한국컴퓨터정보학회 하계학술대회 논문집 제18권 제2호 (2010. 7)

링크 분석에 기반한 웹 문서 중요도 평가 알고리즘의 구현

임성채°
°동덕여자대학교 컴퓨터학과
e-mail: sclim@dongduk.ac.kr

An Implementation of the Ranking Algorithm for Web Documents based on Link Analysis

Sung Chae Lim°
°Dept. of Computer Science, Dongduk Women's University

● 요 약 ●

웹 검색에는 기존의 정보검색(Information Retrieval) 시스템에서와 다르게 문서 간 하이퍼링크 정보를 바탕으로 각 웹 문서의 고유 중요도를 추정하는 방식이 자주 이용된다. 링크 분석에 기반한 알고리즘 중 PageRank 알고리즘은 구글의 웹 검색 서비스에 적용된 것으로 알려져 있다. 이런 PageRank 알고리즘에 따라 중요도를 계산하는 경우 색인된 웹 문서수가 증가함에 따라 계산에 필요한 CPU 자원의 사용도 함께 증가하며, 문서 수가 수 억 페이지에 달하면 하나의 서버에서는 계산을 수행할 수 없다는 문제가 있다. 본 논문에서는 이런 문제점을 해소하기 위해 여러 대의 서버를 PageRank 계산 용 클러스터로 사용할 수 있는 방법을 제시한다. 제시된 방법은 고속의 LAN을 이용하여 여러 대의 서버를 연결하고 반복적인 행렬 계산을 병렬로 수행할 수 있어 계산 시간을 단축시킬 수 있다. 이런 서버 클러스터 구현을 위해 멀티 쓰레딩 프로그램이 작성되었으며, PageRank 계산에 사용되는 행렬 데이터를 적은 양의 메모리만으로 표현 가능하도록 하였다.

키워드: Page Rank, 링크 분석(link analysis), 웹 검색(Web searches)

I. 서론

인터넷의 광범위한 확산과 함께 지난 10여 년간 웹을 통해 접근할 수 있는 문서의 수가 기하급수적으로 증가하였다[1, 2, 6, 7, 8]. 이런 웹 문서의 증가로 필수적으로 요청되는 정보처리 기술은 수 백 억 페이지에 달하는 웹 문서 중에서도 정보 가치가 높고 사용자가 보고자 하는 문서를 빠르게 찾아주는 기술이었다[2]. 현재까지 이런 사용자의 요구에 가장 잘 부합하는 검색 효율을 제공하는 것은 Google이라고 하는 회사의 검색 서비스라고 할 수 있다[3, 4, 11]. 90년대 중반 웹 검색 서비스를 시작한 Google은 당시 다른 검색 서비스와 구별되는 링크 분석을 통한 문서 평가 기법을 통해 매우 특별한 검색 서비스를 제공할 수 있었고 이를 통해 검색어 광고 시장에서 지속적인 성공을 얻을 수 있었다.

구글 이전의 웹 검색 시스템에서는 IR(Information Retrieval)에 기반을 둔 문서 중도도 평가 기법에 기초하였다. 즉, 문서에 나온 키워드(혹은 단어)의 중요도 나 출현 횟수에 기반을 둔 시스템이었다. 이런 IR 기반 문서 중요도 평가 시스템은 색인한 문서가 충분한 정도의 키워드 집합을 가질 때 정확도가 향상되는 경향이 있다. 또한 IR에서는 하이퍼텍스트 개념의 링크 연결이 고려되지 않기 때문에, 웹 문서의 중요도를 암시해 줄 수 있는 링크 연결이란 요소를 반영할 수 없었다[4,8]. 기존 IR 분야에서 주 관심이 되는 문서에 비해 포함하고 있는 키워드 양이 매우 적은 웹 문서에 대해서는 기존 IR 분야에서 사용했던 키워드 기반의 문서 중요도 평가 방식으로는 문서 간의 중요도를 가리기 매우 어려웠다. 따라서 웹 검색 시 사용자가 찾고자 하는 웹 문서를 얻기 위해 상당한 시간이 요구되었고, 이런 불편함을 상당 부분 해소한 검색 서비스가 구글의 서비스라고 할 수 있다.

구글 문서 중요도 계산에 있어 핵심적인 요소인 링크 분석 기법은 PageRank라는 알고리즘으로 알려져 있다. PageRank 알고리즘의 아이디어는 복잡하지 않다. 어떤 임의의 웹 문서가 다른 웹 문서로부터 링크 연결을 많이 받고 있다면 그 문서는 상대적으로 중요한 문서라고 생각될 수 있다는 것이다. 이와 함께, 그런 중요성은 링크 연결을 주고 있는 다른 문서의 중요도가 클 때 링크를 받고 있는 문서의 중요도 역시 함께 커진다는 생각이다. 이런 각 문서의 중요도는 기존 IR에서와는 다르게 키워드와 관련 없이도 각 문서의 고유한 중요도를 부여할 수 있었다.

PageRank의 기본 아이디어는 매우 명료하고 단순해 보이지만 실제 이를 바탕으로 각 문서에 대해 중요도 값을 얻는 것은 또 다른 문제일 수 있다. 가장 기본적인 것은 PageRank를 적용할 수 있을 정도의 충분한 웹 문서를 수집해 내야 한다는 것이다. 어떤 특정 분야나 특정 사이트에 치우쳐 문서가 수집되는 경우 정확한 PageRank 값을 구해낼 수 없다. 또한 웹 에는 중복 사이트나 가비지 페이지들이 많을 수 있기 때문에 이에 대한 적절한 전처리가

요구된다[6] 이런 것에 대한 전처리를 통해 PageRank 계산에 적합한 데이터를 준비해야 한다. 이런 과정을 통해 얻은 링크 연결 정보를 바탕으로 PageRank 계산이 수행될 수 있다[7, 8].

본 논문에서는 이렇게 준비된 페이지 간 링크 연결 정보를 사용하여 반복적(repetitive)인 2차원 행렬 계산을 통해 PageRank를 계산하는 실제적 방법에 관한 것이다. 웹 문서간의 링크 연결 정보는 2차원 행렬로 표현되고 이렇게 구한 2차원 행렬에 대한 사칙연산을 통해 PageRank 값이 일정 정도의 오차 범위 내에서 수렴할 때까지 반복적으로 계산하는 것이 필요하다. 이때, 2 차원 행렬의 크기는 계산할 웹문서의 수에 비례하게 되는데, 웹 문서가 수억 페이지를 상회하는 상황에서는 계산량이 매우 많아 계산의 반복 횟수가 많은 경우 너무 길어지는 계산 시간으로 인해 실제 웹 검색 시스템에 적용하기 어려울 수 있다[8, 10]. 따라서 효과적인 계산 기법이 요구되며, 본 논문에서는 실제 국내 웹 검색 시스템에서 적용하여 구현된 바 있는 시스템을 기반으로 이에 대한 효과적인 방법을 제시한다. 구현된 기법은 PageRank 계산에 적합하도록 링크 연결 행렬을 분할(partition)하고 행렬 연산을 여러 대의 독립된 서버에서 병렬적으로 수행되도록 한다. 이런 분할을 통해 매우 큰 규모의 행렬이라고 하여도 메모리로 적재할 수 있도록 하였고, 병렬 계산을 통해 PageRank 계산에 필요한 시간을 최소화할 수 있었다.

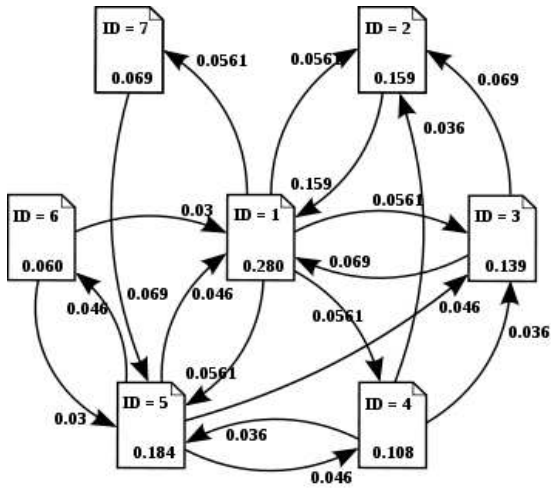

그림 1. 웹 페이지 간 링크 연결의 예.
Fig. 1. example of hyperlinks among web pages.

본 논문의 구성의 다음과 같다. 2장에서는 PageRank의 개념을 소개하고 3장에서는 구현된 서버 시스템에 대해서 설명한다. 그리고 4장에서 결론을 맺는다.

II. 관련 연구

하이퍼링크 연결 정보를 사용한 웹 문서 중요도 계산은 Kleinberg[5]의 논문에서 유래한다고 할 수 있다. [5]에서는 authoritative와 hub 사이트를 구하고 이를 중심으로 각 페이지의 중요도를 구할 수 있다. 하지만 [5]의 아이디어는 계산될 웹 페이지 수가 매우 큰 상황에서는 구현상의 문제가 있었다. 이어 반해 PageRank[] 기법은 행렬 계산을 통해 구현이 그리 어렵지 않다는 장점이 있다[9, 10, 11].

그림 1은 PageRank 알고리즘의 아이디어를 보인[3]다. 그림 1은 모두 7개의 웹 문서가 서로 연결된 모습을 보이고 있으며, 이들 7개 문서를 제외한 문서로의 링크 연결 부분은 생략되었다. PageRank 알고리즘은 확률 모델에 기반을 둔 아이디어이며, 임의의 시점에 상상의 웹 서퍼(surfer)가 각 웹 페이지에 확률로 머물러 있을 지를 구하는 것이다. 이런 페이지 접근 확률이 곧 PageRank 값으로 사용된다. 그림 1에서 ID 값은 해당 문서에 부여된 유일한 번호이고, ID의 아래 소수 값이 예상되는 문서의 접근 확률값이다. 화살표는 링크 연결이며 부여된 실수값은 해당 링크를 따라 서퍼가 이동할 확률이다. 이런 확률 모델에 따르면 ID 값이 2인 문서의 경우 문서 1, 3, 4로부터의 링크 연결을 가지고 있으며 이들을 고려한 접근 확률은 0.029159(=0.28*0.056 + 0.139*0.069 + 0.108*0.036)이다. 여기서 어떤 웹 페이지가 k개의 링크가 있다면 각 링크를 따라 이동할 확률은 모두 동등한 것으로 보고, 각 링크를 따라 이동할 확률을 1/k로 계산한다.

III. 구현된 시스템

1. 행렬 계산식

아래 식 (1)은 임의의 페이지 u에 대한 PageRank 값을 구하는 식이다. 식에서 $N_o(v)$는 페이지 v의 외부참조링크(outgoing link) 수를 나타내며, $B_i(u)$는 페이지 u로 가는 링크를 가진 모든 페이지 집합을 나타낸다. 아래 식 (1)은 PageRank의 기본 개념을 잘 보이고는 있지만 실제 계산에 있어서는 외부로부터 링크 연결을 전혀 가지고 있지 않는 독립된 페이지들로 인해 문제점을 가진다.

$$PR(u) = \sum_{v \in Bi(u)} \frac{PR(v)}{N_o(v)} \quad \text{----------} \quad (1)$$

이런 점을 고려해서 PageRank 계산에서는 댐핑 인자(damping factor)를 고려하며, 이는 가상 서퍼가 링크를 따라 웹을 보는 대신 임의의 웹 페이지로 직접 접근할 수 있는 상황을 고려했다고 할 수 있다. 이런 댐핑 인자를 고려한 것이 식 (2)에서 d의 값은 0보다 크고 1보다 작은 값을 가지며 N은 색인된 전체 웹 페이지의 수를 나타낸다.

$$PR(u) = \frac{1-d}{N} + d \sum_{v \in Si(u)} \frac{PR(v)}{N_o(v)} \quad \text{----} \quad (2)$$

식 (2)에서는 모든 웹 페이지가 외부로 나가는 링크 연결이 있는 것처럼 모델링되며, 식 (2)를 표현하는 2차원 행렬을 사용하여 각 페이지의 PageRank 값을 구할 수 있다. 행렬은 전체 웹 페이

지 수 N에 대해서 N x N의 행렬로 표현되며, 이는 식 (3)의 행렬 M을 이용하여 전체 페이지들의 PageRank 값인 N x 1 행렬 R을 구할 수 있다. 아래 행렬 계산을 하는 과정에서 R의 변화값이 일정 수준 이하로 내려갈 때까지 행렬 R을 반복 계한해 가면 PageRank를 얻을 수 있다.

$$R = \begin{bmatrix} (1-d)/N \\ (1-d)/N \\ \cdots \\ (1-d)/N \end{bmatrix} + d \times M \times R \quad \text{---------- (3)}$$

2. 구현된 클러스터링 시스템

앞에서는 PageRank 계산에 사용되는 식 (3)을 소개했다. 식 (3)에서 2차원 행렬 M은 각 페이지에 대한 외부참조 링크 정보를 담은 희소(sparse) 행렬이다. 이 행렬의 각 원소는 실수값이며 식 (3)의 계산을 수십 번 이상 수행해야 하기 때문에 하나의 서버에서 수행되는 경우 며칠의 CPU 시간이 소요된다. 이런 시간은 색인된 페이지가 커질수록 빠르게 증가한다. 또한 행렬을 모두 주기억 장치에 올려야 하기에 주기억 장치 공간 부족에 의한 문제 또한 심각하다.

이런 문제를 해결하기 위해서는 여러 개의 서버 클러스터를 구성하고 클러스터에 속한 두 개 이상의 서버에서 행렬 계산을 병렬 수행해야 한다. 그림 2는 구현된 클러스터링 시스템의 구조이다. 그림에서 사용된 서버는 4대이며, 이런 서버 수는 필요에 의해 늘릴 수 있다. 이들 서버는 1 Gbps LAN에 연결되어 있고 TCP 연결을 통해 자신이 수행한 계산 결과를 다른 서버들로 전송할 수 있다. 다른 서버들로부터 수집된 계산 결과를 이용하여 식 (3)의 계산을 다시 수행한다. 이런 계산은 새로 계산된 R의 값이 계산 전의 R 값과 차이가 주어진 임계값 보다 작을 때까지 반복 수행된다. 여기서 행렬 R의 차이는 행렬 R에 속한 각 원소들의 차의 절대값을 모두 더한 값이다.

그림 2의 각 서버는 행렬 R의 각 파티션에 대해서 PageRank 값을 계산한다. 이 때 파티션은 색인된 웹 문서를 나눈 것으로 동등한 개수로 나뉜다. 임의의 서버 Si가 k 번째 계산하는 R의 파티션을 $Ri(k)$라 할 때, $Ri(k)$를 구하기 위해서는 이전에 구한 랭킹 값인 $Rj(k-1)(j = 1, 2, 3, 4)$이 필요하다. 이 값은 다른 서버들로부터 얻는 값들이며, 이에 대한 값은 그림 2의 TCP 연결을 통해 얻을 수 있다.

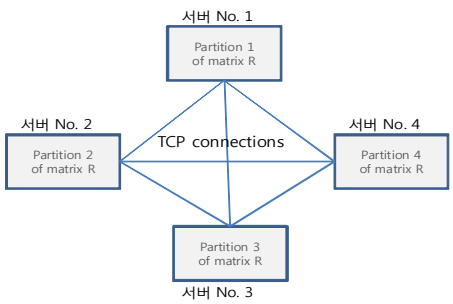

그림 2. 서버 클러스터의 구성도
Fig. 2. architecture of the server cluster

그림 2에서와 같은 서버 클러스터링을 통해 계산 시간이 서버 계수에 비례해서 감소함을 알 수 있었다. 이는 식 (3)은 페이지 별로 파티션되어 계산될 수 있음을 이용한 것이다. 이런 파티션 기법을 통해 웹 페이지 수가 크게 증가하는 상황에서도 간단히 서버 클러스터의 크기를 크게 함으로써 PageRank 계산에 드는 시간이 크게 증가하는 것을 막을 수 있었다.

R을 구함에 있어 메모리 공간이 많이 차지하는 것은 식 (3)의 M이라 할 수 있다. 실제 M이 희소 행렬임을 감안하여 행렬을 압축해서 표현할 수 있다. 실제 다른 페이지와의 링크 연결 개수는 전체 페이지 개수에 비해 매우 극소수이기 때문에 연결이 없는 행렬 요소가 대부분이기 때문에 행렬 압축을 통해 메모리 크기를 크게 줄일 수 있다. 이와 함께 행렬 M도 어떤 서버의 전체가 메모리에 올라와야 하는 것은 아니다. M 역시 수평으로 데이터를 파티션한 뒤에 하나의 파티션을 메모리로 적재하여 행렬 계산을 부분 수행하고, 다시 다음 M의 파티션을 메모리로 적재하여 수행함으로써 메모리 사용량을 줄일 수 있고, 한 번에 많은 양이 데이터를 디스크로부터 읽어 들일 수 있기에 전체 디스크 시간을 단축할 수 있었다.

그림 2의 계산을 보다 효과적으로 하기 위해 크게 쓰레드는 행렬 계산을 위한 쓰레드와 다른 서버로 부터의 행렬 값을 수집하고 자신이 속한 서버에서 계산한 결과 값을 다른 세 개 서버로 전송하는 통신을 위한 서버 쓰레드로 크게 나눌 수 있다. 행렬 계산을 위한 쓰레드는 다시 실제 행렬 계산을 수행하는 쓰레드가 계산에 필요한 행렬 M의 파티션을 디스크로부터 적재해 주는 쓰레드로 구현된다. 또 통신을 위한 쓰레드 들은 서로 네트워크 상에서 동기화되며, 전제 서버에서 $R(k+1)$의 계산이 완료된 시점을 파악한다. 완료된 이후에는 해당 번째의 $R(k+1)$ 값과 $R(k)$의 값의 차이를 수집하고, 이 차이가 일정 수준 이하로 내려간다면 계산이 종료된다.

Ⅳ. 결론

본 논문에서는 웹 페이지 검색에 있어 랭킹 정확도를 높이기 위해 사용되는 하이퍼링크 정보에 따라 웹 페이지 중요도를 결정할 수 있는 PageRank 알고리즘을 소개하고 이를 실제 구현할 때의 문제점에 대해서 기술하였다. 이런 문제점을 해결할 수 있도록 구현된 서버 클러스터링에 대해서 설명하였다. 구현된 기법은 PageRank 알고리즘을 행렬 계산식으로 표현하고, 이렇게 행렬 계산식으로 표현된 계산 방식의 특성을 고려하여 전체 계산을 병렬화하여 계산할 수 있음을 이용한다. 이를 위해 PageRank 구하고자 웹 페이지를 파티션하고 각 파티션에 대한 행렬식을 수행할 수 있도록 하였다. 이를 위해 고속 통신망에 연결된 서버 그룹이 서로 행렬 계산 값을 주고받도록 했으며, 상호 동기화되어 반복적인 행렬 계산을 병렬적으로 수행할 수 있도록 했다. 이를 통해 PageRank 계산 시에 문제가 될 수 있는 지나친 계산 시간문제를 없앨 수 있었다. 구현된 방식은 단순히 서버를 추가함으로써 클러스터의 확장이 가능하기 때문에 PageRank 값을 계산할 웹 페이지위 수가 증가할 때도 매우 효과적으로 대처할 수 있다는 장점이 있다.

참고문헌

[1] Search Engine Report, Http://www.searchenginewatch.com, 2010.

[2] 임성채, 계층적 캐시 기법을 이용한 대용량 웹 검색 질의 처리 시스템의 구현, 정보과학회논문지: 컴퓨팅의 실제 및 레터, Vol. 14(7), pp. 669-679, 2008.

[3] PageRank: in Wikipedia, Http://en.wikipedia.org/wiki/PageRank, 2010.

[4] Larry Page, Sergey Brin, R. Motwani, and T. Winograd, The PageRank Citation Ranking: Bring Order to the Web, Stanford Univ. Technical Report, 1998.

[5] Jon M. Kleinberg, Authoritative Sources in a Hyperlinked Environment, Journal of the ACM, Vol. 46, No. 5, pp. 604-632, 1999.

[6] Maxim Lifantsev and Tzi-cker Chiueh, Implementation of a modern web search engine cluster, In Proc. of the USENIX Annual Technical Conference, 2003.

[7] Taher H. Haveliwala, Topic-sensitive PageRank, Proceedings of the 11th international conference on World Wide Web, 2002.

[8] Monica Bianchini, Marco Gori, Franco Scarselli, Inside PageRank, ACM Transactions on Internet Technology, Vol. 5, No. 1, pp. 92-128, 2005

[9] Arvind Arasu, et al., Searching the Web, ACM Trans. on Internet Technology, Vol. 1(1), pp. 2-43, 2001.

[10] Zheng Chen, Shengping Liu, Liu Wenyin, Geguang Pu, and Wei-Ying Ma, Building a web thesaurus from web link structure, In Proc. of the ACM SIGIR, pp. 48-55, 2003.

[11] C. Lee, G. Golub and S. Zenios. A Fast Two-Stage Algorithm for Computing PageRank, Technical report, Stanford University, 2003.