압축을 이용한 WML 컨텐츠 인코딩의 성능분석

반영미° 최창열
강원대학교 정보통신공학과

Performance Analysis of the WML Contents Encoding based on Compression

Young Mi Bahn Chang Yeol Choi
Dept. of Information and Telecomm. Engineering, Kangwoon National Univ.

요 약

무선 인터넷을 지원하는 WAP 게이트웨이는 프로토콜 변환과 함께 WML 파일을 바이너리 포맷으로 변환하여 무선 환경에 적합하게 한다. 인코딩에서의 공백제거와 태그변환에 따른 효과는 WML 파일의 크기와 특성에 따라 달라진다. 본 논문에서는 게이트웨이에서의 인코딩에 따른 파일크기 감소 효과를 분석하고 압축의 효용성에 대해 기술한다.

1. 서 론

무선 인터넷을 지원하는 표준으로 자리잡아 가고 있는 WAP(Wireless Application Protocol) 기반의 솔루션과 컨텐츠들이 많이 개발되고 있으며, 이러한 움직임은 앞으로 더욱 활발해질 것으로 보인다.

무선 인터넷 서비스의 초기 단계에서는 간단한 메시지 형태의 문서를 전송하는 것이 대부분이었으며 문서 내의 태그의 비중이 대체로 높았다. 최근 들어 기존 인터넷 환경에 친숙한 사용자들의 욕구 증대와 단말기의 성능 향상으로 전자우편과 같은 비교적 장문의 데이터 전송이 가능해졌을 뿐 아니라 한번에 전송할 수 있는 데이터의 크기를 나타내는 덱(deck)의 크기도 증가하고 있다.

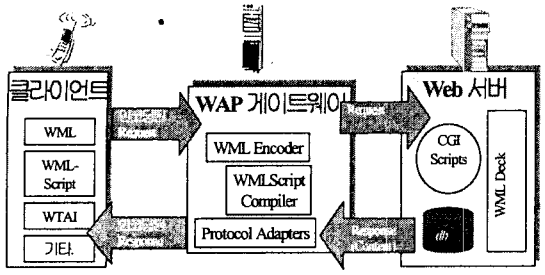
WAP 컨텐츠는 WML을 이용해 제작된다. 앞으로 무선 망과 단말기의 지속적인 발전에 따라[2] 다양한 멀티미디어서비스와 자유로운 유무선 인터넷 접속이과 함께, 텍스트 데이터의 양이 더욱 증가될 것으로 예상된다.

한편 WAP 게이트웨이의 인코딩에서는 무선 망으로 전송할 데이터의 크기를 줄이기 위해 태그만 바이너리 포맷으로 변환하고 텍스트 데이터는 그대로 전송하고 있다. 데이터의 양이 증가되고 미디어가 다양해지면 단순히 변환만으로 파일의 크기를 줄이는 데는 한계가 있을 것이다. 이러한 배경에서 WML과 WML 스크립트를 압축하고자 하는 시도[1]가 있었으나 게이트웨이에서의 공백제거와 태그변환은 고려하지 않고 데이터 압축 알고리즘을 적용하는데 초점을 맞추고 있다.

본 논문에서는 게이트웨이에서의 인코딩에 따른 WML 파일크기 감소 정도를 측정 분석하고, 널리 사용되고 있는 데이터 압축 알고리즘들을 적용할 때의 유용성과 전송시간 단축에 대해 기술한다.

2. WAP의 구조 및 동작

WAP 환경은 [그림 1]과 같이 구성되며, 게이트웨이에서 데이터의 인코딩과 디코딩을 하여 무선 망으로 보내는 데이터의 양을 줄이는 등 무선 망 특성에 최적화되어 있다[3].



[그림 1] WAP 프로그래밍 모델

인코더는 무선 망으로 전송되는 데이터의 크기를 줄이기 위해서 WAP 컨텐츠를 바이너리 포맷으로 변환하고 디코더는 인코딩되기 이전의 의미로 해석하여 변환한다[3]. 현재 게이트웨이에서는 WML 파일에서 정의되지 않은 태그를 무시하고 불필요한 공백을 제거하는 공백제거와 태그들을 사전에 정의된 토큰 값으로 변환하는 태그변환 방식으로 인코딩하여 [5] 의미 정보와 기능의 손실 없이 컴팩트한 바이너리 포맷으로 변환하며, 변환 방법은 바이너리 XML 컨텐츠 포맷 스펙 [4]에 정의하고 있다.

3. WAP 컨텐츠의 압축

초기 무선 인터넷 서비스의 대부분은 간단한 메시지 형태의 문서를 전송하는 것으로 태그의 비중이 높았다. 현재 게이트웨이에서의 인코딩은 태그만 바이너리 포맷으로 변환하고 텍스트 스트링은 그대로 전송되고 있다. 최근 들어 사용자 욕구의 증대와 단말의 발전으로 브라우저와 디스플레이 환경이 좋아지면서 덱(deck) 크기가 증가하고 전자우편과 같은 비교적 장문의 데이터 전송이 가능해졌으며, 앞으로 텍스트 데이터의 양이 더욱 증가될 것으로 예상된다. 따라서 단순히 변환을 통해 파일의 크기를 줄이는 인코딩이 아닌 압축을 통해 압축율과 전송속도 면에서 이득을 얻고자 하는 접근이 자연스러울 수 있다.

많이 사용되고 있는 데이터 압축 알고리즘으로는 허프만, 적응형 허프만, 산술부호, RLE(Run Length Encoding), gzip, bzip2, LZ77, LZSS, LZ78, LZW 등이 있다[8,9]. Gzip과 bzip2[10,11]은 각각 LZ77와 허프만, BWT와 허프만 알고리즘을 조합한 것으로, bzip2는 gzip보다 2배의 압축 속도와 6배의 압축해제 속도를 갖는다[11]. 한편 LZSS는 압축해제 속도가 빠른 것으로 알려져 있다[8,9].

WML 컨텐츠의 압축에서는 압축율 뿐만 아니라 단말에서의 압축 해제 시간도 고려해야 하며, 압축한 데이터의 전송시간과 압축해제 시간을 합한 시간을 TAT(Turn-Around Time)라 한다[1]. TAT는 식 (1)과 같이 나타낼 수 있다.

$$TAT = cr*size(AC)/Speed_tr + cr*size(AC)/Speed_dec \quad (1)$$

cr : 압축율(=압축된 크기/원래 크기)

size(AC) : WML 응용 컨텐츠(AC)의 크기

Speed_tr : 무선 망 전송속도

Speed_dec : 단말의 압축해제속도

압축을 하는 경우에는 식 (2)와 같이 압축을 하지 않았을 때보다 최소한 TAT가 작아야 효용성이 있다.

$$size(AC)/Speed_tr > cr*size(AC)/Speed_tr\ cr*size(AC)/Speed_dec \quad (2)$$

4. 실험 및 결과 분석

4.1 실험 환경 및 고려사항

압축율을 구하기 위해 바이너리 XML 컨텐츠 포맷 스펙 에 따라 공백제거와 태그변환 기능을 구현하였다.

실험에 사용한 10개의 WML 파일은 Phone.com UP.SDK 4.0[6]과 Nokia WAP Tool Kit 2.0[7]의 예제들을 사용했으며, 크기는 [표 1]과 같이 2650~294바이트이다. 시험용 파일들 중 f2, f3, f6는 문서 내 태그의 비중이 크고, f7~f10은 태그보다 텍스트 데이터를 많이 포함하고 있다.

[표 1] 실험에 사용된 WML 파일과 크기

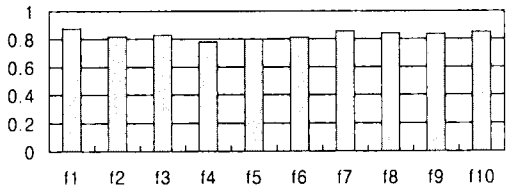
파일	F1	F2	F3	F4	F5
크기	2650	1955	1661	1537	1458
파일	F6	F7	F8	F9	F10
크기	987	872	715	556	383

4.2 실험결과

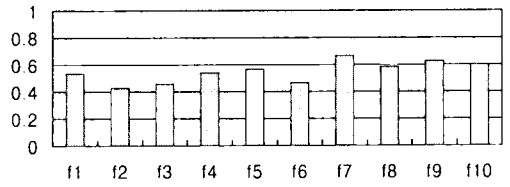
4.2.1 공백제거 및 태그 변환 효과

[그림 2]는 WML 파일에 존재하는 불필요한 공백을 제거하였을 때 얻을 수 있는 효과를 보이고 있다. 파일 내의 정의되지 않은 태그를 무시하고, WML 태그들 사이에 존재하는 공백을 제거함으로써 파일의

크기에 따라 0.78 ~ 0.87 정도의 파일크기 감소 효과를 얻을 수 있다.



[그림 2] 공백제거에 따른 압축율



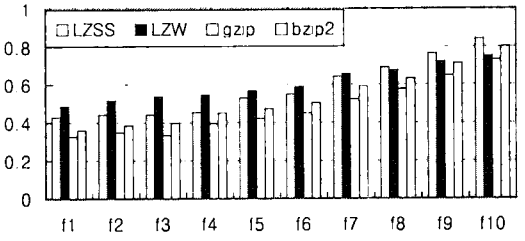
[그림 3] 인코딩(공백제거+태그변환)에 따른 압축율

[그림 3]은 WML 파일의 공백을 제거한 후 태그를 간단한 토큰으로 치환, 즉 인코딩함으로써 얻을 수 있는 파일크기 감소 정도를 보이고 있다. f2와 같이 많은 태그들로 구성된 파일의 경우에는 최대 0.43으로 파일크기 감소 효과를 얻을 수 있다. 태그의 양보다 텍스트 데이터의 양이 많다면 태그 변환만으로는 10% 내외의 효과를 얻을 수 있는 것으로 분석된다.

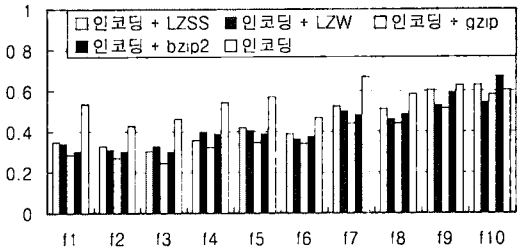
4.2.2 압축 알고리즘별 압축율

WML 파일에 대해 공백제거나 태그변환을 하지 않고 압축했을 때의 압축율은 [그림 4]와 같다. 무선 인터넷 환경에서 많이 사용되고 있는 크기가 대체로 작은 파일들에 대한 압축효과가 있는 알고리즘들만의 결과를 보이고 있으며, 파일 크기가 작을수록 압축효과가 적게 나타나는 것을 볼 수 있다. 한편 허프만, 산술부호, RLE 등은 파일의 크기가 작아짐에 따라 압축 후의 파일 크기가 더 커지는 현상을 보였다.

LZSS와 LZW 알고리즘은 파일의 크기가 클수록 비교적 좋은 압축율을 보이고 있으며, gzip과 bzip2 역시 파일 크기가 클수록 좋은 압축율을 보이는 것은 물론 LZSS와 LZW에 비해 전체적으로 압출율이 좋다.



[그림 4] 데이터 압축 알고리즘에 따른 압축율



[그림 5] 인코딩과 압축에 따른 압축율

[그림 5]는 인코딩하여 바이너리 포맷화된 결과를 압축했을 때의 압축율을 보이고 있다. 파일 크기에 따른 압축율의 차이가 줄어든 것을 알 수 있는데, 이것은 파일 크기에 무관하게 문서 내의 태그의 비중에 따라 파일크기의 감소 정도가 달라지기 때문이다.

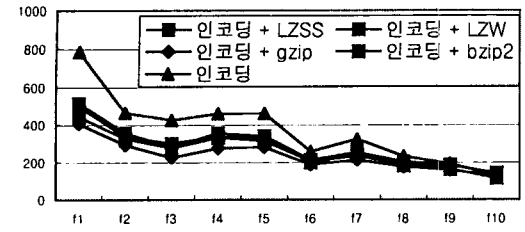
4.2.3 전송시간 비교

WAP 게이트웨이에서의 인코딩과 압축은 전송 파일의 크기를 작게 하여 무선 망의 부하를 줄여 전송속도를 빠르게 하고자 하는 것이다. 앞에서 살펴본 압축율을 바탕으로 전송시간의 이득에 대해 논한다. 회선교환 방식의 IS 95-A와 B, 그리고 패킷교환 방식의 IS 95-C는 각각 ~ 14.4Kbps, 64Kbps, 그리고 144Kbps의 전송속도(ms)를 갖는다. WML 파일의 크기를 각 무선 망의 전송속도로 나눈 값으로 각각에 대한 전송시간(*msec*)을 구한 결과는 [그림 6], [그림 7], [그림 8]과 같다.

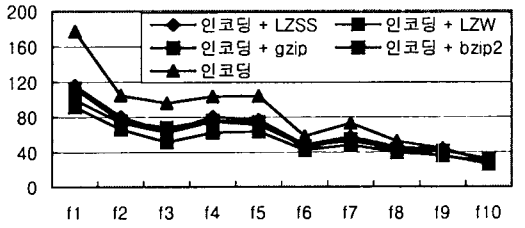
이 결과들로부터 인코딩만 했을 때의 전송시간(t_{enc})과 인코딩과 압축을 함께 했을 때의 전송시간($t_{enc+comp}$)의 차이보다 단말의 압축해제 시간(t_{dec})이 작으면, 아래 식과 같이, 압축을 통한 그 차이만큼의 이득을 얻을 수 있다.

압축으로 얻을 수 있는 시간 이득 =

$$(t_{enc} - t_{enc+comp}) - t_{dec}$$



[그림 6] 전송속도~ 14.4Kbps에서의 전송시간



[그림 7] 전송속도 64Kbps에서의 전송시간

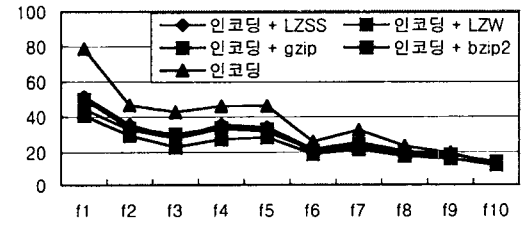


그림 8] 전송속도 144Kbps에서의 전송시간

무선 망의 전송속도가 빨라질수록 그리고 파일 크기가 작아질수록 t_{enc} - $t_{enc+comp}$가 작아지므로 더 빨리 디코딩을 해야만 압축에 따른 이득을 얻을 수 있음을 알 수 있다. 14.4kbps에서는, 파일 크기가 2650바이트(f1)와 987바이트(f5) 경우 각각 200~271ms, 120~180ms 이내에 디코딩이 되면 이득이 있다. 64kbps에서는, 파일 크기가 2650바이트(f1)와 987바이트(f5) 경우 각각 61~86ms, 27~40ms 이내에, 그리고 144kbps의 경우는 각각 27~38ms, 12~18ms이내에 디코딩이 되면 이득이 있다.

5. 결 론

WAP 게이트웨이의 인코딩 단계에서 사용되고 있는 공백제거와 태그변환이 어느 정도의 파일크기 감소를 가져오는 지와 서로 다른 전송속도를 갖는 무선 망에서 데이터 전송시간에 미치는 영향을 분석하였다. 또한 일반적으로 많이 사용하고 있는 데이터 압축 기법들을 WML 컨텐츠에 적용할 때의 효용성과 고려사항들에 대해 논하였다. 앞으로 무선 인터넷을 통해 전송되는 파일의 크기가 커지고 단말의 성능이 향상될 것으로 예상되므로 게이트웨이에서의 압축을 통한 전송시간의 단축이 가능할 것이다.

WML 파일을 압축하여 응답 시간을 단축시키기 위해서는 압축율과 전송시간 뿐만 아니라 단말에서의 압축해제 시간에 대한 고려도 면밀하게 해야 한다. 따라서 단말에서의 오버헤드를 보기 위한 시간 및 공간 복잡도, 디코딩 기법 등에 대한 연구와 WML과 WMLscript에 고유한 압축 기법에 대한 연구는 향후 과제로 남긴다.

참 고 문 헌

[1] Eetu Ojanen and jari Veijalaine, Compressibility of WML and WMLScript byte code : Initial results , *Proceedings of the 10th international Workshop on Research issues in Data Engineering*, pp. 55-62, Feb. 2000.

[2] Pual Lettieri and Mani B. Srivastava, Advances in Wireless Terminals, *IEEE personal Communications, V.6, N.1*, pp.6-19, Feb. 1999.

[3] WAP White Paper : Wireless Internet Today, June 2000, http://www.wapforum.org/what/

[4] WAP Binary XML Content Format Specification version 1.3, May 2000, http://www.wapforum.org/

[5] Wirelss Markup Language version 1.3, WAP Forum, February 2000, http://www.wapforum.org/

[6] http://www.phone.com/index.html

[7] http://www.nokia.com/main.html

[8] http://dogma.net/DataCompression/

[9] Haruhiko Okumura, Data Compression Algorithms of LARC and Lharc , the Science SIG of PC-VAN, April 1989.

[10] The gzip homepage, http://www.gzip.org/

[11] http://sources.redhat.com/bzip2/index.html