오디세우스/Parallel-OOSQL: 오디세우스 정보검색용 밀결합 DBMS를 사용한 병렬 정보 검색 엔진

(Odysseus/Parallel-OOSQL: A Parallel Search Engine using the Odysseus DBMS Tightly-Coupled with IR Capability)

류 재 준 † 황 규 영 †† 이 재 길 † 권 혁 윤 †
(Jae-Joon Ryu) (Kyu-Young Whang) (Jae-Gil Lee) (Hyuk-Yoon Kwon)

김 이 른 † 허 준 석 † 이 기 훈 †
(Yi-Reun Kim) (Jun-Suk Heo) (Ki-Hoon Lee)

요 약 최근 들어 인터넷의 성장으로 인하여 문서의 양이 기하급수적으로 증가함에 따라, 대용량의 문서를 빠르게 검색 할 수 있는 병렬 정보 검색 엔진에 대한 중요성이 더욱 대두되고 있다. 병렬 정보 검색 엔진을 구현하기 위하여서는 역 색인을 분할하고, 분할된 역 색인을 통하여 병렬적으로 검색하는 것이 필요하다. 역 색인을 분할하는 기존 방법으로는 1) 문서 식별자 분할 방법과 2) 키워드 식별자 분할 방법이 있다. 그러나 각 분할 방법은 다음과 같은 단점들을 가지고 있다. 문서 식별자 분할 방법은 문서의 추가가 용이하고 처리량(throughput)이 높은 반면에 top-k 질의 처리 성능이 좋지 않다. 그리고 키워드 식별자 분할 방법은 top-k 질의 처리 성능이 좋은 반면에 문서의 추가가 어렵고 처리량이 낮다. 본 논문에서는 이러한 단점들을 해결하기 위하여 혼합 분할 방법을 제안하고 이를 정보 검색 기능과 밀결합된 DBMS인 오디세우스에 실현한 병렬 정보 검색 엔진을 설계하고 구현한다. 먼저, 제안된 병렬 정보 검색 엔진인 오디세우스/Parallel-OOSQL의 아키텍처를 설명한다. 그리고 체계적인 실험을 통하여 제안된 시스템의 유용성을 보인다. 실험 결과, 문서 식별자 분할 방법은 질의 처리 시간이 역 색인 분할의 블록의 개수에 근사적으로 역 비례함을 보였으며, 키워드 식별자 분할 방법은 top-k 질의 처리에 좋은 성능을 보였다. 본 논문에서 제안된 병렬 정보 검색 엔진은 세 가지 분할 방법을 모두 제공하기 때문에 응용 환경에 따라 분할 방법을 커스터마이즈함으로써 항상 좋은 성능을 낼 수 있다. 오디세우스/Parallel-OOSQL 병렬 정보 검색 엔진은 각 슬레이브 노드 당 1억 건의 웹 문서를, 시스템 전체로는 수십억 건의 웹 문서를 인덱스하여 저장하고 질의를 처리할 수 있다.

키워드 : 병렬, 대용량, 검색 엔진

Abstract As the amount of electronic documents increases rapidly with the growth of the Internet, a parallel search engine capable of handling a large number of documents are becoming ever important. To implement a parallel search engine, we need to partition the inverted index and search through the partitioned index in parallel. There are two methods of partitioning the inverted index: 1) document-

· 본 논문은 2002년도 한국정보과학회 춘계 학술대회에 발표된 초기버전[10]을 내용적으로 상세하게 보강 및 확장한 것임
· 본 논문은 과학기술부/한국과학재단의 국가지정연구사업(No. R0A-2007-000-20101-0) 및 우수연구센터사업(첨단정보기술연구센터)에 의해 수행된 연구임

† 학생회원 : 한국과학기술원 전산학과
jjryu@mozart.kaist.ac.kr
jglee@mozart.kaist.ac.kr
hykwon@mozart.kaist.ac.kr
yrkim@mozart.kaist.ac.kr
jsheo@mozart.kaist.ac.kr
khlee@mozart.kaist.ac.kr

†† 종신회원 : 한국과학기술원 전산학과 교수
kywhang@mozart.kaist.ac.kr

논문접수 : 2007년 12월 26일
심사완료 : 2008년 4월 25일

정보과학회논문지: 컴퓨팅의 실제 및 레터 제14권 제4호(2008.6)

identifier based partitioning and 2) keyword-identifier based partitioning. However, each method alone has the following drawbacks. The former is convenient in inserting documents and has high throughput, but has poor performance for top-k query processing. The latter has good performance for top-k query processing, but is inconvenient in inserting documents and has low throughput. In this paper, we propose a hybrid partitioning method to compensate for the drawback of each method. We design and implement a parallel search engine that supports the hybrid partitioning method using the Odysseus DBMS tightly coupled with information retrieval capability. We first introduce the architecture of the parallel search engine-Odysseus/Parallel-OOSQL. We then show the effectiveness of the proposed system through systematic experiments. The experimental results show that the query processing time of the document-identifier based partitioning method is approximately inversely proportional to the number of blocks in the partition of the inverted index. The results also show that the keyword-identifier based partitioning method has good performance in top-k query processing. The proposed parallel search engine can be optimized for performance by customizing the methods of partitioning the inverted index according to the application environment. The Odysseus/Parallel-OOSQL parallel search engine is capable of indexing, storing, and querying 100 million web documents per node or tens of billions of web documents for the entire system.

Key words : parallel, large-scale, search engine

1. 서 론

최근에 인터넷이 널리 보급되어 전자 문서로 표현된 정보의 양이 급격히 늘어남에 따라 많은 양의 문서에 대한 정보 검색을 효율적으로 지원해 주는 것이 더욱 중요해 지고 있다[1-3]. 그러나 기존의 단일 시스템으로는 빠른 속도로 증가하는 대용량의 정보에 대한 효율적인 검색을 지원에 주는 것에 한계가 있으며, 이러한 한계를 극복하기 위하여 병렬 정보 검색이 연구되어 왔다[4,5].

병렬 정보 검색은 정보 검색 시스템에서 주어진 키워드가 발생한 문서들을 찾기 위하여 사용하는 색인을 병렬적으로 검색 하는 방식으로 구현된다[6]. 여러 가지 정보 검색 색인 구조들 중 가장 유용하고 뛰어난 성능을 가진 것으로 알려져 있는 역 색인(inverted index) 구조[7,8]를 사용하는 정보 검색 시스템의 경우, 병렬 정보 검색은 역 색인을 분할하고 분할된 각 블록(block)을 병렬적으로 검색함으로써 가능하다.

병렬 정보 검색을 구현하기 위하여 역 색인을 분할하는 방법은 크게 문서 식별자 분할 방법과 키워드 식별자 분할 방법으로 나누어진다[6]. 문서 식별자 분할 방법은 같은 문서에 대한 역 색인의 내용이 같은 블록에 존재하도록 역 색인을 분할하는 방법인 반면, 키워드 식별자 분할 방법은 같은 키워드에 대한 역 색인의 내용이 같은 블록에 존재하도록 역 색인을 분할하는 방법이다. 그러므로 문서 식별자 분할 방법에서의 키워드 검색은 모든 역 색인 블록에서 해당 키워드를 검색하여 결과들을 병합함으로써 수행할 수 있으며, 키워드 식별자 분할 방법에서의 키워드 검색은 검색하고자 하는 키워드를 색인하는 역 색인 블록 만을 검색함으로써 수행할 수 있다.

각 분할 방법에서의 키워드 검색 방법으로 인해 문서 식별자 분할 방법을 사용하는 경우 디스크 입/출력 횟수가 많아지는 단점이 있으며, 키워드 식별자 분할 방법을 사용하는 경우 질의 처리 시간이 검색하고자 하는 키워드들 중 가장 큰 포스팅을 가지는 키워드의 검색 시간에 의해 결정되고, 각 키워드가 색인되는 역 색인 블록이 정해져 있어 잦은 문서의 추가 및 삭제가 일어나는 경우 각 역 색인 블록이 가지는 포스팅 수의 불균형이 심해져 시스템 성능 저하의 원인이 되므로 역 색인 블록들이 재구성(reorganization)[6] 되어야 한다는 단점이 있다. 현재까지의 대부분의 병렬 정보 검색 시스템은 역 색인 분할 방법으로서 문서 식별자 분할 방법 또는 키워드 식별자 분할 방법만을 사용하므로 이와 같은 단점들이 그대로 나타나고 있다.

본 논문에서는 한국과학기술원 데이타베이스 및 멀티미디어 연구실에서 개발한 정보 검색 기능이 밀결합된 객체 관계형 데이타베이스 관리 시스템인 오디세우스[9]를 사용하여 병렬 정보 검색 시스템을 설계하고 구현한다. 먼저, 병렬 정보 검색 시스템의 아키텍쳐를 설명한다. 다음으로, 문서 식별자 분할 방법과 키워드 식별자 분할 방법을 분석하고, 각 분할 방법의 단점들을 보완하기 위하여 역 색인 분할 방법들을 혼합 사용하는 방법을 제안한다. 마지막으로, 구현된 병렬 정보 검색의 유용성을 보이기 위하여 단일 정보 검색 시스템과의 성능 평가를 수행한다. 실험 결과, 문서 식별자 기반 분할 방법은 질의 처리 성능이 역 색인 블록의 개수에 근사하게 비례하여 향상됨을 보인다[10]. 그리고, 키워드 식별자 분할 방법은 top-k 질의 처리에서 좋은 성능을 보인다. 본 논문에서 제안된 병렬 정보 검색 시스템은 세 가

지 분할 방법을 모두 제공하기 때문에 응응 환경에 따라 분할 방법을 커스터마이즈함으로써 항상 좋은 성능을 낼 수 있다.

본 논문의 구성은 다음과 같다. 제2절에서는 연구되어 온 정보 검색의 병렬화 방법에 대해서 설명하고, 구현 사례를 소개한다. 제3절에서는 오디세우스 데이타베이스 관리 시스템의 정보 검색 기능에 대해서 설명한다. 제4절에서는 오디세우스를 이용한 병렬 정보 검색 시스템의 설계 및 구현 방법에 대해서 설명한다. 제5절에서는 실험을 통하여 구현된 병렬 정보 검색 시스템의 성능을 측정하고, 결과를 분석한다. 마지막으로 제6절에서는 결론을 내린다.

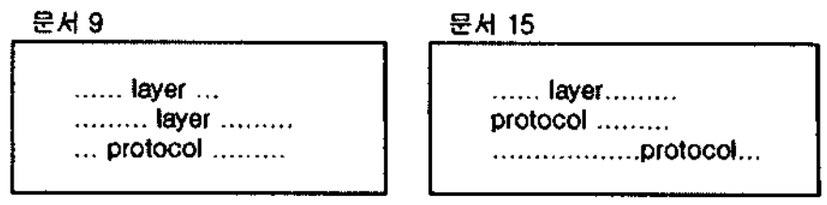
2. 관련 연구

본 절에서는 관련 연구로서 제2.1절에서는 정보 검색 시스템에서 가장 널리 쓰이는 역 색인(inverted index) 구조[13]에 대해서 설명한다. 제2.2절에서는 역 색인 구조를 분할하여 정보 검색을 병렬화하는 방법에 대해서 설명하고, 제2.3절에서는 병렬 정보 검색 시스템의 구축 사례를 알아본다.

2.1 역 색인 구조

정보 검색 시스템은 주어진 키워드를 포함하는 문서들을 찾기 위하여서 색인을 사용한다[1,11]. 정보 검색을 위하여 사용되는 색인 구조로는 역 색인, 시그니쳐 파일(signature file), 서픽스 어레이(suffix array)등이 있으며[12], 여러 색인 구조들 중에서 역 색인 구조가 가장 유용하고 뛰어난 성능을 가짐이 알려져 있다. 이로 인해 대부분의 정보 검색 시스템은 역 색인 구조를 사용한다[13]. 본 논문에서 병렬 정보 검색 시스템을 구현한 기반 시스템인 오디세우스도 역시 정보 검색에 더욱 효율적으로 변형된 역 색인 구조를 사용한다[14].

역 색인은 각 문서에 나타난 키워드와 키워드가 나타는 위치에 대한 정보를 가지며, 크게 사전 파일(dictionary file)과 포스팅 파일(posting file)로 구성된다[11]. 사전 파일은 문서에 나타난 키워드, 키워드의 식별자(keyword identifier), 키워드의 포스팅 개수, 포스팅 파일로의 포인터 등을 유지하며, 포스팅 파일은 해당 키워드가 발생한 정보를 포스팅 리스트로 유지한다. 포스팅 리스트란 포스팅들의 리스트를 말하며, 포스팅은 키워드가 발생한 문서의 식별자(docID: document identifier)와 그 문서 내에서의 발생 위치 정보를 유지한다. 예를 들어, 그림 1의 (a)에서와 같은 문서 9와 문서 15에 대해 역 색인을 구축하는 경우 (b)와 같은 역 색인을 얻는다. (b)의 역 색인에서 "layer"라는 키워드가 문서 9의 2번째와 10번째, 그리고 문서 15의 4번째 단어로서 나타난다는 것을 알 수 있다.



(a) 예제 문서

사전 파일 / 포스팅 파일

키워드	키워드 식별자	포스팅 개수	포스팅 리스트 (DocID, Position)
layer	10	3	(9, 2), (9, 10), (15, 4)
protocol	45	3	(9, 8), (15, 12), (15, 30)

(b) 예제 문서의 역 색인

그림 1 역 색인 예제

역 색인 구조에서 특정 키워드를 검색하는 것은 사전 파일에서 해당 키워드를 검색하여 포스팅 파일에서 포스팅 리스트를 얻어 오는 방식으로 수행된다. 예를 들어, 그림 1(b)의 역 색인 예제에서 "layer"라는 키워드를 포함하는 문서를 검색 하는 경우, 사전 파일에서 "layer"라는 키워드를 찾고 포스팅 파일에서 포스팅 리스트를 읽어서 문서 9와 문서 15가 "layer"라는 키워드를 포함하고 있음을 알게 된다.

2.2 병렬 정보 검색 시스템을 위한 역 색인 분할 방법

많은 양의 문서에 대한 효율적인 검색을 하기 위하여 병렬 정보 검색 시스템을 구축하는 것은 정보 검색 시스템에서의 색인에 대한 검색을 병렬적으로 수행함으로써 가능하다. 즉, 주어진 키워드가 발생한 문서를 찾기 위하여 검색해야 하는 색인을 분할하고, 각 블록들에 대해 병렬적으로 검색을 수행함으로써 검색하는데 필요한 시간을 줄인다. 역 색인을 사용하는 정보 검색 시스템의 경우, 역 색인을 분할하여 각 블록들에 대해 병렬적으로 검색을 수행함으로써 병렬 정보 검색을 할 수 있다.

정보 검색을 병렬화하여 수행하기 위하여 역 색인을 분할하는 방법에는 크게 문서 식별자에 대해 역 색인을 분할하는 문서 식별자 분할 방법[6,15]과 키워드 식별자에 대해 역 색인을 분할하는 키워드 식별자 분할 방법[6,15]이 있다. 본 절에서는 이러한 역 색인의 분할 방법을 소개하고 각 방법의 장단점을 비교한다. 제 2.2.1절에서는 문서 식별자를 이용한 역 색인 분할 방법을 설명하고, 제 2.2.2절에서는 키워드 식별자를 이용한 역 색인 분할 방법을 설명한다. 제 2.2.3절에서는 문서 식별자 분할 방법과 키워드 식별자 분할 방법의 장단점을 비교한다.

2.2.1 문서 식별자 분할 방법

문서 식별자를 이용한 역 색인의 분할 방법은 같은 문서 식별자에 대한 포스팅 리스트는 같은 역 색인 블록에 포함 되도록 분할하는 방법이다[6,15]. 그러므로 하나의 키워드에 대한 포스팅 리스트는 여러 역 색인 블록에 있을 수 있다. 예를 들어, 그림 1(b)의 역 색인을

역 색인 블록 1

키워드	키워드 식별자	포스팅 개수	포스팅 리스트 (DocID, Position)
layer	10	2	(9, 2), (9, 10)
protocol	45	1	(9, 8)

역 색인 블록 2

키워드	키워드 식별자	포스팅 개수	포스팅 리스트 (DocID, Position)
layer	10	1	(15, 4)
protocol	45	2	(15, 12), (15, 30)

그림 2 문서 식별자 분할 방법을 사용한 역 색인 분할의 예

문서 식별자에 따라 분할하는 경우 그림 2와 같이 문서 9에 대한 포스팅 리스트는 역 색인 블록 1에만 포함되고 문서 15에 대한 포스팅 리스트는 역 색인 블록 2에만 포함된다.

역 색인을 문서 식별자에 따라 분할하였을 경우, 특정 키워드를 가지는 문서의 검색은 각 역 색인 블록에서 요구된 키워드를 검색하여 얻어진 결과들을 병합함으로써 수행된다. 예를 들어, 그림 2의 역 색인 분할 예에서 "layer"라는 키워드를 포함하는 문서를 검색하고자 하는 경우 역 색인 블록 1과 역 색인 블록 2를 병렬적으로 검색하여 각각 문서 9와 문서 15가 "layer"라는 키워드를 포함하고 있음을 결과로서 얻게 되고 각 결과를 병합하여 검색 하고자 하는 최종 결과를 얻게 된다.

2.2.2 키워드 식별자 분할 방법

키워드 식별자를 이용한 역 색인의 분할 방법은 같은 키워드 식별자에 대한 포스팅 리스트는 같은 역 색인 블록에 포함되도록 분할하는 방법이다[6,15]. 그러므로 한 문서에 대한 포스팅 리스트는 여러 역 색인 블록에 있을 수 있다. 예를 들어, 그림 1(b)의 역 색인을 키워드 식별자에 따라 분할하는 경우 그림 3과 같이 키워드 "layer"에 대한 포스팅 리스트는 역 색인 블록 1에만 포

역 색인 블록 1

키워드	키워드 식별자	포스팅 개수	포스팅 리스트 (DocID, Position)
layer	10	3	(9, 2), (9, 10), (15, 4)

역 색인 블록 2

키워드	키워드 식별자	포스팅 개수	포스팅 리스트 (DocID, Position)
protocol	45	3	(9, 8), (15, 12), (15,30)

그림 3 키워드 식별자 분할 방법을 사용한 역 색인 분할의 예

함되고, 키워드 "protocol"에 대한 포스팅 리스트들은 역 색인 블록 2에만 포함된다.

역 색인을 키워드 식별자에 따라 분할하였을 경우, 특정 키워드를 가지는 문서의 검색은 특정 키워드에 대한 포스팅을 가지는 역 색인 블록만을 검색함으로써 수행될 수 있다. 예를 들어, 그림 3의 역 색인 분할 예에서 "layer"라는 키워드를 포함하는 문서를 검색하고자 하는 경우 역 색인 블록 1만을 검색하여 문서 9와 문서 15가 "layer"라는 키워드를 포함하고 있음을 알게 된다.

2.2.3 문서 식별자와 키워드 식별자 분할 방법의 비교

두 가지 역 색인 분할 방법인 문서 식별자 분할 방법과 키워드 식별자 분할 방법은 각자 장단점을 가지고 있다. 각 분할 방법에 대해 특정 키워드에 대한 포스팅 리스트의 위치, 필요한 색인의 종류, 디스크 입/출력 병렬성, 문서 추가의 용이성, 처리량(throughput), 그리고 top-k 질의 처리 성능의 측면에서 두 방법의 장단점 및 특징을 설명한다[6,16]. 표 1은 각 분할 방법의 장단점 및 특징들을 정리한 것이다.

특정 키워드에 대한 포스팅 리스트는 문서 식별자 분할 방법의 경우 모든 역 색인 블록에 존재할 수 있고, 키워드 식별자 분할 방법의 경우 하나의 역 색인 블록에만 존재하게 된다. 그러므로 특정 키워드에 대한 검색은 문서 식별자 분할 방법의 경우 모든 역 색인 블록에

표 1 문서 식별자 분할 방법과 키워드 식별자 분할 방법의 비교

항목	문서 식별자 분할 방법	키워드 식별자 분할 방법
한 키워드에 대한 포스팅 리스트의 위치	여러 역 색인 블록	하나의 역 색인 블록
필요한 색인의 종류	지역 색인 (local indexing)	지역 색인 (local indexing), 전역 색인 (global indexing)
한 질의에서의 입/출력 병렬성	시스템을 구성하는 디스크의 개수에 비례	한 질의에서의 키워드의 개수에 비례
포스팅 파일 입/출력 횟수	키워드 식별자 분할 방법과 같거나 많음	문서 식별자 분할 방법과 같거나 적음
문서 추가의 용이성	쉬움	어려움
처리량 (throughput)	높음	낮음
Top-k 질의 처리 성능	나쁨	좋음

서 행해지게 되고, 키워드 식별자 분할 방법의 경우 하나의 역 색인 블록에서만 행해지게 된다. 키워드 식별자 분할 방법을 사용하는 경우, 키워드를 검색하기 위하여 일부의 디스크만을 사용하므로 검색을 위하여 모든 디스크가 사용되어야 하는 문서 식별자 방법을 사용하는 경우 보다 동시에 더 많은 질의를 처리할 수 있게 된다.

필요한 색인의 종류에 대해서는 문서 식별자 분할 방법에서는 각 역 색인 블록, 즉, 지역 색인(local index)만 필요하지만 키워드 식별자 분할 방법에서는 지역 색인 뿐 아니라 어느 키워드가 어느 역 색인 블록에 있는지를 나타내는 전역 색인(global index)이 필요하다.

최대 디스크 입/출력 병렬성 측면에서는 문서 식별자 분할 방법의 경우, 각 역 색인 블록을 병렬적으로 검색할 수 있으므로 시스템을 구성하는 디스크의 수에 비례한다. 그러므로 키워드를 검색하는 질의의 처리 시간은 해당 키워드에 대한 포스팅을 가장 많이 가지는 역 색인 블록을 검색하는 시간에 의해 결정된다. 키워드 식별자 분할 방법의 경우, 서로 다른 역 색인 블록에 색인되어 있는 여러 키워드를 병렬적으로 검색할 수 있으므로 검색 하고자 하는 키워드의 수에 비례한다. 그러므로 단일 키워드를 검색하는 질의에 대해서는 역 색인을 병렬적으로 검색하는 효과를 얻을 수 없으며, 여러 키워드를 검색 하는 질의에 대해서도 가장 많은 포스팅을 가지는 키워드를 검색하는 시간에 의해 질의 처리 시간이 결정된다. 전체 포스팅 파일 입/출력 횟수는 문서 식별자 분할 방법의 경우, 모든 블록을 검색하여야 하므로 하나의 역 색인 블록을 검색하는 키워드 식별자 분할 방법의 경우보다 같거나 많게 된다.

문서 추가의 용이성 측면에서 문서 식별자 분할 방법을 사용하는 경우, 추가하고자 하는 문서들에 대하여 별도의 역 색인을 구축한 후, 구축된 역 색인에 대한 정보를 병렬 정보 검색 시스템에 추가해 줌으로써 문서들을 쉽게 병렬 정보 검색 시스템에 추가할 수 있다. 이에 비해 키워드 식별자 분할 방법을 사용하는 경우, 새로 추가되는 문서의 포스팅들은 이미 정해져 있는 키워드 분할 정보에 따라 각 역 색인 블록에 분배되므로 문서의 추가, 삭제가 빈번히 일어나게 되면 각 역 색인 블록이 가지는 포스팅 수의 불균형이 심해져 시스템 성능 저하의 원인이 되고, 이를 해결하기 위하여서는 역 색인 블록들이 전체적으로 재구성(reorganization)되어야만 하므로 문서의 추가가 쉽지 않다.

처리량(throughput)의 측면에서는 일반적으로 문서 식별자 분할 방법이 키워드 식별자 분할 방법에 비해 높다고 알려져 있다[16]. 문서 식별자 분할 방법의 경우, 모든 역 색인 블록에서 질의가 균등하게 처리되는데 반하여 키워드 식별자 분할 방법의 경우, 질의에 사용된 키워드가 포함된 역 색인 블록에서만 질의가 처리되기 때문이다.

Top-k 질의 처리 성능의 측면에서는 키워드 식별자 분할 방법이 문서 식별자 분할 방법에 비해 성능이 더 좋다. Top-k 질의는 조건을 만족하는 문서 중 중요도가 가장 높은 k개의 결과를 구하는 질의로서, 정보의 양이 방대하여 조건을 만족하는 결과 건수가 많은 경우에 매우 유용하게 사용된다[17-19]. Top-k 질의 처리시, 키워드 식별자 분할 방법은 top-k 질의에 사용된 키워드가 포함된 역 색인 블록에서만 상위 k개씩의 결과를 가져오는데 반하여, 문서 식별자 분할 방법은 모든 역 색인 블록으로부터 k개씩의 결과를 가져오기 때문이다.

2.3 구현 사례 소개

• PLIERS[6]

PLIERS(Parallel Information Retrieval System using MPI)는 메시지 전달 방법(Message Passing Mechanism)의 한 종류인 MPI(Message Passing Interface)를 사용하여 무공유(shared nothing)구조로 병렬 정보 검색 시스템을 구현하였다. 역 색인 분할 방법으로서 문서 식별자 분할 방법과 키워드 식별자 분할 방법을 개별적으로 지원할 수 있도록 구현되었다. 문서 식별자 분할 방법과 키워드 식별자 분할 방법에 대한 개별적인 지원으로 인해 제 2.2.3절에서 설명한 각 분할 방법의 단점들이 그대로 나타난다.

• ORACLE[20]

Oracle을 사용한 정보 검색은 ConText라는 정보 검색을 위한 카트리지를 Oracle 데이타베이스에 추가함으로써 가능하다[20]. 병렬 정보 검색은 Oracle 데이타베이스 시스템을 병렬화하는데 사용되는 Oracle Parallel Server를 사용함으로써 구축할 수 있다. Oracle에서는 테이블 분할(table partition)이라는 개념을 사용하여 문서 식별자 분할 방법을 이용한 병렬 정보 검색 시스템을 구축할 수 있다. 기본적으로 문서 식별자 분할 방법을 지원하므로 이로 인한 단점들이 제 2.2.3절에서 설명한 것과 같이 마찬가지로 발생한다.

• Google[21]

Google은 대표적인 대형 웹 검색시스템으로 현재 약 80억 건의 웹 페이지를 저장하고 있다[22]. Google은 이러한 대규모 웹 페이지에 대한 검색 성능을 높이기 위하여 병렬 정보 검색을 사용하는 것을 밝히고 있지만, 그 이상의 세부적인 사항에 대해서는 밝히고 있지 않다[23].

3. 오디세우스의 소개

본 절에서는 병렬 정보 검색을 구현한 기반 시스템인 오디세우스[9]에 대해 간단히 설명한다. 제 3.1절에서는

오디세우스의 정보 검색 기능에 대해 설명하고, 제 3.2 절에서는 오디세우스에서의 정보 검색 질의에 대해 설명한다.

3.1 정보 검색 기능

오디세우스는 한국과학기술원 데이타베이스 및 멀티미디어 연구실에서 개발한 정보 검색이 밀결합된 객체관계형 데이타베이스 관리 시스템이다[9]. 데이타베이스 관리 시스템(DBMS)과 정보 검색의 밀결합이란 데이타베이스 관리 시스템을 확장하여 시스템의 엔진 수준에서 정보 검색 기능을 가지도록 하는 것이다. 오디세우스는 정보 검색 기능이 밀결합되어 있으므로 데이타베이스 테이블의 특정 컬럼에 대해 역 색인을 자동으로 생성해주는 텍스트(text)타입을 제공하며, 이러한 텍스트 컬럼을 일반적인 컬럼과 동일하게 사용할 수 있다.

또한 오디세우스는 정보 검색을 위하여 많은 양의 문서를 저장하려는 경우, 배치(batch)작업을 통해 빠르게 데이타베이스를 구축할 수 있는 벌크 로딩(bulkloading) 기능을 제공한다[24]. 오디세우스의 벌크 로딩을 통해 정보 검색 시스템을 위한 데이타베이스를 구축하는 과정은 그림 4와 같다. 오디세우스의 벌크 로딩에서는 정보 검색을 위하여 저장하고자 하는 문서들로부터 키워드를 추출하여 키워드 포스팅 리스트를 구한 후, 효율적인 정보 검색 시스템의 구축을 위하여 키워드 포스팅들을 정렬한다. 문서의 내용을 데이타베이스에 저장한 후 역 색인을 구축함으로서 오디세우스의 벌크 로딩 기능을 이용하여 정보 검색 시스템을 구축하게 된다.

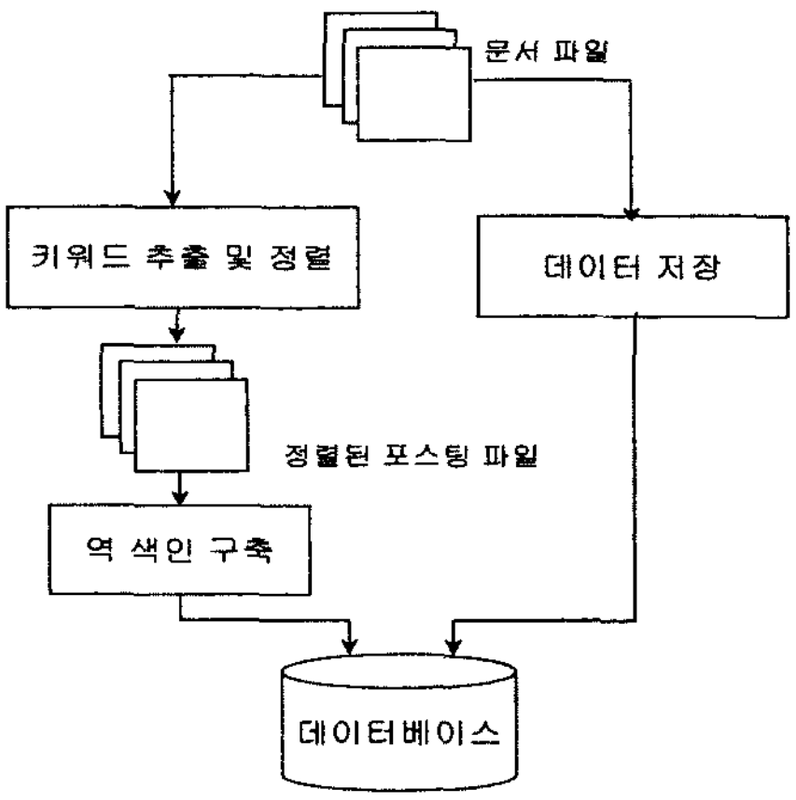


그림 4 오디세우스의 벌크로딩 기능을 이용한 정보 검색 시스템 구축 과정

3.2 정보 검색 질의

오디세우스는 SQL에 객체 지향 개념과 정보 검색 기능을 추가한 OOSQL을 통하여 정보 검색 시스템에서의 문서 검색 뿐 아니라 추가, 삭제를 위한 기능들을 제공한다[9]. OOSQL에서는 SELECT질의와 match라는 내장(built-in)함수를 통하여 문서 검색 기능을 제공한다[9]. match함수에 검색하고자 컬럼의 이름과 키워드들을 명시해 줌으로서 검색 결과로서 얻고자 하는 문서를 지정하도록 한다[9]. 예를 들어, OOSQL을 이용하여 그림 5(a)와 같은 "PaperAbstract"라는 테이블의 "abstract" 컬럼에 대하여 "layer"와 "protocol"라는 키워드를 가지는 문서의 제목을 검색하기 위하여 그림 5(b)와 같은 질의를 사용할 수 있다.

```
table PaperAbstract {
  title text,
  author varchar(20),
  abstract text
};
```

(a) 예제 스키마

```
SELECT title
FROM PaperAbstract
WHERE MATCH( abstract, "layer" & "protocol" ) > 0 ;
```

(b) 예제 검색 질의

그림 5 오디세우스에서의 정보 검색 질의의 예

OOSQL에서는 INSERT, DELETE, UPDATE 질의를 통하여 정보 검색 시스템으로의 문서 추가, 삭제, 변경을 위한 기능도 제공하고 있으나, 본 논문에서는 검색 기능에 대해서만 고려하도록 한다.

4. 병렬 정보 검색 시스템의 설계 및 구현

본 절에서는 오디세우스를 이용한 병렬 정보 검색 시스템의 설계 및 구현에 대해서 설명한다. 제 4.1절에서는 구현된 병렬 정보 검색 시스템의 아키텍처를 설명한다. 제 4.2절에서는 역 색인을 문서 식별자 분할, 키워드 식별자 분할, 그리고 문서 식별자와 키워드 식별자의 혼합(hybrid) 분할을 사용하여 병렬 정보 검색 시스템을 구축하는 방법에 대해서 설명한다. 제 4.3절에서는 정보 검색 질의를 위한 병렬 질의 처리 과정을 설명한다.

4.1 시스템 아키텍쳐

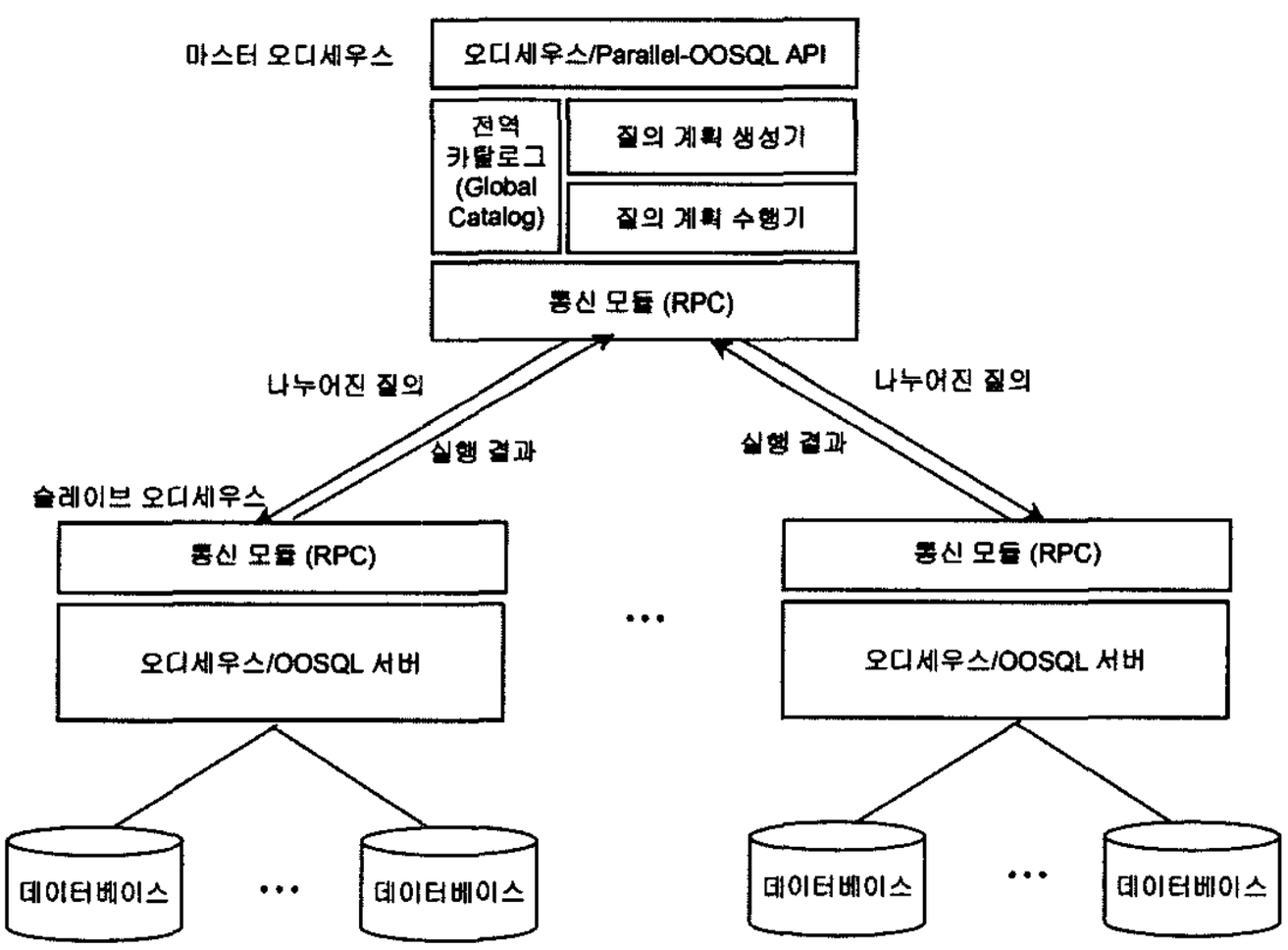
병렬 정보 검색 시스템은 역 색인을 분할하고 각 블록을 병렬적으로 검색함으로써 구현되므로, 일반적으로 사용자로부터의 질의를 분석하여 검색해야 하는 역 색인 블록을 결정하는 부분과 실제로 각 블록을 검색하는 부분으로 구성된다. 사용자로부터의 질의를 분석하는 부분에서는 역 색인의 분할 정보와 검색하고자 하는 키워드에 따라서 검색해야 하는 블록들을 선택하고 최종 결

과를 얻기 위하여 수행되어야 하는 연산들을 결정하게 되며, 각 블록을 검색하는 부분에서는 기존의 정보 검색 시스템과 같이 주어진 키워드에 대해 역 색인 블록을 검색하게 된다.

본 논문을 통하여 구현된 병렬 정보 검색 시스템은 그림 6과 같이 사용자로부터의 질의를 분석하는 부분인 마스터(master) 오디세우스와 실제로 각 블록을 검색하는 부분인 슬레이브(slave) 오디세우스로 나누어진다. 마스터 오디세우스는 사용자로부터 받은 질의를 분석하여 검색을 해야 하는 역 색인 블록을 결정하며, 각 슬레이브 오디세우스가 처리해야 하는 질의를 전달하여 해당 슬레이브 오디세우스가 역 색인 블록을 검색하도록 한다. 슬레이브 오디세우스는 마스터 오디세우스로부터 받을 질의를 처리함으로써 할당된 역 색인 블록을 검색하고 검색 결과로서 처리 결과를 마스터 오디세우스에게 돌려준다. 마스터 오디세우스와 슬레이브 오디세우스는 슬레이브 오디세우스가 처리해야 하는 질의와 슬레이브 오디세우스의 질의 실행 결과를 마스터 오디세우스에게 전달하기 위하여 RPC(Remote Procedure Call)를 사용하여 통신을 한다.

마스터 오디세우스의 구조를 자세히 살펴보면 그림 6에서와 같이 오디세우스/Parallel-OOSQL API, 전역 카탈로그(global catalog), 질의 계획 생성기, 질의 계획 실행기, 통신 모듈로서 구성된다. 오디세우스/Parallel-OOSQL API는 병렬 정보 검색 기능을 사용하여 프로그래밍을 할 수 있도록 인터페이스를 제공해 주는 부분이며, 전역 카탈로그는 사용자로부터 받은 질의를 분석하기 위하여 역 색인 분할 정보, 병렬 정보 검색 시스템을 이루는 슬레이브 오디세우스들에 대한 정보 등을 가지는 부분이다. 질의 계획 생성기는 사용자로부터 받은 병렬 검색 질의를 수행하기 위한 실행 계획을 생성하는 부분이다. 실행 계획은 슬레이브 오디세우스에 질의를 전달하는 것, 슬레이브 오디세우스로부터 질의 실행 결과를 받아 후처리를 하는 계획 등을 포함한다. 후처리란 단일 정보 검색 시스템에서와 동일한 결과를 얻기 위하여 마스터 오디세우스가 슬레이브 오디세우스들로부터 받은 결과를 가지고 수행하게 되는 연산을 말한다. 예를 들어, 문서 식별자 분할 방법을 사용한 경우 최종 결과를 얻기 위하여 각 슬레이브 오디세우스들로부터의 결과를 병합하는 것 등을 말한다. 통신 모듈은 RPC를 사용하기 위한 부분으로서 슬레이브 오디세우스와의 통신을 위하여 사용된다.

슬레이브 오디세우스를 자세히 살펴보면 그림6에서와 같이 오디세우스/OOSQL 서버, 통신 모듈로서 구성된다. 오디세우스/OOSQL 서버 부분은 기존의 오디세우스 데이타베이스 관리 시스템에 해당되는 부분으로서 할당된 역 색인 블록들을 저장하고 있으며 마스터 오디세우스로부터 받은 질의를 처리함으로써 역 색인 블록을 검색하는 부분이다. 일반적으로, 하나의 슬레이브 오디세우스에 여러 역 색인 블록이 저장되도록 할 수 있지만 본 논문에서는 하나의 슬레이브 오디세우스에 하나의 역 색인 블록만을 저장함을 가정하도록 한다. 통신 모듈



마스터 오디세우스 : 병렬 문서 검색을 위해 질의 계획을 생성, 수행하는 부분
슬레이브 오디세우스 : 기존의 오디세우스 서버로서 나누어진 질의를 처리하는 부분

그림 6 오디세우스를 이용한 병렬 정보 검색 시스템의 시스템 아키텍쳐

은 RPC를 사용하기 위한 부분으로서 마스터 오디세우스와의 통신을 위하여 사용된다.

본 논문을 통하여 구현된 병렬 정보 검색 시스템은 AND, OR, NOT, 그리고 절단(*)연산자를 지원한다. AND, OR, NOT은 2개의 피연산자를 갖는 연산자로서, 관계 대수(relational algebra)에서의 집합 연산자인 교집합(intersection), 합집합(union), 차집합(difference)에 각각 해당한다[25]. AND 연산자는 왼쪽 피연산자와 오른쪽 피연산자 모두를 포함하는 문서를 검색하고, OR 연산자는 왼쪽 피연산자와 오른쪽 피연산자 중 하나를 포함하는 문서를 검색한다. NOT 연산자는 왼쪽 피연산자는 포함하지만 오른쪽 피연산자는 포함하지 않는 문서를 검색한다. 마지막으로, 절단 연산자는 연산자 앞의 단어(예: 한국*)나 뒤의 단어(예: *연구)를 포함하는 문서를 검색한다.

4.2 역 색인의 분할

본 절에서는 역 색인을 분할하여 병렬 정보 검색 시스템을 구축하는 방법에 대해서 설명한다. 제 4.2.1절에서는 문서 식별자 분할 방법을 사용하여 병렬 정보 검색 시스템을 구축하는 방법에 대해 설명하고, 제 4.2.2절에서는 키워드 식별자 분할 방법을 사용하여 병렬 정보 검색 시스템을 구축하는 방법에 대해서 설명한다. 제 4.2.3절에서는 문서 식별자 분할 방법과 키워드 식별자 분할 방법을 혼합 사용하여 병렬 정보 검색 시스템을 구축하는 방법에 대해서 설명한다.

4.2.1 문서 식별자 분할

문서 식별자 분할 방법을 사용하여 병렬 정보 검색 시스템을 구현하기 위하여서는 특정 문서의 포스팅 리스트들이 하나의 역 색인 블록에서만 색인되도록 하는 것이 필요하다. 본 논문에서는 이를 위하여 병렬 정보 검색 시스템에 저장하고자 하는 문서들을 여러 블록으로 나누고 각 문서 블록에 대해 역 색인을 구축한다. 문서 식별자 분할 방법을 사용하여 병렬 정보 검색 시스템을 구축하는 상세한 과정은 그림 7과 같다. 병렬 정보 검색 시스템에 저장하고자 하는 문서들로부터 전역 카탈로그를 참조하여 문서 블록들을 만들고 각 슬레이브 오디세우스들에게 분배하여, 분배된 문서에 대하여 정보 검색 시스템 구축 과정을 거쳐 역 색인을 구축하도록 한다.

본 논문에서는 각 문서로의 접근 정도가 동일하다고 가정하여 문서들을 분할하는 방법으로 각 문서 블록에 같은 개수의 문서를 할당하는 방법을 사용한다. 이외에도 문서들을 여러 블록들로 나누는 방법으로써 각 문서의 접근 정도(access frequency)등을 고려하거나 각 문서 블록이 포함하는 문서에 나타나는 키워드들의 출연 개수와 접근 정도 등을 고려하여 각 문서 블록과 관련

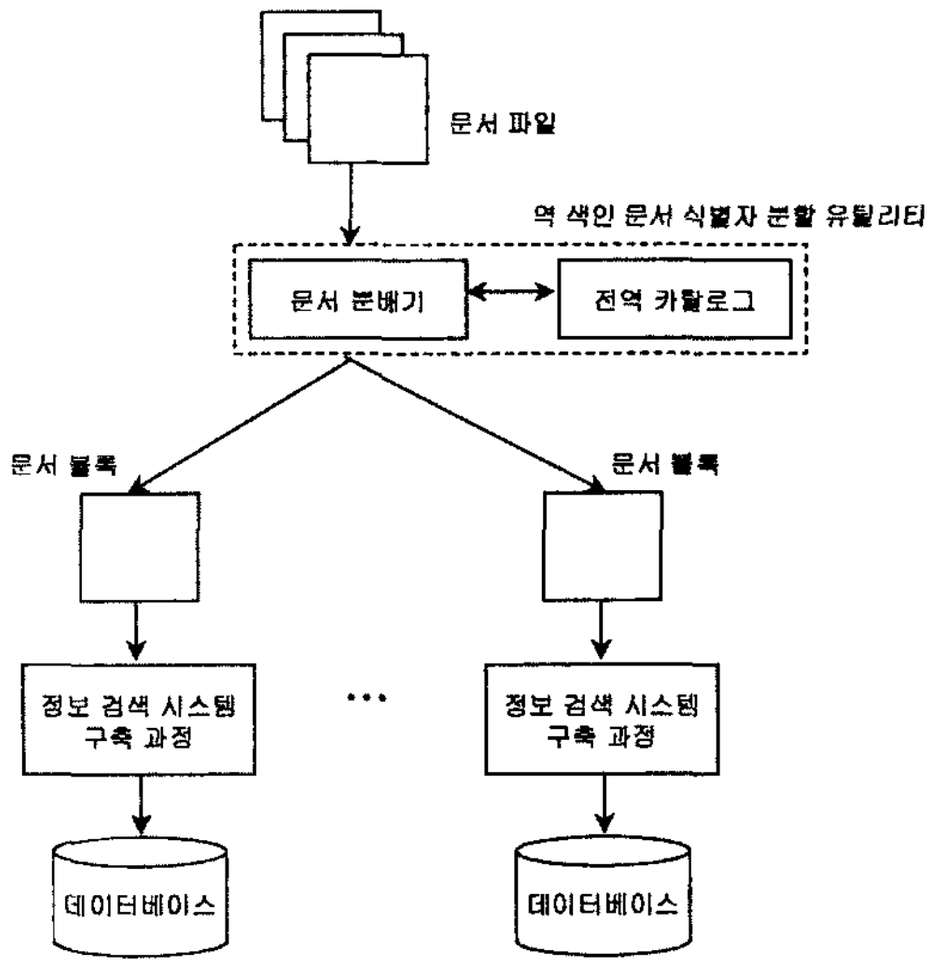


그림 7 역 색인의 문서 식별자 분할을 이용한 병렬 정보 검색 시스템 구축 과정.

되는 역 색인 블록으로의 접근 정도가 같도록 하는 방법 등이 연구되어 왔다[15,26].

4.2.2 키워드 식별자 분할

키워드 식별자 분할 방법을 사용하여 병렬 정보 검색 시스템을 구현하기 위하여서는 특정 키워드의 포스팅 리스트들이 하나의 역 색인 블록에서만 색인되도록 하는 것이 필요하다. 본 논문에서는 이를 위하여 포스팅 리스트들을 키워드에 따라 분할하고, 각 블록에 대해 역 색인을 구축한다. 키워드 식별자 분할 방법을 사용하여 병렬 정보 검색 시스템을 구축하는 상세한 과정은 그림 8과 같다. 병렬 정보 검색 시스템에 저장하고자 하는 문서들의 포스팅 리스트들을 전역 카탈로그를 참조하여 키워드에 따라 분할하고 각 슬레이브 오디세우스들에게 분배하여, 분배된 포스팅 리스트들에 대하여 정보 검색 시스템 구축 과정을 거쳐 역 색인을 구축하도록 한다.

본 논문에서는 키워드 식별자 분할 방법을 사용할 때 요구되는 전역 색인을 위한 구조를 단순화 하고, 포스팅 리스트 분배 과정의 복잡도를 줄이기 위하여 포스팅들을 분할하는 방법으로 각 역 색인 블록에 근사적으로 같은 개수의 포스팅들을 할당하는 방법을 사용한다. 이외에도 키워드 포스팅들을 분할하는 방법으로서 각 키워드의 포스팅 개수 또는 접근 정도 등을 고려하여 각 역 색인 블록으로의 접근 정도가 같도록 하는 여러 가지 분할 방법이 연구되어 왔다[15].

키워드 식별자 분할 방법에서는 문서들이 여러 슬레이브 오디세우스에 중복되어 저장된다. 슬레이브 오디세우스는 질의 처리를 위하여 해당 역 색인 블록이 색인하고 있는 키워드들을 포함하는 문서들을 데이타베이스

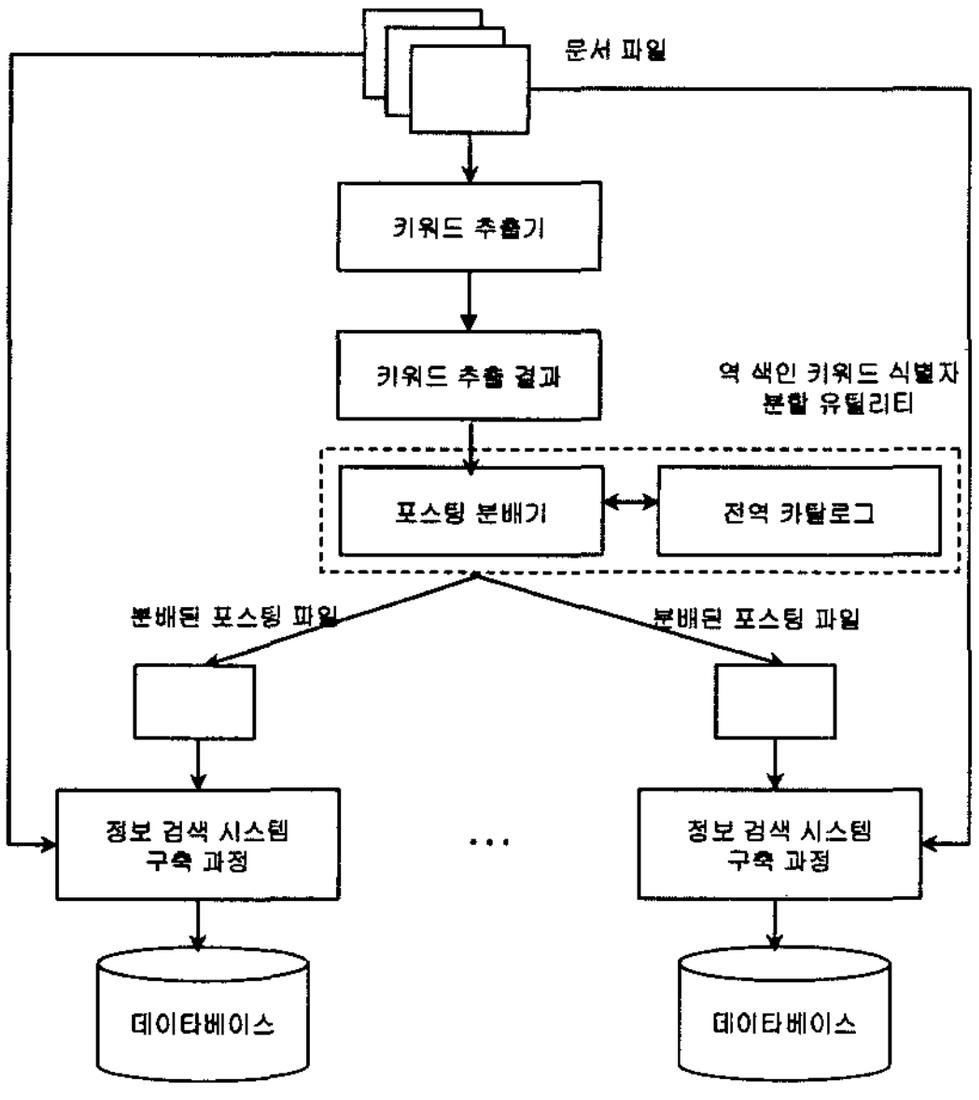


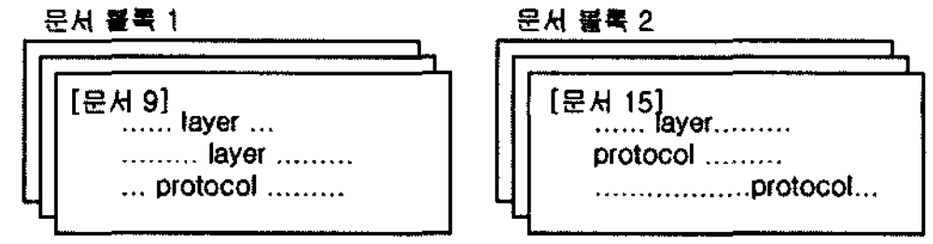
그림 8 키워드 식별자 분할 방법을 이용한 병렬 정보 검색 시스템 구축 과정

에 저장하고 있어야 한다. 그런데, 대부분의 경우 문서들이 각 역 색인 블록이 색인하고 있는 키워드들을 모두 가지고 있기 때문에 문서들이 모든 슬레이브 오디세우스에 중복 저장될 가능성이 높다. 따라서, 본 논문에서는 키워드 식별자 분할 방법을 사용하여 병렬 정보 검색 시스템을 구축할 경우 역 색인 블록들이 색인하고 있는 키워드들을 포함하는 문서들만을 추출하는 오버헤드를 줄이기 위하여 문서들을 모든 슬레이브 오디세우스에 중복하여 저장한다.

4.2.3 문서 식별자(Document ID)와 키워드 식별자(Keyword ID)의 혼합(Hybrid) 분할

문서 식별자 분할 방법과 키워드 식별자 분할 방법은 제 2.2.3절에서 설명한 바와 같이 취약점들이 있으며 이러한 문제점들을 해결하기 위하여 본 논문에서는 문서 식별자와 키워드 식별자를 혼합 사용하여 역 색인을 분할하는 혼합 분할 방법을 제안한다.

혼합 분할 방법에서는 병렬 정보 검색 시스템에 저장하고자 하는 전체 문서에 대해서가 아니라 문서 식별자 분할에 의해 만든 각 문서 블록에 대해서 키워드 식별자 분할 방법을 적용하도록 한다. 혼합 분할 방법을 사용하는 경우, 문서 식별자 분할 방법에서의 하나의 문서 블록에 관련된 역 색인을 다시 키워드 식별자를 사용하여 분할하게 되므로, 각 역 색인 블록은 하나의 문서 블록내의 특정 키워드들에 대한 포스팅 리스트들을 색인한다. 예를 들어, 혼합 분할 방법을 사용하기 위하여 그림 2의 역 색인 예제를 키워드 식별자 분할 방법을 사



(a) 문서 블록

역 색인 블록 1

키워드	키워드 식별자	포스팅 개수	포스팅 리스트 (DocID, Position)
layer	10	2	(9, 2), (9, 10) .....

역 색인 블록 2

키워드	키워드 식별자	포스팅 개수	포스팅 리스트 (DocID, Position)
protocol	45	1	(9, 8) .....

(b) 문서 블록 1에 대해 키워드 식별자 분할 방법을 사용하여 생성된 역 색인 분할

역 색인 블록 3

키워드	키워드 식별자	포스팅 개수	포스팅 리스트 (DocID, Position)
layer	10	1	(15,4) .....

역 색인 블록 4

키워드	키워드 식별자	포스팅 개수	포스팅 리스트 (DocID, Position)
protocol	45	2	(15, 12), (15, 30) .....

(c) 문서 블록 2에 대해 키워드 식별자 분할 방법을 사용하여 생성된 역 색인 분할

그림 9 혼합 분할 방법을 사용한 역 색인 분할의 예

용한 결과는 그림 9(b), 9(c)와 같다. 그림 2의 각 역 색인 블록에 대하여 키워드 식별자 분할 방법을 사용하므로 그림 9(b), 9(c)에서와 같이 "layer"와 "protocol"의 포스팅들이 서로 다른 역 색인 블록에 색인된다.

혼합 분할 방법을 사용하여 병렬 정보 검색 시스템을 구현하기 위하여서는 특정 문서의 특정 키워드에 대한 포스팅 리스트들이 하나의 역 색인 블록에만 포함되도록 하는 것이 필요하다. 본 논문에서는 이를 위하여 병렬 정보 검색 시스템에 저장하고자 하는 문서들을 여러 문서 블록들로 나누고, 각 문서 블록의 포스팅 리스트들을 키워드에 따라 다시 분할한 후, 각 블록에 대해 역 색인을 구축한다. 혼합 분할 방법을 사용하여 병렬 정보 검색 시스템을 구축하는 상세한 과정은 그림 10과 같다. 병렬 정보 검색 시스템에 저장하고자 하는 문서들로부터 문서 블록들을 만든 후, 각 문서 블록의 포스팅 리스트들을 키워드에 따라 분할하고, 각 슬레이브 오디세우스들에게 분배하여, 분배된 포스팅 리스트들에 대하여 정보 검색 시스템 구축 과정을 거쳐 역 색인을 구축하도록 한다.

혼합 분할 방법을 이용한 병렬 정보 검색 시스템 구축 과정 혼합 분할 방법을 사용하는 경우, 특정 키워드를 가지는 문서의 검색은 각 문서 블록에 대하여 해당

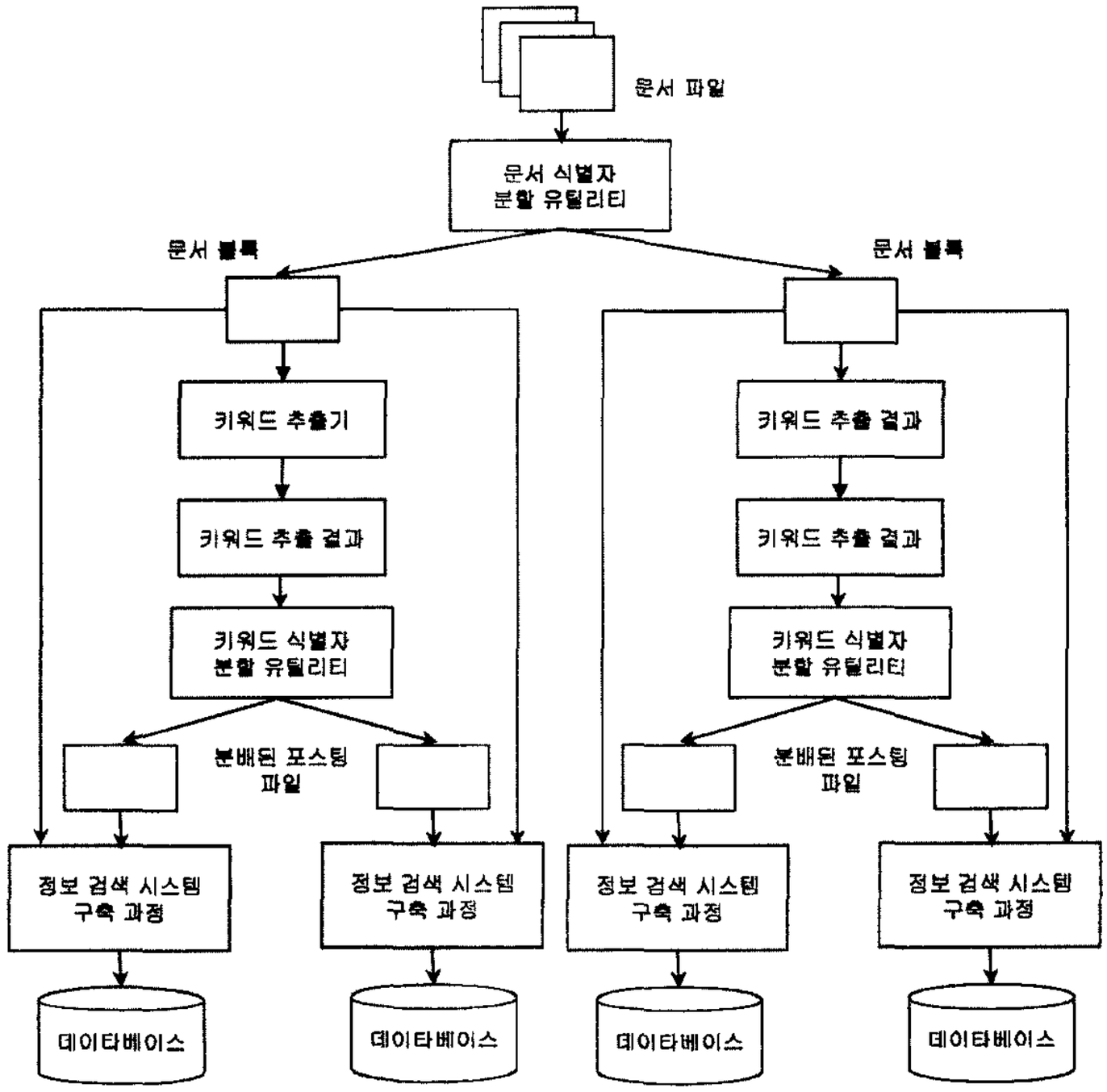


그림 10 혼합 분할 방법을 이용한 병렬 정보 검색 시스템 구축 과정

키워드를 색인하는 역 색인 블록들 만을 검색하고, 각 문서 블록들로부터의 결과들을 병합함으로써 수행된다. 예를 들어, 그림 9의 역 색인 예제에서 "layer"라는 키워드를 포함하는 문서를 검색하고자 하는 경우, 문서 블록 1에 대해서는 역 색인 블록 1을 검색하고, 문서 블록 2에 대해서는 역 색인 블록 3을 검색하여 결과를 병합한다.

혼합 분할 방법은 키워드 식별자 분할 방법의 장점을 유지하면서도 문서 식별자 분할 방법의 장점을 제공할 수 있다는 특징이 있다. 즉, top-k 질의를 효율적으로 처리해야 할 필요가 있는 환경에서 키워드 식별자 분할 방법을 쓰면 문서의 추가가 어렵다는 문제가 있는데, 혼합 분할 방법을 사용하면 문서의 추가가 용이하면서도 top-k 질의를 효율적으로 처리할 수 있다. 이것은 문서 식별자 분할 방법이나 키워드 식별자 분할 방법 중 하나만을 선택해서 사용할 수 있었던 기존의 연구들[6, 20]과는 차별화되는 오디세우스/Parallel-OOSQL만의 특징이다. 결과적으로, 오디세우스/Parallel-OOSQL은 세 가지 분할 방법을 모두 제공하므로 병렬 정보 검색 시스템의 응용 환경에 따라 분할 방법을 커스터마이즈하여 사용할 수 있다.

4.3 질의 처리

본 절에서는 병렬 정보 검색 질의의 처리 과정에 대해서 설명한다. 제 4.3.1절에서는 병렬 정보 검색 질의를 처리하는 과정에 대해 설명하고, 제 4.3.2절에서는 병렬 정보 검색 질의 처리를 위한 질의 계획 생성 과정에 대해서 설명한다. 제 4.3.3절에서는 생성한 질의 계획의 실행 과정에 대해서 설명한다.

4.3.1 질의 처리 과정

그림 11은 구현한 병렬 정보 검색 시스템에서 정보 검색 질의를 병렬적으로 처리하기 위하여서 마스터 오

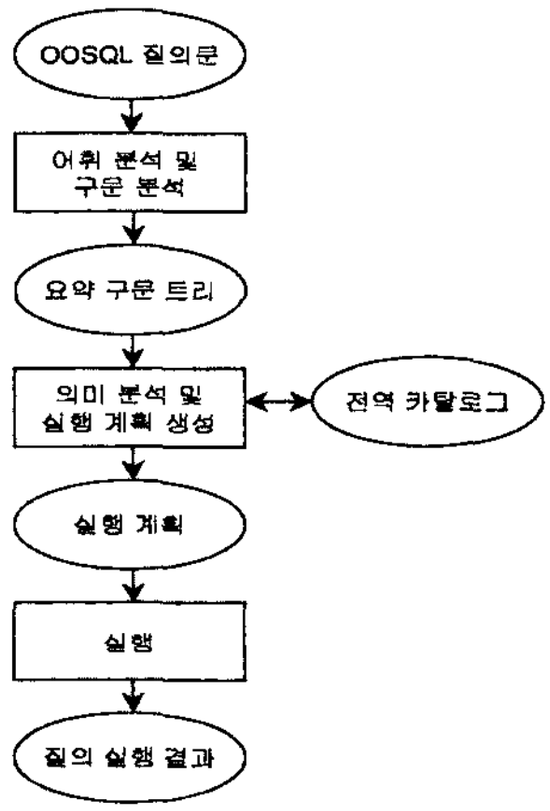


그림 11 병렬 질의 처리 과정

디세우스에서 수행하는 질의 처리 과정을 도시한다. 첫 번째로, 사용자로부터 받은 질의에 대하여 어휘 분석과 구문 분석을 한 후, 질의의 구조적인 정보를 가지는 요약 구문 트리를 얻는다. 두 번째로 전역 카탈로그를 참조하여 요약 구문 트리로부터 질의의 실행 계획을 생성한다. 마지막으로, 전 단계에서 생성된 실행 계획을 실행함으로써 질의의 최종 결과를 얻는다.

4.3.2 질의 실행 계획의 생성

질의 실행 계획은 병렬 정보 검색 질의의 처리를 위하여 마스터 오디세우스에서 수행해야 하는 일들을 나타내며, 어휘 분석 및 구문 분석 단계로부터 얻어진 질의의 구조적인 정보를 가지는 요약 구문 트리로부터 전역 카탈로그의 역 색인 분할 정보를 참조하여 생성된다.

질의 실행 계획은 크게 마스터 오디세우스에서 슬레이브 오디세우스에 질의를 보내 역 색인 블록을 검색하도록 하는 부질의(subquery) 실행 계획과 마스터 오디세우스에서 슬레이브 오디세우스로부터의 질의 결과를 받아 후처리를 하는 연산(operation) 실행 계획으로 나누어 진다. 부질의 실행 계획은 슬레이브 오디세우스에서 키워드를 검색하기 위한 계획이므로 검색해야 하는 키워드가 명시된 WHERE절의 키워드 검색 조건으로부터 생성되며, 연산 실행 계획은 단일 정보 검색 시스템에서와 동일한 결과를 얻기 위하여 부질의 실행 계획의 실행 결과를 대상으로 하여 생성된다.

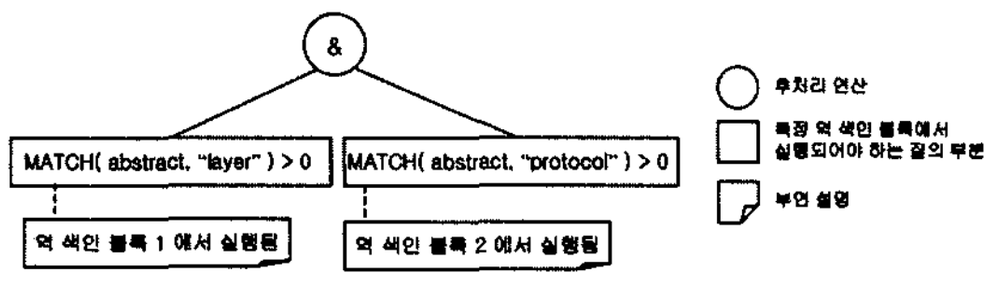
병렬 질의 처리를 위하여 주어진 질의로부터 질의 실행 계획을 생성하는 알고리즘을 요약하면 다음과 같다.

질의 실행 계획 생성 과정

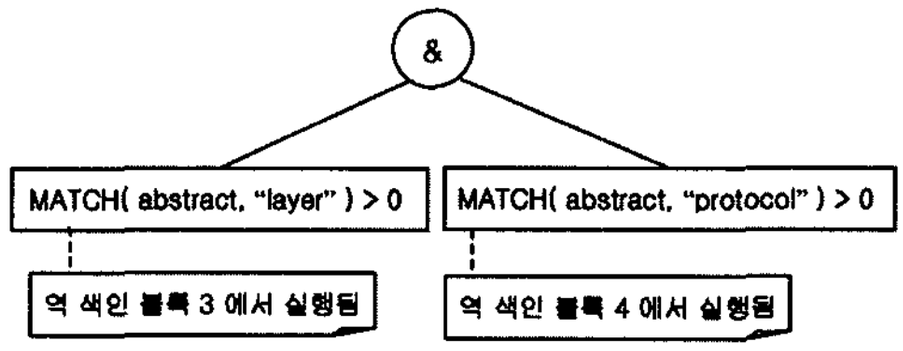
1. 요약 구문 트리를 참조하여 사용자로부터 받은 질의를 분석한다.
2. 전역 카탈로그를 참조하여 질의의 WHERE절에 명시되어 있는 조건들을 실행할 수 있는 슬레이브 오디세우스들을 결정한다.
3. 각 슬레이브 오디세우스에 대해 실행되어야 하는 부질의 실행 계획과 마스터 오디세우스가 실행하게 되는 후처리에 대한 연산 질의 실행 계획을 생성한다.

그림 12는 위의 알고리즘을 이용하여 병렬 질의 처리를 위한 실행 계획을 생성하는 예를 보여준다. 역 색인이 그림 9(b), 9(c)와 같이 분할되어 있고, 그림 5(b)와 같은 질의를 처리하기 위하여서 WHERE절의 검색 조건을 처리해야 하는 슬레이브 오디세우스를 결정한 결과는 그림 12(a), 12(b)와 같으며, 최종적인 질의 실행 계획은 그림 12(c)와 같다.

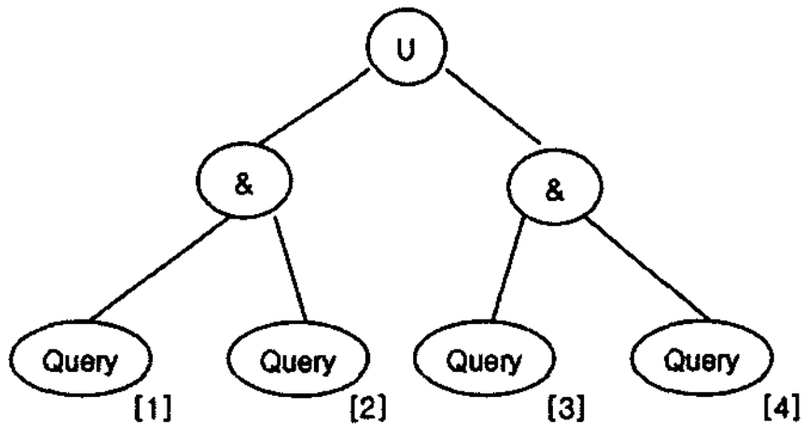
질의 처리 결과를 얻기 위하여 문서 블록 1에서는, 역 색인 블록 1에서 “layer”를 포함하는 문서를 검색하고, 역 색인 블록 2에서 “protocol”을 포함하는 문서를 검색한 후, 결과에 대하여 AND 연산을 하도록 한다. 마찬



(a) 문서 블록 1에 대한 요약 구문 트리의 where절 부분



(b) 문서 블록 2에 대한 요약 구문 트리의 where절 부분




```
부질의 계획 [1], [3]을 위한 질의
    select docID, title from PaperAbstract where match( abstract, "layer" ) > 0 ;
부질의 계획 [2], [4]를 위한 질의
    select docID, title from PaperAbstract where match( abstract, "protocol" ) > 0 ;
```

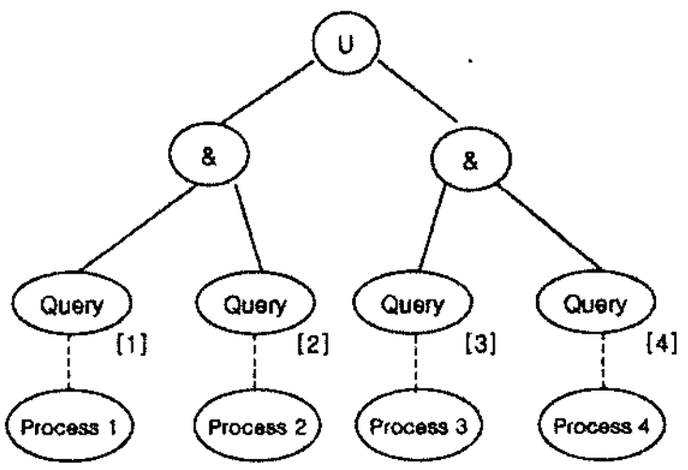
(c) 생성된 질의 실행 계획 및 질의

그림 12 질의 실행 계획의 생성 예제

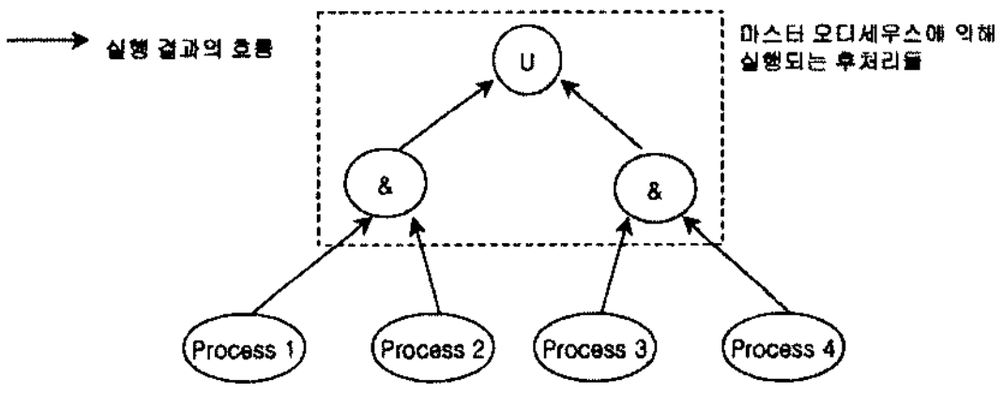
가지로, 문서 블록 2번과 관련된 역 색인 블록 3, 4에서 각각 “layer”와 “protocol”을 포함하는 문서를 검색한 후 결과에 대해 AND 연산을 한다. 마지막으로 최종 결과를 얻기 위하여 두 문서 블록으로부터의 결과를 병합한다.

4.3.3 질의 실행 계획의 실행

병렬 정보 검색 결과는 생성된 실행 계획을 해석하면서 처리해야 하는 실행 계획이 부질의 실행 계획인 경우 생성된 질의를 각 슬레이브 오디세우스에 전달하여 수행하도록 하고 결과를 받아오며, 연산 실행 계획인 경우 각 슬레이브 오디세우스로부터의 질의 수행 결과에 대하여 후처리를 함으로서 얻어진다. 슬레이브 오디세우스로의 부질의 실행 계획을 병렬적으로 실행하기 위하여 각 부질의 실행 계획은 별도의 프로세스(process)에 의해서 실행되고 각 프로세스들과 슬레이브 오디세우스는 RPC를 사용하여 통신을 한다. 부질의 실행 계획을 실행하는 프로세스는 공유 메모리(shared memory)를 사용하여 후처리를 하는 프로세스에게 질의 실행 결과를 전달한다. 예를 들어, 그림 12의 실행 예제인 그림 13을 보면, 프로세스 1, 2 및 3, 4가 실행 계획 [1], [2]



(a) 질의 실행 계획



(b) 실행 결과의 흐름

그림 13 질의 실행 계획의 실행 예제

와 [3], [4]를 위한 질의를 해당 슬레이브 오디세우스에 전달하여 실행하도록 하고 결과를 받음으로써 부질의 실행 계획을 실행하게 되며, 마스터 오디세우스를 실행하고 있는 프로세스가 후처리를 실행함으로써 최종 결과를 얻는다.

5. 성능 평가

본 절에서는 본 논문에서 구현한 병렬 정보 검색 시스템의 성능 평가를 하고 결과를 제시한다. 병렬 정보 검색 시스템은 문서 식별자 분할 방법과 키워드 식별자 분할 방법 및 혼합 분할 방법을 사용하여 구축한다. 제 5.1절에서는 실험 방법에 대해서 설명하고, 제 5.2절에서는 각 역 색인 분할 방법을 사용하여 구축된 병렬 정보 검색 시스템에 대한 실험 결과를 보이고 결과를 분석한다.

5.1 실험 방법

본 실험에서는 단일 정보 검색 시스템과 각 역 색인 분할 방법을 사용하여 구축된 병렬 정보 검색 시스템에서의 질의 처리 시간을 측정하여 비교 한다. 실험을 위한 데이타는 웹 로봇을 통해 얻은 1,300,000건의 웹 페이지이며, 실험 데이타를 데이타베이스에 저장하기 위하여서 사용된 데이타의 스키마는 웹 페이지의 제목과 내용을 나타내는 두 개의 컬럼으로 구성된다.

표 2는 실험에서 사용한 질의와 질의 결과 건수이다. 첫째, 예제 질의 1~3은 하나의 키워드로 구성된 질의로서 결과 건수의 변화에 따른 질의 처리 시간을 측정하기 위하여 사용된다. 둘째, 예제 질의 4~9는 두 개의 키워드로 구성된 질의로서 키워드들이 존재하는 역 색인 블록의 위치 변화에 따른 질의 처리 시간을 측정하기 위하여 사용된다. 예제 질의 10, 11은 세 개의 키워드로 구성된 질의이다. 예제 질의 4~11에 사용된 키워드들을 선정한 기준에 대한 자세한 설명은 제 5.1절과 제 5.2절에서 한다. 마지막으로, 예제 질의 12는 top-k 질의로서 k의 변화에 따른 질의 처리 시간을 측정하기 위하여 사용된다. 본 실험에서는 연산자로 병렬 정보 검색 시스템에서 중요하고 자주 사용되는 AND와 OR 연산자를 사용하였다.

표 2 예제 질의

번호	질의	결과 건수	비고
1	"jpg"	9,544	키워드 1개로 구성된 질의
2	"과학"	89,769	
3	"게시판	600,462	
4	"jpg" AND "공고"	7	키워드 2개로 구성된 질의
5	"jpg" OR "공고"	19,458	
6	"jpg" AND "대통령"	66	
7	"jpg" OR "대통령"	19,341	
8	"jpg" AND "순위"	51	
9	"jpg" OR "순위"	19,513	
10	"jpg" AND "대통령" AND "순위"	2	키워드 3개로 구성된 질의
11	"jpg" OR "대통령" OR "순위"	29,188	
12	"copyright"	748,587	Top-k 질의

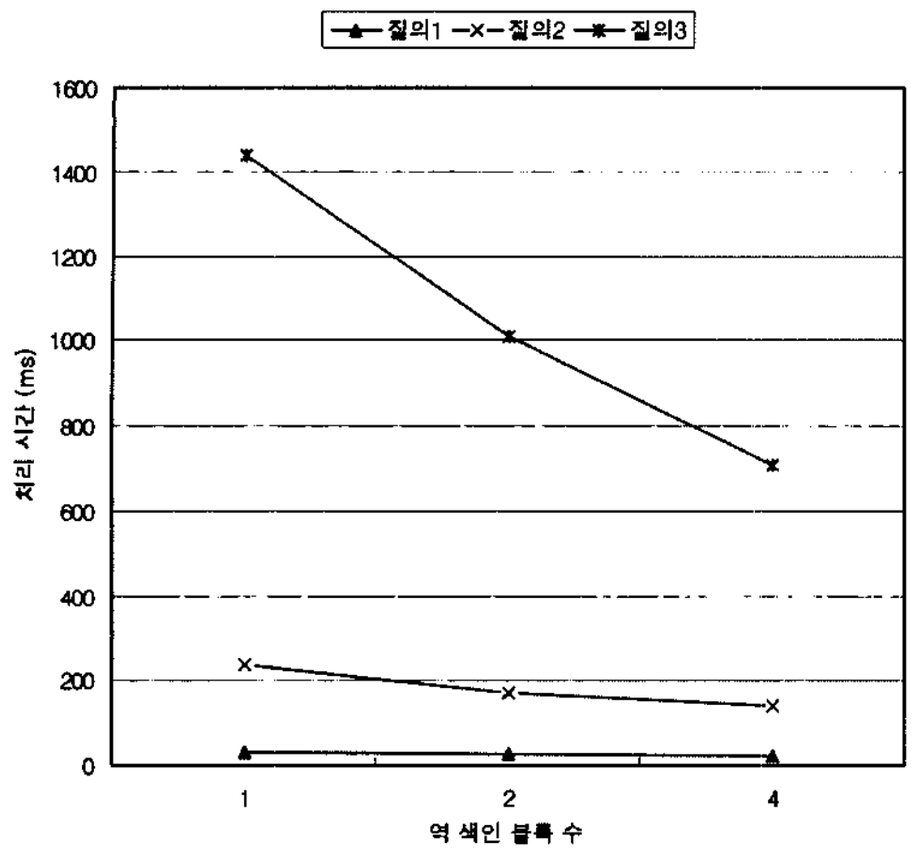
실험은 3.0GHz의 중앙 처리 장치 속도와 2GByte의 메모리, 300GByte 디스크 15개로 구성되는 디스크 어레이(7,200 RPM)를 갖는 다섯 대의 PC들에서 RedHat Linux를 운영체제로 하여 수행한다.

5.2 실험 결과

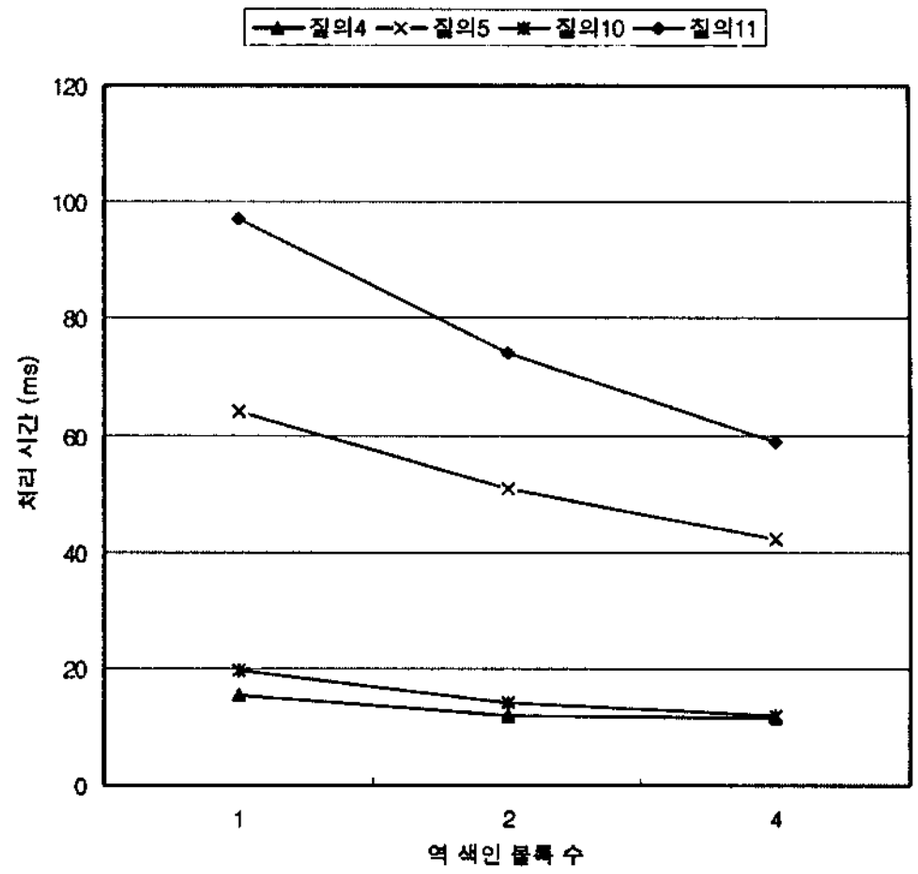
본 절에서는 수행된 실험의 결과를 보이고 결과를 분석한다. 제 5.2.1절에서는 문서 식별자 분할 방법을 사용하여 병렬 정보 검색 시스템을 구축한 경우에 대하여 실험 및 결과를 분석하고 제 5.2.2절에서는 키워드 식별자 분할 방법을 사용하여 병렬 정보 검색 시스템을 구축한 경우에 대하여 실험 및 결과를 분석한다. 제 5.2.3절에서는 혼합 분할 방법을 사용하여 병렬 정보 검색 시스템을 구축한 경우에 대하여 실험 및 결과를 분석하고 제 5.2.4절에서는 각 분할 방법 간의 실험 결과를 비교한다.

5.2.1 문서 식별자 분할 방법

본 절에서는 문서 식별자 분할 방법을 사용하여 구축한 병렬 정보 검색 시스템과 단일 정보 검색 시스템의 성능을 비교한다. 병렬 정보 검색 시스템은 실험 데이타를 두 개의 문서 블록으로 분할한 시스템과 네 개의 문서 블록으로 분할한 시스템을 사용한다. 실험 결과로서 문서 블록 개수의 증가에 따른 질의 처리 시간의 변화



(a) 단일 키워드로 구성된 질의



(b) 두개 이상의 키워드로 구성된 질의

그림 14 문서 식별자 분할 방법에서의 실험 결과

를 측정하고, 결과를 분석한다.

그림 14는 문서 식별자 분할 방법을 사용한 경우, 문서 블록 개수의 증가에 따른 예제 질의 처리 시간의 변화를 나타낸 것이다. (a)는 단일 키워드로 구성된 질의에 대한 실험 결과이고, (b)는 두 개 이상의 키워드로 구성된 질의에 대한 실험 결과이다. 그래프에서 가로축은 문서 블록의 개수를 나타내며, 세로축은 질의 처리 시간이다. 질의 처리 시간에는 마스터 오디세우스나 슬레이브 오디세우스 사이의 통신 시간이 포함되어 있다. 실험 결과, 모든 예제 질의에 대해서 병렬 정보 검색 시스템에서의 질의 처리 시간이 단일 정보 검색 시스템에 비해 더 작다. 이는 문서들이 여러 문서 블록들로 나누어져 포스팅 리스트들이 문서 블록에 따라 여러 역 색인 블록으로 나누어지고, 키워드 검색을 위하여 각 역 색인 블록을 병렬적으로 검색하므로 전체적인 역 색인의 검색 시간이 줄어들기 때문이다. 또한, 질의 처리 시간이 역 색인 블록의 개수에 근사하게 비례하여 감소함을 알 수 있다. 예제 질의3의 경우, 역 색인 블록이 한 개인 단일 정보 시스템의 경우 처리 시간이 1,438ms이고, 역 색인 블록이 두 개인 병렬 정보 시스템의 경우 처리 시간이 1,010ms로 단일 정보 시스템에서 보다 약 1.42배 빠르게 처리되며 역 색인 블록이 네 개일 때는 처리 시간이 708ms로 단일 정보 검색 시스템에서 보다 약 2.03배 빠르게 처리된다. 마지막으로, 예제 질의 6, 7 및 8, 9는 예제 질의 4, 5의 실험 결과와 비슷한 경향을 보였다. 문서 식별자 분할 방법은 모든 역 색인 블록을 검색하므로 질의 처리 시간이 질의에 사용된 키워드 종류에 크게 영향을 받지 않기 때문이다.

5.2.2 키워드 식별자 분할 방법

본 절에서는 키워드 식별자 분할 방법을 사용하여 구축한 병렬 정보 검색 시스템과 단일 정보 검색 시스템의 성능을 비교한다. 병렬 정보 검색 시스템은 역 색인을 키워드 식별자에 따라 두 개로 분할한 시스템과 네 개로 분할한 시스템을 사용한다. 실험 결과로서 예제 질의의 질의 처리 시간을 측정하고 결과를 보인다.

키워드 식별자에 따라 역 색인을 분할한 경우, 각 역 색인 블록이 색인하는 키워드들은 표 3~4와 같다. 역 색인 블록 수가 두 개인 경우(표 3), 예제 질의에서 검색하고자 하는 키워드인 “copyright”, “jpg”, “검정색”, “공고”, “대통령”은 역 색인 블록 1에, “순위”는 역 색인 블록 2에 색인된다. 마찬가지로, 역 색인 블록 수가 네 개인 경우(표 4), “copyright”, “jpg”, “검정색”, “공고”는 역 색인 블록 1에, “대통령”은 역 색인 블록 2에, “순위”는 역 색인 블록 3에 색인된다.

표 3~4의 분할 정보에 의거하여 키워드 분할 방법에서 실험에 사용할 각 예제 질의의 질의 처리 시 접근하는 역 색인 블록의 수를 나타내면 표 5와 같다. 예를 들어, 예제 질의 10에서 역 색인 블록 수가 2인 경우, “jpg”와

표 3 키워드 식별자 분할 방법에서의 역 색인 분할 정보 (역 색인 블록 수 = 2)

키워드 범위 블록	1	2
키워드 범위	10대 ~ copyright ~ jpg ~ 검정색 ~ 공고 ~ 대통령 ~ 사진첨부	사진첩 ~ 순위 ~ 힛트작 ~

표 4 키워드 식별자 분할 방법에서의 역 색인 분할 정보
(역 색인 블록 수 = 4)

키워드 범위 블록	1	2	3	4
키워드 범위	10대 ~ copyright ~ jpg ~ 검정색 ~ 공고 ~ 구운동영상	구운마늘 ~ 대통령 ~ 사진첨부	사진첩 ~ 순위 ~ 잡종	잡지 ~ 힛트작

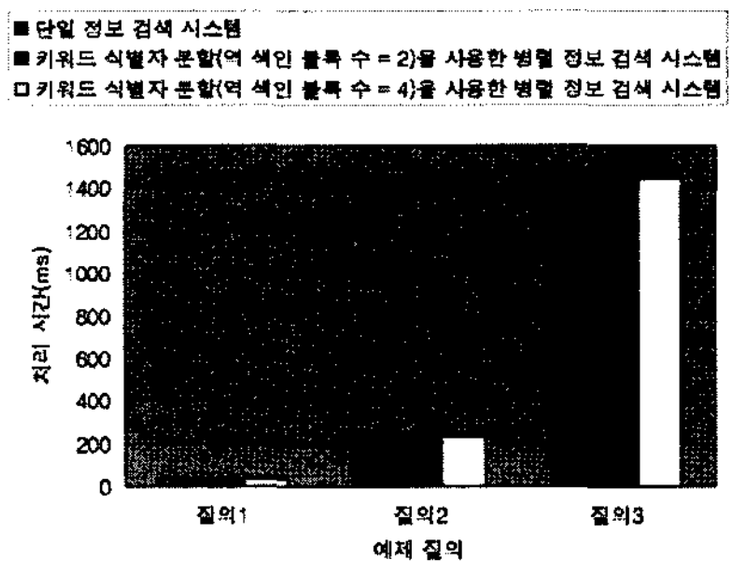
표 5 키워드 분할 방법에서 예제 질의에 따라 접근하는 역 색인 블록의 수

사용된 키워드 개수 / 예제 질의 / 분할 방법	1개	2개			3개
	질의 1, 2, 3	질의 4, 5	질의 6, 7	질의 8, 9	질의 10, 11
단일 정보 검색 시스템	1	1	1	1	1
키워드 식별자 분할 방법 (역 색인 블록 수 = 2)	1	1	1	2	2
키워드 식별자 분할 방법 (역 색인 블록 수 = 4)	1	1	2	2	3

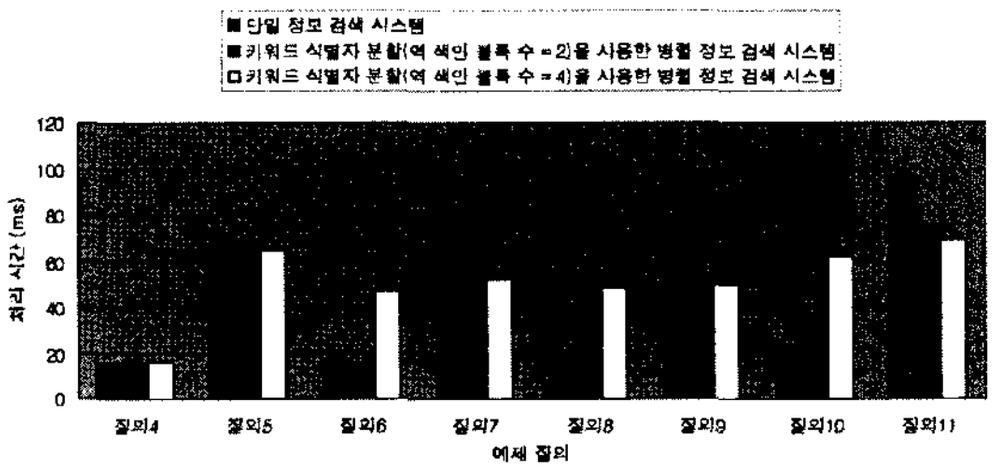
"대통령"은 역 색인 블록 1에, "순위"는 역 색인 블록 2에 저장되어 질의 처리 시 총 2개의 역 색인 블록을 접근하고, 역 색인 블록 수가 4인 경우, "jpg"는 역 색인 블록 1에 "대통령"은 역 색인 블록 2에 "순위"는 역 색인 블록 3에 저장되어 질의 처리 시 총 3개의 역 색인 블록을 접근한다.

그림 15는 키워드 식별자 분할 방법을 사용한 경우의 예제 질의의 처리 시간을 나타낸 것이다. (a)는 단일 키워드로 구성된 질의에 대한 실험 결과이고, (b)는 두 개 이상의 키워드로 구성된 질의에 대한 실험 결과이다. 그래프에서 가로축은 각 예제 질의를 나타내며, 세로축은 질의 처리 시간이다. 질의 처리 시간에는 마스터 오디세우스나 슬레이브 오디세우스 사이의 통신 시간이 포함되어 있다. 실험 결과, 접근하는 역 색인 블록 수가 서로 같은 예제 질의 1, 2, 3, 4, 5에 대해서는 병렬 정보 검색 시스템들에서의 질의 처리 시간이 단일 정보 검색 시스템의 경우와 거의 비슷하다. 이는 키워드 식별자 분할 방법을 사용하는 경우, 키워드 검색이 하나의 역 색인 블록에서만 수행되기 때문이다. 또한, 예제 질의 6과 7에서 역 색인 블록 수가 2일 때도 같은 현상이 나타나는데 이는 "jpg"와 "공고"가 같은 역 색인 블록에 색인되어 있기 때문이다.

한편, 예제 질의 6에서 역 색인 블록 수가 4일 때가



(a) 단일 키워드로 구성된 질의



(b) 두 개 이상의 키워드로 구성된 질의
그림 15 키워드 분할 방법에서의 실험 결과

2일 때 보다 질의 처리 시간이 더 오래 걸린다. 그 이유는 다음과 같다. 예제 질의 6에 사용된 키워드 "jpg"와 "대통령"은 역 색인 블록 수가 2일 때는 같은 역 색인 블록에 나타나고, 4일 때는 서로 다른 역 색인 블록에 나타난다. "jpg"와 "대통령"이 같은 역 색인 블록에 나타날 때는 단일 슬레이브에서 "jpg" AND "대통령"을 처리한 후에 181개의 결과만을 마스터로 전송하지만, "jpg"와 "대통령"이 서로 다른 역 색인 블록에 나타날 때는 "jpg"의 결과 약 10,000건과 "대통령"의 결과 약 10,000건을 각각 마스터로 전송한 뒤에 마스터에서 "jpg" AND "대통령"을 처리한다. 따라서, 역 색인 블록 수가 4일 때는 2일 때보다 데이타 전송량이 더 많아지기 때문에 병렬적으로 검색하더라도 전체적인 질의 처리 시간이 더 오래 걸린다.

예제 질의 11에서 역 색인 블록 수가 4일 때 병렬 정보 검색 시스템에서의 질의 처리 시간이 단일 정보 검색 시스템에 비해 작다. 이는 병렬 정보 검색 시스템에서 "jpg", "대통령", "순위"를 포함하고 있는 각 역 색인 블록들이 모두 병렬적으로 검색되기 때문이다.

5.2.3 문서 식별자와 키워드 식별자의 혼합 분할 방법

본 절에서는 혼합 분할 방법을 사용하여 구축한 병렬 정보 검색 시스템과 단일 정보 검색 시스템의 성능을 비교한다. 병렬 정보 검색 시스템은 실험 데이타를 두 개의 문서 블록으로 나누고 각 문서 블록과 관련된 역 색인을 키워드 분할 방법에 따라 각각 두 개로 분할하

표 6 혼합 분할 방법에서의 역 색인 분할 정보

문서 블록	1		2	
키워드 범위 블록	1	2	3	4
키워드 범위	10대 ~ copyright ~ jpg ~ 검정색 ~ 공고 ~ 대통령 ~ 사운더스	사운드 ~ 순위 ~ 힛트작	10대 ~ copyright ~ jpg ~ 검정색 ~ 공고 ~ 대통령 ~ 산양제	산업 ~ 순위 ~ 힛트작

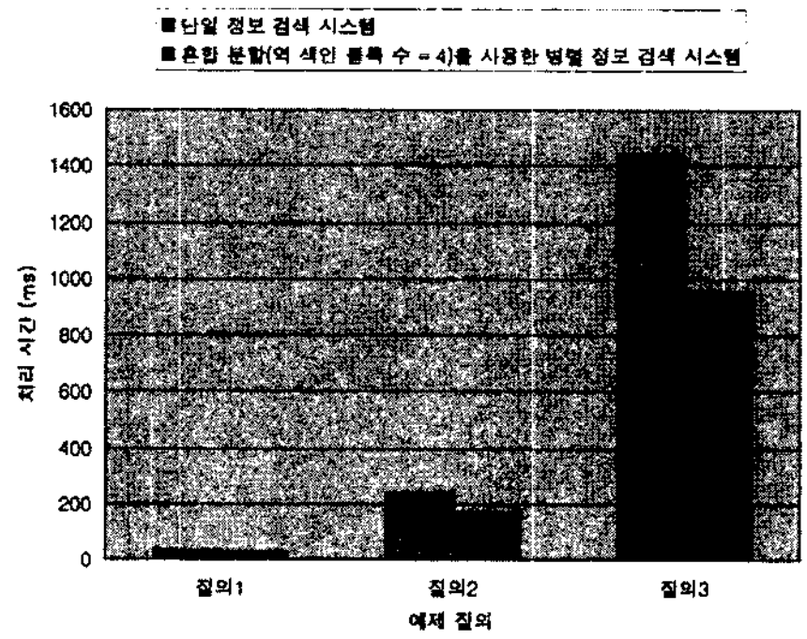
표 7 혼합 분할 방법에서 예제 질의에 따라 접근하는 역 색인 블록의 수

사용된 키워드 개수 / 예제 질의 / 분할 방법	1개	2개		3개
	질의 1, 2, 3	질의 4, 5, 6, 7	질의 8, 9	질의 10, 11
단일 정보 검색 시스템	1	1	1	1
혼합 분할 방법	2	2	4	4

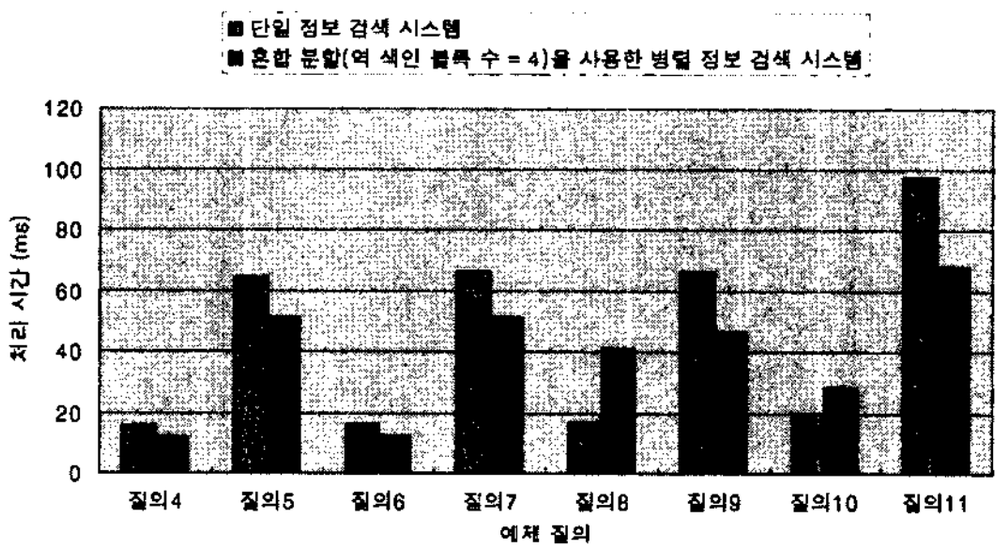
여 구축한다. 각 문서 블록과 관련된 역 색인 블록이 색인하는 키워드 범위는 표 6과 같다. 실험 결과로서 예제 질의의 처리 시간을 측정하고 결과를 보인다.

표 7은 혼합 분할 방법에서 실험에 사용할 각 예제 질의의 질의 처리 시 접근하는 역 색인 블록의 수를 나타낸 것이다. 예를 들어, 예제 질의 8의 경우, "jpg"는 문서 블록 1에서 키워드 블록 1과 문서 블록 2에서 키워드 블록 1에 저장되고, "순위"는 문서 블록 1에서 키워드 블록 2와 문서 블록 2에서 키워드 블록 2에 저장되어 질의 처리 시 총 4개의 역 색인 블록을 접근한다.

그림 16은 혼합 분할 방법을 사용하여 병렬 정보 검색 시스템을 구축한 경우의 예제 질의의 처리 시간을 비교한 것이다. (a)는 단일 키워드로 구성된 질의에 대한 실험 결과이고, (b)는 두 개 이상의 키워드로 구성된 질의에 대한 실험 결과이다. 그래프에서 가로축은 각 예제 질의를 나타내며, 세로축은 질의 처리 시간이다. 질의 처리 시간에는 마스터 오디세우스나 슬레이브 오디세우스 사이의 통신 시간이 포함되어 있다. 실험 결과, 질의 9, 11을 제외한 모든 예제 질의들에 대해 혼합 분할 방법을 사용한 병렬 정보 검색 시스템에서의 질의 처리 시간이 단일 정보 검색 시스템에 비해 더 작다. 예를 들어, 예제 질의 3의 경우 단일 정보 검색 시스템에서의 처리 시간이 1,438ms이고, 혼합 분할 방법을 사용한 병렬 정보 검색 시스템에서의 처리 시간이 953ms로



(a) 단일 키워드로 구성된 질의



(b) 두 개 이상의 키워드로 구성된 질의

그림 16 혼합 분할 방법에서의 실험 결과

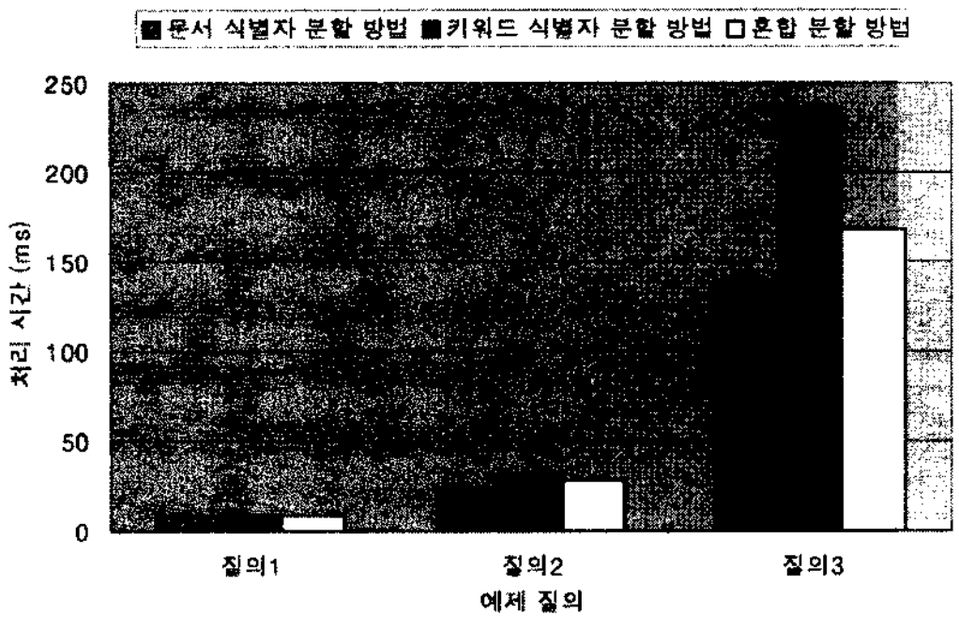
약 1.51배 향상되었다. 이는 키워드 검색을 위하여 각 역 색인 블록을 병렬적으로 검색하므로 전체적인 역 색인의 검색 시간이 줄어들기 때문이다.

5.2.4 분할 방법 별 비교

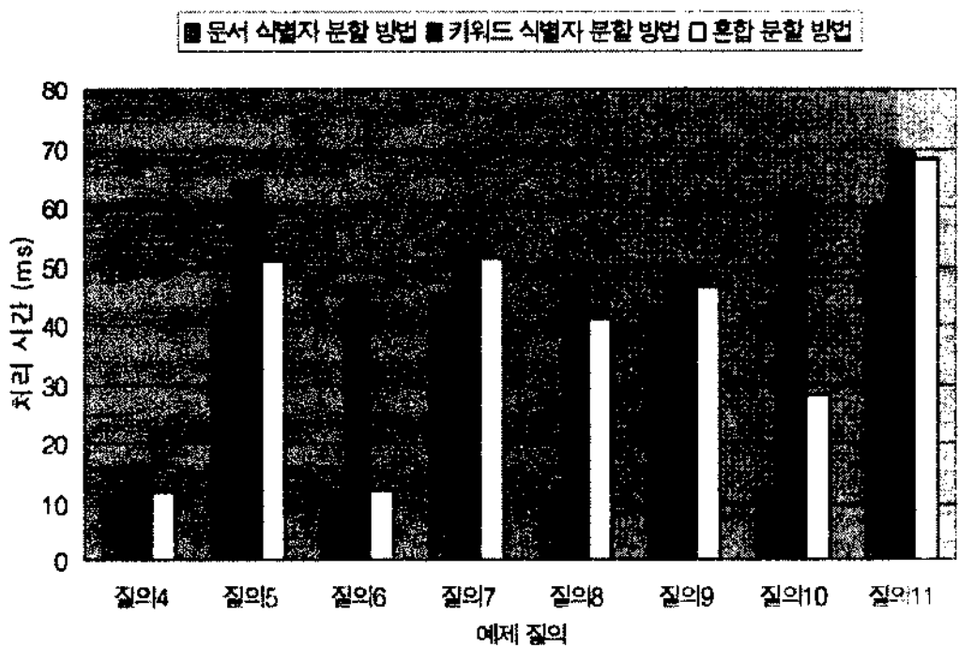
본 절에서는 각 분할 방법 간의 실험 결과를 비교 설명한다. 그림 17의 (a)~(c)에서 보는 바와 같이 문서 식별자 분할 방법은 top-k 질의를 제외한 나머지 질의에서 가장 좋은 성능을 보인 반면 top-k 질의에서는 가장 성능이 떨어진다. 그리고, 키워드 식별자 분할 방법은 top-k 질의에서는 가장 좋은 성능을 보인 반면 top-k 질의를 제외한 나머지 질의에서는 성능이 떨어진다. 그러나, 혼합 분할 방법은 문서 식별자 분할 방법과 키워드 식별자 분할 방법의 장점을 모두 가진다. 즉, top-k 질의를 포함한 모든 질의에서 전반적으로 좋은 성능을 보인다.

6. 결 론

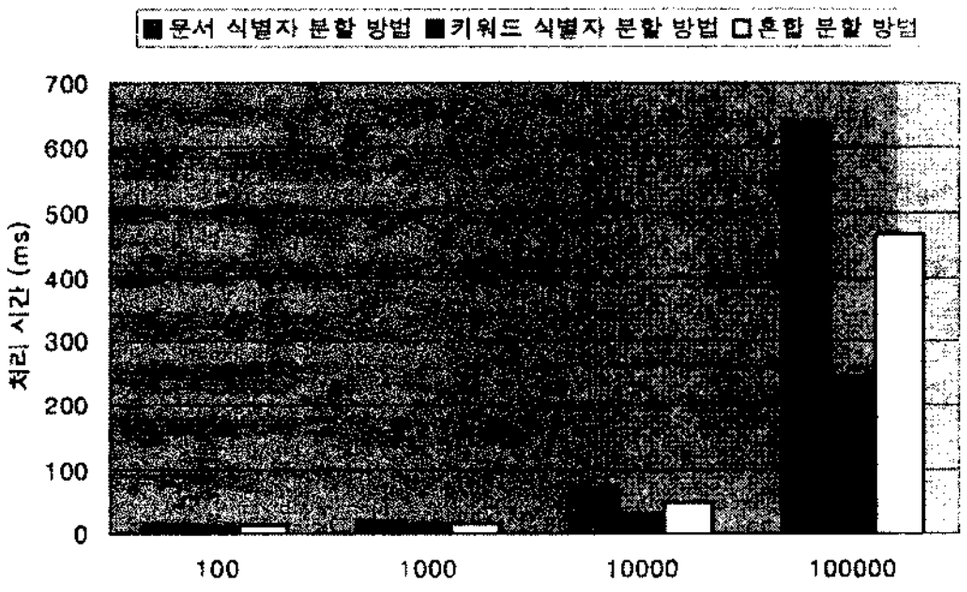
본 논문에서는 대용량의 문서에 대하여 효율적인 정보 검색을 할 수 있도록 오디세우스를 사용하여 병렬 정보 검색 시스템을 구현하였다. 병렬 정보 검색은 정보 검색 시스템에서 주어진 키워드가 발생한 문서를 찾기 위하여 사용하는 역 색인을 분할하고, 이를 병렬적으로



(a) 결과 건수에 따른 분할 방법 별 비교 실험 결과



(b) 키워드들이 존재하는 역 색인 블록의 위치 변화에 따른 비교 실험 결과



(c) Top-k 질의에 대한 분할 방법 별 비교 실험 결과

그림 17 분할 방법간의 비교 실험 결과

검색하여 실행될 수 있다. 본 논문의 공헌은 다음과 같이 요약될 수 있다.

첫째, 병렬 정보 검색 시스템을 구축하기 위한 기존의 역 색인 분할 방법을 분석하고, 이를 이용하여 병렬 정보 검색 시스템을 설계하고 구현하였다. 둘째, 기존의 역 색인 분할 방법인 문서 식별자 분할 방법과 키워드 식별자 분할 방법의 단점들을 보완하기 위하여 문서 식별자와 키워드 식별자를 혼합 사용하여 역 색인을 분할하는 새로운 방법을 제안하고 구현하였다. 셋째, 단일 정보 검색 시스템과의 질의 처리 시간 비교를 통하여 병렬 정보 검색 시스템의 유용성을 보였다. 실험 결과, 문서 식별자 기반 분할 방법은 질의 처리 시간이 역 색인 분할의 블록의 개수에 근사하게 비례하여 줄어들었으며, 키워드 식별자 분할 방법은 top-k 질의 처리에 우수한 성능을 보였다. 본 논문에서 제안한 병렬 정보 검색 시스템은 문서 식별자 분할, 키워드 식별자 분할, 그리고 혼합 분할을 모두 제공하므로 응용 환경에 따라 분할 방법을 커스터마이즈하여 항상 좋은 성능을 낼 수 있다. 따라서, 본 시스템은 정보 검색 분야에 매우 유용하게 사용될 수 있을 것으로 사료된다.

참 고 문 헌

[1] Frakes, W. and Baeze-Yates, R., *Information Retrieval: Data Structures and Algorithms*, Prentice-Hall, 1992.

[2] Tomasic, A., Garcia-Molina, H., and Shoens, K., "Incremental Updates of Inverted Lists for Text Document Retrieval," In *Proc. 1994 ACM SIGMOD Int'l Conf. on Management of Data*, pp. 289-300, June 1994.

[3] Tomasic, A. and Garcia-Molina, H., "Issues in Parallel Information Retrieval," *IEEE Data Engineering Bulletin*, Vol.17, No.3, pp. 41-49, Sept. 1994.

[4] Cahoon, B. and McKinley, K., "Performance Evaluation of a Distributed Architecture for Information Retrieval," In *Proc. 19th Int'l Conf. on Information Retrieval(ACM SIGIR)*, 1996.

[5] Tomasic, A. and Garcia-Molina, H., "Query Processing and Inverted Indices in Shared-Nothing Text Document Information Retrieval Systems," In *The VLDB Journal*, Vol.2, No.3, pp. 243-275, 1993.

[6] MacFarlane, A., McCann, J., and Robertson, S., "PLIERS : A Parallel Information Retrieval System using MPI," In *Proc. 6th Europrean PVM/MPI Users' Group Meeting*, pp. 317-324, Sept. 1999.

[7] Grossman, D. and Frieder, O., *Information Retrieval: Algorithms and Heuristics*, Springer, Dec. 2004.

[8] Zobel, J., Moffat, A., and Ramamohanarao, K., "Inverted Files Versus Signature Files for Text Indexing," *ACM Trans. on Database Systems*, Vol.23, No.4, pp. 453-490, Dec. 1998.

[9] 황 규영, 이 민재, 이 재길, 김 민수, 한 욱신, "오디세우스/IR: 정보 검색 기능과 밀결합된 고성능 객체 관계형 DBMS", *한국정보과학회 논문지: 컴퓨팅의 실제*, Vol.11, No.3, pp. 209-215, 2005년 6월.

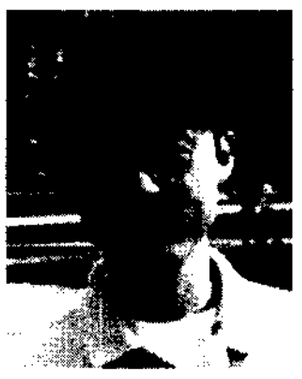
[10] 류 재준, 이 재길, 이 민재, 황 규영, "오디세우스/Parallel-OOSQL: 오디세우스 객체 관계형 데이타베이스 관리 시스템을 사용한 병렬 정보 검색 시스템", *한국정보과학회 봄 학술발표논문집(B)*, pp. 187-189, 2002년 4월.

[11] Salton, G., *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information*

by Computer, Addison-Wesley, Aug. 1988.
[12] Baeze-Yates, R. and Ribeiro-Neto, B., *Modern Information Retrieval*, ACM Press, 1999.
[13] Faloutsos, C. and Oard, D., "A Survey of Information Retrieval and Filtering Methods," Tech. Report: CS-TR 3514, Univ. of Maryland, Aug. 1995.
[14] 박 병권, "정보 검색과 데이타베이스 관리 시스템의 밀결합을 위한 역 색인 구조와 질의 최적화", 박사 학위 논문, KAIST 전산학과, 1998.
[15] Jeong, B. and Omiecinski, E., "Inverted File Partitioning Schemes in Multiple Disk Systems," *IEEE Trans. on Parallel and Distributed Systems*, Vol.6, No.2, pp. 142-153, Feb. 1995.
[16] Baeza-Yates, R., Castillo, C., Junqueira, F., Plachouras, V., and Silvestri, F., "Challenges on Distributed Web Retrieval," In *Proc. 23rd Int'l Conf. on Data Engineering*, Istanbul, Turkey, pp. 6-20, Apr. 2007.
[17] Chaudhuri, S. and Gravano, L., "Evaluating Top-k Selection Queries," In *Proc. 25th Int'l Conf. on Very Large Data Bases (VLDB)*, Edinburgh, Scotland, pp. 399-410, Sept. 1999.
[18] Chang, K. and Hwang, S., "Minimal probing: supporting expensive predicates for top-k queries," In *Proc. 2002 ACM SIGMOD Int'l Conf. on Management of Data*, Madison, Wisconsin, pp. 346-357, June 2002.
[19] Li, C., Chang, K., Ilyas, I., and Song, S., "RankSQL: query algebra and optimization for relational top-k queries," In *Proc. 2005 ACM SIGMOD Int'l Conf. on Management of Data*, Baltimore, Maryland, pp. 131-142, June 2005.
[20] Oracle Corp., interMedia Text, *http://otn.oracle.co.kr/docs/Oracle817/index.htm*, 1999.
[21] Google, http://www.google.com.
[22] Gulli, A. and Signorini, A., "The Indexable Web is More than 11.5 Billion Pages," In *Proc. 14th Int'l Conf. on World Wide Web*, pp. 902-903, Chiba, Japan, May 2005.
[23] Barroso, L. A., Dean, J., and Holzle, U., "Web Search for a Plant: The Google Cluster Architecture," *IEEE Micro*, Vol.23, No.2, pp. 22-28, Mar./Apr. 2003.
[24] 임 효상, 오디세우스/코스모스 객체 저장 시스템을 위한 벌크 로드 기능의 설계 및 구현, 석사 학위 논문, KAIST 전산학과, 1999.
[25] Codd, E. F., "Relational Completeness of Database Sublanguages," Prentice Hall and IBM Research Report RJ 987, San Jose, California, 1972.
[26] Bhatia, S. and Deogun, J., "Cluster Characterization in Information Retrieval," In *Proc. 1993 ACM/SIGAPP Symposium on Applied Computing States of the Art and Practice*, pp. 721-728, Feb. 1993.

류 재 준
1996년 3월~2000년 2월 숭실대학교 컴퓨터학부 학사. 2000년 3월~2002년 2월 한국과학기술원 전산학과 석사. 2002년 3월~현재 한국전자통신연구원 연구원 관심분야는 정보검색(IR), 데이타베이스, 데이타 스트림, 센서 네트워크, 텔레매틱스

황 규 영
1973년 서울대학교 전자공학과 졸업(B.S.) 1975년 한국과학기술원 전기 및 전자학과 졸업(M.S.). 1982년 Stanford University, EE/CSL(M.S.). 1983년 Stanford University, EE/CSL(Ph.D.). 1975년~1978년 국방과학연구소(ADD), 선임연구원. 1983년~1990년 IBM T.J. Watson Research Center, Research Staff Member. 1992년~1994년 한국정보과학회 데이타베이스 연구회(SIGDB) 운영위원장. 2007년 한국정보과학회 회장. 2007년~현재 Fellow, IEEE. Coordinating Editor-in-Chief: The VLDB Journal, 2007년~현재(Editor-in-Chief, 2003년~현재). Associate Editor: IEEE Transactions on Knowledge and Data Engineering, 2002년~2005년. Associate Editor: The VLDB Journal, 1990년~2003년. Associate Editor: The IEEE Data Engineering Bulletin, 1990년~1993년. 1996년~2004년 Trustee, The VLDB Endowment. 2007년~현재 Chair: Steering Committee, DASFAA. 1999년~현재 한국과학기술원 전자전산학과 전산학전공 교수. 관심분야는 데이타베이스 시스템, 멀티미디어, 검색엔진, GIS

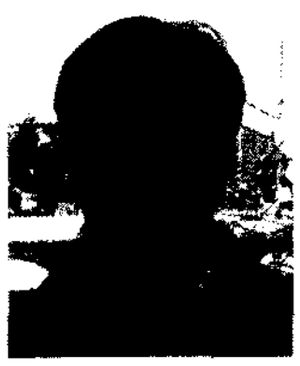


이 재 길
1993년 3월~1997년 2월 한국과학기술원 전산학과(학사). 1997년 3월~1999년 2월 한국과학기술원 전자전산학과 전산학전공(석사). 1999년 3월~2005년 2월 한국과학기술원 전자전산학과 전산학전공(박사). 2005년 3월~2006년 6월 한국과학기술원 전자전산학과 박사후연구원. 2006년 7월~현재 미국 일리노이 주립대학 전산학과 박사후연구원. 관심분야는 이동 객체 데이타 마이닝, 정보 검색 및 검색 엔진, 데이타베이스 보안

권 혁 윤
2001년 3월~2005년 2월 서울시립대학교 컴퓨터통계학과(학사). 2005년 3월~2007년 2월 KAIST 전자전산학과 전산학전공(석사). 2007년 3월~현재 KAIST 전자전산학과 전산학전공(박사). 관심분야는 대용량 정보검색, 질의 최적화, XML 질의 처리

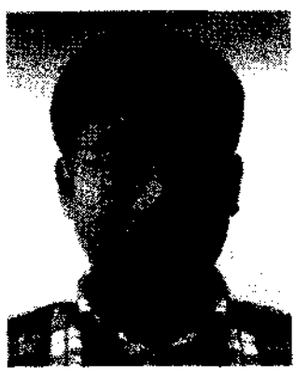
김 이 른

1999년 한국과학기술원 전자전산학과 전산학전공 학사. 2001년 한국과학기술원 전자전산학과 전산학전공 석사. 2001년~현재 한국과학기술원 전자전산학과 전산학전공 박사과정. 관심분야는 저장시스템, 임베디드 시스템



허 준 석

1991년 3월~1995년 2월 서울시립대학교 전산통계학과 학사. 1995년 3월~1997년 2월 서울시립대학교 전산통계학과 석사 1997년 2월~2002년 6월 대우통신㈜(현, ㈜머큐리) 선임연구원. 2002년 9월~2003년 2월 한국과학기술원 첨단정보기술연구센터 위촉연구원. 2003년 3월~현재 한국과학기술원 전자전산학과 전산학전공 박사과정. 관심분야는 정보검색(IR), 공간 데이타베이스, 주기억장치 데이타베이스



이 기 훈

1996년 3월~2000년 2월 KAIST 전자전산학과 전산학전공(학사). 2000년 3월~2002년 8월 KAIST 전자전산학과 전산학전공(석사). 2002년 9월~현재 KAIST 전자전산학과 전산학전공(박사). 관심분야는 XML 데이타베이스, 정보 검색, 질의 최적화