퍼지 개념 네트워크를 이용한 개인화된 링크기반 검색엔진의 개발

(Development of a Personalized Link-based Search Engine using Fuzzy Concept Network)

김 경 중 † 조 성 배 ††
(Kyung-Joong Kim) (Sung-Bae Cho)

요 약 텍스트 정보만을 이용하는 일반적인 검색엔진들의 한계를 극복하여 향상된 결과를 내기 위하여 링크 구조를 이용해 검색을 수행하는 시스템이 새롭게 등장하고 있다. 링크 구조는 사용자의 질의에 대해 중요한 문서들을 가려준다. 본 논문에서는 한 걸음 더 나아가 링크 정보를 이용하여 검색된 웹 페이지들 중 사용자의 기호에 적절한 결과를 도출하는 방법을 제안한다. 사용자 프로파일에 기반한 퍼지 개념 네트워크로 구축된 퍼지 문서 추출 시스템은 사용자의 성향을 반영하여 링크 기반 검색결과를 개인화 한다. 5명의 사용자에 대한 실험결과, 개발한 시스템이 의미 있는 웹 페이지를 검색함은 물론이고 사용자의 성향을 잘 반영함을 알 수 있었다.

Abstract New search engines using link structure will be launched to improved search result to overcome the limitation of general text-based search engines. Link structure distills the important documents about user's query. Going a step further, in this paper, we develop a system that provides user with personalized result from a search engine using link structure. Fuzzy document retrieval system constructed from fuzzy concept network based on user profile personalizes the results of link-based search engine with user's preference. Experimental results with five subjects indicate that the developed system searches not only relevant but also personalized web pages on user's preference.

1. 서 론

좋은 검색엔진을 만드는 것은 상당히 어려운 일이다. 검색엔진은 대규모의 문서를 다루어야 하기 때문에 효과적인 검색을 수행하기 위해서는 대규모의 문서를 저장할만한 공간이 필요하다. 저장장치의 가격이 낮아져서 많은 부분 해결되었지만 아직까지 큰 문제로 남아있다. 또한 문서의 수가 많다보니 웹 문서로부터 일관된 순위생성 규칙을 만드는 것도 쉬운 일이 아니다. 어떠한 질의어를 던지더라도 사용자가 원하는 문서를 가장 높은 순위로 올릴 수 있는 순위생성 방법이 필요하다. 마지막 지막으로 많은 문서를 다루다보니 속도가 느려지기 쉽다. 효과적인 인덱싱 방법과 순위생성 방법으로 적당한 시간 안에서 검색이 이루어지도록 해야 한다. 좋은 검색엔진을 만들기 위해서는 이상에서 열거한 여러 가지 요소를 함께 고려하여야 한다.

검색서비스는 웹의 중요한 서비스의 하나로 자리잡아 Yahoo, Lycos, Altavista와 같은 검색엔진들이 대표적으로 이용되고 있다[1,2,3]. 최근에는 Google 이나 Clever Search 등이 차세대 검색엔진으로 기대를 모으고 있다[4,5]. Google 과 Clever Search 모두 링크 정보를 이용한다는 공통점이 있다. 링크 정보를 바탕으로 웹 문서의 중요도를 계산하고 순위를 생성하는데, 링크 정보는 텍스트 정보만으로 찾을 수 없는 중요한 문서들을 손쉽게 찾아준다. 검색결과는 많은 사람들이 공통적으로 인정하는 대표적인 웹 문서들이다. 링크 정보를 이용한 검색엔진은 앞에서 제시한 문제중의 하나를 해결하도록 도움을 준다. Google의 경우는 링크정보를 이용

• 이 논문은 2000년도 한국학술진흥재단의 지원에 의하여 연구되었음.
† 학생회원 : 연세대학교 컴퓨터과학과
uribyul@candy.yonsei.ac.kr
†† 종신회원 : 연세대학교 컴퓨터과학과 교수
sbcho@csai.yonsei.ac.kr
논문접수 : 2001년 2월 2일
심사완료 : 2001년 4월 30일

하여 웹 문서의 중요도를 사전에 계산해 놓아 검색속도의 문제도 해결했다[6]. Clever Search는 authoritative와 hub 두 종류의 검색결과를 제공한다[5].

Authoritative와 hub문서들은 링크정보로 구해지는데, authoritative문서는 특정 주제에 대해 대표적인 문서이고, hub문서는 많은 authoritative문서를 링크하고 있는 문서이다[7].

링크 정보를 이용해서 검색을 수행하면 검색 성능을 향상시킬 수 있다[6,7]. 텍스트 정보만을 이용해서 검색을 수행하면 질의어를 많이 포함하고 있는 웹 문서를 높은 순위로 평가하게 된다. 만약 "physics"에 관해 가장 중요한 웹 문서를 찾는다면 텍스트기반 검색엔진은 "physics"를 가장 많이 포함하고 있는 문서를 선택할 것이다. 이러한 방법은 사용자가 기대했던 것과 차이가 나는 원인이 된다. 링크 정보를 이용한 방법은 가장 대표적인 문서를 찾아주기 때문에 이러한 문제를 개선한다.

본 논문에서 제안하는 시스템은 링크 정보를 기반으로 검색을 수행한다. 링크 정보를 이용하여 웹 문서를 찾기 때문에 보다 향상된 검색결과를 기대할 수 있다. 또한 사용자에게 보다 만족스러운 결과를 제시하기 위해 퍼지 개념 네트워크를 이용한 개인화작업을 수행하였다. 퍼지 개념 네트워크는 퍼지 논리를 사용하여 개념들 사이의 중요도를 계산함으로써 사용자의 지식을 표현한다[8,9,10]. 퍼지 개념 네트워크의 생성은 사용자 프로파일 정보를 기반으로 한다. 생성된 퍼지 개념 네트워크로 개인화된 문서를 추출하여 최종적으로 사용자의 의도에 가장 적합한 웹 문서를 선택한다. 퍼지 개념 네트워크와 퍼지 문서 추출 시스템은 개인화를 위한 효과적인 방법으로 사용될 수 있을 것이다.

본 논문의 나머지 부분은 다음과 같이 구성된다. 2장은 검색엔진의 현황에 대해 소개하고, 3장은 제안하는 시스템의 구성과 개인화방법에 대해 설명한다. 4장은 검색결과와 개인화 과정을 설명하고, 5장은 논문의 결론과 향후연구에 대해 언급한다.

2. 검색엔진

일반적인 검색엔진은 크로울러, 인덱서, 순위 생성기로 구성되어져 있다[11]. 크로울러는 웹 문서를 가져오는 작업을 수행하는데[12], 크로울러가 웹 문서를 방문하는 순서와 회수는 중요하다. 크로울러에서 가져온 웹 문서를 파싱하여 텍스트 정보를 추출한 후, 추출된 단어들에 대해 인덱싱을 수행한다. 사용자의 질의에 대하여 인덱스 테이블을 이용해서 질의어와 일치하는 단어를 찾고 해당 URL들을 가져온다. 가져온 URL들에 대해 검색엔진의 순위생성 알고리즘을 적용하여 순위를 생성하고 정렬하여 보여주게 된다.

순위를 생성하기 위해 일반적으로 backlink의 개수를 사용한다. 즉 backlink의 개수가 많은 웹 문서를 중요한 웹 문서로 평가한다. 하지만 backlink를 적게 받지만 중요한 문서가 있을 수 있다. 이러한 문제를 해결하기 위해 각각의 링크를 동일하게 처리하지 않는 방법이 사용된다. 즉 각각의 링크에 대해 서로 다른 비중을 부여하는 것이다. 예를 들어 Yahoo로부터의 링크와 개인 홈페이지로부터의 링크를 서로 다르게 처리하는 것이다.

검색엔진의 성능을 결정짓는 중요한 요소는 얼마나 많은 웹 문서를 크로울링하고 있는가와 얼마나 정확한 순위생성 알고리즘을 가지고 있는가라고 할 수 있다. 크로울링하고 있는 웹 문서가 많을수록 사용자가 찾는 문서를 제시할 확률이 높아진다. 하지만 아무리 좋은 검색엔진도 전체 웹의 16%이상을 포함하고 있지 못하다는 연구결과를 보면 많은 웹 문서를 포함하고 있는 것에는 한계가 있다는 것을 알 수 있다[13]. 이상과 같은 문제점에 대안이 될 수 있는 방법이 링크 정보를 이용한 검색이라고 할 수 있다. 즉 웹 문서를 크로울링하고 있지는 않지만 링크 정보를 통해 URL을 가지고 있다면 검색에 포함시키는 것이다. 웹 문서의 중요도는 링크정보를 이용하므로 텍스트 정보가 없어도 된다. 이때 문제가 되는 것이 크로울링을 수행하지 않고 검색에 이용된 URL의 broken여부를 확인할 수 없다는 점이다. 하지만 링크 정보를 이용해서 순위를 생성하는 과정에서 broken된 URL이 높은 순위를 얻기 어렵기 때문에 문제를 해결할 수 있다.

검색엔진이 좋은 성능을 내기 위해서는 많은 웹 문서와 좋은 순위생성 알고리즘이 있어야 한다고 설명했다. 하지만 대부분의 검색엔진들이 저장하고 있을 수 있는 웹 문서의 개수에는 한계가 있기 때문에 크로울링 전략이 필요하다[14]. 즉 전체 웹 문서를 가지고 올 수 없다면 일부를 가져오더라도 중요한 웹 문서를 우선적으로 가져오는 방법을 사용하는 것이다. 가장 손쉬운 방법으로 사용자의 질의어와 일치하는 내용이 많은 웹 문서를 우선적으로 크로울링하는 것이 있다. 이 방법은 사용자의 질의어에 대한 정보를 알아야 하기 때문에 off- line으로 크로울링을 수행하기 어렵다는 문제점이 있다. 다음으로 backlink의 수를 이용하는 방법이 있다. 즉 backlink의 수가 많은 웹 문서를 높은 우선순위를 두고 크로울링하는 방법이다. 이때 backlink의 수는 전체 웹 공간에서 문서를 링크하고 있는 모든 문서의 수이다. 모든 backlink를 동일하게 처리하지 않고 서로 다른 가중

치로 처리하는 방법이 PageRank기법이다[15]. Page Rank기법은 반복적인 연산을 통해 웹 문서의 Page Rank값을 계산한다. 또 다른 방법으로 Fowardlink의 개수가 많은 웹 문서를 높은 가중치로 평가하는 방법이 있다.

데이타베이스로부터 나온 검색결과를 사용자에게 보여주기 위해 사용하는 방법으로 대표적인 것이 순위를 생성하는 방법이다. 하지만 클러스터링을 수행하는 방법도 있다. Grouper라는 검색시스템은 클러스터링 방법을 이용하여 사용자에게 보여질 웹 문서를 정렬한다[16]. Northern Light의 경우도 비슷한 방법을 이용한다[17]. Northern Light의 경우는 웹 문서를 크로울링한 후 custom search folder에 넣는데, folder는 웹 사이트의 종류, 형태, 주제, 그리고 언어에 따라 생성된다. 웹 문서를 folder에 분류한 후 사용자의 질의가 들어왔을 때 결과로 보여질 folder를 결정하게 된다. Northern Light의 경우 미리 클러스터링을 수행해 놓고 사용자의 질의에 답변한다.

Google의 경우 상업화되어 큰 성공을 거두고 있는데, 현재 13억개의 웹 문서를 크로울링하고 있다고 한다. Google은 링크를 통해 획득한 URL도 검색대상에 포함하고 있다. 즉 크로울링되지 않은 웹 문서도 검색결과에 나올 수 있다. 하지만 PageRank를 통해 중요하지 않은 웹 문서에 낮은 순위를 부여함으로써 문제를 해결했다. 기본적으로 backlink의 개수에 기반한 PageRank는 많은 웹 문서 가운데서 중요한 웹 문서에 높은 순위를 할당하는 작업을 훌륭히 수행하고 있다. 다른 점은 각각의 backlink를 동일한 비중으로 보지 않고 서로 다른 비중으로 본다는 점이다. 웹 문서에 대한 가중치를 초기화한 후 링크정보를 이용하여 반복적으로 웹 문서의 가중치를 계산하여 수렴된 PageRank값을 계산한다[6].

Yahoo와 AliWeb은 크로울러를 사용하지 않는 디렉토리기반의 검색엔진이다[1,18]. 전문가를 통해 등록이 요청된 웹 문서에 대한 평가를 실시하여 인덱스에 포함할지 여부를 결정한다. 웹 문서의 등록은 웹 마스터나 웹 페이지 제작자에 의해 수동으로 수행된다. 사람에 의해 작업이 진행되기 때문에 시간이 오래 걸리고 비용도 많이 들지만 현재까지는 가장 잘 정렬된 결과를 제시하는 것으로 평가되고 있다. 하지만 PageRank와 같은 좋은 순위생성 기법이 등장하면서 사람의 부담을 덜어 줄 가능성도 커지고 있다.

Clever 검색엔진은 현재 상용으로 서비스되고 있지는 않지만 IBM에 의해 연구단계에 있는 차세대 검색엔진 중의 하나이다[5]. Clever 검색엔진은 웹 문서간의 링크 관계만을 이용하여 검색을 수행하기 때문에 텍스트 정보를 이용함으로써 발생하는 문제는 없다. IBM은 Clever 검색엔진을 웹뿐만 아니라 다양한 정보 검색 분야에서 사용하기 위해 노력하고 있다. Clever검색엔진은 검색결과를 authoritative와 hub 두 가지 종류로 나누어 제공한다. Authoritative는 웹 문서 중에서 가장 대표적으로 신뢰할 수 있는 문서를 나타낸다. Hub문서는 많은 authoritative문서를 링크하고 있는 문서를 나타낸다. Clever검색엔진의 문제점은 잘 정리된 검색결과에 비해 시간이 많이 걸린다는 점이다.

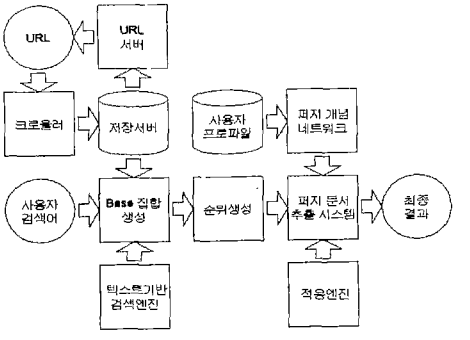

그림 1 개인용 검색엔진 시스템

3. 제안하는 검색엔진

링크 기반 검색엔진의 전체 구조는 그림 1과 같다. 링크 기반 검색엔진 시스템은 크게 크로울링, 링크구조 저장, 순위 생성 그리고 개인화부분으로 나뉜다. 순수하게 링크 정보만을 이용하여 순위를 생성하므로 저장 서버는 링크 구조를 저장한다. 크로울러는 웹 문서를 가져와 링크정보를 추출한다. URL 정보와 링크 정보는 저장 서버에게 전달되어 저장된다. 사용자 질의가 들어오면 검색엔진은 순위 생성 알고리즘을 수행한다. 순위 생성 알고리즘은 텍스트 기반 검색엔진을 이용하여 base 집합을 생성하고 authoritative와 hub문서를 찾아낸다. 개인화과정은 퍼지 개념 네트워크를 바탕으로한 퍼지 문서 추출 시스템이 담당한다. 퍼지 개념 네트워크는 사용자 프로파일의 정보를 바탕으로 사용자 별로 생성된다. 생성된 퍼지 개념 네트워크를 이용하여 퍼지 문서 추출 시스템은 중요한 문서를 찾아낸다.

3.1 크로울링

링크 정보를 추출하기 위해 크로울링 작업을 수행한다. 크로울링 작업은 시작페이지로부터 시작하여 다음과 같은 방식으로 수행된다.

1. 시작페이지의 주소를 크로울러에게 전달한다.
2. 크로울러는 웹 문서를 가져온다.
3. 웹 문서를 분석하여 링크 정보를 추출한 후 queue에 넣는다.
4. Queue로부터 URL을 꺼낸 후 크로울러에게 전달한다.
5. Queue가 빌 때까지 2, 3, 4를 반복한다.

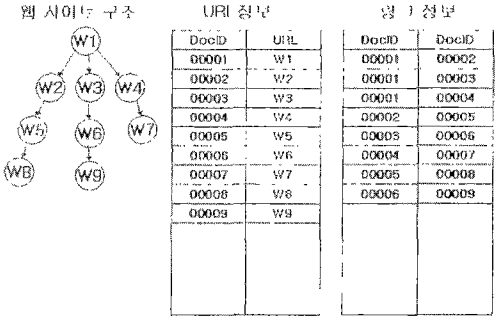

그림 2 URL과 링크정보가 저장되는 과정

3.2 링크 구조의 저장

저장서버는 URL과 링크 정보를 저장한다. URL은 유일한 *DocID*로 표현되고, 링크정보는 <*DocID*, *DocID*> 형태로 저장된다. 그림 2는 링크정보가 저장되는 과정을 보여준다. 텍스트 기반 검색엔진으로부터 가져온 200개의 웹 문서에 대해 forward link와 back link를 구하기 위해 저장 서버에 저장된 내용을 이용한다. Forward link를 구하기 위해서 링크 정보 테이블에서 링크의 시작 *DocID*가 W_1의 *DocID*와 일치하는 경우를 찾는다. 이때 링크의 끝 *DocID*가 W_1의 forward link에 해당된다. W_1의 back link도 마찬가지로 링크 정보 테이블을 이용하여 링크의 끝 *DocID*가 W_1의 *DocID*와 일치하는 경우를 찾는다. 이때 링크의 시작 *DocID*가 W_1의 back link에 해당된다.

Root집합의 forward link와 back link를 모두 찾아서 base 집합을 구성한 후 authoritative와 hub 가중치를 계산하여 순위를 생성해야 한다. 이때 authoritative와 hub가중치를 반복적으로 갱신하기 위해서 base집합에 속한 문서들 사이의 링크 정보를 파악할 필요가 있다. Base집합에 속한 문서의 URL이 각각 W_1과 W_2라고 했을 때 URL 정보 테이블을 이용하여 각각에 대한 *DocID*를 찾는다. 만약 일치하는 *DocID*쌍이 존재하면 문서 W_1과 W_2에 대한 링크가 존재하는 것이다. Base집합에 속한 문서들 사이의 링크 관계가 모두 파악되면 authoritative와 hub가중치가 수렴할 때까지 반복 연산을 수행한다. 수렴여부가 확인되면 authoritative와 hub 가중치의 크기에 따라 문서들을 정렬한다.

3.3 순위 생성

링크 정보를 이용하여 검색을 수행하기 위해서 authoritative와 hub문서를 정의한다. Authoritative문서는 가장 신뢰할 만한 정보를 가지고 있는 웹 문서를 말한다. Hub문서는 많은 authoritative 문서를 링크하고 있는 웹 문서를 말한다. 사용자의 질의어와 관련된 웹 문서집합을 구성하기 위해 텍스트 기반 검색엔진을 이용한다. 텍스트 기반 검색엔진을 통해 사용자의 질의어와 관련된 웹 문서 200개를 얻어낸다. 200개의 웹 문서를 root집합이라 한다. Root집합을 확장하기 위해 root집합에 속한 문서들의 back link와 forward link를 구한다. Root집합, root집합의 back link 그리고 root 집합의 forward link를 모두 포함하여 base집합을 구축한다. 구축된 base집합을 대상으로 authoritative와 hub문서를 구한다. 그림 3은 Base 집합을 구축하는 과정을 보여준다.

텍스트 기반 검색엔진으로부터 구한 root집합은 사용자의 질의에 대한 authoritative문서와 hub문서를 모두 포함하고 있지 못하다. Root집합에 포함되지 못한 authoritative문서와 hub문서는 base집합으로의 확장과정을 통해 모두 포함된다. Base집합에는 사용자가 찾고자하는 중요한 문서들이 모두 포함되어 있게 된다. Base집합에 속한 문서들 중에서 authoritative문서와 hub문서를 찾기 위한 반복적인 계산과정은 다음과 같다.

1. Base집합에 속한 문서 i에 대해 authoritative 가중치 a_i와 hub 가중치 h_i를 부여한다.
2. Base집합에 속한 문서 i에 대해 a_i와 h_i를 1로 초기화한다.
3. Base집합에 속한 모든 문서에 대해 1과 2를 수행한다.
4. 문서 i에 대해 a_i와 h_i를 다음과 같은 방법으로 갱신한다.
   $a_i = \sum h_j$ (j는 i를 링크하는 모든 문서들)
   $h_i = \sum a_j$ (j는 i가 링크하고 있는 모든 문서들)
5. Authoritative 가중치와 hub가중치를 정규화한다. 각각 제곱의 합이 1이 되도록 한다.
6. Authoritative와 hub가중치가 수렴할 때까지 4와 5를 반복한다.

이상과 같은 방법을 통해 계산된 base집합에 속한 문서들의 authoritative 가중치와 hub가중치를 토대로 authoritative문서와 hub문서를 찾는다[7].

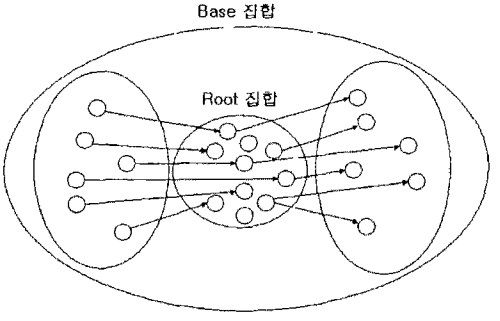

그림 3 Base 집합의 구축

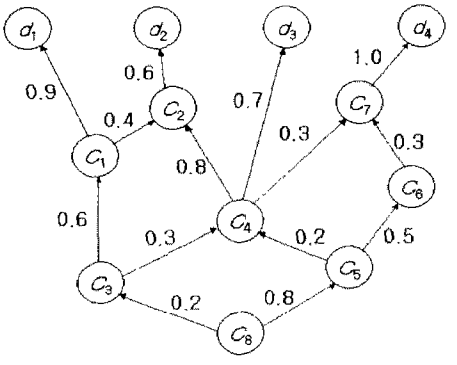

그림 4 퍼지 개념 네트워크

3.4 개인화

Lucarella는 퍼지 정보 추출시스템을 위해 퍼지 개념 네트워크를 제안했다[19]. 퍼지 개념 네트워크는 노드들과 방향성 있는 링크로 구성되어져 있다. 각각의 노드는 개념 또는 문서를 나타낸다. 각각의 방향성 있는 링크는 두 개념을 연결시키거나 개념과 문서사이의 관계를 정의한다. $C=\{C_1, C_2, \ldots, C_n\}$는 개념들의 집합을 나타낸다. $C_i \xrightarrow{\mu} C_j$는 개념 C_i에서 C_j까지의 중요도가 μ라는 표시이다. $C_i \xrightarrow{\mu} d_j$는 개념 C_i에 대한 문서 d_j의 중요도가 μ라는 표시이다. $C_i \xrightarrow{\mu} C_j$를 $f(C_i, C_j)=\mu$로 표시한다. $f(C_i, C_j)=\alpha$, $f(C_j, C_k)=\beta$일때 $f(C_i, C_k)=\min(\alpha, \beta)$로 표시된다. $C_i \xrightarrow{\mu} d_j$를 $g(C_i, d_j)=\mu$로 표시한다. 문서 d_j는 각 개념들에 대해 서로 다른 중요도 값을 가진다. 문서 d_j에 대한 문서 디스크립터는 개념집합의 퍼지 부분집합으로 정의된다. $d_j=\{(C_i, g(C_i, d_j)) \mid C_i \in C\}$ 이다. 만약 C_i에서 C_j로의 경로가 여러 가지가 있을 때 $f(C_i, C_j)$는 가장 큰 값으로 결정한다. 그림 4는 퍼지 개념 네트워크의 예를 보여준다. 그림 4에서 $f(C_3, C_2)=\max(0.4,0.3,0.2)$가 되어 0.4로 결정된다.

퍼지 개념 네트워크를 사용하여 문서 $d_1, d_2, \ldots, d_n$에 대한 문서 디스크립터를 구할 수 있다. 구해진 문서 디스크립터를 이용하면 퍼지 문서 검색을 수행할 수 있다. 개념 C_1이 정의되었다면 문서 $d_1, d_2, \ldots, d_n$중에서 C_1에 대한 중요도가 가장 큰 문서를 선택하면 된다. 이상과 같은 과정은 각 문서 $d_1, d_2, \ldots, d_n$에 대해 문서 디스크립터를 구해주어야 하기 때문에 많은 시간이 소요된다. 빠른 시간안에 개념 네트워크를 이용한 퍼지 문서 추출을 수행하기 위해 퍼지 개념 행렬이 이용된다[9].

퍼지 문서 추출 시스템은 정보검색 과정에서 발생하는 불확실성을 해결하기 위해 퍼지기법을 도입한다. 퍼지 이론은 1965년 Zadeh에 의해 제안되었으며, 불확실성을 다루기 위한 확고한 수학적인 기반을 제공한다[20, 21]. 퍼지 문서 추출 시스템은 아래와 같이 정의되며 퍼지 개념 네트워크는 지식베이스로서 이용된다[10].

$\langle H, C, Q, I, K, \phi, \psi \rangle$

H : 문서 집합

C : 개념들의 집합

Q : 질문들의 집합

I : H에서 C로의 이항 퍼지 인덱싱 관계

K : 지식 베이스

ϕ : $Q \times H \rightarrow [0,1]$, 문서 추출 함수

ψ : $H \times H \rightarrow [0,1]$, 중요도 함수

각각의 (q, h)에 대해, $q \in Q, h \in H$, $\Phi(q, h) \in [0,1]$은 추출 상태값이라고 한다. 각각의 (h_1, h_2)에 대해, $h_1, h_2 \in H$, $\psi(h_1, h_2) \in [0,1]$은 문서 h_1과 h_2 사이의 중요도라고 한다. 이항 퍼지 인덱싱 관계 I는 다음과 같은 형태로 표현된다.

$I=\{\mu_I(h, c), (h, c) \mid h \in H, c \in C\}$

소속 함수 μ_I : $H \times C \rightarrow [0,1]$, 각각의 (h, c)에 대해 개념 c가 문서 h에 대해 어느 정도로 중요한가를 나타낸다. 각각의 $h \in H$에 대해 문서 디스크립터 I_h는 C의 퍼지 부분집합으로 정의된다.

$I_h=\{(\mu_{I_h}, c) \mid c \in C, \mu_{I_h}=\mu_I(h, c)\}$

문서와 개념 사이의 중요도는 행렬 D로 표시되며, 문서 디스크립터 행렬이라고 불리운다.

$H=\{h_1, h_2, \ldots, h_m\}$이고, $C=\{c_1, c_2, \ldots, c_n\}$ 일 때 D는 아래와 같이 정의된다.

$$D=\begin{bmatrix} d_{11} & d_{12} & \cdots & d_{1n} \\ d_{21} & d_{22} & \cdots & d_{2n} \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ d_{m1} & d_{m2} & \cdots & d_{mn} \end{bmatrix}$$

이때 $d_{ij}=I_{h_i}(c_j)$, $1\le i\le m$, $1\le j\le n$로 정의된다.

$C=\{c_1, c_2, \ldots, c_n\}$ 는 개념들의 집합이다. 퍼지 개념 행렬 K는 $K_{ij}\in[0,1]$이다. K의 (i,j)원소는 개념 c_i로부터 c_j까지의 중요도를 나타낸다. $K^2=K\otimes K$는 퍼지 개념 행렬의 곱을 나타낸다.

$$K^2{}_{ij}=\bigvee_{l=1}^{n}(K_{il}\wedge K_{lj}),\ 1\le i,\ j\le n$$

여기서 $\vee$와 $\wedge$는 각각 max 연산과 min 연산을 나타낸다. 이때 $K^{\rho}=K^{\rho+1}=K^{\rho+2}=\ldots$ 를 만족하는 $\rho\le n-1$인 정수 ρ가 존재하게 된다. $K^*=K^{\rho}$ 라고 하면 K^*는 퍼지 개념 행렬 K의 전이폐쇄가 된다[9]. 각 개념에 대한 문서의 중요도는 문서 디스크립터 행렬 D와 퍼지 개념 행렬 K의 곱을 계산함으로써 향상될 수 있다[10].

$D^*=D\otimes K^*$

D^*를 확장된 문서 디스크립터 행렬이라고 부른다.

사용자 프로파일

Java	Book	0.7
Java	Car	0.3
Java	WWW	0.9
Java	Ship	0.1
Book	Car	0.3
Book	WWW	0.5
Book	Ship	0.1
Book	Cafe	0.4
Car	WWW	0.7
Car	Ship	0.6
WWW	Ship	0.5
Ship	Cafe	0.3

퍼지 개념 행렬

	Java	Book	Car	WWW	Ship	Cafe
Java	1.0	0.7	0.3	0.9	0.1	0.0
Book	0.7	1.0	0.3	0.5	0.1	0.4
Car	0.3	0.3	1.0	0.7	0.6	0.0
WWW	0.9	0.5	0.7	1.0	0.5	0.0
Ship	0.1	0.1	0.6	0.5	1.0	0.3
Cafe	0.0	0.4	0.0	0.0	0.3	1.0

그림 5 사용자 프로파일을 이용한 퍼지 개념 행렬의 구축

개인화 과정은 다음과 같이 진행된다. 상위 5개의 authoritative문서를 선택한다. 개인화 대상 문서는 5개로 한정하여, 개인화 성능을 손쉽게 평가할 수 있도록 하였다. 5개의 authoritative문서는 사용자의 질의에 대해 높은 신뢰도를 가지는 문서들이다. 5개의 문서들을 대상으로 퍼지 문서 추출 시스템을 이용하여 새로운 순위를 부여한다. 우선 5개의 문서들에 대한 문서 디스크립터를 정의한다. 5개의 문서들에 대한 문서 디스크립터는 문서에 나타난 개념의 빈도를 이용하여 평가한다. 사용자 프로파일에 사용된 n개의 개념에 대해 각 문서 마다 빈도를 측정하고 0에서 1사이의 값으로 정규화 한다.

퍼지 개념 행렬은 사용자 프로파일을 이용하여 구축된다. 사용자 프로파일은 n개의 개념들 사이의 중요도 중 일부를 표시하도록 하였다. 그림 5는 사용자 프로파일을 이용하여 퍼지 개념 행렬을 구축한 모습을 보여준다. 사용자 프로파일을 이용하여 구축된 퍼지 개념 행렬은 사용자의 관심도를 반영하게 된다. 퍼지 개념 행렬의 원소들은 사용자 프로파일에 담긴 정보만이 표시되어져 있고 표시되지 않은 정보는 0이 된다. 표시되지 않은 정보는 퍼지 개념 행렬의 전이폐쇄를 계산하여 구한다. 전이폐쇄를 통해 구한 퍼지 개념 행렬은 n개의 개념들 사이의 모든 중요도를 표시한다.

링크 정보를 이용한 검색 결과로 나온 상위 5개의 authoritative문서에 대한 문서 디스크립터와 전이폐쇄를 이용하여 구한 퍼지 개념 행렬을 곱하여 확장된 문서 디스크립터를 구한다. 확장된 문서 디스크립터를 이용하여 개인화된 문서 검색결과를 얻게 된다. 사용자 프로파일을 이용하여 구한 퍼지 개념 행렬은 사용자의 관심사를 보다 정확하게 표현하도록 조정될 수 있다. 적응 엔진은 사용자로부터 검색결과에 대한 피드백을 받아서 개인화과정에 반영하는 역할을 담당한다.

4. 실험 및 결과

사용자의 질의가 들어오면 텍스트 기반 검색엔진을 이용하여 100개 가량의 웹 문서의 주소를 얻어온다. J. Kleinberg는 200개 가량의 웹 문서를 사용했으나, 본 연구에서는 속도향상을 위해 100개의 문서를 사용했다[7]. 본 시스템에서는 Altavista를 이용한다. 사용자의 질의를 Altavista검색엔진에 전달하여 상위 100개 문서의 주소를 얻어온다. 100개의 웹 문서 주소는 root집합을 이룬다. 100개의 root집합 문서에 대해 저장서버를 통해 forward link와 back link를 구하여 base집합을 구성한다. 구성된 base집합문서에 대해 순위를 매기는 작업을 수행한다. Base집합의 크기가 커지는 것을 제한하기 위해 forward link와 back link의 수를 각각 3개와 50개로 제한하였다. Forward link의 수는 각 문서마다 앞부분에 나오는 3개를 택했다. Base집합의 크기는 대략 500개에서 1000개 사이에서 결정되었다. Authoritative와 hub가중치를 계산할 때 반복회수는 가중치의 수렴여부에 따라 제한된다. 많은

실험을 통해 5번의 반복이전에 수렴이 결정됨을 확인하였다. 본 시스템에서는 반복회수를 5번으로 설정하였다. 표 1은 링크정보를 이용하여 "Java"에 대해 검색한 결과를 보여준다.

표 1 "Java"에 대한 검색결과

순위	Authoritative결과	Hub결과
1	http://java.sun.com	http://industry.java.sun.com/products
2	http://www.javalobby.org	http://java.sun.com/industry
3	http://javaboutique.internet.com	http://java.sun.com/casestudies
4	http://java.about.com/compute/java/mbody.htm	http://industry.java.sun.com/javanews/developer
5	http://www.javaworld.com	http://industry.java.sun.com/jug

"Java"에 관해 가장 높은 신뢰도를 가지는 "java.sun.com" 사이트를 authoritative결과 1위로 설정하였다. 또한 "www.javalobby.org," "javaboutique.internet.com," "java.about.com/compute/java/mbody.htm," "www.javaworld.com" 등 "Java"와 관련된 신뢰도 높은 사이트를 authoritative 결과로 결정했다. "Java"이외에도 "Java"와 관련된 다양한 키워드에 대해 실험을 수행해 보았다. 표 2는 "Java"와 관련된 다양한 키워드에 대한 실험결과를 보여준다.

링크 정보를 이용하여 얻은 결과를 이용하여 개인화 과정을 수행한다. 사용자 프로파일은 총 10개의 개념을 포함하고 있다. 각각은 "Book," "Computer," "Java," "Internet," "Corba," "Network," "Software," "Unix," "Family," "Newspaper" 등이다. 10개의 개념 사이의 중요도 가운데 20개를 사용자 프로파일에 입력하도록 하였다. 입력되지 않은 정보는 퍼지 개념 행렬의 전이폐쇄를 이용하여 계산한다. 사용자 5명에 대한 프로파일을 이용하였다. 그림 6은 사용자 프로파일 정보를 기초로 작성된 퍼지 개념 네트워크의 예이다. 사용자의 퍼지 개념 네트워크는 사용자 프로파일에 담긴 20개의 정보만을 표시해 주고 있다. 표시되지 않은 개념들 사이의 중요도는 퍼지 논리를 사용하여 추론하여야 한다. 퍼지 개념 네트워크만을 이용하여 개념들간의 모든 중요도를 결정하는 것은 시간이 많이 걸린다. 이러한 단점을 해결하기 위해 퍼지 개념 행렬을 이용하여 사용자의 프로파일을 표현하였다. 사용자 5명으로부터 링크 정보를 이용하여 얻은 "Java"에 대한 검색결과를 기호에 맞게 순위를 매기도록 하였다. 표 3은 사용자가 매긴 "Java"검색결과에 대한 순위이다. 표 4는 개인화 된 검색결과이다. 음영 처리된 부분이 사용자가 매긴 순위와 일치한 부분이다.

표 2 "Java"와 관련된 키워드에 대한 검색결과

키워드	Authoritative검색결과
"Java2"	1 java.sun.com 2 www.appserver-zone.com 3 www.sun.com/service/sunps/jdc/java2.html 4 jdc.sun.co.jp 5 java.sun.com/products/jdk/1.2 6 www.javalobby.org
"Jdk"	1 java.sun.com 2 developer.netscape.com/software/jdk/download.html 3 java.sun.com/products/jdk/1.1/docs/index.html 4 www.ora.com/info/java 5 kbs.cs.tu-berlin.de/~jutta/ht/JDK-beta2-quickref.html 6 www.quick.com.au/java
"Jguru"	1 java.sun.com 2 www.magelang.com 3 www.javaworld.com 4 java.sun.com/products/javamail/index.html 5 developer.java.sun.com 6 www.mindspring.com/~scdrye/java/faq.html
"Jini"	1 www.jini.org 2 java.sun.com 3 www.artima.com 4 archives.java.sun.com/archives/jini-users.html 5 www.sun.com/jini/news/artcliprev.html 6 www.prosyst.com
"Servlet"	1 java.sun.com 2 www.servletcentral.com 3 java.sun.com/products/servlet/index.html 4 archives.java.sun.com/archives/servlet-interest.html 5 www.webmacro.org 6 courses.coreservlets.com
"Javaone"	1 java.sun.com 2 www.togethersoft.com 3 www.javacats.com 4 www.zdevents.com 5 www.washington.edu/bibsys/mattf/javaone

표 3 사용자가 매긴 순위

	사용자 1	사용자 2	사용자 3	사용자 4	사용자 5
java.sun.com	1	1	1	2	1
www.javalobby.org	2	2	2	1	2
javaboutique .internet.com	4	4	5	4	3
java.about.com/compute/java/mbody.htm	3	5	3	3	4
www.javaworld.com	5	3	4	5	5

표 4 개인화 된 검색결과(음영처리부분은 사용자의 순위와 일치한 부분)

	사용자 1	사용자 2	사용자 3	사용자 4	사용자 5
java.sun.com	2	1	2	1	2
www.javalobby.org	1	2	1	3	1
javaboutique. internet.com	3	3	3	2	3
java.about.com/compute /java/mbody.htm	4	4	5	5	4
www.javaworld.com	5	5	4	4	5

사용자 1은 "Java"에 대해 중요한 웹 페이지를 찾기 원한다. 제안한 검색엔진은 authoritative 가중치에 따라 정렬된 "Java"에 관한 5개의 문서를 제공한다. 이 문서들은 순위대로 $h1$, $h2$, $h3$, $h4$, 그리고 $h5$로 대표된다. 사용자 프로파일에서 사용된 10개의 개념은 아래와 같이 표현된다.

$C=\{C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8, C_9, C_{10}\}$

각각은 "Book," "Computer," "Java," "Internet," "Corba," "Network," "Software," "Unix," "Family," "Newspaper" 등이다.

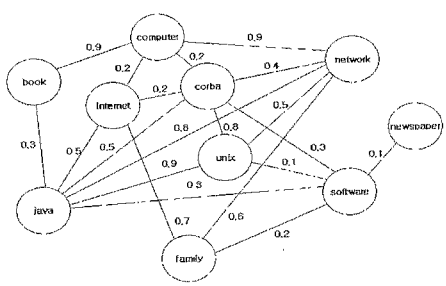

그림 6 사용자 1의 퍼지 개념 네트워크

사용자 1의 사용자 프로파일 정보로부터 아래와 같은 퍼지 개념 행렬을 형성한다. 초기의 퍼지 개념 행렬은 프로파일에 포함된 정보만을 담고 있기 때문에 행렬의 일부 원소가 0으로 표시되어져 있다.

$$K = \begin{matrix} C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \\ C_6 \\ C_7 \\ C_8 \\ C_9 \\ C_{10} \end{matrix} \begin{bmatrix} 1.0 & 0.9 & 0.3 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.9 & 1.0 & 0.0 & 0.2 & 0.2 & 0.9 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.3 & 0.0 & 1.0 & 0.5 & 0.5 & 0.8 & 0.3 & 0.9 & 0.0 & 0.0 \\ 0.0 & 0.2 & 0.5 & 1.0 & 0.2 & 0.0 & 0.0 & 0.0 & 0.7 & 0.0 \\ 0.0 & 0.2 & 0.5 & 0.2 & 1.0 & 0.4 & 0.3 & 0.8 & 0.0 & 0.0 \\ 0.0 & 0.9 & 0.8 & 0.0 & 0.4 & 1.0 & 0.0 & 0.5 & 0.6 & 0.0 \\ 0.0 & 0.0 & 0.3 & 0.0 & 0.3 & 0.0 & 1.0 & 0.1 & 0.2 & 0.1 \\ 0.0 & 0.0 & 0.9 & 0.0 & 0.8 & 0.5 & 0.1 & 1.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.7 & 0.0 & 0.6 & 0.2 & 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.1 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$

사용자 프로파일을 통해 생성된 퍼지 개념 행렬의 전이폐쇄는 아래와 같다. 퍼지 개념행렬의 곱은 퍼지 논리를 이용하여 수행된다. 퍼지 개념 행렬의 전이폐쇄를 계산하여 사용자 프로파일에 표시되지 않은 개념들 사이의 중요도를 계산하였다.

$$K^* = \begin{matrix} C_1 \\ C_2 \\ C_3 \\ C_4 \\ C_5 \\ C_6 \\ C_7 \\ C_8 \\ C_9 \\ C_{10} \end{matrix} \begin{bmatrix} 1.0 & 0.9 & 0.8 & 0.6 & 0.8 & 0.9 & 0.3 & 0.8 & 0.6 & 0.1 \\ 0.9 & 1.0 & 0.8 & 0.6 & 0.8 & 0.9 & 0.3 & 0.8 & 0.6 & 0.1 \\ 0.8 & 0.8 & 1.0 & 0.6 & 0.8 & 0.8 & 0.3 & 0.9 & 0.6 & 0.1 \\ 0.6 & 0.6 & 0.6 & 1.0 & 0.6 & 0.6 & 0.3 & 0.6 & 0.7 & 0.1 \\ 0.8 & 0.8 & 0.8 & 0.6 & 1.0 & 0.8 & 0.3 & 0.8 & 0.6 & 0.1 \\ 0.9 & 0.9 & 0.8 & 0.6 & 0.8 & 1.0 & 0.3 & 0.8 & 0.6 & 0.1 \\ 0.3 & 0.3 & 0.3 & 0.3 & 0.3 & 0.3 & 1.0 & 0.3 & 0.3 & 0.1 \\ 0.8 & 0.8 & 0.9 & 0.6 & 0.8 & 0.8 & 0.3 & 1.0 & 0.6 & 0.1 \\ 0.6 & 0.6 & 0.6 & 0.7 & 0.6 & 0.6 & 0.3 & 0.6 & 1.0 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 1.0 \end{bmatrix}$$

"Java"에 대한 링크기반 검색엔진의 상위 5개의 authoritative문서에 대한 디스크립터 행렬은 아래와 같다

$$D = \begin{matrix} h1 \\ h2 \\ h3 \\ h4 \\ h5 \end{matrix} \begin{bmatrix} 0.0 & 0.0 & 0.2 & 0.0 & 0.0 & 0.4 & 0.4 & 0.0 & 0.0 & 0.0 \\ 0.3 & 0.0 & 0.5 & 0.2 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.2 & 0.5 & 0.0 & 0.1 & 0.1 & 0.0 & 0.1 & 0.0 \\ 0.4 & 0.2 & 0.0 & 0.1 & 0.0 & 0.1 & 0.1 & 0.0 & 0.0 & 0.0 \\ 0.3 & 0.1 & 0.2 & 0.0 & 0.0 & 0.2 & 0.2 & 0.0 & 0.0 & 0.0 \end{bmatrix}$$

퍼지 개념 행렬의 전이폐쇄와 문서 디스크립터 행렬의 곱은 아래와 같다. 곱은 퍼지 논리를 이용하여 계산된다. 확장된 문서 디스크립터는 아래와 같다.

$$D^* = \begin{matrix} h1 \\ h2 \\ h3 \\ h4 \\ h5 \end{matrix} \begin{bmatrix} 0.2 & 0.4 & 0.4 & 0.2 & 0.4 & 0.4 & 0.4 & 0.4 & 0.4 & 0.1 \\ 0.3 & 0.3 & 0.5 & 0.5 & 0.5 & 0.5 & 0.3 & 0.5 & 0.2 & 0.0 \\ 0.2 & 0.2 & 0.5 & 0.5 & 0.2 & 0.2 & 0.2 & 0.2 & 0.5 & 0.1 \\ 0.4 & 0.4 & 0.3 & 0.2 & 0.2 & 0.2 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0.3 & 0.3 & 0.3 & 0.2 & 0.2 & 0.2 & 0.2 & 0.2 & 0.2 & 0.1 \end{bmatrix}$$

각각에 대한 문서 중요도는 아래와 같다. 아래의 중요도를 이용하여 나온 결과는 표 4에 나와 있다.

$$R = \begin{matrix} h1 \\ h2 \\ h3 \\ h4 \\ h5 \end{matrix} \begin{bmatrix} 3.3 \\ 3.6 \\ 2.8 \\ 2.2 \\ 2.2 \end{bmatrix}$$

5개의 authoritative 문서들은 $h2$, $h1$, $h3$, $h4$, $h5$로 재정렬된다. 제안한 검색엔진 시스템은 $h5$의 순위를 정확하게 맞추었다.

5. 결론 및 향후연구

사용자의 관심사에 적합한 정보를 찾기 위해 링크 정보와 퍼지 개념 네트워크를 사용하여 검색 시스템을 구성해 보았다. 링크 정보를 이용하여 많은 웹 문서 가운데서 중요한 문서를 찾고 퍼지 개념 네트워크를 이용하여 사용자의 개념이 반영된 문서 추출을 수행하였다. 전체 검색 시스템은 사용자가 관심 있어할 만한 중요한 문서들을 찾아 주었으며 사용자의 관심도를 반영하여 순위를 바꾸었다. 사용자의 피드백을 반영하여 퍼지 개념 네트워크를 조정해 나가는 작업을 수행한다면 좀 더 만족스러운 결과를 얻을 수 있을 것이다.

참 고 문 헌

[1] Yahoo, http://www.yahoo.com.

[2] Lycos, http://www.lycos.com.

[3] Altavista, http://www.altavista.com.

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

[5] The Clever Project, http://www.almaden.ibm.com/cs/k53/clever.html.

[6] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *The Sevenh International WWW Conference*, 1998, http://www7.scu.edu.au/programme/fullpapers/1921/com1921.htm.

[7] J. Kleinberg, "Authoritative sources in a hyperlinked environment," *IBM Research Report RJ 10076*, 1997.

[8] S.-M. Chen and Y.-J. Horng, "Fuzzy query processing for document retrieval based on extended fuzzy concept networks," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 29, no. 1, pp. 96-104, 1999.

[9] S.-M. Chen and J.-Y. Wang, "Document retrieval using knowledge-based fuzzy information retrieval techniques," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 25, no. 5, pp. 793-803, 1995.

[10] C.-S. Chang and A.L.P. Chen, "Supporting conceptual and neighborhood queries on the world wide web," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 28, no. 2, pp. 300-308, 1998.

[11] L. Introna and H. Nissenbaum, "Defining the web: The politics of search engines," *IEEE Computer*, vol. 33, pp. 54-62, 2000.

[12] B. Pinkerton, "Finding what people want: Experiences with the webcrawler," *The Second International WWW Conference*, Chicago, USA, 1994, http://www.thinkpink.com/bp/WebCrawler/WWW94.html.

[13] S. Lawrence and C.L. Giles, "Accessibility of information on the web," *Nature*, No. 400, pp. 107-109, 1999.

[14] J. Cho, H. Garcia-Molina, and L. Page, "Efficient crawling through URL ordering," *The Seventh International WWW Conference*, 1998, http://www7.scu.edu.au/programme/fullpapers/1919/com1919.htm.

[15] L. Page, "PageRank: Bringing order to the web," *Stanford Digital Libraries Working Paper 1997-0072*, 1997.

[16] O. Zamir and O. Etzioni, "Grouper: A dynamic clustering interface to web search results," *The Eighth International WWW Conference*, Toronto, Canada, 1999.

[17] Northern Light Search, http://www.northernlight.com.

[18] AliWeb, http://www.aliweb.com.

[19] D. Lucarella and R. Morara, "FIRST: Fuzzy information retrieval system," *Journal of Information Science*, vol. 17, no.2, pp. 81-91, 1991.

[20] L.A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, pp. 338-353, 1965.

[21] L.A. Zadeh, "Fuzzy sets as a basis for a theory of possibility," *Fuzzy Sets and Systems*, vol. 1, no. 1, pp. 3-28, 1978.

김 경 중

2000년 2월 연세대학교 컴퓨터과학과 졸업(학사). 2000년 3월 ~ 연세대학교 컴퓨터과학과 석사과정 재학중. 관심분야는 인공지능, 진화연산, 검색엔진, 이동로봇 제어.

조 성 배

1988년 연세대학교 전산과학과(학사). 1990년 한국과학기술원 전산학과(석사). 1993년 한국과학기술원 전산학과(박사). 1993년 ~ 1995년 일본 ATR 인간정보통신연구소 객원 연구원. 1998년 호주 Univ. of New South Wales 초청연구원. 1995년 ~ 현재 연세대학교 컴퓨터과학과 부교수. 관심분야는 신경망, 패턴인식, 지능정보처리.