다중 프로세스 기반 웹 로봇의 수행동작 분석

김희철*, 이용두*
*대구대학교 정보통신공학과

요 약

웹 로봇은 인터넷 검색엔진을 포함한 다양한 웹 응용프로그램에 활용되는 중요한 인터넷 소프트웨어 기술이다. 인터넷의 급격한 성장에 따라 고성능 웹 로봇의 구현이 시급히 요구되고 있다. 이를 위해서는 웹 로봇에 대한 성능확장성에 초점을 둔 연구가 수행되어야 한다. 하지만 기존의 웹 로봇에 대한 연구개발은 주로 구현에 초점을 두고 수행되어 왔으며 따라서 성능확장성에 대한 체계적인 연구 결과는 발표되고 있지 않다. 본 연구에서는 이러한 성능확장성에 관한 선행연구로서 기존 웹 로봇 모델의 수행동작(Execution Behavior)을 성능 측면에서 이해하고자 웹 로봇의 수행동작에 대한 분석 결과를 제공한다. 본 연구에서는 Fork-join을 기반으로 하는 다중프로세스 기반의 웹 로봇 모델에서 웹 로봇이 웹 서버에게로 전송하는 접속요청, 문서헤드요청, 문서본문요청 시에 설정하는 타임아웃(Timeout) 값이 성능에 미치는 영향을 분석하였다. 또한 전체 컴퓨팅 소요시간에서 URL 추출 및 유일성 검사 등이 점유하는 비율을 산출하여 웹 로봇의 동작을 분석하였다. 이러한 분석 결과를 기반으로 하여 향후 웹 로봇의 성능향상을 위한 설계 방향을 제시한다.

Analysis of Execution Behavior for Multprocess-based Web Robots

Hiec-heol Kim* · Yong-Doo Lee*

Web robot is an important Internet software technology used in a variety of Internet application software which includes search engines. As Internet continues to grow, implementations of high performance Web robots are urgently demanded. For this, researches specially geared toward performance scalability of Web robots are required. Hoever, because researches are focused mostly on addressing issues related to commercial implementations, scientific researches and studies are not still made on the performance scalability. In this research, we choose a Web robot model implemented by fork-join based multiprocesses. With respect to the model; we evaluate the effect on the collection efficiency that the timeout values set to requests from Web robots to Web servers have. Also, we analysed the behaviors of Web robots by comparing the execution time between the URL extraction and the uniqueness checking for the extracted URLs. as well as by comparing between the computation time and the network time. Based on the analysis result, we suggest the direction for the design of high performance Web robots.

1. 서 론

웹 로봇은 인터넷 검색엔진 및 웹 기반 에이전트를 구성하는 소프트웨어 엔진으로서 인터넷상에 존재하는 웹 문서들을 추적·수집하는 기능을 갖는다[8]. 그러므로 웹 로봇 기술은 야후와 같은 인터넷 검색시스템 산업, 전자상거래 상품검색 산업, 인트라넷 검색 소프트웨어 산업 등 기존 인터넷 산업의 근간이 되는 핵심 기술이다[5,8,9].

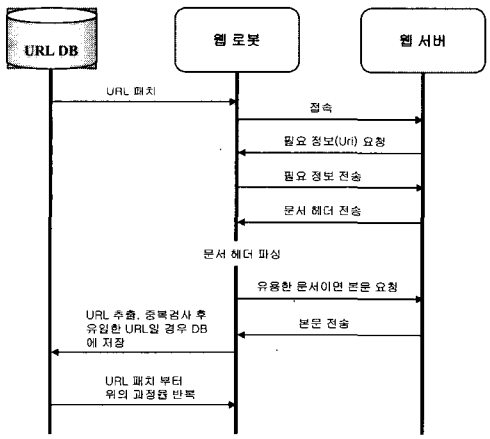
웹의 출현부터 현재에 이르기까지 인터넷의 급격한 성장은 지속되고 있으며 인터넷 상의 웹 문서의 개수는 매우 급격히 증가하고 있다[2]. 따라서 검색엔진을 비롯한 각종 인터넷 응용 소프트웨어에 활용되고 있는 웹 로봇에 대한 성능향상이 요구되고 있다[8]. 이러한 요구에도 불구하고 웹 로봇의 성능확장성(Scalability)에 대한 체계적인 연구는 매우 미흡한 실정이다. 그 주된 이유로는 인터넷이 급격한 성장에 따라 신속한 상용화를 위한 웹 로봇의 구현에 주된 초점을 두고 연구개발이 수행된 때문이라 할 수 있다[6,7]. 하지만 인터넷의 성장이 지속됨에 따라 최근 기존 검색엔진의 성능확장성에 한계에 직면하게 되었다. 따라서 인터넷 상의 문서 수집 및 검색에 성능확장성의 확보를 목표로 하는 연구가 시급히 요청된다.

본 논문에서는 이러한 연구의 일환으로 웹 로봇의 상세 동작분석에 관한 연구결과를 제공한다. 전술한 바와 같이 기존 웹 로봇에 대한 연구개발은 상용화에 중점을 두고 수행되어 왔다. 이러한 연구는 웹 로봇의 성능향상 보다는 신뢰성 있는 구현을 그 주된 목표로 하고 있다. 본 논문에서는 향후 고성능 웹 로봇의 설계를 위한 기초로서 필수적으로 요구되는 웹 로봇의 상세 동작에 대한 분석결과를 제공하게 된다.

2장에서 웹 로봇에 대한 개요를 소개하며 3장에서는 본 논문의 동작분석 대상이 되는 웹 로봇 모델에 관하여 상세히 설명한다. 4장에서는 실험을 통한 웹 로봇의 동작 분석 결과를 제시하며 5장에서는 결론을 맺는다.

2. 웹 로봇 개요

웹 로봇은 웹 서버 상의 웹문서를 수집하기 위한 소프트웨어이다. 웹 로봇은 자동으로 웹 페이지의 내용을 분석하고 그 안의 포함되어 있는 URL들을 추출한 후 그 URL(Universal Resource Locator) 들에 대한 문서를 수집한다.



(그림 2) 웹 로봇의 수집 동작 개요
(Figure 1) Operational overview of Web robots

(그림 1)은 웹 로봇의 문서수집 동작을 도식화하여 보여준다. 문서 수집 과정을 살펴보면 먼저 수집할 문서에 대한 URL 목록을 지니고 있는 URL DB에서 한 개의 URL을 취한 후 그 URL에 대한 호스트에 접속한다. 접속이 성공적으로 이루어지면 웹서버는 웹 로봇에 URI(Universal Resource Identifier)를 요청하게 된다. 요청된

URI에 대하여 웹 서버는 해당 문서의 헤더를 보내준다. 웹 로봇은 그 헤더 정보를 파싱(Parsing)하여 유용한 문서인지를 검사한다. 유용한 문서일 경우 웹 서버에 웹 문서의 본문을 요청·적재한다. 웹 로봇은 그 웹 문서을 파싱하여 문서 내의 인링크(In-link) URL들을 추출해 낸 후에 추출된 URL들을 기 수집한 URL 목록(URL DB)에 대하여 중복검사를 수행한다. 중복검사결과 추출된 URL 중에서 URL DB에 존재하지 않는 URL들만 URL DB에 삽입한다. 웹 로봇은 위의 과정을 반복하며 문서를 수집하게 된다.

위에서 살펴본 바와 같이 웹 로봇의 동작 원리는 매우 단순하다. 하지만 실용화 웹 로봇의 구현에 있어서는 성능에 영향을 주는 몇 가지 사항에 대한 충분한 고려가 수반되어야 한다. 먼저 웹 로봇이 구동되어 수행되는 동안 동일한 URL에 대한 웹 문서 수집은 한 번만 수행되어야 한다. 즉 문서의 중복 수집을 방지하여야 한다. 한편 주어진 URL에 대한 문서의 수집여부를 그 문서의 갱신여부에 준하여 결정하여야 한다. 이것을 현행화 문제라 하며 이전 웹 로봇의 구동 시에 수집된 문서들 중에서 내용이 갱신되지 않는 문서들의 수집을 방지하는 것은 웹 로봇의 성능향상에 필수적으로 요구된다.

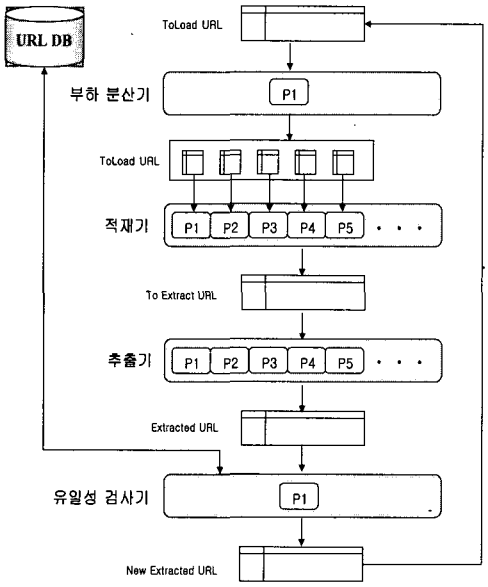
한편 멀티 프로세스를 기반으로 하는 웹 로봇의 구동 시에 여러 개의 프로세스가 동시에 한 서버에 문서를 요청할 경우 그 서버와 네트웍에 과부하를 발생시킬 수 있으며 향후 접근거부를 초래할 수도 있다. 그러므로 웹 로봇은 상대 시스템과 네트웍의 과부하를 방지할 수 있도록 문서 요청에 대한 적절한 스케줄링(Scheduling) 기법을 도입해야 한다. 또한 호스트에 접근거부 설정을 위하여 표준으로 정한 로봇 배제 규약을 준수하여야 한다. 로봇 배제 규약은 상대 시스템에 계속적인 접근으로 인해 과부하를 주거나, 일시적인 정보, CGI 등의 적당치 않은 부분을 검색하는 것을 막아보고자 Martijn Koster에 의하여 그 표준이 제안되었다. 로봇배제 규약을 준수하기 위해서는 상대시스템의 robots.txt 파일도 다운로드(Download)하여 분석하여 해야 한다. 문서 하나를 수집하기 위하여 매번 robots.txt파일을 다운로드 한다면 웹 서버에 부하를 가중시키게 되어 문서수집 속도를 감소시킨다. 그러므로 웹 로봇의 성능향상을 위해서는 이러한 문제에 대한 해결 방안을 제시하여야 한다.

3. 동작 분석을 위한 웹 로봇 모델

본 장에서는 웹 로봇의 동작에 대한 상세 분석을 위하여 채택한 웹 로봇 모델에 대하여 설명한다. 본 연구에서 웹 로봇 모델로서 기존 웹 로봇의 구현에 가장 널리 채택하고 있는 Fork-Join 기반 다중프로세스 웹 로봇을 선정하였다. 이러한 모델의 선정은 웹 로봇의 동작 분석 결과에 대하여 객관성을 제고시켜 향후 본 결과를 웹 로봇의 성능 향상을 위한 제반 설계에 효과적으로 활용할 수 있도록 한다.

3.1 Fork-Join 기반 다중프로세스 웹 로봇

기존 웹 로봇의 구현 및 구동에는 멀티프로세스 방식이 채택되고 있다. 이것은 웹 로봇이 네트웍 I/O 트랜잭션(Transaction)을 기반으로 하여 동작하므로 여러 개의 URL에 대하여 동시 수집이 가능하기 때문이다. 이러한 동시 수집을 위한 웹 로봇은 일반적으로 문서 적재 기능 및 적재된 문서들로부터 신규 URL 추출 기능을 다중프로세서로 구동시킨다.



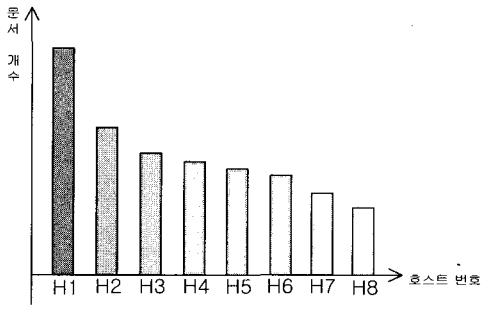
(그림 3) 동작분석을 위한 웹 로봇 모델 구조
(Figure 2) Organization of the Web robot model

(그림 2)는 본 논문에서 채택하고 있는 Fork-Join 기반 다중프로세스 웹 로봇의 구조를 보여준다. 본 모델의 기능 모듈은 크게 부하분산기, 적재기(Web Page Loader), URL 추출기, 유일성 검사기로 구성된다. 이전에 지적한 바와 같이 웹 로봇의 구동에 있어 적재기와 추출기는 다중프로세스로서 구동된다.

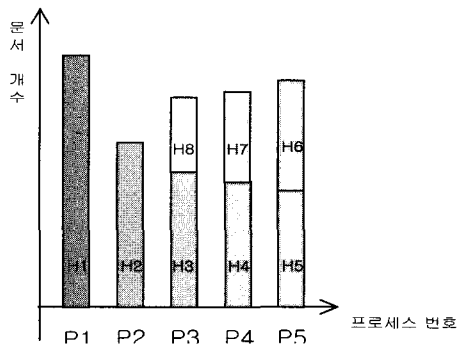
위의 기능 모듈들의 입출력으로서 사용되는 데이터는 크게 URL DB, ToLoad URL 목록, ToExtract URL 목록, Extracted URL 목록, New Extracted URL 목록으로 구성된다. URL DB는 웹 로봇이 수집을 완료한 문서들에 대한 URL 목록으로서 데이터베이스 또는 물리적 파일로 구현할 수 있다. 본 연구에서는 물리적 파일로 구현하고 있다. ToLoad URL 버퍼(Buffer)는 웹 로봇이 수집할 대상 URL들의 목록을 저장한다. ToExtract URL 버퍼는 적재기가 수집한 문서들에 대한 URL 및 해당 웹페이지 파일의 저장 경로 등의 정보를 저장하고 있다. Extracted URL 버퍼는 추출기의 출력인 인링크 URL 목록을 저장한다. 마지막으로 New Extracted URL은 검사를 통하여 중복 및 비신규 URL들을 제거한 비중복, 신규 인링크 URL 목록을 저장한다. 이러한 데이터 저장공간은 메모리 상의 버퍼로서 구현된다.

3.2 부하분산기

부하분산기는 수집 대상 URL들을 각 적재기 프로세서들에게 분배하는 역할을 담당한다. 이러한 URL 분배의 최종 목표는 대상 서버에 과부하를



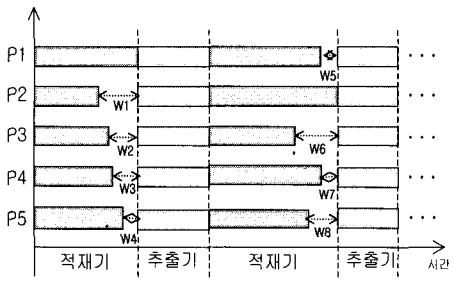
(그림 4) 호스트별 URL 개수
(Figure 3) URL count for each host



(그림 5) 프로세스별 URL 분배
(Figure 4) URL distribution for each process

방지에 있다[3]. 일반적으로 사용되고 있는 정적 스케줄링 기법은 적재기가 동작하기 전에 (그림 4)와 같이 각 호스트를 한 개의 적재 프로세스에 할당한 후 해당 호스트를 갖는 URL들을 모두 그 적재 프로세스에 분배한다. 이러한 URL 분배 방식은 주어진 시간에 호스트에 오직 한 개의 적재 프로세스만 접속되므로 서버의 과부하를 방지할 수 있다. 하지만 적재 프로세스에 호스트 단위로 URL들을 분배하므로 적재기 프로세스별로 서로 다른 개수의 URL들이 분배되어 적재 프로세스 측면에서 부하는 균등하지 않게 된다.

예를 들어 수집해야할 문서들이 호스트 별로 (그림 3)과 같이 존재한다고 가정하자. (그림 4)는 적재 프로세스 개수를 5개로 가정한 후 위의 부하분산 방식을 적용하여 적재 프로세스 별로 URL을 분배한 결과를 보여준다. (그림 5)는 프로세서별 동작을 도식화하여 나타내고 있다. 그림에서 보는 바와 같이 적재 프로세스마다 종료되는



(그림 6) 동작 사례
(Figure 5) Example operation

시점이 다르며 이에 따라 W1 ~ W8로 표시되어 있는 시스템 효율성 저하가 발생한다.

3.3 적재기

적재기는 웹 로봇의 가장 필수적인 모듈로서 URL을 입력으로 하여 해당 URL의 호스트에 소켓 접속한 다음 웹 서버에 URI를 요청하게 된다. 웹 서버는 요청된 URI를 검색한 다음, 검색 결과에 따라 문서의 헤드를 웹 로봇에 전송한다. 웹 로봇은 적재한 헤드를 파싱하여 유용한 문서인지를 검사하고, 유용한 문서이면 본문의 내용을 적재한다. 적재된 문서는 웹 로봇이 구동하는 시스템(Local system)의 디스크에 저장된다.

적재기는 Fork-join 기반 다중프로세스로서 동작하며 각 URL들에 대한 문서들을 해당 호스트들로부터 수집하는 기능을 수행한다. 부하분산기로부터 각 프로세스에 대한 수집대상 URL들의 분배가 완료되면 적재 프로세스들이 생성되어 각 적재 프로세스는 독립적으로 구동하여 문서수집을 수행하게 된다. 모든 적재 프로세스가 해당 문서의 수집을 완료하게 되면 웹 로봇의 수행에 있어 적재 단계가 종료되어 URL 추출 단계로 진입한다 (그림 5).

본 Fork-join 다중프로세스 웹 로봇 모델에서 적재 단계에서 다음 단계인 추출 단계로 진입하려면 모든 적재 프로세스가 종료된 이후에 가능하다. 하지만 이전에 설명한 바와 같이 적재 프로세스는 부하분산기에 의하여 할당된 URL을 대상으로 문서를 수집하므로 각 프로세스가 종료되는 시점이 모두 다르다. 그러므로 적재기의 수행 시간은 가장 긴 프로세스의 수행 시간이 된다

3.4 추출기

추출기는 적재기가 수집한 각 문서를 파싱하여 인링크 URL을 추출하는 역할을 담당한다. 웹 문서의 파싱 시에 가장 큰 문제점은 사용자들이 HTML 문법에 맞지 않는 코드를 생성해 놓기 때문에 사용자의 잘못된 코드까지 충분히 고려해야 한다는 것이다. 본 연구에서는 이러한 문제를 해결하기 위해 문법에 맞지 않는 코드까지 지원할

수 있도록 구현하였다.

추출기는 적재기가 수집한 문서들에 대하여 태그(Tag)만을 추출하여 만든 태그 목록 작성하며 이 태그 목록에서 웹 문서 참조(Link)를 추출하여 Extracted URL 버퍼에 삽입한다.

Fork-join 기반 다중프로세스 웹 모델에서는 본 추출기도 Fork-join 기반의 다중프로세스로 구동한다. 적재기와는 추출기에서는 각 프로세스별로 동일한 개수의 URL들을 분배할 수 있으므로 부하 불균형 문제가 발생하지 않는다 (그림 5).

3.5 유일성 검사기

웹 로봇은 한번 수집한 문서를 다시 수집하는 일이 없어야 한다. 수집한 문서의 URL은 URL DB에 저장하고 새로운 URL이 추출되면 URL DB를 검색하여 존재하는 않는 URL들만 대상으로 신규 적재를 수행하여야 한다.

유일성 검사기는 추출기의 출력인 URL 목록 중에서 중복된 URL을 제거한 후 이를 기 수집된 URL 목록과 비교하여 중복여부를 검사한 후 최종적으로 신규 수집대상 URL 목록을 작성하여 URL DB에 추가시키며 또한 부하분산기의 입력으로 제공하는 기능을 갖는다. 즉 이 신규 수집대상 URL들은 다음 단계의 적재·추출에 활용한다.

본 연구에서는 자료 일관성을 유지하기 위하여 유일성 검사기를 단일 프로세스로 구현하였으며 유일성 검사에 있어 효율성을 높이기 위하여 다중 루트(Root) 이진 트리 기법을 사용하였다. 다중 루트는 해쉬(Hash)로 구성되어 있다. 그러므로 하나의 URL 중복검사는 그 URL에 대한 해쉬 키((Hash Key)를 구한 후 다중 루트에서 해쉬 키에 해당하는 이진트리 만을 검사하면 된다.

4. 웹 로봇의 수행동작 분석

본 절에서는 3절에서 설명한 웹 로봇 모델에 대하여 그 수행동작(Execution Behavior)에 대한 상세 분석 결과를 기술한다.

웹 로봇의 성능은 크게 수집효율(Collection efficiency)과 수집속도(Collection speed)로 구성된다. 수집효율은 수집 가능한 URL에 대하여 얼마나 많은 URL들을 수집하는 가를 나타내며 수집속도는 얼마나 신속하게 수집을 수행하는가를 나타내는 척도이다. 수집효율은 주어진 수집대상 URL에 대하여 수집에 성공한 URL의 비율로, 수집속도는 단위 시간당 수집한 URL의 개수로 나타낼 수 있다.

본 웹 로봇의 수행동작에 있어 소켓 접속을 위한 타임아웃(Timeout) 크기가 수집효율 및 수집성능에 미치는 영향을 분석하였다. 또한 웹 로봇의 각 기능 모듈별 수행시간이 전체 수행시간(Turn around time)에서 점유하는 비율을 산출하여 분석하였다.

본 수행동작의 분석에는 실제 실험을 통해서 얻은 데이터, 즉 2절의 웹 로봇 모델을 소프트웨어로 구현한 후 실제 하드웨어 시스템에서 구동시켜 얻은 데이터를 활용하였다. 본 실험에 활용한 하드웨어 시스템은 Hewlett Packard 사의 PC 서버로서 2개의 Pentium III 731MHz CPU와 2G 메모리, 90G Hard Disk Array를 장착하고 있다. OS는 Linux RedHat 6.2 버전을 사용하였다. 본 서버는 10M Ethernet 카드를 장착하고 있으며 인터넷 외부망에 T3 용량으로 연결되어 있는 네트웍에 위치하고 있다.

본 수행동작 분석을 위한 웹 로봇 문서 수집대상으로 국내 대학교 웹사이트 10개를 임으로 선

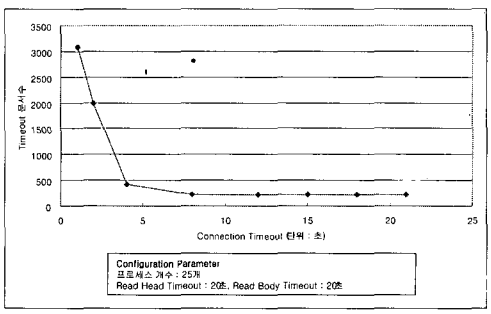
정하였으며 전체 수집 대상 URL의 개수는 약 5만개이며 문서 수집 시에 생성시킨 로그파일에 근거로 하여 분석결과를 산출하였다.

웹 로봇의 수집은 동일한 네트워크 환경에서 밤 12시부터 같은 조건으로 프로그램을 구동하여 시간별 문서 수집량을 측정하였다. (!! 조금더 추가)

4.1 소켓 타임아웃(Timeout) 영향 분석

웹 로봇의 적재 프로세스는 자료수집 과정에 웹 서버에와 상호동작을 반복·수행한다. 이러한 상호동작은 크게 웹 로봇으로부터 웹 서버로의 접속요청(Connection request), 문서 헤더 요청(Read head request), 문서본문 요청(Read body request)으로 구성된다.

이러한 요청은 기본적으로 소켓 통신을 기반으로 구현되며 주어진 웹 로봇의 적재 프로세스 는 요청을 전송한 후 응답이 도달할 때까지 수행을 멈추고 대기 상태에 놓여 있게 된다. 만약 웹 서버 측에서 웹 서버, 해당 호스트, 네트웍 등의 장애로 인하여 정상적인 응답이 가능하지 않은 경우 최대 180초까지의 응답지연이 발생하게 된다.



(그림 6) 접속요청시 Timeout 값과 응답실패 문서수
(Figure 6) Timeout value versus failed document count for connection requests

이러한 경우가 발생할 경우 웹 로봇의 전체 수집속도(Collection speed)가 매우 저하된다.

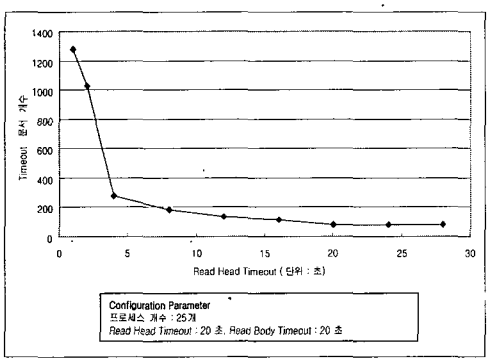
적재 프로세스는 Unix의 알람시스널(Alarm signal)을 기반으로 하여 요청에 대한 Timeout 크기를 조절하여 위의 수집성능 저하문제를 해결해야 한다. Timeout 값이 매우 작을 경우에는 네트웍 또는 호스트의 과부하 또는 일시적인 서비스 지연으로 인하여 문서수집 실패를 초래하게 되어 전체문서 수집효율이 저하된다. Timeout 값이 매우 클 경우에는 시스템 또는 네트워크 장애시, 삭제된 URL에 대한 문서수집 시에 대기 시간이 커지므로 문서 수집속도가 낮아지게 된다. 즉 수집효율과 수집속도 간에는 상충관계(Tradeoff)가 발생한다.

본 실험에서는 위의 웹 로봇으로부터 웹 서버로 전송하는 각 요청에 대하여 Timeout 값의 크기가 응답실패율에 미치는 영향을 분석하였다.

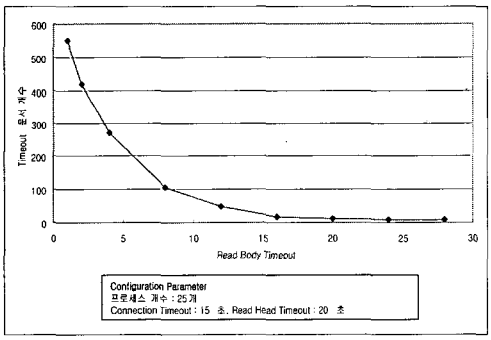
Timeout 값이 매우 작은 경우 접속 요청에 대한 응답실패율이 높아지게 된다. 실제로 (그림 6)에서 보는 바와 같이 Timeout 값이 1초 일 때 접속요청의 경우 3000개 이상의 문서에 대하여 응답실패가 발생하고 있음을 볼 수 있다. 또한 (그림 7과 (그림 8)에서 보면 문서헤드 요청과 문서본문 요청을 비교할 때 Timeout 값이 매우 작은 경우 문서헤드 요청의 경우가 문서본문 요청의 경우보다 약 2배 이상의 응답실패율을 보이고 있음을 볼 수 있다.

수집효율을 최대로 보장할 수 있는 최소 Timeout 크기를 살펴보면 먼저 접속(Connection) 청에 있어서는Timeout 값을 8초로 이상인 경우에는 응답실패 문서수가 더 이상 감소하지 않는 것을 볼 볼 수 있다. 문서헤드 요청에 있어서는 (그림 6)에서 보는 바와 같이 Timeout 값이 약 25초 이상일 경우에는 응답실패 문서수가

Timeout 값에 영향을 받지 않는 최소 수준에 도달함을 볼 수 있다.



(그림 7)) 문서헤드 요청시 Timeout 값과 응답실패 문서수
(Figure 7) Timeout value versus Failed document count for read head requests

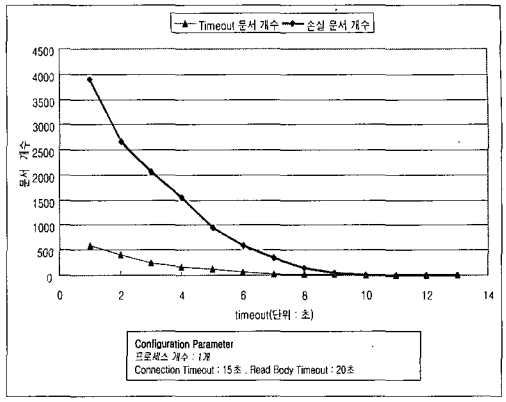


(그림 8) 문서본문 요청시 Timeout 값과 응답실패 문서수
(Figure 8) Timeout value versus Failed document count for read body requests

요

마지막으로 문서본문(Body) 요청에 대해서는 Timeout 값이 약 30초 이상이 되어야 응답실패 문서수가 Timeout 값에 영향을 받지 않고 최소 수준에 도달하게 된다 (그램 7).

웹 문서는 하이퍼링크(Hyperlink)를 기반으로 하여 상호 연결되어 있다. 그러므로 주어진 문서에 대하여 응답실패가 발생하면 그 문서에 링크되어 있는 하위 문서들의 수집도 대부분 가능하지 않게 된다. (그림 9)는 문서헤드 요청에 Timeout



(그림 9) Timeout 값과 전체 문서수집 손실양
(Figure 9) Timeout value versus Total failed document count

값의 변동에 따른 전체 수집실패 문서개수 나타내주는 그래프이다. 본 그래프에 따르면 문서헤드 요청에 대한 응답실패량의 약 8배 정도에 해당하는 문서들에 대하여 수집을 실패하고 있음을 볼 수 있다.

위에서 살펴본 Timeout 값이 문서 수집효율에 미치는 영향에 대한 분석 결과로서 향후 웹 로봇의 설계에 있어 아래와 같은 점을 고려해야 한다. 일반적으로 웹 로봇의 Timeout 값은 요청의 종류에 상관없이 동일한 값으로 설정하여 구동시키는 경향이 많다. 그러나 앞에서 살펴본 바와 같이 접속 요청이나 문서헤드 요청, 문서본문 요청에 따라 수집효율을 최대화하는 Timeout 값은 상이하므로 웹 로봇의 웹서버에의 요청은 그 종류 별로 Timeout 값을 적절하게 설정하여야 한다.

또한 응답실패에 따른 문서의 수집실패는 그 문서 내에 포함되어 있는 인링크 URL에 대한 문서의 수집실패를 초래할 위험성이 있으므로 Timeout값의 선정에 있어서 웹 로봇 수행 중에 응답실패율을 관찰하여 상황에 적합하도록 동적으로 최적의 값으로 변경하여 설정할 수 있도록 하는 적응형(Adaptive) Timeout 설정기법에 대

한 연구도 필요하다.

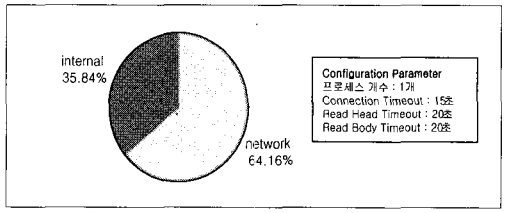
4.2 세부 기능별 성능 분석

Amdahl's 법칙에 따르면 컴퓨터 시스템의 성능향상은 성능에 가장 큰 비중을 차지하고 있는 부분에 우선적인 초점을 두어야 한다. 웹 로봇 또한 수집속도을 향상시키기 위해서는 웹 로봇의 전체 수집시간에 각 세부 기능이 차지하고 있는 비율을 먼저 파악해야 한다.

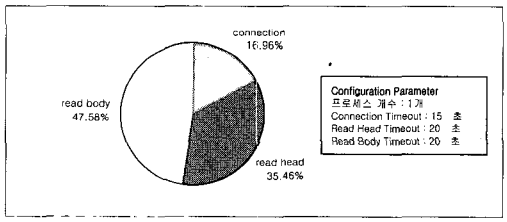
웹 로봇의 수행시간(Turn-around time)은 크게 네트웍 시간과 컴퓨팅 시간으로 구성된다. 네트웍 시간은 웹 로봇이 네트웍 트랜잭션에 소요하는 시간이며 접속(Connection), 문서헤드 적재, 문서내용 적재에 소요되는 시간을 포함한다. 컴퓨팅 시간은 네트웍 트랜잭션을 제외한 시간을 의미하며 부하분산, URL 추출, 유일성 검사에 소요되는 시간을 포함한다. 향후 웹 로봇의 수집속도를 향상시키기 위해서는 이러한 세부 기능별 수행시간 점유율의 산출이 필수적이다.

본 실험에서는 웹 로봇의 수행 시 생성시킨 로그를 분석하여 각 기능별 수행시간 점유율을 산출하였다. 웹 로봇의 적재기 및 산출기 프로세스는 한 개로 설정·구동시켜 각 기능별 수행시간의 상대 크기를 산출하였다.

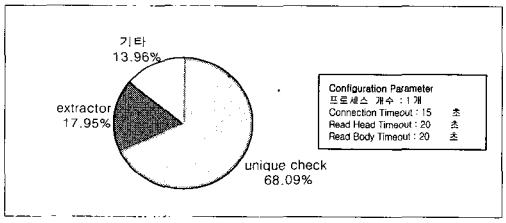
먼저 (그림 10)은 네트웍 시간 대비 컴퓨팅 시간의 비율을 보여준다. 네트웍 시간이 전체 수행시간의 64.2% 컴퓨팅 시간이 36.2%를 점유하고 있음을 알 수 있다. (그림 11)은 네트웍 시간의 세부 구성을 보여 준다. 전체 네트웍 시간에서 접속(Connection), 문서헤드 적재, 문서내용 적재에 소요되는 시간은 각각 17.0%, 35.5%, 47.6%를 점유하고 있음을 볼 수 있다. (그림 12)는 컴퓨팅 시간의 세부 구성을 보여 준다. 전체 컴퓨팅 시간에서 유일성 검사시간이 68.1%, URL 추출시간이 18%를 점유하고 있음을 볼 수 있다. 본 결과는 유일성 검사시간이 URL 추출시간 보다 3배 이상이 된다는 사실은 매우 의외의 결과라 할 수 있다.



(그림 10) 네트워크와 컴퓨팅 시간의 분포
(Figure 10) Network time versus computing time



(그림 11) 네트워크 시간 내의 세부기능 시간분포
(Figure 11) Distribution of network time



(그림 12) 컴퓨팅 시간 내부의 세부기능별 시간분포
(Figure 12) Distribution of computing time

위의 실험 결과를 통하여 볼 수 있는 것은 네트웍시간에서 접속과 문서헤드 적재시간을 합이 문서내용을 적재하는 시간보다 크다는 사실이다. 이러한 결과는 데이터의 전달시간 보다 네트웍 접속에 소요되는 시간이 매우 크다는 사실을 나타낸

다. 그러므로 전체 네트웍 시간을 감소시기 위해서는 문서수집 시에 필요 없는 접속 및 문서헤드 적재 시도 횟수를 제거할 수 있도록 향후 문서 갱신 및 삭제 여부를 검사하는 문서수집 프로세스로부터 독립 프로세스를 구성하는 웹 로봇의 설계 방식을 고려할 수 있다.

5. 결 론

본 논문에서는 웹 로봇의 성능확장성에 관한 연구에 필수적으로 요구되는 선행연구로서 기존 웹 로봇의 모델에 관하여 그 수행 동작을 정확하게 이해할 수 있는 각종 성능평가를 수행하였다. 본 연구의 결과로서 웹 로봇으로부터 웹 서버로의 접속 요청, 문서헤드 요청, 문서본문 요청에 있어 Timeout 값이 문서의 수집효율에 미치는 영향을 분석하였다. 본 분석결과 기존의 선험적 이해와는 달리 웹 서버로의 요청의 유형에 따라 요구되는 Timeout 값은 상이하다는 사실을 발견할 수 있었다. 그러므로 향후 웹 로봇의 설계에 있어 각 유형별로 적합한 Timeout 값을 설정하여야 보다 향상된 성능을 얻을 수 있다. 한편 웹 로봇 수행 쓰레드(Execution thread)의 전체 컴퓨팅 소요 시간에서 유일성 검사 시간이 68.1%를 차지하고 있다. 따라서 향후 고성능 웹 로봇 설계에 있어서 유일성 검사를 위한 보다 향상된 알고리즘의 개발이 이루어 져야 한다.

References

[1] R. Fielding, J.Gettys, J.C. Mogul, H. Frystyk and T. Berners-Lee, "RFC 2068 Hypertext Transfer Protocol HTTP/1.1", UC Irvine, Digital Equipment Corporation, MIT, 1997.

[2] N.J. Yeager and R.E. McGrath, "Web Server Technology", Morgan Kaufman Publishers, 1996.

[3] Michele Colajanni, et al, "Dynamic Load Balancing on Web Servers systems", http://computer.org, May 1999.

[4] Martijin Koster, "A proposed standard for Robot exclusion", Publi shed on the WWW at http://web.nexor.co.uk/mak/doc/robots/norobots.html

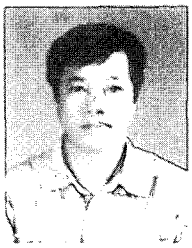
[5] M. Wooldridge and N. Jennings, "Intelligent Agents : Theory and Practice," Knowledge Engineering Review 10(2), 1995

[6] H. Kim, "Design and Implementation of Real-time Contro Search Engine for Local Information Network," Kyungnam University, Master thesis, 1998, 12.

[7] C. Kim, "Design and Implementation of Region Administration-Oriented Search Engine Based on Robot Control Policy,"Kyungnam University, Master thesis, 1999, 6

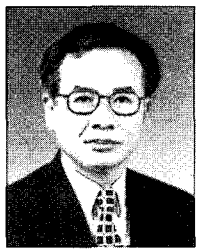
[8] 신동욱, "인터넷 환경에서의 분산정보 검색시스템의 설계 및 구현", 한국과학재단연구과제 961-0911-060-2, 1998, 충남대

[9] 강종백, 김성훈, 정원교, 이용두, "결정성 유한 오토마타를 이용한 한글 색인용어생성 알고리즘", 한국정보처리학회, 제3권 제2호,



김 희 철

1983년 연세대학교 전자공학과 졸업(공학사)
1991년 Univ. of Southern Californi (Computer Eng. M.S.)
1996년 Univ. of Southern Californi.(Computer Eng. Ph.D.)
1983년 - 1988년 (주)삼성전자 주임연구원
1996년 - 1997년 (주)삼성SDS 수석연구원
1997년 - 현재 대구대학교 정보통신학부 조교수
관심분야 : 병렬처리, 컴퓨터구조, 컴파일러



이 용 두

1975년 한국항공대학교 통신학과 졸업(공학사)
1982년 영남대학교 대학원 전자공학과(공학석사)
1995년 한국항공대학교 대학원 전자공학과 (공학박사)
1982년 - 현재 대구대학교 정보통신공학부 교수
1981년 - 1982년 (일)동경대학 전자공학과 객원 교수
1991년 - 1993년 Univ. of Southern California 교환교수
관심분야 : 컴퓨터구조, Internet 응용 기술