웹 정보 검색 엔진을 위한 저장구조의 설계 및 실험

이동광, 안동언, 정성종
전북대학교 컴퓨터공학과

e-mail : dklee@duan.chonbuk.ac.kr
duan,sjchung@moak.chonbuk.ac.kr

Design and Study on Structure of Storage for Web Search Engine

Dong-Kwang Lee, Dong-Un Ahn, Sung-Jong Chung
Dept of Computer Engineering, Chonbuk National University

요약

인터넷의 발달은 월드 와이드 웹을 탄생 시켰고 그로 인한 인터넷의 폭발적 성장은 인터넷을 하나의 생활로 만들었다. 인터넷의 엄청난 자료의 양과 친숙해진 인터넷으로 인해 인터넷은 하나의 정보창구의 역할을 하게 되었고, 그에 따라 정보검색이 발전하게 되었다. 초기의 월드 와이드 웹은 많은 웹 문서가 아니었지만, 구글이 현재 20억 페이지를 색인할 만큼 엄청난 규모가 되었다. 또한 앞으로의 검색엔진은 요약정보나, 웹상의 링크 정보를 통한 그 문서의 중요도를 분석하여 문서의 중요도를 판단하게 될 것이며, 지금까지의 검색엔진의 저장구조와는 다른 구조를 가지게 될 것이다. 그에 따라 웹 정보검색엔진의 저장구조는 효율적 저장과 속도 향상을 위해 중요한 구조가 되어가고 있다.

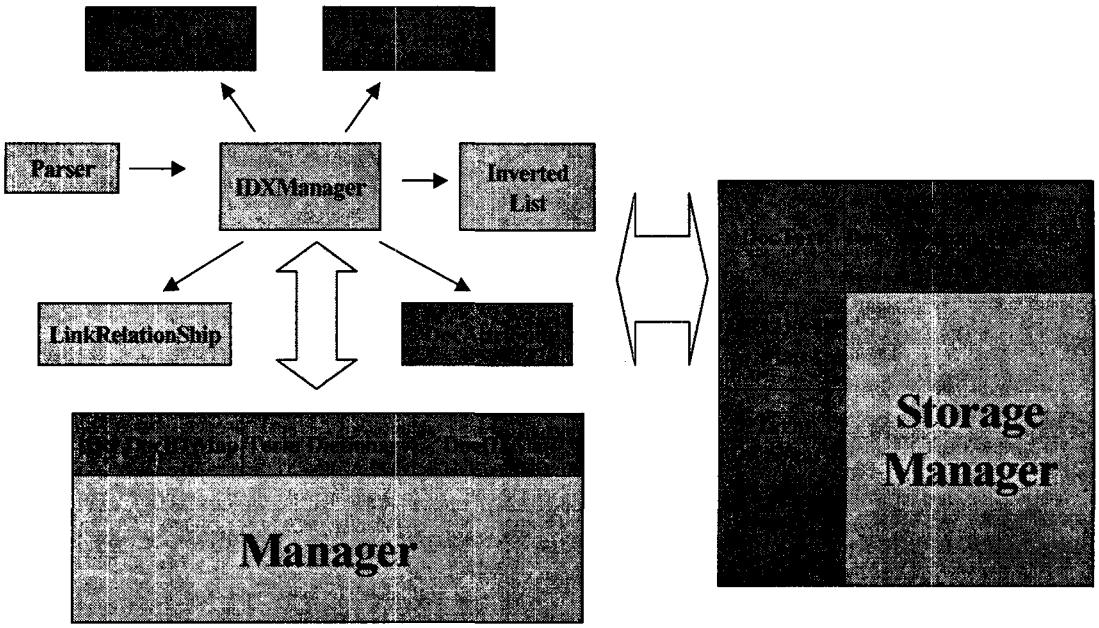
본 논문에서는 검색엔진의 저장구조에 따른 용량의 변화와 앞으로의 웹 검색엔진에서 등장할 기능인 문서의 요약 정보나, 문서간의 링크 정보를 통한 문서의 중요도 분석 등을 수행할 수 있는 저장구조를 만들어보고 실험해 보았다.

1. 서론

월드 와이드 웹(World Wide Web)은 1980년대 말에 시작되었는데[1], 아무도 현재와 같은 충격을 상상하지 못했으며, 이제 웹의 열풍과 그 잠재적 성장은 잘 알려져 있다. 오늘날 200개국 이상에서 4000만대 이상의 컴퓨터들이 인터넷에 연결되어 있으며, 그들 중 다수가 웹 서버 역할을 한다.[2] 우리는 이러한 웹을 흔히 '정보의 바다' 라고 표현하기도 한다. 또한 웹을 통한 정보 검색은 사용자들에게 가장 친숙한 정보 검색 방법이 되었다.[3]

현재 제일 많은 웹 문서를 색인하고 있는 Google의 경우 20억 페이지를, wisenut은 16억 페이지에 육박하는 웹문서를 색인하여 저장하고 있다. 우리는 다른 많은 웹 페이지에서 구글을 검색 사이트로 추천하는 것을 볼 수 있다.[3] 구글의 우수성이라 할 수도 있지만 그만큼 정보의 양이 중요함을 알 수 있다. 이러한 많은 웹 문서로 인해 검색엔진의 저장구조는 검색엔진을 위한 저장구조를 만들게 되었고, 속도와 효율적인 저장구조를 갖기 위해 발전되어 갔다.

현재까지의 대부분의 검색엔진은 질의어로 들어온 단어가 문서 내에서 중요하게 쓰이는 문서를 찾아서 순위를 매겨 보여주는 구조를 취해 왔었다. 하지만 앞으로의 검색엔진은 그 문서의 요약정보나, 웹 상의 링크 정보를 통한 그 문서의 중요도를 분석하여 순위에 반영하는 방법을 사용하게 될 것이다. 따라서 지금까지의 검색엔진의 저장구조와는 다른 구조를 가지게 될 것이며, 이러한 저장구조의 변화에 따른 분석 문서와 저장된 문서의 용량변화를 영문문서와 한글문서로 분류하여 살펴보자.



<그림 1> 시스템 구성

2. 정보검색 시스템 저장 구조의 설계

저장구조를 다음과 같이 4가지로 구분하였다.

① DocVector
② DocAttribute
③ DocText
④ LinkRelationShip

※ ③,④는 요약정보와 링크간 관계를 분석하기 위해 필요한 정보들이다.

2.1 DocVector

파서가 원 문서를 분석하여 얻어진 의미 있는 단어 정보(앞으로는 Term이라 한다)만을 저장한다.

Term을 문자열로 저장하고 처리할 경우 메모리 문제와 속도 문제가 발생하기 때문에 각 텀에 대응하는 숫자를 부여하는 Term Dictionary를 만들어 Term의 관리를 맡긴다. 각 Term이 파서로부터 넘어 올 때마다 Term Dictionary에게 Term정보를 요청하고 Term Dictionary에게서 받아온 Term에 대응하는 숫자만을 저장한다.

또한 각 Term의 원래 문자열을 DocText로부터 읽어오기 위한 정보인 문자열 시작 위치정보를 저장한다.

2.2 DocAttribute

한 문서의 기타 정보를 저장한다. 기타 정보에는 그 문서의 길이와 Term의 갯수, DocText, DocVector의 저장 위치 정보 등이다. DocAttribute 역시 Storage Manager에 저장되며 각 문서의 DocAttribute는 DocIDMap에서 저장한다.

2.3 DocText

2.1에서 소개된 DocVector와 연동되어 사용되는 구조이다. 파서가 원 문서를 분석하면서 그림이나, 글자의 크기, 색상정보 등을 제거하고 얻어진 원문서의 내용과 문장간의 연관 관계를 분석하여 문장 구분 정보만을 저장한다. 원문서 내용을 보관하여 직접 그 문서를 보지 않고도 문서의 내용을 보여줄 수도 있으며, 검색엔진에 요약 기능을 추가하는 등 앞으로의 검색엔진의 발전 방향에서 없어서는 안될 정보이다.

2.4 LinkRelationShip

문서간의 연관 관계를 계산하여 각 문서별 중요도를 계산하기 위한 저장 구조이다. 각 문서의 번호(URL을 기준으로 생성된 문서번호)와 그 문서에서 링크된 문서들의 번호가 저장되는 간단한 구조로 이루어진다. 모든 문서정보가 모아져야만 계산이 가능하기 때문에 인덱싱 과정에서는 자료만 모아두고 인덱싱이 끝난 뒤에 계산된다. 이 정보 역시 앞으로의

검색엔진을 위해 필요한 정보이다.

3. 정보검색 시스템 저장 구조의 구현

3.1 구현 환경

본 논문은 Pentium-Ⅲ 1Ghz CPU와 1GByte의 메모리의 Linux RedHat 7.2에서 구현되었다.

3.2 구조체

① DocVector 배열

데이터 타입	크기(byte)	변수명
unsigned int	4	TermID
unsigned int	4	Start

한 문서에 들어있는 Term들을 저장하는 배열의 구조로써, 각 Term의 ID를 저장하는 TermID와 각 텀의 실제 스트링의 시작위치와 길이를 저장하고 있는 Start로 구성된다.

② DocAttribute

데이터 타입	크기(byte)	변수명
char *	4 (+size)	DocURL
char *	4 (+size)	DocTitle
float	4	LinkScore
구조체	10	DocTextID
구조체	10	DocVectorID

각 문서의 URL정보를 저장하는 DocURL과 제목을 저장하는 DocTitle, 각 문서간의 연관관계를 통한 문서의 중요도 정보를 저장할 LinkScore, ① DocVector와 다음에 나올 ③DocText가 저장된 위치정보를 저장할 DocTextID, DocVectorID로 구성된다.

③ DocText

데이터 타입	크기(byte)	변수명
unsigned int	4	TextSize
char *	4 (+TextSize)	Text

현재 문서의 크기를 저장하는 TextSize와 문서를 저장한 메모리인 Text로 이루어진다.

④ LinkRelationShip

데이터 타입	크기(byte)	변수명
unsigned int	4	URLID
unsigned int	4 * 갯수	Link_URLID

각 문서의 URLID와 그 문서 내에서 나타난 링크 정보들의 배열인 Link_URLID로 구성된다.

4. 실험

실험은 다음과 같은 2가지 조건을 영문 문서와 한글 문서로 나누어 분석하였다.

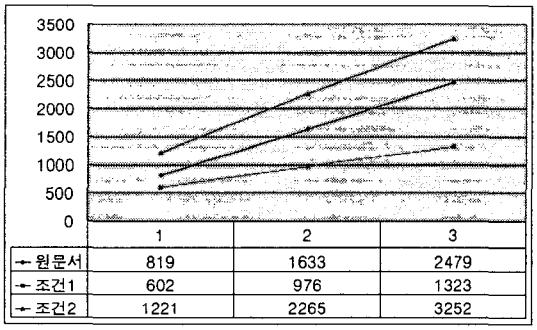
(조건1). 일반적인 검색엔진을 위한 저장구조

(조건2). 요약, 링크 정보 분석 기능을 위한 저장구조

조건1의 저장구조는 DocVector배열의 구조에서 Start와, DocText의 크기, LinkRelationShip 정보가 제거된다. 조건2의 저장구조는 앞 3.2에서 소개된 모든 정보가 포함된다.

실험에 사용된 데이터는 영문의 경우 TREC 데이터로써 CD1의 내용이며, 용량은 800M, 1.6G, 2.4G로 나누어 비교하였다. 한글 문서는 ac.kr로 끝나는 한국내 대학교 홈페이지의 웹 문서이며, 용량은 영문문서와 마찬가지로 800M, 1.6G, 2.4G로 나누어 비교하였다.

4.1 영어문서(TREC 데이터)

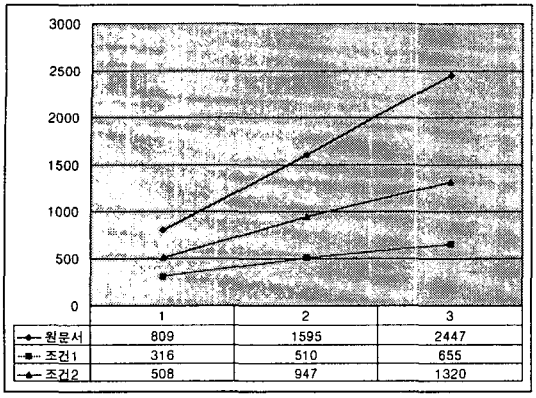


TREC의 분석 결과 중 조건2의 경우 원문서보다 많은 저장 공간이 필요하였으며 조건1의 2배가 넘는 용량을 필요로 하였다.

[표 1] 영문문서(TREC) 분석 결과 (단위 Mbyte)

영 문	819	1633	2479
MapManager	257.80	274.48	298.10
Inverted list	151.48	303.11	433.67
LinkRelationShip	1.17	2.28	3.27
DocAttr	25.51	51.01	73.22
조건1 DocVector	167.73	347.65	518.58
조건2 DocVector	335.46	695.29	1037.17
DocText	449.59	939.28	1406.87

4.2 한글문서



한국내 대학교 홈페이지 문서의 분석결과는 원문서보다 낮은 저장 공간을 필요로 하였으며, 역시 조건1의 필요 용량의 2배에 가까운 저장공간을 조건2는 필요로 했다.

[표 2] 한글 문서 분석 결과 (단위 Mbyte)

한 글	809	1595	2447
MapManager	197.73	233.69	254.33
inverted list	53.45	125.43	186.18
LinkRelationShip	0.58	1.34	1.94
DocAttr	12.85	30.92	48.25
조건1 DocVector	52.78	120.62	166.31
조건2 DocVector	105.56	241.24	332.63
DocText	138.37	314.76	497.28

4.3 분석

영문문서의 테스트를 위해 사용한 TREC데이터의 경우 자바 스크립트, 태그 정보가 없는 그 내용에 충실한 웹문서 였다. 하지만 한글의 분석 데이터로 사용한 문서는 자바스크립트, 기타 태그정보, 그림 등이 많은 요즘 일반적으로 볼 수 있는 웹 문서 였다. 이러한 데이터의 차이 때문에 영문과 한글 문서에의 실험데이터가 원 문서 용량을 넘는 경우도 생겨나게 된 것이다. 하지만 조건2에서 필요로 하는 용량이 조건1에서 필요로 하는 용량의 2배정도가 된다는 것을 확인 할 수 있었다. 또한 MapManager의 경우 처음엔 엄청난 양으로 증가하다가 그 증가량이 감소하는데 이것은 처음엔 대부분의 텀이 새로운 텀이기 때문에 증가만을 하지만 어느 정도 텀이 모이면 중복되는 텀이 많기 때문에 일정시간이 지나면 텀의 증가는 거의 일어나지 않고, 비교적 적은 용량인 문서 번호와 URL정보만 증가하게 된다.

5. 결론

웹은 앞으로 더욱 발전해 나갈 것이며, 생활과 더욱 밀접해질 것이다. 사람들은 좀더 가까워진 웹을 통해 많은 정보를 얻을 것이며, 웹 정보 검색 보다 나은 결과를 얻기 위하여 발전해 나아갈 것이며, 더욱 많은 정보를 필요로 할 것이다.

본 논문은 그 일부로 요약정보, 링크분석에 필요한 정보를 포함시켜 보았다. 표1, 표2에서 볼 수 있듯이, 조건2의 저장공간에 큰 영향을 미치는 것은 DocText와 DocVector로 전체 데이터의 2/3의 용량을 차지하였다. 따라서 앞으로 이 두 요소의 용량을 줄일 수 있는 방안에 대하여 연구가 필요할 것이다.

참고문헌

[1] Tim Berners-Lee, Robert Cailliau, Ari Luotonen, Henrik Frystyk Nielsen, and Arthur Secret. The World-Wide Web. Communication of the ACM, 37(8):76-82, 1994

[2] 김명철, 김덕봉, 김유성, 김재훈, 박혁로, 이하규 공역. (한국어판) 최신정보검색론 홍릉과학출판사

[3] 강인호, 김여진, 한영석, 김길창. Ergodic Markov Model을 이용한 정보 검색 모델. 제 13회 한글 및 한국어 정보처리 학술대회 논문집

[4] http://www.google.com

[5] http://www.wisenut.com

[6] http://lycos.co.kr

[7] G.salton and M. McGill "Introduction to Modern Information Retrieval", McGraw-Hill, New York, NY, 1983

[8] 류근호, 김진호 공역. 정보검색. 시그마프레스

[9] Jaime Carbonell and Jade Goldstein, "The use of MMR, diversity-based reranking for reordering documents and producing summaries," in Proceedings of the 21st ACM-SIGIR International Conference on Research and Development in Information Retrieval, Melbourne, Australia, 1998.

[10] Ray R. Larson s Lecture Notes of Principles of Information Retrieval.

[11] Prabhakar Raghavan s Lecture Notes of Principles of Information Retrieval.