엔트로피를 기반으로한 Web 문서들의 복잡도 척도

김 갑 수
서울교육대학교 컴퓨터 교육과

요 약

본 연구에서는 HTML이나 XML로 작성한 Web 문서들의 복잡도를 측정하는 모델을 제안한다. 문서들의 복잡도는 문서들을 이해하는 데 밀접한 영향을 미치고, 이 이해도가 높은 Web 문서들은 결국 WEI에 좋은 효과를 거둘 수 있다. 본 연구에서 제안한 복잡도는 Web 문서들 간의 주고받는 정보의 흐름의 정도를 표현하기 위하여 엔트로피의 함수를 이용한다. 제안한 문서 복잡도는 문서들간의 정보 이동 관계에 의해서 문서들 내의 정보 흐름을 측정한다. 논 연구에서 제안한 문서 복잡도의 타당도는 Weyuker가 제안한 프로그램의 복잡도 평가 방법을 이용하여 평가하였고, 실제 문서들의 복잡도를 측정하였다. 또한 문서화일의 수와 문서 복잡도간의 상관관계를 분석하여 본 연구에서 제안한 문서 복잡도의 효율성을 제시하였다.

A Complexity Metric for Web Documentation Based on Entropy

Kapsu Kim
Seoul National University of Education, Department of Computer Education

ABSTRACT

In this paper, I propose a metric model for measuring complexity of Web documentations which are wrote by HTML and XML. The complexity of Web documentation has effect on documentation understandability which is an important metric in maintenance and reusing of Web documentation. The understandable documents have more effect on WEI. The proposed metric uses the entropy to represent the degree of information flows between Web documentations. The proposed documentation complexity measures the information flows in a Web document based on the information passing relationship between Web document files. I evaluate the proposed metric by using the complexity properties proposed by Weyuker, and measure the document complexity. I show effectiveness of analyzing the correlation between the number of document file and document complexity.

1. 서론

인터넷 환경 하에서 모든 생활이 변화하고 있다. 교육 환경도 마찬가지이다. 현재 교육용 Web 문서들을 많이 생성하고 이를 현장에 이용하고 있다. 그렇지만, Web 문서들을 정량적인 평가 기준이나, 문서들의 품질을 고려한 Web 문서들의 복잡도 등에 대한 연구가 전혀 되어 있지 않다. 그렇기 때문에 교육용 Web 문서들을 작성할 때에 Web 문서들의 이해도 등을 고려한 Web 문서의 작성 가이드라인이 없다. 교육용 Web 문서들을 이용하여 교육하거나 Web 문서들의 유지보수성을 높이기 위하여 Web 문서들을 이해하는 것이 매우 중요하다. 따라서, 교육용 Web 문서들을 만들 때에 유지보수를 쉽게 하고 많이 사람들이 재사용 하기 위하여 가능한 Web 문서들을 쉽게 이해할 수 있게 작성하여야 하고, Web 문서 관리자는 Web 문서들을 관리하는 측면에서 Web 문서들의 이해도를 측정할 수 있어야 한다. 이를 보통 Web 문서들의 품질 척도(Metric)중에 한 개이며, 복잡도가 그의 대표적인 예이다.

Web 문서들에 대한 평가들에 대한 연구는 대부분 문서들의 정성적인 평가 기분을 제시하고 있다. Hope N. Tillman[1]은 Web 문서들의 평가 기준으로 정보의 정확성, 완전성, 형식의 적합성, 저자 등에 대한 정성적인 평가 기준을 제안하였다. Alexander, Jan, and Marsha Tate[2]의 연구들과 Beck, Susan E[3]의 연구들을 살펴보면 Web 문서들을 평가하는 기준들에 대한 품질 점검 표들을 만들었다. 이 품질 점검 표는 저작권에 대한 것, 정확성에 대한 것, 목적성에 대한 것, 현행성에 대한 것, 범위에 관한 것으로 구성되어 있다. 물론 다른 연구들에 의해서 콘텐트 자체에 대한 점검 표를 만들어서 문서들의 품질을 향상하는 안내 역할을 하고 있다. 그렇지만, 이들을 정량적으로 평가하는 방법 등이 전혀 개발되고 있지 않다. 따라서, 지금까지 Web 문서들의 복잡도를 측정하는 방법은 전혀 제안되고 있지 않다.

좋은 Web 문서들은 이해하기 쉽게 구성되어야 한다. 물론, Hope N. Tillman[1], Alexander, Jan, and Marsha Tate[2] 및 Beck, Susan E[3]등이 제안한 문서들의 품질 조건인 문서에서의 정확성이나 현행 데이터의 유지, 문서의 범위의 명확성, 저자들에 대한 확신성, 목적의 명확성들은 기본 조건을 만족한다고 가정하고 하고, 이런 조건하에서 물질적인 문서의 구성을 효과적으로 함으로서 Web 문서들의 이해도를 높게 하는 방법이 필요하다. 그러므로, 문서의 이해도를 측정하는 방법으로서 문서의 복잡도를 측정하는 것이 필요하다.

본 연구에서는 이런 필요성을 위해서 Web 복잡도를 측정하는 방법을 개발한다. 본 연구에서는 문서들의 구성 요소인 문서 파일들의 구성이 문서의 이해도에 영향을 미치고, 특히 문서간에 주고받는 구성요소들간의 정보의 양에 문서들의 복잡도에 영향을 미친다. 따라서, 본 연구에서는 엔트로피 함수를 이용하여 Web 문서의 복잡도를 계산하는 방법을 제안한다.

본 논문의 구성은 다음과 같다. 제2장에서는 본 연구에서 이용되는 정의와 복잡도 측정에 이용되는 엔트로피 개념을 설명한다. 제3장에서는 Web 문서의 복잡도를 설명한다. 제4장에서는 본 논문에서 제안한 복잡도의 특징을 살펴본다. 제5장에서는 본 논문에서 제안한 복잡도를 Weyuker의 복잡도 특성을 이용하여 검증한다. 제6장에서는 본 연구에서 제안한 복잡도 식들을 Web 문서에 실제 적용하여 문서 복잡도를 검증한다. 제7장에서는 결론을 맺는다.

2. 기본 정의 및 엔트로피 이론

2.1 정의

본 절에서는 본 연구에서 사용하는 정의에 대해서 간단히 설명한다. 일반적으로 Web 문서를 구축할 때에는 어떤 목적 하에 여러 개의 파일들로 구성되게 한다. 따라서, 문서의 정의는 어떤 목적 하에 Web 문서를 구축하고 하는 1개이상의 문서화일들의 집합으로서 다음 정의1과 같이 정의한다.

정의 1 : 문서(Documents)
<Document>= <Document file>+

문서화일의 정의는 문서를 구성하는 구성 요소로

서 하나의 html 파일이다. 문서 파일의 정의는 다음 정의2와 같이 정의한다.

정의 2 : 문서화일(Documents file)
<Document file>= < X.html>

2.2 엔트로피 이론

Shannon이 제안한 엔트로피[4]는 자료 처리나 신호 처리 분야에 많이 이용되고 있는 것으로서 무질서의 정도를 나타내는 것이다. 이 이론은 지금까지 여러 소프트웨어의 프로그램 복잡도 척도에 이용되었다[5][6][7]. Shannon의 정의에 의하면 정보량은 임의의 메세지를 전송할 때에 자주 전송되는 메세지가 자주 전송되지 않는 메세지보다 정보량이 적다는 것으로서 정보량을 다음과 같은 식1로 정의한다. 메세지의 집합 $\{A_1,A_2,,,,,A_n\}$에서 메세지 Ai가 정보를 포함할 확률은 다음 식1과 같다.

$$I(A_i) = -\log_2 P(A_i)$$ ----------식1

엔트로피는 정보량의 평균이므로 다음 식2와 같이 유도된다.

$$H = \sum_{i=1}^{n} I(A_i) \times P(A_i)$$ --------식2

여기에서, n은 메세지들의 수이고, $P(A_i)$는 A_i 메세지가 나타날 확률이다.

식2와 같은 엔트로피 함수에서 메세지의 확률 $P(A_i)$는 문서의 구성요소인 문서 화일들 간에 상호 참조하는 관계에 의해 결정된다. 문서에서 문서 구성요소인 새로운 문서화일을 첨가하면 기존 문서 화일들간의 구성 요소간의 관련성이 증가하기 때문에 문서의 복잡도가 증가하게 된다. 이러한 효과는 엔트로피 함수(H)에서도 메세지의 개수를 증가함으로서 엔트로피 값을 증가시키는 것과 동일하다. 따라서, 문서 복잡도는 엔트로피 함수를 이용하여 측정할 수 있다.

3. 문서의 복잡도

일반적으로 Web 문서는 HTML 또는 XML로 구성된다. 본 연구에서 이용할 문서들은 1개 이상의 문서파일들의 집합이다. Html문서들은 여러 문서파일들이 서로 링크 되어 있다. 이 링크가 문서들을 더욱 복잡하게 만든다. 문서들의 복잡도에 영향을 미치는 요소로서 문서들을 구성하는 문서 파일과 문서화일들간에 링크하는 개수라고 볼 수 있다. 같은 문서화일의 구조도 링크의 개수가 많으면 많을수록 문서의 복잡도는 높다고 볼 수 있다. 따라서, 본 연구에서 문서들의 복잡도에 영향을 미치는 요소로서 문서 파일의 개수와 문서화일들간의 링크로 정의한다.

먼저, 문서 복잡도를 계산하기 위하여 이용되는 정의를 살펴본다. 문서들을 구성하는 요소들의 문서파일들을 이용하여 문서내의 문서파일들 간의 관계를 정의3과 같은 문서 파일 관계 그래프(DR 그래프 - Documentations File Relationship Di-Graph)로 표현한다. 앞으로 문서 파일 관계 그래프를 DR 그래프로 표현한다.

정의 3 : DR 그래프
DR=<N,A,R>, 단 N=<문서>, A= 가중치 있는 호선 R은 노드들 사이의 관계이다.

이 DR 그래프는 방향성 그래프이므로 쉽게 정형화할 수 있다. DR 그래프는 다음 세 개의 집합들로 구성된다.(1) 유한한 노드들의 집합으로 이 노드들은 문서 화일들로 구성된다. (2) 유한한 간선들의 집합으로 두 노드들 사이의 관계를 표현하고 가중치가 있다.(3)관계 R : A--> N × N 는 노드들의 순서화된 쌍이다. 만약 $A_k=(N_i, N_j)$이 존재하면 N_i를 간선 A_k의 시작 노드라 하고, N_j를 간선 A_k의 단말 노드라고 한다. 간선 A_k의 가중치는 DR 그래프에서 (N_i,N_j)의 쌍들의 개 수이다.

어떤 문서의 문서 복잡도 C를 계산할 때에 이용되는 노드는 문서화일이다. 즉, 한 개의 html 화일이다. 간선은 문서화일들간의 종속성을 나타내고, 간선의 가중치는 문서화일들간의 참조하는 하는 빈도 수를 나타낸다. 여기서, 빈도 수는 구문 분석을 통해서

문서들간의 링크 관계를 분석하는 것이다. 이를 그래프로 표현할 때에는 문서는 원으로 표현하고, 간선은 화살표로 표현한다.

문서들의 문서 복잡도를 계산하기 위하여 문서의 구성요소인 문서화일들로부터 DR 그래프를 생성하고, 이를 이용하여 모든 노드들의 참조 확률 값을 계산하여 이들 값을 엔트로피 함수에 대입하여 문서 복잡도를 계산한다. 이에 대한 상세한 절차는 다음과 같다.

먼저, DR 그래프를 구성하기 위한 첫 번째 단계로서 각 문서를 구성하는 문서화일들을 찾아내어 노드들의 집합에 첨가한다. 즉, 복잡도를 계산하기 위한 모든 문서화일들이 노드에 첨가된다. 두번째 단계에서는 문서화일들의 집합인 노드들의 집합의 모든 원소들 간의 관계를 간선과 가중치로 표시한다. 노드 i와 노드 j가 서로 참조하는 관계가 존재하면 이를 간선으로 표시하고, 이들의 가중치는 노드i와 노드j가 서로 참조하는 빈도 수를 계산하여 $W(E_{i,j})$로 표현한다.

예를 들어, 다음과 같은 문서가 있다고 가정한다. 이 문서들은 8개의 문서화일로 구성되어 있고, 각 문서의 구조는 다음과 같다.

<Documents> = <index.html>+<com.html>+<result.html>+<error.html>+<source.html>+<reset.html>+<output1.html>+<output2.thml>

<index.html>=<A HREF= "com.html">+ <A HREF="result.html">+<A HREF= "error.html">+<A HREF= "output1.html">+<A HREF= "output2.thml"> +<A HREF= "reset.html">

<com.html>=<A HREF= "index.html">

<result.html>=<A HREF="index.html">+<A HREF="source.html">

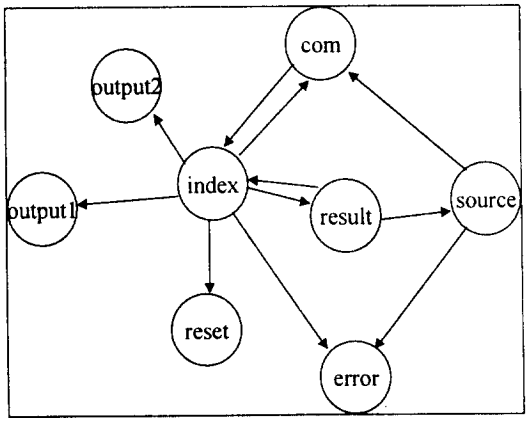
<source.html"> +<A HREF= "com.html">+<A HREF= "error.thml">

위의 문서 구조에서 데이타 index을 노드1라고 하고, error가 노드2라고 하면 이들 간에는 노드2가 노드1을 참조하여 데이타를 기록하기 때문에 이들 간에 간선은 존재하고 간선을 $E_{2,1}$로 표시하며 이들의 가중치($W(E_{2,1})$)는 1이 된다. 이러한 방법으로 모든 데이타 노드들과 함수 노드들 간의 간선과 가중치를 결정함으로써 DR 그래프가 생성된다. 예를 들어 (그림1)은 위의 문서1의 DR 그래프를 나타낸다.

DR 그래프를 생성한 후에 DR 그래프에서 각 노드들에 연결된 간선 수와 각 간선의 가중치를 계산하여 각 노드에서의 참조 확률 값을 계산한다. i 노드에서의 참조 확률 값은 다음 식3으로 정의한다.

$$P(N_i) = \frac{\sum_{j=1}^{n} W(E_{i,j}) + \sum_{j=1}^{n} W(E_{j,i})}{2 \times \sum_{k=1}^{n}\sum_{j=1}^{n} W(E_{k,j})} \quad \text{---식3}$$

DR 그래프에서 총 노드의 수는 n이고, 노드 N_i와 노드 N_j사이에 간선이 없을 경우에 $W(E_{i,j})$의 값은 0이다. 노드 N_i에 연결된 모든 간선들의 가중치는 $\sum_{j=1}^{n} W(E_{i,j}) + \sum_{j=1}^{n} W(E_{j,i})$이고, DR 그래프의 총 간선들의 가중치는 $\sum_{k=1}^{n}\sum_{j=1}^{n} W(E_{k,j})$이다. 따라서 노드 N_i의 참조 확률값은 식3과 같이 정의할 수 있다. 예를 들어, <그림1>을 분석하여 모든 노드들의 참조 확률 값을 계산하면 <표1>과 같다.



<그림1> DR 그래프

마지막으로 각 노드의 참조 확률 값을 엔트로피 함수 식2에 대입한다. 예를 들어, <표1>을 바탕으로

(그림1)과 같은 DR 그래프의 문서 복잡도를 계산하면 식2에 의해서 문서 복잡도는 2.624이 된다.

<표1> 참조 확률 값

노드 이름	참조 확률
index	0.364
source	0.136
result	0.136
com	0.136
reset	0.045
ouput1	0.045
output2	0.045
error	0.091

4 복잡도의 특징

다음은 본 복잡도 척도에 이용되는 엔트로피 함수의 특징으로 복잡도의 특징을 살펴본다.

특징1 : 복잡도(엔트로피)의 최대값 = $\log_2 N$

평가하고자 하는 Web 문서들이 문서화일 N 개의 노드와 E개의 간선으로 구성된 DR 그래프로 표현된다고 가정하자. 이 때에 최대의 엔트로피를 갖는 조건은 각 노드의 참조 확률이 같다는 것이다. 따라서, 각 노드의 참조 확률은 $P(A_1)=1/N$, $P(A_2)=1/N$,,,,,, $P(A_n)=1/N$이다. 엔트로피의 최대값은 다음 식4와 같고, 본 연구에서 제안한 문서 복잡도의 최대 값도 $\log_2 N$이다.

$$H=-\sum_{i=1}^{N}(\frac{1}{N})\log_2(\frac{1}{N})=\log_2(N) \quad \text{--식4}$$

특징2 : 복잡도(엔트로피)의 최소값 = 0

엔트로피의 최소 값은 문서가 한 개의 문서화일로 구성되는 경우 즉, 한 개의 노드만 구성된 경우이다. 이 경우 노드를 참조하는 문서화일이 자신을 참조하지 않는 경우에는 참조 확률 값이 0이거나 자신을 호출하는 경우에는 노드의 참조 확률 값이 1일 수 있다. 따라서, P(N)=0 또는 1이다. 이 경우의 엔트로피 값은 다음 식5와 같다.

한 노드의 참조 확률 값이 0인 경우는 $P(A_1)=0$ 임으로 식5가 0이 된다.

$$H=-\sum_{i=1}^{n}P(A_i)\log_2 P(A_i)=-P(A_1)\log_2 P(A_1) \quad \text{-식5}$$

한 노드의 참조 확률 값이 1인 경우는 $\log_2(1)=0$ 임으로 식5가 0이 된다. 따라서, 본 연구에서 제안한 문서 복잡도의 최소 값은 0이다.

5. 검증

본 연구에서 제안한 Web문서들의 복잡도 척도 방법을 검증하는 방법이 현재는 특별히 존재하지 않는다. 따라서, 컴퓨터 프로그램 복잡도의 타당성을 검증하기 위하여 Weyuker[8]의 복잡도 성질을 이용한다. 물론, Weyuker의 복잡도 성질은 소프트웨어 복잡도에 대한 필요 충분 조건은 아니지만[9], 복잡도의 이론적인 검증의 타당성을 증명하기 위하여 Weyuker의 복잡도 성질을 많이 이용하고 있다[10][11]. 따라서, Weyuker의 복잡도 성질을 Web 문서에 전적으로 적합하지도 않을 수 도 있지만, 본 연구에서 제안한 Web 문서들의 복잡도의 타당도를 증명하는 방법으로 이용한다. Web 문서 복잡도를 검증할 때에, |P|는 P 문서들의 문서 복잡도라고 가정하고, |Q|는 Q 문서들의 문서 복잡도라고 가정한다.

성질1. $(\exists P)(\exists Q)(|P| \neq |Q|)$

의미 : 성질1은 복잡도가 다른 프로그램이 존재함을 나타낸다. 여기서 문서 복잡도가 다른 문서들이 존재한다는 의미다.

증명 : 두 문서 A, 문서 B에서 문서들을 구성하는 문서화일들의 개수가 다르면 DR 그래프에서 노드의 개수와 간선이 다르다. 따라서, 각 노드의 참조 확률 값이 다를 수 있기 때문에 복잡도도 다를 수 있다. 따라서, 문서 복잡도가 다른 문서들이 존재할 수 있다. 그러므로 성질1은 명백하게 성립한다.

성질2. 복잡도가 c인 문서의 수는 유한 개이다. 단, c는 음이 아닌 정수이다.

의미 : 성질2는 문서 복잡도가 같은 문서가 유한 개를 의미한다.

증명 : Web 문서는 가상공간이지만 유한한 문서

화일들로 구성된다. 또한 각 Web 문서화일를 작성할 때, 링크의 갯수가 유한하다. 따라서, 문서 복잡도가 C인 문서들은 유한하다. 그러므로 성질2를 만족한다.

성질3. (∃ P)(∃ Q)(P ≠ Q & |P| = |Q|)

의미 : 성질3은 Web 문서 복잡도는 같지만 다른 기능을 수행하는 Web 문서들이 존재할 수 있다는 의미이다. 여기서 서로 다른 기능을 수행하는 Web 문서이지만 Web 문서 복잡도가 같다는 의미이다.

증명 : 다른 기능을 수행하는 두 문서가 있다고 가정한다. 두 문서들을 구성요소인 문서화일들간의 관계를 표현하는 문서구조는 같을 수 있다. 따라서, 두 Web 문서들의 문서 복잡도는 같다. 그러므로 서로 다른 기능을 수행하는 Web 문서이지만 Web 문서의 복잡도는 같다. 그래서 Web 문서 복잡도는 성질3은 만족한다.

성질4. (∃P)(∃ Q)(P = Q & |P| ≠ |Q|)

의미 : Web 문서 복잡도는 다르지만 같은 기능을 수행하는 문서들이 존재할 수 있다는 의미이다. 즉, 같은 기능이지만 구현하는 방법에 따라서 복잡도가 다를 수 있다. 여기서는 같은 기능을 수행하는 두 문서들의 문서 복잡도가 다른 것이 존재할 수 있고, 같은 기능을 수행하는 두 문서의 복잡도도 다를 수 있다는 의미이다.

증명 : 같은 기능을 수행하는 두 문서이지만, 두 문서를 정의하는 방법에 따라 두 문서들의 링서 개수가 다를 수 있다. 따라서, 두 문서의 DR 그래프가 다를 수 있기 때문에 문서 복잡도가 다르다. 따라서, 같은 기능을 구현하는 방법에 따라서 문서 복잡도가 다르기 때문에 문서 복잡도는 성질4를 만족한다.

성질5. (∀ P)(∀Q)(|P| ≤ |P;Q| & |Q| ≤ |P;Q|)

의미 : 이 성질은 단조성을 만족한다는 의미이다. 임의의 문서에 새로운 문서의 내용을 삽입하면 문서 복잡도가 단조 증가한다. 여기서는 한 문서에서 새로운 파일을 첨가하거나 한 문서에서 문서의 내용을 설명하는 링크의 수를 첨가하여 새로운 문서를 만들면 새로 만든 문서의 복잡도가 단조 증가한다는 의미이다.

증명 : 한 문서(P)에 새로운 화일(Q)을 삽입하여 새로운 문서(P;Q)를 만든 경우에 기존 문서의 DR 그래프에 새로운 문서의 DR 그래프가 첨가하기 때문에 노드의 수가 많아진다. 노드 수가 많아진다고 반드시 문서의 복잡도가 증가한다고 볼 수 없다. 왜냐하면, 현재 5개의 노드를 갖는 DR 그래프에서 노드 N_1, N_2, N_3, N_4, N_5의 참조 확률이 모두 0.2다고 가정하면 이 DR 그래프의 문서 복잡도는 2.322이다. 이 그래프에 새로운 노드 N_6을 첨가하고, 이 노드 N_6과 DR 그래프의 모든 노드들과 간선의 가중치가 10으로 연결하면 새로 생성된 DR 그래프의 노드 N_1, N_2, N_3, N_4, N_5의 참조 확률이 모두 0.105이고, 노드 N_6의 참조 확률은 0.475이다. 따라서 새로 생성된 DR 그래프의 문서 복잡도는 2.217이다. 그러므로, 문서 복잡도는 단조 증가한다고 볼 수 없다. 그러므로 문서 복잡도는 성질5를 만족한다고 볼 수 없다.

성질6.a. (∃P)(∃ Q)(∃R)(|P|=|Q| & |P;R| ≠ |Q;R|)

의미 : 이 성질은 같은 복잡도를 갖는 두 문서에 새로운 문서화일을 두 문서 뒤에 삽입하면 복잡도가 다르다는 의미이다. 여기서의 의미는 다른 기능을 수행하는 두 문서에 새로운 문서를 첨가하여 만든 새로운 두 문서의 복잡도가 다를 수 있다는 의미이다.

증명 : 문서 복잡도가 같고 다른 기능을 수행하는 두 문서 P, Q에 새로운 문서 R을 각각 첨가할 때에 문서 P와 문서 R의 구성 요소가 같은 문서 파일이 존재할 수 있거나 문서 Q와 문서 R의 구성 요소가 같은 문서 화일이 존재하지 않을 수 있다. 그러므로 문서 P에 문서 R을 첨가했을 때의 DR 그래프와 문서 Q에 문서 R을 첨가했을 때의 DR 그래프에서 노드들의 수가 다를 수 있다. 따라서, 각 노드의 참조 확률 값도 다른 것이 존재할 수 있다. 그러므로 두 문서 P, Q에 새로운 문서 R을 각각 첨가할 때에 문서 P와 문서 R을 합친 새로운 문서 복잡도와 문서 Q와 문서 R을 합친 새로운 문서 복잡도가 다를 수 있다. 예들 들어, 문서 P의 구성 요소들을 p_1, p_2, p_3, p_4라고 가정하고 이들의 참조확률은 각각 0.25라

고 가정하고, 문서 Q의 구성 요소들을 q_1, q_2, q_3, q_4라고 가정하고 이들의 참조확률은 각각 0.25라고 가정하고, 문서 R의 구성 요소들을 r_1, r_2, r_3, r_4라 가정하고, 이들의 참조 확률도 0.25라고 가정한다. 그러면, 문서 P, Q, R의 문서 복잡도는 모두 2이다. 만약 문서 P의 구성요소인 p_1과 문서 R의 구성 요소인 r1이 같은 것이고, 문서 P의 구성 요소인 p_2와 문서 R의 구성요소인 r_2가 같은 것이고, 나머지 구성 요소들은 서로 다른 구성 요소들이라고 가정한다. Q 문서와 R 문서를 합병한 새로운 문서는 8개의 노드(q_1, q_2, q_3, q_4, r_1, r_2, r_3, r_4)를 갖고 각 노드의 참조 확률은 0.125가 되기 때문에 문서 복잡도는 3이고, P 문서와 R 문서를 합병한 새로운 문서는 6개의 노드(p_1, p_2, p_3, p_4, r_3, r_4)를 갖고 p_3, p_4, r_3, r_4 노드의 각각의 참조 확률은 0.125이고 p_1과 p_2의 노드의 참조 확률은 각각 0.25이기 때문에 문서 복잡도는 2.5이다. 따라서 P 문서와 R 문서를 합병한 문서와 Q 문서와 R문서를 합병한 문서의 문서 복잡도는 다르기 때문에 문서 복잡도는 성질6.a를 만족한다.

성질6.b. (∃ P)(∃ Q)(∃ R)(|P|=|Q| & |R;P| ≠ |R;Q|)

의미 : 이 성질도 같은 복잡도를 갖는 두 문서에 새로운 문서를 두 문서 앞에 삽입하면 복잡도가 다르다는 의미이다.

증명 : 성질6.b는 성질6.a와 마찬가지로 증명할 수 있다.

성질7. 현재의 문서 P에서 문장(Statement)의 순서를 조합하여 새로 만든 문서들을 Q라 하면, 두 프로그램P, Q 의 복잡도는 다르다.

의미 : 이 성질도 문서를 구성하는 문서화일의 순서에 따라서 복잡도가 다르다는 의미이다.

증명 : 문서 복잡도는 문서를 구성하는 문서화일들의 구성 순서에는 아무 상관없다. 따라서, 본 연구에서 제안한 목잡도 측도는 성질을 만족하지 않는다.

성질8. 프로그램 P는 프로그램 Q의 변수 이름을 바꾼 프로그램이라면, 두 프로그램 P, Q의 복잡도는 같다.

의미 : 성질8은 문서에서 문서화일의 이름과 문서화일 내용을 구성하는 문서상의 이름을 바꾸어도 복잡도는 변하지 않는다는 의미이다.

증명 : 문서를 구성하는 문서화일들의 이름과 문서화일의 내용을 구성하는 이름을 변경하여도 문서의 DR 그래프가 변경되지 않기 때문에 성질8을 만족한다.

성질9. (∃ P)(∃ Q)(|P| + |Q| < |P; Q|)

의미 : 이 성질은 두 문서를 결합하여 한 문서로 만들면 두 문서 각각의 문서 복잡도의 합보다 결합한 문서의 복잡도가 더 큰 것이 존재할 수 있다는 의미이다.

증명 : 문서 P, 문서 Q가 있다고 가정하면, 이 문서를 합병하여 새로운 한 개의 문서를 만든다면 가정한다. 만약 P문서의 구성요소인 노드가 4개이고, Q 문서의 구성요소인 노드가 3개라하고, P 문서의 노드1의 참조 확률 값은 0.7이고, 노드2, 노드3과 노드4의 참조 확률 값은 0.1이라고 가정하고, Q 문서의 노드1과 노드2의 참조 확률 값은 0.1이고, 노드3의 참조 확률 값은 0.8이라고 가정한다. 이 때에 P 문서의 문서 복잡도는 1.36이고, Q 문서의 문서 복잡도는 0.92이고, 이들의 합은 2.28이다. 여기서 P 문서와 Q 문서를 합하여 새로운 문서 K를 만들었다고 가정한다. 새로 만든 문서는 기존 P 문서와 Q 문서의 모든 노드들로 구성되고, P 문서의 노드들과 Q 문서의 노드들 간의 호출 관계가 있을 수 있기 때문에 새로 만든 문서 K의 7개 노드 모두의 참조 확률 값이 같을 수 있다. 합병한 문서의 최대 문서 복잡도는 2.81이다. 따라서 복잡도 특징1에 의해서 두 문서의 문서 복잡도 합은 두 문서를 합하여 새로운 문서를 만든 문서의 문서 복잡도가 증가할 수 있기 때문에 문서 복잡도는 성질9를 만족한다.

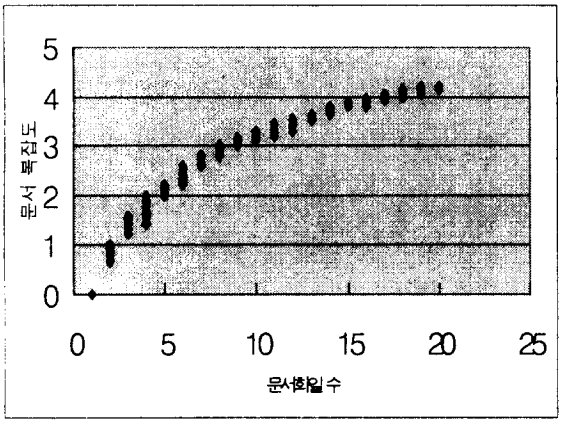
본 연구에서 제안한 문서 복잡도 척도는 Weyuker의 성질5와 성질7을 만족하지 않는다. 성질5의 경우에는 일반적으로 만족하지 않을 뿐이지만 6장에서 설명한 실제 모의 실험에서는 상관관계가 매우 높다는 것을 알 수 있다. 성질7은 일반적으로 객체지향 프로그램에서는 만족하지 않는 성질이다. 따라서, 문서도 객체로 볼 수 있기 때문에 이 성질이 만족하지

않는 다고 본 연구에서 제안한 복잡도 척도가 의미 없다는 것은 아니다. Shyam과 Chris가 제안한 객체 지향 프로그래밍의 복잡도 척도들은 Weyuker의 성질7과 성질9는 만족하지 않는다[12]. 그렇지만 이 복잡도 척도는 객체지향 프로그램을 개발할 때에 많이 사용하는 척도이다.

따라서, 본 연구에서 제안한 문서 복잡도는 문서의 복잡도로서 적합하다고 볼 수 있다.

6. 실험적인 검증

먼저 본 연구에서 문서들의 문서 복잡도를 계산한다. 문서들은 주변에 있는 Web 문서들을 뽑아서 문서 복잡도에 대해서 실험적인 계산을 하여 복잡도를 검증하였다. 본 연구는 Web 문서들의 225개에 대한 링크 문서 자료를 이용하였다. 이들 문서들에 대해서 문서 복잡도를 계산한 결과는 <그림2>와 같다.



<그림 2> 문서 복잡도와 문서화일수의 관계

<그림2>를 분석하여 살펴보면, 일반적으로 문서들의 구성 요소의 수를 증가하면 문서의 복잡도가 증가한다는 것을 알 수 있다. 이 자료들을 이용하여 문서 복잡도를 검증하기 위하여 Pearson의 상관계수(Correlation coefficient R)를 이용한다. 이 분석 방법은 두 변수사이의 선형 관계의 정도를 추정하는 방법으로 일반적인 복잡도의 실험값 검증에 많이 사용하고 있다. 본 연구에서의 문서화일 수 증가와 문서 복잡도간의 Pearson 의 상관 관계를 계산하니 0.949이다. 이 것은 상관 관계가 매우 높다는 것을 알 수 있다. 따라서, 5장의 성질5는 수학적으로는 만족하지 않지만 실험적인 검증으로는 만족한다고 볼 수 있다.

7. 결론

본 연구에서는 엔트로피의 개념을 이용하여 Web 문서들의 복잡도를 측정하는 방법을 제안하였다. 문서 복잡도는 한 문서들의 구성 요소인 문서 화일들 간의 상호 연계 관계를 분석하여 DR그래프를 작성하고, 이 그래프의 노드의 참조 확률 값을 계산하여 이를 엔트로피 함수에 대입하여 문서 복잡도를 계산하는 것이다.

본 연구에서 제안한 문서 복잡도 척도는 Weyuker의 성질5와 성질7을 만족하지 않는다. 성질5의 경우에는 실험적인 검증에 의해서는 만족한다고 볼 수 있다. 즉, 문서화일의 수와 문서 복잡도의 상관 관계가 0.949이기 때문에 통계적인 측면에서는 만족한다고 할 수 있다. 따라서, 성질7은 객체지향 프로그램에서도 일반적으로 만족하지 않고, 문서는 객체로 간주할 수 있기 때문에 성질7은 만족하지 않아도 상관없다. 따라서, 본 연구에서 제안한 문서 복잡도는 의미 있다.

따라서, 본 연구에서 제안한 문서 복잡도를 이용하여 Web 문서를 효과적으로 작성하는 데 가이드라인이 될 수 있다.

앞으로의 연구는 본 연구에서 제안 복잡도들은 작성하는 문서들의 링크 구조만을 다루었다. 특히, 외부 사이트에 대한 링크는 고려하지 않았다. 또한 문서의 링크들은 같은 가중치를 두어서 계산하였다. 앞으로 이를 확장한 모델을 만들어서 이해도를 평가하는 방법을 연구해 보는 것이다.

참고문헌

[1] Tillman, Hope N. "Evaluating Quality on the Net." From a paperpresented at Computers in Libraries, Hyatt Regency Crystal City,Arlington, Virginia, Monday,February 26,

1996. http://www.tiac.net/users/hope /findqual.html

[2] Alexander, Jan, and Marsha Tate. "Teaching Critical Evaluation Skillsfor World Wide Web Resources." Computers in Libraries16, no. 10 (November/December 1996): 49-55.

[3] Beck, Susan E. "Examples." The Good, The Bad & The Ugly: or, Why It's a Good Idea to Evaluate Web Sources. July 1997. http://lib.nmsu.edu/staff/susabeck/evalexpl.html

[4] C. E. Shannon, " A mathematical theory of communications," Bell System Technical Journal, Vol. 27, pp. 379-423, 1948.

[5] John Stephen Davis and Richard J. Leblanc, " A Study of the Application of complexity Measures," IEEE Transactions on Software Engineering, Vol.14, No.9, pp.1366-1372, 1988.

[6] Pierre N.Robillard and Germinal Boloix, " The Interconnectivity Metrics: A new Metric Showing How a program is Organized," J. Systems Software Vol.10, pp.29-39, 1989.

[7] Srinivasarao Damerla and Sol M.Shatz, "Software Complexity and Ada Rendezvous: Metrics Based on Nondeterminism", J. Systems Software Vol.17, pp.119-127, 1992.

[8] Elaine J. Weyuker, " Evaluating Software Complexity Measures," IEEE Transactions on Software Engineering, Vol.14, No.9, pp.1357-1365, 1988.

[9] John C. Cherniavsky and Carl H. Smith, " On Weyuker's Axioms For Software Complexity Measures," IEEE Transactions on Software Engineering, Vol.17, No.6, pp.636-638, 1991.

[10] K.B.Lakshmannan, S.Jayaprakash and P.K.Sinha, "Properties of Control-Flow Complexity Measures," IEEE Transactions on Software Engineering, Vol.17, No.12, pp.1289-1295, 1991.

[11] R.beth McColl and James C. McKim, Jr, "Evaluating and Extending NPath as a Software Complexity Measure," J. Systems Software Vol.17, pp.275-279, 1992.