한국산업정보학회논문지 제6권 제2호 (2001. 6)

인터넷 정보 추출을 이용한 웹문서 구조화
(Web Site Construction Using Internet Information Extraction)

박 은주*, 임 한규**
(Eun-Ju Park Han-kyu Lim)

요 약 본 논문은 웹사이트에서 문서를 삽입하거나 삭제할 경우, <A HREF>태그를 생성하는 수동적인 방법을 사용하지 않고, 자동적인 방법으로 문서를 삽입·삭제하는 알고리즘을 제안한다. 자동적인 방법으로 문서를 삽입·삭제하기 위하여 문서의 HTML 태그 중 문서와 문서를 연결하여 주는 <A HREF>태그를 사용한다. 그래프 구조의 상하계층은 한 문서의 <A HREF>...</A>태그사이의 텍스트들을 추출하여 추출된 텍스트를 이 문서의 하위노드의 노드명으로 링크를 생성하는 방법을 사용한다.
웹문서들간의 링크를 새로이 설정하고자 하는 경우 <A HREF>태그를 생성하지 않고 구조화된 그래프 형태를 이용하여, 그래프에서 노드를 삽입하거나 삭제한 후 새로운 구조를 웹에 적용한다. 문서를 삭제할 때에는 삭제될 노드와 링크되어 있는 노드들에 대하여 삭제되는 노드의 부모노드와의 링크를 새로이 설정해 줌으로써 단절 링크를 없애준다.

Abstract In this paper, we suggest the algorithm that inserts or deletes documents into web sites without creating <A HREF> tag. This algorithm uses <A HREF> tag which links between documents to automatically insert or delete the web documents. This study extracts the texts in the <A HREF>...</A> tag of the document to put into the structure as a type of graph creating the link as a node name of sub-node.

That is, in this case of configurating new link between web documents, this algorithm allows to insert or delete the node to or from the graph without creating the <A HREF> tag. In the case of deleting the document, it removes the broken link connecting the sub-nodes of deleted node newly to its parent node.

1. 서 론

HTML은 인터넷상에서 다양한 하이퍼텍스트(HyperText) 문서를 만들기에 적합하고, 웹을 일반화하는데 중요한 역할을 수행해왔다[1]. HTML로 작성되어진 웹문서에서 문서간의 링크는 <A HREF="URL">...</A>라고만 쓰면 간단하게 링크를 구현할 수 있다. 그러나 링크가 설정된 문서에 대하여 문서간 링크를 다시 구성하고자 할 경우에도 <A HREF="URL">...</A>태그를 사용하여 링크설정을 새로이 구성해 주어야만 한다.

웹사이트가 점점 더 커짐에 따라 새로운 웹 사이트를 디자인하고, 일일이 웹 페이지들을 만들고 그들간의 링크를 다시 만들어 갱신하는 작업은 몹시도 지루한 작업이 되었고, 전통적으로 하이퍼텍스트들은 하이퍼 텍스트 링크들을 만드는 것에 요구되는 수동의 노력에 의해 크기에 많은 제한이 있어왔다[2-3].

하이퍼텍스트는 비순차적 문서이며, 각각의 노드가 일정량의 텍스트나 다른 자료를 포함하고 있는 방향성 그래프라 할 수 있다. 웹을 노드와 링크를 가진 그래프 구조로 표현하고 링크들을 재조정하여 그래프 구조 자체를 변화시키는 것이 가능하다[4].

본 논문은 웹공간을 그래프를 사용하여 시각화하고, 웹문서를 그래프로 표현하기 위하여 HTML 문서의 태그들 중 문서를 연결하는 태그인 <A HREF="URL">...</A>인 태그들을 추출한다. 태그를 추출한 문서를 부모노드로, <A HREF="URL">...</A> 태그 사이에서 추출된 텍스트를 자식 노드의 노드명으로 두 노드간의 링크가 생성된다. 웹사이트의 모든 문서들에 대하여 링크를 설정함으로써 웹사이트를 그래프 구조로 구조화한다. 웹사이트에서 문서를 삽입하거나 삭제하고자 할 경우, 구조화된 그래프 형태를 사용하여 그래프에서 노드를 삽입, 삭제한다.

본 논문은 문서간 링크생성 시 수동적인 방법으로 <A HREF="URL">...</A>태그를 생성하여 주지 않고 구조화된 그래프 구조의 링크를 재조정함으로써 자동적인 방법으

* 안동대학교 컴퓨터공학 전공

** 안동대학교 전자정보산업학부 부교수

로 문서간의 링크를 생성시키고, 문서 삭제시 삭제되는 문서에 링크되어 있는 문서들에 대한 링크를 재설정함으로써 단절 링크(broken link)의 문제를 해결한다.

2. 기반연구

2.1 하이퍼텍스트

웹은 <그림 1>에서 보여주는 것처럼 파일 시스템의 전통적인 구조와 하이퍼텍스트의 복잡한 연결구조인 2개의 환경들 내에서 존재한다. 이 2개의 환경들 내에서 리소스들 사이의 관계들뿐만 아니라 리소스들에서 인터페이스들도 기본적으로 다르다[5].

하이퍼텍스트는 비순차적 문서이며, 각각의 노드가 일정량의 텍스트나 다른 자료를 포함하고 있는 방향 그래프(Directed graph)라고 할 수 있다. 하이퍼텍스트 문서의 기본적인 정보단위는 노드라고 불리며, 텍스트와 그래픽스 또는 기타 다른 형태의 데이터의 복합체일 수 있다.

정의 1 (노드) 노드(Node)는 노드를 다른 노드와 구별시켜주는 노드번호(NodeID)와 제목(Title), 그리고 개념설명(Description)의 튜플이다[6].

$$Node=(NodeID,\ Title,\ Description)$$
$$Title(T)=[tw_1,\ tw_2,\ tw_3,\ ...,\ tw_t]$$
$$Description(D)=[dw_1,\ dw_2,\ dw_3,\ ...,\ dw_t]$$

노드는 링크라고 불리는 상호참조기구(Cross-reference)에 의하여 연결된다. 특별한 주제에 관해 더 많은 정보를 얻기 위해서 사용자는 노드 중에 존재하는 앵커를 활성화시킨다. 앵커는 노드 내부 링크의 존재를 나타내는 노드의 일부분으로, 앵커가 활성화되면 컴퓨터 시스템은 현재 노드를 링크의 목적 노드로 대치하여 화면을 변화시킨다.

정의 2 (링크) 링크(link)는 노드 중의 일부분인 앵커와 다른 노드를 연결시켜주는 구조이며, 앵커와 노드번호의 튜플로 정의할 수 있다[6].

$$Link=(Anchor,\ NodeID)$$

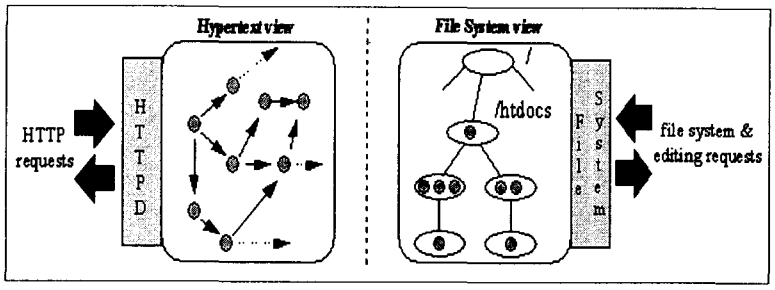
2.2 정보 시각화

최근 엄청난 속도로 증가하고 있는 인터넷, 특히 월드와이드 웹의 정보는 다양한 부분에서 지식의 원천으로 사용되고 있으며, WWW 정보공간은 하루가 다르게 팽창되고 증가하고 있다. 이에 따른 정보량도 기하급수적이라고 할 수 있고, WWW를 이용하는 이용자의 수도 증가하고 있다. 이러한 이용자들은 웹브라우저(예, Netscape Communicator, Internet Explorer)를 이용하여 정보접근과 브라우징을 하고 있다[7].

정보공간의 전체적인 구조와 내용을 하나의 화면상에 시각적으로 표현하게 되면, 이용자는 시각적인 표현을 이용하여 그들이 정보공간의 어느 부분에 있고, 어떤 정보가 이용이 가능하며 다른 정보를 어떻게 접근해야 하는지를 쉽게 파악할 수 있다. 이런 것들을 가능하게 한 것이 여러 가지 시각화 기술을 기반으로 정보를 2차원, 3차원으로 표현하는 웹시각화이이며, 더 쉽게 정보를 얻을 수 있도록 웹 문서 구조를 보다 시각적으로 나타낸다[8].

정보 시각화(Information Visualization)의 목적은 복잡하고 차원이 많은 정보 데이터(Information Data)를 이해하기 쉽게 그림이나 도표와 같은 특정한 형식을 이용하여 효과적으로 나타내고 비교하는데 있다.

이 분야의 초기연구들은 입자의 운동이나 공기의 흐름과 같은 물리적 현상들을 시각화에 치중하였으나, 최근의 연구들은 데이터베이스나 소프트웨어 구조같이 훨씬 추상적인 정보들을 시각화하는데 초점이 맞춰지고 있다. 이들 구조들은 단지 메모리나 디스크 안에 존재하는 정보일 뿐이지 실제 형태를 가지고 있는 것이 아니다. 이들 구조들에 어떤 형태를 주기 위한 연구와 사람들이 이들 정보구조를 쉽게 접근하고 질의할 수 있도록 해주는 환경을 만드는



<그림 1> 웹 리소스들의 서로 다른 인터페이스

연구들이 많이 진행되고 있다[9].

정보시각화 방법에는 Focus+Context 방법과 Widgets for Information Visualization 방법이 있다.

Focus+Context방법은 많은 정보를 모두 표시하면서도 특정 영역에 초점을 맞추고 확대하여 볼 수 있게 하는 기법을 말한다. 이 방법에는 Fisheyes Views, Cone Tree, Perspective Wall, Tree Maps 등의 방법이 있다.

Widgets for Information Visualization 방법은 이용자와의 상호작용을 통해 초점을 선정하고 구체적인 질의 조건을 통해 초과 정보를 filtering 하는 것으로 이 기술은 위에서 제시한 기법들과 함께 사용하며, 이 방법에는 Alphaslider 방법과 Zoom bar 방법이 있다.

2.3 인터넷 정보추출

정보추출은 한 문서에서 그 문서의 중심적 의미를 나타내는 특정 구성요소를 인식하여 추출하는 작업을 의미한다. 정보추출의 예로는 날씨 정보를 제공하는 웹 문서로부터 지역, 날짜, 최고온도, 최저온도, 습도 등의 정보를 뽑아내거나, 아파트 정보 문서로부터 방의 개수, 매매가, 전세가, 전화번호 등을 추출하는 것을 들 수 있다.

인터넷 정보추출 작업은 본질적으로 확장성이 없는데, 그 이유는 정보소스의 문서들은 원칙적으로 사람들이 읽기 편하도록 작성되었기 때문에 프로그램이 쉽게 처리할 수 있도록 문서 구성시의 포맷 관행 등에 대한 정보를 제공해 주는 사이트가 거의 없고, 한 사이트에서 사용된 독특한 포맷관행이 다른 사이트에도 적용될 가능성이 거의 없고, 사이트들이 포맷을 자주 바꾸기 때문에 이전에 만들었던 wrapper가 동작하지 않는 결과들을 가져온다[10-11].

추출 작업을 하는 문서는 HTML로 작성된 문서이며 같은 스타일이라고 하더라도 여러 가지 다른 방법으로 기술하는 것이 가능하다. 예를 들어, <FONT color=red><BIG>은 <FONT size=+1 color=#ff0000>와 똑같은 효과를 얻는다[10].

2.4 단절 링크

단절 링크는 HTML 페이지들의 위치를 바꾸거나 상위 문서의 삭제시 외부의 사이트들(검색 엔진들, 주제별 카탈로그 등)로부터의 링크들을 표현할 수 없어지는 것으로 끊어진 링크들은 인터넷 항해자들에게는 웹항해를 중단하게 하는 요인이 된다. 단절 링크를 막기 위한 방법으로 D. Ingham의 Object-Oriented 방법[5]은 웹을 객체로 정의하여, 한 객체에서 다른 객체로의 하이퍼링크를 나타내는 동안, 링크의 불필요성을 나타낼 때까지는 객체들을 삭제할 수도, 다른 객체들로부터의 하이퍼텍스트 링크들의 삭제도 가질 수 없는 방법으로 단절링크를 막는 방법을 제안하였고, John Garaofalakis는 페이지의 URL의 맵을 새로운 URL로 대체하고, 데이터베이스(웹서버로부터 접근이 가능한 간단한 파일이나 관계 데이터베이스)에 저장한다. 그리고, 관리자는 사용자가 업데이트되지 않은 외부의 링크를 사용할 때, 이 데이터베이스를 체크하여 에러를 표시하여 주는 방법[1]으로 단절 링크를 막는 방법을 제안하였고, F. Kappe는 문서의 삭제나 이동시 업데이트 정보를 기억하는 서버 업데이트 프로토콜[11]로 단절 링크를 막았다. 본 논문은 John Garaofalakis의 방법을 사용하여 단절 링크를 막아준다.

3. 그래프를 사용한 웹문서 구조화

3.1 웹문서 구조화

HTML로 작성되어진 웹문서들은 하이퍼텍스트의 형태를 가지고, 각 문서들간의 링크는 <A HREF="URL">...</A>태그를 사용하면 간단하게 구현이 가능하다. HTML에서는 항상 HTML 문서 내에 명시적으로 하이퍼텍스트 링크를 표현해야만 링크 제공이 가능하며, HTML 문서에 포함된 텍스트에 링크를 표현하는 방법 외에는 링크를 제공할 수 없다.

HTML 문서는 링크된 문서들의 링크를 다시 구성하고자 할 경우 수동적인 방법으로 <A HREF="URL">...</A> 태그를 사용하여 링크설정을 새로이 구성해 주어야만 한다.

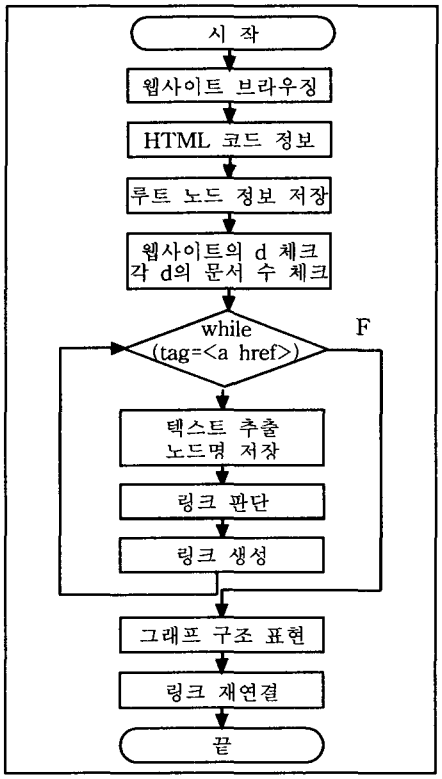
전통적으로 하이퍼텍스트들은 하이퍼텍스트 링크들을 만드는 것에 요구되는 수동의 노력에 의해 크기에 많은 제한이 있어왔고, 링크 생성을 관리하기 위하여 요구되어지는 시간의 양이 너무나 엄청나다[2].

웹사이트를 시각화 방법의 하나인 그래프로 표현함으로써, 웹 사용자들이 원하는 웹사이트로 이동할 때 요청한 자료로의 빠른 접근을 가능하게 해주기 위하여 페이지의 구조를 동적으로 바꾸거나, 문서들간의 링크의 구조를 바꾸어 줄 경우, 수동적인 방법인 <A HREF> 태그를 설정하여 주지 않고 그래프 구조에서 노드를 삽입, 삭제하여 얻어진 구조를 웹사이트에 적용하는 자동적인 방법으로 웹문서를 재조정하는 것이 가능하다. 이 때 삭제되는 노드에 링크되어 있는 하위노드들은 삭제되어지는 노드의 부모노드로의 링크를 재연결함으로써 노드 삭제시 단절 링크가 발생되지 않도록 한다.

그래프로 구조화된 웹 문서의 재구조화는 그래프 구조로 이루어진 웹사이트에서 링크들만을 재조정함으로써 가능하게 되며, 바뀌어지는 것은 그래프 구조내에서 페이지

들의 위치이다.

<그림 2>는 본 논문의 전체적인 구조를 보여주고 있다. 구조화하고자 하는 웹사이트를 읽어들인 다음 웹 문서들간의 링크관계를 그래프로 표현한다. 이 때 링크의 생성은 HTML 문서에서 문서와 문서를 연결해주는 태그인 <A> 태그를 이용한다. 과정을 거침으로써 웹사이트의 문서들은 새로운 연결구조를 가진 웹사이트가 된다.



<그림 2> 웹사이트 구조화 처리 절차

본 논문에서는 HTML 태그 중 문서들을 연결해주는 <A HREF>태그를 이용하여 상하계층을 만들어냄으로써 그래프 형태의 데이터 구조를 만드는 것이 가능하다.

각 웹문서들은 그래프로 표현시 홈페이지로부터 얼마의 깊이를 가지는지의 깊이(d) 체크가 반드시 필요하다. d는 그래프 구조 표현시 상위계층, 하위계층을 표현하고 링크 설정을 결정짓는 중요한 판단수단이 된다.

3.2 텍스트 추출

본 논문에서 기술하는 인터넷 정보추출 기법의 대상이 되는 것은 HTML로 작성된 텍스트 형태의 문서이다. <h1>이나 <p>와 같은 대부분의 HTML 태그들은 의미를 가지기보다는 화면에 출력되는 포맷을 지정하기 위한 태그로 정보 추출에 도움이 되지 못한다[10].

<그림 3>의 알고리즘은 웹사이트의 문서를 불러오는 처리절차를 표현하고 있다.

1. 그래프로 구조화할 웹사이트 URL을 읽어온다.
2. 읽어온 사이트의 주소를 저장한다.
3. 읽어온 사이트의 주소가 공백이면 URL을 다시 입력하라는 메시지를 띄워주고, 읽어온 URL이 접속이 불가능하면 에러메시지를 보여준다. 만약 읽어온 URL이 접속가능한 URL이면 이 문서의 HTML 코드 정보를 저장하고, 저장한 HTML코드 정보를 대문자로 바꾸어준다.

본 논문에서 추출의 대상은 웹사이트를 그래프로 표현하기 위하여 HTML 태그들 중 시각적으로도 구분이 가능하고, 문서와 문서들을 연결해 주는 태그인 <A HREF="URL">...</A>태그 사이의 텍스트이다.

```
shtml as string
hurl as string
txtcode as string

//웹문서를 읽어온다
  Browser.Navigator(URL)
//웹문서의 주소를 저장한다
  hurl=Browser.LocationURL
//URL 에 대한 처리
  if hurl = " " 이면
     URL 재입력
  else if hurl = "접속 불가능" 이면
     <그림 7>의 화면 출력
  else
//접속가능한 URL이면 HTML 코드를 저
  장하고 HTML 태그를 대문자화
    shtml = Inet.OpenURL(HURL, 0)
```

<그림 3> 웹사이트 문서를 불러오는 알고리즘

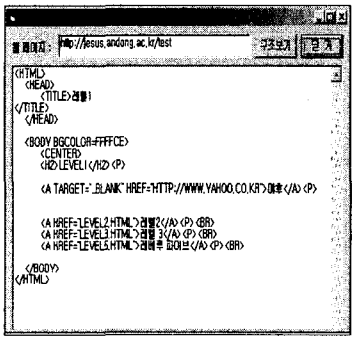
그래프로 표현하고자 하는 웹사이트의 HTML 태그 정보를 읽어들여 <A HREF= "URL">...</A>태그들에 대하여 파일명(xxx.html)과 태그 사이의 텍스트를 추출한다. 텍스트 추출을 위하여 추출 텍스트의 시작위치와 추출이 끝나는 지점을 정해준다. 추출의 시작위치는 파일명의 경우 "<A HREF=>"다음에 오는 텍스트이므로 "HREF="로부터 +6번째 위치를 추출의 시작 위치로 잡아준다. 추출위치의 끝은 <A HREF="URL">태그에서 ">"기호 바로 앞에는 늘 "가 오게 되므로 ">"기호로부터 -2번째 위치를

추출의 끝지점으로 잡아주어 이 사이의 모든 문자열의 값을 저장한다. 추출된 텍스트는 노드의 키값이 된다. <A HREF="URL">...</A>태그 사이의 텍스트도 <표 1>에서 보는 것과 같이 같은 방법을 사용하여 추출한다. 이 때 추출된 문자열은 노드명으로 그래프에 나타나게 된다.

루트 노드의 경우 최상위노드로 부모노드를 가지지 않으므로, 노드명을 추출할 수가 없게 된다. 그러므로 루트 노드인 경우에는 일반적인 <A HREF="URL">...</A>태그 사이의 텍스트 추출 이외에 문서의 제목을 나타내는 <TITLE>...</TITLE>태그 사이의 텍스트를 추출하여, 추출된 텍스트를 루트노드의 노드명으로 준다.

이 추출규칙은 가장 일반적인 방법으로 태그를 사용하였을 때 적용 가능한 추출규칙이며, 웹사이트들은 한 사이트의 포맷 관행이 다른 사이트들에 적용될 가능성이 거의 없으므로, 이 포맷을 따르지 않는 경우가 대부분이다. 이런 경우에는 <표 1>의 추출 규칙을 적용하여 텍스트를 추출하면 그래프 구조에서 노드들은 원하지 않는 노드명을 가지게 되므로, 이 때에는 wrapper 구성에서 추출의 시작점과 추출 끝점의 변경, 띄워쓰기 등에 대한 wrapper 변경을 한 후 텍스트를 추출하여야 한다.

<그림 4>은 웹사이트의 HTML 태그정보들을 읽어들인 화면으로, 대·소문자의 구분이 없이 기재된 HTML 코드 정보들을 모두 대문자로 표시하게 되며, 이 HTML 코드들은 텍스트 추출의 대상이 된다.



<그림 4> 페이지의 HTML 태그

3.3 그래프 형태로 구조화

본 논문에서는 HTML 태그 중 문서와 문서를 연결하여 주는 <A HREF>태그를 이용하여 상하계층을 만들어냄으로써 그래프 형태의 데이터 구조를 만드는 것이 가능하다.

웹사이트를 그래프 형태로 구조화하기 위하여 HTML 문서에서 문서를 연결하는 태그인 <A HREF="URL">....</A>태그 사이의 텍스트를 추출한 후, 추출한 텍스트를 그래프의 하위노드의 노드명으로 링크를 생성한다. 그러므로 <A HREF>태그의 수에 따라 하위노드의 수가 결정되어진다.

그래프의 노드는 각각의 노드에 대하여 "키값"과 "노드명"의 두 개의 정보를 가지게 된다. "키값"은 NodeID로 노드를 구별하는 고유한 값이며 노드의 깊이와 깊이에서의 위치 정보를 가지게 되며 webstep(i, j, 1)에 저장한다. "노드명"은 노드가 그래프로 표시될 때 가지게 되는 노드의 이름으로 webstep(i, j, 2)에 저장한다.

웹 문서를 그래프 형태로 표현할 때는 깊이 d를 고려하여야만 한다.

정의 3 (깊이) *d: 페이지의 깊이 (홈페이지에서 떨어져 있는 단계 수)[4]*

루트페이지로부터 얼마나 많은 단계를 가지는 가를 나타내는 페이지의 깊이 d에 대하여 루트노드는 d=1의 값을 가지게 되며, d값이 작을수록 루트페이지에 가까이 위치한 페이지이며 d값이 클수록 루트페이지에서 멀리 위치한 페이지이다. 그래프 표현시 홈페이지로부터의 깊이 d를 고려하여 d를 가지는 페이지들은 깊이 d-1을 가지는 페이지의 하위레벨에 위치시킨다.

본 논문에서 웹사이트를 그래프로 표현하는 것은 <A HREF>태그의 생성 없이 웹사이트를 그래프 구조로 구조화한 후 그래프 구조에서 노드를 삽입하거나 삭제하는 자동적인 방법으로 웹문서들을 삭제하거나 삽입하기 위해서이다. 그러므로 아래와 같은 링크들에 대하여는 그래프 구조로 표현시 링크 생성에 제한을 둔다.

<표 2> 가장 일반적인 텍스트 추출 규칙에 대한 wrapper

추출 대상 태그	추출대상 노드	추출 시작 위치	추출< 끝 위치	그래프에서의 표현
<TITLE>... </TITLE>	루트 노드	"<TITLE>"의 "<"기호 + 7번째	"</>" - 2번째	루트노드의 노드명 webstep(1, 1, 2)
<A HREF= "URL">...</A>	모든 노드	"HREF=" + 6번째	">" - 2번째	노드의 키 값 webstep(i, j, 1)
		">" + 1번째	"</A>"의 "<" - 1번째	노드명 webstep(i, j, 2)

링크생성에 제한을 가지는 링크

1. 한 문서에서 다른 문서로의 중복적인 문서 연결은 그래프 구조에서 같은 링크를 반복적으로 나타내게 된다. 그러므로 중복 링크에 대하여서는 제일 처음 한 번만 링크를 생성하여 준다.
2. <A HREF=#>로 링크되어지는 같은 문서 내에 서의 연결은 링크 생성을 하지 않는다.
3. 루트페이지로부터의 깊이의 수인 d가 낮은 문서가 d가 높은 문서를 부모노드를 가질 경우에는 그래프로 표현시 부모노드-하위노드로서 링크 생성을 표시할 뿐 하위노드에 연결되는 링크들은 생성하지 않는다.
4. 다른 컴퓨터 문서로의 연결은 하위노드로 링크만 하여줄 뿐 이 문서에 링크된 문서들에 대한 링크는 표현하여 주지 않는다.

각 문서들을 읽어들여 링크를 생성할 때 웹사이트의 모든 문서들에 대하여 그래프의 노드로 표현하기 위하여 그래프 내의 모든 노드들을 체계적으로 방문하고 모든 링크들을 조사하는 방법의 하나인 넓이-우선탐색(Breath-First Search)를 사용하여 모든 웹문서들의 링크를 생성한다.

<그림 5>은 HTML 정보를 읽어서 그래프로 표현하는 과정을 나타낸다. 그래프로 표현할 때는 먼

저 루트 노드를 설정하고, 웹사이트의 레벨과 각 레벨의 문서의 수를 체크한다. 레벨수와 각 레벨의 문서의 수에 의해 그래프에서의 노드를 설정하고, 각 노드마다 하위노드에 대한 정보를 읽어와 2레벨부터 추출된 문자열을 노드명으로 준다. 텍스트를 추출하여 웹문서를 그래프로 표현해 줄 때는 루트 노드를 나타낸 후 루트노드에 링크된 서브노드들을 나타내고, 이 노드에 링크된 서브노드들을 나타내게 된다.

3.4 구조화된 그래프에서 노드의 삽입과 삭제

본 논문은 웹문서들간의 링크를 새로이 설정하고자 하는 경우 구조화된 그래프 형태를 이용하여, 그래프에서의 노드를 삽입하거나 삭제한다. 이 때 삽입과 삭제가 가능한 문서는 구조화된 그래프에서 노드로 표시된 문서로 HTML 태그가 구성되어지지 않은 새로운 문서는 삽입이나 삭제가 불가능하다.

웹문서를 특정한 곳에 삽입하고자 하는 경우 구조화된 그래프에서 노드를 삽입한다. 노드의 삽입 위치는 부모노드와 링크된 노드들의 마지막 노드 다음 위치에 삽입한다.

```
webstep(i, j, 1): 노드의 키값
webstep(i, j, 2): 노드명
webstep(i, j, 3): 자식노드의 수
webstep(i, j, 4): 부모노드의 값

0 <i < 문서의 레벨
0 <j < 각 레벨에서의 문서의 수

1. 웹 사이트의 레벨과 각 레벨에서의 노드수 체크
  //루트 노드를 설정해 준다

  //레벨 2에서 마지막 레벨까지
   for i = 2 to 마지막 레벨까지
     j=1
   //키값이 공백이 아닌동안
      while webstep(i, j, 1) <> " "동안
   //웹사이트의 레벨값과 레벨에서 문서의 위치 설정
   //키값이 숫자일 때의 처리를 한다.
   //그래프에 노드를 추가한다
      j = j + 1
    next

2. 각 노드의 서브 노드수 체크와 텍스트 추출
  //웹사이트의 마지막 레벨까지
  //노드명이 존재하면
   if webstep(i)(j)(2)  <> " "이면
    //웹사이트를 불러와 html 코드를 대문자로 저장
      txtcode=UCase(Inet.Openurl)
    //만약 루트노드이면
      if webstep(i,j,1) = webstep(1,1,1) 이면
     //<TITLE>...</TITLE>태그 사이의 텍스트 추
       출하여 s(1) 저장
      else
     //자식 노드의 수 체크
     //<A HREF="파일명">에서 파일명을 추출해
       urladdr(count)에 저장
     //<A HREF="파일명">...</A>태그 사이의 텍tm
       트 추출하여  nameface(count)에 저장
    else
  //링크되는 노드가 실제 파일인가의 체크

3. 그래프 형태로 표현
    count = 1
    count1 = 1
  //루트 노드에 노드명 값을 준다
    web(1,1,2) = s(1)
  //웹사이트의 마지막 레벨까지
  //노드의 키값과 노드명 값을 준다
    for i = 2 to 마지막 레벨가지
      for j  = 1 to 각 레벨의 마지막 노드까지
        webstep(i, j, 1) = urladdr(count)
        webstep(i, j, 2) = nameface(count1)
        count = count + 1
        count1 = count1 + 1
      next
    next
```

<그림 5> 텍스트 추출과 노드 생성 방법 알고리즘

문서의 삽입절차는 <그림 6>의 알고리즘과 같다.

```
webstep(i, j, 1): 노드의 키값
webstep(i, j, 2): 노드명
webstep(i, j, 3): 자식노드의 수
webstep(i, j, 4): 부모노드의 값

0 <i < 문서의 레벨
0 <j < 각 레벨에서의 문서의 수

ppp, pp as string
level is . integer

  //삽입노드의 키값을 저장한다.
  ppp=node.key
//부모노드의 키값을 저장한다.
  pp=node.key
//레벨을 1증가시키고, 노드를 추가한다.
  level= level + 1
//pp의 하위노드에 노드를 추가한다.
//삽입된 노드에 대한 설정을 한다
  webstep(i, j, 1)=ppp+str(level)
  webstep(i, j, 2)=ppp
  webstep(i, j, 3)=webstep(i, j, 3)
  webstep(i, j, 4)=pp
```

<그림 6> 노드 삽입 절차 알고리즘

1. 삽입할 노드의 키값을 저장한다.
2. 부모노드의 키값을 저장한다. 이 때 노드명의 정보를 읽어오게 되면 그래프에서 동일한 노드가 여러 번 표현되었을 경우, 레벨과 위치에 대한 설정이 불가능하다. 그러므로 그래프에서 레벨과 레벨에서의 위치 정보를 가지는 키값의 정보를 읽어와서 저장한다.
3. 부모노드의 하위노드에 노드를 추가하고, 삽입된 노드의 정보를 갱신하여 준다. 이 때 노드의 키값 webstep(i, j, 1)은 부모노드의 아래노드에 위치하게 되므로 부모노드의 레벨값 +1의 키값을 가지며, 부모노드에 대한 정보 webstep(i, j, 4)는 현재 삽입된 위치에서의 부모노드에 대한 키값으로 설정한다.

문서의 삭제는 문서의 삽입과는 달리 그래프에서 노드를 삭제하였을 경우 노드가 가진 하위노드들로의 링크가 모두 삭제되어버려 단절 링크의 결과를 가져오게 된다. 단절 링크는 웹 환경안에서 한 리소스를 삭제할 경우, 하위노드들에 대하여 노드가 존재는 하고 있지만 노드로의 링크는 끊겨버려 노드로의 참조가 불가능하게 되는 결과를 가져오게 된다. 이 단절 링크는 현재 웹안에서 브라우징하는 사용자들이 직면하는 가장 성가신 문제의 하나이다[3].

하나의 노드가 삭제될 때 이 노드를 부모 노드로 가지는 자식노드들은 d=d-1의 값을 가지면서 삭제되는 노드의 부모노드를 부모노드로 하여 그래프에 표현된다. 이 방법을 사용함으로써 어떤 노드가 삭제되더라도 그 하위노드로의 링크 정보는 잃어버리지 않게 된다.

문서의 삭제절차는 <그림 7>의 알고리즘과 같다.

```
webstep(i, j, 1): 노드의 키값
webstep(i, j, 2): 노드명
webstep(i, j, 3): 자식노드의 수
webstep(i, j, 4): 부모노드의 값

0 <i < 문서의 레벨
0 <j < 각 레벨에서의 문서의
ppp, pp as string

//삭제노드의 키값을 저장한다.
  pp=node.key
//삭제 노드의 모든 하위노드들의 부모 노드 정
보를 바꾸고 노드 추가
  for j=1 to 하위노드의 수
    webstep(i)(j)(1)=webstep(i+1)(j)(1)
    webstep(i)(j)(2)=webstep(i+1)(j)(2)
  next j
//새로운 트리를 그래프에 표현
  wep(i, j, k) = webstep(i, j, k)
//노드 삭제
```

<그림 7> 노드 삭제 절차 알고리즘

1. 삭제할 노드의 키값을 읽어온다.
2. 삭제될 노드의 모든 하위노드들은 부모 노드에 대한 정보로 삭제될 노드값을 가지고 있다. 그러므로 노드가 삭제될 경우 부모 노드에 대한 정보를 잃게 되어 단절 링크가 되어지므로, 부모 노드의 정보를 삭제노드의 부모노드의 정보로 바꾸어 준 후, 노드 삭제시 하위노드를 표현하기 위해 정보를 저장해 둔다.
3. 삭제할 노드의 키값을 삭제한다.

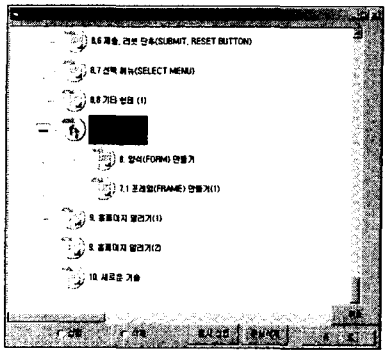
4. 실험 결과 및 고찰

제안방법의 성능을 평가하기 위하여 펜티엄 II, 메모리 64M, Visual Basic 6.0으로 텍스트를 추출하고, 텍스트 추출 결과를 하위노드의 노드명으로 주는 방법을 사용하여 그래프 구조로 표현하였다.

웹사이트들은 한 사이트에서 사용된 독특한 포맷 관행이 다른 사이트들의 구성에 전혀 영향을 미치지 않으므로 사용된 포맷들에서 규칙을 발견한다는 것이 쉽지는 않다. 그러나 태그의 시작점과 끝점의 위치만을 바꾸는 등의 간단한 wrapper의 변경으로 원하는 텍스트를 추출하는 것이 가능하다.

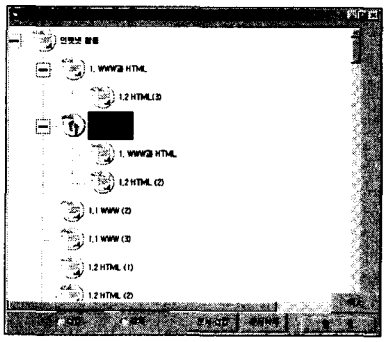
임의의 웹사이트를 그래프로 구현한 후, 구현된 그래프의 임의의 노드를 삽입, 삭제하여 보았다. 이 웹사이트의 태그들은 노드들을 나타내는 태그가 비교적 동일한 포맷을

가졌기에 wrapper의 변경 없이도 노드명을 나타내는 텍스트들을 추출하여 구조화가 가능하였다.



<그림 8> 웹사이트 그래프 표현

문서의 삽입은 삽입하고자 하는 노드를 클릭한 후 삽입하고자 하는 위치를 클릭하면 삽입이 이루어진다. <그림 8>는 임의의 웹문서를 그래프로 표현한 모습이고, <그림 9>는 노드가 삽입된 모습이다. <그림 10>은 "8. 양식(FORM)만들기" 노드와 "7.1 프레임(FRAME) 만들기(1)" 노드가 "8.9 기타형태 (2)" 노드에 링크되어 있다 "8.9 기타형태 (2)" 노드가 삭제됨으로 인하여 한 레벨의 마지막 노드들로 추가된 모습이다.

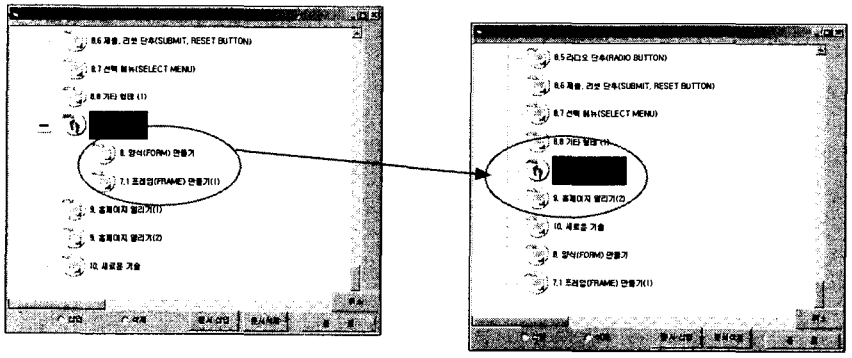


<그림 9>임의의 노드가 삽입된 모습

5. 결론

본 논문은 웹사이트에서 문서를 삽입하거나 삭제할 경우, <A HREF>태그를 생성하는 수동적인 방법을 사용하지 않고, 자동적인 방법으로 문서를 삽입·삭제한다. 자동적인 방법으로 문서를 삽입·삭제하기 위하여 문서의 HTML 태그 중 문서와 문서를 연결하여 주는 <A HREF>태그를 이용하여 상하계층을 만들어 웹사이트를 그래프 형태의 구조로 만든다. 그래프 구조의 상하계층은 한 문서의 <A HREF>...</A>태그사이의 텍스트들을 추출하여 추출된 텍스트를 이 문서의 하위노드의 노드명으로 링크를 생성하는 방법을 사용한다.

웹문서들간의 링크를 새로 설정하고자 하는 경우 <A HREF>태그를 생성하지 않고 구조화된 그래프 형태를 이용하여, 그래프에서 노드를 삽입하거나 삭제한 후 새로운 구조를 웹에 적용한다. 이 때 삽입과 삭제가 가능한 문서는 구조화된 그래프에서 노드로 표시된 문서 즉 HTML 태그가 구성되어진 문서로 제한된다. 문서를 삭제할 때에는 삭제될 노드와 링크되어 있는 노드들에 대하여 삭제되



<그림 10> 노드 삭제시 단절링크가 발생되지 않은 모습

는 노드의 부모노드와의 링크를 새로이 설정해 줌으로써 단절 링크를 없애준다.

본 논문은 인터넷 사이트들은 선택할 수 있는 포맷 스타일의 수가 매우 많으며 그 스타일 중에서 임의의 하나를 선택하기 때문에 텍스트 추출에 규칙을 부여하는 것이 어렵고, 특정한 포맷 스타일을 가지지 않은 인터넷 사이트들에서 추출된 결과는 원하지 않는 결과가 포함되었다. 이때는 추출 규칙에서 추출이 시작되는 위치와 추출을 끝마치는 위치를 바꾸어 주어야만 한다.

향후 연구로는 웹사이트에서 문서의 삽입, 삭제후 변화된 구조를 웹사이트에 적용하여 주는 방법과 웹사용자에게 요청한 자료로의 빠른 접근을 가능하게 해주기 위해 사이트들에서 페이지의 조직을 동적으로 바꾸기 위한 방법에 대한 연구가 지속되어야 할 것이다.

참 고 문 헌

[1] 정희경, "차세대 웹 문서 표준 XML," 정보처리학회지, vol. 6, no. 3, 1999.

[2] 최진숙, "Labeled Tree 모델을 매개로 한 웹 데이터의 구조화 방법," 포항공대 전자계산학과 석사학위논문, 1999.

[3] G. Golovchinsky, "What the Query Told the link: The Integration of Hypertext and Information Retrieval," Proc. 8th ACM Conf. Hypertext, ACM Press, New York, 1997, pp. 67-74.

[4] John Garaofalakis, "Web site optimization using page popularity," IEEE INTERNET COMPUTING, vol. 3, no. 4, pp. 22-29, 1999.

[5] D.Ingham, et al., "W3Object: Bringing Object-Oriented Technology to the Web," The Web Journal, vol. 1, pp. 89-105.

[6] 전경현, "가상경로 정보를 이용한 하이퍼텍스트 링크 생성 모델," 한국과학기술원 전산학과 석사학위논문, 1993.

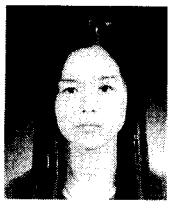
[7] 조민재, 황수철, 김기태, "웹 문서의 개념 지식과 그래프 구조를 이용한 정보 분류 및 추출 시스템," 한국정보과학회 추계학술발표대회 논문집, vol. 26, no. 1, pp. 280-282, 1999.

[8] 김성운, 김성진, 강현석, "웹 문서 분석/검색 시스템의 설계 및 구현," 한국정보처리학회 춘계학술발표논문집, vol. 6, no. 1, pp. 235-238, 1999.

[9] Ivan Herman, et al., "What is the Graph Visualization," http://www.cwi.nl/InfoVisu/General.html, 2000.

[10] 최중문, "인터넷 정보 추출 에이전트," 정보과학회지, 제18권, 제5호, pp. 48-53, 2000.

[11] F.Kappe, "A Scalable Architecture for Maintaining Refrential Integrity in Distributed Information Systems," Journal of Universal Computer Science, vol. 1, no. 2, pp. 84-104, 1995.



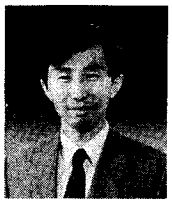
박 은 주 (Eun-Ju Park)

1993년 안동대학교 전산통계학과 졸업(이학사)

2001년 안동대학교 공과대학 컴퓨터공학과 졸업(석사)

2000년~현재 안동대학교 전자계산소 강사

관심분야 : 인터넷, 멀티미디어, 정보시각화



임 한 규 (Han-kyu Lim)

1981년 경북대학교 전자계산기공학 전공(학사)

1984년 연세대학교 산업대학원 전산 전공(석사)

1997년 성균관대학교 대학원 전자계산공학 전공(박사)

1981년~1982년 대한주택공사 전산실

1982년~1986년 한국전자통신연구소 위성통신연구실 연구원

1986년~1994년 한국IBM 소프트웨어연구소 선임연구원

1994년~1998년 한서대학교 전산정보학과 조교수

1998년~현재 안동대학교 저자정보산업학부 부교수

관심분야 : 자연언어처리, 영상처리, 멀티미디어