웹 어플리케이션을 위한 URL 정규화
(URL Normalization for Web Applications)

홍 석 후 † 김 성 진 †† 이 상 호 †††
(Seok Hoo Hong) (Sung Jin Kim) (Sang Ho Lee)

요 약 웹에서는 문법적으로 서로 다른 문자열의 URL들이 동일 자원을 나타낼 수 있다. URL 정규화는 동일 자원을 나타내는 서로 다른 URL들을 통일된 형태로 변환하는 과정이다. 현재 URL 정규화에 대한 표준화가 진행 중에 있다. 표준 URL 정규화는 "잘못된 긍정"을 허용하지 않으며 "잘못된 부정"을 최소화하는 것을 목적으로 한다. 본 논문에서는 표준 URL 정규화에서 고려되지 않은 네 가지 정규화 요소를 고려한다. 본 논문은 "잘못된 긍정"을 부분적으로 허용하여 표준 URL 정규화에서 빈번히 발생하는 "잘못된 부정"을 제거하는 것을 목적으로 한다. 또한, 제안된 정규화 고려 요소의 효과를 평가하기 위하여 두 개의 척도가 정의되었다. 마지막으로, 본 논문은 실제 웹 문서 중에 발견된 약 1억 7천만 개의 URL에 대하여 실험을 수행하고 결과를 기술한다.

키워드 : URL, URL 정규화, 웹 어플리케이션, 잘못된 긍정

Abstract In the WWW, syntactically different URLs could represent the same resource. The URL normalization is a process that transform a URL, syntactically different and represent the same resource, into canonical form. There are on-going efforts to define standard URL normalization. The standard URL normalization designed to minimize false negative while strictly avoiding false positive. This paper considers the four URL normalization issues beyond ones specified in the standard URL normalization. The idea behind our work is that in the URL normalization we want to minimize false negatives further while allowing false positives in a limited level. Two metrics are defined to analyze the effect of each step in the URL normalization. Over 170 million URLs that were collected in the real web pages, we did an experiment, and interesting statistical results are reported in this paper.

Key words : Uniform Resource Locator, URL normalization, web application, false positive

1. 서 론

URL(Uniform Resource Locator)은 웹에 존재하는 자원(또는 문서)의 위치를 가리키는 문자열이다. URL은 스키마(scheme), 권한(authority), 경로(path), 질의(query), 단편(fragment)의 5개의 구성요소로 구분된다[1]. URL 표기 규칙은 한 문서가 서로 다른 문자열의 URL로 표현되는 것을 허용한다. 본 논문에서는 같은 문서를 나타내는 URL들을 동일하다고 한다. 웹에서는 서로 다른 문자열로 표현되었으나 실제 동일한 URL들이 다수 존재한다.

URL 정규화(normalization)는 다양하게 표현된 동일 URL들을 하나의 통일된 형태(canonical form)로 변환하는 과정이다. 웹 어플리케이션은 동일 문자열로 정규화되는 URL들을 동일하다고 간주한다. 예를 들어, 웹 로봇[2-4]은 웹 문서 수집 중 발견되는 URL들을 정규화하고 이미 보유한(문서의 다운로드 요청에 사용된) 정규화된 URL과의 중복 여부를 판별한다. 서로 다른 문자열의 두 URL에 대해 동일 여부를 정확히 판단하는 것은 사실상 불가능하다. "잘못된 부정"(false negative)은 동일한 두 URL을 동일하지 않다고 판단하는 것이다. 웹 어플리케이션이 사용하는 URL 정규화 방법에 "잘못된 부정"이 많이 발생할수록, 동일 URL들이 동일 문자열로 변환되지 않는다. "잘못된 부정"은 웹 어플리케이션들이 하나의 문서를 중복하여 요청하고 저장하도록 하여, 네트워크 사용량 증가, 디스크 I/O의 증가, 불필요한 디스크 공간의 사용 등의 문제를 유발한다.

현재 URL 정규화의 표준안을 정의하기 위한 연구가

· This work was supported by grant number KRF-2004-005-D00172 from the Korea Science and Engineering Foundation
† 학생회원 : 숭실대학교 컴퓨터학부
vonsaki@korea.com
†† 학생회원 : 서울대학교 전기컴퓨터공학부
sjkim@oopsla.snu.ac.kr
††† 종신회원 : 숭실대학교 컴퓨터학부 교수
shlee@computing.ssu.ac.kr
논문접수 : 2005년 1월 26일
심사완료 : 2005년 9월 2일

진행 중이다[1]. “잘못된 긍정”(false positive)은 비동일 URL을 동일 URL로 판단하는 것이다. 표준 URL 정규화는 “잘못된 긍정”의 발생을 허용하지 않으면서, “잘못된 부정”을 최소화는 것을 목적으로 한다. 즉, 어플리케이션에서 표준 URL 정규화를 수행하는 경우 비동일 URL을 동일한 문자열로 변환하는 경우가 발생하지 않는다.

표준 URL 정규화를 수행한 후에도 웹 어플리케이션은 다수의 중복된 URL을 보유한다. 실제 웹 환경에서는 표준 URL 정규화에서 제한하는 “잘못된 긍정”을 부분적으로 허용하여 “잘못된 부정”을 현저히 감소시킬 수 있다. 예를 들어, “http://www.acm.org/pubs”와 “http://www.acm.org/pubs/”는 실제 동일한 URL이나, 표준 URL 정규화를 수행하는 경우 동일한 문자열의 URL로 변환하지 않는다. 원칙적으로 두 URL은 동일한 URL이 아니다. 그러나, 실제 대부분의 웹 서버는 전자의 URL이 후자의 URL로 리다이렉션되도록 설정되어 있다. URL 정규화는 웹 어플리케이션의 개발에 중요한 부분을 차지함에도 불구하고, 현재까지 URL 정규화 방안의 수행 시 고려되어야하는 사항들을 제기한 연구가 거의 진행되어 있지 않다. 실세계의 웹 어플리케이션이 사용하는 URL 정규화는 문헌상으로 거의 발표되지 않고 있으며, 많은 개발자들은 자신의 경험과 직관에 의존하여 URL 정규화를 수행하고 있다. 본 논문은 이러한 경우를 고려하여 표준 URL 정규화의 성능을 향상할 수 있는 요소에 대하여 조사한다.

본 논문은 표준 URL 정규화에 기술되지 않은 경로부의 대소문자 구분, 경로부의 마지막 슬래시 문자, 기본 문서, 호스트의 ‘www’접두사의 유무의 4가지 추가적인 URL 정규화 고려 사항에 대하여 기술한다. 또한, 중복도(redundancy rate)와 커버리지 손실률(coverage loss rate)의 두 가지 척도를 제안한다. 중복도는 “잘못된 부정”으로 발생하는 중복 문서의 양을 나타낸다. 커버리지 손실률은 “잘못된 긍정”으로 손실되는 문서의 양을 나타낸다. 본 논문에서는 한국 사이트에서 수집한 약 1억 7천만 개의 실제 URL들을 대상으로 네 가지 정규화 방법의 중복도와 커버리지 손실률을 산출한다. 산출된 결과를 바탕으로 “잘못된 긍정”을 부분적으로(약1%) 허용하여 “잘못된 부정”을 현저히(약 50%) 감소시키는 URL 정규화 방안을 제시한다.

본 논문은 다음과 같이 구성되었다. 2장에서는 URL의 구성과 표준 URL 정규화에 대해 알아보고, 3장에서는 본 논문에서 사용되는 두 가지의 척도를 기술한다. 4장에서는 각 URL 정규화의 수행에 따른 실험 결과를 기술하며, 마지막으로 5장에서 결론을 맺고 향후 연구계획을 기술한다.

2. URL 정규화

2.1 URL의 구성요소

URL은 스키마, 권한, 경로, 질의, 단편으로 구성된다. 스키마는 웹 서버와 클라이언트간의 통신에 사용되는 프로토콜을 기술한다. 권한은 사용자 정보, 호스트(host), 포트번호 3개의 요소로 구성된다. 사용자 정보는 사용자의 인증을 필요로 하는 경우 계정 이름과 비밀번호의 인증 정보를 기술한다. 계정 이름과 비밀번호는 엣(‘@’) 기호를 구분자로 사용한다. 호스트는 웹 서버의 위치를 나타낸다. 웹 서버의 위치는 도메인 이름(domain name) 또는 IP 주소(Internet Protocol address)로 나타낸다. 포트번호는 도메인 이름 또는 IP 주소와 콜론(‘:’)으로 구분된다. 예를 들어, 80번 포트를 사용하는 ACM 사이트(www.acm.org)의 웹 서버는 호스트에 “www.acm.org:80”로 기술할 수 있다. 또한, ACM 사이트의 IP주소를 사용하여 “199.222.69.250:80”로 표기하여도 동일한 호스트이다. 경로는 웹 문서가 위치한 디렉토리와 파일명이 기술된다. 각 디렉토리와 파일명은 슬래시(‘/’) 문자로 구분된다. 디렉토리는 현재 디렉토리와 상위 디렉토리를 의미하는 문자열로서 ‘.’과 ‘..’을 사용한다. 예를 들어, 동일 사이트에서의 두 경로 “/membership/benefits.html”과 “/memebership/../../membership/benefits.html”은 동일 문서를 나타낸다. 질의는 웹 어플리케이션에 입력되는 파라미터(parameter)들의 이름과 값을 기술한다. 질의는 물음표(‘?’)로 시작한다. 파라미터 이름과 값은 등호(‘=’)로 구분한다. 하나 이상의 파라미터가 존재할 때 각 파라미터는 앰퍼센트(‘&’)로 구분한다. 예를 들어, 질의 “?name=smith&age=25”는 두개의 파라미터 ‘name’과 ‘age’의 값이 ‘smith’와 ‘25’임을 의미한다. 단편은 문서내의 특정 단락을 나타낸다. 단편의 시작은 샵(‘#’) 문자로 시작한다. 예를 들어, URL “http://example.com/list.htm#chap1”은 “list.htm”에서 “chap1”이라고 명명된 단락을 가리킨다. 그림 1은 URL의 모든 구성요소를 포함한 예이다.

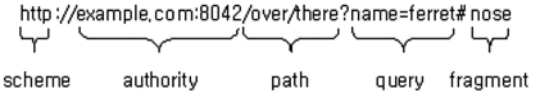

그림 1 URL의 구성

2.2 표준 URL 정규화

URL 정규화는 다양하게 표현된 동일한 URL들을 통일된 형태로 문자열 변환을 한다. 표준 URL 정규화[1]는 다음의 방법으로 URL 정규화를 수행한다.

첫째, 기본포트 번호를 포함한 URL과 포트 번호가

생략된 URL은 동일한 URL이다. 포트번호가 생략된 경우, 웹 서버와 클라이언트는 프로토콜의 기본포트(HTTP는 80번 포트)를 이용하여 통신을 한다. 즉, "http://example.com:80/"과 "http://example.com/"은 동일한 문서를 나타낸다. 표준 URL 정규화는 기본포트 번호를 포함한 URL을 포트번호를 생략한 URL로 변환한다.

둘째, 스키마와 권한의 영문자는 대소문자 구분을 적용하지 않는다. 즉, 대소문자 구분만이 다른 두 URL "http://EXAMPLE.com"과 "http://example.com"은 동일한 URL이다. 표준 URL 정규화는 스키마와 호스트의 영문자들을 모두 소문자로 변환한다.

셋째, 경로가 주어지지 않은 URL은 웹 서버의 루트 디렉토리를 요청하는 URL과 동일하다. 예를 들어, "http://example.com"와 "http://example.com/"은 동일 문서를 나타낸다. 표준 URL 정규화는 경로가 주어지지 않은 경우 '/'문자를 추가한다.

넷째, [1]에서 예약된 문자(reserved char)와 안전하지 않은 문자(unsafe char)를 제외한 모든 문자는 인코딩하여 나타낼 수 있다. 인코딩된 문자는 퍼센트 기호('%')로 시작하고 ASCII 값을 나타내는 16진수 두 자리를 결합한 세 자리 문자로 변환된다. 예를 들어, 두 URL "http://example.com/~smith"과 "http://example.com/%7Esmith"은 동일 URL이다. 표준 URL 정규화는 예약된 문자와 안전하지 않은 문자는 인코딩을 수행하고, 그 외의 문자는 디코딩을 수행한다.

다섯째, 단편을 포함한 URL과 단편 이후를 삭제한 URL은 동일한 URL이다. 예를 들어, "http://example.com/list.htm#chap1"과 "http://example.com/list.htm"은 모두 동일한 문서 "list.htm"을 나타낸다. 표준 URL 정규화는 URL에서 단편에 해당하는 문자열을 삭제한다.

3. 중복도와 커버리지 손실률

본 장에서는 각 URL 정규화 고려 요소의 효과를 평가하는 두 개의 척도(중복도와 커버리지 손실률)를 기술한다. 임의의 정규화 고려 요소 F를 가정하자. 동일 후보 URL 집합은 F가 적용될 때 동일 URL로 간주되는 URL들의 집합이다. 전체 집합 U는 웹 어플리케이션이 보유한 모든 URL의 집합이다. 전체 집합 U는 정규화 방법의 선택에 따라 서로 다른 동일 후보 URL 집합 U_i로 나누어진다. i번째의 동일 후보 URL 집합은 U_i로 표기한다. 전체 URL 집합 U는 $U_1 \cup U_2 \cup ... \cup U_n$로 구성되고, $i \neq j$ 인 경우 $U_i \cap U_j = \varnothing$이다. 1개의 원소만을 가진 동일 후보 URL 집합은 동일 여부를 비교할 다른 동일 후보 URL이 존재하지 않으므로, 중복도와 커버리지 손실률의 계산에서 1개의 원소만을 가진 동일 후보 URL 집합은 제외하였다.

두 개 이상의 원소를 보유한 N개의 동일 후보 URL 집합이 있다고 가정한다. D_i는 동일 후보 URL 집합 U_i에서 성공적으로 다운로드된 문서 집합이다. $n(Di)$는 D_i의 문서 개수이며, $un(Di)$는 고유한 문서의 개수이다. (본 논문에서는 고유한 내용을 가진 문서의 수를 말한다)중복도는 수식 (1)과 같이 정의된다.

$$\text{중복도} = \frac{\sum_{i=1}^{N}(n(Di) - un(Di))}{\sum_{i=1}^{N} n(Di)} \quad (1)$$

중복도는 동일 후보 URL들로 인하여 수집된 문서 집합에서 발생하는 중복된 문서의 비율을 나타낸다. 만약 동일 후보 URL 집합의 모든 URL이 고유한 URL이라면, 중복도는 0이 된다. 이것은 "잘못된 부정"이 발생하지 않음을 의미한다.

동일 후보 URL 집합 U_i에 속한 모든 URL을 동일 URL로 간주하는 특정한 정규화를 수행하는 경우를 가정하자. 이 경우 동일 후보 URL 집합 U_i의 모든 URL은 하나의 동일한 문자열의 URL로 변환된다. 동일 후보 URL 집합 U_i의 모든 URL이 동일한 URL이 아닌 경우, URL 정규화의 수행은 서로 다른 URL을 동일 URL로 변환을 하는 "잘못된 긍정"을 발생한다. "잘못된 긍정"은 수집 가능한 문서에 대한 손실을 발생하게 한다. 예를 들어, 동일 후보 URL 집합 U_i의 모든 문서가 서로 다르다면, 정규화를 수행하지 않는 경우 $un(Di)$만큼의 고유한 문서를 수집하지만, 정규화 수행으로 인해 최대 $(un(Di)-1)$만큼의 문서 수집 손실이 발생한다. 커버리지 손실률은 수식 (2)와 같이 정의한다. Nnormal은 변환된 이후의 URL들로부터 성공적으로 다운로드된 문서 개수이다.

$$\text{커버리지 손실율} = 1 - \frac{Nnormal}{\sum_{i=1}^{N} un(Di)} \quad (2)$$

중복도는 "잘못된 부정"에 의해 발생하는 중복 수집의 비율을 의미하고, 커버리지 손실률은 "잘못된 긍정"으로 인해 손실이 되는 다운로드 가능한 문서의 수를 나타낸다. URL 정규화의 확장에 대한 본 논문의 방법은 제한된(사용자에 의해) 커버리지의 손실을 허용하여, 중복도를 최소한으로 낮추는데 목적을 둔다.

Example :

예를 들어, 전체 집합 U가 다음과 같이 10개의 URL(u1에서 u10)을 보유하고 있다고 가정하자. 표기의 편리를 위해 스키마("http://")는 생략한다.

U = { "www.microsoft.com/" (u1),
"www.microsoft.com/default.asp" (u2),
"www.microsoft.com/index.htm" (u3),
"www.microsoft.com/index.html" (u4),

"nasdaq.com/asp/ownership.asp" (u5),
"nasdaq.com/ASP/ownership.asp" (u6),
"ietf.org/tao.html" (u7),
"ietf.org/TAO.html" (u8),
"ietf.org/Tao.html" (u9),
"www.acm.org/" (u10) }

전체 집합 U는 분류기준에 따라 서로 다른 동일 후보 URL 집합을 생성하는 것이 가능하다. 웹 서버의 기본 페이지에 따라 동일 후보 URL 집합을 분류하면, 다음과 같은 7개의 동일 후보 URL 집합이 있다.

U_1 = {u1, u2, u3, u4}, U_2 = {u5}, U_3 = {u6},
U_4 = {u7}, U_5 = {u8}, U_6 = {u9}, U_7 = {u10}

U_2, ..., U_7은 하나의 URL만을 보유하고 있으므로, 고려 대상에서 제외한다. URL u1, u2, u3, u4를 요청하여 다운로드된 각각의 문서는 ⓐ, ⓐ, ⓑ, ⓑ 이다. U_1의 동일 후보 URL을 u1로 정규화하면, 중복도와 커버리지 손실률은 다음과 같이 계산된다.

중복도 = ((4-2)/4) = 0.5,
커버리지 손실률 = (1-(1/2)) = 0.5 (문서 ⓑ의 수집 손실이 발생)

전체 집합 U를 다른 기준(경로의 문자열에 대한 대소문자 구분에 의한)으로 분류하면, 7개의 동일 후보 URL 집합이 있다..

U_1 = {u1}, U_2 = {u2}, U_3 = {u3}, U_4 = {u4},
U_5 = {u5, u6}, U_6 = {u7, u8, u9}, U_7 = {u10}

URL u5, u6, u7, u8, u9를 요청하여 각각 {ⓒ, ⓒ, ⓓ, ●, ●}의 문서를 수집하였다. ●는 다운로드에 실패한 URL이다. U_5와 U_6의 동일 후보 URL을 각각 u5와 u7로 정규화하는 경우, 중복도와 커버리지 손실률은 다음과 같이 계산된다.

중복도 = ((2-1)+(1-1))/(2+1) = 0.33
커버리지손실률 = 1-(2/2) =0

전체 집합 U를 나누는 기준에 따라 분류되는 동일 후보 URL 집합이 다르게 생성됨을 알 수 있다. ■

4. 확장 정규화 방안

본 장에서는 표준 URL 정규화 방안의 수행 이후의 추가적인 4가지의 정규화 고려 요소를 기술하였다. 실제 웹에서 수집된 URL을 사용하여, 4가지 정규화 고려 요소에 대하여 표준 URL 정규화와 "잘못된 긍정"을 허용하는 정규화의 성능을 중복도와 커버리지 손실률로 비교하였다.

실험을 위해, 웹 로봇을[5] 이용하여 2003년 12월에 1주간 35만개의 한국 웹 사이트에서 넓이 우선 방식(bread-first fashion)으로 웹 문서를 수집하였다. 웹 로봇은 사이트의 루트(root) 문서로부터 9홉(hop) 이내의 문서를 요청하였다. 웹 로봇은 동적으로 생성된 문서의 다운로드를 방지하기 위하여, 쿼리를 포함한 URL은 요청을 하지 않았다. 그 결과, 약 5천만개의 웹 문서를 성공적으로 다운로드 하였다. 우리는 수집된 웹 문서에서 1억 7천만개의 URL을 추출하고, 표준 문서의 정규화 방안을 따라 정규화를 수행하였다.

4.1 경로의 대소문자 구분

윈도우즈(Windows) 운영체제는 디렉토리와 파일에 대한 대소문자 구분을 하지 않는다. 윈도우 운영체제 기반의 웹 서버에서, 한 웹 문서는 경로의 대소문자 표현을 달리하여 다양한 URL로 표현될 수 있다. 예를 들어, 두 URL "http://www.nasdaq.com/asp/ownership.asp"과 "http://www.nasdaq.com/ASP/ownership.asp"은 동일한 문서를 나타내는 URL이다. 한편, 유닉스(Unix)나 리눅스(Linux) 운영체제는 디렉토리와 파일 이름에 대한 대소문자를 구분하여, 대소문자 구분이 다른 두 URL문자열은 서로 다른 문서를 나타낸다. 예를 들어, "http://www.acm.org/pubs/journals.html"와 "http://www.acm.org/PUBS/ journals.html"은 동일한 문서를 나타내지 않는다.

URL의 스키마와 호스트는 대소문자 구분을 하지 않는다. URL의 경로는 원칙적으로는 대소문자의 구분을 한다. 본 실험은 경로의 대소문자 구분을 하지 않는 경우의 효과를 관찰하였다. 하나의 동일 후보 URL 집합은 경로의 대소문자 구분만이 다른 URL들로 이루어지며, 다음과 같은 3가지 방법을 비교하였다.

(1) URL 경로를 변환하지 않은 방법
(2) URL 경로를 모두 소문자로 변환하는 방법
(3) URL 경로를 모두 대문자로 변환하는 방법

첫 번째 방법은 표준 URL 정규화의 방법이다. 두 번째와 세 번째 방법은 UNIX 운영체제에서 "잘못된 긍정"을 발생할 수 있다. 대소문자의 구분만이 다른 동일 후보 URL 383,444개를 포함하는 집합 185,474개를 실험 대상으로 하였다. 그림 2는 3가지의 방법에 따른 요청 횟수, 다운로드된 문서의 수와 수집된 문서 중 고유한 문서의 수를 나타낸다. 첫 번째 방법은 중복도가 0.52로 나타났다. 이것은 다운로드된 문서의 약 50%가 중복된 문서라는 것을 의미한다.

두 번째 방법의 경우 경로의 모든 문자열을 소문자로 변환하므로, 각각의 동일 후보 URL 집합은 하나의 URL로 정규화가 수행된다. 그러므로, 하나의 동일 후보 URL 집합은 하나의 URL만을 요청한다. 동일 후보 URL 집합의 정규화 수행으로 URL 요청은 48%(=185,474/383,444) 감소하였다. 두 번째 방법의 커버리지 손실률은 0.01이며, 첫 번째 방법에서 1%의 문서를 손실한 것으로 나타났다. 세 번째 방법의 커버리지 손실률은 0.11이다.

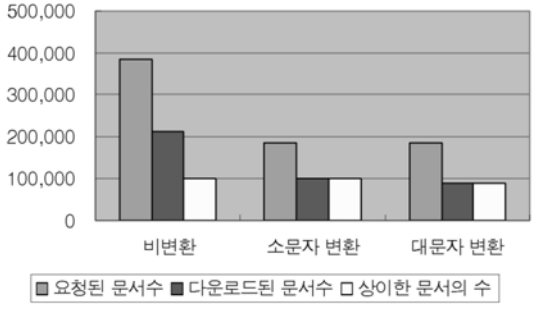

그림 2 대소문자 구분에 의한 동일 후보 URL의 문서 수집 결과

경로의 대소문자 구분만이 다른 동일 후보 URL 집합 내의 동일 후보 URL들은 실제 웹 상에서 대부분 동일한 문서를 나타냄을 알 수 있다. 또한, 소문자로 변환하는 두 번째 방법이 대문자로 변환하는 세 번째 방법에 비해 더 적은 커버리지 손실을 보인다. URL의 경로를 변환하지 않는 첫 번째 방법에 비해, 경로의 모든 문자를 소문자로 변환하는 방법인 두 번째 방법이 URL 요청을 48%감소할 뿐 아니라, 1%의 고유한 문서에 대한 손실로 52%의 중복된 웹 문서를 감소하였다. 두 번째 방법이 얻는 중복 제거의 잇점을 고려한다면, 1%의 커버리지 손실률은 작은 것으로 나타난다.

4.2 마지막 슬래시 문자의 유무

표준 URL 정규화는 경로가 주어지지 않은 URL을 경로가 '/'인 URL로 변환한다. 즉, "http://example.com"은 "http://example.com/"와 동일하다. 그러나, 경로가 주어진 URL의 마지막 슬래시 문자에 대해서는 문자열 변화를 수행하지 않는다. 즉, "http://example.com/list"와 "http://example.com/list/"는 서로 다른 URL로 간주한다.

본 절에서는 경로가 주어진 URL의 마지막 슬래시 문자에 대한 정규화 수행의 효과를 관찰한다. 마지막 문자가 슬래시로 끝나는 URL은 디렉토리를 나타내는 URL이다. 웹 서버는 디렉토리를 나타내는 URL을 요청 받은 경우, 요청된 디렉토리내의 기본 문서로 응답하거나 디렉토리가 포함하는 모든 파일을 보여주는 문서를 생성하여 응답한다. 사용자들은 디렉토리를 요청하는 경우 종종 마지막 슬래시 문자를 생략하곤 한다. 이 경우 웹 서버는 마지막 슬래시 문자를 포함한 URL로 리다이렉션을 수행한다. 예를 들어, "http://acm.org/pubs"는 "http://acm.org/pubs/"로 리다이렉션이 발생한다. 마지막 슬래시 문자에 대한 웹 서버의 자동화된 처리는 사용자로 하여금 마지막 슬래시 문자의 유무에 대하여 크게 고려하지 않게 한다.

경로가 주어지고, 마지막 슬래시 문자의 유무만이 다른 두 URL은 하나의 동일 후보 URL집합을 구성한다. 마지막 슬래시 문자의 동일 후보 URL 집합에 대하여 다음의 세 가지 방법을 고려하였다.

(1) 마지막 슬래시 문자를 변환하지 않는 방법

(2) URL 정규화의 형태로 마지막 슬래시 문자를 포함한 형태를 선택하는 방법
즉, 마지막 슬래시 문자를 포함하지 않은 동일 후보 URL에 슬래시 문자를 추가

(3) URL 정규화의 형태로 마지막 슬래시 문자를 제거한 형태를 선택하는 방법
즉, 마지막 슬래시 문자를 포함한 동일 후보 URL에 슬래시 문자를 제거

경로의 마지막 슬래시 문자 유무만이 다른 동일 후보 URL 152,924개를 포함하는 집합 76,462개를 실험 대상으로 하였다. 그림 3은 3가지의 방법에 따른 요청 횟수와 다운로드된 문서의 수, 수집된 문서 중 고유한 문서의 수를 나타낸다. 첫 번째 방법은 중복도가 0.49이며, 다운로드된 문서의 약 50%가 중복된 문서라는 것을 의미한다.두 번째와 세 번째 방법의 커버리지 손실률은 각각 약 0.01과 0.03으로 나타났다.

실험 결과는 마지막 슬래시 문자의 동일 후보 URL이 서로 다른 문서를 나타내는 경우가 실제로 매우 드물다는 것을 나타낸다. 마지막 슬래시 문자를 추가하는 두 번째 방법은 1%의 다운로드 가능한 문서를 손실하여, 약 50%의 중복된 웹 문서를 제거한다.

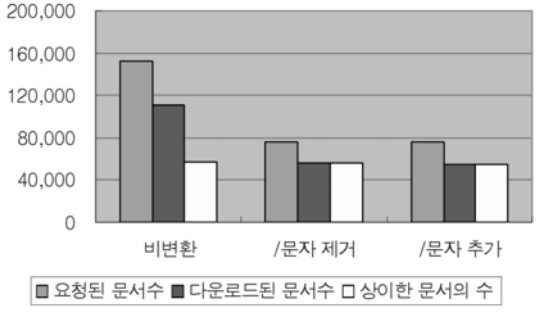

그림 3 마지막 슬래시 문자에 의한 동일 후보 URL의 문서 수집 결과

4.3 'www' 접두어의 유무

대부분의 URL은 'www' 접두사를 붙이거나 붙이지 않은 호스트로 시작한다. 사용자들은 종종 호스트의 접두사 'www'을 생략하여 URL을 요청하거나, 웹 문서에 표기한다. 실제 웹에서 대부분의 웹 서버는 'www' 접두사가 없는 URL의 요청을 수행할 수 있도록 구성되어진다. 예를 들어 "http://acm.org/"와 "http://www.acm.org/"는 실제로 동일한 사이트이다. 본 절에서는 'www'

접두사의 유무만이 다른 URL에 대하여 동일 URL인지를 살펴본다.

본 실험에서는 'www' 접두사 유무만이 다른 두 URL이 하나의 동일 후보 URL 집합을 구성한다. 'www' 접두사 유무의 동일 후보 URL 집합에 대하여 다음의 3가지 방법을 고려한다.

(1) 호스트의 접두사를 변환하지 않는 방법

(2) URL 정규화의 형태로 'www' 접두사를 포함한 형태를 선택한다.
즉, 'www' 접두사를 포함하지 않은 동일 후보 URL에 'www'를 추가한다.

(3) URL 정규화의 형태로 'www' 접두사를 제거한 형태를 선택한다.
즉. 'www' 접두사를 포함한 동일 후보 URL에 'www'를 제거한다.

'www' 접두사의 유무만이 다른 동일 후보 URL 770,092개를 포함하는 집합 385,046개를 실험 대상으로 하였다. 그림 4는 3가지의 방법에 따른 요청 횟수와 다운로드된 문서의 수, 수집된 문서 중 고유한 문서의 수를 나타낸다. 다운로드 성공된 문서는 305,968개의 고유한 문서를 보유하고 있다. 첫 번째 방법은 중복도가 0.41이며, 다운로드된 문서의 약 41%가 중복된 문서라는 것을 의미한다. 두 번째와 세 번째 방법의 커버리지 손실률은 각각 약 0.13과 0.09으로 나타났다.

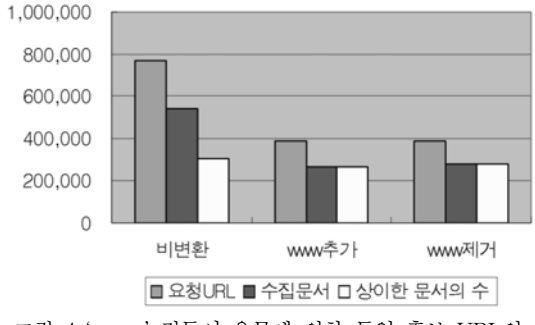

그림 4 'www' 접두어 유무에 의한 동일 후보 URL의 문서 수집 결과

4.4 기본 문서

기본 문서는 웹 클라이언트가 웹 서버에 디렉토리(URL 문자열의 마지막 문자가 슬래시)를 요청하는 경우에 웹 서버가 응답하는 문서이다. 웹 서버 관리자는 기본 문서로 사용될 문서의 파일명을 설정할 수 있다. 기본 문서로 설정된 파일을 요청하는 URL과 기본 문서를 생략하고 디렉토리를 요청하는 두 URL은 동일하다. 예를 들어, 기본 문서가 "index.htm"으로 설정되어 있는 웹 사이트 (www.acm.org)에서 두 URL "http://www.acm.org/"과 "http://www.acm.org/index.htm"은 동일한 URL이다.

실제 웹에서는 어떤 파일도 기본 문서로 사용될 수 있다. [6]의 보고에 따르면, 실제 웹에서 사용되는 웹 서버의 85%를 아파치(Apache)와 IIS(Internet Information Server)가 차지하고 있다. 사용자가 웹 서버의 설정을 변경하지 않고, 두 웹 서버를 설치하는 경우 "index.htm". "index.html", "default.htm"을 기본 문서로 사용하도록 설정된다. 본 논문에서는 기본 문서로서 "index.htm", "index.html", "default.htm" 파일 만을 고려한다.

기본문서 중복 수집 실험에서의 동일후보 URL 집합은 디렉토리를 요청하는 URL u와 URL u에 기본문서 파일명이 추가된 URL의 집합이다. 전체 URL 중에서 305,835개의 동일 후보 URL들이 147,266개의 동일 후보 URL 집합을 형성하였다. 다음 두 가지를 고려하였다.

(1) 발견된 기본 문서 이름을 보존하는 방식

(2) 발견된 기본 문서 이름을 제거하는 방식 (즉, URL의 마지막 문자는 슬래시)

기본 문서의 유무만이 다른 동일 후보 URL 305,835개를 포함하는 집합 147,266개를 실험 대상으로 하였다. 그림 5는 기본 문서의 정규화 방법에 따른 각각의 요청 횟수와 다운로드된 문서의 수, 수집된 문서 중 고유한 문서의 수를 나타낸다. 첫 번째 방법은 중복도가 0.42이다. 두 번째 방법의 커버리지 손실률은 0.19이다. 즉, 두 번째 방법을 사용하는 경우 42%의 중복된 문서를 감소할 수 있으나, 19%의 고유한 문서를 손실할 수 있다는 것을 의미한다.

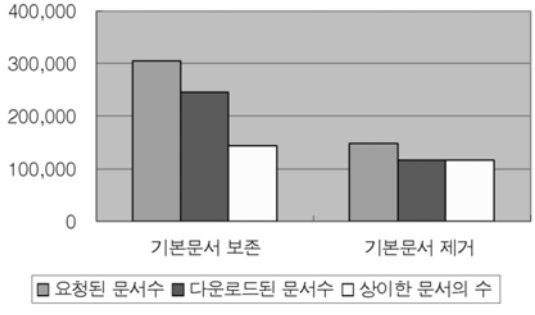

그림 5 기본문서의 동일 후보 URL에 대한 문서 수집 결과

5. 결론 및 향후 계획

본 논문에서는 표준 URL 정규화에서 고려되지 않은 4개의 추가적인 정규화 고려 요소를 기술하였다. 본 논문에서 제안한 두개의 척도를 이용하여, 실제 한국 웹 사이트에서 수집된 URL을 대상으로 각각의 요소에 대한 정규화 수행의 효과를 분석하였다. 실험 결과는 표 1

로 요약하였다. 경로의 모든 문자를 소문자로 변환하는 방법과 마지막 슬래시 문자를 명시하는 방법은 0.01의 적은 커버리지 손실률을 보이면서, 약 50%의 URL 중복을 감소한다. ‘www’ 접두사의 표기 또는 제거하는 방법과 기본 문서를 제거하는 방법은 중복도를 감소할 수 있으나, 약 10% 이상의 많은 문서에 대하여 수집 손실이 발생하였다.

표 1 중복 수집 요인에 대한 비교 결과

중복 수집 요인	문자열 변환 방법	중복도	커버리지 감소률
경로의 대소문자 구분	비변환	0.52	0
	소문자로 변환	0	0.01
	대문자로 변환	0	0.11
마지막 슬래시 문자	비변환	0.49	0
	슬래시 제거	0	0.01
	슬래시 추가	0	0.03
‘www’접두어	비변환	0.41	0
	‘www’ 추가	0	0.13
	‘www’ 제거	0	0.09
기본문서	비변환	0.42	0
	기본문서 제거	0	0.19

제한된 범위 내의 “잘못된 긍정”의 유발을 허용하여, 상당량의 중복 URL의 감소 효과를 얻을 수 있는 본 논문의 정규화 방법은 경로의 대소문자 구분과 마지막 슬래시 문자의 두 가지 경우에 매우 잘 부합된다. 두 가지 경우 모두 약 1%의 다운로드 가능한 문서의 손실을 허용하여 다운로드된 문서의 크기를 약 50% 감소할 수 있었다. 본 논문의 실험 결과는 대용량의 URL을 처리하는 웹 데이타베이스 어플리케이션에 중요한 연관을 가진다.

“잘못된 긍정”을 허용하여, “잘못된 부정”을 감소하는 본 논문의 정규화 방법의 웹 어플리케이션에 대한 적용은 웹 어플리케이션의 목적에 달려있다. 만약, 정확성보다 효율성이 더 중시되는 웹 어플리케이션이라면 URL 정규화 방법으로 본 논문에서 제안한 방법은 좋은 선택이 될 것이다. 그러나 효율성 보다 정확성이 더 우선시되는 웹 어플리케이션이라면, 표준 정규화 방안을 선택하여야 한다.

URL 정규화는 여러 단계의 문자열 변환을 수행한다. 하나의 URL은 문자열 변환이 적용되는 순서에 따라 상이한 URL로 정규화될 수 있다. 향후, [3]에서 기술된 정규화 고려요소와 본 논문에서 기술된 추가적인 정규화 고려요소들의 간의 효과적인 적용 순서에 대한 연구가 필요하다.

참 고 문 헌

[1] T. Berners-Lee, R. Fielding and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax," http://www.ietf.org/rfc/rfc3986.txt? number=3986, 2005.

[2] M. Burner, "Crawling Towards Eternity: Building an Archive of the World Wide Web," Web Techniques Magazine, Vol. 2, No. 5, pp. 37-40, 1997.

[3] A. Heydon and M. Najork, "Mercator: A Scalable, Extensible Web Crawler," International Journal of WWW, Vol. 2, No. 4, pp. 219-229, 1999.

[4] S.J. Kim and S.H. Lee, "How Web Pages Change: An Empirical Study," submitted for publication.

[5] S.J. Kim and S.H. Lee, "Implementation of a Web Robot and Statistics on the Korean Web," Proc. Journal of KIPS, Vol. 10-c, No. 4, pp. 509-518, 2003.

[6] Netcraft. http://news.netcraft.com/archives/web_server_survey.html

[7] V. Shkapenyuk and T. Suel, "Design and Implementation of a High-performance Distributed Web Crawler," Proc. 18th Data Engineering Conf., pp. 357-368, 2002.

홍 석 후

2003년 숭실대학교 컴퓨터학부(학사). 2005년 숭실대학교 대학원 컴퓨터학과(석사) 2005년~현재 SK Communications

김 성 진

1998년 숭실대학교 소프트웨어공학과(학사). 2000년 숭실대학교 대학원 컴퓨터학과(석사). 2004년 숭실대학교 대학원 컴퓨터학과(박사). 2004년~현재 서울대학교 제어계측신기술연구소 연구원

이 상 호

1984년 서울대학교 전산공학과(학사) 1986년 미국 노스웨스턴대 전산학과(석사). 1989년 미국 노스웨스턴대 전산학과(박사). 1990년~1992년 한국전자통신연구원, 선임연구원. 1999년~2000년 미국 조지메이슨대 소프트웨어정보공학과 교환 교수. 1992년~현재 숭실대학교 교수