효율적인 문서수집을 위한 로봇엔진 스케줄링 기법

정용훈*, 김창근**, 박규석***
*,***경남대학교 컴퓨터공학과
**진주산업대학교 컴퓨터공학과

A scheduling technique of Robot engine for efficient page gathering

Young Hun Jung*, Chang Geun Kim**, Kyoo Seok Park***
*,***Dept. of Computer Engineering, Kyungnam University
**Dept. of Computer Engineering, Chinju National University

요 약

인터넷망의 규모가 빠른 속도로 성장함에 따라 검색 시스템의 문서 수집 시간 또한 그에 비례하여 늘어나게 되었다. 이로 인하여 정보검색의 속도에 상당한 문제점이 유발하게 되면서 로봇에 대한 연구가 필요하게 되었다. 본 논문에서는 기존의 로봇 스케줄링 기법을 보완한 혼합 Sorting 방법을 제시하며, 또한 기존의 무작위 방법과 Sorting 방법, 그리고 본 논문에서 제시한 혼합 Sorting 방법에 대한 문서 수집량을 시뮬레이션을 통하여 비교 평가하였다.

1. 서론

세계적으로 Internet의 규모 및 사용자의 범위가 커짐에 따라 Internet은 사용자간의 통신수단 및 중요한 정보의 원천으로 등장하게 되었다. Internet은 사용자를 포함한 다양한 정보자원이 전체 네트워크에 산재되어 있는 정보사회의 기반구조 중 하나로 볼 수 있으며, 이러한 자원을 사용자 측면에서 효율적으로 사용할 수 있도록 하기 위해서 나온 것이 검색 시스템이다.

검색 시스템의 장점은 사용자가 원하는 정보를 쉽게 찾을 수 있는데 있지만 인터넷망의 규모가 커져 감에 따라 검색 시스템 규모 또한 커지게 되었고, 검색로봇이 문서를 수집하는데 걸리는 시간도 갈수록 길어지고 있다. 검색엔진마다 차이는 있지만 보통 빠른 경우는 3-4주에서 길게는 몇 달이 걸려야만 검색 시스템에 등록되어 있는 도메인 서버의 문서를 수집할 수 있으며 등록되어 있지 않은 도메인 의 문서까지 수집한다면 한번 수집하는데 상당한 시간을 요하게 된다. 이로 인하여 최근의 정보인 경우는 찾고자 검색을 해도 나타나지 않으

며 사용자가 질의를 던졌을 때 검색된 내용중 대다수는 404 Not Found 에러가 발생하고 있다. 이런 문제를 해결하기 위한 방법으로 첫 번째는 지역별 검색 시스템을 구축하는 것이다. 예로 야후나 알타비스타의 경우 나라별로 검색 시스템을 구축하고 있다. 두 번째 방법은 멀티 로봇을 운영하는 것이다. 하지만 너무 많은 로봇을 운영하게 되면 그에 따른 부하도 같이 증가하는 문제가 발생한다. 본 논문에서는 효율적인 문서수집용 로봇엔진 스케줄링기법을 제시하며, 4-멀티 로봇을 이용하여 무작위로 문서를 수집했을 경우와 Sort기법을 사용했을 경우, 그리고 본 논문에서 제시한 혼합 Sorting 기법을 사용했을 경우의 문서 수집 양을 비교 분석한다.

2. 관련 연구

2.1 로봇 에이전트 구성도

로봇 에이전트란 원하는 정보를 얻기 위하여 웹상의 문서를 검색하고, 참조되는 문서들을 재귀적으로 검색하면서 웹의 하이퍼텍스트 구조를 자동으로 추적하여 정보를 저장해 주는 프로그램을 말한다. 로봇 에이전트의 구성은 그림1과 같으며, 각 모듈별 기능은 다음과 같다.

- 스케줄러
  스케줄러는 서버들의 네트워크 부하를 고려하여 시간대별 로봇이 수행해야 할 서버의 리스트를 작성한 후 대기큐가 비워 있다면 서버의 URL 정보를 리스트 순으로 대기큐에 보낸다.

- 로봇제어기
  로봇의 상태를 관리하는 곳으로 로봇이 휴지상태라면 대기큐로부터 수행해야할 서버 URL 주소를 가져와서 로봇을 활성화시킨다.

- 대기큐
  스케줄러에서 제공한 서버의 URL 주소를 임시적으로 저장하는 공간이다. 로봇제어기에서 URL을 요구시 제공한다.

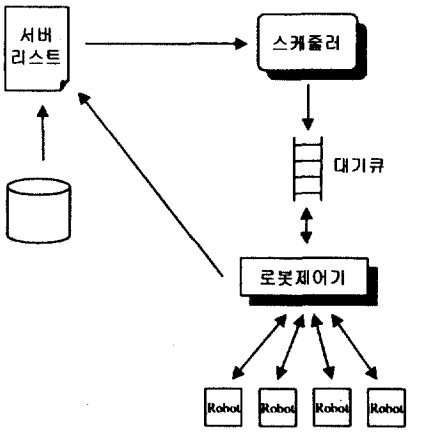

그림 1. 로봇에이전트 구성도

- 서버리스트
  서버 리스트는 검색 시스템에 등록되어있는 서버의 URL주소, 네트워크 부하, 상태정보를 가진다.

2.2 로봇 동작 알고리즘

로봇 동작 알고리즘은 다음과 같다.

1. 로봇 에이전트 프로그램을 구동한다. 기본적으로 처음 접근할 URL이 주어져야 한다.
2. URL의 호스트 이름에 따라 robots.txt에 접근한다.
3. robots.txt 파일의 내용을 받는다.
4. robots.txt 파일 내용을 분석하여 그 호스트에 서 접근 가능한 URL을 추출한다.
5. 로봇의 접근을 배제하는 사이트가 아니면 다시 사이트에 접근한다.
6. 해당 사이트에 접근하여 사이트의 문서를 수집한다.
7. 수집된 문서를 분석하여 사이트의 URL을 추출한다.

8. 수집한 문서를 분석해 키워드를 추출하고 필요한 정보를 저장한다.

위의 알고리즘은 그림 2와 같이 나타낼 수 있다.

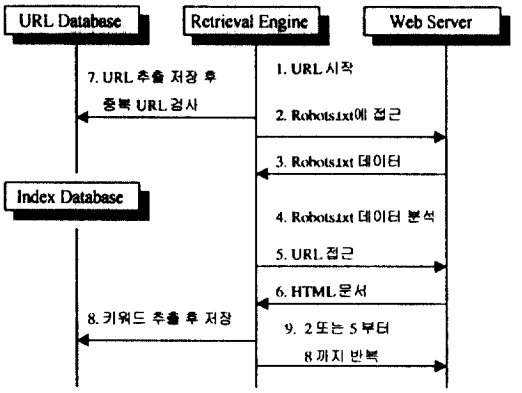

그림 2. 로봇의 동작 알고리즘

2.3 기존의 로봇 검색 스케줄링 기법

2.3.1 무작위 검색

방문해야되는 각 서버를 무작위로 선정하여 보내는 방법으로 서버리스트를 작성했던 순서대로 로봇을 보내게된다. 이 방법은 네트워크의 부하를 전혀 고려하지 않은 경우로서 초창기 검색 시스템의 경우 대부분 이 방법을 사용하고 있다.

2.3.2 Sorting 스케줄러

Sorting 방법은 서버들의 각 시간대별 네트워크 부하량을 주기적으로 검사, 평균을 내어서 이들 서버들의 네트워크 부하량을 오름차순으로 소팅 하여 시간대별로 수행해야할 서버 리스트를 작성하는 방법이다. 이 방법은 무작위 방법과는 달리 네트워크 부하를 고려함으로써 시간당 평균 문서 수집량의 효율을 높일 수 있지만 시간이 지날수록 그 수행 능력이 떨어져서 나중에는 무작위 방법보다도 수행 능력이 떨어지게 되는 단점이 있다.

3. 혼합 Sorting 스케줄링 기법

Sorting 방법에서의 단점은 현재시간에 어떤 서버의 네트워크 부하율이 낮다고 할지라도 가장 부하율이 낮다고는 말할 수 가 없다. 또한 보통 네트워크 부하량을 하루 주기로 보았을 때 그림 3에서와 같이 4가지 형태로 나눌 수 있다.

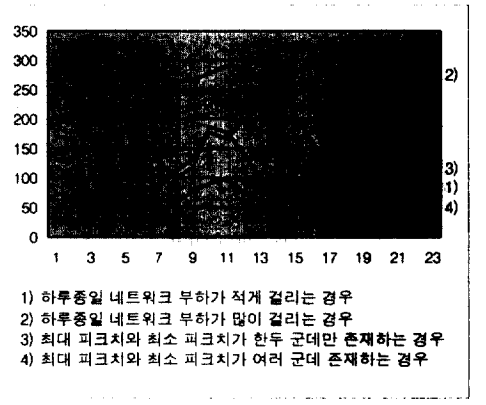

그림 3 네트워크 부하상태

그림 3에서 1)과 4)의 경우는 Sorting 스케줄링방법을 사용할 경우 대체적으로 최적의 상태에서 문서를 수집할 수가 있다. 그러나 3)의 경우 부하량이 최소치일 때 상대적으로 다른 서버들에 비해서 부하량이 높다면 그 시간대에 문서수집을 할 수 없게 된다. 경우에 따라서는 계속 Sorting 순서에 밀려서 최악의 상태일 때 문서를 수집하는 경우가 발생할 수도 있다.

2)의 경우는 전반적으로 부하량이 높기 때문에 Sorting 스케줄링 방법에서 문서수집 순서가 가장 뒤쪽으로 밀려나게되므로 네트워크의 부하를 전혀 고려할 수 가 없다. 특히 부하량이 높을 경우 문서 수집량도 그에 따라 급격히 줄어들게 된다.

따라서 본 논문에서는 2)와 3)의 경우를 어느 정도 해결할 수 있는 혼합 Sorting 기법을 제시한다.

혼합 Sorting 스케줄러 구성도 그림 4와 같으며 각

모듈의 기능은 다음과 같다.

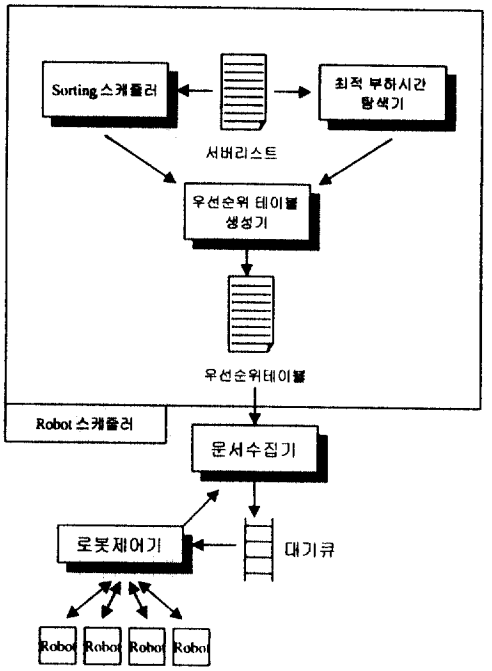

그림 4. 혼합 Sorting 스케줄러 구성도

· Sorting 스케줄러
기존의 Sorting 스케줄러와 동일한 기능을 갖는다.

· 최적부하탐색기
각 서버에 대한 네트워크 부하가 가장 적게 발생하는 시간대를 탐색한다.

· 우선순위 테이블생성기
최적부하탐색기와 Sorting 스케줄러로부터 얻은 정보를 가지고 우선 순위 테이블을 구성한다.

· 문서 수집기
문서 수집기는 우선순위 테이블로부터 문서를 수집해야 할 서버리스트를 입수하여 대기큐로 서버의 URL정보를 전송한다. 그리고 로봇 제어기로부터 수집된 문서를 관리하고 문서수집이 완료된 서버는 다음 시간대에는 제외되어야 하기 때문에 서버리스트 정보를 갱신한다.

· 우선순위 테이블
우선순위 테이블은 2차원 배열로 이루어져 있다. 열은 0부터 23까지 각 시간대를 나타내며 행은 각 시간대별 우선적으로 처리해야하는 서버번호를 저장한다. 이때 열은 각 서버의 부하량에 따라 오름차순으로 저장된다.

혼합 Sorting 기법의 스케줄링은 다음과 같다.

1. 최적부하시간 탐색기에서 서버리스트로부터 서버의 시간대별 네트워크 부하를 전송 받아 최소값 알고리즘을 사용하여 네트워크 부하가 가장 적게 발생하는 시간대를 찾는다.
2. Sorting 스케줄러에서 구성되는 정보와 최적 부하시간 탐색기에서 구한 정보로 우선순위 테이블을 구성하게 된다. 우선순위 테이블은 최적 부하시간 탐색기에서 구한 서버번호가 먼저 기록되며 그 뒤로 Sorting 스케줄러에서 구한 서버 번호를 기록하게된다.
   먼저 최적 부하시간 탐색기에서 구한 각 서버의 최적 시간대를 가지고 우선순위 테이블의 해당 시간대에 서버 번호를 기록한다. 이때 아직 그 시간대에 기록되어 있는 서버 번호가 없다면 첫 번째 행에 서버 번호를 기록하고 이미 그 시간대에 기록되어 있는 서버 번호가 있다면 그 시간대에 해당하는 서버 의 번호를 부하량에 따라서 오름차순으로 재배치한다.
3. 2단계가 끝나면 Sorting 스케줄러를 이용하여 각 시간대별로 우선순위 테이블에 기록되어 있는 서버를 제외한 모든 서버에 대한 네트워크 부하 순으로 Sorting 하여 우선순위 테이블에 기록한다. 기록시 2단계에서 구한 서버 번호가 이미 그 시간대에 기록되어 있다면 그 다음부터 기록한다.
4. 문서수집기는 해당 시간대에 우선순위 테이블에 기록되어 있는 서버 정보를 대기큐에 보낸다. 대기큐가 비워져 있다면 서버 정보를 계속해서 보

내고 대기큐가 꽉 찼다면 대기큐가 비워질 때 까지 기다린다. 그리고 로봇제어기로부터 수행이 완료된 서버 번호를 넘겨받아서 다음에 수행되지 않도록 그 정보를 기록해 둔다. 위 작업은 한시간 을 기준으로 하며 다음 시간으로 넘어갈 경우 우선 대기큐를 초기화한다. 그런 후 우선순위 테이블의 다음 시간대에 기록되어 있는 서버번호를 대기큐로 보낸다. 대기큐로 보낼 때는 이미 수행 완료된 서버인지 확인을 한다. 수행이 완료된 서버인 경우는 무시하고 그 다음에 기록되어 있는 서버번호를 대기큐로 보낸다.

5. 한 시간대에서 다음 시간대로 넘어갈 때 문서수집이 완료되지 않은 서버의 경우는 그 다음 시간대의 네트워크 부하와 그 이후의 네트워크 부하를 비교하여 다음 시간대의 네트워크 부하가 더 크다면 현재의 시점에서 검색을 중단한 후 중단된 위치 에 대한 정보를 저장하여 다음에 검색할 때는 그 중단점부터 검색을 시작한다.
6. 모든 서버들에 대한 문서수집이 완료됐다면 로봇 에이전트를 종료한다.

4. 시뮬레이션 및 평가

본 논문에서 제시한 스케줄러의 성능 평가를 위하여 시뮬레이터를 구성하여 시뮬레이션 하였다. 본 스케줄러를 위한 시뮬레이터의 환경은 다음과 같다.

- 100여개의 서버를 대상으로 며칠간 시간대별로 Ping을 날려 네트워크 부하를 구하였다.
- 모든 서버는 응답 가능하다는 가정을 하였다.
- 각 서버의 용량은 랜덤하게 발생 시켰다.
- 로봇의 문서 수집시 발생하는 부가적인 네트워크 부하는 무시하였다.

시뮬레이션에서 무작위 방법과, Sorting 방법, 그리고 본 논문이 제시한 스케줄러 방법을 동일한 조건하에 시뮬레이션 한 결과는 그림 5와 같다.

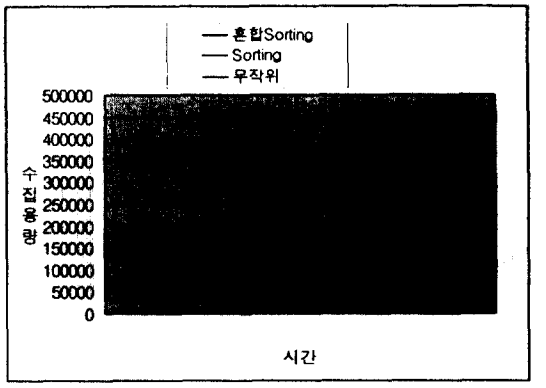

그림 5. 시뮬레이션 결과

결과를 분석한 결과 첫 시간대의 문서수집 능력은 Sorting 방법이 가장 우수한 것으로 나타났다. 하지만 시간이 갈수록 그 성능은 확연히 떨어짐을 알 수 있다. 혼합 Sorting법은 첫 시간대의 문서수집 능력은 Sorting법보다는 떨어지지만 둘째 시간대부터 9시간대까지는 수집능력이 뛰어남을 알 수 있다. 그 후로는 두 방법의 수집능력이 거의 비슷하게 나타났다. 최종 완료시간을 보면 무작위 방법이 26시간이 걸렸으며 Sorting 방법이 21시간, 혼합 Sorting 방법이 19시간이 소요되었다. 혼합 Sorting 방법이 시간이 가장 적게 소요되었으며, 무작위 방법보다는 Sorting방법이 약 26%의 향상을 가져왔고 혼합 Sorting 방법은 34%의 향상을 가져왔다.

5. 결론

본 논문에서는 기존의 방법보다 시간당 문서수집량을 향상할 수 있는 문서 수집용 로봇 엔진 스케줄링 알고리즘을 제시하였으며, 네트워크 부하를 고려하지 않은 무작위 방법과 네트워크 부하를 고려한 Sorting 방법, 그리고 본 논문에서 제시한 혼합 Sorting 방법에 대하여 시뮬레이션을 통하여 분석한 결과 혼합 Sorting방법은 기존 Sorting 방법보다는 약 6%의 성능 향상을 가져왔으며 혼합

Sorting 방법은 수행해야할 서버의 수가 많으면 많을수록 그 성능이 더욱 발휘될 수 있다. 특히 갈수록 늘어나는 도메인 서버를 고려할 때 상당히 유용하리라고 생각된다.

본 논문에서는 문서 수집시 발생할 수 있는 네트워크 부하를 고려하지 않았으나 앞으로는 이를 고려할 경우와 일부 서버가 다운되거나 네트워크 상태가 갑자기 좋지 않게된 경우의 로봇 동작에 대한 연구가 요구된다.

참고문헌

[1] 김일, "지역정보망을 위한 실시간 제어 검색 엔진의 설계 및 구현", 1998년 경남대학교 컴퓨터공학과 졸업논문

[2] 심해청, 김병만, 김태남, 이준호, 최윤수, "효율적인 웹 로봇의 설계 및 구현에 관한 연구", 1997 한국정보과학회 학술발표논문집 Vol. 24 No. 2

[3] 김강희, 박노삼, 조은희, 안광선, "웹 서버 접속 로그에서 웹 서버 부하 패턴 분석", 1998 한국정보과학회 봄 학술발표논문집 Vol. 25 No. 1

[4] 박민우, "로봇 에이전트의 설계와 구현", 1996.10 마이크로소프트웨어

[5] World Wide Web Robots, Wanderers, and Spiders(http://info.webcrawler.com/mark/projects/robots/robots.html)

[6] Junghoo Cho, Hector Garcia-Molina," Efficient Crawling through URL ordering", http://www.elsevier.nl:80/cas/tree/store/comnet/free/www7/1919/com1919/htm