Fig. 10은 JSA UI의 실행과 구현 결과를 나타낸 것이다. 실행 순서에서 ①은 관리자를 인증하기 위한 부분으로 id 텍스트 박스에 관리자의 id를 입력하고 pw 텍스트 박스에 관리자의 pw를 입력한다. ②은 JSA 관리자가 Root URL을 선택한 후, START 버튼을 클릭하면 JSA가 접속한 JCA에게 URL을 분배한다. ③은 JCA가 JSA에게 접속한 것을 실시간으로 JSA 관리자에게 보여준다. ④은 START 버튼을 클릭하였을 경우, 현 시점부터 경과되는 시간을 보여준다. ⑤는 JCA가 JSA에게 전송한 데이터를 DB 서버에 저장이 완료되었을 경우에 횟수가 올라간다. ⑥는 JSA와 JCA에 대한 실시간 로그를 보여준다. 위와 같은 과정을 거쳐 웹 크롤링 시스템 작업을 실행한다.

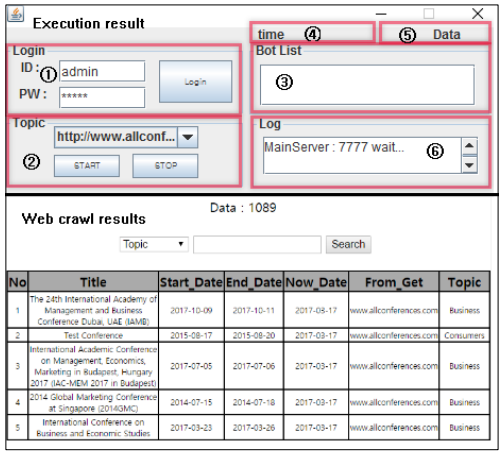

Fig. 10. Connection and connection result of JSA

3.2.3 JCA UI 구현

JCA는 “클라이언트 고유번호”를 생성 한 후, JSA 서버와 연결을 한다. 정상적으로 연결한 경우, JSA 서버로부터 “Topic URL”을 전송받고, URL 큐에 저장한다. 이후 URL 큐를 기반으로 웹을 탐색 및 수집하고 분석한다. 분석된 정보는 데이터 큐에 저장하여 JSA 서버로 전송한다. JCA의 의사코드는 Fig. 11과 같다.

```
set client unique number;

if Server Connection() {
     send unique number to server;

     While() {
      set Topic URL to URL Queue;
              if Topic URL Count > 0 {
                 set Web Searching;
                 set Analysis;
              }
              if Analysis() > 0 {
                set Data to Data Queue;
                send Data Queue to server;
              }
     }
}
```

Fig. 11. Pseudo code of JCA

Fig. 12은 JCA UI의 구현 결과를 나타낸 것이다. 실행 순서에서 ①은 JCA에서 사용할 스레드의 개수를 정하고, 다른 옵션을 선택하여 JSA에게 접속 요청을 한다. ②은 JCA가 JSA에게 성공적으로 연결이 되었을 경우, 메시지와 함께 현재 JCA의 고유번호가 출력된다.

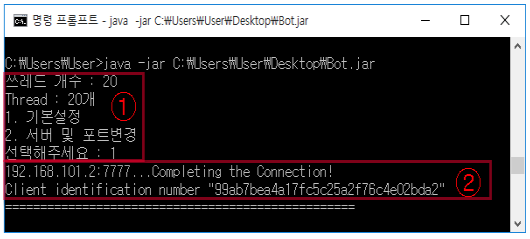

Fig. 12. Execution result of JCA UI

4. 시스템 분석

기존의 웹 크롤링 방식은 대부분 분산처리 방식으로 동작한다. 분산처리 방식은 데이터 중복성 문제, 가능성 문제, 병목현상 문제 등의 원인으로 인해 속도의 저하현상이 있다. 이러한 문제점들 중 병목현상의 속도 문제에 대한 오버헤드를 최소한으로 줄일 수 있는 다중 스레드 방식을 이용하여 시스템을 설계 및 구현하여 분석한다. 이후 분석 단계에서는 상황별로 속도 차이점들을 시간대별로 실험했다.

JCA의 시간당 처리량과 다중 스레드의 시간당 처리량을 통하여 시간대별 수치를 분석했다. 이러한 수치는 웹 크롤러 스레드 개수에 비례하여 JCA 수가 많을수록 속도와 처리량은 높지만 JCA의 사양과 스레드 개수가 비슷할 경우, 처리량의 차이는 별 차이가 없다. 스레드 1의 JCA 3과 스레드 1의 JCA 4는 5.85%(± 2) 정도의 차이 값이 존재한다. 이와 같은 계산은 다음의 수식을 활용해서 값을 확인할 수 있다. x의 값은 x >= y, y의 값은 x <= y을 대입한 후, z라는 전체 값으로 평균한 것이다.

$$result = ((y-x)/z)*100$$

위 수식은 JCA의 시간당 처리량과 다중 스레드의 시간당 처리량에서 동일하게 사용한다. 데이터양은 Fig. 13과 같다.

JCA의 시간당 처리량과 다중 스레드의 시간당 처리량을 비교 분석한다. 먼저, 스레드가 1개일 경우, 각각의 독립된 JCA의 개수가 1대부터 4대까지의 전체 처리량은 3.40%(± 2)~4.39(± 2)%정도가 측정된다. 이후 수시간당 처리량을 보면 JCA 1, 2과 JCA 2, 3은 0.14%의 수치로 가장 작은 처리량을 보였고, JCA 2, 3과 JCA 3, 4은 0.85%로 처리량이 큰 값을 보였다. 이러한 결과를 통해 JCA 3, 4에서 최대의 효율성이 나온 것을 확인 할 수 있다. 스레드의 시간당 처리량을 비교를 통해 작업의 효율성을 확인 할 수 있다. 효율이 높은 순으로 정리하면 JCA 1의 경우 스레드 개수는 5~10개, JCA 2, 3의 경우 스레드 개수는 1개~5개, JCA 4의 경우 스레드 개수는 10~20개 정도에서 최대의 효과가 나온 것을 수치상으로 확인할 수 있다.

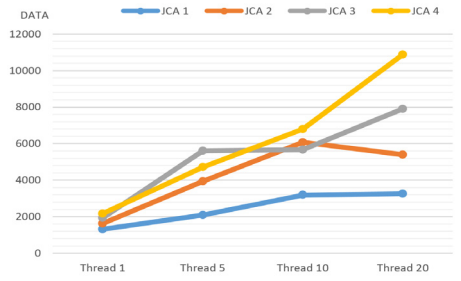

Fig. 13. Thread of JCA

처리량이 100 이하의 JCA 1(Thread 10, 20)과 JCA 3(Thread 5, 10)은 앞의 Thread 처리량과 큰 수치 차이가 없으며 처리량의 효율성을 확인하기가 어렵다. 분석결과에서 JCA 1은 Thread 5, 10일 때와 JCA 3는 Thread 1, 5일 때 처리량이 높아진다. 이와 같은 처리량은 다음의 수식을 활용해서 값을 확인할 수 있다.

$$|(x-y)| = z$$

x의 값은 x >= y, y의 값은 x <= y을 대입한 후, z라는 절댓값으로 계산한 것이다. JCA 별 Thread의 테이블은 Table 2과 같다.

Table 2. Thread of JCA

JCA / Thread	JCA 1		JCA 2	
	Throughput (Total)	statistic (%)	Throughput (Total)	statistic (%)
Thread 1,5	770 (2090)	7.80	2310 (3940)	13.55
Thread 5,10	1100 (3190)	11.15	2130 (6070)	12.50
Thread 10,20	70 (3260)	0.70	670 (5400)	3.93
JCA / Thread	JCA 3		JCA 4	
	Throughput (Total)	statistic (%)	Throughput (Total)	statistic (%)
Thread 1,5	3680 (5610)	17.43	2550 (4720)	10.37
Thread 5,10	60(5670)	0.28	2080 (6800)	8.46
Thread 10,20	2230 (7900)	10.56	4080(10880)	16.60

JCA의 수와 스레드 개수의 관계는 서로 비례하지만, 오버헤드 부분이 발생되는 구간에서는 차이점이 크게 나타났다. 수많은 JCA를 사용할 경우, 속도 처리의 효율성은 높아지게 된다. 이에 따라 비용에 대한 문제도 높아지므로, 상황에 맞게 적절한 JCA를 사용할 경우는 속도처리 부분에서 업무의 효율성이 다소 떨어지게 된다. 이러한 속도 부분 문제와 비용에 대한 문제의 해결 방안으로 독립적인 JCA를 추가하는 것 보다 스레드의 개수를 추가하여 업무의 효율성을 수많은 JCA를 사용하는 경우와 비슷하게 구현할 수 있다.

또한 JCA들의 사양에 맞게 스레드 개수의 변경이 가능하므로 사양에 대한 선택의 폭도 넓어지게 된다. 이에 따라 검색엔진을 서비스하는 여러 다양한 기업들의 경우 천문학적인 많은 비용을 지출하는 것보다 적절하게 JCA를 사용하는 것을 선택할 수 도 있다.

5. 결론

본 논문에서는 큐를 이용하여 다중 스레드 방식의 웹 크롤러 기법을 제안했다. 또한, 웹 문서의 처리량을 수식에 따라 JCA와 스레드 별로 나누어 분석하고, 각각 효율성 비교를 통해 최적의 JCA의 개수와 스레드의 개수를 확인 했다. 제안하는 웹 크롤러의 설계 방식은 분산처리를 기반으로 각각의 독립된 환경에서의 JCA는 큐와 스레드를 이용하여 빠르고 신뢰성이 높은 웹 문서를 제공한다.

향후에는 특정 사이트를 대상으로 하는 웹 크롤러 설계가 아닌 범용 웹 크롤러에 큐와 다중 스레드를 적용하여 다양한 웹 사이트를 빠르고 효율적으로 탐색, 수집 그리고 분석한 결과를 빅 데이터 응용 분야에 활용될 수 있을 것이다.

ACKNOWLEDGMENTS

This works was supported by the National Research Foundation of Korea Grant Funded by the Korea Government(NRF-2014S1A5B6035600)

REFERENCES

[1] D. M. Seo and H. M. Jung, “Intelligent Web Crawler for Supporting Big Data Analysis Services,” *Journal of the Korea Contents Association*, Vol. 13, No. 12, pp. 575-584, Dec. 2013. DOI: 10.5392/JKCA.2013.13.12.575

[2] H. W. Kim and Y. S. Han, “Web Crawler Design for the ARANES Search Engine,” *Korean Society For Internet Information*, Vol. 2, No. 1, pp. 294-299. May. 2001.

[3] K. Y. Kim, W. G. Lee, H. M. Yoon, S. H. Shin and M. . Lee, “Development of Web Crawler for Archiving Web Resources,” *Journal of the Korea Contents Association*, Vol. 11, No. 9, pp. 9-16, Sep. 2011. DOI: 10.5392/JKCA.2011.11.9.009

[4] D. Y. Kim and J. T. Kim, “Efficient Design of Web Searching Robot Engine Using Distributed Processing Method with Javascript Function,” *Journal of the Korea Institute of Information and Communication Engineering*, Vol. 13, No. 12, pp. 2595-2602, Dec. 2009.

[5] K. H. Kim and J. H. Lee, “A Methodology for Performance Evaluation of Web Robots,” *Information Processing Society*, Vol. 11, No. 3, pp. 563-570, 2006.

[6] A. Heydon and M. Najork, “Mercator : A Scalable : Extensible Web Crawler,” *Journal World Wide Web*, Vol. 2, Issue. 4, 1999. DOI: 10.1023/A:1019213109274

[7] M. Najork and A. Heydon, *High-Performance Web Crawling*, SRC Research Report 173, Compaq Systems Research Center, 2001.

[8] W. S. Cho, J. E. Lee and C. H. Choi, “Refresh Cycle Optimization for Web Crawlers,” *Journal of the Korea Contents Association*, Vol. 13, No. 6, pp. 30-39, Jun. 2013. DOI: 10.5392/JKCA.2013.13.06.030

[9] M. S. Choi, *A Study on the Improvement of the Web-Crawler Performance based on Weighted Directed Graph*, Department of Computer Science, Graduate School, Kyungpook National University, 2010.

[10] D. K. Jung and S. G. Min, “A study of Client Response Time Collection Method by changing the Dynamic HTML in the Web Application Server System,” *Korea Information Science Society*, Vol. 39, No. 1, pp. 305-307, Jun. 2012.

[11] H. H. Kim, Y. W. Kim and P. W. Lee, “A Method of GridIR System Configuration over Distributed Experiment of Web Crawler,” *Korean Society For Internet Information*, Vol. 8, No. 2, pp. 281-286, Nov. 2007.

[12] H. Garcia-Molina and J. cho, “Parallel Crawler,” *Proceedings of the 11th International World Wide Web Conference*, pp. 124-135, 2002. DOI: 10.1145/511446.511464

[13] H. C. Kim and S. H. Chae, “Design and Implementation of a High Performance Web Crawler,” *Journal of Digital Contents Society*, Vol. 4, No. 2, pp. 127-137, 2003.

저 자 소 개

김 효 종(Hyo-Jong Kim) [정회원]

- 2016년 2월 : 동명대학교 정보보호학과(공학사)
- 2017년 ~ 현재 : 동명대학교 일반대학원 정보보호학과 석사과정

<관심분야> : 웹 크롤링, IoT, 빅데이터

이 준 연(Jun-Yun Lee) [정회원]

- 1990년 8월 : 중앙대학교 컴퓨터공학과(공학사)
- 1992년 8월 : 중앙대학교 컴퓨터공학과(공학석사)
- 2000년 2월 : 중앙대학교 컴퓨터공학과(공학박사)
- 2000년 3월 ~ 현재 : 동명대학교 미디어공학과 교수

<관심분야> : 클라우드 컴퓨팅, IoT, IaaS, ITS

신 승 수(Seung-Soo Shin) [정회원]

- 2001년 2월 : 충북대학교 수학과(이학박사)
- 2004년 8월 : 충북대학교 컴퓨터공학과(공학박사)
- 2005년 3월 ~ 현재 : 동명대학교 정보보호학과 교수

<관심분야> : 암호프로토콜, 무선 PKI, 네트워크 보안, U-헬스 케어, IoT