

서버 부하를 고려한 동적 로봇에이전트 시스템의 설계 및 구현

박 규 석[†] · 이 충 석^{††} · 김 성^{††}

요 약

급속한 인터넷 사이트와 이용자들의 증가로 인해 이용자들의 요구에 부응하는 정보를 제공하기 위한 검색 엔진들의 개발이 가속화되고 있다. 이로 인해 많은 검색엔진들의 문서 수집 활동으로 인한 대상 호스트들에 대한 부하가 초래되었고, 방대한 양의 새로운 정보가 추가되어 가는 상황에서 모든 정보의 주기적인 갱신이 필요하게 되었다.

이러한 시대적인 상황과 검색시스템의 기본요건인 빠른 속도와 정확한 정보 수집을 이루기 위하여 방대한 호스트내의 문서를 수집할 수 있는 기술의 필요성이 높아졌으며, 이용자들의 다양한 요구와 WWW(World Wide Web)간의 상호 유기적인 작용을 위한 검색 엔진의 역할이 더욱 커지고 있다. 본 연구에서는 기존에 제시되었던 로봇 에이전트 시스템에 대한 비교 분석 후 이들을 보완하여 대상 서버에 대한 부하량 측정치와 로봇이 수집 활동시에 체험한 부하량을 함께 고려하여 문서수집이 빠른시간 내에 이루어지며 대상 서버에 과도한 부하를 주지 않는 로봇 에이전트와 원격 관리 시스템을 설계 및 구현한다.

Design and Implementation of a Dynamic Robot Agent System Considering the Server's Workload

Kyoo-Seok Park[†] · Chung-Seok Lee^{††} · Sung Kim^{††}

ABSTRACT

As the Internet sites and users have rapidly been increased, the development for search engines has also been accelerated to satisfy users' expectations. As the result, not only the action of collecting documents through many search engines gave hosts workload, but also regular updating all the information is needed since information is newly added.

With the circumstances, the necessity of the technology to collect massive information in hosts has been increased for the speed which is a basic requisite of search systems, and for more accurate collection of documents. Also, the role of search engines grows bigger for Internet users' various demands and flexible process through World Wide Web.

In this paper, we design and implement a robot agent and a remote control system which doesn't give an excessive workload on a target server and makes the collection of documents done in a short period by considering an average workload rate on the target server and the rate of the workload that a robot experience in collection time, after we compare and analyze the existing Robot Agent Systems and supplement their weak points.

1. 서 론

인터넷의 발전과 더불어, 검색엔진의 발전 또한 커

다란 변화를 가져왔다. 인터넷이 처음 등장할 당시에는 텍스트 위주의 정보교환이나 의사 전달과 같은 커뮤니케이션 기능위주이었던 때문에 검색 엔진 또한 단순히 문서를 수집해서 분류한 후 실시간으로 제공하는 것이 주된 역할이었다. 그러나 현재는 인터넷기술의 급격한 발전과 이용자들의 증가 및 다양한 요구로 인

[†] 정 회 원 : 경남대학교 컴퓨터공학과 교수
^{††} 준 회 원 : 경남대학교 대학원 컴퓨터공학과
논문접수 : 2000년 11월 2일, 심사완료 : 2001년 1월 5일

해서 실시간 정보 제공과, 이용자의 특성을 분석한 후 이용자가 원하는 정보를 카테고리별로 서비스 할 수 있으며, 네트워크 부하가 최대한 고려된 검색엔진이 필요하게 되었다.

본 논문에서는 인터넷을 이용한 사용자의 특성에 맞는 인터넷 정보 시스템 모델을 제안하고, 웹을 통해 원격 제어가 가능한 실시간 제어 검색엔진과 플랫폼에 독립적인 로봇 에이전트를 설계 및 구현하였다. 따라서 제안 시스템은 분산되어 있는 정보를 쉽게 모아서 갱신해 줄 수 있는 지능적인 로봇의 구현과 주기적으로 네트워크의 부하를 모니터링하고 그 결과를 기반으로 전체 시스템의 성능을 향상시킬 수 있는 실시간 로봇 제어 정책이 가능하다.

본 논문의 구성은 다음과 같다. 2장에서는 검색 기법에 관련된 연구를 살펴보고, 3장에서는 네트워크 부하가 고려된 원격 제어가 가능한 실시간 제어 검색엔진 시스템을 설계하며, 4장에서는 구현 및 평가, 5장에서는 결론 및 향후 연구방향을 제시한다.

2. 관련연구

2.1 HTTP

HTTP(Hyper Text Transfer Protocol)는 분산 환경 및 공동작업 환경에 이용할 하이퍼미디어 정보시스템의 개발을 목적으로 설계된 응용계층의 프로토콜로서 웹(WWW-World Wide Web)에서의 하이퍼텍스트 문서의 전송을 위해 쓰인다[1].

실제적인 정보시스템에서는 검색, 정보 갱신 및 주석 등과 같은 간단한 작업보다는 더욱 다양한 기능을 필요로 하고 있다. 또한 분산 환경, 공동작업 환경, 하이퍼미디어 정보시스템 등의 상황에서 필요로 하는 빠른 속도와 간편성을 제공하기 위한 응용계층의 프로토콜이 필요하게 되었고, 이를 목적으로 HTTP 프로토콜이 설계되었다.

2.2 로봇 에이전트

로봇 에이전트란 원하는 정보를 얻기 위해 웹 상의 문서들을 검색하고 참조되는 문서들을 재귀적으로 검색하면서 웹의 하이퍼텍스트 구조를 추적하여 정보를 저장해 주는 프로그램이다[2].

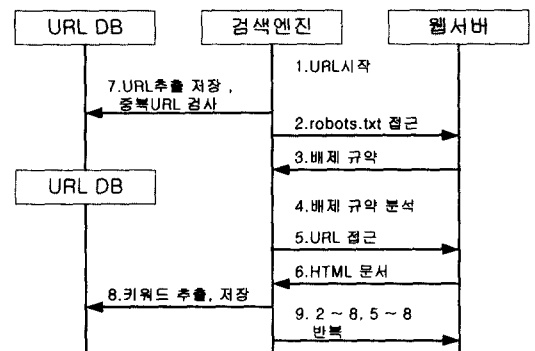
인터넷에서 정보 검색 서비스를 제공하려면 일단 웹 문서를 수집해야 하는데, 웹 문서를 어떻게 수집하느냐에 따라 검색 결과도 크게 달라진다.

로봇의 순회 방법에 따라 넓이-우선 순회(breadth-first traversal)와 깊이-우선 순회(depth-first traversal)로 나누어 볼 수 있다.

로봇들은 통계분석, 유지보수, 미러링(mirroring), 리소스 발견, 복합적인 사용 등에 이용되며, 상대 시스템에 과도한 부하를 부여, 로봇 프로세스의 동작 오류, URL 검사시의 로봇 성능 저하, 인덱싱 결과의 관리 등과 같은 문제들을 발생시킬 수도 있다.

상대 시스템의 과부하를 막기 위한 방법으로 페이지 요구문 사이에 지연문을 넣거나 robots.txt 파일을 참고하여 로봇 배제 규약을 준수하도록 하고 있다.

로봇의 동작 알고리즘은 (그림 1)과 같다.



(그림 1) 로봇 동작 알고리즘

2.3 로봇 혼합 정렬 스케줄러

현재 기존의 검색 시스템의 경우 문서를 수집하기 위해 특정한 로봇 스케줄링기법을 사용하지 않고 등록된 순서대로 로봇을 보내어 문서를 수집하고 있다. 하지만 이 방법의 가장 큰 문제는 해당 서버의 네트워크 부하를 고려하지 않기 때문에 부하가 많이 발생하고 있는 시점에도 로봇이 문서를 수집하는 경우가 발생한다. 이로 인하여 로봇의 성능이 저하된다.

이러한 문제를 보완한 혼합정렬 스케줄러는 정렬스케줄러에 의해서 서버관리기로부터 생성된 서버 리스트와 최적 부하시간 탐색기의 결과로 나온 리스트를 결합하여 우선순위 테이블을 생성한다[2].

2.4 언어 분석 기술

인터넷 공간의 정보는 멀티미디어 데이터로 되어 있지만 정보검색의 초점은 주로 문자로 표현된 언어 정

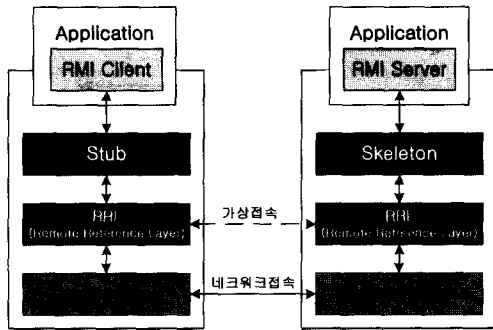
보에 맞춰져 있다. 일반적으로 정보검색은 사용자의 질의를 분석하여 사용자가 원하는 정보를 찾아주는 함수라고 할 수 있다[1].

사용자 질의 속에 담긴 의도를 알아내기 위하여 정보검색 시스템은 질의의 구성과 핵심어의 의미 등을 분석할 수 있어야 하고, 검색 대상이 되는 정보가 무엇에 관한 것이고 사용자의 의도와 얼마나 일치하는지를 판단할 수 있어야 한다.

언어처리는 문장이나 구절 형태로 주어진 텍스트로부터 가능한 많은 정보를 추출하는 것을 목표로 하는데, 그 형태는 단어나 구절의 처리에서부터 대응어, 불용어 처리에 이르기까지 그 수준이 다양하다. 그러나 최근의 웹 정보검색엔진에서는 언어처리를 위한 알고리즘 개발보다는 사전 및 지식정보에 의존하는 경향이 강세를 이루고 있다[3].

2.5 RMI

RMI(Remote Method Invocation)를 이용하여 클라이언트 컴퓨터에서 실행되는 프로그램이 원격서버 컴퓨터에 위치한 객체의 메소드를 호출할 수 있다.[4]



(그림 2) RMI 구조도

RMI는 (그림 2)에서와 같이 Stub/Skeleton, RRL, Transport Layer의 세 계층으로 구성된다.

Stub은 원격 객체를 대변하는 클라이언트 측 대리자로서, 원격 객체가 구현하는 인터페이스들을 모두 정의하며, 프로그램 내에서 지역적인 다른 객체처럼 참조되어 서버 측 객체의 접속을 유지한다.

원격 객체가 호출되면 Stub은 파라미터 데이터를 직렬화 시킨 후 정렬 스트림(Marshal stream)형태로 원격참조계층으로 전달한 후, 원격 메소드 수행이 끝나면 다시 원격참조계층으로부터 정렬스트림을 받아 역

직렬화를 통해 값을 리턴한다. 정렬 스트림은 서버 측의 RRL(Remote Reference Layer : 원격참조 계층)과 클라이언트 측의 RRL과의 통신에 사용된다.

Skeleton은 서버 측 RRL과 인터페이스하는 서버측의 생성체로서 클라이언트 측 RRL로 부터 메소드 호출 요청을 받으면 서버 측 RRL은 원격 메소드로 전송되어온 파라미터들을 복원시킨 후 Skeleton이 서버 측에 구현된 실제 객체를 호출한다. 원격 객체에서 리턴 값을 받아 다시 정렬 스트림으로 정렬 후 서버측의 RRL에 전달한다.

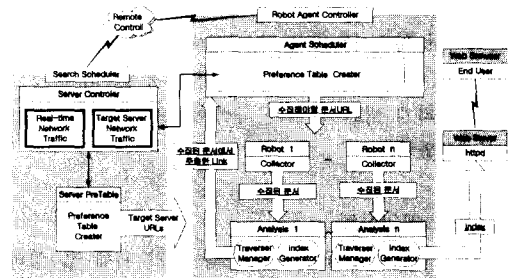
원격참조계층은 특정한 Stub 또는 Skeleton 모델에 종속적이지 않은 독립 참조 프로토콜을 관리하여 다른 두 계층에 영향을 주지 않고 RRL교체가 가능하다. 하위 운송 인터페이스와 작용하여 Stub과 Skeleton 계층에 스트림을 제공한다.

전송계층은 클라이언트와 서버사이의 접속을 만들고 기존의 접속을 관리하며, 자신의 주소 공간에 상주하는 원격 객체를 처리한다. 전송계층이 클라이언트측 RRL에서의 요청을 접수하면 원격 객체에 적합한 RMI 서버를 검색하여 그 서버에 대한 소켓을 만들고 이 접속을 클라이언트 측 RRL에 전달한후 내부 테이블에 그 원격 객체의 참조를 추가시켜 연결을 확립한다.

3. 시스템 설계

3.1 시스템 구성

제안하는 정보 검색 시스템의 구성도는 (그림 3)과 같다.



(그림 3) 시스템 구성도

Search Scheduler는 정보검색 작업을 위한 수집대상 서버 URL들을 관리하고 이를 효율적인 시간대에 수집 작업을 수행할 수 있도록 스케줄링하고, Robot Agent

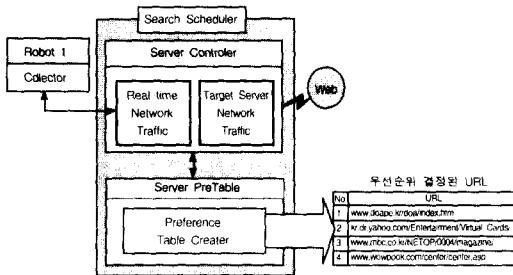
Scheduler는 수집 대상 서버 URL을 받아 실제 적인 수집작업을 수행하며, Web Server는 수집 작업의 결과를 End User에게 서비스 해준다.

Search Scheduler는 수집 대상 서버들의 URL을 가지고 시간대별 부하량을 측정하는 서버관리기와 작업 수행시 가장 효율적인 대상들을 선택해주는 우선 순위 테이블 생성기로 구성된다.

Robot Agent Scheduler는 우선 순위 테이블 생성기와 우선순위에 따라 재구성된 검색대상 URL의 문서를 수집하는 로봇의 관리를 맡는 로봇관리기(Robot Controller), 그 수집된 문서를 분석하는 문서분석기(Analysis), 그리고 RMI를 이용한 원격관리 시스템으로 구성되며 이 원격관리 시스템에 의해 통괄 관리되며 수시로 변하는 상황에 민첩한 대처를 해나갈 수 있도록 서비스되어진다.

3.2 Search Scheduler

Search Scheduler의 구성도는 (그림 4)와 같다.



(그림 4) Search Scheduler 구성도

Search Scheduler는 서버관리기와 우선 순위 테이블 생성기로 구성되며, 서버관리기는 다수의 웹서버에 대한 정보를 관리하는 모듈로서 웹서버마다의 시간대별 상황 및 실시간으로 로봇이 체한 부하량을 체크한다. Search Scheduler의 처리 알고리즘은 (그림 5)와 같다.

웹서버들은 서버관리기에 의해 매시간 Ping을 통한 시간대별 부하량이 측정되어 그 값을 매번 평균값으로 저장 하게 되며 문서 수집시 설정된 값과 함께 점수가 부여되게 된다. 이는 평균화된 판단의 근거를 마련하기 위한 한 방편으로 검색대상 URL의 우선 순위 결정시 사용된다.

서버들의 점수부여 기준은 서버관리기에 의한 시간

대별 부하량을 70%, 로봇이 실시간으로 체한 부하량에 대하여는 30%의 가중치를 부여하여 시간대별 점수를 계산하여 가장 효율적인 시간대에 대하여 우선권을 높게 주는 방법으로 각 서버들마다 효율적인 시간대가 하나씩 존재할 수 있도록 설정하여 시간대별 효율적인 서버에 대한 리스트를 제공한다.

```
// 수집동작을 대기하는 수가 없을시 최초 시작할 URL을
// 생성한다.
if( count == 0 ) {
    makeStartUrl();
    continue;
}

//검색대상 URL들 모두 POP
for( int i=0 ; i<count ; i++ )
{
    _aUrl[i]=new CUrl(_pCtr._LOGEnable);
    _aUrl[i].pop(_pDB.getConnection(),_pCtr._iReadyDB);
    // 우선순위 테이블 생성
    _aUrl=_pTabelCreat.MakeTable(_aUrl);
    // 우선순위 결정된 URL리스트 생성
    for(int i=0;i<_aUrl.length;i++) {
        _aUrl[i].push(_pDB.getConnection(),_pCtr._iGoingDB);
    }
}
}
```

(그림 5) Search Scheduler 알고리즘

서버관리기에서 제공된 서버들의 정보를 가지고 우선 순위 테이블 생성기는 (그림 6)에서와 같이 각 서버들의 목록 중 해당 시간대에 가장 효율적인 서버들의 목록을 선택하고 그 서버들의 점수에 따라 우선 순위로 정렬된 리스트를 제공한다.

제공되어진 리스트는 순차적으로 수집동작에 들어가게 된다.

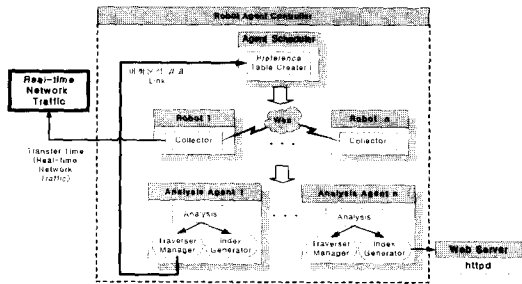
검색 대상 URL				선택된 대상 URL			
No	URL	속도	부하량	No	URL	속도	부하량
1	203.253.173.203/~/index.html	185	700	1	http://www.webbook.com/~/index.html	185	700
2	www.webbook.com/~/index.html	185	700	2	http://www.webbook.com/~/index.html	185	700
3	http://www.webbook.com/~/index.html	185	700	3	203.253.173.203/~/index.html	185	700
4	www.webbook.com/~/index.html	185	700	4	www.webbook.com/~/index.html	185	700
5	www.webbook.com/~/index.html	185	700	5	www.webbook.com/~/index.html	185	700
6	www.webbook.com/~/index.html	185	700	6	www.webbook.com/~/index.html	185	700
7	www.webbook.com/~/index.html	185	700	7	www.webbook.com/~/index.html	185	700
8	www.webbook.com/~/index.html	185	700	8	www.webbook.com/~/index.html	185	700

우선순위 결정된 URL			
No	URL	속도	부하량
1	www.webbook.com/~/index.html	185	700
2	www.webbook.com/~/index.html	185	700
3	www.webbook.com/~/index.html	185	700
4	www.webbook.com/~/index.html	185	700
5	www.webbook.com/~/index.html	185	700
6	www.webbook.com/~/index.html	185	700
7	www.webbook.com/~/index.html	185	700
8	www.webbook.com/~/index.html	185	700

(그림 6) 우선 순위 결정도

3.3 Robot Agent Controller

Robot Agent Controller의 구성도는 (그림 7)과 같다.



(그림 7) Robot Agent Controller 구성도

Robot Agent Controller는 우선 순위 테이블 생성기와 로봇관리기, 문서분석기로 구성되며 Search Scheduler에서 넘어온 수집 대상 서버들의 URL을 중심으로 실제적인 수집 작업을 수행한다.

우선 순위 테이블 생성기는 대상 서버들의 정보를 바탕으로 해당 시간대에 가장 효율적인 서버URL내의 링크를 먼저 제공해 준다. 생성된 우선 순위 테이블을 기준으로 최상위 URL부터 n개의 로봇을 생성하여 동시에 문서수집을 수행한다. 로봇의 개수 및 동작시간은 RMI를 이용한 원격관리 모듈에 의해 설정하며, 실시간에 관리자가 설정을 변경하여 네트워크의 과도한 부하를 방지할 수 있다.

문서분석기는 로봇이 대상 URL에서 가져온 문서를 분석하여 다른곳에 Link되어 있는 URL을 추출하여 스케줄러에게 넘겨준다. 검색대상 웹서버의 Depth설정으로 무한 재귀 루프 상태가 되지 않도록 규제한다, 그리고 이미 수집한 URL과 검색대상 웹서버를 벗어나는 URL에 대해서는 추출시 버린다[5]. 단, 삭제 작업은 원격관리모듈 및 환경파일의 설정을 변경 가능하게 구현하여 특성을 부가한 웹 검색을 가능하게 하여 특정 네트워크별, 또는 특정 사이트들을 대상으로 하는 기업형 검색엔진으로서의 기능을 가지게 한다.

대상 URL추출 및 HTML TAG제거후 어휘분석기를 통해 색인어와 불용어에 대한 분류와 인덱스DB를 생성한다.

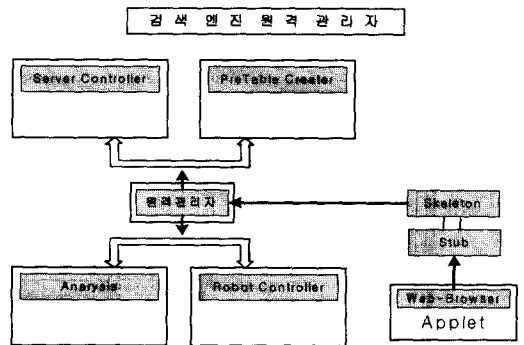
3.4 질의서버 와 등록서버

질의서버는 일반사용자가 웹 브라우저를 통해 특정 용어에 대한 검색질의시 인덱스 DB 및 디렉토리 DB에 미리 분류 저장되어 있는 자료를 일반사용자에게 보여준다. 등록서버는 디렉토리DB생성을 담당하여 자신의 홈페이지를 홍보하고자 하는 일반사용자들의 등

록품을 생성하며 필요한 정보를 얻어 이를 이용한 디렉토리DB를 생성한다. 생성된 디렉토리정보의 사실 유무 확인을 위해 대상 URL은 스케줄러를 통해 n번의 상태확인을 거쳐 확인된 디렉토리에 대한 정보를 관리하고 사용자에게 디렉토리의 불량여부를 통보하기 위한 정보를 생성한다.

3.5 원격관리

검색시스템의 관리를 위해 시스템 제반설정사항을 RMI서버 모듈로 구성하여 모듈 내 변수 값의 변경을 통해 실시간으로 변경사항을 적용할 수 있도록 구성한다. 이는 시스템의 재시작이 필요 없이 특수한 상황 발생시 제반설정사항을 원격으로 변경할 수 있는 이점을 제공한다. 원격관리모듈은 Applet으로 구성하여 웹 브라우저를 이용한 검색엔진 관리를 가능하게 한다. (그림 8)은 원격관리의 제어 흐름을 표현한 것이다.



(그림 8) RMI를 이용한 원격관리자 흐름도

자바의 RMI를 이용하여 웹상의 Applet에서의 설정 값 변경시 검색시스템내의 설정 모듈의 함수가 호출, 수행되어 검색시스템내의 설정값을 변경, 실시간으로 적용할 수 있다.

RMI를 이용한 원격관리 구현 알고리즘은 (그림 9)와 같다.

```

※ 관리자 측
try {
    ControlImpl obj
        = (ControlImpl) Naming.lookup("//ControlServer");
} catch (Exception e) {
    _txtHelp.setText("원격 관리 접속에 실패하였습니다 ... "+
        e.toString());
}
    
```

```

* 검색엔진 측
try {
    // RMI-Registry 생성
    LocateRegistry.createRegistry(
        (new Integer(registryPort)).intValue());
    ControlServer pServer = new ControlServer();

    // bind the server to the registry
    Naming.rebind("://" + registryPort +
        "/ControlServer", pServer);
} catch (Exception e) {
    System.out.println("ERROR : " + e.getMessage());
}
    
```

(그림 9) 원격관리 알고리즘

RMIRegistry를 생성하고 원격관리자를 바인딩시킨 후 원격관리 모듈의 접속이 있을때까지 대기한다. 원격관리모듈의 접속시 시스템의 자바보안관리자의 인증을 통해 원격관리모듈은 원격관리자의 내부 함수를 사용할 수 있으며, 내부함수의 실행을 통해 검색 시스템의 제반설정사항을 변경할 수 있다.

4. 구현 및 평가

제안한 시스템은 사용자의 질의를 받아 검색 결과를 HTML문서로 사용자에게 보내주는 질의서버, 사용자의 디렉토리입력을 위한 등록서버, 스케줄러, 로봇관리기, 문서분석기, 원격관리모듈로 구성되며 시스템 구현 환경은 <표 1>과 같다.

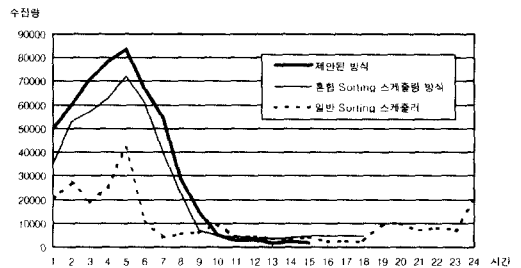
<표 1> 시스템 구현 환경

분 류	질의서버 등록서버	스케줄러, 로봇관리기	어휘분석기	원격관리모듈
사용OS	알짜 LINUX			
웹서버	아파치 웹서버			
프로세스	번티업 166			
개발도구	PHP	JAVA	RML, Swing	
DBMS	MySQL			

사용자 인터페이스에 관련된 모든 웹 문서는 PHP언어를 이용하여 구현하였는데 PHP언어는 대부분의 운영체제 및 데이터베이스를 지원할 수 있으며, 타 언어와 달리 서버에 독립적인 프로세스를 생성시키지 않으므로 서버에 대한 자원부담을 줄일 수 있어서 속도가 빠른 것으로 평가되고 있다.

원격관리 모듈은 JAVA Applet으로 구현하여 웹문서에 포함시켰으며, Swing을 이용한 인터페이스 구현으로 관리자의 간편한 조작에 의하여 검색엔진시스템의 제반사항을 관리할 수 있다.

네트워크상 임의의 서버 50곳을 선정, 새벽 1시부터 같은 조건으로 500Gbyte의 문서들을 수집하여 기존 방식(일반 Sorting 스케줄링, 혼합 Sorting 스케줄링 방식)과 제안 시스템(개선된 혼합 Sorting 스케줄링 방식)의 정보 수집량을 비교한 결과 그림 10에서와 같은 결과를 얻을 수 있었다.



(그림 10) 정보 수집량 비교 그래프

(그림 10)에서 일반 Sorting 스케줄링 방식(점선)의 경우 전체 문서 수집에 24시간을 초과하였으며, 혼합 Sorting 스케줄링 방식(열은 실선)은 약 18시간이 소요되었고, 본 논문에서 제안한 개선된 스케줄링 방식(짙은 실선)으로는 약 15시간이 소요됨을 알 수 있다.

따라서, 각 로봇의 수집 활동시에 실시간으로 체험한 대상 서버의 부하량을 적용하여 수집활동을 하고 있는 동안에도 보다 효율적인 사이트를 우선 처리함으로써 많이 효율을 가져올 수 있음을 알 수 있다.

비교되는 방식 모두가 새벽시간대의 문서수집량이 낮 시간대보다 높게 나타남을 알 수 있으며, 이는 문서수집 시간대가 24시간으로 설정되었기 때문에 가장 효율적인 시간대가 집중되어 있는 새벽에 많은 수집량을 보이고 있음을 알 수 있다. 본 제안 시스템에서는 원격관리를 통해 수집시간대를 조절하여 전자상거래 등과 같은 정확한 가격정보를 요구하는 시스템에서 정보 갱신의 신속성을 향상시킬 수 있다.

또한 스케줄러 또는 원격관리에 의해 효율이 높은 시간대 로봇의 개수를 증가시킴으로써 효율을 더욱 향상시킬 수 있을 것으로 기대된다.

5. 결 론

기존의 혼합스케줄링기법은 특정 시간에 측정된 서버의 부하량을 기준으로 스케줄링이 이루어짐으로서 네트워크 부하를 측정하는데에는 한계가 있다. 이를 보완하기 위하여 본 논문에서 제시한 개선된 혼합 스케줄링기법은 특정 주기를 정하여 서버의 부하량을 측정하여 매회 부하량의 평균값과 실제 로봇이 대상 웹서버를 대상으로 문서수집시 체험한 실시간 부하량정보를 기준으로 스케줄링함으로써 네트워크 부하를 더욱 정확히 측정할 수 있어서 시간대별 대상서버 부하가 최대한으로 고려된 효율적인 문서 수집이 가능하도록 하였으며, 또한 검색시스템의 효율적인 측면에서도 커다란 향상을 가져왔다.

본 논문에서 제안한 검색시스템은 특정 상황발생시 또는 특정 설정변경시 웹브라우저를 통하여 관리자의 실시간 관리 및 변경이 가능하여 능동적인 상황대처가 가능하다. 또한 본 검색시스템은 자바기반으로 개발되었기 때문에 플랫폼의 제약이 없다.

앞으로 검색 대상 서버들의 부하량 관리에 있어서 각각의 라우터마다의 연결경로를 고려하여 우회하는 방법을 첨가한 에이전트 연구와, 검색시스템과 원격관리모듈간 작업에서의 보안관련 사항에 대한 계속적인 연구가 요구된다.

참 고 문 헌

- [1] 金 壹, "Design and Implementation of Real-time Contro Search Engine for Local Information Network," Kyungnam University 碩士學位, 1998, 12.
- [2] 鄭龍勳, "A Search Robot Engine using Mixed Sorting Scheduling Method," Kyungnam University 碩士學位, 1999, 12.
- [3] Todd Courtois, "Java Networking and Communications," pp.43-88, 1997.
- [4] Philip Heller & Simon Roberts, "Inside Secrets Java 2 Developer's Handbook," pp.648-693.
- [5] 金昌根, "Design and Implementation of Region Administration-Oriented Search Engine Based on

Robot Control Policy," Kyungnam University 博士學位, 1999, 6.



박 규 석

e-mail : kspark@hanma.kyungnam.ac.kr
 1980년 중앙대학교 전자계산학과
 전산학전공(석사)
 1988년 중앙대학교 전자계산학과
 전산학전공(박사)
 1982년~현재 경남대학교 컴퓨터
 공학과 교수

1992년~1996년 경남대학교 전산정보원 원장
 1995년~1996년 한국정보과학회 이사, 영남지부장
 1999년~현재 경남대학교 정보통신연구소 소장
 2000년~현재 한국멀티미디어 학회/부회장
 관심분야 : 분산처리시스템, 정보통신 소프트웨어,
 멀티미디어 시스템



이 충 석

e-mail : cslee@sslab.net
 1997년 경남대학교 컴퓨터공학과
 졸업(공학사)
 1996년~1997년 日本 Core Dump
 Group(NTT : 개발연구원)
 1997년~1998년 서울 PowerNet(주)
 LG Soft : 프로그램 개발

1999년~현재 경남대학교 컴퓨터공학과 석사졸업예정
 관심분야 : 전자상거래, 에이전트기술, 분산시스템



김 성

e-mail : bgbest@sslab.net
 2000년 경남대학교 컴퓨터공학과
 졸업(공학사)
 2000년~현재 경남대학교 컴퓨터
 공학과 석사과정
 관심분야 : 에이전트기술, 웹 프로
 그래밍, 멀티미디어