웹 수집 로봇 엔진의 설계 및 구현

김대유, 김정태
목원대학교

Implementation and Design of Robot Engine for Web Collection

Dae You Kim, Jung-Tae Kim
Mokwon University

E-mail : jtkim3050@mokwon.ac.kr

요 약

인터넷의 이용이 활발해짐에 따라 수많은 정보들이 웹을 통하여 공개되고 있으며, 이용자는 웹 검색 서비스를 이용하여 이러한 정보들에 효과적으로 접근할 수 있다. 웹 검색 서비스의 구축을 위해서는 웹 로봇을 사용한 웹 문서 수집이 선행되어야 하며, 웹 문서들의 수가 급격히 증가하면서 양질의 웹 문서들을 효과적으로 수집할 수 있는 웹 로봇에 대한 필요성이 증가되고 있으며, 그에 따른 많은 웹 수집 로봇이 탄생되고 있다. 본 논문에서는 효과적인 웹 수집 로봇의 설계와 동적인 웹페이지에서 사용하는 자바스크립트의 링크 추출방안에 대하여 제시하고자 한다.

Ⅰ. 서 론

인터넷 이용이 활발해짐에 따라 수많은 정보들이 웹 문서의 형태로 공개되고 있으며, 이러한 웹 문서들을 효과적으로 검색하기 위하여 웹 검색 서비스들이 이용되고 있다. 웹 로봇은 지정된 URL 리스트에서 시작하여 웹 문서를 수집하고, 수집된 웹 문서에 포함된 URL들을 추출과정과 새롭게 발견된 URL에 대한 웹 문서 수집과정을 반복하는 소프트웨어로서 웹 검색 서비스의 구축을 위해서는 웹 로봇을 이용한 웹 문서 수집이 선행되어야 한다. 1990년대 중반의 웹 문서 수는 현재에 비하여 매우 적었기 때문에, 최초로 개발된 웹 로봇 Wanderer를 포함하여 이 당시 개발된 다수의 웹 로봇들은 대용량의 웹문서들을 수집하도록 설계되지 않았다. 현재는 전 세계적으로 30억 개 이상의 웹 문서들이 존재하며, 국내에도 5천만 개 이상의 웹 문서들이 존재하고 있다. 따라서 이처럼 많은 수의 웹 문서들은 효율적으로 수집 할 수 있는, 즉 초당 수백 또는 수 천개의 웹 문서들을 수집 할 수 있는 웹 로봇의 필요성이 증가되고 있다.

Ⅱ. 연구배경

인터넷이 발달됨에 따라서 동적인 웹사이트가 증가 하고 있다. 사용자들이 원하는 게시물을 등록하고, 삭제 하고 수정할 수 있는 웹사이트가 증가됨에 따라서 웹 로봇은 이러한 특성들을 고려하여야 한다. 또한 동적인 웹사이트에서 사용되는 자바스크립트의 경우에는 기존에 사용되었던 정규표현식만으로는 찾아 낼 수 없다.

로봇은 URL을 통해서 웹 문서를 수집하게 되는데, 자바스크립트에 의한 링크를 URL로 생성할 수 없다는 많은 어려움이 있다. 웹 로봇 구현에 대한 연구는 웹 사이트에 대한 부하를 최소화 하면서 문서수집 속도를 최대화 하는 것을 중요한 목적이다. 또한 사이트 기반의 정적 또는 동적 웹 문서 수집을 하기 위해서는 동적인 링크(자바스크립트)를 처리 할 수 있어야 하며, 중복 URL처리 기술 또 한 매우 중요한 기술이라

할 수 있다. 이미 수집되었던 웹 페이지를 중복으로 처리 하지 않기 위해서 이다. 본 논문에서는 웹 로봇 구현에 대한 고려되어야 하는 다양한 사항들에 대해서 테스트 하여 구현해 보았다. 첫째는 단일수집 로봇과 복수수집 로봇을 구현하여 테스트 하였고, 또한 자바스크립트의 함수를 실행하여 그 결과를 페이지 주소로 구현하고 하였다.

III. 제안 웹로봇 모델 구현

A. 헤더분석 모델 구현

헤더분석은 단순히 HTTP소켓을 가지고 정보를 가져올 수 있으며, HTTP 헤더의 정보 중에 Content-Type의 내용이 text/html 일 경우 분석처리 모델로 내용을 전달하고 Application Data 경우, 다운로드 처리 모델로 전달하여 처리 하도록 하는 역할을 하기도 한다. 또한 접속이 불가능 할 경우 페이지 에러 처리가 가능해야 한다. 소켓이 에러 났을 경우 Try ... Except 구문으로 처리할 수 있다. HTTP소켓 에 러의 메시지 정보는
http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html' 에 상세히 표기 되어 있다. 네트워크의 상태에 따른 예외처리가 잘 되어 있지 않으면 잘못된 페이지를 처리 하기 위한 수집 모델이 Dead Lock 상태로 빠질 수 있기 때문에 상태에 따른 Excpet(예외) 처리가 잘 되어 있어야 한다. 페이지 요청 후 접속이 되었을 경우 헤더의 정보를 다운로드 받는 부분의 예외처리와 헤더 정보를 다운로드가 정상적으로 처리 된 후 문서를 다운로드 받는 부분의 예외처리 또한 되어 있어야 한다.

B. 수집 쓰레드 처리 모델

쓰레드(Thread)는 주로 프로세스(실행되고 있는 프로그램)에서 만들어진다. 만약, 하나의 데이터가 있다고 가정하면. 이 데이터를 서로 다른 방법으로 처리하면서 결과 값 을 주고받아야 합니다. 하지만 프로세스는 서로의 영역에 통신을 할 수 가없습니다. 그때 프로세스가 통신을 위해 파생시키는 것이 쓰레드(Thread) 이다. 쓰레드는 메모리의 같은 공간을 공유하고 있으므로 하나의 쓰레드(Thread)에 이상이 있으면 다른 쓰레드(Thread)에도 영향을 주게 됩니다. 대개의 경우 프로세스는 단독으로 실행이 가능하지만 쓰레드는 프로세스가 있고 그 하위에 쓰레드가 여러 개 생성되는 것이 일반적입니다. 같은 메모리 공간을 사용할 때 문제의 발생을 줄기 위해서 쓰레드 락(Thread Lock)이라는 프로그래밍 또한 가능하며 또한 쓰레드 방식은 2가지의 방식이 있는데, 싱글 쓰레드 방식과 멀티 쓰레드 방식이 있습니다. 독립적으로 1개의 메소드(Method)만 호출 하는 것을 싱글 쓰레드 방식 싱글 쓰레드 방식은 멀티 쓰레드 방식보다 처리 속도가 빠르며, 쓰레드 락(Thread Lock)에 대한 고려또한 필요가 없다. 2개 이상의 메소드(Method)를 동시에 호출 할 때에는 멀티 쓰레드 방식이라고 합니다. 멀티 쓰레드 방식은 쓰레드 리스트(Thread List)라는 큐를 가지고 있으며 싱글 쓰레드 방식과 멀티 쓰레드 방식으로 작업을 처리 하게 되면 싱글 쓰레드 보다 처리 속도가 늦습니다. 이유는 바로 CPU *Context Switching* 작업이 추가되기 때문이다. 90개의 웹 페이지를 생성하여 업로드 한 뒤에, 멀티쓰레드 방식과 싱글쓰레드 방식으로 테스트 한 결과 멀티 쓰레드 방식은 9.250 Sec 이 소요 되었으며, 싱글 쓰레드 방식은 15.578 Sec 이 소요 되었음을 알 수 있습니다. 만약 웹 사이트에서 네트워크 장애가 발생하였을 경우 싱글 쓰레드의 경우에는 소요시간이 더 지체 될 수 있다. 또한 점검하려는 대상의 사이트가 많아지는 경우에는 소요 시간의 차이가 더욱 커지게 된다.

C. 스크립트 처리 모델

쓰레드(Thread)는 주로 프로세스(실행되고 있는 프로그램)에서 만들어진다. 만약, 하나의 데이터가 있다고 가정하면. 이 데이터를 서로 다른 방법으로 처리하면서 결과 값을 주고받아야 합니다. 하지만 프로세스는 서로의 영역에 통신을 할 수 가없습니다. 그때 프로세스가 통신을 위해 파생시키는 것이 쓰레드(Thread) 이다. 쓰레드는 메모리의 같은 공간을 공유하고 있으므로 하나의 쓰레드(Thread)에 이상이 있으면 다른 쓰레드(Thread)에도 영향을 주게 됩니다. 대개의 경우 프로세스는 단독으로 실행이 가능하지만 쓰레드는 프로세스가 있고 그 하위에 쓰레드가 여러 개 생성되는 것이 일반적입니다. 같은 메모리 공간을 사용할 때 문제의 발생을 줄기 위해서 쓰레드 락(Thread

Lock)이라는 프로그래밍 또한 가능하며 또한 쓰레드 방식은 2가지의 방식이 있는데, 싱글 쓰레드 방식과 멀티 쓰레드 방식이 있습니다. 독립적으로 1개의 메소드(Method)만 호출 하는 것을 싱글 쓰레드 방식 싱글 쓰레드 방식은 멀티 쓰레드 방식보다 처리 속도가 빠르며, 쓰레드 락(Thread Lock)에 대한 고려또한 필요가 없다. 2개 이상의 메소드(Method)를 동시에 호출 할 때에는 멀티 쓰레드 방식이라고 합니다. 멀티 쓰레드 방식은 쓰레드 리스트(Thread List)라는 큐를 가지고 있으며 싱글 쓰레드 방식과 멀티 쓰레드 방식으로 작업을 처리 하게 되면 싱글 쓰레드 보다 처리 속도가 늦습니다. 이유는 바로 CPU Context Switching 작업이 추가되기 때문이다.

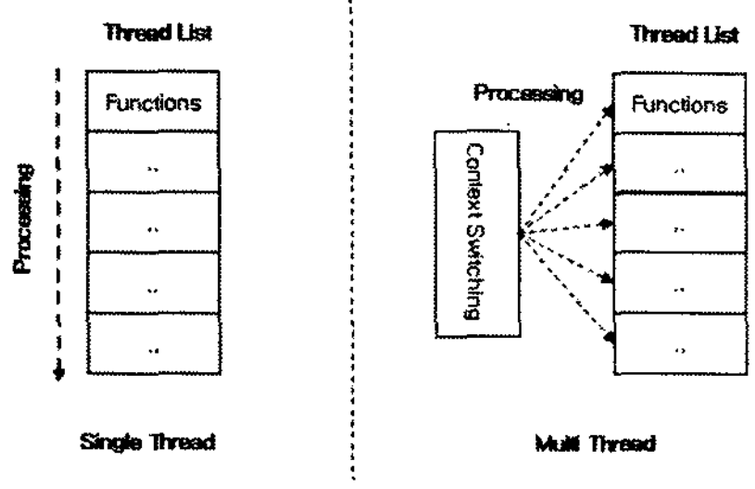

그림 1. 싱글, 멀티 쓰레드의 처리방법]

IV. 멀티쓰레드 방식과 싱글쓰레드 방식 처리 속도 비교

90개의 웹 페이지를 생성하여 업로드 한 뒤에, 멀티쓰레드 방식과 싱글쓰레드 방식으로 테스트 한 결과 멀티 쓰레드 방식은 9.250 Sec 이 소요 되었으며, 싱글 쓰레드 방식은 15.578 Sec 이 소요 되었음을 알 수 있습니다. 만약 웹 사이트에서 네트워크 장애가 발생하였을 경우 싱글 쓰레드의 경우에는 소요시간이 더 지체 될 수 있다. 또한 점검하려는 대상의 사이트가 많아지는 경우에는 소요 시간의 차이가 더욱 커지게 된다. 수집모델의 개수에 따른 처리 속도의 비교를 하였을 경우, 속도의 차이가 확연하게 차이가 나게 됩니다. 하지만 점검 하려는 웹 서버의 설정이나 성능에 따라서 문제가 발생 할 수 있습니다. 수집 모델이 많을 경우 IDS 나 IPS에서 차단을 당할 수 있으며 IDS나 IPS가 없을 경우 서버가 다운될 수 있으며 점검하려는 컴퓨터의 사양에 따라 많은 차이가 나게 됩니다.
수집모델의 개수에 따라 달라질 수 있으며 네트워크 상태에 달라질 수 있으므로 정확한 성능평가가 매우 어려웠으며, 이 문제를 해결하기 위해서 내부 네트워크에 연결된 서버의 웹 사이트를 대상으로 테스트 하였다.

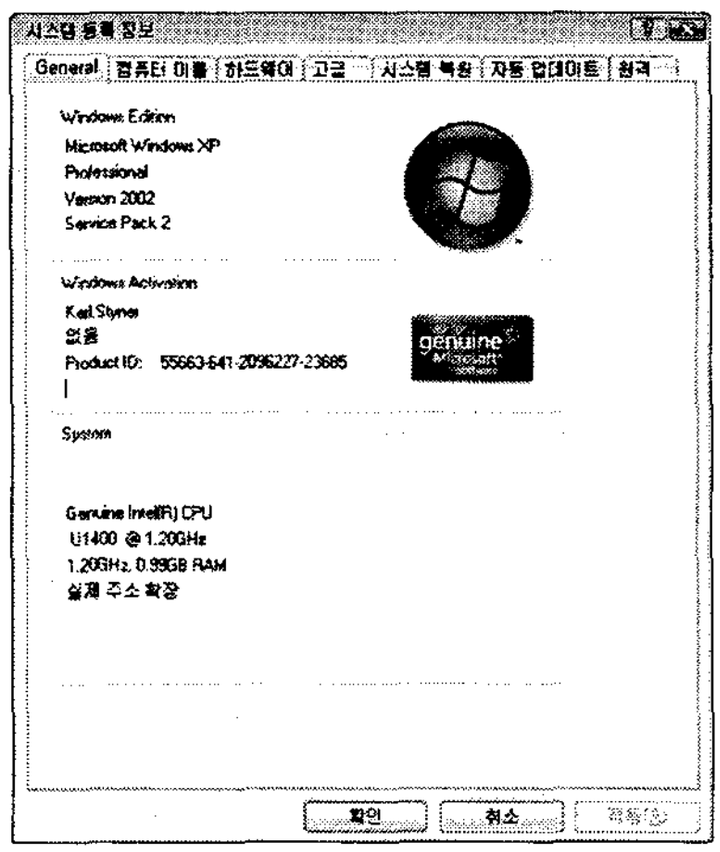

그림 2. 수집 모델 성능평가를 위해 사용된 컴퓨터정보

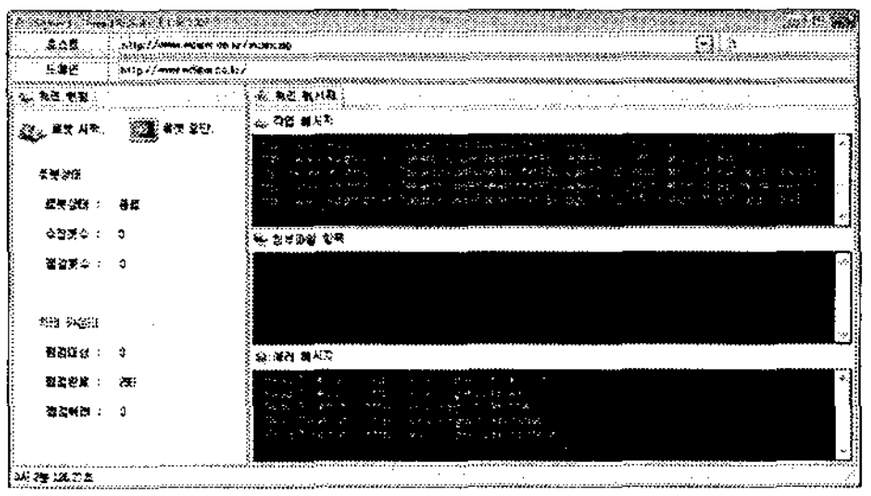

그림 3. 수집모델의 카운터를 1개로 설정

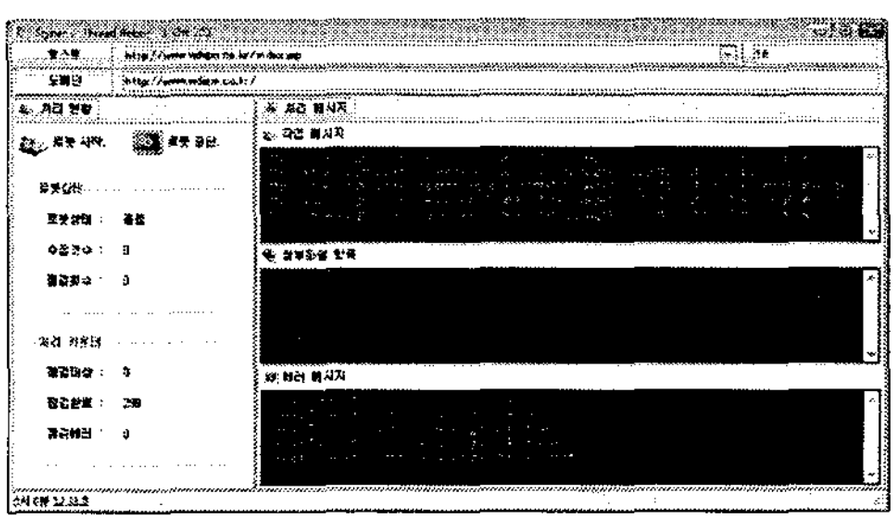

그림 4. 수집 모델을 10개로 생성으로 점검

수집모델을 1개로 설정해놓고 299개의 웹 페이지를 점검하였을 경우 2분 126.27초가 소요되었음을 확인 할 수 있었으며 수집 모델을 10개로 생성하여 점검하였을 경우 12.33초가 소요되었음을 확인 할 수 있었다.

웹 사이트에서는 문서와 문서와의 연결을 링크로 하게 됩니다. <a href="문서경로"> 로 표기 되는데, 이것을 하이퍼링크라고 합니다. 웹 로봇이 하이퍼링크를 추출하는 방법은 정규표현식으로 하게 되는데 하이퍼링크를 추출하는 정규 표현식은 그림 [3-3]과 같습니다.

이렇게 정규표현식을 사용하여 링크를 추출하게 되는데, 몇 가지 문제가 있을 수 있습니다. 그 문제는 바로 자바 스크립트 함수를 호출하여 사이트를 연결해서 사용 할 수 있기 때문입니다. <a href="javascript:this.href='./mokwon.php'"> 이와 같은 링크에서는 로봇에서 링크를 추출하지 못하는 문제가 발생 됩니다. 이런 자바스크립트를 위해서 웹 브라우져 객체와 MS MHTML 를 사용 하여 처리 할 수 있습니다. Microsoft에서 제공하는 MS HTML는 HTML파서의 기능을 가지고 있으며, Web 객체를 사용하기 위해서는 별도의 Embedded Web Componuts (http://www.bsalsa.com/intro.html)를 사용하였습니다. IHTMLWindow2에 Web 객체를 삽입한 뒤에, 스크립트를 실행하면 웹 객체가 BeforeNavigate 함수를 호출하게 되는데 이때에 URL정보와 PostData 정보를 가지고 자바스크립트 함수로 이동되는 URL을 알 수 있다.

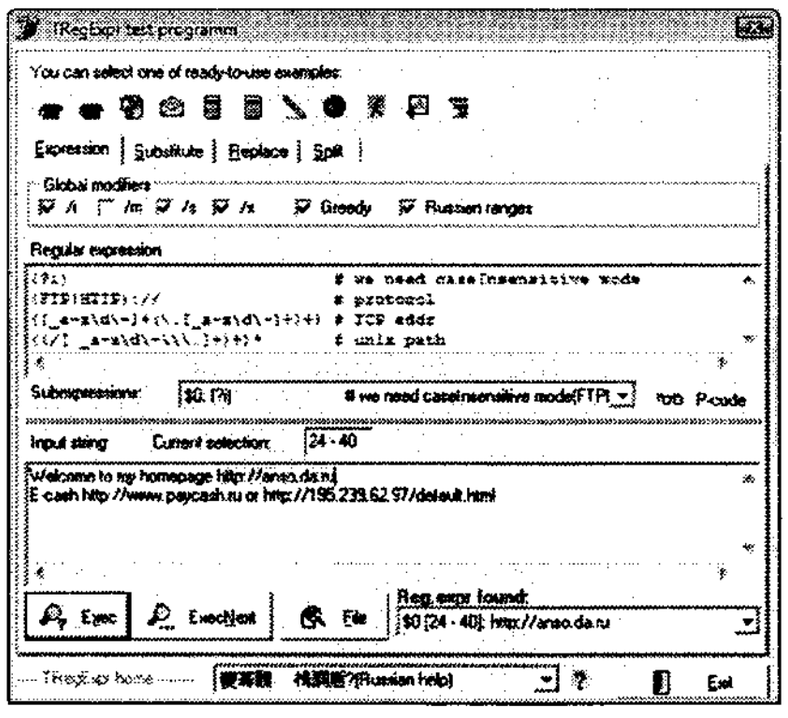

그림 5. 정규표현식을 통한 링크 추출

V. 실험 및 고찰

웹 로봇을 구현하기 위해서 사용되는 모델 중, 자바스크립트 처리 모델에서는 메모리 누수 문제가 있는데, 이 문제는 구현에 사용되는 Embedded Web Componuts (http://www.bsalsa.com/intro.html)는 자체적인 문제로 로봇을 제작 하는데 사용하기에는 문제가 있다. 또한 수집 모델에서 사용되는 쓰레드 모델 처리 후 웹 사이트를 분석하는 기타 알고리즘을 구현하는데 있어서 쓰레드로 수행되어도 문제가 없어야 하도록 구현해야 하므로 이러한 부분을 고려해야 할 것이다. 또한 CPU Core와도 많은 차이가 있었다. Single Core에서 싱글, 멀티 쓰레드를 돌렸을 경우에는 Single Core의 경우 속도 차이가 거의 없었으며, Dual Core 에서는 수집 속도의 차이가 조금 더 많이 났으며, Quad CPU의 경우에는 Dual Core 보다 더 많은 차이가 났다. 웹 로봇을 어떤 작업환경에서 동작시키느냐에 따라서 수집모델의 Thread Counter를 결정해야 했다. 또한 성능평가를 하기 위해서 여러 도메인(웹 서버)에 놓고 테스트를 하여야 했지만 수집 개수가 3개 이상 될 경우 해킹시도의 위험이 있으므로 할 수 내부의 네트워크에 연결된 서버에 점검해야 했으므로 정확한 성능을 평가 하는데 많은 어려움이 있었다. Single Core에서 싱글, 멀티 쓰레드를 돌렸을 경우에는 Single Core의 경우 속도 차이가 거의 없었으며, Dual Core 에서는 수집 속도의 차이가 조금 더 많이 났으며, Quad CPU의 경우에는 Dual Core 보다 더 많은 차이가 났다. 웹 로봇을 어떤 작업환경에서 동작시키느냐에 따라서 수집모델의 Thread Counter를 결정해야 했다.

VI. 결론

본 논문에서는 수집모델을 싱글과 멀티로 성능평가를 하여 웹 로봇을 구현하여 테스트 하였으며, 또한 웹 문서에 링크로 연결되어있는 자바스크립트 함수 처리 모델을 구현하였다. 하지만 앞으로 링크 수집 모델이 쓰레드 모델처럼 더 정형화 되어야 할 것이며, 플래시로 구현된 사이트의 경우에는 웹 로봇이 하이퍼링크를 수집 못하는 문제점을 가지고 있으며, 플래시의 링크를 추출하려면 링크추출 모델에 더 몇 몇 기능이

추가 되어야 하기 때문이다. 웹이 발전에 따라서 웹 수집 로봇도 더욱 발전해야 정확하고 빠른 로봇을 구현 할 수 있을 것이다.

참고문헌

[1] 웹 로봇의 성능 평가를 위한 방법론

[2] D. E. Denning. An intrusion detection model. *IEEE Transactions on Software Engineering, SE-13(2):222-232, February 1987.*