

工學碩士 學位論文

웹 문서와 접근로그의 하이퍼링크  
추출을 통한 웹 구조 마이닝

Web Structure Mining by Extracting Hyperlinks  
from Web Documents and Access Logs

指導教授 金 載 薰

2007年 2月

韓國海洋大學校 大學院

컴퓨터工學科

朴 哲 賢

本 論文을 朴哲賢의 工學碩士 學位論文으로 認准함

委員長 工學博士 柳 吉 洙



委 員 工學博士 金 載 熏



委 員 工學博士 李 章 世



2006年 12月

韓國海洋大學校 大學院

컴퓨터工學科

朴 哲 賢

## 목 차

제 1 장 서 론.....	1
제 2 장 관련 연구.....	4
2.1 웹 마이닝.....	4
2.2 웹 로그.....	6
2.3 웹 로그 전처리 과정.....	10
2.4 클릭스트림.....	14
제 3 장 웹 문서 구조화.....	15
3.1 웹 문서의 하이퍼링크 구조화.....	16
3.2 접근로그 전처리 과정.....	23
3.3 클릭스트림 수집.....	25
3.4 접근로그를 통한 정점과 간선 추가.....	28
제 4 장 시스템 구현 및 실험.....	37
4.1 시스템 구현환경.....	37
4.2 시뮬레이션.....	42
제 5 장 결론 및 향후 과제.....	51
참고문헌.....	53

# Web Structure Mining by Extracting Hyperlinks from Web Documents and Access Logs

Chul-Hyun Park

Department of Computer Engineering,  
Graduate School of Korea Maritime University, Busan, Korea

Advisor: Jae-Hoon Kim

## **Abstract**

It is difficult to predict Web structures for being rapidly changed with frequent updates of documents on the Web. Nevertheless, given the structures, information providers can discover users' behavior patterns and characteristics and supply better services to users, and users can find useful information easily and exactly. This paper proposes an improved method for extracting Web structures. The method consists of two steps. The first is constructing a directed graph on Web documents as node with their hyperlinks using the depth-first search algorithm. The second is making up for the direct graph by discovering the hyperlinks, which are not extracted in the first step, called hidden hyperlinks. They can be found by analyzing Web access logs, in which click streams are contained. The click streams do not include clicks on 'Back' buttons because of the local cache problem of Web browsers. This causes the problem not finding correct hidden hyperlinks. To cope with the

problems, this paper propose an algorithm on searching hidden hyperlinks. We have simulated the discovery of the hidden hyperlinks to evaluate the proposed method experimentally. Through the simulations, we have observed that the proposed method discovers most hidden hyperlinks appeared on clickstreams. In the future we should develop some tools for visualizing discovered Web structures and do study on discovering more correct hidden hyperlinks through improving the proposed algorithm.

## 제 1 장 서 론

컴퓨터 기술의 발달과 웹의 급속한 성장으로 웹 사이트의 사용자가 기하급수적으로 증가하였고, 사용자에게 다양한 정보를 실시간으로 제공하고 있다. 기존의 다양한 정보 서비스 시스템들은 웹을 기반으로 통합하거나 웹 사이트를 구축하여 질 높은 정보의 공유와 보다 안정성 있는 정보의 흐름을 보여주고 있다[1]. 이러한 웹 사이트들은 정보의 양이 방대해지고 웹 문서들의 잦은 생성, 삭제와 수정이 반복되면서 더욱 복잡한 구조의 양상을 띄게 되었으며 사용자가 원하는 최적의 정보에 접근하는 방법 또한 예측하기 힘든 구조를 가지게 되었다. 이런 구조에서의 사용자의 행동패턴과 웹 문서간의 관계를 예측할 수 있다면 개발자들은 보다 나은 서비스를 제공하여 합리적이고 효율적인 시스템을 구축할 수 있을 것이다.

웹 로그를 이용하여 웹 사이트의 성능 분석과 웹사이트 디자인 개선, 웹 트래픽의 이해, 웹사이트 개인화 방향성을 제시하여 다양한 마이닝 기법을 적용하고 최적의 정보 서비스 시스템을 구축하는 작업들이 이루어지고 있다[2]. 하지만 근래의 웹 문서들은 텍스트보다 그래픽 위주의 인터페이스가 주를 이루고 있다. 특히 애드온(Add-on) 형태로 동작하는 플래시(Macromedia Flash)와 애플릿(Java Applet)등이 포함되고 있는 웹 문서는 경로 추적이 어려워지면서 구조적 패턴연구가 힘들어졌다. 그래픽 위주로 변화하는 웹은 사용자들이 정보를 보다 쉽게 검색할 수 있는 요구를 일관성 있는 서비스로 제공하기 위해 웹을 구조화하는 연구가 필요하다. 웹을 구조화하기 위해서는 통계적이고 경험적인 분석을 통해 사용자들의 행동패턴을 관찰하고 실질적인 데이터를 수집하는 것이 요구된다[2,3]. 그리고 정확한 데이터를 수집하기 위해서 웹 문서의 분석과 데이

터의 흐름을 분석하여 많은 링크구조를 추출하고 체계적으로 구조화해야 한다. 이러한 과정으로 웹 사용의 효율을 증대시켜 사용자의 특성과 행동 패턴에 적용하고 웹으로 부터의 유익한 정보를 찾아내어 웹 마이닝화 하여야 한다[4,5].

일반적으로 사용자들의 행동패턴을 추적하는 것은 대부분 시스템 자체에서 제공하는 경로분석 기능이나 기존의 로그분석 유틸리티를 통해 이뤄진다[6,7]. 그러나 이런 경로분석 기능들은 몇 가지 문제점들을 가지고 있다. 그 중에 가장 큰 문제점은 플래시와 애플릿과 같은 애드온 프로그램들의 경로추적이 어렵다는 것이다. 이 문제점으로 인해 구조화된 웹 경로들은 문서와 데이터들의 흐름을 전혀 파악할 수 없다.

이를 해결하기 위하여 본 논문에서는 동적인 웹 구조화 방법을 크게 두 가지 방법을 사용하여 제안한다. 첫 번째, 웹 문서의 태그구조 분석을 통해 하이퍼링크를 탐색하고 탐색된 링크의 우선 순서로 깊이 우선 탐색 (Depth First Search) 알고리즘을 적용함으로써 방향그래프 형태로 구조화한다[8]. 두 번째, 웹 로그를 이용하여 첫 번째 과정에서 만들어진 방향그래프를 보완하는 방법으로 이것을 좀 더 자세히 설명하면, 첫 번째 과정에서 생성된 방향그래프는 플래시와 애플릿 등의 히든하이퍼링크는 찾을 수 없다. 이를 해결하기 위한 방법으로 접근로그 정제과정을 거쳐 세션별 클릭스트림(Clickstream)을 추출하여 큐(Queue)와 스택(Stack)의 구조적 방법을 이용해 히든하이퍼링크를 찾아내고 새로운 정점과 간선을 그래프에 갱신한다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 연구에서 제시된 웹 마이닝과 접근로그, 전처리 과정, 클릭스트림에 대하여 논하고, 3장에서는 웹 문서에서 하이퍼링크를 추출하고 이를 방향그래프로 만드는 방법과 접

근로그에서 클릭스트림을 추출하고 새로운 정점과 간선을 갱신하는 방법을 설명한다. 4장에서는 시스템의 구현 및 실험에 관하여 설명하고, 끝으로 5장에서는 결론과 향후 연구방향을 제시한다.



## 제 2 장 관련 연구

본 장에서는 웹 사이트를 웹 구조 마이닝을 적용하기 위해 필요한 관련 연구에 대해 살펴보도록 하겠다. 웹 구조 마이닝을 하기위한 방법으로 웹 문서의 하이퍼링크 탐사를 통해 구조화하는 방법과 웹 구조를 보완하기 위해 필요한 방법으로 접근로그의 전처리 과정과 세션별 클릭스트림 추출 방법으로 살펴보겠다.

### 2.1 웹 마이닝

웹 마이닝은 웹 환경에서 적용되어지는 데이터마이닝 기술로서 웹으로부터 얻어지는 방대한 양의 정보를 필터링하여 의미 있고 유용한 정보를 찾아내어 이를 분석하는 것이다[4,9]. 웹으로부터 얻어지는 유용한 정보는 표 2.1에서 보는 것과 같이 로그 데이터와 사용자 프로파일, 컨텐츠, 웹 문서, 하이퍼링크 등이 있으며, 이를 데이터베이스화하여 정보를 자동으로 검색하고 추출하는 것이다. 웹 마이닝 기법은 세 가지 영역으로 분류할 수 있다[10]. 세 가지 영역은 웹 내용 마이닝(Web Content Mining), 웹 사용 마이닝(Web Usage Mining), 웹 구조 마이닝(Web Structure Mining)으로 구분된다.

표 2.1 웹 마이닝에서 다루는 데이터 유형

Table 2.1 Data types handling in Web mining

	웹 사용 마이닝	웹 내용 마이닝	웹 구조 마이닝
데이터 유형	로그 데이터, 사용자 프로파일 등	컨텐츠(텍스트, 이미지 오디오, 비디오 등)	웹 문서, 하이퍼링크

### (1) 웹 내용 마이닝

웹 내용 마이닝은 웹 사이트를 구성하고 있는 페이지로부터 의미 있는 유용한 내용을 추출하는 기법이다. 즉, 웹 기반의 구조화되지 않은 데이터로부터 원하는 내용을 찾기 위한 데이터마이닝 기법이다.

### (2) 웹 사용 마이닝

웹 사용 마이닝은 웹서버로부터 사용자들이 브라우징 했던 기록을 남기는 웹서버 로그를 이용하여 사용자들의 접속 유형을 탐색하는 데이터마이닝 기법이다.

### (3) 웹 구조 마이닝

웹 구조 마이닝은 하이퍼링크를 포함하고 있는 웹 문서들의 구조를 기반으로, 하이퍼링크를 통해 참조되거나 참조하는 문서들 간의 관계구조에 대해 요약한 정보를 구조화시켜 생성하는 기법이다. 이런 기법은 사용자의 행동패턴과 정보를 쉽고 빠르게 찾을 수 있도록 하며 양질의 서비스를 제공하는데 도움을 준다. 본 논문에서는 웹 사이트를 구조화시키는 방법 중의 하나로서 웹 문서를 방향그래프의 자료구조 형태로 생성하는 방법을 제안하고 있다. 실제 웹 문서는 방향 그래프 형태와 유사하며, 방향 그래프 표현 시 웹 문서 순회에 용이하다[11,14]. 순회 패턴 탐사를 표현하기 위해서 웹 문서의 태그를 분석하여 하이퍼링크를 추출하고 웹 문서 구조를 방향그래프로 표현한다[8]. 그러나 웹 문서에서 순회 패턴 탐사시 플래시와 애플릿에 대한 하이퍼링크 경로는 추출할 수 없어 완전한 그래프를 생성하지 못한다. 본 논문에서는 접근로그 분석을 통해 하이퍼링크를 추출하여 정점과 간선을 방향그래프로 갱신함으로써 이를 보완하고자 한다.

## 2.2 웹 로그

웹 로그는 사용자가 웹서버에 접속하게 되면 그 이후 클릭하는 모든 작업들이 웹서버의 로그데이터에 기록되게 된다. 기록되는 내용으로는 IP 주소와 데이터 크기, 접속시간, 상태코드, 프로토콜, 전송된 문서경로 등 다양한 정보가 함께 기록되고, 사용자가 요청하는 특정 웹 문서뿐만 아니라 해당 웹 문서와 관련된 이미지 파일과 이미지 데이터, 인클루드(Include) 파일 등에 대한 정보까지 웹 로그파일에 기록되게 된다. 따라서 웹서버는 웹 서비스에 대한 요청과 제공에 대한 기록을 모두 웹 로그에 저장하게 된다. 이것은 웹 마이닝의 기초자료가 되고 본 논문에서는 웹 문서에서 알 수 없던 히든하이퍼링크를 찾기 위한 자료로 사용하겠다.

웹 로그는 웹서버에 따라 여러 형식으로 나누어지는데, CLF(Common Log Format)와 IIS(Internet Information Server)형식이 대표적이다. 본 논문에서는 CLF 형식의 웹 로그를 사용하였다. CLF 형식은 웹서버의 원조라 할 수 있는 NCSA(National Center for Supercomputing Application)<sup>1)</sup> 계열의 웹서버에서 사용하는 파일형식으로서 웹 서비스를 하고 있는 대부분의 웹서버가 이 형식을 사용하고 있다. 웹 로그는 기록되는 내용에 따라 접근로그(Access Log), 에러로그(Error Log), 참조로그(Referrer Log), 에이전트로그(Agent Log)로 나눌 수 있다[2,15]. 접근로그 파일은 트랜스퍼로그(Transfer Log)라고도 부르며, 반드시 존재하는 파일로 접속한 사용자에게 관한 일반적인 사항이 기록되며, 나머지 로그 파일은 선택적으로 기록할 수 있다. 표 2.2는 웹 로그종류에 따른 로그정보이다.

---

1) 슈퍼컴퓨터망에서 이용하게 될 각종 프로그램과 통신 규약을 연구하는 기관

표 2.2 웹 로그의 종류

Table 2.2 Types of Web logs

로그파일	특 징
접근로그	일반적인 사항을 기록
에러로그	에러, 접속실패의 시간과 내용
참조로그	소개해 준 사이트와 소개 받은 웹 문서
에이전트로그	브라우저와 OS에 관한정보

접근로그는 웹서버에 호스트가 접속하고 요청한 정보를 기록하는 웹 로그 파일이다. 웹 사이트의 트래픽에 대한 가장 기초적인 정보를 제공하며 웹서버로부터 브라우저에 파일이 전송된 기록들이다. 접근로그파일은 Host, AuthUser, Time, Request, Status, Volume으로 구성되어 있다. 그림 2.1과 표 2.3은 CLF 웹 로그파일 형식과 내용이다.

Host	RFC-931	AuthUser	Time	
203.230.252.17	-	-	[22/Sep/2006:21:02:44+ 0900]	
Request			Status	Volume
GET	/main.gif	HTTP/1.0	200	3921

그림 2.1 CLF 형식을 따르는 웹 로그파일 형식

Fig. 2.1 File format of Web logs in CLF

표 2.3 CLF 형식의 웹 필드 설명

Table 2.3 Field description of CLF

필드	내 용
Host	클라이언트의 도메인 이름 또는 IP 주소로 기록되며, 선택은 웹서버 관리자의 선택사항이다. 도메인 이름으로 기록할 경우 웹서버는 매 접속 때마다 도메인 이름을 역 추적해야 한다. 이는 웹서버에 상당한 부하를 주게 되므로, IP 주소를 사용하는 것이 바람직하다.
RFC-931	현재 RFC-931 부분을 사용하고 있는 웹서버는 없으며, 항상 “-”로 기록된다.
Auth User	등록된 사용자 이름이다. 웹서버에서 특정 디렉터리를 지정하여 사용자 이름과 암호를 설정해 놓은 경우, 해당 디렉터리는 등록된 사용자에게만 자료검색을 허락한다. 또 현재 사용자가 등록 사용자가 아닌 경우 “-”로 기록된다.
Time	접속일자와 시간을 기록한 것으로 형식은 [dd/mm/yyyy : hh:mm:ss #00] 이다. #00은 그리니치 표준시로부터의 시차를 나타낸다.
Request	Method/URL/Protocol 세 개의 필드로 구성된다. Method는 사용자의 요청방법으로 Get, Post, Head 나타내고, URL은 사용자가 요구한 파일의 이름이 기록되고, Protocol은 현재 대부분이 HTTP를 사용하며, 버전정보와 함께 기록된다.
Status	접속 상태와 데이터 이동상태의 현황은 5개의 카테고리로 구분된다. 표 2.4에서 알아보겠다.
Volume	사용자가 실제로 웹서버로부터 가져간 데이터양을 바이트 단위로 기록한 것이다. Status가 200 카테고리(성공)인 경우에만 기록되며, 그 외의 경우는 “-”으로 기록된다. 단, Request 명령이 Head인 경우에도 “-”으로 기록된다.

표 2.4는 표 2.3의 상태코드(Status Code)의 카테고리별 내용이다.

표 2.4 CLF의 상태코드

Table 2.4 Status Codes of CLF

상태코드	설 명
1XX 정보 상태코드	부가적인 정보를 전달하는데 사용
2XX 성공 상태코드	요청이 성공한 경우
3XX 돌아온 상태코드	URL이 옮겨진 상태
4XX 클라이언트 에러코드	구분에러와 같은 클라이언트 측의 에러를 나타내는 상태
5XX 서버 에러코드	서버 측의 에러

에러로그는 웹서버가 호스트의 요청에 서비스 할 때 발생할 수 있는 오류사항을 기록하는 파일이며, 접속실패, 문서 서비스실패 등의 오류내용, 오류 발생시간 등에 관하여 기록되어 있다.

참조로그는 사용 중인 웹서버를 소개해 준 사이트와 소개받은 문서를 방향으로 기록함으로써 해당 웹사이트를 보기 위해서 어떤 문서를 거쳐 왔는지에 대한 기록이 남아있다. 이 로그를 분석하면 방문자가 해당 사이트를 방문하기 위해 어떤 검색엔진을 활용했으며 사이트에 접속하기 위해 어떤 키워드를 검색했는지, 또는 URL 경로 등에 대한 정보를 알 수 있다.

에이전트로그는 사이트 방문자의 웹 브라우저 및 버전, 운영체제의 종류, 화면 해상도, 프로그램의 종류 등에 관한 정보를 제공해 최적화된 웹 사이

트를 구성할 수 있는 단서를 제공한다. 일반적으로 웹 로그를 통해 얻을 수 있는 항목을 분류하면 표 2.6과 같다.

표 2.6 웹 로그 분석을 통해 구해진 정보

Table 2.6 Information obtained by analyzing Web logs

분 석	분석 결과
페이지 분석	총 페이지뷰 수, 일평균 페이지뷰 수, 기본 페이지뷰 수, 방문 당 페이지뷰 수
파일 분석	총 히트수, 일평균 히트수, 방문 당 히트수
방문자 분석	방문 수, 일평균 방문 수, 순 방문 수, 총 방문 수, 일평균 방문자, 방문자 1인당 방문 수, 평균 이용시간
에러 분석	총 에러 수, 일평균 에러 수, 페이지 당 에러 수
서버 분석	총 전송 데이터 량, 일평균 전송 데이터 량, 히트 당 전송된 평균 데이터 량, 브라우저별 사용 율

### 2.3 웹 로그 전처리 과정

웹 로그는 사용자가 웹서버에 접근한 모든 기록이 저장되어 있다. 이것은 웹 로그분석에 꼭 필요한 정보만 있는 것이 아니며 불필요한 정보도 같이 포함되어 있다. 전처리 과정은 웹 로그분석에 불필요한 내용을 필터링하는 것이며, 웹 로그분석을 위해 데이터를 추출, 변환, 정제하는 일련의 과정이다. 전처리 과정은 불필요한 정보를 정제하는 데이터 정제(Data Cleaning)과정과 사용자를 식별하는 사용자 구분(User Identification)과정 그리고 세션을 식별하는 세션 구분(Session Identification)과정이 필요하다 [2,16]. 그림 2.2는 웹 로그 전처리 과정을 보여주고 있다.

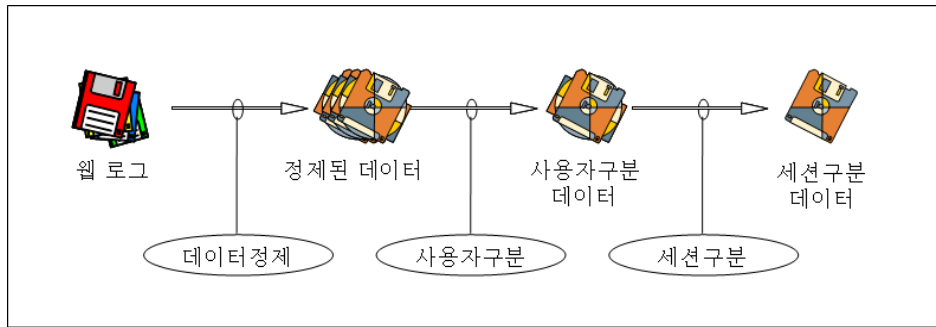


그림 2.2 웹 로그의 전처리 과정  
 Fig. 2.2 Preprocessing of Web logs

(1) 데이터 정제

데이터 정제 과정은 웹서버에 기록된 웹 로그의 불필요한 정보를 필터링하는 과정이다. 접근로그에는 IP주소와 방문시간, 사용자, IP주소, 요청시간, HTTP 방식, 접속한 웹 문서, HTTP 버전, 상태코드, 전송된 바이트 수 등이 기록되어있으며, 웹 로그분석에 불필요한 항목을 필터링하고, 페이지 뷰 당 중복적인 내용을 제거하는 과정이다[2,12].

(2) 사용자 구분

사용자 구분은 웹 사이트에 접근한 사용자를 구분하는 과정으로 표준 웹 로그 형식만으로는 실제 사용자를 구분하는데 있어서 로컬캐시와 방화벽, 프록시서버 등의 존재로 매우 복잡해진다. 접근한 사용자를 구별하기 위해서는 웹 로그에 사용자 정보가 포함되도록 웹서버를 수정하거나, 각각의 웹 문서에 로그를 저장하도록 프로그래밍하고 쿠키를 사용하는 등의 방법이 필요하다. 그렇지 않고 기존의 표준 웹 로그 형식을 사용하는 일반적인 경우에는 사용자를 구분하기 위해서는 IP주소를 사용할 수밖에 없다. 사용자 구분에 사용되는 방법은 IP주소와 에이전트(Agent), 세션ID, 사용자등록, 쿠키(Cookie)를 사용하는 방법, 캐시 버스팅을 이용하는 방



법, 웹 문서에 에이전트나 애플릿을 삽입하는 방법 등 여러 가지 방법들이 쓰이고 있다. 본 논문에서는 IP주소와 에이전트, 타임아웃 시간을 사용하여 구분하겠다. 표 2.7에서는 사용자와 세션의 구분에 사용되어지는 방법들을 비교하였다[12].

표 2.7 사용자 구분을 위한 방법들의 비교

Table 2.7 Comparison on methods for classification of users

방법	설명	보안 관계	장점	단점
IP주소, 에이전트	▶ 각각의 유일한 IP와 에이전트 쌍을 독립된 사용자로 추정.	낮음	▶ 항상 사용가능. ▶ 추가적인 기술이 필요하지 않음.	▶ 유일한 IP임을 보증할 수 없음. ▶ 유동IP일 경우 식별하기 어려움.
세션 ID	▶ 모든 링크에 동적으로 생성된 페이지 ID를 삽입함.	낮음 / 보통	▶ 항상 사용가능. ▶ IP 주소에 독립적임.	▶ 반복된 접근에 대한 개념이 없음. ▶ 완전히 동적인 사이트가 요구됨.
등록	▶ 사용자가 사이트에 직접 등록함.	보통	▶ 브라우저에 국한되지 않고 각 개인을 인식할 수 있음.	▶ 모든 사용자가 등록하는 것을 원하지 않음.
쿠키	▶ 쿠키를 사용자의 컴퓨터에 저장함.	보통 / 높음	▶ 반복된 접근을 인식할 수 있음.	▶ 사용자들의 부정적인 견해에 의해 거부될 수 있음.
소프트 웨어 에이전트	▶ 에이전트 프로그램이 브라우저에 삽입되어 사용데이터를 웹서버에 보내줌.	높음	▶ 모든 사이트에 대하여 정확한 사용데이터를 얻을 수 있음.	▶ 사용자에게 의해 거부될 수 있음.
브라우저 수정	▶ 브라우저가 사용데이터를 기록함.	매우 높 음	▶ 모든 사이트에 대하여 정확한 사용데이터를 얻을 수 있음.	▶ 사용자가 브라우저에 요구사항을 명시해 주어야 함.

### (3) 세션 구분

세션 구분은 한 사용자가 웹 사이트에 접속하여 웹 탐색을 수행한 후 접속을 종료할 때까지의 일련의 행위이다[12]. 따라서 실제 사용자의 웹 사용패턴을 탐색하기 위한 세션 구분은 데이터마이닝의 입력 데이터로서 매우 중요한 의미를 가진다. 그러나 웹 로그만으로 세션을 구별하기에는 어려움이 있다. 사용자들은 로그아웃 버튼을 누르지 않고, 웹 브라우저를 닫는 경우와, 사용자가 웹 문서를 접근할 때 접근하는 정보가 로컬캐시로 인해 로그에 기록되지 않을 수도 있다는 점이다. 이것은 '뒤로' 버튼을 사용한 경우이며, 이미 한번 가져온 웹 문서는 브라우저의 캐시에 저장되었다가 다시 요청했을 때 캐시에 있는 내용이 사용자에게 다시 보여 지기 때문에 서버의 접근로그에는 기록되지 않는 것이다. 본 논문에서는 큐와 스택의 구조를 사용하여 '뒤로' 버튼을 알아내어 로컬캐시 문제를 해결하도록 하겠다.

## 2.4 클릭스트림

클릭스트림(Clickstream)은 웹 사이트상에서의 사용자의 이동경로를 분석할 수 있는 모든 행위 패턴정보를 말한다. 이것은 컴퓨터 화면상의 어떤 객체를 선택하거나 원하는 항목을 실행하도록 하기위해 마우스를 이용하여 클릭하는 순차적인 흐름의 정보들이다. 이것은 고객의 행동과 선호도, 구매성향 등에 대한 지식기반이 될 수 있으며, 사용자의 최근 행위를 반영하는 개인에 맞춰진 역동적인 콘텐츠제공에 이용할 수 있는 자료가 된다. 본 논문에서는 클릭스트림을 이용하여 사용자 행동을 분석하고 경로를 파악하여 웹 문서에서 찾지 못한 히든하이퍼링크의 올바른 간선을 찾고, 방향그래프를 보완하는데 사용한다.

### 제 3 장 웹 문서 구조화

웹 문서 내의 하이퍼링크는 각 문서를 연결하는 자료의 흐름이다. 본 논문에서는 그림 3.1과 같이 웹 구조화 방법으로 웹 문서와 접근로그를 사용하여 웹 문서를 구조화 한다. 구조화 방법은 문서의 하이퍼링크를 통해 웹 사이트를 구조화하고 자료의 흐름을 알아냄으로써 사용자의 행동패턴을 파악한다. 사용자의 행동패턴은 미처 탐색 하지 못한 히든하이퍼링크의 구조를 알아내는 기초자료가 되며, 이를 바탕으로 접근로그를 사용하여 적합한 자료구조 형태로 표현한다.

본 논문에서는 웹 문서를 적합한 자료구조 형태로 표현하기 위해 웹 문서의 하이퍼링크 구조를 깊이 우선 탐색 알고리즘을 적용하여 사이트의 모든 하이퍼링크를 추출하고 방향그래프 형태로 구조화한다. 하지만 웹 문서 내의 플래시나 애플릿과 같은 애드온 형식의 프로그램에서는 완전하지 못하다. 플래시파일의 경우 해당 플래시파일의 하이퍼링크는 존재하지만 플래시 파일내의 실제 웹 문서를 연결하는 하이퍼링크는 보이지 않는다. 보이지 않는 히든하이퍼링크를 보완하기 위하여 접근로그의 전처리 과정을 거쳐 순차적인 행동패턴을 파악할 수 있는 클릭스트림을 스택과 큐의 구조를 이용하여 새로운 정점과 간선을 추가하고 이를 방향그래프에 갱신한다.

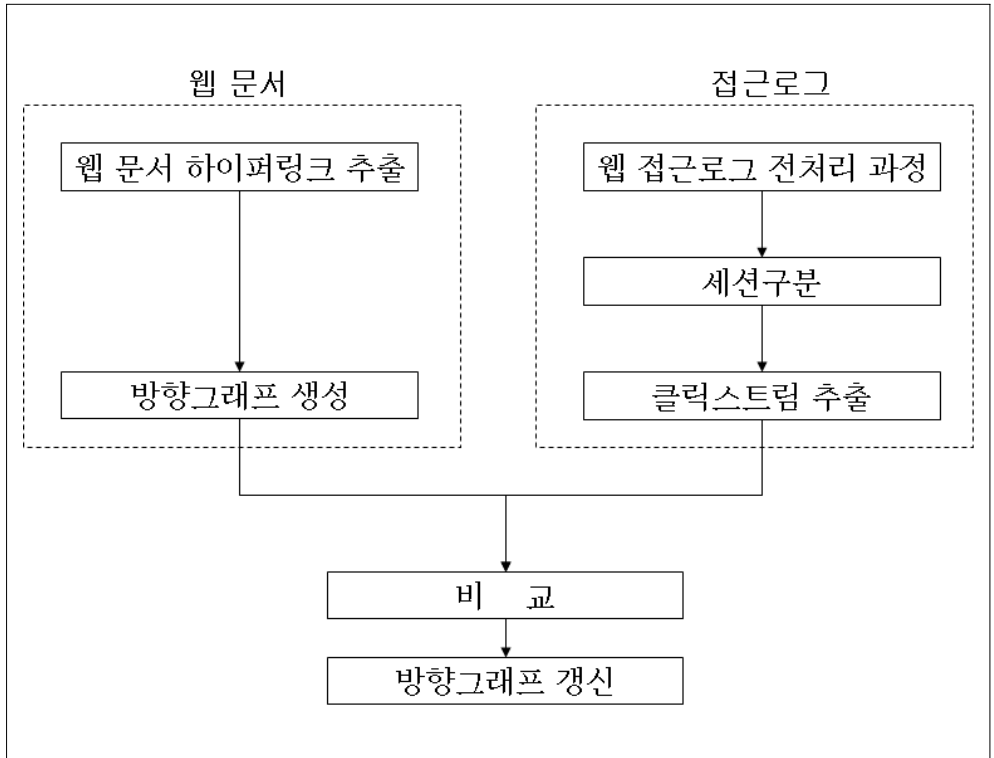


그림 3.1 제안한 시스템 흐름도

Fig. 3.1 The flowchart of the proposed system

### 3.1 웹 문서의 하이퍼링크 구조화

웹 문서들은 HTML 태그를 사용하여 문서들을 링크한다. 웹 사이트를 구조화하기 위해서는 문서와 문서를 연결해주는 링크태그를 분석하여 하이퍼링크를 추출하고, 추출된 하이퍼링크를 깊이 우선 탐색 알고리즘을 적용하여 적합한 자료구조 형태인 방향그래프로 표현한다. 3.1절에서는 HTML 태그의 하이퍼링크를 분류하는 과정과 분류된 하이퍼링크를 웹 문서 순환알고리즘을 적용해 방향그래프를 생성하는 과정으로 살펴보겠다.

(1) HTML 태그의 하이퍼링크 분류

그림 3.2는 웹 문서의 일부이며 HTML 태그로 구성되어 있다[13]. HTML 태그는 다양한 속성을 가지며 속성은 값을 가질 수 있다. 웹 문서의 HTML 태그들을 분리하면 해당 태그가 표현하는 링크의 정보들을 얻어 올 수 있다. 본 논문에서는 태그들 가운데 하이퍼링크를 표현하는 태그만을 처리하기 위해 HTML에 기술되는 태그속성 중 <a>태그의 href와 <frame>의 src, <iframe>의 src, <form>의 action 속성의 하이퍼링크만 사용한다. 하이퍼링크를 표현하는 HTML 태그와 속성은 표 3.1과 같다.

그림 3.2 HTML 소스코드의 일부분

Fig. 3.2 The part of a source code in HTML

```
<html>
...
<form action="form.html" method="get" enctype="mime type">
...
</form>
...
<frame src="top.html" name="top" >
...
</frameset>
...
<a href="menu1.html"></a>
<a href="menu2.html"></a>
<a href="menu3.html"></a>
...
<iframe src="bottom.html" name="bottom">
...
</iframe>
...
</html>
```

표 3.1 하이퍼링크를 표현하는 HTML 태그와 속성

Table 3.1 HTML tags and attributes for hyperlink expression

HTML 태그	속성
<a>	href
<applet>	codebase, code
<area>	href
<bgsound>	src
<body>	background
<embed>	src
<fig>	src
<form>	action
<frame>	src
<iframe>	src
<img>	src, dynsrc, usemap
<input>	src
<insert>	classid, usemap, data
<item>	classid, usemap, data
<map>	name
<object>	codebase
<overlay>	src
<param>	value
<script>	src, for
<td>	background
<th>	background
<tr>	background

하이퍼링크를 표현하는 HTML 태그속성 중 형식이 텍스트인 하이퍼링크만 추출하여야 한다. 표 3.2의 내용 중 타입이 텍스트인 것은 또 다른

문서 링크들을 포함할 수 있기 때문에 이를 추출하여 깊이 우선 탐색 알고리즘에 적용한다. 그 외의 타입은 탐색 시 배제한다.

표 3.2 내용 타입과 하위 타입

Table 3.2 Types and subtypes of contents

타입	서브타입	타입	서브타입
텍스트	html, htm, sgml, php, asp, jsp	그림	gif, jpg, jpeg, tiff, bmp
멀티파트	mixed, form-data, enc	오디오	mp3, basic, wma, au, wmf, mid, wav, wsz
메시지	http, news, ffc822	비디오	mpeg, wmv, mpg, dvi, quicktime asx, asf, smi, ra, ram, rm, avi,
응용 프로그램	pdf, zip, hwp, txt, ppt, xls, mdb, doc, bak, eps, ps, rar, tar, tex, rm, exe, mov, gdb, gz	플래시	swf

탐색 시 추출하고자 하는 하이퍼링크는 해당 웹서버에 존재하는 문서를 대상으로 하므로 웹서버 외부에 존재하는 문서의 하이퍼링크들과 웹서버 자체에 가지고 있는 정보인 내부 하이퍼링크들을 구분해야한다. 구분은 도메인주소와 IP주소로 구분되며, 외부 하이퍼링크의 경우 추출대상이 아니기 때문에 하위까지 추출할 필요가 없고 해당 문서에 대한 검증과 정보만 얻으면 된다. 표 3.3은 그림 3.2의 웹 문서에서 추출한 하이퍼링크이다.



표 3.3 추출한 하이퍼링크

Table 3.3 Extracted hyperlinks

태그	속성	하이퍼링크 경로
<a>	href	menu1.html
<a>	href	menu2.html
<a>	href	menu3.html
<iframe>	src	bottom.html
<frame>	src	top.html
<form>	action	form.html

(2) 웹 문서 순환탐색 알고리즘

그림 3.3은 방문기록 순환탐색 알고리즘으로 웹 문서가 주어졌을 때 최초의 정점에서 출발하는 첫 웹 문서로부터 표 3.3에서 얻어진 링크만을 분석하여 연결된 모든 문서 집합을 깊이 우선 탐색 알고리즘을 적용하여 모든 경로를 끝까지 탐색한다. 순환탐색 시 정점을 방문하는 순서대로 스택에 넣고 방문기록을 하여 이미 방문한 정점을 방문 할 경우 방문기록에 있으므로 순환이 탐색되었음을 알 수 있고, 스택을 이용해서 순환경로를 출력할 수 있다. 방문기록이 되지 않은 정점을 탐색했을 경우에는 그 정점을 스택에 넣고 반복한다. 재귀호출을 이용하며 정점을 스택에 넣고 재귀순환이 끝나면 호출했던 함수로 돌아와서 다시 스택에서 제거한다. 스택이 모두 비워지면 순환탐색은 종료된다. 이 알고리즘은 깊이 우선 탐색을 하기 때문에 탐색 시 정점 'd'가 방문 될 때마다, 그 정점과 인접하면서 방문되지 않은 모든 정점들을 순환적으로 찾을 수 있다. 이런 과정을 통하여 정점과 간선을 추출하면 표 3.4의 결과와 그림 3.4와 같이 방향그래프로 나타낼 수 있다.

```

** 웹 문서 순환탐색 알고리즘 **
//웹 사이트가 주어졌을 때 최초의 정점에서 출발하는 첫 웹 문서
 로부터 하이퍼링크만을 분석하여 연결된 모든 문서 집합을 깊이
  우선 탐색하는 웹 문서 순환탐색 알고리즘
//입력 : 웹 사이트의 웹 문서 d
//출력 : 웹 사이트의 웹 문서 d와 하이퍼링크h를 추출한 방향
  그래프
// isEqualSite()은 동일 사이트인지를 도메인주소의 하이퍼링크
  주소 h를 참고하여 알아내는 함수
// isVisited()은 하이퍼링크 h의 방문여부를 알아내는 함수
// AddVisitedDocument()은 문서 d의 방문을 추가하는 함수
// VisitedDocument[]=∅ // 방문여부저장 배열 초기화 선행선언
// HyperLinks[][]=0 // 문서에 대한 간선 초기화 선행선언
Find_Hyperlink(document d)
{
  // VisitedDocument에 대해서 문서d 방문 추가
  AddVisitedDocument(d);
  //문서d에 있는 각 하이퍼링크 h
  foreach hyperlink h in document d {
    // 동일 사이트인지 검사
    if (isEqualSite(h) == true) {
      // d → h 링크 추가
      HyperLinks[d][h]=1;
      // isVisited(h) 방문 문서 검사
      if (isVisited(h)==false){
        Find_Hyperlink(h);
      }
    }
  }
  return;
}

```

그림 3.3 웹 문서 순환탐색 알고리즘

Fig. 3.3 An algorithm of recursive search of Web documents

표 3.4 웹 문서에서 추출한 하이퍼링크

Table 3.4 Hyperlinks extracted from web documents

정점	→	정점
A.html	→	B.html
A.html	→	C.html
A.html	→	D.html
A.html	→	E.html
A.html	→	F.html
B.html	→	G.html
B.html	→	H.html
C.html	→	I.html
C.html	→	J.html
D.html	→	A.html
D.html	→	J.html
D.html	→	K.html
D.html	→	L.html
E.html	→	L.html
I.html	→	B.html
L.html	→	A.html

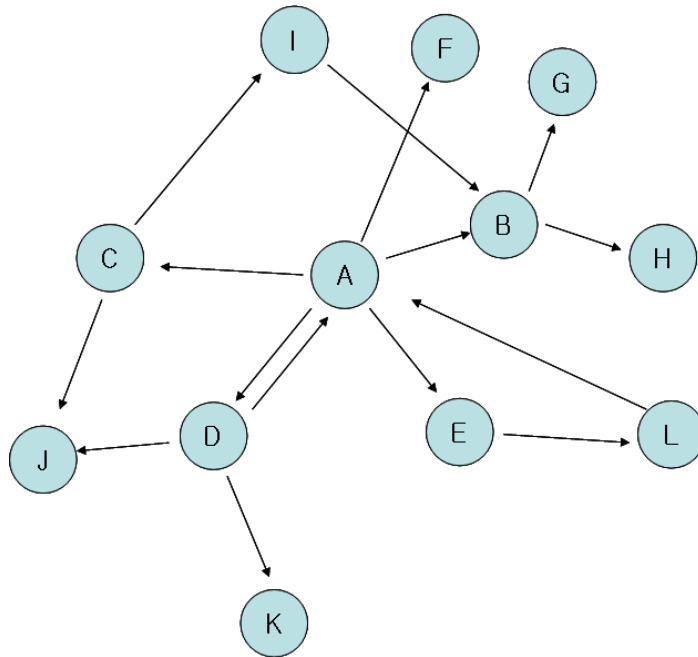


그림 3.4 웹 구조를 표현한 방향그래프

Fig. 3.4 The directed graph represented the Web structure

따라서 플래시파일의 하이퍼링크가 탐색되면 해당 플래시파일의 하이퍼링크는 존재하지만 플래시파일내의 실제 웹 문서를 연결하는 하이퍼링크는 웹 문서를 통해 알 수가 없다. 그림 3.4는 웹 문서에서 미처 탐색하지 못한 히든하이퍼링크가 포함될 수 있기 때문에 신뢰할 수 있는 그래프가 아니다. 본 논문에서는 접근로그에서 탐색한 클릭스트림을 이용해 새로운 정점과 간선들을 갱신하여 방향그래프를 보완한다.

### 3.2 접근로그 전처리 과정

접근로그 데이터를 이용하여 새로운 정점과 간선들을 갱신하여야한다. 그러기 위해서는 접근로그의 정제과정을 거친 후 사용자 구분과 세션 구분을 하는 전처리 과정이 필요하다.

#### (1) 접근로그 정제과정

접근로그의 정제과정은 원시 데이터로부터 불필요한 항목을 필터링하는 과정이다. 웹 로그는 HTTP 프로토콜의 연결속성 때문에 해당 웹 문서뿐만 아니라 해당 웹 문서와 관련된 이미지파일, 미디어파일, 인크루드파일 등 사용자가 한 번의 행동으로 관련된 모든 파일들이 웹 로그에 기록되게 된다. 로그를 분석하는 항목으로는 독립적인 세션을 구하기 위한 방법으로 IP주소, 같은 IP주소의 에이전트 구분, 타임아웃 시간, 서비스상태코드를 고려하여 각각의 세션에 대한 문서접근을 구분한다. 데이터 정제는 페이지 뷰 당 하나의 성분만 남기고 중복적인 모든 파일을 제거한다는 것을 의미하며 이는 사용자의 요청과 직접적인 관련이 없거나 분석에 있어서 불필요한 이미지파일과 미디어파일, 인크루드파일 등을 제거하는 작업으로, 요청된 페이지에 포함된 이미지나 비디오 클립 등이 이에 해당될 수 있다. 이처럼 불필요한 페이지에 대한 필터링은 웹 로그의 요

청된 파일에 대한 확장자를 통해 구분되어질 수 있다. 그림 3.5는 접근로그 원시데이터이며, 이를 정제하면 그림 3.6과 같이 확장자가 텍스트 파일만 남겨지고, 그 외 확장자 파일은 모두 삭제되게 된다.

```
210.114.57.238 - - [21/Sep/2006:11:34:39 +0900] "GET /index.html HTTP/1.1" 200 2363
210.114.57.238 - - [21/Sep/2006:11:34:39 +0900] "GET / HTTP/1.1" 200 2261
210.114.57.238 - - [21/Sep/2006:11:34:40 +0900] "GET /icons/blank.gif HTTP/1.1" 200 148
210.114.57.238 - - [21/Sep/2006:11:34:40 +0900] "GET /icons/folder.gif HTTP/1.1" 200 225
210.114.57.238 - - [21/Sep/2006:11:34:40 +0900] "GET /icons/back.gif HTTP/1.1" 200 216
210.114.57.238 - - [21/Sep/2006:11:34:40 +0900] "GET /icons/text.gif HTTP/1.1" 200 229
210.114.57.238 - - [21/Sep/2006:11:34:41 +0900] "GET /main.php HTTP/1.1" 200 245
210.114.57.238 - - [21/Sep/2006:11:34:41 +0900] "GET /icons/tar.gif HTTP/1.1" 200 219
210.114.57.238 - - [21/Sep/2006:11:34:41 +0900] "GET /icons/folder.gif HTTP/1.1" 200 225
210.114.57.238 - - [21/Sep/2006:11:34:41 +0900] "GET /aro.html HTTP/1.1" 200 2134
210.114.57.238 - - [21/Sep/2006:11:34:42 +0900] "GET /aro_01.html HTTP/1.1" 200 922
210.114.57.238 - - [21/Sep/2006:11:34:42 +0900] "GET /icons/unknown.gif HTTP/1.1" 200 245
210.114.57.238 - - [21/Sep/2006:11:34:42 +0900] "GET /icons/text.gif HTTP/1.1" 200 229
210.114.57.238 - - [21/Sep/2006:11:34:42 +0900] "GET /icons/tar.gif HTTP/1.1" 200 219
```

그림 3.5 접근로그  
Fig. 3.5 Access logs

```
210.114.57.238 - - [21/Sep/2006:11:34:39 +0900] "GET /index.html HTTP/1.1" 200 2363
210.114.57.238 - - [21/Sep/2006:11:34:41 +0900] "GET /main.php HTTP/1.1" 200 245
210.114.57.238 - - [21/Sep/2006:11:34:41 +0900] "GET /aro.html HTTP/1.1" 200 2134
210.114.57.238 - - [21/Sep/2006:11:34:42 +0900] "GET /aro_01.html HTTP/1.1" 200 922
```

그림 3.6 정제된 접근로그  
Fig. 3.6 Cleaned access logs

## (2) 사용자와 세션의 구분

세션별 클릭스트림을 저장하기 위해서는 사용자 식별이 이루어져야 한다. 하지만 대부분의 웹 사용자들은 개인정보 유출 등의 문제로 인해 자신들의 익명성을 원하기 때문에 인증이 꼭 필요한 곳에서만 인증절차를 거치게 된다. 그리고 표준 웹 로그 데이터만으로는 사용자를 완전히 구분할 수 없으므로 정확한 사용자 구분을 위해 고유의 세션 ID를 부여한다.

세션 ID는 접속시간과 사용자 IP주소, 에이전트를 구분하여 총 45자리로 설계하였다. 1-14번째 자리까지는 년 4자리, 월 2자리, 일 2자리, 시 2자리, 분 2자리, 초 2자리로 이루어져 있다. 15번째 자리는 구분자를 삽입하였다. 16-30번째 자리는 사용자의 IP주소로 구성되어 있다. 그림 3.7은 세션 ID 설계구성이다.

세션 ID : 20060922210244_203.230.252.16 년/월/일/시/분/초 사용자 IP 주소
----------------------------------------------------------------

그림 3.7 세션 ID 설계

Fig. 3.7 Design of session IDs

세션은 사용자가 접속을 시작한 시점부터 접속을 종료한 시점까지를 말한다. 세션이 종료되기 전까지는 동일한 사용자에 의한 레코드로 처리하지만 사용자가 일정한 시간동안 아무런 응답이 없을 경우 새로운 세션 ID를 부여하고 다른 사용자로 구분하여야 한다. 본 논문에서는 30분으로 제한하였다.

### 3.3 클릭스트림 수집

클릭스트림은 웹사이트에서 사용자의 이동경로를 분석할 수 있는 모든

행위 패턴정보를 말한다. 사용자가 웹사이트에 접속한 순간부터 웹 문서를 이동하기 위해 마우스를 이용하여 클릭 이벤트를 발생시키는 순차적인 흐름의 정보이다. 이런 클릭스트림을 수집하기 위해서는 3.2절의 과정을 통해 세션별로 웹 문서 형식의 하이퍼링크를 수집한다. 클릭스트림은 웹 문서 이동경로와 문서 당 머문 시간으로 구분하고, 문서 이동 경로는 동일한 IP를 묶고 요청시간별로 정렬하여 해당 IP가 접근한 문서들의 이동 경로를 추출한다. 표 3.5는 그림 3.6의 정제된 접근로그 데이터를 통해 수집된 클릭스트림이다.

210.114.57.238	- -	[21/Sep/2006:11:34:39 +0900]	"GET /index.html HTTP/1.1"	200	2363
210.114.57.238	- -	[21/Sep/2006:11:34:40 +0900]	"GET /bottom.html HTTP/1.1"	200	483
210.114.57.238	- -	[21/Sep/2006:11:34:40 +0900]	"GET /main.php HTTP/1.1"	200	2540
210.114.57.238	- -	[21/Sep/2006:11:34:42 +0900]	"GET /company1.html HTTP/1.1"	200	1637
210.114.57.238	- -	[21/Sep/2006:11:36:59 +0900]	"GET /aro.html HTTP/1.1"	200	2134
220.93.55.27	- -	[21/Sep/2006:11:35:09 +0900]	"GET /index.html HTTP/1.1"	200	2363
220.93.55.27	- -	[21/Sep/2006:11:35:10 +0900]	"GET /bottom.html HTTP/1.1"	200	483
220.93.55.27	- -	[21/Sep/2006:11:35:10 +0900]	"GET /main.php HTTP/1.1"	200	245
220.93.55.27	- -	[21/Sep/2006:11:35:17 +0900]	"GET /h_pressure.html HTTP/1.1"	200	2206
220.93.55.27	- -	[21/Sep/2006:11:35:18 +0900]	"GET /h_pressure_01.html HTTP/1.1"	200	2907
220.93.55.27	- -	[21/Sep/2006:11:35:18 +0900]	"GET /h_pressure_01a.html HTTP/1.1"	200	1232
220.93.55.27	- -	[21/Sep/2006:11:37:22 +0900]	"GET /aro.html HTTP/1.1"	200	2134
220.93.55.27	- -	[21/Sep/2006:11:37:25 +0900]	"GET /aro_01.html HTTP/1.1"	200	1226
220.93.55.27	- -	[21/Sep/2006:11:37:25 +0900]	"GET /arno_01b.html HTTP/1.1"	200	4071
24.69.255.237	- -	[21/Sep/2006:12:22:34 +0900]	"GET /wilden.html HTTP/1.1"	200	2126
24.69.255.237	- -	[21/Sep/2006:12:22:40 +0900]	"GET /wilden_01.html HTTP/1.1"	200	1376
24.69.255.237	- -	[21/Sep/2006:12:22:41 +0900]	"GET /wilden_01b.html HTTP/1.1"	200	4506
24.69.255.237	- -	[21/Sep/2006:12:25:22 +0900]	"GET /wilden_02.html HTTP/1.1"	200	1376
24.69.255.237	- -	[21/Sep/2006:12:25:22 +0900]	"GET /wilden_02b.html HTTP/1.1"	200	4491

그림 3.6 접근로그 데이터 예제  
 Fig. 3.6 Examples of Web access logs

표 3.5 세션별 클릭스트림

Table 3.5 Clickstreams for sessions

세션 구분	세션 ID	클릭스트림
세션1	20060921113439_210.114.57.238	index.html
	20060921113440_210.114.57.238	bottom.html
	20060921113440_210.114.57.238	main.php
	20060921113442_210.114.57.238	company1.html
	20060921113659_210.114.57.238	aro.html
세션2	20060921113509_220.93.55.27	index.html
	20060921113510_220.93.55.27	bottom.html
	20060921113510_220.93.55.27	main.php
	20060921113517_220.93.55.27	h_pressure.html
	20060921113518_220.93.55.27	h_pressure_01.html
	20060921113518_220.93.55.27	h_pressure_01a.html
	20060921113722_220.93.55.27	aro.html
	20060921113725_220.93.55.27	aro_01.html
	20060921113725_220.93.55.27	aro_01b.html
세션3	20060921122234_220.93.55.27	wilden.html
	20060921122240_220.93.55.27	wilden_01.html
	20060921122241_220.93.55.27	wilden_01b.html
	20060921122522_220.93.55.27	wilden_02.html
	20060921122522_220.93.55.27	wilden_02b.html



### 3.4 접근로그를 통한 정점과 간선 추가

실제 존재하지만 웹 문서의 하이퍼링크에서 추출하지 못한 히든하이퍼링크에 대해 접근로그의 전처리 과정에서 추출한 세션별 클릭스트림을 이용하여 새로운 정점과 간선을 추가한다. 이때 '뒤로' 버튼에 따른 문제가 발생하게 된다. 이는 항해하는 사용자들은 '뒤로' 버튼을 자주 사용하는데 이미 접근한 문서는 브라우저캐시에 저장되어지고, 다시 요청이 일어나면 캐시에 있는 내용이 사용자에게 보내져 서버의 접근로그에는 기록되지 않는다는 문제점을 가지게 된다. 이를 해결하기 위한 방법으로 본 논문에서는 그림 3.7의 알고리즘을 사용하여 '뒤로' 버튼을 알아내고 히든하이퍼링크 또는 새롭게 나타난 정점과 간선을 추가한다.

접근로그를 통한 정점과 간선추가 방법은 클릭스트림을 큐에 인큐(Enqueue)하고 큐에 들어간 클릭스트림을 디큐(Dequeue)하여 스택에 푸쉬(Push)를 한다. 큐의 헤드와 스택의 탑을 앞의 과정에서 만들어진 방향 그래프와 비교한다. 동일한 간선이 있을 경우 큐를 디큐하고 스택에 푸쉬한다. 다음 큐의 헤드와 스택의 탑을 비교한다. 만약 동일한 간선이 없을 경우 스택에서 팝(Pop)하고 큐의 헤드와 스택의 탑을 비교한다. 이런 과정을 새로운 정점이 나타날 때 까지 반복한다. 새로운 정점이 나타나면 이를 해싱함수를 이용하여 새로운 정점에 대한 패턴후보를 카운트하게 된다. 패턴후보 중 짧은 경로이면서 카운터가 가장 많은 정점에 새로운 간선을 추가한다.

**\*\* 히든링크 탐색 알고리즘 \*\***

```
//웹 문서의 하이퍼링크에서 추출하지 못한 히든하이퍼링크에 대해 접근
  로그의 전처리 과정에서 추출한 세션별 클릭스트림을 이용하여 새로운
  정점과 간선을 추가하는 히든링크 탐색 알고리즘
//입력 : 클릭스트림 집합 CSet, G=(V, E) 방향그래프
//출력 : 새로운 정점에 대해 G<s.top(), e.nv>을 추가한 방향그래프
//EdgeLengthpass() 새로운 정점에 대해 패턴후보의 길이가 짧은 간선
  을 찾는 함수
//EdgeMax() 새로운 정점에 대해 패턴후보 중 빈발도가 많은 간선을
  찾는 함수
//updateGraph() 새로운 정점을 그래프에 간선을 추가하는 함수
//CSet : 클릭스트림 집합
//V : 방향그래프 정점집합
//E : 방향그래프 간선집합
SearchHiddenLink(CSet, G)
{
  Queue Q;
  Stack S;
  HashSet NewEdgeSet {
    Stack s;
    Vertex nv;
    int count;
  }
  foreach c in CSet {
    Q.Enqueue(c); //클릭스트림의 모든 정점을 큐에 저장
    S.push(Q.DeQueue());
    while (not Q.empty()) {
      V1 = S.top();
```

그림 3.7 히든링크 탐색 알고리즘

Fig. 3.7 An algorithm to search the hidden links in directed graph

```

    V2 = Q.DeQueue();
    if (<V1 V2> ∈ E ) {
        S.push (V2);
    } else {
        if (V2 ∉ V) {
            V.addVertex(V2);
            // 새로운 정점, 패턴후보
            NewEdgeSet.hash(S, V2) ++;
            break; // while문 루프 Exit
        } else {
            S.pop();
        }
    }
}
S.init(); //스택 초기화
Q.init(); //큐 초기화
}
foreach e in NewEdgeSet
{
    //새로운 정점 e.nv에 대해 경로의 길이가 가장 짧은 간선을
    선택
    if (e.nv is EdgeLengthPass(e.s) ) {
        //새로운 정점 e.nv에 대해 가장 빈발한 간선을 선택
        if (e.nv is EdgeMax(s.count))
            updateGraph(G<s.top(), e.nv>); // 그래프 추가
    }
}
}
}

```

그림 3.7 히든링크 탐색 알고리즘(계속)

Fig. 3.7 An algorithm to search the hidden links in directed graph  
(Continued)

예를 들어, 하나의 세션에 (A, B, H, D, J, K, X) 클릭 스트림이 있다면 다음의 순서로 방향그래프를 보완한다.

① 그림 3.8과 같이 클릭스트림을 큐에 인큐한다.

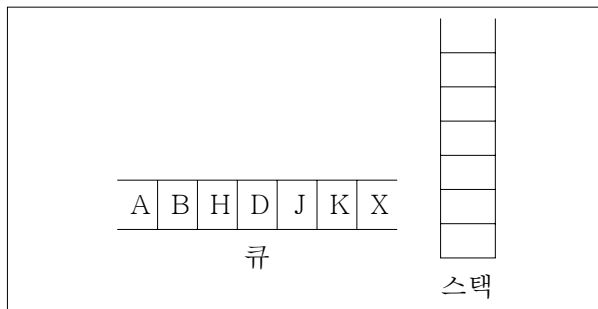


그림 3.8 처음 단계

Fig. 3.8 The first process

② 클릭스트림이 저장된 큐를 그림 3.9와 같이 디큐하여 스택에 푸쉬한다.

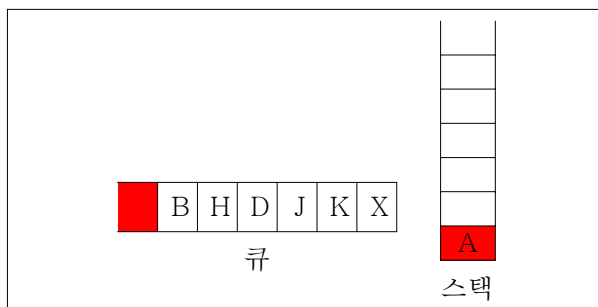


그림 3.9 두 번째 단계

Fig. 3.9 The second process

③ 그림 10은 큐의 헤드('B')와 스택의 탑('A')을 이미 찾은 방향그래프

그림 3.4와 비교하여 간선이 있는지 확인한다.

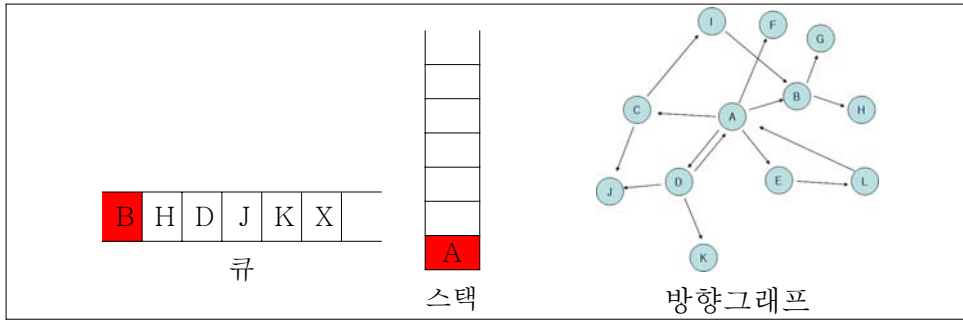


그림 3.10 세 번째 단계

Fig. 3.10 The third process

- ④ 동일한 간선이 존재하면 큐를 디큐하고 스택에 푸쉬한다. ('A'→'B', 'B'→'H' 간선이 존재 하므로 큐에서 디큐하여 스택에 푸쉬한다.)

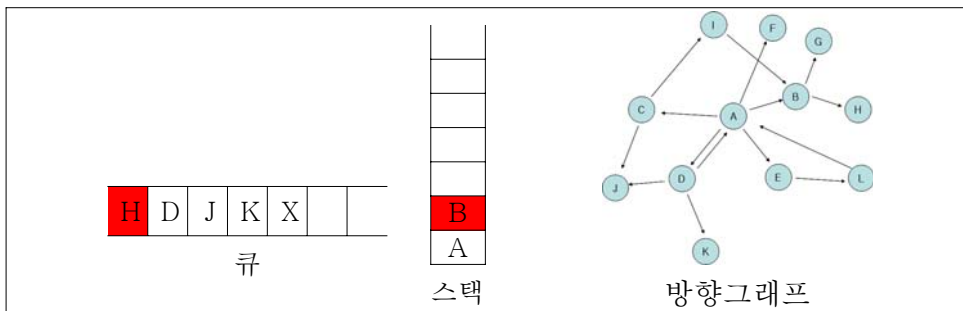


그림 3.11 네 번째 단계

Fig. 3.11 The fourth process

- ⑤ 동일한 간선이 나타나면 ③~④ 과정을 반복한다. 만약 그래프와 비교하여 동일한 간선이 존재하지 않으면 스택에서 팝한다.('H'→'D' 간선이 존재하지 않으므로 'H'를 스택에서 팝한다.) 이 경우는 '뒤로' 버튼을 클릭한 상태에 해당한다.

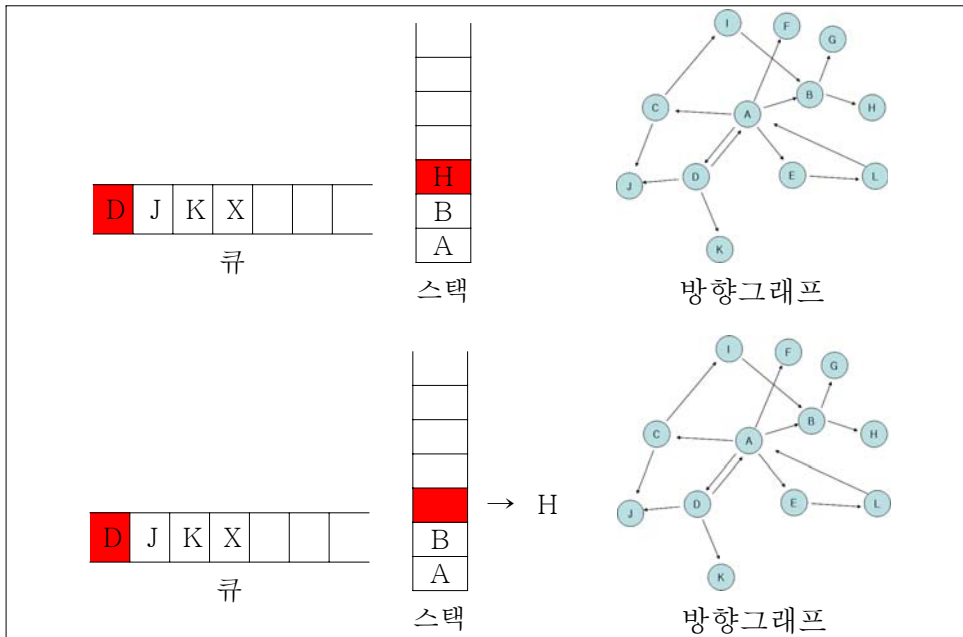


그림 3.12 다섯 번째 단계  
 Fig. 3.12 The fifth process

⑥ 새로운 정점이 나타날 때 까지 위의 과정을 반복한다. 큐가 비워질 때 까지 새로운 정점이 나타나지 않는다면 새로운 정점이 없으므로 다음 세션 클릭스트림을 검사한다.

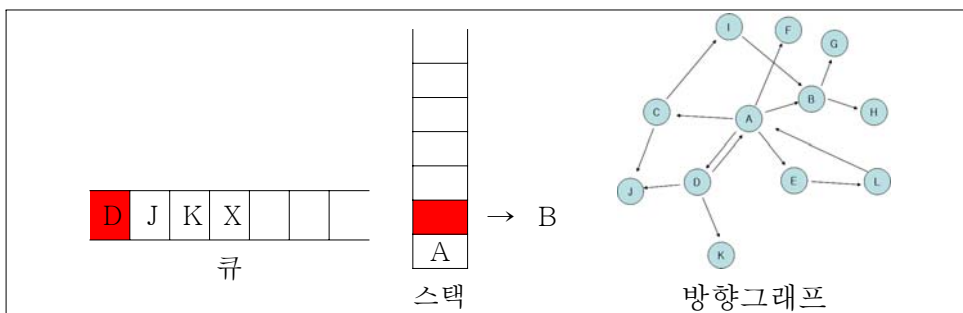


그림 3.13 여섯 번째  
 Fig. 3.13 The sixth process

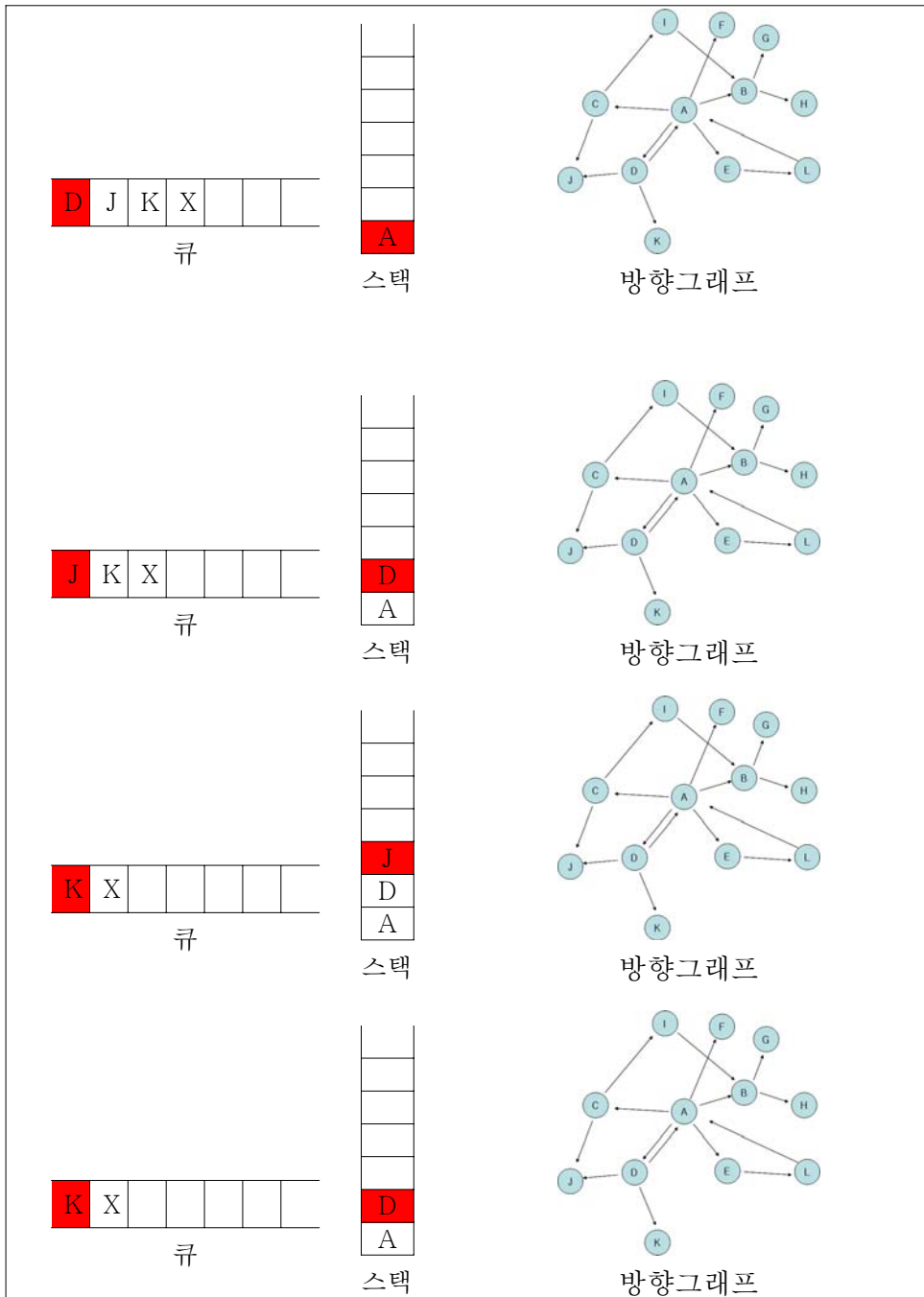


그림 3.14 The seventh process

Fig. 3.14 일곱 번째 단계

- ⑦ 새로운 정점이 탐색 되었을 때 새로운 정점과 스택의 내용을 해싱함수에 카운트하여 저장하고 큐와 스택을 초기화 한다.

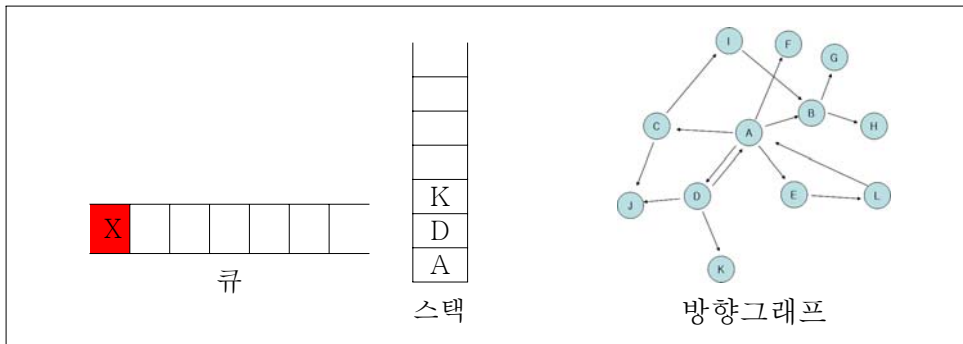


그림 3.15 여덟 번째 단계  
Fig. 3.15 The eighth process

- ⑧ 위의 과정을 반복하여 세션별 클릭스트림을 모두 검사한다.
- ⑨ 위의 예제에서는 새로운 정점 X가 탐색 되었을 때 얻은 패턴 후보는 A, D, K가 된다.

표 3.6은 새로운 정점 X에 대한 세션별 패턴후보이다. 새로운 정점에 대한 패턴후보의 간선추가는 패턴후보의 길이가 짧으면서, 카운터가 가장 많은 단말정점에 새로운 간선을 추가한다. 그 이유는 사용자는 새로운 정점을 가기위해 짧은 경로를 선택하는 사람이 있으며, 짧은 경로 중 빈발하게 사용되는 정점에 간선을 추가함으로써 뒤로 버튼을 최소화한 결과가 될 것이다. 실험에서 많은 클릭스트림을 사용하였을 때 새로운 정점에 대해 신뢰할 수 있는 간선을 100% 찾을 수 있었으며, 탐색 된 정점에 여러 간선이 존재할 경우 잡음은 발생하지만 그중에 신뢰할 수 있는 간선은 꼭 존재함을 알 수 있었다. 위의 예제에서 새로 탐색 된 정점에 대해서 패턴 후보의 크기가 가장 짧은 정점 즉, 짧은 경로(A→D, A→F) 이면서, 카운터가 가장 많은(A→D)후보의 단말경로에 새로운 정점 X를 간선으로 추가



한다. 그림 3.8은 표 3.7에서 새로운 정점 X를 패턴후보 중 짧은 경로인 A→D의 D정점에 새로운 간선 D→X를 추가한 결과이다.

표 3.6 세션에 따른 새로운 정점의 탐색 결과

Table 3.6 Results from discovering new vertices according to sessions

세션 구분	클릭스트림	패턴후보	새로운 정점
세션1	A B H D J K X	A D K	X
세션2	A C J D E L D X	A D	X
세션3	A B D X	A D	X
세션4	A D X	A D	X
세션5	A F B H E D X	A D	X
세션6	A C J D E L A D X	A E L A D	X
세션7	A D A X	A D A	X
세션8	A F B D K X	A D K	X
세션9	A B H G F	A F	X
세션10	A D K L X	∅	X

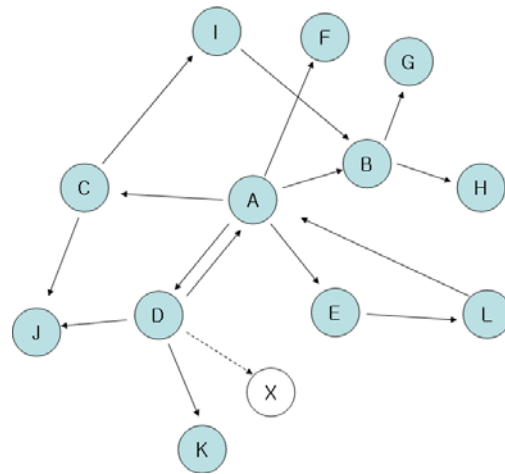


그림 3.16 제안된 방향그래프 갱신 결과

Fig. 3.16 The proposal result of directed graph update

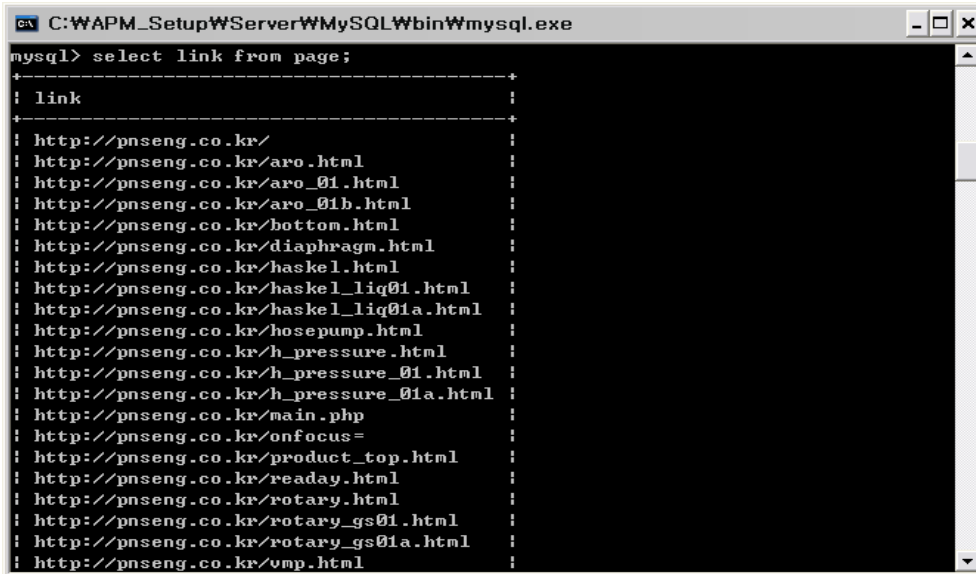
## 제 4 장 시스템 구현 및 실험

### 4.1 시스템 구현환경

본 논문에서 제안하는 시스템은 <http://www.pnseng.co.kr>의 웹 문서와 접근로그에서 각기 다른 사용자를 이용하여 2개월간 모아진 데이터를 기반으로 하여 테스트를 하였다. 웹서버는 Linux기반이고, DBMS는 Mysql을 사용하였다. CPU는 Intel P3 Dual, 메모리는 1G (ECC), 하드디스크는 SCSI 10G이다. 클라이언트 PC는 윈도우 XP 운영체제에서 실험하였다. 그리고 웹 로그는 보안의 문제가 있어 여러 조건에서 여러 웹사이트를 실험하기는 힘들기 때문에 좀 더 신뢰성을 보장하기위해 웹을 모형화하여 시뮬레이션도 병행하였다.

(1) 웹 문서에서 하이퍼링크 추출

그림 4.1은 처음 웹 문서에 포함되어있는 하이퍼링크를 통해 다음 페이지로 이동하면서 하나의 웹 문서에 연결되어 있는 모든 HTML페이지를 추출한 것이다. 웹 문서 순환탐색 알고리즘으로 추출한 정점은 25개, 간선은 32개이다.



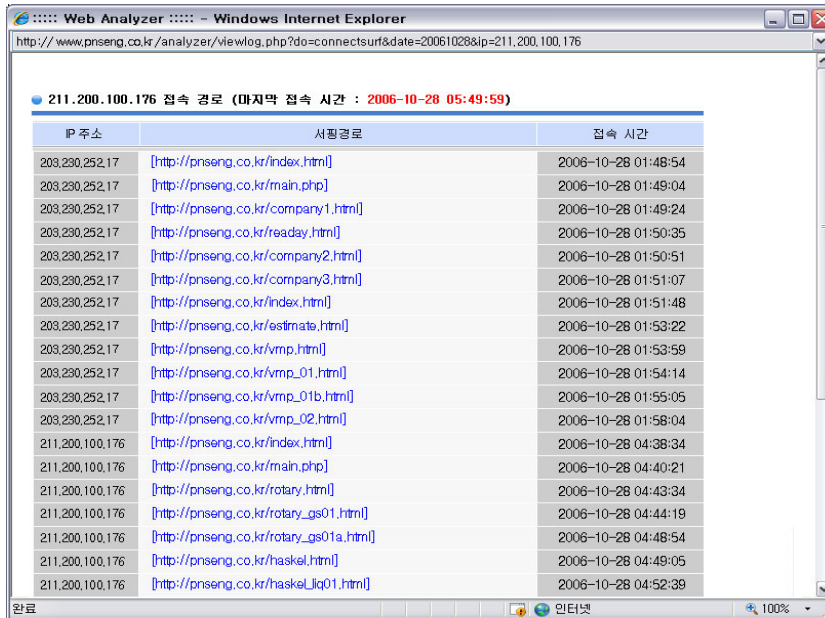
```
C:\WAPM_Setup\Server\MySQL\bin\mysql.exe
mysql> select link from page;
+-----+
| link                                     |
+-----+
| http://pnseng.co.kr/                   |
| http://pnseng.co.kr/aro.html           |
| http://pnseng.co.kr/aro_01.html        |
| http://pnseng.co.kr/aro_01b.html       |
| http://pnseng.co.kr/bottom.html        |
| http://pnseng.co.kr/diaphragm.html     |
| http://pnseng.co.kr/haskel.html        |
| http://pnseng.co.kr/haskel_liq01.html  |
| http://pnseng.co.kr/haskel_liq01a.html |
| http://pnseng.co.kr/hosepump.html      |
| http://pnseng.co.kr/h_pressure.html    |
| http://pnseng.co.kr/h_pressure_01.html |
| http://pnseng.co.kr/h_pressure_01a.html |
| http://pnseng.co.kr/main.php           |
| http://pnseng.co.kr/onfocus=          |
| http://pnseng.co.kr/product_top.html   |
| http://pnseng.co.kr/readay.html        |
| http://pnseng.co.kr/rotary.html        |
| http://pnseng.co.kr/rotary_gs01.html   |
| http://pnseng.co.kr/rotary_gs01a.html  |
| http://pnseng.co.kr/vmp.html           |
+-----+
```

그림 4.1 웹 문서로부터 하이퍼링크 추출

Fig. 4.1 The hyperlinks extracted from Web documents

## (2) 접근로그의 클릭스트림 추출

그림 4.2는 접근로그의 전처리과정을 통해 사용자 구분과 세션 구분을 적용하여 얻어진 클릭스트림이다.



The screenshot shows a web browser window titled "Web Analyzer" displaying a table of access logs. The table has three columns: "IP 주소" (IP Address), "서핑경로" (Surfing Path), and "접속 시간" (Access Time). The data is organized into two groups based on IP address: 203.230.252.17 and 211.200.100.176. The first group contains 14 entries, and the second group contains 6 entries. The table is sorted by access time in descending order.

IP 주소	서핑경로	접속 시간
203.230.252.17	[http://pnseng.co.kr/index.html]	2006-10-28 01:48:54
203.230.252.17	[http://pnseng.co.kr/main.php]	2006-10-28 01:49:04
203.230.252.17	[http://pnseng.co.kr/company1.html]	2006-10-28 01:49:24
203.230.252.17	[http://pnseng.co.kr/readay.html]	2006-10-28 01:50:35
203.230.252.17	[http://pnseng.co.kr/company2.html]	2006-10-28 01:50:51
203.230.252.17	[http://pnseng.co.kr/company3.html]	2006-10-28 01:51:07
203.230.252.17	[http://pnseng.co.kr/index.html]	2006-10-28 01:51:48
203.230.252.17	[http://pnseng.co.kr/estimate.html]	2006-10-28 01:53:22
203.230.252.17	[http://pnseng.co.kr/vmp.html]	2006-10-28 01:53:59
203.230.252.17	[http://pnseng.co.kr/vmp_01.html]	2006-10-28 01:54:14
203.230.252.17	[http://pnseng.co.kr/vmp_01b.html]	2006-10-28 01:55:05
203.230.252.17	[http://pnseng.co.kr/vmp_02.html]	2006-10-28 01:58:04
211.200.100.176	[http://pnseng.co.kr/index.html]	2006-10-28 04:38:34
211.200.100.176	[http://pnseng.co.kr/main.php]	2006-10-28 04:40:21
211.200.100.176	[http://pnseng.co.kr/rotary.html]	2006-10-28 04:43:34
211.200.100.176	[http://pnseng.co.kr/rotary_gs01.html]	2006-10-28 04:44:19
211.200.100.176	[http://pnseng.co.kr/rotary_gs01a.html]	2006-10-28 04:48:54
211.200.100.176	[http://pnseng.co.kr/haskel.html]	2006-10-28 04:49:05
211.200.100.176	[http://pnseng.co.kr/haskel_ljq01.html]	2006-10-28 04:52:39

그림 4.2 접근로그의 클릭스트림  
Fig. 4.2 Clickstreams of access logs

## (3) 방향 그래프 갱신

표 4.1은 본 논문에서 제안한 웹 문서의 하이퍼링크 추출을 통해 만들어진 방향그래프를 접근로그를 이용하여 히든하이퍼링크를 찾아내어 갱신된 방향그래프이다. 웹 문서의 실제구조는 정점은 35개 간선은 48개이다. 실험결과로 웹 문서 하이퍼링크에서 추출한 정점은 25개, 간선은 32개, 찾지 못한 정점은 10개, 간선은 16개이다. 접근로그를 이용하여 추가된 정점은 10개로 찾지 못한 정점을 모두 찾는데 성공하였고 추가된 간선은 14개로 찾지 못한 간선이 2개 존재하였다.

표 4.1 하이퍼링크의 정점

Table. 4.1 Vertices of hyperlinks

정점	경로	갱신
vertex0	pnseng.co.kr/	
vertex1	pnseng.co.kr/aro.html	
vertex2	pnseng.co.kr/aro_01.html	
vertex3	pnseng.co.kr/aro_01b.html	
vertex4	pnseng.co.kr/bottom.html	
vertex5	pnseng.co.kr/diaphragm.html	
vertex6	pnseng.co.kr/haskel.html	
vertex7	pnseng.co.kr/haskel_liq01.html	
vertex8	pnseng.co.kr/haskel_liq01a.html	
vertex9	pnseng.co.kr/hosepump.html	
vertex10	pnseng.co.kr/h_pressure.html	
vertex11	pnseng.co.kr/h_pressure_01.html	
vertex12	pnseng.co.kr/h_pressure_01a.html	
vertex13	pnseng.co.kr/main.php	
vertex14	pnseng.co.kr/product_top.html	
vertex15	pnseng.co.kr/readay.html	
vertex16	pnseng.co.kr/rotary.html	
vertex17	pnseng.co.kr/rotary_gs01.html	
vertex18	pnseng.co.kr/rotary_gs01a.html	
vertex19	pnseng.co.kr/vmp.html	
vertex20	pnseng.co.kr/vmp_01.html	
vertex21	pnseng.co.kr/vmp_01b.html	
vertex22	pnseng.co.kr/wilden.html	
vertex23	pnseng.co.kr/wilden_01.html	
vertex24	pnseng.co.kr/wilden_01b.html	
vertex25	pnseng.co.kr/company1.html	추가
vertex26	pnseng.co.kr/company2.html	추가
vertex27	pnseng.co.kr/company3.html	추가
vertex28	pnseng.co.kr/estimate.html	추가
vertex29	pnseng.co.kr/vmp_02.html	추가
vertex30	pnseng.co.kr/vmp_02b.html	추가
vertex31	pnseng.co.kr/haskel_liq02.html	추가
vertex32	pnseng.co.kr/haskel_liq02a.html	추가
vertex33	pnseng.co.kr/aro_02.html	추가
vertex34	pnseng.co.kr/aro_02b.html	추가

표 4.2 하이퍼링크의 간선

Table. 4.2 Edges of hyperlinks

정점	→	정점	갱신
vertex0	→	vertex13	
vertex13	→	vertex4	
vertex13	→	vertex1	
vertex13	→	vertex5	
vertex13	→	vertex6	
vertex13	→	vertex9	
vertex13	→	vertex10	
vertex13	→	vertex15	
vertex15	→	vertex13	
vertex13	→	vertex16	
vertex13	→	vertex19	
vertex13	→	vertex22	
vertex13	→	vertex14	
vertex1	→	vertex2	
vertex2	→	vertex3	
vertex6	→	vertex7	
vertex7	→	vertex8	
vertex10	→	vertex11	
vertex11	→	vertex12	
vertex16	→	vertex17	
vertex17	→	vertex18	
vertex19	→	vertex20	
vertex20	→	vertex21	
vertex22	→	vertex23	
vertex23	→	vertex24	
vertex3	→	vertex13	
vertex8	→	vertex13	
vertex12	→	vertex13	
vertex18	→	vertex13	
vertex21	→	vertex13	
vertex24	→	vertex13	
vertex5	→	vertex13	
vertex13	→	vertex25	추가
vertex13	→	vertex26	추가
vertex25	→	vertex26	추가
vertex13	→	vertex28	추가
vertex13	→	vertex29	추가
vertex19	→	vertex29	추가
vertex29	→	vertex30	추가
vertex13	→	vertex31	추가
vertex31	→	vertex32	추가
vertex1	→	vertex33	추가
vertex33	→	vertex34	추가
vertex26	→	vertex0	추가
vertex13	→	vertex27	추가
vertex26	→	vertex27	추가

## 4.2 시뮬레이션

본 연구에서 웹 로그는 보안의 문제가 있어 여러 조건에서 여러 웹사이트를 실험하기에는 힘들기 때문에 좀 더 신뢰성을 보장하기 위해 웹을 모형화하여 시뮬레이션 하였다.

구현은 본 논문에서 제안한 방법과 같이 진행하였다. 먼저 웹 문서에서 하이퍼링크를 추출하는 과정은 랜덤하게 100개의 정점과 1,787개의 간선을 방향성 그래프로 만들었다. 접근로그에서 추출되는 클릭스트림은 앞의 과정에서 만든 방향그래프를 탐색하며 사용자들이 웹 향해하는 방법과 같이 '뒤로'버튼을 랜덤하게 추가하여 실제 웹 환경과 똑같이 클릭스트림 패턴을 50,000개 만들어 실험하였다. 실험은 히든정점을 찾아 신뢰성 있는 간선만을 찾는 실험과, 히든정점에 대해 모든간선을 찾는 실험을 하였다. 평가는 식 4.1의 정확률(Precision)과 식 4.2의 재현률(Recall), 식 4.3의 정확도(Accuracy), 식 4.4의 오류율(Error)을 계산하고 평가하였다. 표 4.3 분할표는 실제구조를 정의한 정점, 간선과 본 논문의 제안을 통해 찾은 구조의 정점, 간선을 의미한다.

표 4.3 분할표

Table. 4.3 Contingency table

실제구조 \ 찾은 구조		찾은 구조	
		있을 때	없을 때
실제구조	있을 때	A	B
	없을 때	C	D

정확률은 본 논문에서 제안한 알고리즘을 통해 찾은 정점과 간선의 범주에서 실제 웹 구조와 일치하는 비율이며 식 4.1과 같다.

$$\text{정확률(Precision)} : \frac{A}{A+B} \quad (\text{식 4.1})$$

재현률은 실제구조의 정점과 간선 범주에서 찾은 구조의 정점과 간선이 일치하는 비율을 나타내며 식 4.2와 같다.

$$\text{재현률(Recall)} : \frac{A}{A+C} \quad (\text{식 4.2})$$

정확도는 전체 구조에서 일치하게 분류된 정점과 간선개수의 비율이며 식 4.3과 같다.

$$\text{정확도(Accuracy)} : \frac{A+D}{A+B+C+D} \quad (\text{식 4.3})$$

오류율은 전체 구조에서 잘못 분류된 정점과 간선개수의 비율이며 식 4.4와 같다.

$$\text{오류율(Error)} : \frac{B+C}{A+B+C+D} \quad (\text{식 4.4})$$



(1) 히든정점의 신뢰성 있는 간선을 찾는 실험

히든 정점과 간선은 만들어진 방향그래프에서 랜덤하게 제거하여 본 논문의 제안을 통해 찾아내고 비교하여 실험하였다. 표 4.4는 실험 결과이다.

표 4.4 히든정점에 대한 신뢰도가 가장 높은 간선

Table. 4.4 The highest confidential edge for the hidden vertex

	실제	갱신	히든	추가	일치	불일치	정확률 Precision	재현률 Recall	정확도 Accuracy	오류율 Error
간선	1787	1429	358	100	97	3	0.97	0.27	0.27	0.58
정점	100	80	20	20	20	0	1.00	1.00	1.00	0.00

표 4.4를 살펴보면 실제구조의 정점은 100개이고 이중 히든정점은 20개이다. 본 논문에서 제안하는 방법을 통해 히든정점 20개 모두 찾을 수 있었다. 그리고 실제구조의 간선은 1,787개이고 이중 히든간선은 358개이다. 찾아낸 히든간선은 100개이고 그중 일치하는 히든간선은 97개를 찾을 수 있었다. 히든간선은 하나의 정점에 여러 간선이 있을 경우 모든 간선은 찾는 것이 아니라 신뢰성이 보장된 최소 하나의 간선은 찾는다. 모든 간선을 찾지 못하는 이유는 정점을 기준으로 신뢰성 있는 간선 즉, 빈발도가 높으면서 패턴경로가 짧은 간선에 히든간선을 추가하기 때문에 전체 정점의 개수보다 많은 간선은 찾을 수 없다. 그리고 탐색 한 히든정점에 대해서는 신뢰할 수 있는 간선을 찾을 수 있었다. 실험결과를 살펴보면 정점에 대한 정확률과 재현률, 정확도는 1로써 정확한 정점을 찾을 수 있음을 알 수 있고, 간선에 대한 정확률은 찾아 낸 간선 100개가 실제 히든간선과 97개와 일치하기 때문에 높은 수치를 보였다. 하지만 재현률과 정확도는 히든 간선 358개 중 97개만을 찾았기 때문에 낮은 비율을

보였다. 모든 히든간선을 찾기 위해서는 본 논문에서 제안한 히든링크탐색 알고리즘을 탐색된 간선을 추가하면서 새로운 간선이 나타나지 않을 때 까지 반복하여 수행하면 된다. 이 과정은 표 4.5의 실험결과에서 살펴 보겠다.

## (2) 히든정점의 모든 간선을 찾는 실험

표 4.5의 결과는 히든링크탐색 알고리즘을 히든간선을 모두 찾을 때까지 반복하여 수행한 결과이다. 그러나 일치하는 모든 간선은 찾을 수 있지만 불일치하는 간선도 같이 찾을 수 있었다. 본 논문의 제안은 새로운 정점으로 가기위해 패턴후보 길이가 짧으면서 빈발도가 많은 간선에 추가하므로 만약 일치하지 않는 간선이 새롭게 만들어지면 그 간선으로 인해 새로운 정점까지 가는 최단간선이 만들어질 수 있는 문제점이 생긴다. 표 4.5의 결과에서는 열 번째 단계에서 히든간선을 탐색 하지 못하였기 때문에 반복수행을 멈추었고, 아홉 번째의 단계에서 모든 간선을 찾을 수 있었다.

표. 4.5 히든정점에 대한 모든 간선

Table. 4.5 The edges for the hidden vertices

단계	간선 정점	실제	갱신	히든	추가	일치	불일치	정확률 Precision	재현률 Recall	정확도 Accuracy	오류율 Error
1	간선	1787	1429	358	100	97	3	0.97	0.27	0.27	0.58
	정점	100	80	20	20	20	0	1.00	1.00	1.00	0.00
2	간선	1787	1529	358	94	89	5	0.96	0.52	0.51	0.33
	정점	100	80	20	0	0	0	1.00	1.00	1.00	0.00
3	간선	1787	1623	358	78	72	6	0.95	0.72	0.69	0.18
	정점	100	80	20	0	0	0	1.00	1.00	1.00	0.00
4	간선	1787	1701	358	69	51	18	0.91	0.86	0.79	0.12
	정점	100	80	20	0	0	0	1.00	1.00	1.00	0.00
5	간선	1787	1770	358	72	23	49	0.80	0.93	0.76	0.14
	정점	100	80	20	0	0	0	1.00	1.00	1.00	0.00
6	간선	1787	1842	358	79	15	64	0.71	0.97	0.69	0.18
	정점	100	80	20	0	0	0	1.00	1.00	1.00	0.00
7	간선	1787	1921	358	50	7	43	0.65	0.99	0.65	0.21
	정점	100	80	20	0	0	0	1.00	1.00	1.00	0.00
8	간선	1787	1971	358	22	3	19	0.63	1.00	0.63	0.23
	정점	100	80	20	0	0	0	1.00	1.00	1.00	0.00
9	간선	1787	1993	358	7	1	6	0.63	1.00	0.63	0.23
	정점	100	80	20	0	0	0	1.00	1.00	1.00	0.00
10	간선	1787	2000	358	0	0	0	0.00	0.00	0.00	1.00
	정점	100	80	20	0	0	0	1.00	1.00	1.00	0.00
평균	간선							0.80	0.81	0.62	0.24
	정점							1.00	1.00	1.00	0.00

그림 4.3은 히든링크탐색 알고리즘을 히든간선을 모두 찾을 때까지 단계별로 나타낸 정확률 그래프이다. 정확률은 정답 범주일거라고 예측된 링크가 실제로 정답일 확률이다. 1단계에서는 0.97의 비율을 나타내지만 4단계 이후는 급격히 정확률이 떨어졌다. 이유는 일치하지 않는 간선의 추가로 인해 새로운 정점까지 가는 최단 간선이 만들어져 많은 잡음이 급격히 늘어남을 볼 수 있다.

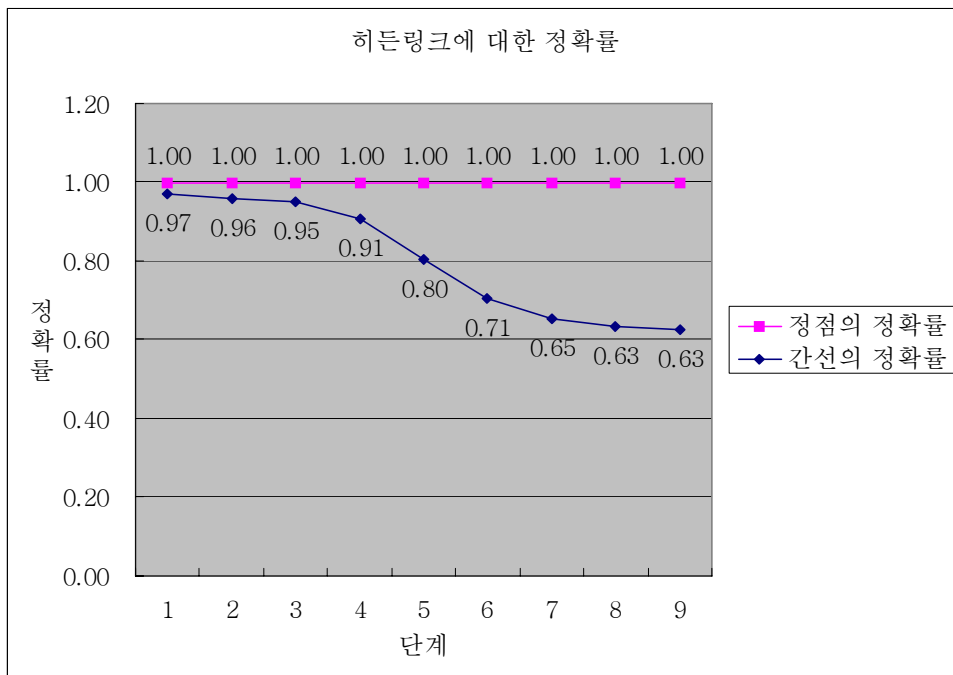


그림 4.3 히든링크에 대한 정확률 그래프

Fig. 4.3 The graph of precision for the hidden links

그림 4.4는 히든링크탐색 알고리즘을 히든간선을 모두 찾을 때까지 단계별로 나타낸 재현률 그래프이다. 재현율은 정답 범주에 속하는 링크가 정답으로 분류될 확률이다. 1단계에서는 정점을 기준으로 찾을 수 있는 간선의 수가 100개 이하이기 때문에 358개를 모두 찾지 못하고 단계별로 추가를 하며 찾아간다. 4단계까지는 빠르게 증가하지만 4단계 이후는 일치하지 않는 간선들의 잡음으로 인해 불일치하는 간선이 많아져 낮은 증가율을 보이고, 9단계에서는 모든 히든간선을 찾은 결과를 보여준다.

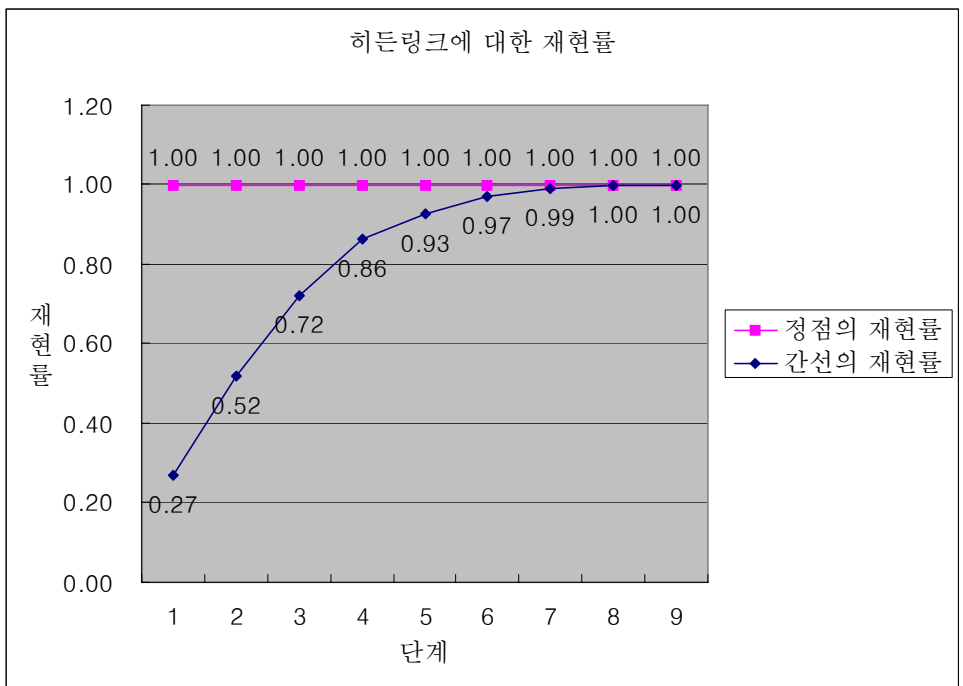


그림 4.4 히든링크에 대한 재현률 그래프  
 Fig. 4.4 The graph of recall for the hidden links

그림 4.3은 히든링크탐색 알고리즘을 히든간선을 모두 찾을 때까지 단계별로 나타낸 정확도 그래프이다. 정확도는 전체 실험 데이터에서 바르게 분류된 데이터 개수의 비율이다. 1단계에서는 정점을 기준으로 찾을 수 있는 간선의 수가 100개 이하이기 때문에 358개를 모두 찾지 못하므로 낮은 정확도를 보여주고 있다. 4단계까지는 신뢰성 있는 간선의 추가로 정확도가 빠르게 높아지지만, 4단계 이후부터는 일치하지 않은 간선들의 잡음으로 정확도가 줄어드는 것을 알 수 있다.

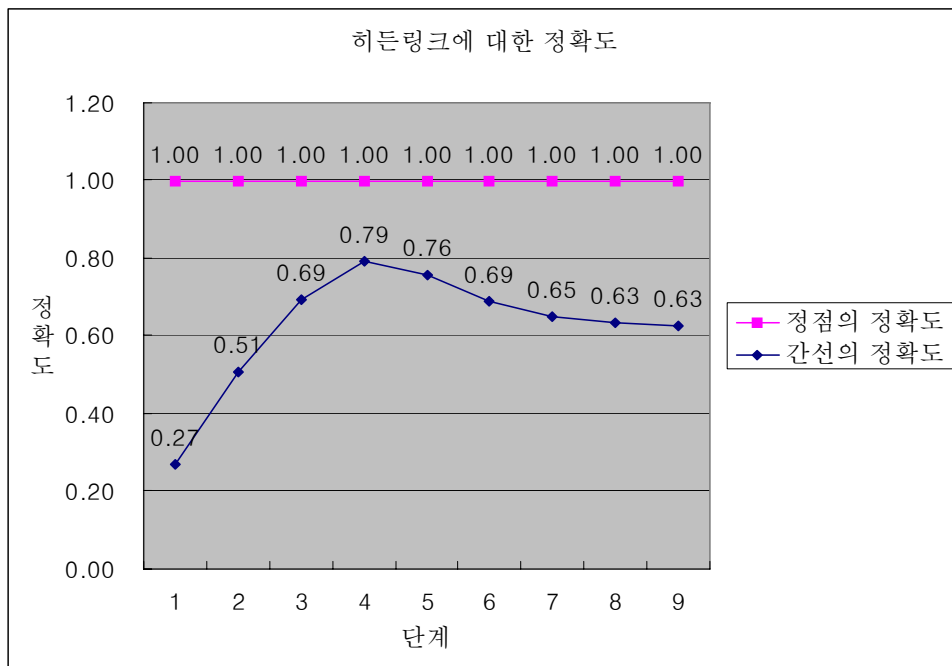


그림 4.5 히든링크에 대한 정확도 그래프

Fig. 4.5 The graph of accuracy for the hidden links

그림 4.3은 히든링크탐색 알고리즘을 히든간선을 모두 찾을 때까지 단계별로 나타낸 오류 그래프이다. 오류는 전체 실험 데이터에서 잘못 분류된 데이터 개수의 비율이다. 1단계에서는 정점을 기준으로 찾을 수 있는 간선의 수가 100개 이하이기 때문에 358개를 모두 찾지 못한다. 그래서 많은 오류를 나타내고 있다. 4단계까지는 신뢰성 있는 간선의 추가로 오류가 빠르게 줄어들지만, 4단계 이후부터는 일치하지 않은 간선들의 잡음으로 조금씩 오류가 늘어남을 알 수 있다.

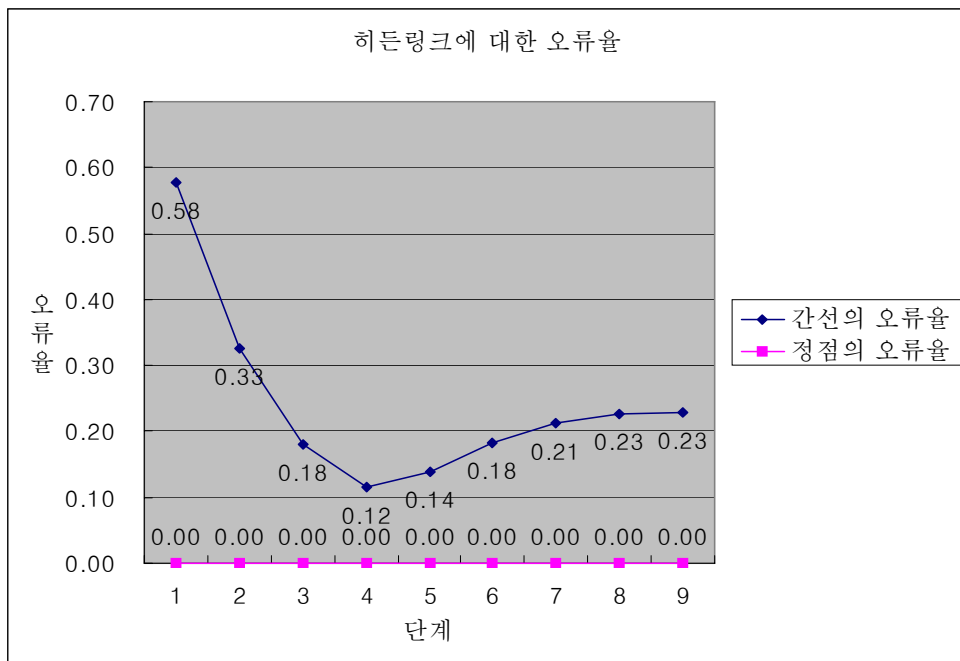


그림 4.6 히든링크에 대한 오류율 그래프

Fig. 4.6 The graph of error for the hidden links

결과적으로 모든 간선을 찾을 때까지 히든링크 탐색 알고리즘을 반복 수행하는 것보다 신뢰할 수 있는 구간을 4단계까지로 하여 알고리즘을 반복수행하는 것이 좀 더 신뢰할 수 있는 결과를 얻을 수 있다.

## 제 5 장 결론 및 향후 과제

본 논문에서는 웹 사이트를 구조화 하는 방법으로 웹 문서의 하이퍼링크와 접근로그를 사용하여 구조화하는 방법을 제시하였다. 웹 문서의 태그분석을 통해 하이퍼링크만 추출하여 웹을 구조화하는 기존의 방법을 향상시키는 방법으로서, 접근로그 정제과정을 거쳐 세션별 클릭스트림을 추출하고 웹 문서에서 만들어진 구조를 이용한다. 이를 통해 큐와 스택의 구조적 방법을 사용하여 히든 하이퍼링크를 찾아내고 새로운 정점과 간선을 방향그래프의 구조로 생성하는 방법을 제시하였다. 결과적으로 웹 문서의 태그분석으로 알 수 없었던 기존의 방식을 개선시켜 본 논문에서 제시한 알고리즘을 통해 히든 하이퍼링크를 찾아 이를 추가하는 것이 가능하였고 구조화를 통해 사용자의 행동패턴을 관찰하고 웹 사용의 효율을 증대시킬 수 있을 것이다. 이것은 다양한 웹 구조개선 및 웹 마이닝을 위한 핵심 자료로 활용될 뿐 아니라 쉽고 빠른 검색을 사용자들에게 제공할 수 있으며 웹 문서의 효율적인 개발과 유지보수를 가능하게 하여 관리자가 쉽게 접근할 수 있는 구조가 된다.

향후연구 과제로서 본 논문이 브라우저의 특성에 맞는 알고리즘을 제시했음에도 불구하고 현재의 웹 브라우저에서 HTML 이외의 웹 문서들은 정확한 하이퍼링크 경로를 찾기가 그 구조적 특성상 어려움이 있어 이를 연구하여 보완할 필요성이 있다. 그리고 접근로그를 통해 새로운 간선을 추가하거나 정점 하나에 여러 간선이 존재할 경우, 하나의 간선에 대한 경로 탐색은 신뢰성을 가지지만 그 이상의 간선에 대해서는 잡음이 발생하는 문제점이 존재했다. 이 잡음을 줄이기 위해 웹 구조에 대한 사용자별 패턴 분석 연구가 필요할 것이다. 그 외 연구로서, 웹 사이트에서 추출한 여러 가지 정보의 활용도를 높이기 위한 구조적이면서 쉽게 이해될



수 있는 표현 방법에 대한 연구와, 다양한 뷰를 제공할 수 있는 방법들에 대한 연구가 필요하다.

## 참 고 문 헌

- [1] Y. Zou and K. Kontogiannis, "Migrating and Specifying Services for Web Integration", Lecture Notes In Computer Science, Vol. 1999, pp. 253-270, 2000.
- [2] M. Baglioni, U. Ferrara, A. Romei, S. Ruggieri and F. Turini, "Preprocessing and Mining Web Log Data for Web Personalization", Lecture Note in Computer Science, Vol. 2829, pp. 237-249, 2003.
- [3] S. D. Lee and H. C. Park, "Mining Frequent Patterns from Weighted Traversals on Graph using Confidence Interval and Pattern Priority", International Journal of Computer Science and Network Security, Vol. 6, No. 5A, pp. 136-141, 2006.
- [4] Y. Kosala and H. Blockeel, "Web Mining Research, A Survey", Newsletter of the Special Interest Group on Knowledge Discovery & Data Mining, Vol. 2(1), pp. 1-15, 2000.
- [5] J. Huysmans, B. Baesens and J. Vanthienen, "Web Usage Mining: A Practical Study", Proceedings of the Twelfth Conference on Knowledge Acquisition and Management, pp. 86-99, 2004.
- [6] R. Cooley, B. Mobasher and J. Srivastava, "Data Preparation for Mining World Wide Web Browsing Patterns", Knowledge and Information System, Vol. 1, Springer Verlag, pp. 1-26, 1999.
- [7] M. S. Chen, "Efficient Data Mining for Path Traversal Pattern",

- IEEE Transactions on Knowledge and Data Engineering, Vol. 10, No. 2, pp. 209-221, 1996.
- [8] Jop F. Sibeyn, J. Abello and U. Meyer, "Heuristics for semi-external depth first search on directed graphs", Association for Computing Machinery Symposium on Parallel Algorithms and Architectures, Vol. 52, pp. 282-292, 2002.
- [9] S. Chakrabarti, Mining the Web, Morgan Kaufmann Pub., 2002.
- [10] Thuraisingham and Bhavani M., Web Data Mining and Business Intelligence Analysis, CRC Press Pub., 2003.
- [11] G. Nivasch, "Cycle Detection Using a Stack", Information Processing Letters, Vol. 90, pp. 135-140, 2004.
- [12] M. Koutri, N. Avouris and S. Daskalaki, "A Survey on Web Usage Mining Techniques for Web-Based Adaptive Hypermedia System", Information Resources Management Association, Vol. 13, pp. 125-149, 2005.
- [13] D. Embley, C. Tao and S. Liddle, "Automating the Extraction of Data from HTML Table with Unknown Structure", Data & Knowledge Engineering, Vol. 54, pp. 3-28, 2005.
- [14] 박종수, 윤지영, "웹 로그 파일에서 순회 패턴 탐사를 위한 시스템", 한국정보과학회 추계 학술발표 논문집, 제28권 제2호, pp. 4-6, 2001.
- [15] 김진수, 김태용, 최준혁, 임기욱, 이정현, "사용자 로그 분석과 클러스터 내의 문서 유사도를 이용한 동적 추천 시스템", 한국정보과학회 정보과학논문지, 소프트웨어 및 응용 제31권 제5호, pp.

586-594, 2004.

- [16] 최영환, 이상용, "웹 마이닝을 위한 입력 데이터의 전처리과정에서 사용자구분과 세션보정", 한국정보과학회 정보과학회논문지, 소프트웨어 및 응용, 제30권 제9.10호, pp. 843-849, 2003.

## 감사의 글

마지막 수정작업을 마치고 나니 마음은 한결 편안해졌지만 이 논문을 쓰던 시간들을 돌이켜보면, 이렇게 쓰면 되겠지 라는 안일한 생각으로 많은 시간을 보내고 있었습니다. 시간이 흘러 편집창을 열었을 땐 한 문장도 쉽게 나아가질 않았고 논문을 쓴다는 것은 쉬운 일이 아니란 것을 새삼스럽게 느낄 수 있었습니다. 이렇게 철없고 부족한 저를 졸업할 수 있게끔 도움을 주신 많은 분들이 없었다면 지금 이 글을 적지 못하고 뒤늦은 후회와 한숨만 내쉬며 시간을 보내고 있었을지 모릅니다. 먼저 대학원생활을 무사히 마칠 수 있게 멀리서 물심양면으로 이끌어주신 박휴찬 교수님께 감사를 드립니다. 또한 지도교수님의 빈자리를 세심한 지도와 가르침을 주신 김재훈 교수님께 감사를 드립니다. 그리고 바쁘신 중에도 본 논문이 작은 결실을 맺을 수 있도록 관심과 지도를 해주신 류길수 교수님과 이장세 교수님께도 감사를 드립니다. 학부과정에서 많은 가르침을 주신 손주영 교수님, 신옥근 교수님께도 감사를 드립니다. 그리고 함께한 시간은 짧았지만 가르침을 주신 이서정 교수님께도 감사드립니다.

석사과정을 무사히 마칠 수 있게끔 안 밖으로 많은 지도와 격려로 이끌어 주신 이성대 형님께 진심으로 감사의 마음을 전합니다. 또한 파트타임으로 멀리서 관심과 지도를 해주신 곽용원 형님께도 감사드립니다. 졸업은 함께 하지 못해 아쉽지만 대학원생활 동안 옆에서 힘이 되어준 유흥섭 형님께도 고마움을 전합니다. 같이 석사과정에 들어와 지금은 한가정의 가장이되어 행복한 결혼생활을 하시는 감승철 형님 감사드립니다.

학부 때부터 지금까지 환한 웃음으로 따뜻하게 대해주시고 많은 도움을 주신 강군호 조교님, 김경언 조교님께 고마움을 전합니다. 그리고 따뜻한 관심과 도움을 주신 하동경 형님과 박정임 형수님, 유강주 형님께 감사드

립니다. 인공지능 연구실의 큰형님으로 많은 관심과 격려를 해주신 성장규 형님과 대학원생활 동안 동고동락하며 든직한 형님이자 친구처럼 다정하게 챙겨준 박종일 형님께 감사드립니다. 언제나 활발한 웃음을 잃지 않는 네트워크 연구실의 문성미님께 감사함을 전합니다. 함께 즐거워하며 웃음을 나누었던 자연언어처리의 서형원님과 조희영님에게 감사함을 전합니다. 그 외 학부와 대학원 생활을 하는 동안 함께한 선배님과 후배님 그리고 동기들에게 고마움을 전합니다.

같은 연구실은 아니었지만 언제나 따뜻하게 반겨주고 챙겨준 원라경 누나와 후배 김태진님에게 감사를 전합니다. 그리고 대학원 석사과정을 함께 생활하며 함께 마무리 하는 우리 연구실의 든직한 막둥이 전성환님에게 감사를 전합니다.

마지막으로, 제 모든 힘의 원천이며 살아가는 원동력인 사랑하는 가족 모두에게 감사드립니다. 여러분들의 도움이 없었다면 이 논문은 완성되지 못했을 것입니다.