DOI: 10.3745/KIPSTD.2009.16-D.3.423

적응형 웹 사이트 구축을 위한 연관규칙 알고리즘 개발과 적용

최 윤 희[†] · 전 우 천[††]

요 약

컴퓨터의 보급과 인터넷의 발달로 인해 데이터의 유통은 증가하고 있으나 전통적인 방법으로는 가치 있고 의미 있는 정보를 획득하는 것은 어렵다. 또한, 정보화 사회에서의 많은 정보 중에서 자신에게 알맞은 정보를 탐사하는 데이터 마이닝의 필요성이 대두되고 있다. 또한 사용자들의 편리한 인터넷 항해를 돕고 적절한 정보를 제공할 수 있는 적응형 웹 사이트에 관한 연구도 필요하다.

본 연구의 목적은 사용자들에게 연관성이 있는 웹 페이지를 연결해 주는 적응형 웹 사이트 구축을 위해 웹 로그 분석을 통한 웹 사이트 사용자들의 행동 패턴을 발견하는 연관규칙 알고리즘의 개발에 있다. 데이터 마이닝의 기법 중에서 연관규칙은 웹 사이트에 접속하는 사용자들의 행동을 파악하는데 효과적이다. 본 논문에서는 웹 사용 마이닝을 이용하여 웹 서버의 로그 데이터를 분석하여 트랜잭션을 구성하고, 사용자들의 행동 패턴을 발견하기 위한 의미 있는 문서만을 추출하여 추출된 문서를 대상으로 발견한 빈발 항목으로 연결리스트를 구성하며, 빈발 패턴을 찾아 웹 페이지에 적용하는 일련의 알고리즘을 제안한다. 제안한 알고리즘의 특징은 첫째, 빈발패턴 발견을 위해 생성하는 연결리스트 이외에는 마이닝 과정에서 다른 중간생성물이 필요하지 않으므로 공간 사용면에 있어 효율적이다. 둘째, 기존의 연관규칙 알고리즘에 비해 데이터베이스의 스캔 횟수를 줄이고, 시간복잡도를 개선하였다.

키워드 : 연관규칙, 적응형 웹 사이트

Development and Application of An Adaptive Web Site Construction Algorithm

Yunhee Choi[†] · Woochun Jun[††]

ABSTRACT

Advances in information and communication technologies are changing our society greatly. In knowledge-based society, information can be obtained easily via communication tools such as web and e-mail. However, obtaining right and up-to-date information is difficult in spite of overflowing information.

The concept of adaptive web site has been initiated recently. The purpose of the site is to provide information only users want out of tons of data gathered. In this paper, an algorithm is developed for adaptive web site construction. The proposed algorithm is based on association rules that are major principle in adaptive web site construction.

The algorithm is constructed by analysing log data in web server and extracting meaning documents through finding behavior patterns of users. The proposed algorithm has the following characteristics. First, it is superior to existing algorithms using association rules in time complexity. Its superiority is proved theoretically. Second, the proposed algorithm is effective in space complexity. This is due to that it does not need any intermediate products except a linked list that is essential for finding frequent item sets.

Keywords : Association Rules, Adaptive Web Site

1. 서 론

정보화 사회에서는 얼마나 많은 정보를 가지고 있느냐가 아니라 얼마나 자신에게 맞는 정보를 가지고 있느냐가 더 중요하다. 하지만 전통적인 방법으로는 방대한 데이터로부터 가치 있는 정보를 획득하는 작업이 쉽지 않기 때문에 데이터의 양은 증가하고 있으나 정보의 양은 오히려 감소하고 있다. 이로 인해 불필요한 데이터를 삭제하거나 적절한 방법으로 정보를 탐사해야 할 전문적인 정보 시스템의 필요성이 대두되고 있다. 데이터베이스에 쌓인 데이터에서 잘 알려져 있지 않고 추출하기 어려운 정보를 발견해나가는 과정

† 정 회 원 : 서울오류남초등학교 교사
†† 정 회 원 : 서울교육대학교 컴퓨터교육과 교수
논문접수 : 2009년 1월 16일
심사완료 : 2009년 1월 31일

을 데이터 마이닝 (Data Mining)이라고 한다[8]. 웹 서버에 축적된 대용량의 로그데이터에 데이터 마이닝의 기술을 적용하여 웹 사이트 사용자들의 행동을 분석하고 필요한 정보를 발견하고 해석하는 과정을 웹 마이닝이라고 한다[2]. 이런 과정을 통하여 발견된 정보들은 웹 사이트 사용자들의 편리한 항해를 돕고 적당한 정보를 제공하기 위한 적응형 웹 사이트 (Adaptive Web Site)의 기본 자료가 된다. 적응형 웹 사이트란 방문객들의 접근 패턴을 학습하여 구조나 외형을 자동적으로 개선시켜 나가는 웹 사이트로 방문객들이 탐색 과정에서 소요하는 시간과 노력을 줄여 주면서 좀 더 효율적으로 웹 정보를 이용하는 것을 가능하게 한다[6].

본 연구의 목적은 웹 사이트 관리자의 부담을 줄이면서도 사용자들의 다양한 요구에 반응할 수 있도록 웹 사이트를 자동적으로 갱신하고 원하는 문서로 쉽게 이동할 수 있는 적응형 웹 사이트를 구축하기 위한 알고리즘을 개발하는 것이다.

본 논문의 구성은 다음과 같다. 2장에서는 데이터 마이닝과 웹 마이닝, 연관규칙에 관한 이론적 배경을 살펴보고, 3장에서는 연관규칙을 이용하여 빈발문서를 추출하고 빈발 패턴을 발견하는 알고리즘을 제시한다. 4장에서는 추출된 빈발 패턴을 학교 웹 사이트에 적용하는 일련의 과정을 설명하며, 5장에서 결론 및 향후 연구 과제를 제시한다.

2. 이론적 배경

2.1 데이터 마이닝과 웹 마이닝

데이터 마이닝은 데이터에서 유효하고, 귀하고, 잠재적으로 유용하고, 궁극적으로 이해될 수 있는 패턴을 알아내는 지식 탐사 과정의 핵심에 위치하는 한 단계로써 존재하며, 관찰된 데이터로부터 패턴이나 모델을 추출하는 과정을 뜻한다[13].

데이터 마이닝을 통해 얻을 수 있는 정보는 매우 다양하며, 얻고자 하는 결과나 데이터의 상태 등에 따라 적용할 수 있는 다양한 기법들이 존재하고, 데이터에서 더 많은 정보를 얻어내는데 도움을 주는 것은 어떤 방법이라도 유용하다. 적절한 데이터 마이닝 기법을 선정하여 적용하는 과정은 지식 발견 과정에서 가장 중요한 단계이며, 일반적으로 사용되는 방법들은 의사결정 나무, 신경망, 군집화, 연관규칙, 사례기반추론, 유전자 알고리즘 등이 있다.

웹 마이닝은 분산된 정보 환경에서 상호 작용 접근을 촉진시키기 위해 서로 링크되어 있는 웹 문서로부터 자동적으로 정보를 찾을 때 사용된다. 웹 마이닝은 분석 대상의 유형에 따라 웹 구조 마이닝 (Web Structure Mining), 웹 내용 마이닝 (Web Content Mining), 웹 사용 마이닝 (Web Usage Mining)으로 구분할 수 있다. 웹 구조 마이닝은 웹 내용을 기술하는데 사용하는 구조화된 정보를 분석하는 과정으로서 하이퍼텍스트로 구성된 문서들의 구조에 대해 마이닝하는 것이다. 웹 내용 마이닝은 웹 사이트를 구성하는 페이지 내용 중에서 텍스트, 이미지, 오디오, 동영상 등과 같은 다양한 데이터로부터 얻어지는 내용에 대한 분석을 통하여 웹 사이트에 대한 유용한 정보를 찾아내는 과정이다. 한편 웹 사용 마이닝은 웹 서버에 저장된 웹 로그 파일을 이용하여 사용자들의 행동 패턴에 대한 정보를 분석하는 과정으로 웹 사이트에서 사용자들의 웹 페이지 사용 패턴을 분석하고, 사용자가 웹 서핑하면서 발생되는 로그 데이터와 사용자가 직접 작성한 등록 정보 등에 의해 얻어지는 데이터를 사용하여 수행한다. 이런 방법으로 얻어진 웹 사용자들의 항해 행동에 대한 지식은 앞으로의 행동을 예측하는데 사용된다. 웹 마이닝은 웹 상에서 발생하는 데이터에 대한 실시간 분석이 진행돼야 하기 때문에 분석 횟수가 많아지고, 웹 로그 데이터 분석과 기존 오프라인의 데이터 분석이 함께 이루어져야 정확한 정보를 제공할 수 있다[2, 4, 5].

CLF (Common Logfile Format) 형식의 엑세스 로그 파일에는 방문객의 IP 주소, Authuser, 웹 문서에 접근한 날짜와 시간, 요청 방법, 접근한 문서의 URL, 접속 상태와 이동 상태의 현황, 전송 바이트 수 등에 관한 정보가 기록된다. 로그 파일은 사용자의 웹 탐색에 관한 자세한 정보를 담고 있지만 로그 데이터만 가지고 연관성을 발견하는 것에는 어려움이 있다. 웹 사용 마이닝을 위해서는 로그 데이터를 사용하기 쉬운 형태로 가공하는 전처리 단계와 트랜잭션 데이터에서 다양한 패턴 발견 기법들을 이용하여 사용자의 접속 패턴을 찾아내는 패턴 탐색 단계, 유용하고 의미 있는 규칙과 패턴을 이해하고 해석하는 패턴 분석 단계를 거쳐야 한다.

2.2 연관규칙

연관규칙 X→Y에서 신뢰도 (Confidence) C는 X를 포함하는 트랜잭션 중에서 Y를 포함하는 트랜잭션의 비율로 정의된다. 신뢰도는 연관규칙에 대한 정확도를 측정할 수 있는 지표로 정확한 예측을 가능하게 해 준다. 지지도 (Support) S는 전체 트랜잭션에서 X와 Y가 함께 포함된 트랜잭션의 비율로써, 빈번하게 발생하는 패턴의 유용성이 증대되려면 지지도가 커야 한다. 트랜잭션 집합에서 연관규칙을 찾는 문제는 사용자가 정의한 최소 지지도와 최소 신뢰도보다 큰 지지도와 신뢰도를 갖는 항목간의 연관성을 찾는 것을 말한다.

연관규칙을 탐사하는 문제는 트랜잭션 데이터베이스에서 최소 지지도 이상을 만족하는 빈발항목을 찾아내는 첫 번째 과정과 빈발항목에서 최소 신뢰도 이상을 만족하는 연관규칙을 찾아내는 두 번째 과정으로 나뉜다. 마이닝의 전체적인 성능은 많은 노력과 비용이 수반되는 첫 번째 단계에서 주로 결정된다. 거대한 데이터베이스 분석을 통해 빈발항목을 효율적으로 찾는 것이 찾아진 빈발항목에서 연관규칙을 생성하는 것보다 어렵기 때문에 일반적으로 빈발항목을 찾는 문제에 대한 연구가 많이 이루어지고 있다. 빈발항목집합을 찾는 과정은 일반적으로 두 가지 방법으로 접근할 수 있다. 첫 번째는 후보항목을 생성하고 그 빈발도를 테스트하는 Apriori 알고리즘과 Apriori를 기반으로 하는 변형된 방법들이다. 두 번째는 패턴을 성장시켜 가는 방법으로 트리 기반의 FP-Growth 알고리즘과 배열 기반의 H-mine 알고리즘이 있다[14]. 두 가지 방법의 장점과 단점을 비교하면 <표 1>과 같다[10].

〈표 1〉 빈발항목 탐색 방법의 비교

방법	후보항목을 생성하고 확인하는 방법	빈발 패턴을 성장시키는 방법
장점	1. 트랜잭션을 저장할 필요가 없다. 2. 알고리즘이 단순하다.	1. 후보항목집합을 생성할 필요가 없다. 2. 입출력이 적다.
단점	1. 후보항목을 생성한다. 2. 입출력이 많다.	1. 트랜잭션을 저장할 구조를 만들어야 한다. 2. 알고리즘이 복잡하다.

2.2.1 Apriori 알고리즘

Apriori 알고리즘은 데이터베이스를 스캔하여 각 항목들의 지지도를 계산한 후 최소지지도 이상을 만족하는 빈발항목집합을 생성한 후 전단계 빈발항목집합으로부터 후보항목집합을 만들고, 각 단계마다 데이터베이스 스캔을 통해 지지도를 계산하여 빈발항목집합을 확정한다[11]. 이러한 Apriori 알고리즘에는 두 가지 문제점이 있다. 첫째, 반복적인 데이터베이스 스캔과 대규모 후보 집합에 대한 패턴 매칭 검사가 필요하다. 특히 빈발항목들이 아닌 항목이나 현재 후보보다 더 작은 항목을 포함한 트랜잭션들이 알고리즘의 후반부에서는 더 이상 필요 없음에도 불구하고 항상 매 패스마다 전체 데이터셋을 스캔한다. 둘째, 상당한 크기의 후보항목집합들의 생성이 필요할 수 있으므로, 방대한 크기의 메모리 공간이 필요하고, 후보항목에 대한 검색 및 갱신에 많은 시간이 소모된다. 빈발항목집합의 원소수가 증가할수록 데이터베이스의 스캔 횟수가 증가하고, 계산의 복잡도가 지수적으로 증가한다[15].

Apriori의 단점을 보완하기 위한 Apriori 기반의 DIC (Dynamic Itemset Counting), 분할 (Partition) 알고리즘, 샘플링 (Sampling) 알고리즘, DHP (Direct Hashing and Pruning)와 같은 알고리즘들은 데이터베이스의 스캔과 많은 후보항목을 생성하는 점은 부분적으로 보완했다. 하지만 공통적으로 많은 수의 후보항목집합들을 생성해야 하고 각 후보항목에 대해 빈발항목여부를 판단해야한다. 주된 비용은 데이터 스캔 횟수보다는 방대한 크기의 메모리 공간을 필요로 하는 후보항목집합을 생성하는데 있으며, 후보항목에 대한 검색 및 갱신에 많은 시간이 소모된다[9].

2.2.2 FP-growth 알고리즘

FP-growth 알고리즘은 빈발항목집합을 구하기 위해 후보를 생성하지 않으며, 두 번의 데이터베이스 스캔으로 FP-tree라는 자료구조를 구축하고 그것을 통해 빈발항목집합을 구한다[9]. FP-tree는 각 트랜잭션을 구성하고 있는 항목들이 유사할 경우에는 공유되는 부분이 많아져서 메모리 요구량이 적어지고, 트리를 구성하는데 단지 2회의 데이터베이스 스캔이 필요하며 추가적인 데이터베이스 스캔을 요구하지 않는다는 장점이 있다. 하지만 트랜잭션 간의 항목의 공유가 적은 희소 데이터 집합에서는 FP-tree의 메모리 요구량이 트랜잭션 데이터베이스의 크기에 근접하게 되어 방대한 메모리 공간이 필요하게 된다. 또한, 빈발항목 집합을 탐색하는 과정에서 동적으로 메모리 공간에 생성하는 중간 결과물의 크기도 커지게 되고, 수행 시간이 긴 대형 어플리케이션에서는 FP-tree와 같은 핵심 데이터 구조나 중간 결과물들이 메인 메모리에 상주하기가 적합하지 않고 쉽게 수행이 중단되는 원인이 된다[3].

2.2.3 H-mine 알고리즘

H-mine 알고리즘은 FP-tree 기반의 문제점을 개선하기 위하여 고안된 하이퍼구조를 사용한 메모리 기반의 빈발패턴 방식 알고리즘이다. 매우 작은 공간 오버헤드를 통한 빠른 마이닝을 위해 설계되었고, 데이터베이스의 환경에 따라 밀집 데이터일 때는 압축률이 뛰어난 FP-growth 알고리즘을 사용하고 희소 데이터일 때는 H-struct를 구축하여 마이닝을 수행한다[7, 9]. H-mine은 밀집 데이터나 희소 데이터일 경우 모두 Apriori나 FP-growth보다 실행시간이 단축되고 공간 사용이 효율적이며, 대용량의 데이터에도 잘 수행된다. 그러나 입력 데이터가 밀집 상태인지 희소 상태인지를 구분할 명확한 기준 제시가 어려우며, H-struct를 사용할 경우 사용하지 않는 링크 공간으로 인한 저장 공간의 낭비가 발생하고 압축 효과를 기대하기 어렵다. 또한 빈발항목집합 발견 과정 시 큐를 사용하기 때문에 해당 항목에 대한 링크가 부족할 경우 백트래킹 (Backtracking)을 해야 하는 문제점이 있다[16, 17].

2.3 관련 연구

빈발패턴을 발견하기 위한 연관규칙 알고리즘 개발에 관한 연구는 다음과 같다.

[5]에서는 빈발항목 탐색을 위하여 Matrix 알고리즘을 제안하였다. 우선 전체 트랜잭션 데이터베이스를 스캔하여 Matrix 형태로 변환한 후 각 항목들의 지지도를 계산하여 최소지지도 이상의 트랜잭션으로 구성된 후보 Matrix를 생성한다. 이후 같은 항목 수를 갖는 트랜잭션들을 선별하여 k-partition Matrix를 구성하고 최대항목부터 (k-1)-partition Matrix까지 빈발항목패턴을 찾는 과정을 반복한다. 이 알고리즘은 전체 트랜잭션 데이터베이스의 스캔 횟수는 1번이지만 빈발항목 탐사 과정에 있어 트랜잭션이 포함하는 최대항목수 만큼의 후보 Matrix를 생성해야 하는 문제가 있다.

[7]에서는 트랜잭션 내에서의 발생빈도는 적지만 의미 있는 항목들간의 패턴을 발견하기 위해 다중항목지지도 (Multiple Item Supports)의 개념을 이용한 변형된 H-mine 알고리즘을 제안하였다. 의사결정자의 중요 특성치를 반영한 다중항목지지도를 도출하여 각 항목들을 오름차순으로 정렬한 MISH-struct 메모리 구조와 빈발항목집합인 MISH-list를 생성한다. 이후 MISH-list의 순서대로 항목-헤더 테이블 MISH을 만들고 항목-큐를 순회하며 빈발항목을 탐색한 뒤 발견된 빈발항목과 헤더 테이블을 연결한다. 이 알고리즘은 H-mine의 특성을 살린 빠른 수행속도를 보이나 사용되지 않는 링크로 인한 공간의 낭비와 탐색 자료가 불충분할 경우 백트래킹을 해야 하는 문제가 있다.

[12]에서는 Prefix Tree와 배열구조를 이용한 순환적인 제거 알고리즘을 제안하였다. 데이터베이스를 스캔하여 빈발

문서를 추출하고 오름차순으로 정렬 한 후 각 트랜잭션 데이터베이스의 첫 항목을 헤더테이블의 인덱스로 사용하는 간단한 배열 구조를 생성한다. 가장 적은 지지도를 갖는 항목부터 마이닝이 진행되며, 각 단계별로 인덱스에 연결된 트랜잭션의 수가 달라지기 때문에 지지도가 변한다. 이 알고리즘은 트랜잭션 내의 항목들이 재배치되기 때문에 특별한 메모리 관리가 필요하고, 마이닝의 과정이 복잡하고 Prefix Tree라는 복잡한 데이터 구조를 사용하는 문제가 있다.

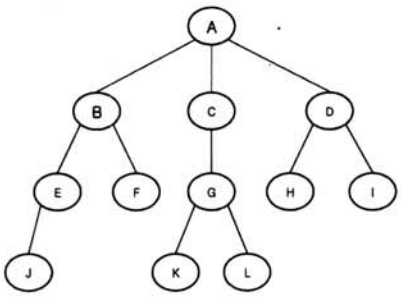
[6]에서는 학교 웹 사이트 방문자들의 접근 패턴 추출 알고리즘과 이를 통한 적응형 학교 웹 사이트 구현 방안을 제시하였다. 연관성 높은 페이지 목록 추출을 위해 액세스 로그 파일 전처리하여 사용자들이 최종적으로 접근하고자 했던 문서만을 추출하고, 로그 파일 정보를 FWDP-tree로 메모리에 재구성하였다. 이 Tree 구조를 바탕으로 한 FWDP-mine 알고리즘을 통해 최대 길이를 가지는 빈발패턴을 출력하여 어떤 문서와 연관성이 있는 모든 문서를 제시한다. 그러나 FP-tree 구조에 기반을 둔 FWDP-tree는 밀집 데이터에는 압축 효과로 인해 효과적이지만 희소 데이터 환경에서는 트리의 크기가 트랜잭션 크기에 근접하게 되어 압축 효과가 거의 없고, 메모리 공간의 사용량이 많아진다.

이상의 연구에서 볼 때 연관규칙 알고리즘에 관한 연구는 점점 성능을 향상시켜가는 방향으로 진행되고 있음을 알 수 있다. 따라서 연관규칙 알고리즘 개발은 많은 비용을 요구하는 데이터베이스의 스캔 횟수를 줄이고 후보항목 집합을 생성하지 않으면서, 빠른 계산으로 밀집 환경이나 희소 환경의 모든 형태의 자료에 적용이 가능하도록 성능을 향상시켜 가는 방향으로 진행되어야 할 것이다.

3. 연관규칙 탐사 알고리즘

3.1 트랜잭션 재구성 알고리즘

(그림 1)은 웹 문서의 연결이나 방향성을 고려하지 않았을 때의 웹 사이트의 전체적인 구조를 예시한 것이다. A는 웹 사이트의 첫 화면을 나타내며 각각의 문서들은 계층적으로 서로 연결되어 있다. 이 경우 하위 문서로 가기 위해서는 계층적으로 찾아가는 방법과 첫 페이지에 제시되는 새로운 내용을 클릭함으로써 이동하는 방법, 그리고 마우스를 상위 메뉴에 올려놓았을 때 나타나는 메뉴를 이용하는 방법 등이 있다. 하이퍼링크가 잘 연결된 경우에는 중간 경로가 생략되거나 단축될 수도 있다.

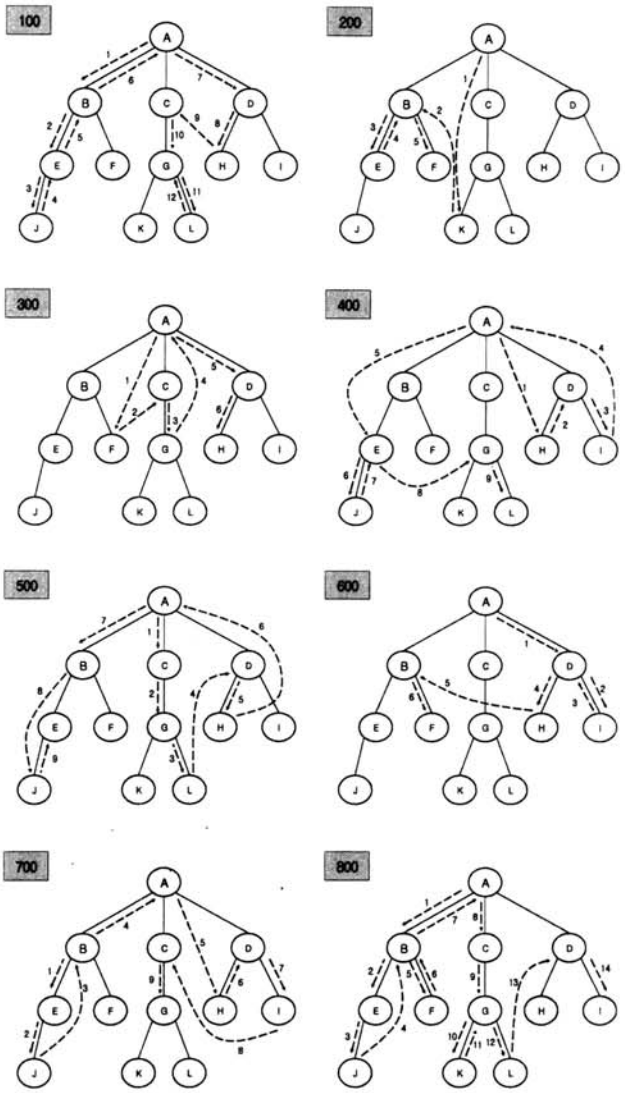


(그림 1) 웹 사이트의 구조 예시

(그림 2)는 방문자의 운행 경로를 예시한 것이며, <표 2>는 (그림 2)의 운행 경로를 트랜잭션 데이터베이스 (TDB)에 저장한 예이다.

대부분의 웹 사이트는 네트워크형으로 연결되어 있기는 하지만 계층적 구조로도 되어 있기 때문에 사용자가 원하는 문서로 이동하기 위해서는 중간 단계를 거쳐야 할 경우가 있다. 이로 인해 상위 문서는 빈발 패턴이 될 확률이 높고, 동일한 문서가 한 트랜잭션에서 반복되어 나타나기도 하므로 사용자가 원하는 유효한 문서를 식별하는 것이 부정확해지기 때문에 트랜잭션을 재구성할 필요가 있다.

트랜잭션 재구성은 다음과 같은 사항을 고려한다. 첫째, 사용자가 원하는 문서는 웹 사이트에서 제공하는 자료의 형태가 문서나 그림 형식이기 때문에 일반적으로 하위에 위치하게 된다. (그림 2)에서 사용자가 하위 문서인 'F, H, I, J, K, L'을 열어보기 위해 중간 문서를 경유하였다면 전체적인 운행 경로는 'A→D→H'처럼 하위 문서를 향해 진행하는 형태이거나 'A→B→E→J→E→B→A'처럼 대칭형을 이루게 된다.



(그림 2) 웹 사이트 운행 경로 예시

〈표 2〉 웹 문서 운행경로 트랜잭션 데이터베이스 예제

TID	Items
100	A B E J E B A D H C G L G
200	A K B E B F
300	A B C G A D H
400	A H D I A E J E G L
500	A C G L D H A B J E
600	A D I D H B F
700	B E J B A H D I C G
800	A B E J B F B A C G K G L D I

이런 경우 해당 하위 문서의 중간 문서들은 단지 마지막 문서를 보기 위해 경유하기 위한 문서로 간주하여 하위 문서만을 선택한다. 또한, 경로를 건너뛰어 부모 노드와의 직접적인 연결 없이 하위 문서로 연결된 경우에도 하위 문서를 선택한다. 둘째, 보통 게시판 형태의 동적 컨텐츠는 새로운 자료의 업데이트 여부에 따라 목록이 유의미한 경우가 있다. 이 경우 'A→B→E'처럼 하위 문서까지 진행하지 않고 다른 경로로 이동하게 되므로 게시판의 목록에 해당되는 부분을 기준 문서로 설정하여 하위 문서를 열어보지 않은 경우 기준 문서를 선택한다. 셋째, 각 운행 경로의 마지막 문서는 사용자가 그 세션에서 원하는 마지막 문서이므로 경로에 상관없이 선택한다. 넷째, 동일한 문서가 반복되거나 선택된 문서가 중복되는 경우 한 번만 선택한다.

트랜잭션 데이터베이스를 구성하기 위한 알고리즘은 (그림 3)과 같다. 각 트랜잭션 데이터 시퀀스 중에서 하위 문서와 기준 문서를 설정하여 하위 문서를 선택하고, 기준 문서가 발견되면 기준 문서 이후의 문서 종류에 따라 기능을 수행한다. D_k가 기준 문서일 때 D_{k+1}이 하위 문서라면 D_{k+1}을 저장하고, D_{k+1}이 하위 문서가 아니라면 D_k를 저장하며, 그 외의 경우는 데이터 시퀀스를 계속 탐색하다가 가장 마지막 문서를 저장하고 TID의 탐색을 마친다. 트랜잭션 데이터베이스가 재구성되면 각 트랜잭션에 존재하는 항목들을 내림차순으로 정렬하여 Item-list를 만든다. 그리고 각 트랜잭션을 Item-list의 순서대로 정렬한다.

```
Input : source DB
Output : ordered TDB, Item-list
Method
 Step 1 : Recompose TDB
  Select standard documents of Homepage, S;
  Store terminal node of Homepage, T;
  For each transaction TID in DB do {
    For (k=0; k<n; k++) do {
                //n is the number of TID data
      Compare web document Dk with S;
      If Dk is in S
        then { Compare Dk+1 with S;
        If Dk+1 is in S
          then { Store Dk in TDB;
               }
        else If Dk+1 is not in S
          then { Compare Dk+1 with T;
            If Dk+1 is in T
              then { Store Dk+1 in TDB;
                   }
               }
        If Dk is in T
          then { Store Dk in TDB;
               }
         }
     Store Dn in TDB;
   }

 Step 2 : order Item-list
   Create Item-list;
   Store item in Item-list without repetition item;
   Order Item-list;

 Step 3 : order TID
   For each transaction TID in TDB do {
       Order transaction data in sequence of Item-list;
   }
```

(그림 3) 트랜잭션 재구성 알고리즘

(그림 1)의 웹 사이트 구조도에서는 하위 문서의 전 문서인 'D, E, G'가 기준 문서가 된다. <표 2>의 TID 200과 같이 'A → K → B → E → B → F'로 운행한 경우 앞부분부터 순차적으로 탐색하여 하위 문서인 K를 선택하고, 기준 문서인 E 뒤의 문서 B가 하위 문서가 아니므로 E를 선택하며, 가장 마지막에 열어본 문서인 F를 선택한 후 탐색을 마친다.

<표 3>은 (그림 3)의 알고리즘을 이용하여 트랜잭션 데이터베이스를 재구성한 것이다. (그림 2) 예제의 경우 Item-list = {E, F, G, H, I, J, K, L}이고, 트랜잭션 데이터베이스를 내림차순으로 정렬한 것이 <표 4>이다.

〈표 3〉 재구성된 트랜잭션 데이터베이스

TID	Items	(Recomposed) Items
100	A B E J E B A D H C G L G	J H L G
200	A K B E B F	K E F
300	A B C G A D H	F G H
400	A H D I A E J E G L	H I J L
500	A C G L D H A B J E	L H J E
600	A D I D H B F	I H F
700	B E J B A H D I C G	J H I G
800	A B E J B F B A C G K G L D I	J F K L I

〈표 4〉 정렬된 트랜잭션 데이터베이스

TID	(Recomposed) Items	(Ordered) Items
100	J H L G	G H J L
200	K E F	E F K
300	F G H	F G H
400	H I J L	H I J L
500	L H J E	E H J L
600	I H F	F H I
700	J H I G	G H I J
800	J F K L I	F I J K L

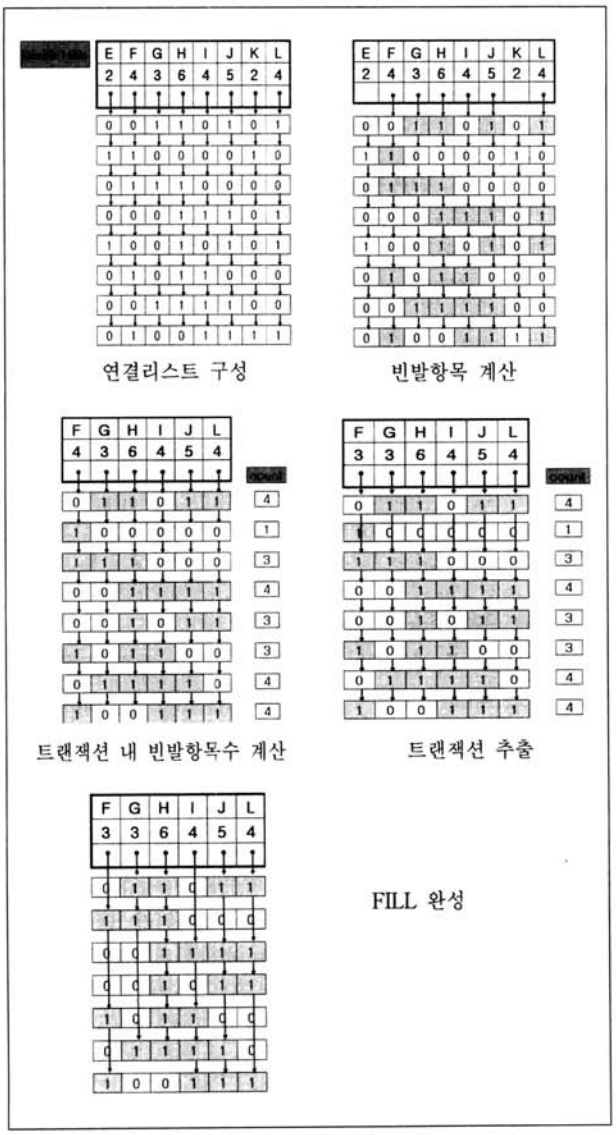
3.2 빈발 문서 추출 알고리즘

트랜잭션의 재구성과 각 트랜잭션 항목들의 정렬을 통해 전처리된 트랜잭션 데이터베이스를 구축한 이후에는 Item-list를 이용하여 각 항목을 연결리스트로 저장한 FILL (Frequent Item Linked List)를 생성한다. 첫 번째 단계에서는 Item-list의 item, 지지도, 링크로 구성된 헤더테이블을 생성하고 TDB를 스캔하여 각 트랜잭션 데이터와 Item-list를 비교하여 각 TID내의 Item-list 항목들의 존재 여부를 비트맵 형태로 기록하여 각 항목의 지지도를 계산한다. 두 번째 단계에서는 최소지지도 미만의 항목을 삭제하여 빈발항목으로만 구성된 리스트를 생성한다. 세 번째 단계에서는 각 TID 내의 빈발항목의 개수를 카운트하여 빈발항목이 1개 이하인 TID를 삭제한다. 연관성 발견을 위해서는 최소 2개 이상의 항목들이 필요하므로 트랜잭션에 남아있는 항목의 수가 2 미만일 경우는 패턴을 발견할 가능성이 없으므로 리스트에서 제외

하고 빈발 패턴 발견이 가능한 빈발항목으로만 구성된 연결리스트 FILL을 최종 완성한다.

(그림 4)는 최소지지도를 3이라고 가정했을 때 <표 4>의 정렬된 트랜잭션 데이터베이스를 이용하여 FILL을 생성하는 과정이다. Item-list = {E, F, G, H, I, J, K, L}의 헤더 테이블을 생성하고, 각 TID의 항목들을 헤더 테이블의 항목과 비교하여 TID 내에 있으면 1로, 없으면 0의 값을 갖는 연결리스트로 표현한다. 각 항목의 지지도를 계산하여 최소지지도 3 미만인 'E, G, K' 항목을 제외한 빈발항목 리스트를 생성하였다. 다음 단계로 각 TID 내의 항목 수를 계산하여 빈발 패턴을 발견할 수 없는 1개의 빈발항목만을 갖는 TID 200을 삭제하고 최종 FILL을 완성하였다. (그림 5)는 FILL 생성 알고리즘이다.

연결리스트를 사용하여 링크로 인한 기억 공간이 추가적으로 필요하지만 비트맵 형태로 데이터를 표시하였기 때문에 링크를 포함해도 전체 저장 공간이 절약되며, 데이터의 삭제와 노드의 재배열이 용이하다는 장점이 있다.



(그림 4) 빈발 문서 추출을 위한 FILL 생성 예제

```
Input : TDB, Item-list, Minsup
Output : FILL
Method
  Step 1 : generate linked list
    Create Header table with Item-list;
    Scan TDB;
    For each transaction TID in TDB do {
        For (k=0; k<n; k++) do {
                    //n is the number of Item-list
          write binary code according to existence;
        }
    Link the same item between header table and TID;
    Count the number of true;
                    //count support
    }

  Step 2 : find frequent item
    Compare support with Minsup;
                //minsup is minimum support
    Delete link of item lower than Minsup in header table;
    Delete item lower than Minsup in TID;

  Step 3 : select TID and complete FILL
    Count the number of TID item;
    Delete link of TID lower than 2;
    Connect true link with the exception of false;
```

(그림 5) FILL 생성 알고리즘

3.3 빈발 패턴 마이닝 알고리즘

FILL이 구축되면 (그림 6)의 알고리즘을 이용해 빈발 패턴을 마이닝하게 된다. k-빈발항목집합의 모든 부분집합은 (k-1)-빈발항목집합이 된다는 Apriori 알고리즘의 특징에 의해 최대 길이의 빈발 패턴을 발견하면 그 부분집합은 모두 빈발 패턴이 된다. 본 연구에서 제안하는 알고리즘은 이 특징을 기반으로 최대 길이의 빈발패턴 발견을 목적으로 한다.

전체적으로 보면 빈발항목 k를 포함하는 k-FILL과 k-헤더 테이블이 생성되고, 최소지지도 이상을 만족하는 지역빈발항목과 빈발패턴의 발견이 가능한 빈발항목수 2 이상의 트랜잭션을 추출하여 마이닝을 시작한다. 이 때 기준 항목 k와 같은 지지도를 가지는 지역빈발항목은 기준 항목과 항상 함께 나타나는 빈발패턴이므로 별도로 저장한다.

마이닝의 세부 과정은 다음과 같다. 완성된 k-FILL 내에서 각 트랜잭션별로 카운트 된 지역빈발항목들의 수를 내림차순으로 정렬하여 최소지지도 이상을 만족하는 수 c(Minsup ≤c<1)를 찾는다. c의 값 이상을 포함하는 해당 트랜잭션내

```
Input : FILL, Minsup
Output : frequent pattern
Method
  For each frequent item Ak(1<k<n) in Item-list do {
    Creat local Header table;
    Count support of local frequent items;
    Exclude items under Minsup;
    Count Ct(1<t<n-k) the number of frequent items of each
      transaction;
    Exclude transaction under value 2;
    Get c(the number of descending Ct and count) over Minsup;
    Call function find_pattern in order to find frequent item
      pattern Pk from c to 2 of Ct;
    Store frequent pattern Pk;
    }

function find_pattern (Ak-FILL(Ct,si<t), minsup)
    {
    Creat Ak local Header table and count support Ak local
      frequent items;
    Exclude items under minsup and transaction the value of Ct
      under c;
    Scan local FILL;
    Store candidate frequent pattern and increase the count of
      overlapped pattern exception of a subset of already detected
      frequent patterns;
    }
```

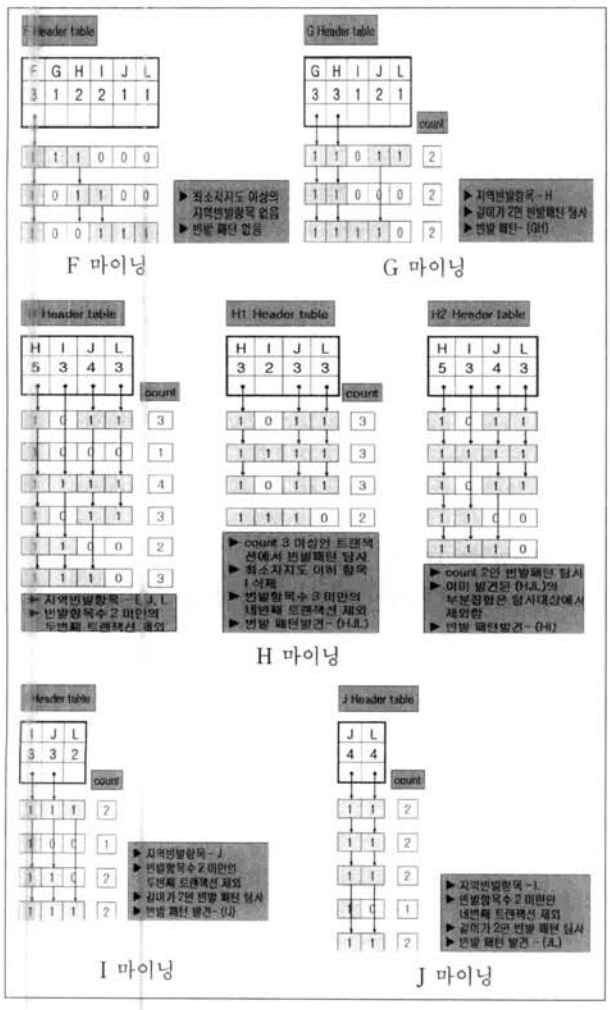
(그림 6) FILL-mine 알고리즘

의 빈발항목 중 최소지지도를 만족하는 항목에 한해 순차적으로 탐사하여 후보빈발패턴을 저장해가며 최종 빈발패턴을 찾는다. 이후 c의 값을 하나씩 줄여가며 이 과정을 반복하고 이미 빈발 패턴이 발견된 빈발항목들의 부분집합은 이후의 탐사에서 제외한다.

(그림 7)은 (그림 4)에서 생성된 FILL를 마이닝하는 과정으로 각 단계는 다음과 같다.

- 기준 항목이 F인 경우: 최소지지도 이상을 만족하는 항목이 없으므로 빈발패턴 탐사 종료
- 기준 항목이 G인 경우: 지역빈발항목 H 발견 → 빈발항목수 2 인 트랜잭션의 빈발항목 탐사 → 빈발패턴 GH 저장
- 기준 항목이 H인 경우: 지역빈발항목 I, J, L 발견 → 트랜잭션 내의 지역빈발항목수 계산 → 빈발항목수 2 미만의 트랜잭션 제외 → 최소지지도 이상을 만족하는 트랜잭션내의 빈발항목수 3 → 3 항목 이상의 트랜잭션의 빈발항목 탐사 → 최소지지도 이하 항목 삭제 → 빈발항목수 3 미만의 트랜잭션 제외 → 빈발패턴 HJL 저장 → 2 항목 이상의 트랜잭션의 빈발항목 탐사 → 이미 발견된 빈발항목 HJL의 부분집합인 HJ, HL, JL은 탐사 대상에서 제외 → 빈발패턴 HI 저장
- 기준 항목이 I인 경우: 지역빈발항목 J 발견 → 트랜잭션내의 빈발항목수 2 미만의 트랜잭션 제외 → 최소지지도 이상을 만족하는 트랜잭션내의 지역빈발항목수 2 → 빈발패턴 IJ 저장
- 기준 항목이 J인 경우: 지역빈발항목 L 발견 → 트랜잭션내의 빈발항목수 2 미만의 트랜잭션 제외 → 최소지지도 이상을 만족하는 트랜잭션내의 지역빈발항목수 2 → 후보빈발패턴 JL 발견 → 기준항목 H의 빈발패턴의 부분집합이므로 탐사 종료

마이닝 과정을 마친 후 최종적으로 발견된 빈발패턴은 GH, HJL, HI, IJ, JL이며, 이는 적응형 웹 사이트의 기본 자료가 된다.



(그림 7) 빈발패턴 마이닝 예제

3.4 알고리즘 분석

3.4.1 제안된 알고리즘의 성능 평가

[정리] n이 빈발항목의 수일 때, FILL 알고리즘의 시간복잡도는 O(n)이다.

[증명] 빈발항목 집합으로 구성된 FILL의 헤더테이블 항목을 $F=\{A_1, A_2, \cdots, A_n\}$이라고 할 때 어떤 빈발문서 A_k의 빈발패턴 P_k를 구하는 경우의 시간복잡도는 다음과 같이 계산할 수 있다. <표 5>는 FILL_mine 알고리즘의 시간복잡도를 구하기 위한 변수를 나타낸 것이다.

A_k의 빈발패턴을 찾기 위해서는 A_k를 포함하는 트랜잭션을 선별하여 지역빈발항목 집합 $F_{Ak}=\{A_{k1}, A_{k2}, \cdots, A_{kn}\}$을 먼저 찾아내고 선별된 트랜잭션들의 빈발항목 수 C_t를 계산하게 된다. 동일한 C_t의 값을 누적하고 누적된 수를 최대값부터 카운트하여 최소지지도 Minsup 이상을 만족하는 값 c를 찾는다. 빈발패턴 마이닝은 c부터 시작하여 C_t가 2가 될 때까지 반복되므로 기준 항목 A_k에 대해 총 (c-1)번 이루어진다.

기준 항목을 포함하는 트랜잭션의 수를 T_k라고 할 때 FILL-mine은 각 단계별로 트랜잭션을 순차적으로 읽어서 발견되는 패턴의 개수만을 증가시켜 가기 때문에 모든 경우를 탐사해야 하는 최악의 경우라도 시간복잡도는 $\{(c-1)T_k\}n=O(n)$이 된다.

〈표 5〉 시간복잡도를 구하기 위한 변수들

FILL	빈발패턴 탐사를 위한 연결리스트
Minsup	최소지지도
A_k	빈발 패턴을 구하고자 하는 기준 항목
T_k	기준 항목을 포함하는 트랜잭션의 수
C_t	트랜잭션내의 빈발항목 수
c	빈발항목 수를 내림차순으로 정렬하여 카운트했을 때 최소지지도 이상의 값
P_k	발견된 빈발패턴

3.4.2 다른 연관규칙 알고리즘과의 비교

본 논문에서 제안한 FILL-mine 알고리즘과 기존에 연구되었던 연관규칙에 관한 중요한 알고리즘을 비교하면 다음과 같다. <표 6>은 각 알고리즘을 비교하기 위한 변수를 나타낸 것이다.

Apriori 알고리즘은 빈발패턴탐사에 있어 빈발항목의 수가 n이라고 할 때 첫 번째 단계에서 n(n+1)/2개의 후보항목을 생성하며, 이후 단계에서도 후보항목이 계속 생성된다.

〈표 6〉 알고리즘을 비교하기 위한 변수들

C_n	최대빈발항목의 길이
n	빈발항목 수
Minsup	최소지지도
A_k	빈발패턴을 구하고자 하는 항목
F_k	A_k의 지역빈발항목집합
A_m	빈발항목을 포함하는 전체 노드의 수
m	최소지지도를 만족하는 노드의 수
T_k	빈발항목을 포함하는 트랜잭션의 수

또한, 최대빈발항목의 길이 C_n에 대해 (C_n+1)만큼의 트랜잭션 데이터베이스 스캔이 필요하다. 실제 실행 시간은 최소지지도 Minsup에 의존하게 되지만 최악의 경우 시간복잡도는 $O(2^n)$이 된다.

FP-growth 알고리즘은 두 번의 트랜잭션 데이터베이스 스캔으로 빈발항목을 구성하고 FP-tree를 생성한다. 생성된 FP-tree상에서 어떤 항목 A_k에 대한 빈발항목을 구하기 위해서는 A_k를 포함하는 모든 경로수의 합 중에서 최소지지도를 만족하는 노드의 수를 m이라고 할 때 단일 경로만을 갖는 가장 좋은 경우에도 시간복잡도는 $O(n\times(2^m-1))$이 된다. m의 값에 따라 기하급수적으로 늘어나게 된다.

H-mine 알고리즘은 두 번의 트랜잭션 데이터베이스 스캔으로 빈발항목을 구성하고 H-struct을 생성한다. 생성된 H-struct상에서 어떤 항목 A_k에 대한 빈발항목을 구하기 위해서는 전단계에서 연결된 노드를 탐색하게 된다. 항목 A_k의 지역빈발항목 F_k와 빈발항목을 포함하는 노드의 수 A_m에 영향을 받게 되며, 정보가 부족한 경우에는 백트래킹을 하므로 전체 트랜잭션의 수 T_k와도 관련이 있어서 최악의 경우 시간복잡도는 $(T_kF_k)n=O(n)$이 된다.

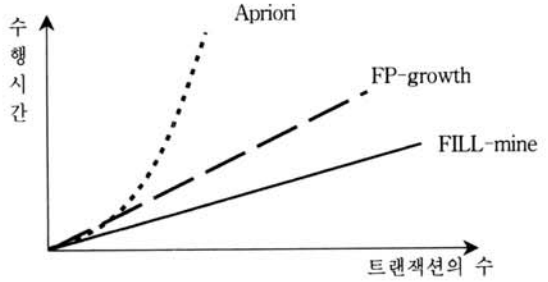
본 논문에서 제안한 FILL-mine 알고리즘은 한번의 TDB 스캔으로 빈발항목 탐사와 FILL를 구성할 수 있으며, 최악의 경우 시간복잡도는 O(n)이 된다.

위에서 살펴본 내용을 정리하면 <표 7>과 같으며, (그림 8)은 알고리즘의 복잡도를 비교한 그래프이다.

〈표 7〉 다양한 연관규칙 알고리즘의 비교

	Apriori	FP-growth	H-mine	FILL-mine
데이터 스캔 횟수	C_n+1	2	2	1
후보항목 생성	Y	N	N	N
시간 복잡도	$O(2^n)$	$O(n\times(2^m-1))^{1)}$	$O(n)^{1)}$	O(n)

1) 최상의 경우 시간 복잡도



(그림 8) 알고리즘의 복잡도 비교

4. 연관규칙 알고리즘의 적용

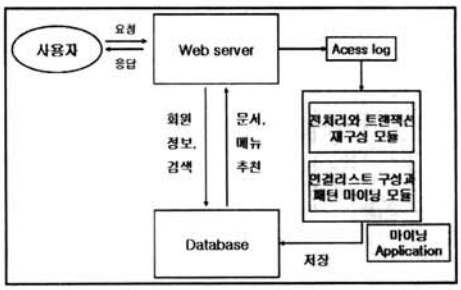
4.1 시스템 구성

적응형 웹 사이트 구성은 전처리와 트랜잭션 재구성, 연결리스트 구성과 패턴 마이닝, 웹 사이트 적용의 세 단계로 진행되며, 각 단계별로 실행 모듈을 두어 처리할 수 있게 하였다. 전체 과정을 도식화하면 (그림 9)와 같다.

사용자가 웹 서버에 접속하여 어떤 문서를 요청하면 웹 서버는 회원 가입 시에 저장한 사용자의 정보와 함께 사용자가 요청한 문서와 연관된 빈발문서가 있는지 데이터베이스에 질의한다. 마이닝 Application에 의해 저장된 웹 페이지 목록이 데이터베이스에 저장되어 있다면, 웹 서버는 사용자가 요청한 문서에 개인 정보와 연관되어 사용할 수 있는 특정한 페이지와 연관성이 높은 문서를 지정된 위치에 삽입하여 함께 제공한다.

마이닝 어플리케이션은 두 개의 모듈로 구성된다.

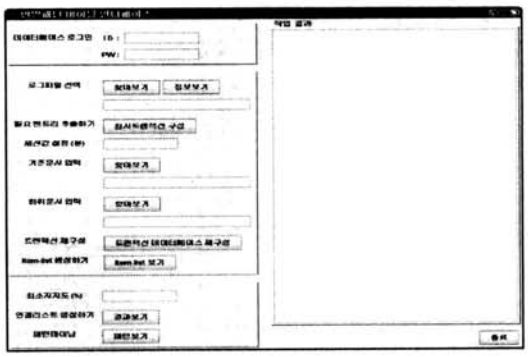
- 전처리와 트랜잭션 재구성 모듈: 원시 엑세스 로그 파일에서 마이닝에 필요한 데이터만을 추출하는 과정을 수행한다. 또한, 추출된 항목 중에서 사용자의 행동 패턴을 분석할 수 있는 부분만을 선별하는 트랜잭션 재구성을 수행한다.
- 연결리스트 구성과 패턴 마이닝 모듈: 전처리 모듈을 통해 완성된 트랜잭션 데이터베이스를 스캔하여 연결리스트를 구성하여 빈발 패턴을 마이닝하고 데이터베이스에 저장한다.



(그림 9) 적응형 학교 웹 사이트 시스템 구성도

4.2 적응형 웹 사이트 구축 과정

(그림 10)은 빈발패턴을 탐사하기 위한 인터페이스이다.



(그림 10) 빈발패턴탐사 인터페이스

데이터베이스 로그인, 전처리 과정, 연결리스트 구성 및 마이닝의 세 과정으로 나뉘어 작업이 진행되며, 단계별로 진행 결과가 오른쪽 작업 결과 화면에 나타난다.

4.2.1 전처리와 트랜잭션 재구성 과정

전처리와 트랜잭션 재구성 과정에서는 사용자가 직접 요청한 웹 문서를 추출하여 세션을 구분하여 개별 사용자별로 요청한 웹 문서를 저장한 후 트랜잭션 데이터베이스를 구성하는 작업이 이루어진다. 전체적으로는 로그파일의 선택, 엔트리 추출, 트랜잭션 재구성, Item-list 생성의 네 단계로 나뉘어 진행된다.

첫 번째 단계는 로그 파일을 선택하는 단계이다. '찾아보기' 버튼을 이용하면 해당 로그 파일을 선택할 수 있고, '정보보기' 버튼을 누르면 선택한 로그파일의 이름과 경로, 로그 파일의 크기, 총 라인 수의 정보가 작업 결과 화면에 표시된다.

두 번째 단계는 필요 엔트리를 추출하는 단계로써 선택된 로그 파일에서 사용자가 직접 요청한 파일을 선별하여 필요한 항목을 추출하고 데이터베이스에 저장하는 작업이다. 이 단계에서는 빈발 패턴 발견에 의미 있는 웹 문서만을 추출하고 방문자의 IP 주소, 방문 시간, 요청한 문서의 URL, 파일 이름을 구분하여 데이터베이스에 저장하는 작업이 이루어진다. 사용자 구분을 위해서는 IP 주소를 이용하며 서로 다른 IP 주소는 서로 다른 사용자라고 가정한다. 원시트랜잭션 구성이 끝나면 작업창에 결과가 표시된다.

세 번째 단계는 원시트랜잭션 데이터베이스를 사용자별로 정렬하고 지정된 세션값에 따라 트랜잭션을 구분하여 의미있는 문서만을 추출하는 트랜잭션 데이터베이스 구축 단계이다. 첫 번째 과정으로 IP 주소값에 따라 사용자를 구분하고 설정된 세션값에 따라 트랜잭션을 구분하여 각 사용자들이 요청한 문서를 정렬한다. 세션값에 의해 접속을 시작하여 마칠 때까지를 하나의 트랜잭션으로 구분하므로 IP 주소값의 수보다 트랜잭션의 수는 많아지게 된다. 두 번째 과정에서는 기준 문서와 하위 문서를 설정하여 트랜잭션내의 웹 문서 중에서 웹 사이트의 구조로 인해 접근하게 되는 의미없는 문서들을 제거하는 작업이 이루어진다. 이 작업으로 선별된 문서들은 데이터베이스에 저장된다. (그림 11)은 30분의 세션값을 적용하였을 때의 트랜잭션 데이터베이스 재구성 작업까지 완료된 결과 화면이다.

(그림 11) 트랜잭션 재구성 작업 결과 화면

네 번째 단계는 재구성된 트랜잭션 데이터베이스 중에서 연결리스트를 구성하기 위한 Item-list를 생성하는 단계이다. 트랜잭션 데이터베이스의 재구성이 끝나면 트랜잭션 내에 존재하는 문서들의 집합을 생성하여 내림차순으로 정렬한다. 이렇게 구성된 트랜잭션 내에 존재하는 웹 문서들의 집합을 Item-list라고 한다.

4.2.2 연결리스트 구성과 패턴 마이닝 과정

연결리스트 구성과 패턴 마이닝 과정은 최대빈발패턴 탐사를 위한 중간구조물인 연결리스트 생성과 빈발패턴을 탐사하는 패턴 마이닝의 두 단계로 진행된다.

첫 번째 단계에서는 트랜잭션 데이터베이스 재구성 과정에서 생성된 Item-list를 헤더 테이블로 하여 연결리스트를 구성하는 작업이 이루어진다. 최소지지도를 입력하면 데이터베이스 스캔을 통해 Item-list 중에서 빈발항목이 추출되고, 빈발패턴을 발견할 수 있는 트랜잭션들만을 선별하여 연결리스트를 완성한다. (그림 12)는 Item-list생성 후 연결리스트를 구성한 작업 결과 화면이다.

두 번째 단계에서는 생성된 연결리스트를 이용하여 빈발항목을 단계적으로 마이닝하여, 발견된 빈발패턴을 데이터베이스에 저장한다.

(그림 12) 연결리스트 구성 작업 결과 화면

4.2.3 웹 사이트 적용 과정

적응형 웹 사이트를 적용하는 방법은 기존 웹 구조의 변형이 없는 색인페이지 작성과 같은 비파괴적인 방법과 하이퍼링크나 글자의 강조 등 구조를 변형시키는 파괴적인 방법이 있다[1]. 학교 웹 사이트 사용자들에게 맞춤형 정보를 제공하기 위해서는 회원 가입을 할 때 입력한 정보를 이용한다. 또한 서로 연관성이 발견된 문서로 바로 이동할 수 있도록 하이퍼링크로 연결한다.

5. 결론 및 향후 연구과제

본 논문에서는 적응형 웹 사이트 구축을 위한 연관규칙 알고리즘을 제안하고 초등학교 현장에 적용하여 구현하였다. 즉, 웹 로그 파일을 분석하여 웹 사이트 사용자들이 즐겨 찾는 빈발문서들을 추출하고 추출된 문서들 간의 연관성 탐색을

통하여 얻어진 빈발패턴을 적응형 웹 사이트에 적용하였다.

본 알고리즘의 특징은 다음과 같다. 엑세스 로그 파일의 전처리 과정을 통해 사용자가 요청한 문서만을 추출하고, 트랜잭션 재구성을 통하여 웹 사이트의 구조상의 특징으로 인한 빈발문서를 제거하였고, 기준문서와 하위문서를 설정하여 웹 사이트에서 제공하는 문서들의 시간적 특성을 반영하여 사용자들의 행동을 분석할 수 있는 문서들만을 추출하였다. 이렇게 구성된 트랜잭션데이터베이스는 빈발문서를 추출하고 연관성을 발견할 수 있는 트랜잭션만을 선별하는 과정을 통하여 FILL이라는 연결리스트의 자료구조로 재구성된다. 연관규칙을 이용한 FILL-mine 알고리즘을 통하여 각 빈발항목에 대한 헤더테이블을 생성하고 최대길이의 빈발패턴을 마이닝하여 발견된 패턴을 통하여 사용자들이 함께 열어본 문서를 파악할 수 있게 된다. 본 알고리즘의 장점은 연결리스트 이외에는 중간 생성물을 생성하지 않으며, 기존의 다른 알고리즘과 비교했을 때 시간 복잡도를 개선한 것이다.

적응형 웹 사이트 구현을 위하여 전처리 모듈, 연결리스트 구성 및 패턴 마이닝 모듈의 두 가지 과정을 거치게 되며, 사용자가 특정 웹 문서를 요청하면 데이터베이스에 저장된 패턴이 웹 사이트에 자동적으로 반영되게 된다. 또한, 회원 가입 시에 저장된 정보를 이용한 개인화 된 맞춤형 정보도 함께 추천되어 사용자들이 웹 사이트를 편리하게 이용할 수 있도록 돕는다.

이후의 연구에서는 사용자 개인의 행동 패턴을 개별적으로 분석하여 개별 사용자의 특성에 맞추어 웹 사이트를 제공하는 연구가 필요하다. 또한 웹 사이트 사용자들의 행동을 순차적으로 분석하여 행동 패턴에 따라 적절한 웹 문서를 제공하는 연구도 있어야 한다.

참 고 문 헌

[1] 고경자, 웹 마이닝을 이용한 적응형 웹 사이트 구축에 관한 연구, 석사학위논문, 경기대학교, 2001.

[2] 김영태, 웹 로그와 구매 DB를 이용한 개인화 시스템에 관한 연구, 석사학위논문, 조선대학교, 2004.

[3] 박인창, L2-tree 기반의 빈발항목집합 탐사 기법, 석사학위논문, 연세대학교, 2003.

[4] 박종배, 클릭스트림 분석을 이용한 사용자별 메뉴 생성 시스템의 설계 및 구현, 석사학위논문, 순천대학교, 2005.

[5] indeutation 오근정, 쇼핑몰의 상품간 연관규칙 탐색을 위한 빈발항목 발견에 관한 연구 : Matrix 접근법, 석사학위논문, 대전대학교, 2004.

[6] 이정민, 연관규칙을 이용한 적응형 학교 웹사이트 구축 알고리즘, 석사학위논문, 서울교육대학교, 2005.

[7] 주종성, 다중항목지지도를 고려한 수정된 H-마이닝 빈발 패턴 탐색, 석사학위논문, 한양대학교, 2005.

[8] 박종수, 웹 로그 파일에서 빈발항목 집합 탐사, 성신여자대학교 기초과학연구지 Vol.17 pp.1-16, 1999.

[9] Pei, J., *Pattern growth methods for frequent pattern mining*, the degree of doctor of philosophy, Simon Fraser university, 2002.

[10] Dai, Y. M., *A data mining system for mining library borrowing history records*, A Master's Thesis, National Chiao-Tung University, 2001.

[11] Agrawl, R., Srikant, R., Fast algorithm for mining association rules, In Proc. 1994 Int. Conf. Very Large Data Bases (VLDB'94), pp.487-499, 1994.

[12] Borgelt, C., *Keeping things simple : Finding frequent item sets by recursive elimination*, Workshop Open Source Data Mining Software (OSDM'05, Chicago, IL), pp.66-70, 2005.

[13] Goebel, M., Gruenwald, L., *A survey of data mining software tools*, ACM SIGKDD Exploration Vol.1 Issue1, pp.20-33, 1999.

[14] Gopalan, R. P., Sucahyo, Y. G., *Fast frequent itemset mining using compressed data representation*, Applied Informatics 2003. pp.1203-1208, 2003.

[15] Mehmed Kantardzic, Data Mining : Concepts, Models, Methods and Algorithms, Wiley-IEEE Press, 2002.

[16] Pei, J., Han, H., Lu, H., Nishio, S., Tang, S., Yang, D., *H-mine: hyper-struct mining of frequent patterns in large databases*, In Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM'01), pp.441-448, 2001.

[17] Prasetyo, B., Pramudiono, I., Kitsuregawa, M., *Hmine- rev: toward H-mine parellelization on mining frequent patterns in large databases*, 情報處理學會技術研究報告 2005-DBS-137(44), No.68, pp.329-336, 2005.

최 윤 희

e-mail : yunee11@paran.com

1993년 서울교육대학교 초등과학교육학과(학사)

2006년 서울교육대학원 컴퓨터교육과(교육학석사)

1993년~2006년 서울 신구로, 고척, 오류초등학교 교사

2006년~현 재 서울오류남초등학교 교사

관심분야 : 데이터 마이닝, 정보통신윤리

전 우 천

e-mail : wocjun@snue.ac.kr

1985년 서강대학교 전자계산학과(공학사)

1987년 서강대학교 전자계산학과(공학석사)

1997년 University of Oklahoma, Dept. of Computer Science(공학박사)

1998년~현 재 서울교육대학교 컴퓨터교육과 교수

관심분야 : 초등컴퓨터교육, 데이터 마이닝, 정보영재, 모바일 학습