커뮤니티 제한 검색을 위한 웹 크롤링 및 PageRank 계산

김계정[O†] 김민수[†] 김이른[†] 황규영[†]
한국과학기술원 전산학과/첨단정보기술연구센터
{gjkim[O], mskim, yrkim_s, kywhang}@mozart.kaist.ac.kr

Web Crawling and PageRank Calculation for Community-Limited Search

Gye-Jeong Kim[O†] Min-Soo Kim[†] Yi-Reun Kim[†] Kyu-Young Whang[†]
[†] Department of Computer Science &
Advanced Information Technology Research Center
Korea Advanced Institute of Science and Technology

요 약

최근 웹 검색 분야에서는 검색 질을 높이기 위한 기법들이 많이 연구되어 왔으며, 대표적인 연구로는 제한 검색, focused crawling, 웹 클러스터링 등이 있다. 그러나 제한 검색은 검색 범위를 의미적으로 관련된 사이트들로 제한할 수 없으며, focused crawling은 질의 시점에 크롤링하기 때문에 질의 처리 시간이 오래 걸리고, 웹 클러스터링은 많은 웹 페이지들을 대상으로 클러스터링하기 위한 오버헤드가 크다. 본 논문에서는 검색 범위를 특정 커뮤니티로 제한하여 검색 하는 *커뮤니티 제한 검색*과 커뮤니티를 구하는 방법으로 *cluster crawler*를 제안하여 이러한 문제점을 해결한다. 또한, 커뮤니티를 이용하여 PageRank를 2단계로 계산하는 방법을 제안한다. 제안된 방법은 첫 번째 과정에서 커뮤니티 단위로 지역적으로 PageRank를 계산한 후, 두 번째 과정에서 이를 바탕으로 전역적으로 PageRank를 계산한다. 제안된 방법은 Wang에 의해 제안된 방법에 비해 PageRank 근사치의 오차를 59%정도로 줄일 수 있다.

1. 서론

웹 검색 시스템은 웹에서 원하는 정보를 보다 쉽게 찾기 위한 도구로서 그 중요성이 점차 부각되고 있다. 최근 들어 웹 검색 시스템에서 사용자의 질의에 대해 정확도가 높은 충분한 양의 검색 결과를 제공하고자 하는 연구들이 활발하게 진행되고 있다. 그 대표적인 연구로는 제한 검색, focused crawler, 웹 클러스터링 등이 있다. 제한 검색은 검색 범위를 특정 사이트 또는 도메인으로 한정시켜 검색 결과를 제공하는 방법이며[10][1], focused crawler는 질의가 주어진 시점에 질의와 관련 있는 웹 페이지들만을 수집하여 결과로 반환하는 방법이다[2]. 웹 클러스터링은 수집된 웹 페이지들을 서로 관련 있는 웹 페이지들끼리 클러스터링하는 방법이다[3].

그러나 위에서 설명한 최근의 연구들은 다음과 같은 단점을 가지고 있다. 제한 검색은 검색의 범위를 URL에 의해 명시되는 사이트 또는 도메인들로만 제한할 수 있을 뿐이며, 의미적으로 관련된 사이트들로 제한할 수 없다. Focused crawler는 질의 시점에 웹 페이지들을 수집하기 때문에 질의 처리 시간이 오래 걸린다. 웹 클러스터링은 클러스터를 구하기 위해 많은 양의 사이트들 또는 웹 페이지들을 대상으로 복잡한 처리를 수행하므로 공간적 시간적 비용이 크다.

본 논문에서는 주제별로 검색 범위를 지정할 수 있고, 빠른 검색을 제공하는 *커뮤니티(community) 제한 검색*을 제안한다. 커뮤니티 제한 검색은 검색 범위를 특정 커뮤니티로 지정하여 제한 검색을 수행하는 방법으로서, *커뮤니티*는 의미적으로 관련된 사이트들의 집합으로 정의된다. 커뮤니티를 구하는 방법으로는 *cluster crawler*를 제안한다. Cluster crawler는 크롤링 중에 점증적(incremental)으로 클러스터를 구하여 커뮤니티에 속하는 사이트들만을 크롤링함으로써 클러스터링에 소요되는 시간을 최소화한다.

본 논문에서 제안하는 커뮤니티 개념을 사용하면 최근 대부분의 검색시스템에서 검색 결과의 중요도를 계산하기 위해 사용하는 PageRank 알고리즘[4]의 근사치의 정확도를 높일 수 있는 추가적인 이점이 있다. 최근 PageRank를 계산하는 시간을 단축하기 위한 많은 연구들이 있었으며, 그 중 Wang[5]은 블록 내의 랭크와 블록 간의 랭크를 이용하여 PageRank를 2단계로 계산함으로써 계산 시간을 단축시킨 방법을 제안하였다. Wang의 방법은 블록의 단위로 사이트(서버)를 사용하고 있으나, 본 논문에서 제안한 커뮤니티를 블록의 단위로 사용하게 되면, Wang의 방법에 비해 블록 간의 링크가 줄어들어 PageRank 근사치의 정확도를 높일 수 있다.

* 본 연구는 첨단정보기술연구센터(AITrc)를 통한 한국과학재단, BK21 (Brain Korea 21) 및 한국과학기술원의 기본연구사업 지원을 받았음.

본 논문의 구성은 다음과 같다. 제 2장에서는 관련 연구들에 대해 설명한다. 제 3장에서는 커뮤니티 제한 검색과 cluster crawler를 제안하고, 커뮤니티를 이용하여 PageRank를 2단계로 계산하는 방법에 대해 설명한다. 제 4장에서는 cluster crawler와 2단계 PageRank 알고리즘의 성능을 평가한다. 제 5장에서는 향후 연구를 기술하고 결론을 내린다.

2. 관련 연구

본 장에서는 관련 연구로서 웹 검색 시스템과 관련된 주요 기술들에 대해 설명한다. 제 2.1절에서는 웹 클러스터링 방법을 설명하고, 제 2.2절에서는 PageRank에 대해 설명한다.

2.1. 웹 페이지의 클러스터링

웹 페이지의 클러스터링은 단어 또는 링크 등을 이용하여 웹 페이지 간의 유사도를 측정는 과정과 측정된 유사도를 바탕으로 기존의 데이터 클러스터링 알고리즘을 적용하는 과정으로 이루어 진다.

Minimum spanning tree(MST) 클러스터링은 클러스터링 알고리즘 중 하나로 MST를 subtree들로 나누어 클러스터를 구하는 방법이다[6]. MST 클러스터링은 클러스터의 개수를 사전에 정하지 않아도 클러스터를 구할 수 있으며 여러 가지 데이터 분포에서도 잘 동작하는 장점이 있다.

2.2. PageRank 계산 방법

PageRank는 웹 검색 시스템의 결과를 정렬하기 위하여 웹 페이지의 중요도를 측정하는 방법이다[4]. PageRank는 사용자가 링크를 따라 이동하는 것을 시뮬레이션함으로써 사용자가 각 웹 페이지를 방문할 확률을 구하여 웹 페이지의 중요도로 사용하며 구글[7] 등 많은 웹 검색 엔진에서 쓰이고 있다.

PageRank 알고리즘은 수집된 전체 웹 페이지에 대하여 행렬 곱셈을 수렴할 때까지 반복하기 때문에 웹의 크기가 커질수록 많은 시간을 소요한다. 이에 따라 PageRank 계산 시간을 단축하기 위한 여러 가지 연구가 수행되었다. 최근에는 PageRank를 블록 단위로 계산하여 PageRank의 근사치를 구할 수 있는 PageRank 2단계 계산 방법이 연구되었다[5][8]. 제안된 방법은 우선 링크의 대부분이 블록 내의 링크가 되도록 웹 페이지들의 블록을 나눈 후, 블록 내의 랭크와 블록 간의 랭크의 2단계로 PageRank를 계산한다. 그러나 제안된 방법은 PageRank의 초기값을 예측하는 데에만 사용되고 있거나 PageRank를 교정하는 작업이 필요한 문제점이 있다.

3. 커뮤니티 제한 검색을 위한 Cluster Crawler 및 PageRank 계산 방법

본 장에서는 커뮤니티의 및 커뮤니티 제한 검색의 개념을 제안하고 cluster crawler와 2단계 PageRank 계산 방법을 설명한다. 제 3.1절에서는 커뮤니티의 개념을 정의하고, 제 3.2절에서는 커뮤니티 제한 검색에 대해 설명한다. 제 3.3절에서는 cluster crawler의 개요와 아키텍처에 대해 설명한다. 제 3.4절에서는 커뮤니티를 이용한 2단계 PageRank 알고리즘을 설명한다

3.1. 커뮤니티

본 논문에서는 의미적으로 관련된 사이트들의 집합을 *커뮤니티*로 정의한다. 커뮤니티를 구하는 방법은 크게 수작업으로 구하는 방법과 자동으로 구하는 방법이 가능하며, 본 논문에서는 링크 기반의 클러스터링을 통해 자동으로 커뮤니티를 구하는 방법을 중심으로 설명한다.

3.2. 커뮤니티 제한 검색

본 논문에서는 검색 범위를 특정 커뮤니티로 지정하여 제한 검색을 수행하는 방법을 *커뮤니티 제한 검색*으로 정의한다. 제안된 검색 방법은 의미적으로 관련된 사이트들을 검색 범위로 지정할 수 있으며, 이미 수집된 웹 페이지들과 커뮤니티에 대해 구성된 색인을 이용하므로 빠른 검색을 지원한다.

커뮤니티 제한 검색의 예는 다음과 같다. 삼성 계열사의 채용 정보를 찾고자 할 경우 전체 웹 페이지를 대상으로 "삼성 채용"이라는 질의를 수행하면 "삼성"과 "채용"이란 단어를 가진 수많은 질의 결과를 얻게 되어 원하는 정보를 정확하게 찾기 힘든 반면, 커뮤니티 제한 검색을 이용하면 "삼성"이라는 질의를 통해 삼성 커뮤니티를 찾은 후, 커뮤니티 제한 검색으로 "채용" 정보를 찾음으로써 삼성 커뮤니티 내의 채용 정보만을 쉽게 찾을 수 있다.

3.3. Cluster Crawler의 개요 및 아키텍처

커뮤니티 제한 검색을 위해서는 웹 페이지를 수집하는 과정(크롤링)과 커뮤니티를 생성하는 과정(클러스터링)이 필요하며, 본 논문에서는 이를 위하여 *cluster crawler*를 제안한다. Cluster crawler는 크롤링과 클러스터링을 동시에 수행하는 크롤러로 웹의 부분 정보만을 이용하여 점증적으로 클러스터링하는 방법이다. Cluster crawler는 MST 클러스터링과 유사한 결과를 얻으면서 seed와 관련 있는 사이트들만을 선별하여 크롤링하고, 크롤링 시 링크 정보를 메모리 상에서 실시간으로 처리하여 클러스터링하기 때문에 소요 시간이 줄어드는 장점이 있다.

Cluster crawler는 각 seed에 대하여 Prim의 MST 알고리즘을 적용하여 클러스터에 속하는 사이트들을 발견하고 이들을 크롤링한다. 각 seed는 각 클러스터를 구성하는 MST의 시작점이 되며, 이를 위하여 cluster crawler는 클러스터 개수만큼의 seed를 필요로 한다. Cluster crawler의 알고리즘은 그림 1과 같다. s는 seed 사이트로 Prim의 알고리즘에서 MST의 구축을 시작하는 vertex이다. V_{cr}는 크롤링된(crawled) 사이트들의 집합을 나타내며, MST에 포함된 vertex들의 집합이다. V_{fr}는 크롤링된 사이트들과 링크로 연결된 사이트들 중 아직 크롤링되지 않은(fringe) 사이트들의 집합을 나타내며, 발견되었으나 아직 MST에 포함되지 않은 vertex들의 집합이다. Edge는 사이트 간의 링크를 나타내며, edge의 weight는 edge가 연결하고 있는 두 사이트 간의 유사도를 나타낸다. Weight threshold는 MST 클러스터링 알고리즘에서 클러스터들을 분리하는 edge의 weight 조건으로 사용되는 것으로 cluster crawler에서도 이와 유사하게 seed의 클러스터를 다른 클러스터들과 분리하는 edge의 조건으로 사용된다. Cluster crawler는 각 seed에 대하여 seed가 속한 클러스터의 MST만을 발견하고 종료된다.

Edge의 weight, 즉 사이트 간의 유사도를 측정하는 방법은 다음과 같다. 사이트 s_i가 사이트 s_j를 가리키는 링크를 많이 가지고 있을수록 s_i와 s_j가 관련 있다고 가정하고 s_i의 out link의 총 개수를 n_i, s_i에서 사이트 s_j를 가리키는 링크의 개수를 n_{ij}라고 할 때 사이트 간의 유사도는 n_{ij}/n_i로 정의한다. 이 때 링크로 연결되지 않은 두 사이트 간의 유사도는 0으로 정의한다.

3.4. 커뮤니티를 이용한 PageRank 계산 방법

커뮤니티를 이용하면 참고 문헌 [5][8]과 같이 PageRank 계산을 2단계로 분리하여 보다 빨리 계산할 수 있다. PageRank는 링크를 통하여 서로의 랭크에 영향을 주는 알고리즘으로 링크가 연결되지 않은 웹 페이지들은 서로의

Cluster Crawler 알고리즘

입력: seed set S, weight threshold wt
출력: 클러스터 별로 크롤링한 사이트들의 집합
알고리즘:
1. s∈S인 모든 s에 대하여 다음의 과정을 실행한다.
   1.1. seed 사이트 s부터 크롤링을 시작한다.
   $V_{cr} = \{\}$
   $V_{fr} = \{s\}$
   1.2. V∈V_{cr}이고, W∈V_{fr}인 edge (V,W) 중 weight가 가장 큰 edge를 선택하고 W를 크롤링한다.
   $V_{cr}=V_{cr}\cup\{W\}$
   1.3. W를 크롤링한 후, W의 링크가 가리키는 사이트들 중 V_{cr}에 속하지 않는 사이트의 집합 $V_{fr}{}^{W}$를 구한다.
   $V_{fr}= V_{fr} - \{W\}\cup V_{bd}{}^{W}$
   1.4. V∈V_{cr}이고, W∈V_{fr}인 edge (V,W) 중 wt보다 큰 weight를 가진 edge가 있는 동안 과정 1.2와 1.3을 반복한다.
2. 과정 1에 의해 크롤링된 사이트의 집합을 s에 대한 클러스터로 생성한다.

그림 1. Cluster crawler의 알고리즘

랭크에 영향을 거의 주지 않는다. 따라서 서로 링크가 연결되지 않은 웹 페이지들의 PageRank를 분리하여 계산할 수 있다. 우리나라 대학 115개의 사이트 1,191개를 대상으로 측정한 결과, 전체 링크 수 약 28만 개 중 사이트 간의 링크는 전체 링크의 8.4%를, 커뮤니티 간의 링크는 전체 링크의 0.55%를 차지하였다. 따라서 본 논문에서는 사이트 대신 커뮤니티를 블록 단위로 사용하여 PageRank를 2단계로 계산하는 방법을 제안한다.

PageRank를 2단계로 분리하여 계산할 경우 각 커뮤니티 내의 PageRank (이를 *Local PageRank*라 함)와 커뮤니티 간의 랭크(이를 *Community-Rank*라 함)는 링크 파일의 크기가 작기 때문에 메모리 내에서 계산 가능하여 계산 시간을 단축할 수 있다. 또한 새로운 커뮤니티가 추가 되어도 기존 커뮤니티의 Local PageRank를 재사용할 수 있으므로 PageRank의 업데이트 시 계산 시간이 빨라진다. PageRank 2단계 계산은 Local PageRank 계산, Community-Rank 계산, *Global PageRank* 계산으로 이루어진다. 각 랭크의 개념 및 계산 방법은 다음과 같다.

Local PageRank는 사용자가 커뮤니티 C 내에 있을 때 웹 페이지 p에 머물 확률 $P(p|C)$을 나타낸다. 계산은 커뮤니티 내의 웹 페이지를 대상으로 커뮤니티 내의 링크만을 이용하여 PageRank 알고리즘을 적용한다.

Community-Rank는 사용자가 커뮤니티 C 내에 있을 확률 $P(C)$를 나타낸다. 계산은 PageRank 알고리즘을 이용하는 방법(이를 CR_P라 함)과 웹 페이지의 개수에 비례하는 방법(이를 CR_R이라 함)으로 나누어 진다. CR_P는 커뮤니티들을 대상으로 커뮤니티 간의 링크를 이용하여 PageRank 알고리즘을 적용하여 계산한다. CR_R은 전체 웹 페이지 개수를 n, 커뮤니티 C_i에 속하는 웹 페이지의 개수를 n_{C_i}라고 할 때, n_{C_i}/n로 계산한다.

Global PageRank는 수집된 전체 웹 페이지에 대한 PageRank로 사용자가 웹 페이지 p에 있을 확률 $P(p)$를 나타낸다., $P(p)=P(C)\cdot P(p|C)$이므로 Global PageRank는 수식 (1)과 같이 계산할 수 있다.

$$GlobalPageRank(p_i) = CommunityRank(C_j)\cdot LocalPageRank(p_i) \quad \text{, where } p_i \in C_j \quad (1)$$

4. 성능 평가

본 장에서는 cluster crawler와 PageRank 2단계 계산 방법의 성능을 평가한다. 제 4.1절에서는 cluster crawler를 이용한 클러스터링 시간과 클러스터의 quality를 평가하고, 제 4.2절에서는 PageRank 2단계 계산 방법으로 구한 PageRank의 정확도를 평가한다.

4.1. Cluster crawler의 성능 평가

Cluster crawler의 우수성을 입증하기 위하여 cluster crawler의 클러스터링 시간과 클러스터의 quality를 측정한다. 본 논문의 모든 실험은 펜티엄4 1.7GHz 512MB PC에서 리눅스 2.4를 사용하였다.

클러스터링 시간 비교는 cluster crawler과 MST 클러스터링의 소요 시간을 비교한다. Cluster crawler는 크롤링과 클러스터링이 결합되어 있기 때문에 기본 크롤링 코드 이외의 클러스터링을 위해 추가된 모든 코드에서 소요되는 시간만을 측정하였다. MST 클러스터링은 크롤링이 끝난 후에 링크를 처리하여 사이트 간 유사도를 측정하고 클러스터링 하기까지 소요되는 시간을 측정하였다. 실험에 사용된 데이터는 임의로 선택한 seed를 사용하여

cluster crawler로 크롤링한 사이트들의 집합이며, weight threshold로는 0.01을 사용하였다. 실험 파라미터는 seed의 개수를 사용하였다.

클러스터링 시간 측정 실험의 결과는 그림 2과 같다. 그래프에서 cluster crawler가 MST 클러스터링보다 100배 정도 적은 클러스터링 시간을 소요하는 것을 알 수 있다. 이것은 cluster crawler에서는 크롤링과 동시에 클러스터링하기 때문에 클러스터링에 필요한 링크 정보가 메모리 상에 이미 존재하며, 사이트 간 유사도를 측정하는 과정 중 일부가 크롤링에서 이미 수행되었기 때문으로 분석된다. 이에 반해 MST 클러스터링에서는 링크 파일을 읽어 들이고 사이트 간 유사도를 측정하는 데 많은 시간이 소요된다.

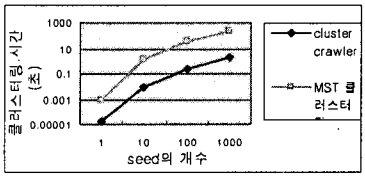

그림 2. Seed의 개수에 따른 클러스터링

Cluster crawler의 클러스터링 quality를 측정하는 실험은 위 실험과 동일한 데이터에서 cluster crawler의 클러스터가 MST 클러스터링의 클러스터와 일치하는 정도를 측정한다. 이를 위하여 precision과 recall을 측정하는 데 MST 클러스터링으로 구한 클러스터를 C_T, cluster crawler로 구한 클러스터를 C_A라 할 때 각각은 수식 (2)와 같이 정의된다.

$$\text{Precision} = \frac{|C_T \cap C_A|}{|C_A|}, \quad \text{Recall} = \frac{|C_T \cap C_A|}{|C_T|} \qquad (2)$$

실험 결과는 그림 3과 같다. 왼쪽 그래프는 precision을 오른쪽 그래프는 recall을 나타낸다. 그래프에서 precision은 최대값인 1의 가까운 값을 나타냄을 알 수 있으며, recall은 seed의 개수가 100개와 1,000개일 때 0.7정도로 떨어짐을 알 수 있다. Recall이 떨어지는 이유는 cluster crawler에서는 서로 다른 seed에 대한 클러스터 간의 링크는 무시하나 전역적으로 클러스터를 구할 경우에는 클러스터 간의 링크로 인하여 cluster crawler에서의 여러 개의 클러스터가 하나의 클러스터로 병합되는 현상이 일어나기 때문이라고 분석된다. Cluster crawler에서 클러스터 간의 링크가 발견될 경우 두 클러스터를 병합하도록 처리함으로써 recall을 높일 수 있을 것으로 기대된다.

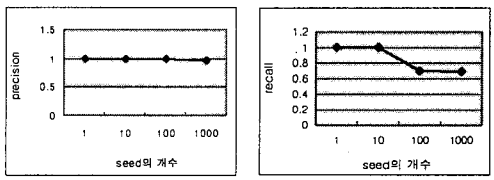

그림 3. weight threshold에 따른 클러스터의 크기와 precision

4.2. 2단계 PageRank 알고리즘의 성능 평가

본 논문에서는 제안한 알고리즘으로 계산한 PageRank의 오차를 Wang의 알고리즘[5]으로 계산한 PageRank의 오차와 비교한다. 오차 측정 방법으로는 Wang의 논문[5]에서 사용한 방법과 같이 각 알고리즘으로 구한 PageRank와 기본 PageRank 알고리즘으로 구한 PageRank와의 Kendall τ-distance[9]를 사용한다. τ-distance는 두 개의 정렬된 리스트 간의 순서가 바뀐 쌍(pair)의 비율을 측정한다. 두 리스트의 순서가 완전히 일치할 경우 τ-distance는 0이 되며, 두 리스트의 순서가 역순일 경우 τ-distance는 1이 된다.

실험 데이터는 다음과 같은 두 가지 데이터 집합을 사용한다. 데이터 집합 1은 대학 홈페이지 115개를 seed로 사용하여 클러스터 크롤링한 웹 페이지들의 집합이며, 데이터 집합 2는 "co.kr"을 도메인으로 가진 사이트 2,000개를 seed로 사용하여 클러스터 크롤링한 웹 페이지의 집합이다.

데이터 집합 1과 2에 대한 실험 결과가 각각 그림 4에 나타나 있다. 가로축은 Community-Rank를 계산하는 방법으로 순서대로 CR_P와 CR_R을 나타내며, 세로축은 τ-distance를 나타낸다. 왼쪽 막대는 블록의 단위로 사이트를 사용한 것이며, 오른쪽 막대는 커뮤니티를 사용한 것이다. 그래프를 통하여 Community-Rank를 계산하는 방법으로 CR_R을 사용하였을 때 τ-distance가 작으며, 이 때 커뮤니티를 블록의 단위로 사용한 방법이 사이트를 블록의 단위로 사용한 방법보다 τ-distance를 데이터 집합 1의 경우 약 55%, 데이터 집합 2의 경우 약 59%로 줄임을 알 수 있다.

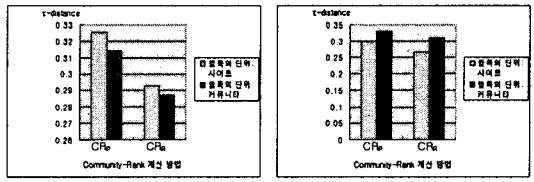

그림 4. PageRank의 τ-distance

CR_R의 결과가 좋은 이유는 CR_P의 경우 커뮤니티 내의 웹 페이지 개수에 관계 없이 모든 커뮤니티를 동등한 하나의 단위로 가정하며 블록 간의 링크가 매우 적기 때문에 PageRank 알고리즘의 결과가 부정확하기 때문이라고 분석된다. CR_R에서는 커뮤니티를 블록의 단위로 사용할 경우 사이트를 블록의 단위로 사용할 때보다 블록 간의 링크가 줄어들어 PageRank 근사치의 오차가 줄어드는 것으로 분석된다. 실제 블록 간의 링크 비율은 표 1과 같다.

표 1. 블록 간의 링크 비율

데이터 집합	사이트 간 링크 비율	커뮤니티 간 링크 비율
1	8.4%	0.55%
2	10.3%	2.2%

6. 결론

본 논문에서는 커뮤니티의 개념을 정의하고 검색 범위를 특정 커뮤니티로 지정하여 제한 검색을 수행하는 커뮤니티 제한 검색을 제안하였다. 커뮤니티 제한 검색은 주제별로 검색 범위를 지정할 수 있고, 빠른 검색을 제공한다. 다음으로, 커뮤니티를 구하는 방법으로서 cluster crawler를 제안하였다. Cluster crawler는 점증적으로 클러스터를 구하여 의미 있는 사이트들만을 크롤링하며, 크롤링에서 처리된 링크 정보들을 이용하기 때문에 기존 클러스터링 방법에 비해 클러스터링 시간이 줄어든다. 다음으로, 커뮤니티를 이용한 2단계 PageRank 계산 알고리즘을 제안하였다. 커뮤니티를 이용한 2단계 Page-Rank 계산은 블록 간의 링크를 줄임으로써 PageRank의 근사치를 구하는 기존 연구들보다 근사치의 오차를 55~59%로 줄인다.

향후에는 커뮤니티의 질을 높이기 위한 방법을 연구하고, cluster crawler의 성능을 측정하는 실험을 추가적으로 수행할 예정이다.

참고문헌

[1] Google Help, http://www.google.com/help/refinesearch.html, 2004.

[2] Brin, S. and Page, L. "The Anatomy of a Large-Scale Hypertextual Web Search Engine," In *Proc. 7th Int'l World Wide Web Conf.*, pp. 107-117, Brisbane, Australia, Apr. 1998.

[3] Zamir, O. and Etzioni, O., "Web Document Clustering: a Feasibility Demonstration," In *Proc. 19 Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval*, pp. 46-54, Melbourne, Australia, June 1998.

[4] Page, L. et al., The PageRank Citation Ranking: Bringing Order to the Web, Technical Report, Stanford University, 1998.

[5] Wang, Y. and Dewitt, D., "Computing PageRank in a Distributed Internet Search System," In *Proc. 30th Int'l Conf. on Very Large Data Bases*, pp. 420-431, Toronto, Canada, Aug. 2004.

[6] Zahn, C., "Graph Theoretical Methods for Detecting and Describing Gestalt Clusters," *IEEE Trans. on Computers*, Vol. C-20, No. 1, pp. 68-86, Jan. 1971.

[7] Google, http://www.google.com

[8] Kamvar, S. et al., Exploiting the Block Structure of the Web for Computing PageRank, Technical Report, Stanford University, 2003.

[9] Kendall, M. and Gibbons, J., *Rank Correlation Methods*, Edward Arnold, 1990.

[10] 이재길, 이민재, 김민수, 황규영, "오디세우스 객체관계형 DBMS를 사용한 사이트 제한 검색의 구현," 한국정보과학회 춘계학술발표회 논문집, pp. 755-757, 2003년 4월.