제14차 High Performance Computing
국제 컨퍼런스 국외출장 결과보고

2008. 1.

통 계 정 보 국

목 차

14차 High Performance Computing
국제 컨퍼런스 국외출장 결과보고

Ⅰ. 출장개요

1. 출장목적

□ 고성능 컴퓨팅의 운영체제 등 최신 정보기술에 대한 지식 습득 및 동향파악을 바탕으로 현재 우리 청에서 추진 중인 **각종 통계정보 생산 및 서비스 시스템의 효율적 구축방안 모색**

□ 정보시스템 관련 선진기술을 비교 연구함으로써 향후 우리 청의 **효율적인 통계정보 서비스 체계 구축에 반영**

2. 출장기간 : 2007. 12. 16 ~ 12. 23. (8일간)

3. 출장장소 : 인도 고아

4. 출장자

□ 전산개발과 과장 김규영,
정보서비스과 5급 한상선, 전산개발과 5급 김상진

5. 주요 회의내용

□ 고성능 컴퓨팅의 기술, 적용 및 경험
□ 고성능 컴퓨팅 운영체제(부하의 밸런싱, 스케쥴링 및 자원관리)
□ 병렬 및 분산처리 컴퓨팅과 알고리즘

Ⅱ. 컨퍼런스 소개

1. 명칭 : HiPC 2007

(14th International Conference on High Performance Computing)

2. 컨퍼런스 내용 : Multiprocessor Architecture 등 병렬처리 관련내용

□ 인도, 미국 등 31개 국가 참가(253개 논문 발표)

3. 연혁

□ HiPC ’95 (1995년) ~ HiPC ’06 (2006년)

4. 발표 논문집은 통계청 자료실에 보관

□ HiPC 2007(14th International Conference on High Performance Computing)

5. 14차 컨퍼런스 조직구성

구 분	성명 및 소속
General Co-chairs	- Manish Parashar (Rutgers University, USA) - Ramamurthy Badrinath (HP, India)
Vice General Co-chairs	- Rajendra V. Boppana (University of Texas an San Antonio, USA) - Rajeev Muralidhar (Intel, India)
Program Chair	- Srinivas Aluru (Iowa State University, USA)

구 분	성명 및 소속
Program Vice-chairs	- Algorithms : Peter Sanders (Universität Karlsruhe, Germany) - Applications : Sivan Toledo (Tel Aviv University, Israel) - Architecture : Peter J. Varman (Rice University, USA) - Communication Networks : Dhabaleswar K. Panda(Ohio State University, USA) - Mobile and Sensor Computing : Ahmed Helmy(University of Florida, USA) - Systems Software : Manish Gupta (IBM Corporation, India)
Steering Chair	- Viktor K. Prasanna (University of Southern California, USA)
Workshops Chair	- Manimaran Govindarasu (Iowa State University, USA)
Poster/Presentation Chair	- Rajeev Thakur (Argonne National Laboratory, USA)
Tutorial Chair	- Rajeev Sivaram (Google, USA)
HiPC User Symposium Co-chairs	- T.K. Ramesh (The Boeing Company, USA) - Raghuram Tupuri (Advance Micro Devices, India) - Santosh Sreenivasan (Talentain, India)
Steering Committee	- P. Anandan (Microsoft Research, India) - David A. Bader (Georgia Institute of Technology, USA) - Ramamurthy Badrinath (HP, India) - Rudramuni B. (Dell India, Banglore, India) - Frank Baetke (HP, USA) - R. Govindarajan (India Institute of Science, India) - Harish Grama(IBM, India) - Manish Gupta (India Systems & Technology Lab, IBM, India) - Vittal Kini (Intel, India) - N. S. Nagaraj (Infosys, India) - Viktor K. Prasanna : **Chair** (University of Southern California, USA) - Venkat Ramana (Cray-Hinditron, India) - Sartaj Sahni (University of Florida, USA) - V. Sridhar (Satyam Computer Services, India) - Harrick M. Vin (Tata Research, Development & Design Center)

구 분	성명 및 소속
Program Committee	
- Algorithms	- Thomas Cormen (Dartmouth College, USA) - Devdatt Dubhashi (Chamers University of Tech & Göteborg University, Sweden) - Matteo Frigo (University of Paderborn, Germany) - Klaus Jansen (University of Kiel, Germany) - Christos Kaklamanis (University of Patras, Greece) - Samir Khuller (University of Maryland, USA) - Dariuzs Kowalski (University of Liverpool, UK) - Jesper Larsson Traeff (C&C Research Labs, NEC Europe Ltd., Germany)
- Applications	- David Abrahamson (Monash University, Australis) - Nikos Chrisochoides (College of William and Mary, USA) - Luc Giraud (INSEEIHT, Toulouse, France) - Ananth Grama (Purdue University, USA) - Phalguni Gupta (IIT Kanpur, India) - Eldad Haber (Emory University, Japan)
- Architecture	- Stergios Anastasiadis (University of Ioannian, Greece) - Rajeev Balasubramonian (University of Utah, USA) - Anasua Bhowmik (AMD, India) - Kshitij Doshi(Intel, USA) - Maria Garzaran (University of Illinois Urbana, USA)
- Communication Networks	- Pavan Balaji (Argonne National Laboratory, USA) - Mohammad Banikazemi (IBM T.J. Watson Research Center, USA) - Ron Brightwell (Sandia National Laboratory, USA) - Darius Buntinas (Argonne National Laboratory, USA) - Jose Flich (Universidad Politécnica de Valencia, Spain)
- Mobile & Sensor Computing	- Kevin Almeroth (UC Santa Barbara, USA) - Fan Bai (General Motors Research, USA) - Polly Huang (National Taiwan University, Taiwan) - Lyad Kanj (Depaul University, USA) - Karim Seada (Nokia Research, USA)
- System Software	- Gianfranco Bilardi (University of Padova, Italy) - R. Govindarajan (IISc Bangalore, India) - Rinku Gupta (Dell, USA) - Yanyong Zhang (Rutgers University, USA)

Ⅲ. 14차 컨퍼런스 내용(요약)

1. Keynote Addresses(Abstracts)

가. The future is parallel but it may not be easy (미래는 현재와 비슷하지만 쉽지는 않을 것이다)

연사 : Michael Flynn

(약력 : Maxeler Corporation, London, UK and Stanford University, USA)

클록 주파수(clock frequency) 향상에 의한 프로세서의 성능 확장(scaling)은 이제 한계에 도달하였다. 멀티코어 구조에 대한 강조는 병렬 프로그래밍이나 구조의 약진 때문이 아니라 주파수 확장의 실패에서 비롯되었다.

직렬 프로그램을 멀티 태스크 프로그램으로 자동 편집하는 일의 발전은 이제 느려졌다. 과거의 병렬 프로젝트를 살펴보면 성능과 프로그램화 가능성의 문제점을 알 수 있다.

이러한 문제점 해결을 위해서는 통제 구조를 병렬화하고 메모리의 병목 해결 등의 근원적 문제점을 이해할 필요가 있다. 많은 애플리케이션의 경우 성능은 프로그램화 가능성의 희생을 요구하며 신뢰도는 성능의 희생을 요구한다.

나. Web Search : bridging information retrieval and microeconomic modeling (웹 검색: 정보 검색과 미시경제모델의 브리징(bridging))

연사 : Prabhakar Raghavan

(약력 : Head, Yahoo! Research and Consulting Professor, Stanford University, USA)

웹 검색은 우리가 당연한 것으로 받아들이고 있는 편리한 도구로, 또한 광고주와 바이어를 연결해주는 매개체로, 그리고 그러한 서비스를 제공하는 회사로서는 급성장하는 수입원으로 우리의 의식을 지배하게 되었다.

최신 기술과 어떻게 그러한 기술을 성취하게 되었나에 대한 간략한 개요를 살펴본 후에 본 프리젠테이션에서는 웹 검색에서 발생하는 일련의 기술적 도전과제를 -스팸 탐지에서 옥션 메커니즘에 이르기까지- 논의할 것이다.

다. Petaflap/s, Seriously(페타플럽, 진지하게)

연사 : David Keyes
(약력 : Fu Foundation Professor, Columbia University, USA)

Gordon Bell Prize로 추적한 바와 같이, 애플리케이션의 일정한 부동 소수점수는 1 기가플럽/s가 구조 시뮬레이션에서 보고된 1988년으로부터 200 테라플럽(teraflop)이 분자 동력학 시뮬레이션에서 보고된 2006년까지 5자리 수 단위(five orders of magnitude)만큼 증가하였다.

같은 기간에 대한 무어 법칙의 다양한 버전은 개별 프로세서의 경우 2에서 3자리 수 단위만큼의 향상을 보였다. 나머지 인자는 동시성에서 나오며 이것은 최근 Bell Prize 최종 후보 대다수가 선택한 플랫폼인 BlueGene/L 컴퓨터의 경우 자리 100,000이다.

반도체 산업의 경우 실리콘기반 로직 및 메모리의 로드맵과 비교해서 상대적으로 퇴보하기 시작했기 때문에, 실용적 애플리케이션에서 오래동안 고대한 1페타플럽/s 이정표에 도착하기위해서 동시성은 다음 자리 수 단위를 획득하는데 있어서 중요한 역할을 할 것이며 이것은 2009년 쯤 발생할 것이다. 편미분공식의 Eulerian 공식에 기초한 시뮬레이션은 페타 규모 용량을 이용한 최초 애플리케이션의 하나일 것이다.

그러나 현재 가장 추구하는 방식은 아니다. 약한 규모만이 암달의 법칙으로 표현되는 근본적 한계를 극복할 수 있으며, 최적의 묵시적 공식만이 편미분공식(PDE)의 약한 규모하의 Courant-Friedrichs-Lewy 안정성 이론의 직접적 결과인 규모 관련 또 다른 한계점을 극복할 수 있다. PDE 기반 애플리케이션과 양자 크로모 역학과 같은 페타 규모 로드맵의 기타 격자 기반 애플리케이션은 강제로 최적의 묵시적 해결자를 채택해야할 것이다.

그러나 페타 규모 시뮬레이션으로 가는 이 좁은 경로조차도 역동적인 적응력의 필요로 인해서 불안정해지며, 이것은 우리로 하여금 오늘날 널리 사용되는 것보다 덜 동시성을 띠는 알고리즘과 대기행렬을 고려하도록 한다. 가장 최근 ITRS 로드맵인 SCsLeS 보고서(2003-04), 봉투 뒷면 추정값과 최근 이용 가능한 플랫폼의 PDE기반 코드에 의한 숫자 경험에 비추어볼 때, 대표적인 애플리케이션에 대해서 페타플럽/s경로를 계획해야할 것이다.

라. High Performance Data Mining (고성능 데이터 마이닝 - 지구기후시스템의 패턴 발견 애플리케이션)

연사 : Vipin Kumar

(약력 : William Norris Professor, University of Minnesota, USA)

과학기술과 작업처리량 실험 테크닉이 발달함에 따라 민간 기업체와 여러 과학 및 공학 분야에서 대량 자료 집합을 이용할 수 있게 되었다. 테라바이트 자료는 오늘날 흔히 존재하며 과학, 공학, 사업, 생물정보학, 의학 등의 많은 응용 분야에서 가까운 미래에 페타바이트에 도달하리라고 예상된다. 이로써 유용한 자료를 추출하기위해 자동화된 데이터 중심 테크닉을 개발할 전례에 없던 기회가 형성되었다.

이러한 지식 발견 과정의 중요한 단계인 데이터마이닝은 데이터 안에 숨겨진 흥미롭고, 중요하며 유용한 패턴을 찾아내는 방법으로 구성된다. 본 프리젠테이션은 지구기후시스템의 패턴을 이해하기위한 우리 그룹의 수많은 데이터마이닝 연구와 도전과제에 대한 개요를 제공할 것이다.

마. The Transformation Hierarchy in the Era of Multi-core (멀티코어 시대의 변형 계층)

연사 : Yale Patt

(약력 : Professor of Electrical and Computer Engineering, University of Texas at Austin, USA)

변형 계층은 자연언어(영어, 스페인어, 힌디어, 일본어 등)로 진술한 문제점을 실제 솔루션 생성 작업을 하는 컴퓨터의 전자회로로 전환하는 메커니즘에 대해 내가 붙여준 이름이다.

문제는 먼저 자연언어로 서술한 것을 알고리즘으로 변형하여 그 후 기계언어 프로그램으로 변환하여 특정 프로세서의 ISA로 집계하는 것으로, 이것은 회로에서 만들어진 마이크로구조로 실행된다. 변형 계층 각 단계에서 몇 가지 선택사항이 있다. 이 같은 선택을 통해 몇 가지 최적화기준에 부합하도록 과정을 최적화할 수 있게 된다. 보통 그러한 기준은 마이크로프로세서 성능이다.

지금까지 최적화는 대개 각각의 층 안에서 실시되었으며, 층 사이에는 인공 경계선이 형성되어있다. (몇 가지 예외는 있지만) 어느 한 층의 지식은 다른 층의 최적화에 영향을 주고자 사용되는 경우는 없었다. 현재의 반도체기술 성장 속도로는 이 같은 변

형 층 안에서의 만족감은 더 이상 일반적인 사례가 아니라는 점을 나는 말하고 싶다. 이러한 성장 속도는 (현재는 칩에서 십억 트랜지스터 이상이 가능함) 칩 멀티프로세서 시대를 선도하였다.

다시 말해서 우리는 마이크로프로세서 성능 개선 II 단계에 들어서고 있으며, 여기서 개선은 변형 층을 분리하는 장벽을 파괴하는 것에서 비롯될 것이다. 본 프리젠테이션에서 나는 몇 가지 실행 방안을 제안하고자한다.

2. Plenary Session - Best Paper
Distributed Ranked Search(분산형 순위검색)

Vijay Gopalakrishnan(AT&T Labs - Research)
Ruggero Morselli(Google Inc)
Bobby Bhattacharjee(University of Maryland)
Pete Keleher(University of Maryland)
Aravind Srinivasan(University of Maryland)

개 요

P2P 배치는 분산형 검색 네트워크 구축을 위한 자연적인 인프라이다. 제안 시스템은 모든 검색 결과의 위치를 찾아내서 검색하도록 지원하지만, 결과의 순위를 정하는데 필요한 정보는 부족하다. 그러나 이용자는 주로 가장 관련이 있는 결과에만 관심이 있으며 모든 가능한 결과에 반드시 관심이 있는 것은 아니다.

무작위 표본추출을 통해서 분산형 상황에 적용할 수 있도록 잘 알려져 있는 정보검색 순위 알고리즘을 확대하고자한다. 우리는 우리의 접근법의 간접비용(overhead)을 분석할 것이다. 그리고 문서의 개수가 증가하는 경우와, 시스템 규모에 대해서, 또 문서를 노드로 나타내기(균일 대 비균일), 검색의 종류(희귀한 용어와 인기 있는 용어)에 따라 우리의 시스템이 어떻게 확장되는지를 수량화할 것이다.

우리의 분석과 시뮬레이션은 a) 이와 같은 확대가 효과적이라는 점을 보여주며 간접비용이 거의 없이 거대 시스템으로 확장하며, b) 분산형 순위 결정을 이용해서 얻은 결과의 정확도가 집중형 정확도에 필적한다는 점을 보여줄 것이다.

I. 도입(Introduction)

검색 인프라는 종종 애플리케이션 고유의 순위 개념에 따라 검색 결과를 순서대로 보여준다. 일반적으로 이용자는 모든 결과를 순서 없이 배열한 것보다는 순위에 따라 소규모 결과 집합으로 제시된 것을 선호한다. 예를 들어서 최근에 구글에서 “HiPC 2007"을 검색하면 475,000 웹페이지가 검색되었다. 검색 결과의 전체 집합은 거의 쓸모가 없다.

반면에 매우 작은 상위 결과에 희망 웹사이트가 포함되어 있을 것이다. 게다가 더 적은 수의 결과를 수집하면 소모되는 네트워크 주파수대역폭(bandwidth)이 줄어들게 되므로 -많은 이용자, 호스트, 자료 항목으로서는- 시스템이 확장되며, 하위 주파수대역폭 링크와 파워가 낮은 장치를 포함하도록 축소되도록 한다.

분산형으로 결과의 순위를 정하는 일은 어렵다. 왜냐하면 어떤 결과를 내보낼 것인가에 대한 결정은 지역적으로(locally) 이루어지는 반면에 그 같은 결정의 기초인 순위는 전체적(global)인 속성을 띠기 때문이다.

기술적인 측면에서는 하나의 노드를 지정하여 모든 검색 결과에 대해서 순위를 정하도록 책임지도록 할 수 있다. 그러나 그와 같은 접근법은 이 노드로 하여금 불공평한 작업량을 받아들이도록 할 것이다. 더구나 한 개 노드를 사용하는 경우 확장성(scalability)과 내(耐)고장성(fault-tolerance)에 문제가 있다.

본 논문의 중요한 기여는 임의 문서에 대해서 검색 결과를 능률적으로 그리고 일관성 있게 순위를 정하는 분산형 알고리즘을 설계하고 평가한 것이다. 우리의 접근법은 균일 무작위표본추출을 이용한 근사값 테크닉과 고전적인 집중형 벡터공간모델(VSM)에 기초하였다[1]. 우리의 결과는 구조화된 네트워크와 구조화되지 않은 네트워크 둘 다에 적용된다.

우리의 분석은 표본추출에 기초한 알고리즘 비용은 보통 작으며 시스템 크기가 증가해도 비용은 불변이라는 점을 보여준다. 우리의 분석을 뒷받침해줄 TREC 콜렉션[2]의 실제 문서 집합에 기초하여 시뮬레이션 결과를 제시할 것이다. 더 나아가서 결과는 프로토콜의 상수(constant)가 낮다는 것을 보여준다.

예를 들어서 5,000 노드 네트워크에서 검색당 20 노드 표본으로도 프로토콜은 매우 훌륭하게 작용한다. 또한 이 접근법은 표본추출오차, 최초 문서 분포 및 검색 위치에 대해서도 매우 훌륭한 것으로 드러났다.

본 논문의 나머지는 다음과 같이 구성된다. 먼저 섹션 2에서는 고전적인 정보 검색에서 순위를 정하는 것의 배경을 제시한다. 섹션 3에서는 결과의 순위를 결정하기 위한 우리의 설계를 토론하고 그것의 특징을 분석한다. 섹션 4에서는 분산형 순위 결정 시스템과 집중형 시스템의 작업 수행 결과를 비교하는 실험 결과를 제시한다. 섹션 6에서 결론을 맺기 전에 섹션 5에서는 그 외 관련 문제점을 논의한다.

II. 벡터공간모델(VSM, The Vector Space Model)

벡터공간모델(VSM)은 결과의 순위 결정을 위한 고전적인 정보 검색 모델이다. VSM은 문서와 검색을 T-디멘션 용어 공간에서 벡터로 나타내며 여기서 T는 수집한 문서 안의 용어의 개수이다. 문서 d 안의 각각의 용어 i에는 가중치 $w_{i,d}$가 주어진다. 문서 d에 대한 벡터는 $\boldsymbol{d} = (w_{1,d}, w_{2,d}, \ldots, w_{T,d})$ 로 정의된다. 검색어는 또한 벡터 q = $(w_{1,q}, w_{2,q}, \ldots, w_{T,q})$로 나타내지며 여기서 q는 문서로 취급된다.

유사한 벡터는 벡터 사이에 각(angle)이 작다. VSM은 이 같은 직관을 이용하여 주어진 검색어에 대한 관련 문서 집합을 계산한다.: 관련 문서는 작은 각 만큼 검색어 벡터와 다를 것이다.

반면에 관련 없는 문서는 큰 각 만큼 다를 것이다. 두개의 벡터 X와 Y에 대해서 그 둘 사이의 각 Θ는 $\cos\theta = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}}$를 이용하여 계산할 수 있다. 이 공식은 또한 코사인 유사성으로 알려져 있으며 관련 결과를 찾아내어 순위를 정할 때 전통적인 정보 검색에서 이용되어왔다.

1. 벡터표상 생성(Generating Vector Representation)

문서의 벡터 표상은 문서 안의 각 용어에 대한 가중치를 계산하여 생성한다. 중요한 것은 문서에서 의미를 포착하여 문서를 구별하도록 도와주는 용어에 대해서 더 높은 가중치를 주도록 가중치를 정하는 것이다.

효과적인 용어 가중치 공식은 많은 연구가 진행된 분야(e.g.,[3, 4])이지만 불행하게도 의견의 일치를 보지 못하고 있다. 널리 사용되고 있는 공식 중 어느 것이라도 우리의 시스템에 사용할 수 있겠으나, 실제로 훌륭한 검색 특징을 가진 것으로 입증된 SMART [5] 시스템에 이용된 가중치 공식을 사용할 것이다.:

$$w_{t,d} = (\ln f_{t,d} + 1) \cdot \ln\left(\frac{D}{D_t}\right) \qquad (1)$$

여기서 $w_{t,d}$는 문서 d의 용어 t 가중치이며, $f_{t,d}$는 문서 d의 용어 t 빈도수이다. D는 수집한 문서의 총 개수이고 D_t는 용어 t가 포함된 수집한 문서의 개수이다.

III. 분산형 VSM 순위(Distributed VSM Ranking)

이 섹션에서는 키워드 기반 검색에 대한 우리의 분산형 VSM 순위 시스템을 제시한다. 순위 결정에 필요한 주요 구성성분으로는 세 가지가 있는데, 내보낸 문서에 대한 벡터 표상 생성, 문서 벡터를 적절하게 저장하기, 그리고 검색 결과를 계산하여 순위를 결정하기이다. 먼저 우리가 가정하고 있는 시스템을 서술하고 그 후 이 구성성분 각각을 자세하게 논의할 것이다.

1. 시스템 모델(System Model)

우리의 순위결정 알고리즘은 구조화된 P2P 시스템과 구조화되지 않은 시스템 둘 다를 위해 설계되었다. 우리의 알고리즘은 각각의 키워드에 대해서 역(inverted) 색인정보(index)를 구성하며 이러한 색인정보를 참여 노드에 대해서 분산시킨다. (노드는 협력한다고 가정한다.)

키워드의 역 색인정보는 키워드가 들어있는 문서 모두의 목록을 저장한다. 기저의 P2P 시스템은 색인정보를 노드로 (색인정보를 저장하는) 나타내기 위해 필요한 자료조회(lookup) 메커니즘을 제공한다고 가정한다.

구조화된 시스템은 모두 자료조회 API가 있는 반면에, 구조화되지 않은 시스템에 대해서는 자료조회의 경우 LMS [6]와 Yappers [7]등의 접근법에 의존한다. 기저의 P2P 시스템은 색인정보가 노드에 어떻게 나타내는지를 명령하는데, 구조화된 P2P 시스템은 색인정보를 한 장소에 저장하지만 구조화되지 않은 시스템의 경우 여러 장소에 분산 저장될 것이다.

각각의 노드는 시스템에 연결되면 문서 집합을 내보낸다. 키워드 집합은 (디폴트로 문서안의 모든 단어) 문서와 관련이 있다. 문서를 내보내는 과정은 각 키워드와 관련된 색인정보 안의 문서에 하나의 개체를 추가하는 것으로 이루어진다.

검색이 이루어질 때 이용자는 키워드가 포함된 검색어를 입력하고 가장 상위 K 결과만을 보여주도록 지정할 수 있다. 그러한 경우 시스템은 이 K 결과를 분산형으로 계산하여 이용자에게 보여준다.

2. 문서 벡터 생성(Generating Document Vectors)

문서 벡터를 생성하기위해서 문서의 각 용어에 가중치를 주어야한다는 점을 상기해보자. 또한 공식 (1)을 상기해보자. 공식 (1)은 문서안의 각 용어 t의 가중치 계산에 사용된다.

공식은 두 가지 구성 성분으로 되어있는데, 하나는 로컬 구성성분, $\ln f_{t,d} +1$ 이며 이것은 주어진 문서의 상대적 중요도를 포착한다. 또 다른 하나는 전체 구성성분, $\ln(D/D_t)$이며 이것은 얼마나 드물게 이 용어가 문서 전체에서 사용되는지를 설명한다.

로컬 구성성분은 문서의 단어 빈도수를 계산하여 쉽게 얻을 수 있다. 전체 구성성분은 시스템의 문서 D의 숫자와 용어 t가 들어있는 문서 D_t의 숫자로 언급된다. 우리는 이 측정값을 추정하기 위해서 무작위 표본추출을 이용한다.

N을 시스템의 노드 수로 하고 D와 D_t를 위의 정의를 따르도록 해보자. 무작위로 k 노드를 선택한다. 이것은 구조화되지 않은 시스템에서는 랜덤워크(random walk)로 하거나 [6], 구조화된 시스템에서는 네임스페이스의 랜덤포인트로 경로를 지정하거나 [8]하여 실시할 수 있다.

그 후 문서의 총 수 $\tilde{D}$와 표본 노드에 용어 t가 포함된 문서의 총 수 $\tilde{D}_t$를 계산한다. 간단하게 말하자면 동일 노드는 한 번 이상 표본이 될 수 있음을 인정한다. E가 무작위변수의 기댓값을 가리키는 $\mathrm{E}[\tilde{D}] = k\frac{D}{N}$와 $\mathrm{E}[\tilde{D}_t] = k\frac{D_t}{N}$라는 점을 이해하기는 쉽다.

직관적으로 만약 표본이 충분하다면, $\tilde{D}$와 $\tilde{D}_t$는 기댓값에 상당히 근접한다는 것을 알 수 있다. 그러한 경우 D/D_t는 $\frac{D}{D_t} \approx \frac{\tilde{D}}{D_t}$로 추정할 수 있다. 이 근사값의 충분조건을 찾아내기 위해 두 가지 새로운 수량을 도입하자. M과 M_t을 각각 노드의 최대 문서수와 용어 t가 들어있는 최대 문서수로 해보자.

추정값 $\tilde{D}$(각각 $\tilde{D}_t$)의 경우 만약 기댓값이 계수 $(1\pm\delta)$이내라면 "훌륭하다"고 한다. 추정값은 작은 확률(€)의 경우 "나쁘다"고 할 수 있다. 보통의 경우처럼 e는 자연로그의 밑(base)이라고 정하자.

정리 1. D, N, k, M은 위의 정의를 따르자. $0 < \delta \leq 1$와 $\epsilon > 0$에 대해서 만약

$$k \geq \frac{3}{\delta^2} \frac{M}{D/N} \ln(2/\epsilon) \tag{2}$$

이라면 무작위변수 $\tilde{D}$는 (위에서 정의한 바와 같이) 최대 확률 ϵ인 경우를 제외하면 평균에 매우 가깝다. 보다 엄밀하게 말해서 다음이다.

$$\Pr[(1-\delta)kD/N \leq \tilde{D} \leq (1+\delta)kD/N] \geq 1-\epsilon. \tag{3}$$

증명. 증명은 Chernoff 유계를 응용하는 것이다 [9]. i = 1, . . ., k에 대해서 Y_i를 i번째 표본에서 발견된 문서의 수를 나타내는 무작위변수라고 하자. $\tilde{D} = \sum_{i=1}^{k} Y_i$임에 주목하자. Chernoff 유계를 적용하기 위해서는 [0,1] 구간의 무작위변수가 필요하다. $X_i = Y_i/M$으로 하고 $X = \sum_{i=1}^{k} X_i = \tilde{D}/M$로 하자. 정의하면:

$$\mu = \mathrm{E}[X] = \frac{kD}{MN}.$$

X_i는 [0,1] 안에 있고 독립적이므로 우리는 Chernoff 유계를 사용할 수 있다. 이것은 $0 < \delta \leq 1$에 대해서 다음임을 알려준다.

$$\Pr[|X - \mathrm{E}[X]| > \delta \,\mathrm{E}[X]] \leq 2e^{-\frac{\mu\delta^2}{3}},$$

이것은 다음과 같이 다시 정리할 수 있다.

$$\Pr[(1-\delta)kD/N \leq \tilde{D} \leq (1+\delta)kD/N] \geq 1 - 2e^{-\frac{\mu\delta^2}{3}}.$$

이제 우리는 위의 확률이 최소 1 - ϵ인 제약을 부과해보자.

$$2e^{-\frac{\mu\delta^2}{3}} \leq \epsilon,$$

이것으로부터 k에 대한 유계를 구할 수 있다.:

$$k \geq \frac{3}{\delta^2} \frac{M}{D/N} \ln(2/\epsilon).$$

만약 D, M, $\tilde{D}$를 D_t, M_t, $\tilde{D}_t$로 대체하고, 정리가 의미하는 바가 만약 다음이라면

$$k \geq \frac{3}{\delta^2} \frac{M_t}{D_t/N} \ln(2/\epsilon) \tag{4}$$

무작위변수 $\tilde{D}_t$는 또한 좋은 추정값이다.

다음의 관찰은 정리 1에서 온 것이다.:

- 정리 1은 좋은 추정값의 경우 필요한 표본의 수는 N에 직접 의존하는 것은 아니지만 수량 D/N과 D_t/N에는 의존하며 덜 중요하지만 M과 M_t에도 의존한다는 점을 보여준다. 이것이 뜻하는 바는 시스템규모가 커짐에 따라, 내보내는 문서의 수가 (용어 t가 들어있는) 증가하는 한 더 많은 표본이 필요하지는 않다는 것이다.

- 만약 문서의 수 D가 시스템 규모 N보다 훨씬 더 크고 검색이 인기가 있는 용어로 구성된다면 (D_t = Ω(N)), 우리의 알고리즘은 수행결과에 대해서 이상적인 확장 행동을 가져다준다. 일정한 수의 노드를 추출하면 시스템 규모와 관계없이, 정확한 결과를 가져다준다.

- 실제로 문서와 쿼리에는 희귀한(다시 말해서 인기 없는) 용어가 포함되며, 그것에 대해서 1n(D/D_t)는 부정확하게 추정될 수 있다. 그러나 그와 같은 추정오차는 중요하지 않으며 피할 수 없는 것이다. 추정은 비교적 중요하지 않다. 왜냐하면 쿼리에 희귀한 용어가 포함되면 전체 결과 집합은 비교적 작으며, 소규모 집합의 순위 결정은 중요한 것은 아니다. 일반적으로 표본 추출은 희귀한 특징을 추정하기에는 좋은 접근법은 아니므로 기타의 접근법이 필요하다.

- 표본의 수는 노드(즉, $\frac{M}{D/N}$와 $\frac{M_t}{D_t/N}$)에 저장된 최대 문서의 수와 평균 문서 수의 비율에 비례한다. 이것이 뜻하는 바는 시스템의 문서 분포가 불균형적일수록 정확한 결과를 얻기 위해서는 더 많은 표본이 필요하다.

특별 사례 : 균일 분포. 이제 기저 저장 시스템이 노드에 무작위로, 균일하게 무작위로 그리고 독립적으로 문서를 분산하는 특별 사례로 관심을 제한해보자. 그와 같은 분산은 대략 DHT의 행동을 모형화한다. 이 특별 사례에서 만약 대체 없이 표본을 추출한다면 정리 1의 강한 버전이 유효하다. (대신 이미 표본으로 뽑힌 노드를 표본으로 추출한다면 이 표본은 기각되며 다시 표본을 추출하는 기각 표본추출이 이루어진다.)

정리 2. D, N, k를 위의 정의와 같이 정의해보자. 그리고 각각의 문서는 독립적으로 그리고 하나의 노드에 균등하게 무작위로 저장된다고 가정해보자. €>0인 경우, 만약 k

표본을 대체 없이 다음으로 선택한다면,

$$k \geq \frac{3}{\delta^2}\frac{1}{D/N}\ln(2/\epsilon), \tag{5}$$

무작위 변수 $\tilde{D}$는 (위의 정의와 같이) 최대 확률 ϵ의 경우를 제외하면 평균 kD/N에 매우 근접하다. 보다 엄밀하게 말해서 다음이다.:

$$\Pr[(1-\delta)kD/N \leq \tilde{D} \leq (1+\delta)kD/N] \geq 1-\epsilon. \tag{6}$$

이 확률은 우리의 표본추출의 임의성 뿐만 아니라 문서의 임의 분포에 대해서도 채택한다.

증명. 정리 1의 증명과는 반대로, 문서를 독립적으로 나타낸다는 사실을 이제 결정적으로 이용한다. 모든 문서 d에 대해서 임의변수 X_d를 문서 d가 k 표본 노드 중 하나에 저장되는 사건의 지표로 정의해보자. (다시 말해서 만약 이 사건이 발생하면 X_d는 1이며 그렇지 않으면 0이다.) 그러므로 $E[X_d] = k/N$이다. $X = \sum_d X_d$로 정의하면 $E[X] = kD/N$임을 알 수 있다.

서로 다른 임의변수 X_d는 독립적이라는 점을 주목하자. 이것은 우리가 대체 없이 표본을 추출한다는 사실의 결과이다. 이 점을 이해하기 위해 별개의 문서 $d_1, d_2, \ldots, d_i$가 있다고 가정해보자. 우리는 $\Pr[\bigwedge_{j=1}^{i} X_{d_j} = 1] = \prod_{j=1}^{i} \Pr[X_{d_j} = 1]$를 보여주고자 한다. 여기서 S는 k 별개 표본 집합을 나타내는 임의변수라고 하자. 그리고 U는 k 노드 전체 집합으로 하자. S 값이 알려지면 다음과 같이 쓸 수 있다.

$$\Pr[\bigwedge_{j=1}^{i} X_{d_j} = 1] = \sum_{S^* \subset U} \Pr[S = S^*] \Pr[\bigwedge_{j=1}^{i} X_{d_j} = 1 | S = S^*] =$$

$$= \sum_{S^* \subset U} \Pr[S = S^*](k/N)^i = (k/N)^i = \prod_{j=1}^{i} \Pr[X_{d_j} = 1],$$

여기서 문서는 독립적으로 저장되며 d_i가 S*에 저장될 확률은 정확하게 k/N이라는 사실을 이용한다. X_d는 이진수 값이므로 위의 공식이 의미하는 바는 X_d는 서로 독립적이라는 것이다. 이것은 X_d에 대한 가능한 할당 확률은 위에서와 같이 선형확률 조합으로 쓸 수 있기 때문이다. (예를 들어서 이것은 Fourier 분석을 따른다.) 예에서와 같이 다음을 주목하자.

$$\Pr[X_{d_1} = 1,\ X_{d_2} = 0] \ = \ \Pr[X_{d_1} = 1] - \Pr[X_{d_1} = X_{d_2} = 1].$$

이제 우리가 원하는 결과를 얻기 위해 X에 Chernoff 유계를 적용하자. 세부 계산식은 정리 1의 증명에서와 동일하다.

그러므로 균등한 경우 표본의 수는 어떤 노드에서든 문서의 최대수 M에 비례할 필요는 없다. 그러므로 우리의 표본추출 알고리즘 비용은 상당히 감소한다.

3. 문서 벡터 저장

문서 벡터는 문서에 관한 검색에서 문서 벡터의 위치를 빠르게 찾을 수 있도록 저장해야한다. 문서 벡터는 분산형 역 색인정보에 저장한다. 앞서 언급하였듯이 키워드 t에 대한 역 색인정보는 t가 포함된 모든 문서의 목록이다. 각각의 키워드 t에 대해서 우리의 시스템에서는 기저 P2P 자료조회 시스템 안의 다른 물체와 마찬가지로 해당 역 색인정보를 저장한다. 이렇게 함으로써 검색어와 최소한 용어 하나를 공유하는 모든 문서의 벡터를 효과적으로 찾아낼 수 있게 된다.

Figure.1은 문서를 내보내는 과정을 보여준다. 먼저 용어의 가중치를 계산함으로써 해당 문서 벡터를 생성하는데 이때 섹션 3.2에서 서술한 절차를 적용한다. 그 후 기저 저장 시스템 API를 이용하여 문서안의 각 용어와 관련된 색인정보를 저장하는 노드를 찾아내서 색인정보에 추가한다.

역 색인정보에 문서 벡터를 저장하는 문제는 기저 자료조회 프로토콜에 달려있다. 구조화된 시스템의 경우 키워드 t에 대해서 t의 색인정보는 t에 해당하는 키를 책임지고 있는 노드에 저장된다. 기저 프로토콜을 사용하여 효과적으로 이 노드의 위치를 찾아낸다. 역 색인정보를 이용하는 유사 접근법은 기존에 구조화된 시스템에서 검색을 위해 [10, 11, 12, 13]에 의해 사용되었다. 구조화되지 않은 네트워크에서 색인정보는 분할되거나 복제되어야한다 [7, 6].

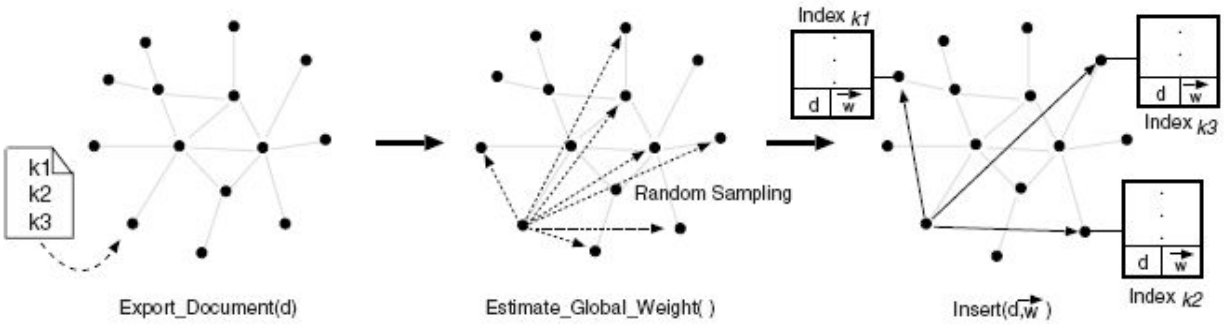

Fig.1. 문서 이출(exporting)의 다양한 단계와 백터표상

저장비용 절감. 지금까지는 문서 안의 각각의 단어가 키워드라고 가정하였다. 그러므로 하나의 개체는 문서 안에 있는 모든 단어로 된 색인 정보 문서에 첨가된다. 그러나 문서는 검색 용어 가중치가 낮은 경우 상위 결과에 나타나지 않는다. 그러므로 색인정보에 이렇게 낮은 가중치의 개체를 갖지 않으면 상위결과의 검색 품질이 감소하지 않는다.

이 같은 직관을 이용하여 색인정보에 벡터를 전달하고 저장하는 비용을 줄인다. 불변의 분계점 w_{min}이 있다고 가정하는데 이것은 문서 개체가 색인정보에 추가될 것인지 아닌지를 결정한다. t의 가중치가 분계점 w_{min} 이하라면 벡터는 용어와 일치하는 색인정보에 추가되지 않는다. 이러한 분계점 이하의 가중치를 가지는 용어도 여전히 벡터의 일부라는 점을 명심하자. 이러한 발견적 접근법은 또한 eSearch[10]에서 성공적으로 적용되었다.

4. 검색 결과 평가

검색은 벡터 표상으로 변환한 후 각각의 "관련" 문서 벡터 측면에서 코사인 유사성을 계산함으로써 평가한다. 문서 벡터 생성에 이용하는 것과 동일한 테크닉을 이용하여 검색 벡터를 계산한다. 그 다음 단계에서 관련 문서 집합의 위치를 찾아낸다. 검색의 각 키워드에 대해서 해당 키워드 색인정보를 저장하고 있는 노드를 찾아내기 위해서 기저 시스템이 제공한 자료조화 함수를 이용한다.

그 후 검색과 색인정보에 저장된 문서 벡터 각각 사이의 코사인 유사성을 계산한다. 이렇게 하면 이 색인정보의 이용 가능한 문서 순위가 주어진다. 마지막으로 색인정보 각각에 대해 계산한 상위-K 결과를 인출하고 이 결과 집합의 합집합을 계산한다. 코사인 유사성의 내림차순으로 분류한 이 합집합 안의 상위-K 문서는 우리에게 최종 결과 집합을 제공한다.

Ⅳ. 평가

이 섹션에서는 시뮬레이션을 통해 우리의 분산형 순위 결정 시스템을 검증한다. 우리의 알고리즘에 의한 검색 결과의 품질과 VSM의 집중형 시스템의 검색 결과를 비교함으로써 수행 결과를 측정한다.

실험 구성. 우리의 문서에 대해서 TREC [2] Web-10G 자료 집합을 이용한다. 실험에서 이 자료 집합의 처음 십만 개 문서를 사용하였다. 이 100K 문서에는 대략 418K 유일

용어가 들어있다. 디폴트 시스템 크기는 1000 노드로 구성되어 있다. 노드에 대해서 두 개의 서로 다른 문서 분산을 이용하는데, 구조화된 P2P 시스템의 경우 문서 분산 모델에 균등 분산과 구조화되지 않은 시스템의 경우 분산 모델에 Zipf 분산을 이용한다.

우리의 대량 자료 집합(100K 문서)은 그것과 관련된 검색이 없었기 때문에 서로 다른 길이의 검색을 생성하였다. 우리의 디폴트 검색 집합은 대략 5000개의 문서에 나타나는 용어로만 구성된다.

우리의 실험에서 이 검색 집합을 Q_{5k}로 표시한다. 이렇게 검색 용어를 골라내는 이면의 직관은 상당한 수의 문서에 나타나므로 인기가 있다는 점이다. 동시에 문서를 구분할 만큼 충분히 유용하다. 또한 매우 인기가 있거나 (10K 문서 이상에서 나타나는) 또는 매우 희귀한 (200개의 문서 이하에서 나타나는) 키워드를 배타적으로 포함하는 검색집합을 사용 한다.

이와 같은 검색 집합을 각각 Q_{pop}와 Q_{rare}로 표시한다. 제시되는 각각의 결과는(개별 실행에 대한 세부 정보는 제외) 평균적으로 50회 실행(run)된다. 분산 순위 결정의 품질 평가에는 세 가지 메트릭스를 이용한다.

1) **자료 범위**: 자료 범위는 동일 검색에 대해서 집중형 VSM을 실행하여 얻은 상위-K 결과에도 나타나면서 분산형 시스템으로 얻은 상위-K 검색결과의 수로 정의한다. 예를 들어서 만약 상위 3개의 결과에 관심이 있고 분산형 시스템에서는 (A, C, D) 문서를 보여주는 반면에, 집중형 시스템에서는 (A, B, C) 문서를 보여준다면, 이 검색의 자료 범위는 2이다.

2) **자료 인출**: 자료 인출은 이용자가 분산형 시스템의 순위에 따른 R′ 결과 집합을 획득하는 경우 동일 검색에 대해서 R′에 집중형 시스템을 실행하여 얻은 상위-K 결과 모두를 포함하도록 하는 최소의 숫자 R′로 정의한다. 앞의 예에서 분산형 사례로 얻은 4번째 결과가 B라면 K=3에 대한 인출은 4가 될 것이다.

3) **일관성**: 일관성은 동일 검색에 대해서 서로 다른 표본을 이용한 서로 다른 런의 경우 검색 순위에서의 유사성으로 정의한다.

1. 자료 범위

첫 번째 실험에서 우리는 분산형 자료 검색 시스템의 자료 범위를 측정한다. 제법 대규모인 시스템에서 몇 개의 노드만을 표본 추출하는 것으로도 우리의 시스템이 집중형 시스템과 매우 근접한 결과를 생성한다는 사실을 보여줄 것이다.

기본 결과를 보면 우리는 1,000개의 노드 네트워크를 이용한다. 문서는 노드에 균등하게 나타내진다. 용어 t의 전반적 가중치를 계산하기위해서 우리는 실험의 서로 다른 실행에서 10, 20, 50 노드를 추출한다. 검색은 Q_{5K} 검색 집합의 키워드로 구성된다. 다시 말해서 키워드는 대략 5000개의 문서에 나타난다.

결과가 표 1에 들어있다. 표 1에서 분명한 것은 분산형 순위결정 시스템은 집중형 시스템 이행 결과와 매우 유사한 수행을 보인다는 것이다. 문서를 균등하게 분산하는 1000개 노드 네트워크의 경우 상위-K 결과의 평균 정확도는 50개 무작위 표본의 경우 93%에 이른다. 10개 무작위 표본의 경우에도 결과는 85% 정확도로 약간만 나빠질 뿐이다.

5,000개 노드의 경우 검색의 품질은 1,000개 노드의 네트워크에서만큼 높지는 않다. 20개 무작위 표본의 경우 상위-K 결과에 대해서 평균 정확도는 77%이다. 표본추출 수준을 높이고 노드의 1%(50)을 방문하는 경우 평균 정확도는 8% 증가한다. 이 결과는 정리 2의 직접적인 결과이다. 여기서 문서의 수는 동일하다. 그러나 노드의 수는 증가하였다. 그러므로 표본으로 추출되는 노드의 수가 높을수록 더 나은 측정값이 얻어진다.

Table 1. 분산형 순위결정시스템에 의한 자료범위의 평균과 표준편차

Network Setup	Number of Samples	Top-K results				
		10	20	30	40	50
1000 uniform	10	8.49 (1.08)	16.99 (1.20)	25.30 (1.55)	33.68 (2.07)	42.28 (2.01)
	20	8.90 (0.99)	17.81 (1.04)	26.44 (1.26)	35.23 (1.87)	44.30 (1.82)
	50	9.28 (0.82)	18.63 (0.82)	27.66 (1.04)	36.08 (1.45)	46.30 (1.46)
5000 uniform	10	6.78 (1.39)	13.58 (1.74)	20.43 (2.39)	27.35 (2.99)	34.59 (3.40)
	20	7.74 (1.29)	15.41 (1.46)	22.92 (1.96)	30.50 (2.47)	38.49 (2.58)
	50	8.52 (1.09)	16.96 (1.18)	25.20 (1.56)	33.59 (2.11)	42.34 (1.98)
1000 Zipf	10	8.27 (1.15)	16.52 (1.26)	24.66 (1.71)	32.82 (2.21)	41.20 (2.27)
	20	8.82 (0.99)	17.63 (1.06)	26.22 (1.35)	34.83 (1.93)	43.70 (1.88)
	50	9.26 (0.80)	18.54 (0.88)	27.52 (1.12)	36.71 (1.49)	46.12 (1.56)
5000 Zipf	10	6.09 (1.54)	12.29 (1.97)	18.58 (2.68)	25.01 (3.39)	31.67 (3.97)
	20	7.34 (1.31)	14.71 (1.62)	21.89 (2.10)	29.34 (2.64)	36.93 (2.90)
	50	8.41 (1.13)	16.73 (1.22)	24.92 (1.61)	33.22 (2.08)	41.71 (2.03)

또한 Table 1은 모수를 0.80으로 하여 Zipf 분산을 이용하는 경우 노드로 나타낸 문서의 검색 품질을 보여준다. 1,000개 노드와 50개 표본의 경우 검색 품질은 균등 분산의 품질과 비슷하다. 그러나 10개 표본의 경우 평균 정확도는 82~83% 사이로 몇 %p 하락한다. 5,000개 노드와 50개 표본의 경우 이와 유사한 추이를 보인다. 품질은 균등 분산형 자료의 경우처럼 좋지는 않지만, 2% 이상 다르지는 않다. 10개 표본의 경우 그 결과는 7% 정도 악화된다. 그러므로 우리의 시스템은 품질 하락을 느끼지 못하면서 구조화되지 않은 네트워크상의 자료조화 프로토콜에서도 적용될 수 있다.

2. 자료 인출

이 실험에서 우리는 집중형 시스템 시행에 의한 상위-K 결과 모두를 이용할 수 있게 되기 전에 얼마나 많은 수의 결과를 인출해야 하는지를 측정한다. (이 측정값을 우리는 자료 인출이라고 명명하였다.) 문서가 균등하게 분산된 1,000개 노드와 5,000개 노드 둘 다를 가지고 실험한다. 평가에 Q_{5K} 검색 집합을 이용하였다. Figure 2와 3에서 결과를 구상한다. x축은 집중형 시스템 이행에 의한 상위-K 결과이고, y축은 상대 평균 자료 인출을 나타낸다.

1,000개 노드 네트워크의 경우 비록 10개 노드의 표본을 추출한다고 할지라도 자료 인출은 매우 작다는 것을 알 수 있다. 예를 들어서 10개 노드의 표본을 추출할 때 집중화된 케이스의 상위-10개 결과와 매치하기 위해 13개의 결과가 필요하다. 50개 노드의 표본을 추출할 때 덜 관련된 문서의 경우 자료 인출은 최소이다. 집중형 시스템을 이행하는 경우 희망하는 상위-10 결과를 대응시키려면 11개 개체가 필요하며 상위-50 결과를 대응시키려면 63개가 필요하다.

예상한 것과 같이 네트워크의 규모가 증가하면 동일 문서 집합에 대해서 자료 인출은 증가한다. 5000개 노드의 1%를 표본 추출하는 경우 상위-10 결과를 포함하려면 13개의 결과가 필요하며 상위-50 결과를 포함하려면 88개의 결과가 필요하다. 이 같은 행동 또한 정리 2에서 예견되었다.: 해당 문서의 수가 증가하지 않고 노드의 수가 증가하는 경우 표본추출 오차의 한계를 보장하기위한 표본 또한 증가한다.

다른 실험에서도 문서 분산이 비대칭인 경우 유사한 결과를 보여준다. 그 결과를 여기에 간단히 요약해보자. 1,000개 노드 네트워크와 10개 무작위 표본의 경우 자료 인출은 문서를 노드에 균등하게 나타낸 경우의 네트워크와 비교해서 10% 증가한다. 5,000개 노드 네트워크의 경우 균등 사례와 비교해서 35% 증가한다. 그러나 50개 무작위 표본에 대한 이 두 네트워크의 결과는 균등 사례와 비슷하다.

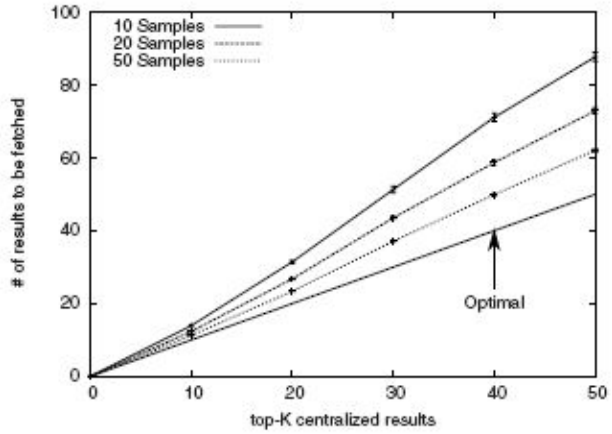

Fig.2. 1,000개 노드 분산형순위결정시스템의 평균자료인출. 오차막대는 95% 신뢰구간에 해당한다.

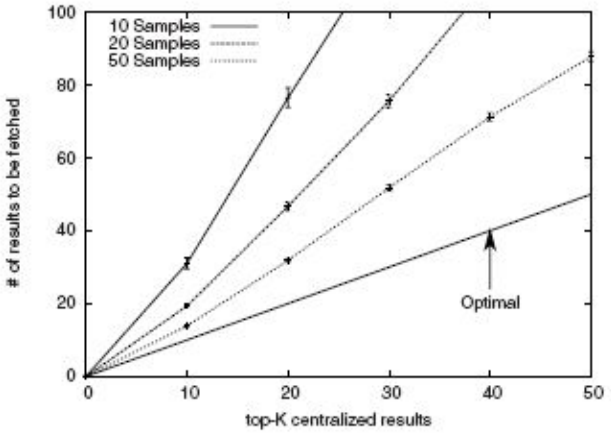

Fig.3. 5,000개 노드 분산형순위결정시스템의 평균자료인출. 오차막대는 95% 신뢰구간에 해당한다.

3. 일관성

우리의 시스템에서 새 검색 벡터는 검색을 평가할 때마다 생성된다. 이렇게 되면 동일 검색의 서로 다른 평가에 대해서 용어에는 서로 다른 가중치가 주어진다. 이렇게 되면 순위의 분산이 증가할 수 있으며 잠재적으로 검색에 대한 서로 다른 평가에 대해서 서로 다른 결과가 나올 수 있다. 이 실험에서 우리는 그렇지 않다는 것을 보여주며 결과는 서로 다른 표본에 의해서 최소한으로 영향을 받는다는 것을 보여준다.

1,000개 노드 네트워크를 이용하고 문서는 이 노드에 균등하게 나타냈다. 검색 벡터를 계산하는 동안 20개 무작위 노드의 표본을 추출한다. Q_{5K}를 이용하고 서로 다른 런에 대해서 상위-50개의 결과를 기록하고 서로 결과를 비교하고 집중형 시스템 이행과도 비교한다.

Figure 4는 세 가지 대표 검색 각각에 대해서 5개의 대표적인 Run으로 획득한 결과를 보여준다. 각각의 Run에 대해서 그림은 만약 문서를 이와 같은 Run 동안 검색한 경우 집중형 VSM으로 상위-50에 랭크된 문서에 해당하는 작은 박스를 포함한다.

예를 들어서 Figure 4에서 검색 1, 런 2는 1 . . . 25 랭크된 문서를 검색하였다. 그러나 상위 50개의 결과에서는 26으로 랭크된 문서를 보여주지 않는다. 또한 주목할 점은 처음 25개 집중형의 순위 문서는 반드시 그러한 순서로 순위가 매겨질 필요는 없지만 각각은 상위-50 이내로 검색되었다.

두 가지 관찰 사항이 있다. 첫째, 표본추출은 결과의 일관성에 역으로 영향을 끼치지는 않는다. 서로 다른 수행에 의해 본질적으로는 동일한 결과가 돌아온다. 더구나 이 같은 결과는 상위 결과의 자료 범위는 균등하게 우수하다는 것을 보여준다는 점을 주목하자. 그리고 검색되지 않은 문서는 일반적으로 집중형 순위에 따르면 상위-50의 하위부분에 랭크된다는 점을 주목하자. 사실상 우리의 자료를 상세하게 분석하면 이 추이는 우리의 다른 실험에서도 유효한 것으로 나타난다.

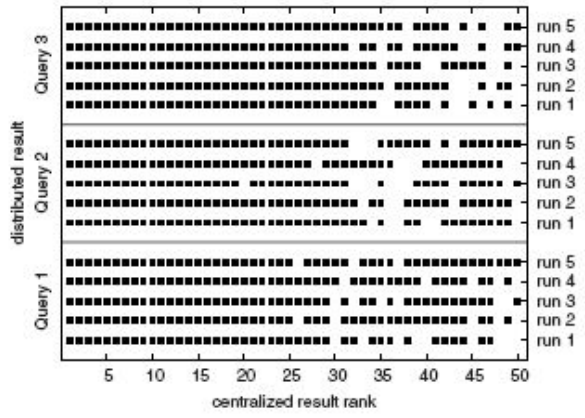

Figure.4. Q_{5k}집합에서 세 개의 서로다른 검색의 경우 분산형 순위결정에서 상위-50 결과의 일관상

Table 2. 노드와 문서의 수가 비례하여 확장되는 경우의 평균 자료 범위

Network Setup	Top-K results 10	20	30	40	50
500 nodes	7.75	16.11	25.08	33.62	42.24
1000 nodes	7.99	16.33	24.58	32.98	41.59
2000 nodes	7.67	15.85	23.96	32.00	40.11
5000 nodes	6.95	15.21	22.99	30.66	38.85

4. 확장성

이 실험에서는 시스템 크기가 증가하는 경우 우리의 시스템의 확장성을 평가한다. 정리 1과 2는 문서 집합의 크기가 노드의 수에 비례하여 증가하는 경우, 필요한 표본의 수는 시스템 크기와 상관이 없는 것으로 언급하고 있다. 우리는 동일한 수의 노드를(20) 추출하면서 시스템 크기가 10배 증가할 때(500에서 5,000으로) 자료 범위는 대략 불변으로 있음을 보여줌으로써 이러한 사실을 입증한다.

각각의 실험에서 문서의 수는 시스템이 노드 수의 20배이다. 모든 구성의 경우, 검색에 사용된 용어는 총 문서의 10% 이상에 나타난다. 5,000개 노드 네트워크의 경우 이것은 Q_{pop} 검색 집합에 해당한다. 각각의 경우에 우리는 전반적인 가중치 추정에 20개 무작위 노드를 추출한다.

Table 2는 우리의 분산형 시스템의 평균 자료 범위를 보여준다. 표에서 보여주는 것처럼 분산형 자료 검색의 자료 범위는 대부분의 경우 매우 유사하다. 이 결과는 우리의 시스템은 노드마다의 문서 밀도에 전적으로 의존함을 확증한다. 그리고 밀도가 유사하면 시스템이 확장됨을 확증한다.

5. 저장 비용 절감

키워드의 가중치가 분계점 w_{min}보다 큰 경우 키워드의 색인정보에서 문서 벡터를 저장하기위한 최적화를 연상해보자. 본 실험에서 이러한 최적화의 결과를 수량화한다. 이 실험의 경우 문서는 노드에 대해서 무작위로 균등하게 분산되며 1000개 노드 네트워크를 이용하였다. 가중치를 추정할 때 세 가지 검색 집합 모두를 이용하며 20개 노드를 표본 추출한다. 벡터를 표준화함을 주목하자. 그러므로 용어 가중치는 0.00에서 1.00 범위에 있다. 우리는 0.00에서 0.30까지의 분계점에 대해서 결과를 제시한다. 집중형 시스템을 이행한 것으로 얻은 검색 결과를 w_{min} = 0.00와 비교한다.

이 실험의 결과는 Table 3으로 작성한다. 분산 순위의 자료 범위는 분계점이 0.05에서 0.10으로 정해지는 경우 역으로 영향 받지는 않는다. 그러나 분계점이(가령 0.20 이상의 경우) 큰 경우 관련 개체는 폐기한다. 그 결과 순위 품질은 눈에 띠게 감소한다. 분계점을 이용하여 얻어진 절감을 이해하기위해 각 분기점별로 시스템의 색인정보 개체의 총 개수를 기록하였다.

Weight	Q_{5K}			Q_{pop}			Q_{rare}		
Threshold	10	30	50	10	30	50	10	30	50
0.0 (0.0)	8.90	26.44	44.30	8.32	26.31	44.49	8.59	26.01	44.47
0.05 (55.5)	8.90	26.44	44.34	8.33	26.32	44.40	8.50	26.01	44.47
0.10 (85.0)	8.90	26.40	44.22	8.32	26.17	43.87	8.59	26.01	43.54
0.20 (97.2)	7.64	20.43	30.97	6.39	17.90	26.70	8.46	21.41	28.43
0.30 (99.3)	4.53	7.98	8.88	2.79	6.84	9.90	6.66	9.78	9.99

Table 3. 가중치 분계점이 서로 다른 경우의 분산형 순위결정시스템의 평균자료범위. 괄호안의 숫자는 서로 다른 분계점에 해당하는 색인정보 구모의 백분율 감소이다.

우리의 시스템에서 분계점이 0.0인 경우 색인정보 개체의 총 개수는 15.9M이다. 본 실험은 분계점을 0.05로 하는 경우 개체가 55.5% 축소하는 것으로 나타났다. 분계점을 0.1로 늘리면 색인정보 크기는 30% 추가로 축소된다. 분계점 값 0.1는 검색 품질과 색인정보 크기 감소 사이의 적정한 거래 조건인 것으로 보인다.

Ⅴ. 관련 연구

본 논문은 능률적인 자료 조회 및 저장 시스템에 대한 기존 작업을 토대로 하고 있다. 그러한 자료 조회 프로토콜은 구조화된 상황과(예. Chord [14], Pastry [15]) 구조화되지 않은 상황(예. Yappers [7], LMS [6]) 둘 다에서 자세하게 연구되었다. 유용한 검색 기능의 제공은 중요한 연구 분야였다. 검색의 기존 작업은 광범위하게 두 범주(전통적인 집중형 접근법과 구조화된 P2P 네트워트상의 검색 전략)로 분류할 수 있다.

고전적인 정보 검색. 정보 검색과 순위 분야에 많은 노력을 기울였다. 섹션 2에서 벡터 공간모델[1]에 대해 토론하였다. 잠재 의미론 색인(LSI) [16]은 동의어와 다의어 문제를 해결하고자하는 VSM의 확장이다. LSI는 VSM으로 생성한 매트릭스를 줄이기 위해 특이값 분해(SVD)를 이용한다. LSI를 완전 분산형으로 어떻게 실행할 수 있을지에 대해서는 여전히 의문이다. 또한 분산형 상황에서 (예. [18]) PageRank [17]을 시행하는 것에 관한 연구 또한 있었다. 그러나 하이퍼링크가 부족하기 때문에 임의 문서 상황에서는 적용될 수 없다.

Fagin 등의 분계점 알로리즘(TA) [19] 또한 상위-K 결과 계산에 사용할 수 있다. Cao et al. [20], [21]은 분산형 상황에서 TA에 최적화를 제공한다. 본 논문에서 우리는 각 색인정보 개체에 전체 벡터를 저장하며 결과를 분류할 때 합집합을 이용한다. 대신에 색인정보에 용어 가중치를 저장할 수 있으며 최종 결과 집합 계산에 TA를 이용할 수 있다. 이 방안에 대해서는 나중에 탐구할 계획이다.

P2P 시스템상의 분산 검색. 벡터 공간 모델을 이용하는 아이디어는 기존에 P2P 검색 맥락에서 적용되었다. PlanetP [22]는 VSM을 이용하는 컨텐츠 중심 검색 시스템이다. 노드는 지역적으로 벡터를 저장하지만 로컬 건텐츠의 gossip digest는 저장되지 않는다. 검색은 먼저 노드의 순위를 결정하고 그 후 상위 랭크된 노드에서 VSM을 이용한 검색을 평가함으로써 평가된다.

pSearch [23]은 VSM과 LSI를 이용하여 문서와 검색 벡터를 생성하며 이 벡터를 높은 디멘션 P2P 시스템에 나타낸다. Bhattacharya 등은 [24] DHT에 대해서 유사한 문서를 계산하기위해 유사성 보존해쉬(SPH)와 코사인 유사성을 이용한다. P2P 분산 검색 인프라 Odissea [21]는 검색 결과의 순위 결정에 TA를 이용할 것을 제안한다.

그러나 이 시스템 어느 것도 벡터를 계산하는 방법을 토론하지는 않는다. 본 논문에

제시한 연구는 문서와 검색 벡터를 생성하기 위해 이 모든 상황에 적용할 수 있다. 부울(boolean) 검색 모델을 지원하기 위한 분산형 역 색인정보를 이용하는 것에 대한 제안이 있었다.

Reynolds 등과[12] eSearch [10]은 구조화된 P2P 시스템에서 검색에 역 색인정보를 이용한다. Loo 등은 [13] 희귀한 문서의 위치를 찾아내기 위해서는 역 색인정보를 가지고 DHT를 이용하며 인기가 있는 문서의 경우 Gnutella 스타일 flooding을 이용하는 혼합 해결책을 설계하였다.

Ⅵ. 결론

본 논문에서 우리는 검색 결과의 순위 결정을 위한 분산형 알고리즘을 제시하였다. 우리의 해결책은 분산형 순위 결정은 네트워크 간접비용이 거의 없이도 실행가능하다는 점을 보여준다.

대신에 우리의 알고리즘은 용어 가중치 추정에 무작위 표본추출을 이용함으로써 그러한 벡터를 계산한다. 시뮬레이션과 공적 분석을 통해서 우리의 접근법의 검색 품질은 VSM 집중형 모델 이행에 의한 검색 품질에 견줄 수 있음을 보여준다. 또한 우리의 접근법은 문서 집합의 규모가 노드의 개수와 더불어 증가하는 상황에서 잘 확장된다는 점을 보여준다.

References

1. Salton, G., Wong, A., Yang, C.: A vector space model for information retrieval. Journal of the American Society for Information Retrieval 18(11), 613–620 (1975)
2. TREC: Text REtrieval Conference. `http://trec.nist.gov/()`
3. Dumais, S.T.: Improving the retrieval of information from external sources. Behavior Research Methods, Instruments, and Computers 23(2), 229–236 (1991)
4. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing and Management 24(5), 513–523 (1988)
5. Buckley, C.: Implementation of the SMART information retrieval system. Technical report, Dept. of Computer Science, Cornell University, Ithaca, NY, USA (1985)
6. Morselli, R., Bhattacharjee, B., Srinivasan, A., Marsh, M.A.: Efficient lookup on unstructured topologies. In: PODC 2005. Proceedings of the 24th symposium on Principles of distributed computing, New York, NY, USA, pp. 77–86 (2005)
7. Ganesan, P., Sun, Q., Garcia-Molina, H.: Yappers: A peer-to-peer lookup service over arbitrary topology. In: INFOCOM. 22nd Annual Joint Conf. of the IEEE Computer and Communications Societies, San Francisco, USA (2003)
8. King, V., Saia, J.: Choosing a random peer. In: PODC 2004. Proceedings of the 23rd symposium on Principles of distributed computing, New York, NY, USA, pp. 125–130 (2004)
9. Chernoff, H.: A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. Annals of Mathematical Statistics 23, 493–509 (1952)
10. Tang, C., Dwarakadas, S.: Hybrid global-local indexing for efficient peer-to-peer information retrieval. In: Proceedings of USENIX NSDI 2004 Conference, San Fransisco, CA (2004)
11. Gopalakrishnan, V., Bhattacharjee, B., Chawathe, S., Keleher, P.: Efficient peer-to-peer namespace searches. Technical Report CS-TR-4568, University of Maryland, College Park, MD (2004)
12. Reynolds, P., Vahdat, A.: Efficient peer-to-peer keyword searching. In: Endler, M., Schmidt, D.C. (eds.) Middleware 2003. LNCS, vol. 2672, pp. 21–40. Springer, Heidelberg (2003)
13. Loo, B.T., Hellerstein, J.M., Huebsch, R., Shenker, S., Stoica, I.: Enhancing P2P file-sharing with an internet-scale query processor. In: VLDB 2004. Thirtieth International Conference on Very Large Data Bases, Toronto, Canada, pp. 432–443 (2004)
14. Stoica, I., Morris, R., Karger, D., Kaashoek, M.F., Balakrishnan, H.: Chord: A scalable peer-to-peer lookup service for internet applications. In: Proceedings of the ACM SIGCOMM 2001, San Diego, California (2001)
15. Rowstron, A., Druschel, P.: Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In: Guerraoui, R. (ed.) Middleware 2001. LNCS, vol. 2218, pp. 329–350. Springer, Heidelberg (2001)
16. Deerwester, S., Dumais, S., Landauer, T., Furnas, G., Harshman, R.: Indexing by latent semantic analysis. Journal of the American Society for Information Science 41(6), 391–407 (1990)
17. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation algorithm: bringing order to the web. Technical report, Dept. of Computer Science, Stanford University (1999)

18. Wang, Y., DeWitt, D.J.: Computing PageRank in a distributed internet search engine system. In: VLDB 2004. Thirtieth International Conference on Very Large Data Bases, Toronto, Canada, pp. 420–431 (2004)
19. Fagin, R., Lotem, A., Naor, M.: Optimal aggregation algorithms for middleware. Journal of Computer and System Sciences (JCSS) 66(4), 614–656 (2003)
20. Cao, P., Wang, Z.: Efficient top-k query calculation in distributed networks. In: PODC 2004. Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing, pp. 206–215. ACM Press, New York (2004)
21. Michel, S., Triantafillou, P., Weikum, G.: KLEE: A framework for distributed top-k query algorithms. In: VLDB 2005. Proceedings of the 31st International Conference on Very Large Data Bases, Trondheim, Norway, pp. 637–648 (2005)
22. Cuenca-Acuna, F.M., Peery, C., Martin, R.P., Nguyen, T.D.: PlanetP: Using Gossiping to Build Content Addressable Peer-to-Peer Information Sharing Communities. In: HPDC-12. Proceedings of the 12th Symposium on High Performance Distributed Computing, IEEE Press, Los Alamitos (2003)
23. Tang, C., Xu, Z., Dwarkadas, S.: Peer-to-peer information retrieval using self-organizing semantic overlay networks. In: Proceedings of ACM SIGCOMM 2003, pp. 175–186. ACM Press, New York (2003)
24. Bhattacharya, I., Kashyap, S.R., Parthasarathy, S.: Similarity searching in peer-to-peer databases. In: ICDCS 2005. Proceedings of the 25th International Conference on Distributed Computing Systems, pp. 329–338 (2005)