스타이너트리 기반의 효과적인 모바일 웹 오브젝트 네비게이션*

†이우기** · 송종수** · 이정훈**

An Effective Mobile Web Object Navigation Based on the Steiner Tree Approach

†Wookey Lee** · Justin Jongsu Song** · James J. H. Lee**

■ Abstract ■

One of the fundamental roles of web object navigation is to support what the user wants precisely and efficiently from the enormous web database to the web browser. As long as the web search results are a set of individual lists, it is all right to display each and every web result for the web browser to display a web object one by one. However, in case the search results are a collection of multiple interrelated web objects, then there is a need to represent for a new mechanism for linked web objects at a time. We define a unit of web objects derived from a Steiner tree where the web objects include a set of specific keywords calculated by the weight from which the solutions are extracted. Even if a web object does not include all the keywords, then the related hypertext linked web objects are derived and displayed onto the mobile web browser with meta data in one shot. In this paper, it is applied for the mobile browser that the web contents can dynamically be displayed with Steiner trees until each renewal of the navigation request may be issued. In this paper, a new synchronized mobile browsing method is developed so that the navigating time can drastically be reduced and the web navigating efficiency can be dramatically enhanced without sacrificing memory consumption.

Keywords : Steiner Tree, Top-k, Search Engine, Mobile Web

논문접수일 : 2010년 11월 19일 논문게재확정일 : 2011년 01월 24일

* This work was partially supported by Defense Acquisition Program Administration and Agency for Defense Development under the contract UD080042AD, Korea.

** 인하대학교 산업공학과

† 교신저자

1. 서 론

웹의 폭발적 성장은 질적 양적으로 괄목할 만하다. 양적으로 웹은 현재 조 단위를 넘어선지 오래되었다. 질적으로는 HTML, XML 등의 단순한 웹 페이지 뿐만 아니라, 그림이나 동영상, 전자화된 매뉴얼이나 데이터베이스부터 트위터나 이메일, 뉴스, 동영상, 웹 서비스 및 앱(App) 스토어 등을 포괄하므로, 이를 포괄적으로 표현하는 용어로서 본 연구에서는 웹 오브젝트(Web Object)라고 부르기로 한다. 물론 Small World 효과도 있지만, 이렇게 정보의 원천 자체가 초대형화 되어가고 웹 오브젝트의 종류가 다양해질수록, 사용자는 점점 더 원하는 정확한 내용을 확인하기가 어려워지는 정보화의 역설이 성립하므로, 이를 해결하기 위한 다종의 해법이 제안되고 있다. 현재 가장 중요한 접근법의 하나로서 웹 정보에 대한 구조화 즉, 그래프 접근법이 있는데, 이는 최근 웹 데이터베이스 및 정보처리 연구 집단에서 핵심적인 방법의 하나로 최근 2~3년 사이에 집중적으로 연구되고 있다. 이는 웹을 그래프 관점으로 나타내는 것으로 이를 웹 그래프(Web as a graph)라고도 부른다[7, 21]. 또한 최근 모바일 환경의 급격한 확산에 따라 위에 언급된 다양한 형태의 통합된 형태인 모바일 웹은 모바일 기기의 여러 가지 제약조건 즉, 디스플레이의 크기, 메모리 및 CPU용량, 네트웍의 비균질성, 프로토콜의 단순성 등에서 말미암아, 이러한 그래프 방식의 해법 더욱 효과적이다. 이러한 배경 하에 좀 더 근본적인 다음과 같은 질문을 생각해보기로 하자.

왜 그래프 접근법인가? 그래프는 개별 노드보다 더 일반적(more general) 표현이기 때문이다. 정보 네비게이션이란 근본적으로 (1)사용자의 질의와 (2)대상이 되는 웹 오브젝트(노드)들 중에서 (3)상호 일치하는 정도에 따라 순위를 매겨 찾아주는 것이다. 이때 가장 흔히 사용되는 방식으로서 질의와 웹 오브젝트가 모두 키워드로 되어있다고 가정할 경우, 사용자 질의가 복수개의 키워드로 구성되어 있을 때즉, (1)항 및 (2)항에 대한 설명이다. 네비게이션대상이 되는 웹 오브젝트들 중에서 그 여러 키워드를 모두 가지고 있을 이상적인 단 하나의 오브젝트를 네비게이션하기 보다는, 각 웹 오브젝트들이 키워드를 일부를 가지고 있으면서(partly informed) 하이퍼링크로 서로 연결된 다중 웹 오브젝트 즉, 그래프로 인식하는 것이 더 현실적이다. 물론 그러한 단 하나의 오브젝트가 존재할 수도 있으나, 이는 그래프 해의 한 특수해가 된다. 한편 키워드 방식이 아닌 네비게이션에 대한 연구도 많이 있는데, 이는 상기 (1)항 및 (2)항에 대한 설명이다. 이러한 시도들은 키워드 방식에 비해 연구에서나 실제에 있어서 활용도가 여전히 낮은 수준인 이유는 데이터 표현과 순위매기는 방법이 키워드 방식에 비해 아직 부정확하고, 이로 인해 해의 대상 역시 키워드 방식보다 더욱 넓어질 수밖에 없고, 따라서 단일 노드보다는 여러 노드가 포함된 그래프가 훨씬 적합한 방식이 된다 하겠다[15].

왜 스타이너 트리인가? 웹을 그래프로 표현할 경우 그 네비게이션 결과물은 스타이너 그래프 혹은 스타이너 트리로서 매우 적합하게 표현될 수 있기 때문이다. 웹 그래프에서 키워드를 가진 노드들과 하이퍼링크를 통해 연결고리 역할을 하나 키워드를 가지지 못한 노드 즉, 스타이너 노드의 결합물은 스타이너 그래프가 된다[11, 22-24]. 특히 모바일 환경에서는 네비게이션 결과의 시작노드 혹은 현재의 노드를 루트노드로 인식하고 거기에서 연결된 단일 경로를 따라가면서 네비게이션하는 것은 데스크 탑에 비해 제약조건이 많은 모바일 웹 네비게이션에 현실적인 대안이 되기 때문이다.

또한 이러한 트리의 형성이 웹 사용자의 선택에 따라 동적으로 이루어질 경우, 네비게이션의 시작을 어느 노드에서 하느냐에 따라 그 위치를 루트로 하는 트리를 만들 수 있도록 하는 것이다. 여기서 얻어질 수 있는 트리는 구체적으로는 루트노드를 가진 유향 스패닝 트리(rooted directed spanning tree) 혹은 루티드 아보레슨스(rooted arborescence)라고 부르는데, 여기서는 네비게이션 트리라고 줄여 부르도록 한다. 이러한 네비게이션 트리에서 개별노

드로 입력되는 아크는 하나로서 모두 n-1개의 아크가 n개의 노드에 연결되며 사이클이 없는 유향 그래프가 되어야 하고, 결과적으로 네비게이션 트리는 단 하나의 루트로부터 모든 노드에 유일한 경로가 존재하는 경우를 의미한다[3, 4, 8, 9, 11].

이러한 웹 노드(N) 및 웹 아크(A)로 구성된 웹 그래프에 대하여, 이를 수치화하는 방법은 상기 제(3)항에 대한 설명이다. 일반적인 가중치(weight) 혹은 거리기준(distance measure)으로서 정보검색 연구그룹에서는 글로벌 가중치와 로컬 가중치로 표현하고 있다[2, 12]. 이 두 가지 가중치는 어떤 의미에서 상호 독립적이다. 즉, 글로벌가중치는 개별적 네비게이션 키워드를 고려하지 않고 하이퍼링크 정보만을 이용하며, 한편 로컬가중치는 하이퍼링크를 배제하고 키워드를 사용하여 가중치를 구하기 때문이다. 현재 대표적인 글로벌가중치 방식에는 페이지랭크 방법[21] 및 그 변종들 그리고 Kleinberg의 HITS 방법[14] 등이 있다. 로컬가중치로서는 벡터공간모형(VSM : Vector Space Model)에 기반을 둔 코사인가중치, 확률적가중치, 퍼지가중치 및 흔히 쓰이는 tf-idf(term frequency and inverse document frequency)방법[4, 7, 12] 등이 있다. 노드의 내용(키워드)에 기반을 둔 노드 가중치 평가방법과 달리 아크정보를 활용한 아크기반 노드가중치 평가방법이 구글의 성공과 함께 주목받으며 매우 다양한 랭킹기법이 제안되고 있다. 이 방법에서는 인접행렬에 대해 출력링크 수를 기준으로 정의하고, 이러한 인접행렬에 대하여 행렬의 수렴조건을 강제하기위한 전처리를 위해 감쇠요소(damping factor) 등의 조건을 부여하고 난 다음, 마코프 체인에 따른 노드별 가중치를 구하게 된다. 이러한 페이지랭크의 성공에 기인하여 여러 가지 성능향상 및 보완을 시도하는 다양한 가중치 계산 기법들이 개발되고 있다[7, 12]. 한편 HITS는 웹 오브젝트들을 중요 오브젝트를 권위있는 노드(Authority node)로 분류하는 가정에서 출발하여, 다른 오브젝트에 연결기능이 강화된 노드를 허브노드(Hub node)로 정의하면, 이 경우 중요한 허브노드는 권위있는 노드에 많이 연결되고, 역으로 권위있는 노드는 중요한 허브노드에 연결되는 경향이 강하다는 가정 하에 개발되었다. 이 개념은 정작 처음 의도했던 정보검색 및 네비게이션 시스템 개발로는 이어지지 못하고, 대신에 사회망(Social Network)이나 논문참조방식(Citation reference), 특허정보 응용 시스템(Patent information) 등의 분야에서 본원적인 알고리즘이 되어 주목받고 있다[14, 20].

이때 페이지랭크 및 HITS와 같은 방법은 개별적인 키워드와는 무관하게 웹 오브젝트 사이의 하이퍼링크 숫자만을 사용하여 가중치를 구하므로 글로벌 가중치라고 부를 수 있다. 그러나 글로벌 가중치 만 가지고는 개별적 네비게이션 결과에 대한 지원되지 않으므로 키워드에 기반을 둔 전통적인 방법들 즉, 로컬가중치(local weight)가 보완적으로 필요하다. 그 대부분이 전통적 정보검색에서 사용하는 벡터공간 모델(VSM : vector space model)에 기반하고 있고, 그 중에서도 가장 많이 사용하는 방법의 하나가 *tf-idf*이다. 그러므로 로컬 가중치는 노드 기반의 노드 가중치 평가방식이라 명명될 수 있으며, 글로벌 가중치는 링크기반 노드 가중치라 부를 수 있다. 본 연구에서는 글로벌가중치와 로컬가중치의 곱을 사용하여 가중치로 사용하였지만, 가중치방식의 선정과 그 결합방식이 본 연구에 제약요인이 되지는 않는다. 그러므로 웹을 기본적으로 유향그래프로 인식하고, 이러한 웹 그래프에서 특정 키워드를 포함하는 노드와 아크들의 부분집합인 스타이너 트리들을 구조적 네비게이션 트리(structured navigation tree)라고 부르고, 이를 모바일 환경에 적용하는 것이 본 연구의 목표이다.

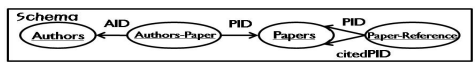
본 논문은 다음과 같이 구성되어 있다. 제 2장은 웹 그래프 모형에 대해 정규적 표현 및 예제 그리고 Top-k에 대한 표현을 설명했으며, 제 3장에서 스타이너트리를 구현하였고 계층적 웹 네비게이션 시스템의 구현절차를 각각 데스크 탑 및 서버 시스템의 경우와 모바일 응용에 구현한 다양한 실시 예를 보였고 이에 대한 구현결과를 분석하였으며, 제 4장 통해 논문의 결론을 기술한다.

2. 웹 그래프의 정규적 표현 (Formal Representation)

웹을 유방향 그래프(a directed graph) $G(N, A)$로 표현할 때, N은 웹 오브젝트를 표현하는 웹 노드 집합으로 A는 웹 노드를 연결하는 하이퍼링크들을 표현하는 웹 아크 집합을 의미한다. 그래프의 행렬 표현인 인접 행렬(adjacency matrix) $M = [m_{ij}]$는 웹 오브젝트 i에서 j로의 아크가 존재하면 m_{ij} = 1로, 존재하지 않으면 m_{ij} = 0으로 표현된다. 인접 행렬은 다양한 그래프 표현에서 웹 오브젝트들의 정량적인 순위를 제공하는데 사용되고 있다.

2.1 키워드 기반 스타이너 그래프 네비게이션 기법

차세대 검색엔진은 하이퍼링크로 연결된 정보들을 통합하는 '연결인지'가 중요한 문제가 되고 있다. 하지만 기존 역색인을 사용하면 XML이나 관계형 데이터베이스와 같이 복잡한 구조로 연결된 정보에서 효과적인 결과를 확인하기가 곤란하다. 그래서 구조적 데이터에 대한 효율적인 네비게이션 방식을 고안하여 효과적인 스타이너 그래프 및 스타이너 트리 네비게이션을 시도할 필요가 있다 [10, 17, 18].



Authors

AID	Name
a1	J. Shanmugasundaram
a2	L. Guo
a3	V. Hristidis
a4	Y. Papakonstantinou
a5	A. Balmin

Paper-Reference

PID	citedPID
p1	p2
p2	p3
p3	p4
p4	p5
p5	P6

Author-Paper

AID	PID
a1	p1
a1	p2
a2	p1
a3	p4
a3	p5
a4	p5
a4	p6
a4	p7
a5	p6

Papers

PID	Title
p1	Topology Search over Biological Database
p2	XRANK: Ranked Keyword Search over XML Documents
p3	Bidirectional Expansion for Keyword Search on Graphs
p4	Finding *top-k* Answers in Keyword Proximity Search
p5	Efficient IR-Style Keyword Search over Relational Databases
p6	Keyword Proximity Search on XML Graphs
P7	DISCOVER: Keyword Search in Relational Databases

〈그림 1〉 DBLP 데이터베이스 예

그리고 데이터베이스와 정보 네비게이션 측면을 모두 고려한 평가기준을 가지고 Top-k의 네비게이션 결과에 대한 정확도 등을 평가할 수 있다. 우선 이를 위해 예제로 그래프 방식의 산출물을 제시하고 이를 통해 본 문제를 설명하기로 한다. 본 예제는 논문 데이터베이스로 DBLP를 용용한 내용의 일부[10]로서, 4개 테이블(Authors, Paper Reference, Author-Paper, Papers) 및 그 메타정보가 제시되어 있다. 여기서 저자와 논문을 노드로 인식할 경우 인접행렬 및 그 2단계 행렬은 <그림 2>와 같이 표현될 수 있다.

(1)M

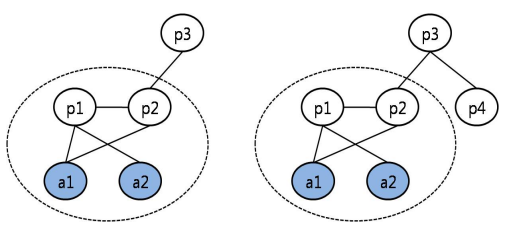
	a1	p1	a2	p2	p3	p4	a3	p5	a4	p6	p7	a5
a1	1	1	0	1	0	0	0	0	0	0	0	0
p1	1	1	1	1	0	0	0	0	0	0	0	0
a2	0	1	1	0	0	0	0	0	0	0	0	0
p2	1	1	0	1	1	0	0	0	0	0	0	0
p3	0	0	0	1	1	1	0	0	0	0	0	0
p4	0	0	0	0	1	1	1	1	0	0	0	0
a3	0	0	0	0	0	1	1	1	0	0	0	0
p5	0	0	0	0	0	1	1	1	1	1	0	0
a4	0	0	0	0	0	0	0	1	1	1	1	0
p6	0	0	0	0	0	0	0	1	1	1	0	1
p7	0	0	0	0	0	0	0	0	1	0	1	0
a5	0	0	0	0	0	0	0	0	0	1	0	1

(2)M²

	a1	p1	a2	P2	p3	p4	a3	p5	a4	p6	p7	a5
a1	1	1	1	1	1	0	0	0	0	0	0	0
p1	1	1	1	1	1	0	0	0	0	0	0	0
a2	1	1	1	1	0	0	0	0	0	0	0	0
p2	1	1	1	1	1	1	0	0	0	0	0	0
p3	1	1	0	1	1	1	1	1	0	0	0	0
p4	0	0	0	1	1	1	1	1	1	1	0	0
a3	0	0	0	0	1	1	1	1	1	1	0	0
p5	0	0	0	0	1	1	1	1	1	1	1	1
a4	0	0	0	0	0	1	1	1	1	1	1	1
p6	0	0	0	0	0	1	1	1	1	1	1	1
p7	0	0	0	0	0	0	0	1	1	1	1	0
a5	0	0	0	0	0	0	0	1	1	1	0	1

〈그림 2〉 DBLP 데이터베이스의 인접 행렬

이 예에서 <그림 1>은 <그림 2>에 대한 연결을 인접 행렬로 나타낸 것이다. <그림 2>(b)는 M^2 = M×M로서 M^2_{ij} = 1일 때, i와 j가 2번의 하이퍼링크로 연결되어 있다는 것을 나타낸다. r번 곱한 행렬의 i열이 1인 노드들을 조합하면 r만큼의 반지름을 가진 그래프 g_i^r가 된다.



〈그림 3〉 2-반지름 그래프

예컨대, <그림 3>의 g^2_{p1}는 M^2 행렬의 노드 p1에서 1의 값을 가지는 노드들의 조합으로 이루어진 그래프이다. 하지만 인접 행렬에서 얻은 그래프는 사용자가 입력한 키워드와 관련 없는 노드들도 포함되어 있기 때문에, 불필요한 노드를 제거한 스타이너 그래프로 바꾸어야 한다. 그래프 g^r에서 키워드를 포함한 콘텐츠 노드를 c_1, c_2, …, c_q라고 할 때, 스타이너 그래프를 얻는 알고리즘은 다음과 같다.

Algorithm : SteinerGraph
Input : g_i, c_i, r | Output : g^r, P
초기조건 : $g^r = P$ = null, r = integer
단계 1 : g_i에 속한 노드 중에 c_i로 향하는 경로 중에 있는 노드들의 집합 $P(c_i)$를 구한다. 콘텐츠 노드 $\{c_1, c_2, \cdots, c_{q-1}, c_q\}$와 콘텐츠 노드와 연결된 아크들을 제외한다.
단계 2 : 다음의 식을 사용하여 g^r에 속한 스타이너 노드 집합 P를 얻는다.

$$P = \cup_{i=1}^{q} \cup_{j=i+1}^{q} (P(c_i) \cap P(c_j)) \cup c_1, c_2, \cdots, c_q$$

단계 3 : 스타이너 노드 집합 P에 속한 노드들과 P와 연관된 아크들로 구성된 스타이너 그래프를 얻는다.

본 연구에서는 유향 그래프 스타이너 문제를 이용하여 웹을 거대한 그래프로 인식하고 그 중에서 키워드가 있는 부분적인 트리 혹인 그래프를 찾는 방법을 이용하여 최대 중요도 트리생성 문제로 치환할 수 있으며, 이는 기본적으로 최소비용트리 문제로 다음과 같은 트리생성 알고리즘으로 생성할 수 있다. 아크 가중치는 [19]에서와 같이 임의로 부여하거나, 트리의 높이[13], 노드 간 키워드 합과 거리계수의 곱[10] 등의 방법으로 부여할 수 있다. 본 연구에서는 [10]의 계수를 사용했다. 본 방법은 [16, 17]에 기반을 둔 엔진에 의거하여 모바일 시스템으로 새로이 구현하였으며, 주어진 그래프에서 모든 아크의 중요도 및 루트 노드 r을 입력으로 하여 다음 알고리즘은 중요도 합을 최대로 하는 네비게이션 트리구조를 생성하는 것이다. 참고로 기존의 연구에서 웹 그래프 구조에 대한 다양한 전처리 과정 및 데이터베이스에 로딩하는 과정을 채용한 것임을 밝히는 바이다.

2.2 Top-K 질의 문제

Top-k 질의 문제는 각각 n개의 데이터를 가진 m개의 리스트가 있다고 가정한다. 그리고 각 리스트에 속한 데이터는 지역 값을 갖는다. 특정 랭킹 함수에 따라 각 리스트들의 지역 값을 계산하여 데이터에 대한 종합 점수를 얻을 수 있다. 이때, 종합 점수를 기준으로 가장 높은 값을 갖는 k개를 얻는 문제를 Top-k 질의 문제라고 부른다.

Top-k 문제의 예로서 관계형 데이터베이스 테이블에서 점수화 함수를 테이블의 속성에 적용하여 k개의 튜플을 찾는 문제를 들 수 있다. k개의 튜플들은 속성들의 값들로 얻은 종합 점수가 가장 높은 k개를 뜻한다. 또 다른 예로는 주어진 키워드에 대한 k개의 문서들을 얻는 문제가 있다. 키워드에 따라 순위가 결정된 리스트들에서 점수화 함수로서 문서들의 종합 순위를 결정하여 k개의 문서를 얻을 수 있다. 이러한 Top-k 문제를 표현하기에 앞서 사용되는 기호를 <표 1>과 같이 정리하였다.

n개의 데이터들의 집합을 D라고 하고, L_1, L_2, …, L_m이라는 m개의 리스트들이 있다. 각 리스트들은 $(d, s_i(d))$로 구성되는데, d는 $d \in D$ 데이터이고 $s_i(d)$는 리스트 L_i에 속한 d의 지역 값이다. 각 리스트 L_i가 내림차순으로 정렬되어 있는 경우 "정렬된 리스트"로 불린다. 종합 점수는 각 리스트에 포함된 해당 데이터 d의 지역 값들$(s_1(d), s_2(d), \cdots, s_i(d))$을 단조함수인 점수화 함수 $f(s_1(d), s_2(d), \cdots, s_i(d))$에 대입하여 얻는다. 점수화 함수는 단조함수이기 때문에 만약 $x_i \le x'_i$이라면 $f(x_1, \cdots, x_m) \le f(x'_1, \cdots, x'_m)$이다. 예컨대, 자주 사용되는 점수화 함수로는 최소, 최대, 평균 등의 단조함수가 있다. 이때 점수화 함수의 점수를 기준으로 높은 값을 가지는 k개의 데이터를 Top-k 데이터라고 부른다.

정렬된 리스트의 데이터에 접근하는 방법은 크게 두 가지가 있다. 첫째는 리스트의 정렬된 순서대로 접근하는 정렬접근(Sequential access)이다. 정렬접근으로 데이터를 읽을 경우 다음 단계로 읽을 데이터는 현재 데이터의 다음 순위 데이터가 된다. 두 번째 방법은 임의 접근(Random access)이

다. 임의 접근은 리스트 내에서 주어진 특정 데이터의 지역 값을 찾는 것을 나타낸다. 정렬접근의 비용을 c_s, 임의접근의 비용을 c_r이라 하자. 그리고 정렬접근의 총 횟수를 a_s, 임의접근의 총 횟수를 a_r이라고 할 경우, 결과적으로 Top-k 스타이너 그래프 즉, 네비게이션트리를 찾기 위한 점수화 함수는 $a_s \times c_s + a_r \times c_r$로 구할 수 있다.

〈표 1〉 Top-k 문제 기호 설명

기호	설명
D	데이터 집합{d_1, d_2, …, d_n}
L	데이터의 지역 값을 갖는 리스트의 집합{L_1, …, L_m}
n	데이터의 수
m	데이터 속성의 차원 수
$s_i(d)$	데이터 d의 i번째 속성 지역 값
$f()$	Top-k 집합 선정의 기준이 되는 함수. $f(s_1(d), …, s_m(d))$ 데이터의 지역 값들을 통해 계산되는 점수화 함수이다.
δ	TA 알고리즘에서 종료 조건으로 사용되는 임계값
bp_i	BPA 알고리즘에서 L_i의 최적위치
λ	BPA 알고리즘에서 종료 조건으로 사용되는 임계값

3. 시스템 구현(Implementation)

여기서는 다양한 시스템으로 구현하였는데, 우선 데스크 탑 및 서버방식의 시스템이며 다음으로 이를 구체화한 모바일 방식이 그것이다. 모바일에서도 다양한 데이터 조합에 대해 제대로 된 결과를 얻는지 확인하기위해 조합한 데이터에 대해 실험하였고, 또한 실제 데이터에 대해서도 시스템을 구현하였다.

본 연구에서는 웹사이트를 구조화하기 위해, 아크 중요도의 합을 최대화하는 계층적인 구조를 만들어 내는 것이 목표가 된다. 이러한 문제는 스타이너 트리 문제라고 할 수 있다. 스타이너 트리 문제란 주어진 노드들 중 특정 노드들(terminal node)을 연결하는 최소 신장 트리를 형성하는 문제이다. 즉, 전체 n개의 노드가 주어졌을 때, t(≤ n)개의 특정 노드를 연결하는 최소비용의 트리를 찾는다. 다만, 비용감소에 도움이 된다면 이들 특정 노드들 중간에(n-t)개의 비특정 노드(non-terminal node)를 추가하여 트리를 형성하는 문제라고 할 수 있다. 따라서 주어진 노드들 중 특정 노드의 개수가 두 개이면 최단거리 문제(shortest path problem)가 되며, 특정 노드의 수가 주어진 전체 노드의 개수와 같으면 최소신장트리(minimum spanning tree)가 된다. 따라서 일반적인 스타이너 트리 문제는 특정 노드의 수가 두 개 이상이면서 주어진 모든 노드의 수보다 작은 문제들 이라고 할 수 있다. 노드들의 집합이 주어져 있을 때, 여러 형태의 스타이너 트리들이 가능하다.

다음으로 웹 그래프에서 Top-k 네비게이션트리를 찾아내는 것으로 그 절차는 다음과 같다. 대상이 되는 웹 사이트를 선정하는 방법은 우선 하나의 네비게이션엔진에서 키워드를 입력하여 얻어진 결과를 가지고 네비게이션트리로 구현하는 것이다. 본 연구에서 구현한 시스템의 수행절차는 다음과 같다.

3.1 구현 절차

시작집합(seed set)을 얻는 단계로서 네비게이션을 시작할 수 있는 대상 URL은 웹에서 랜덤서치를 하는 방법과 WHOIS 등의 메타정보로부터 얻는 방법, 어떤 네비게이션엔진에 키워드를 입력하고 얻어지는 URL들을 대상으로 하는 방법 등이 있다. 여기서는 마지막 방법을 채택하였다.

얻어진 시작노드 집합에서 특정 URL을 하나 선정해 루트노드라고 이름하며, 이 루트노드를 시스템에 입력하여 도달하는 웹 오브젝트 및 그 웹 오브젝트와 연결된 도달 가능한 웹 오브젝트들을 선정하기 시작한다. 물론 이때 모든 URL들은 각각 구조적 네비게이션에서 루트노드가 될 수 있으며 이중 사용자가 임의로 혹은 Top 1위의 URL을 루트노드로 선정할 수 있다.

앞에서 선정된 루트노드로부터 도달 가능한 URL 집합(N^*)이 네비게이션 대상 도메인(G^*)이 된다. 루트노드를 결정하면 사용자의 브라우저에는 해당 URL이 지칭하는 웹 오브젝트의 내용이 제시된다. 또한 웹사이트의 노드와 아크가 분석된다. 이때, 네비게이션가능(indexable) 웹 오브젝트들의 범위를 결정해주는데 네비게이션에 불요한 잡음(Noise)을 제거하는 절차를 동시에 수행하게 된다. 여기에는 Hidden Web 및 redirection 기능 등을 배제하고 네비게이션가능 웹 오브젝트들을 결정해주게 되는데, 이 과정은 확장자에 기반을 두어 제거하는 절차로 이루어진다. 예컨대, png, img, jpg, c, java, txt, doc, xls, ppt, hwp, 등이 해당된다.

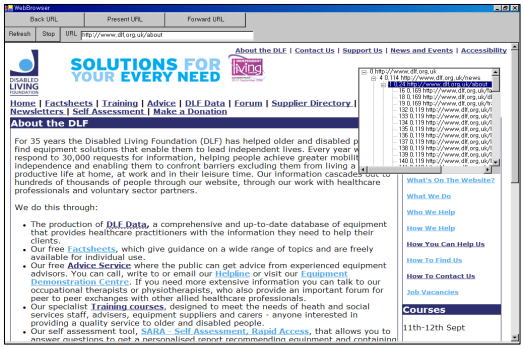
노드 및 아크 집합을 확정하고 난 다음 아크의 가중치를 구하게 된다. 여기서는 글로벌 가중치로서 PageRank, 로컬 가중치로서 tf-idf를 구하고 이들 양자의 곱으로 아크 가중치를 정의하며, 이에 따라 가중치를 계산하였다.

상기 방식에 따라 얻어지는 자료를 대상으로 본 연구에서는 앞서 설명한 방식에 기반을 두어 트리를 유도하였다. 이때 다양한 네비게이션 요구기준을 적용하여 필요한 트리를 구한다. 또한 사용자가 어떤 URL을 선정하는가에 따라 루트를 바꿀 수 있고, 그 루트로부터 상기 언급된 단계를 적용하여 새로운 네비게이션 결과를 얻을 수 있으며, 이러한 강점을 따서 동적(dynamic)이라고 할 수 있다.

3.2 서버 시스템

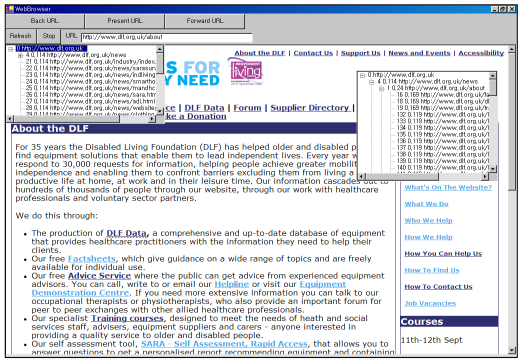
우선 데스크 탑 및 서버 방식에 대하여, 앞서 언급한 절차를 따라 특정 도메인에 대한 웹 오브젝트 정보들을 수집하여 그래프 형태의 구조를 트리 형태로 변환하고 웹 브라우저에 구현한 결과를 다음 예에서 확인할 수 있다.

우선 <그림 4>는 http://www.dlf.org.uk/about 이 루트노드가 된 것을 확인할 수 있으며, 이들의 구조 정보가 웹 브라우저에 나타나 있다. 특정 도메인 내의 웹 오브젝트의 계층 구조는 오른쪽 상단에 있는 Tree View 컨트롤에 나타난다. 이 컨트롤은 위치의 이동과 크기의 축소와 확장이 가능하다. 컨트롤 내에서 웹 오브젝트는 웹 오브젝트 번호와 함께 Tree Node로 표현된다. Tree Node는 한 개의 부모 노드와 여러 개의 자식 노드를 갖는다. 현재 오브젝트의 노드 번호는 1번이며, 웹 오브젝트 번호가 4번인 노드 http://www.dlf.org.uk/news의 자식 노드이다.



〈그림 4〉 화면 구성

상위구조정보는 Tree View 컨트롤로 화면에 표시된다. 이 Tree View는 현재 오브젝트와 현재 오브젝트의 하위 오브젝트를 제외한 웹 오브젝트 노드로 구성되어 있다. <그림 5>에서 표시되어 있는 1번 노드는 오른쪽의 Tree View에는 표시가 되어 있지만 왼쪽의 Tree View에는 1번 노드와 1번 노드의 자식 노드들은 표시되어 있지 않다.

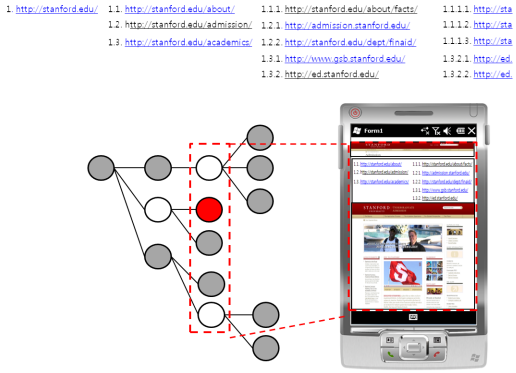


〈그림 5〉 상위구조 정보

하위구조정보의 경우 Forward 버튼을 누르면 Back 버튼을 눌렀을 때와 같이 Tree View 컨트롤이 화면에 표시된다. 하지만 Back 버튼을 눌렀을 때와 다르게 Forward를 눌렀을 때 나타나는 Tree View 컨트롤은 현재 오브젝트 노드와 현재 오브젝트 노드의 자식 웹 오브젝트 노드들을 표시한다.

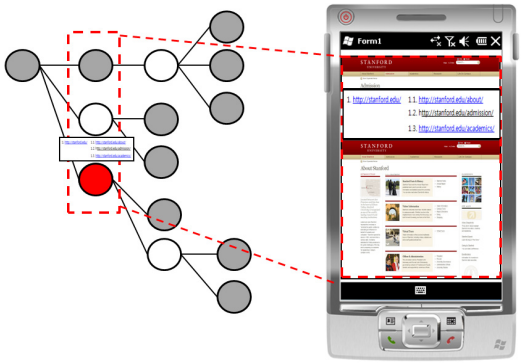
3.3 모바일 시스템

일반 웹 화면을 모바일 환경에 적용하려면 모바일 특성에 대한 고려가 이루어 져야 한다. 모바일 웹은 최근 풀 브라우징의 도입으로 실제 일반 웹 환경과 유사하게 가까워졌지만, 화면 크기의 제약이라는 단점은 해결되지 않았다. 형제 노드들을 가중치 값에 근거하여 순차적으로 한 화면에 나열하는 그래픽적인 표현을 하게 되면 모바일 디바이스의 터치기능을 이용하여 화면의 종이동(horizontal movement)을 통한 브라우징의 네비게이션에서 시간을 절약 할 수 있다는 장점이 있다.



〈그림 6〉 그래프 기반 계층적 웹 네비게이션 시스템의 모바일 네비게이션

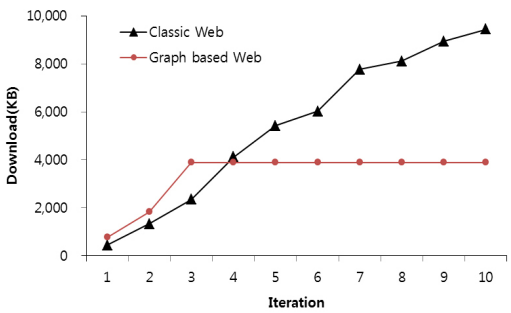
<그림 6>는 모바일 환경에 적용한 모습이다. www.stanford.edu 사이트에 대한 스타이너 트리 네비게이션을 실행한 후 이를 모바일의 한 화면에 나타내면 네비게이션 키워드를 가지고 있는 오브젝트를 트리의 깊이별로 보여줌으로써 원하는 네비게이션 결과를 효율적으로 네비게이션할 수 있다는 장점이 있다. <그림 7>은 원하는 오브젝트 바로 위에 네비게이션 트리 네비게이션이 위치한 모습이다. 트리 네비게이션을 이용하여 네비게이션 하려고 하는 오브젝트의 이동도 더욱 효과적으로 할 수 있다.



〈그림 7〉 유동적으로 위치하는 네비게이션 트리 네비게이션

Cache를 저장한다고 가정 하면, 그래프 기반 웹(Graph based Web) 화면은 일반 웹(Classic Web) 보다 클릭의 수(iteration)의 증가에 따라서 더욱 효율적인 모습을 보여준다. 모바일을 이용하여 일반 웹 네비게이션을 할 때 원하는 오브젝트를 찾아가기 위해서 많은 클릭을 함으로써 시간과 비용을 낭비하는데, 그래프 기반 웹 화면은 트리의 차수까지는 일반 웹 화면 보다 많은 비용을 보이지만, 그 이상의 오브젝트 네비게이션에서는 오브젝트를 저장하고 있어서, 시간뿐 아니라 비용도 절약 된다. <그림 8>은 Iteration에 따른 누적되는 오브젝트 다운로드를 비교한 그림이다.

양자 간의 성능의 차이에서 결정적인 부분은 웹 오브젝트의 양의 차이에서 오는 장점도 조금 있지만, 오히려 전체 오브젝트를 재요청하는 Classic 방식에 비해 Graph based 방식은 변경된 부분만 요청하는 브라우징 기술적 차이에 따른 장점이 메모리의 부담이 없이도 가능하다는 것이다. 결과적으로 네비게이션된 웹 오브젝트들을 어떻게 사용자에게 표현할 것인지 문제도 중요한데, 본 연구에서



〈그림 8〉 그래프 기반 웹과 일반 웹의 비교

처럼 스타이너트리라는 새로운 네비게이션 접근법으로 여러 오브젝트로 구성된 네비게이션 결과도 매우 효과적으로 지원하므로, 따라서 이는 새로운 네비게이션 패러다임이 될 수 있다.

4. 결 론

본 연구에서는 새로운 웹 네비게이션 접근법을 구현한 것으로서 동적인 스타이너 트리 및 그 시스템에 대한 구현한 결과를 제시하였고, 특히 모바일 환경에 적용한 것을 보여주었다. 이러한 웹 네비게이션 시스템은 기존의 검색엔진들과는 전혀 다른 새로운 접근법이다. 이는 기존의 단일 웹 객체들의 나열을 포함하는 좀 더 일반화된 네비게이션 결과로서 스타이너 트리방식을 채택하였다. 웹 객체의 중요도 평가 즉, 가중치를 구하는 방법은 기존의 표준적인 방식을 따랐다. 다음으로, 네비게이션 절차에서 중요한 점은 우선 시작단계에서 네비게이션 대상을 결정한 다음, 루트노드 선정하되, 사용자가 임의로 결정할 수 있으므로 동적인 특성을 가진다. 다음으로 루트노드로부터 도달 가능한 URL 집합 즉, 네비게이션 대상 도메인에 따라 해당 URL 웹 사이트의 노드와 아크가 분석하고, 최종적으로 절차에 따라 네비게이션 요구기준을 바꿀 수 있는 네비게이션 트리를 메타정보로서 구현한 것이다. 실험을 통해 보여준 것은 웹 오브젝트들의 연결 노드를 통합하여 모바일 브라우저에 제시해 준다. 결과적으로 새로운 모바일 브라우저를 구현하여 각 웹 오브젝트를 동기적으로 보여주는 새로운 네비게이션 방식을 구현하여, 시간을 획기적으로 단축하는 효율성 증가를 입증하였다. 또한 그래프를 네비게이션하여도 메모리를 희생시키지 않고, 동시에 효과적으로 지원할 수 있는 현실성 있는 기법이라 것을 보였다.

본 연구는 다음 몇 가지 방향으로 확장될 수 있다. 여기서는 유방향 웹 그래프를 다루었으나, 일반적인 그래프 구조 즉, 역방향까지 다룰 수 있도록 확장할 수 있다. 물론 이는 장차 모바일 디바이스의 기능 및 프로토콜이 이를 포함하도록 개선되어야 할 것이다. 또한 본 연구에서 그래프 인덱스를 결합할 경우, 상업적인 검색엔진 및 모바일 네비게이션 엔진이 가능하며 이를 위한 연구가 진행중이다.

참 고 문 헌

[1] Akbarinia, R., E. Pacitti, and P. Valduriez, "Best Position Algorithms for Top-k Queries," VLDB, (2007), pp.495-506.

[2] Amer-Yahia, S., A. Doan, J.M. Kleinberg, N. Koudas, and M. Franklin, "Crowds, clouds, and algorithms : exploring the human side of 'big data' applications," SIGMOD, (2010), pp.1259-1260.

[3] Byrka, J., F. Grandoni, T. Rothvoß, and L. Sanità, "An improved LP-based approximation for steiner tree," STOC, (2010), pp.583-592.

[4] Consoli, S., K. Darby-Dowman, N. Mladenovic, and J.A. Moreno-Pérez, "Variable neighbourhood search for the minimum labelling Steiner tree problem," *Annals OR*, Vol. 172, No.1(2009), pp.71-96.

[5] Fagin, R., A. Lotem, and M. Naor, "Optimal aggregation algorithms for middleware," PODS, (2001), pp.102-113.

[6] Fagin, R., "Combining Fuzzy Information from Multiple Systems," PODS, (1996), pp.216-226.

[7] Glover, E.J., D.M. Pennock, S. Lawrence, and R. Krovetz, "Inferring Hierarchical Descriptions," Proc. CIKM, (2002), pp.507-514.

[8] Gouveia, L. and J. Telhada, "The multi-weighted Steiner tree problem : A reformulation by intersection," *Computers and OR*, Vol.35, No.11(2008), pp.3599-3611.

[9] Gouveia, L., P. Moura, and A. Sousa, "Prize collecting Steiner trees with node degree dependent costs," *Computers and OR*, Vol.38, No.1(2011), pp.234-245.

[10] Guoliang, L., O.C. Beng, F. Jianhua, W. Jianyong, and Z. Lizhu, "EASE : an effective 3-in-1 keyword search method for unstructured, semi-structured and structured data," In Proc. SIGMOD, (2008), pp.903 - 914.

[11] Gupta, A. and A. Kumar, "A constant-factor approximation for stochastic Steiner forest. STOC," (2009), pp.659-668.

[12] Hammami, M.Y. Chahir, and L. Chen, "WebGuard : A Web Filtering Engine Combining Textual, Structural, and Visual Content-Based Analysis," *IEEE TKDE*, Vol.18, No.2 (2006), pp.272-284.

[13] Kasneci, G., M. Ramanath, M. Sozio, F. M. Suchanek, and G. Weikum, "STAR : Steiner-Tree Approximation in Relationship Graphs," In Proc. ICDE, (2009), pp.868-879.

[14] Kleinberg, J.M., A. Slivkins, and T. Wexler, "Triangulation and embedding using small sets of beacons," *J. ACM*, Vol.56, No.6(2009).

[15] Kleinberg, J.M. and S. Lawrence, "The Structure of the Web," *Science*, Vol.294(2001), p.1849.

[16] Lee, W., K.S. Carson, J. Leung, J.H. Lee, "Mobile Web Navigation in Digital Ecosystems Using Rooted Directed Trees," IEEE TIE, 2011(in-print).

[17] Lee, W.J., J.-H. Lee, Y. Kim, C.K-S. Leung, "AnchorWoman : top-k structured mobile web search engine," *CIKM*, (2009), pp.2089-2090.

[18] Lee, W., J. Song, J. Baek, and J. Lee., "Steiner Tree based Effective Mobile Web Search," Proc. KORMS Conference, 2010.

[19] Li, W., K. Candan, Q. Vu, and D. Agrawal, "Retrieving and Organizing Web Pages by Information Unit," In Proc. WWW, (2001), pp.230-244.

[20] Maserrat, H. and J. Pei, "Neighbor query friendly compression of social networks," KDD, (2010), pp.533-542.

[21] Pandurangan, G., P. Raghavan, and E. Upfal, "Using PageRank to Characterize web Structure," Proc. COCOON, (2002), pp.330-339.

[22] Pinto, L.L. and G. Laporte, "An efficient algorithm for the Steiner Tree Problem with revenue, bottleneck and hop objective functions," *European Journal of Operational Research*, Vol.207, No.1(2010), pp.45-49.

[23] Rothlauf, F., "On Optimal Solutions for the Optimal Communication Spanning Tree Problem," *Operations Research*, Vol.57, No.2 (2009), pp.413-425.

[24] Whittaker, G., R. Confesor Jr., S.M. Griffith, R. Färe, S. Grosskopf, J.J. Steiner, G.W. Mueller-Warrant, and G.M. Banowetz "A hybrid genetic algorithm for multiobjective problems with activity analysis-based local search," *European Journal of Operational Research*, Vol.193, No.1(2009), pp.195-203.