Regular paper http://dx.doi.org/10.13067/JKIECS.2015.10.9.1019

UTF-8 부호의 HDB-3스크램블링 최소화를 위한 문자의 원천부호화 규칙

홍완표*

Source Coding Rule of Characters to Minimize HDB-3 Scrambling in Line Coder for UTF-8 code

Wan-Pyo Hong*

요 약

본 논문은 UTF-8 부호화된 문자의 원천부호가 회선부호기에 입력될 때 HDB-3 스크램블링을 최소화하도록 하는 문자의 원천부호화 규칙을 연구하였다. 기존 연구는 원천부호자체가 회선부호기에 입력될 때 HDB-3 스크램블링을 최소화하기 위한 문자의 원천부호화 규칙에 관한 내용이었으나 이번 연구에서는 원천부호가 UTF-8부호로 변환되면서 UTF-8부호와 원천부호간의 스크램블링 관계가 상호 대응적이지 않음을 분석하였다. 따라서 UTF-8 부호의 HDB-3 스크램블링 최소화를 위한 문자의 원천부호화 규칙이 없을 경우, UTF-8부호에서 스크램블링이 발생하는 부호를 분석하기 위해 원천부호를 모두 UTF-8부호로 변환한 후에 분석을 통해 스크램블링이 발생하지 않는 원천부호영역에서 부호화해야 한다. 제안된 UTF-8 부호에 대한 문자의 원천부호화 규칙을 적용할 경우, 이러한 복잡한 과정을 거치지 않고 스크램블링을 최소화 할 수 있는 문자의 원천부호화가 가능하다.

ABSTRACT

This paper studied the source coding rule of the characters to minimize the HDB-3 scrambling for UTF-8 code. An existing source coding rule of the characters to minimize the HDB-3 scrambling in the line coder is for the source codes which are directly entered into the line coder without any transformation. Therefore the existing source coding rule can't apply the UTF-8 code which is directly came into an input of line coder. The reason is that the scrambling code in the source codes are not same as UTF-8 codes. So, if they want to analysis the scrambling occurrence situation in UTF-8 codes and make an unscrambling UTF-8 code, they should make a UTF-8 code table for the source codes, find out the scrambling occurrence codes and then encode the unscrambling source code. The source coding rule for UTF-8 code showing this paper can omit such a complicated procedure to encode an unscrambling source code.

키워드

Source Coding, Line Coding, UTF-8, Unicode, Scrambling
원천 부호화, 회선 부호화, UTF-8, 유니 코드, 뒤섞기

Ⅰ. 서 론

유니코드는 문자의 국제표준 부호시스템이다. ISO와 유니코드 협의체는 1991년 이래 문자 원천부호체계의 발전을 위해 협력해 오고 있다. 유니코드는 국제표준 ISO/IEC 10646으로 정의 되어 있는 보편적 부호문자집합UCS(: Universal Coded Character Set)와 대체적으로 호환된다[1]. 유니코드와 UCS의 차이점은

* 교신저자 (corresponding author) : 한세대학교 정보통신공학과(wphong@hansei.ac.kr)
접수일자 : 2015. 08. 04 심사(수정)일자 : 2015. 09. 13 게재확정일자 : 2015. 09. 23

유니코드가 UCS에서는 고려하고 있지 않은 규칙과 사양을 가지고 있다는 것이다. ISO 10646은 단순한 문자의 도해(map)에 의하여 문자부호를 확장해 가는 방법을 사용하고 있다. 그러나 유니코드는 대조를 위한 규칙, 형식의 정형화 및 아랍어와 히브리어같은 우방향에서 좌방향의 글자체를 위한 양방향 알고리즘을 사용하고 있다[2]. 그러므로 유니코드는 UCS와 호환성을 갖기 위해 여러 가지 특성들을 추가하여 사용하고 있다. ISO/IEC 10646:2014과 이에 대한 개정은 유니코드 8.0과 호환된다. UTF-8부호는 WWW(: World Wide Web)의 문자부호용으로 널리 사용되고 있다. 2015년 통계에 의하면 전 세계 인터넷 홈페이지의 84%이상이 UTF-8부호체계를 사용하고 있다. IMC(: Internet Mail Consortium)에서는 모든 전자우편 프로그램이 UTF-8을 사용하도록 권고하고 있다. 또한 W3C는 이 부호체계를 XML과 HTML에서 자동선택 부호로 권고하고 있다[1]. 즉 UTF-8부호는 유니코드에 있는 총 1,114,112개의 부호 점에서 대용 부호 점 2,048개를 제외한 모든 부호 점을 부호화한다[1], [3]. 이 부호체계는 8비트를 기본 단위 1바이트로 하여 총 32비트 4바이트로 부호화된다. 이와 같이 UTF-8부호체계는 문자 원천부호를 통신망에 전송하는데 사용된다. 그러므로 이 부호체계는 통신망의 데이터 전송품질에 직접적인 영향을 미치게 된다. 본 논문에서는 회선부호기내에서 발생하는 HDB-3 스크램블링 측면에서 유니코드와 UTF-8부호간의 관계를 분석하였다. 그리고 UTF-8부호에서 스크램블링이 최소화 되는 문자의 원천부호화 규칙을 제시하였다.

II. 유니코드/UTF-8 부호체계와 원천부호화 규칙

2.1 유니코드와 UTF-8부호체계

1991-1995년의 유니코드 1.0버전은 16-bit 부호체계이었으나 1996년 7월에 발효된 2.0버전 부터는 U+0000~U+10FFFF의 21-bit 부호체계를 갖게 되었다. 2015년 6월 17일 발효된 유니코드 8.0버전에 의하면, 현재 U+0000~U+E007F의 부호를 부여하여 한글낱자 등 현재 총 5,860여개 유형의 문자부호로 구성되어 있다. 1991년에, ISO/IEC 10646 (JTC 1/SC 2/WG 2)에 ISO 작업반과 유니코드 컨소시엄은 다언어 텍스트 부호화를 위한 하나의 보편적 표준을 만들도록 결정했다. UTF(: Unicode transformation format(유니코드 컨소시엄) 또는 UCS Transformation Format(ISO/IEC 10646))는 유니코드의 대용 부호점을 제외한 모든 부호 점을 고유 바이트 시퀀스로 알고리즘적으로 맵핑하는 것으로 UTF-8 부호체계는 현재 웹사이트 구축용 및 UTF-32와 함께 리눅스와 다양한 유닉스 시스템에, UTF-16 부호체계는 자바와 윈도우즈에서 사용되고 있다. 표 1과 표 2는 유니코드를 UTF부호체계로 변환하는 방법과 그 한 예를 보여주는 것이다. 이 예에서 보듯이 유니코드 16진수 U+0000이 UTF-8 부호 16진수 E08080으로 변환되었음을 알 수 있다. 유니코드에서 라틴어 알파벳 부호점은 U+0041-007A영역으로 UTF-8로 변환시 8비트 1바이트로 변환된다. 라틴어확장 A와 B의 유니코드 부호 점은 U+0100-024F로서 UTF-8로 변환시 16비트 2바이트가 된다. 즉 라틴어의 경우는 히브리어 등 몇몇 글자를 제외하고 다른 문자에 비하여 UTF-8의 구성비트 수가 적다. 이것은 통신망에 전송되는 데이터 트래픽을 고려할 때 라틴어는 다른 문자에 비해 트래픽양이 적게 발생되는 것을 의미하며 데이터의 전송속도에도 영향을 주게 된다. 한글낱자의 경우 유니코드 부호점이 U+1100~11FF이다. UTF-8 부호로 변환할 때 라틴어 기본 알파벳의 구성 비트수 보다 두 배 더 많은 비트수를 갖게 된다[4].

표 1. 유니코드의 UTF-8 부호 변환[5]

Table 1. UTF-8 encoding of unicode

Bits	Final code point	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
7	$U+002_A0_B$~ $U+007_AF_B$	$0_{AAABBBB}$					
11	$U+00_A8_B0_C$~ $U+07_AF_BF_C$	110_{AAABB}	10_{BBCCCC}				
16	$U+0_A8_B0_C0_D$~ $U+F_AF_BF_CF_D$	1110_{AAAA}	10_{BBBBCC}	10_{CCDDDD}			
21	$U+0_A1_B0_C0_D0_E0_F$~ $U+1_AF_BF_CF_DF_EF_F$	11110_{ABB}	10_{BBCCCC}	10_{DDDDEE}	10_{EEFFFF}		
26	$U+3_AF_BF_CF_DF_EF_FF_G$	111110_{AA}	10_{BBBBCC}	10_{CCDDDD}	10_{EEEEFF}	10_{FFGGGG}	
31	$U+7_AF_BF_CF_DF_EF_FF_GF_H$	1111110_A	10_{AABBBB}	10_{CCCCDD}	10_{DDEEEE}	10_{FFFFGG}	10_{GGHHHH}

표 2. 유니코드의 UTF-8 부호 변환 예[5]

Table 2. Example of UTF-8 Encoding Of Unicode

UTF-8 transformation rule	1	1	1	0	A	A	A	A	1	0	B	B	B	B	C	C	1	0	C	C	D	D	D	D
UTF-8 code; Source code 0000 (hexa)	1	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

2.2 UTF-8 부호 사용현황

표 3은 인터넷 웹페이지에서 사용되고 있는 각종 부호화 방식들에 대한 적용현황이다. UTF-8 부호화 방식에 의한 웹페이지가 전체의 약68%를 점유하고 있다. 그림 1은 UTF-8 부호화 방식과 ISO/IEC 8859 부호화 방식의 웹페이지 적용 추세이다. 이 그림에서 볼 때 UTF-8 부호화 방식의 적용은 상승추세인 반면 ISO/IEC 부호화 방식의 적용은 하강추세임을 알 수 있다.

표 3. 원천부호화유형별 웹사이트 적용현황

Table 3. Website application situation of source code type

Coding type	Number of website	Occupying rate (%)
UTF-8	29,744,253	67.52
ISO/IEC 8859	12,191,940	27.68
Shift JIS	954,090	2.17
GB 2312	946,827	2.15
GBK	98,051	0.22
Big5	60,180	0.14
Windows-1256	52,892	0.12
UTF-16/UCS-2	4,444	0.01

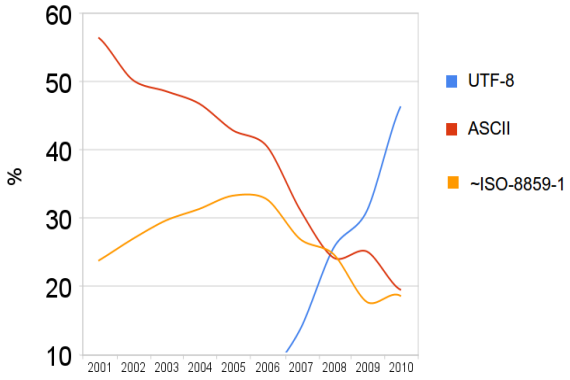


그림 1. UTF-8 부호체계의 웹페이지 적용추세

Fig. 1 Application trend to webpage of UTF-8 code

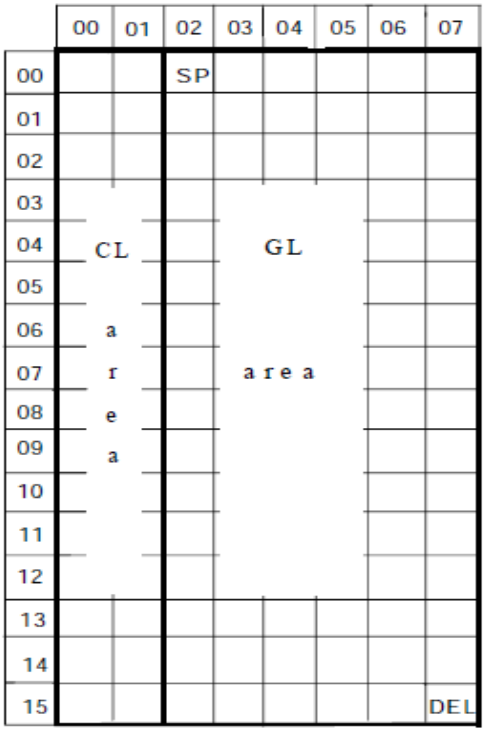
2.3 문자의 원천부호화 표준

1994년 12월 ISO JTC1/SC2 (이전 ISO/TC97/SC 2) ECMA/TC1은 문자의 원천부호화 규칙인 문자부호구조와 확장기술인 ECMA-35 표준을 발표하였다 [6]. 이 표준에 의하면 문자부호가 7비트와 8비트를 기본단위로 하여 부호화된다. ASCII부호는 대표적인 7비트 1바이트 부호체계이다. 표 4의 (a)와 (b)는 이것을 각각 보여 주고 있다. 이 표준에서는 문자를 크게 세 개의 영역, 즉 고정부호문자, 제어부호문자, 그래픽부호문자 영역으로 구분하여 부호화하고 있다. 고정부호문자는 ESCAPE, DELETE, SPACE이다. 즉 SP(: SPACE)와 DEL(: DELETE)는 표 4의 (a)와 (b)에서 동일한 부호영역인 02/00, 7/15부호로 고정되어 있다. 제어부호문자나 그래픽부호문자 영역에 대한 문자의 배열규칙에 대하여는 제시하고 있지 않다. 유니코드 등 실제 문자부호체계에서는 영어의 경우에는 알파벳 순으로, 한글의 경우에는 자모순 등으로 배열하여 부호화하고 있다. 이와 같이 ECMA-35 문자의 원천부호화체계에서는 고정길이와 알파벳 순서 부호화방법을 적용하고 있다.

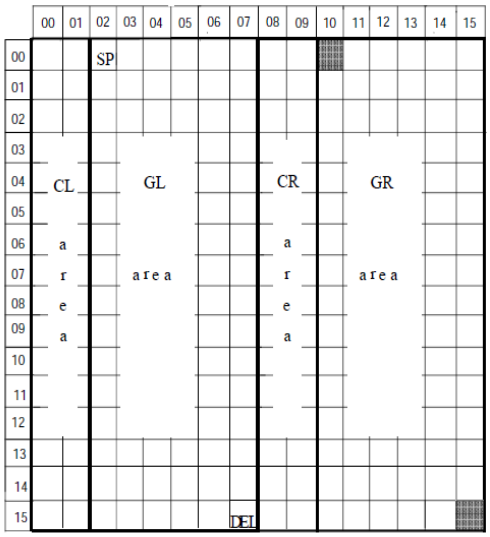
표 4. ECMA-35 문자부호구조와 부호화 기법[7]

Table 4. Character code structure and extension techniques in ECMA-35

(a) For 7bits 1Byte



(b) For 8bit 1Byte



2.4 회선부호기의 스크램블링을 고려한 문자의 원천부호화 규칙

정보기기내에서 발생한 문자부호는 통신 전송로에 전송되기 전에 회선부호기에 입력되어 회선부호화 된다. 이 회선부호화 과정에서 원천부호로부터 연속되어 있는 일정개수 이상의 “0”이나 “1”의 비트들이 비연속 “0”이나 “1”의 비트로 변환된다. 이러한 변환과정을 스크램블링이라고 한다. 회선부호기의 스크램블링을 고려한 문자의 원천부호화 규칙은 문자를 원천부호화할 때 원천부호자체에서 연속되는 “0”이나 “1”이 발생하지 않도록 하는 것이다. 본 논문에서는 HDB-3 스크램블링[8]에 관한 것으로 문자의 원천부호화 규칙은 한 개의 문자부호가 7비트와 8비트를 기본단위로 하여 부호화되는 문자부호에 대한 것이다[9-10]. 표 5와 표 6은 각각 문자의 원천부호화 규칙을 보여준다[8-9]. 이 규칙은 문자의 원천부호에서 연속되는 “0”이 네 개이상 발생되지 않도록 부호화하는 것이다. 표 5는 표 4 (a), 표 6은 4(b)를 고려한 것이다. 표 5와 표 6에서 좌측의 상위비트열을 기준으로 우측의 하위비트열의 조합가능비트열이 조합되면 연속 네 개이상의 비트“0”이 발생되지 않는 부호가 만들어진다. 이 규칙에 의하여 스크램블링이 발생하는 부호영역에는 문자부호를 부여하지 않고 부호화한다. 만약에 부호화하여야 할 문자의 수가 스크램블링이 발생하지 않는 부호영역보다 적을 경우에는 문자의 사용빈도 확률에 의하여 사용빈도가 상대적으로 낮은 문자를 스크램블링이 발생하는 영역에 부여하게 된다[11-12]. 따라서 문자의 부호화가 알파벳순서로 되지 않을 수 있어 문서작성기 등에서 문자를 알파벳 순서로 정렬하고자 하는 경우에 문제가 된다[13]. 이 문제의 해결방안에 대한 논문은 향후 발표할 예정이다.

표 5. 7비트 기본단위 문자부호화 규칙[9]

Table 5 7-bits unit code character coding rule

Upper bits (hexa)	Lower bits (hexa)	
	Composition limitation	Composition possible
0	0, 1, 2, 3, 4, 5, 6, 7	8, 9, A, B, C, D, E, F
1	0	1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
2	0, 1	2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
3	0	1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
4	0, 1, 2, 3	4, 5, 6, 7, 8, 9, A, B, C, D, E, F
5	0	1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
6	0, 1	2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
7	0	1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

표 6. 8비트 기본단위 문자부호화 규칙[10]

Table 6. 8-bits unit code character coding rule

Upper bits (hexa)	Lower bits (hexa)	
	Composition limitation	Composition possible
0	0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F	-
1	0	1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
2	0,1	2,3,4,5,6,7,8,9,A,B,C,D,E,F
3	0	1,2,3,4,5,6,7,8,9,A,B,C,D,,E,F
4	0,1,2,3	4,5,6,7,8,9,A,B,C,D,E,F
5	0	1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
6	0,1	2,3,4,5,6,7,8,9,A,B,C,D,E,F
7	0	1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
8	0,1,2,34,5,6,7	8,9,A,B,C,D,E,F
9	0	1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
A	0,1	2,3,4,5,6,7,8,9,A,B,C,D,E,F
B	0	1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
C	0,1,2,3	4,5,6,7,8,9,A,B,C,D,E,F
D	0	1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
E	0,1	2,3,4,5,6,7,8,9,A,B,C,D,E,F
F	0	1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

2.5 유니코드와 UTF-8부호의 스크램블링

위에서 언급한 바와 같이 유니코드와 이를 토대로 한 UTF-8부호체계는 회선부호기에서의 스크램블링

을 고려하여 부호화하는 부호체계가 아님을 알 수 있다. 한편 UTF-8부호체계는 유니코드를 기반으로 하고 있지만 별도의 코드체계를 갖고 있어서 각각의 부호체계에서 발생하는 스크램블링 현황이 상이하다. 그러므로 유니코드의 부호가 위 2.4의 문자의 원천부호화규칙에 의하여 스크램블링이 발생하지 않는 부호체계로 부호화된다하여도 UTF-8부호체계에서는 스크램블링이 발생하는 부호가 있을 수 있다. 표 7은 이에 대한 예를 보여주는 것이다. 유니코드 부호영역 AC00~D7A3까지 약 11,172개의 부호영역중 일부에 대한 것이다. 이 부호영역의 문자부호는 한글글자마디에 대한 부호영역이다. 표 7내에서 회색영역은 스크램블링이 발생하지 않는 영역이다. 예를 들어 유니코드 B111영역은 스크램블링이 발생하지 않는 부호이다. 그러나 이 B111의 UTF-8 부호인 eb8491부호는 스크램블링이 발생하고 있는 것을 알 수 있다. 반대로 유니코드 BB08의 부호는 스크램블링이 발생하지만 이 부호의 UTF-8 부호인 ebac88에서는 스크램블링이 발생하지 않고 있음을 알 수 있다.

표 7. 원천부호와 UTF-8의 스크램블링발생관계
Table 7. Relationship of scrambling between source sode and UTF-8 sode

Source code	UTF-8	Source code	UTF-8	Source code	UTF-8
B110	eb8490	B700	eb9c80	BB00	ebac80
B111	eb8491	B701	eb9c81	BB01	ebac81
B112	eb8492	B702	eb9c82	BB02	ebac82
B113	eb8493	B703	eb9c83	BB03	ebac83
B114	eb8494	B704	eb9c84	BB04	ebac84
B115	eb8495	B705	eb9c85	BB05	ebac85
B116	eb8496	B706	eb9c86	BB06	ebac86
B117	eb8497	B707	eb9c87	BB07	ebac87
B118	eb8498	B708	eb9c88	BB08	ebac88
B119	eb8499	B709	eb9c89	BB09	ebac89
B11A	eb849a	B70A	eb9c8a	BB0A	ebac8a
B11B	eb849b	B70B	eb9c8b	BB0B	ebac8b
B11C	eb849c	B70C	eb9c8c	BB0C	ebac8c
B11D	eb849d	B70D	eb9c8d	BB0D	ebac8d
B11E	eb849e	B70E	eb9c8e	BB0E	ebac8e
B11F	eb849f	B70F	eb9c8f	BB0F	ebac8f
B120	eb84a0	B710	eb9c90	BB10	ebac90

III. UTF-8 부호체계의 스크램블링을 고려한 문자의 부호화규칙

회선부호기에 입력되는 UTF-8부호체계의 원천문자부호영역이 0A8B0C0D~FAFBFCFD 라고 할 때 위 2.4에서 제시하고 있는 문자의 원천부호화 규칙을 적용할 경우에 스크램블링 발생방지를 최적화 할 수 없다. 즉, UTF-8 부호체계를 고려할 경우에는 2.4항의 부호화 규칙과는 다른 문자의 원천부호화 규칙이 제시되어야 한다. 표 8은 이러한 점을 착안하여 2.4에서 제시한 문자의 원천부호화 규칙을 응용하여 UTF-8부호체계에서 스크램블링이 발생하지 않도록 문자를 원천부호화하는 규칙을 제시하였다. 표 8에서 A는 문자의 원천부호를 구성하는 최상 좌측 비트 4비트에 대한 16진수이고 D는 최하 우측비트 4비트에 대한 16진수이다. A, B, C 및 D의 각란에서 X와 Y가 없는 부호영역은 사용되지 않아야 한다. 예를 들어 A, B영역의 0과 1 그리고 D영역의 0 부호영역이 이에 해당된다. 표 8에서 그 외의 영역에 대한 16진수는 조건부 사용부호영역들이다. 예를 들어 B와 C영역에 표시된 16진수는 각각 C와 D영역의 X와 Y 16진수 값과 조합되지 않아야 한다. 즉 B의 16진수 4영역은 C의 0, 1, 2 및 3의 영역과 조합되지 않아야 한다. C의 0, 4, 8, C영역은 D영역의 0, 1 ,2, 3, 4, 5, 6, 및 7영역과 조합되지 않아야 한다. 결과적으로 A와 B영역에서 0과 1의 영역, D영역에서 0의 영역을 사용하지 않고 B와 C영역에서 4, 8, 및 C영역에 따른 C와 D영역의 일부를 제한적으로 사용한 유니코드에 의하여 문자를 부호화한다면 스크램블링을 완전하게 제거할 수 있는 UTF-8 부호체계를 갖게 된다. 유니코드 한글글자마디부호영역인 AC00~D7A3에 이 규칙을 적용할 수 있다. 이 규칙에 의하여 스크램블링이 발생하는 영역의 부호를 사용하지 않거나 사용빈도가 낮은 문자를 스크램블링이 발생하는 영역에서 부호화한다면 스크램블링을 완전히 제거하거나 최소화할 수 있게 된다. 유니코드의 한글글자마디의 경우에 문자부호수가 총11,172개이나 실제로 범용으로 사용되는 문자수는 2,400자 이내이다. 그러므로 본 논문에서 제시하는 규칙을 적용하여 한글글자마디의 사용빈도에 따라 한글글자마디를 적정히 배치하여 부호화하면 스크램블링이

완전하게 발생하지 않는 UTF-8 한글글자 마디 부호 체계를 만들 수 있다. 표 7에서 UTF-8부호는 스크램블링 발생현황을 분석하기 위해 유니코드 부호를 UTF-8부호체계로 변환하였다. 그러나 표 8에서 제시한 규칙에 의하여 분석할 경우에는 UTF-8 부호로 변환할 필요 없이 유니코드 자체에서 UTF-8부호에서 발생하는 스크램블링의 현황을 분석할 수 있다. 표 9는 이것에 대한 것을 보여주고 있다.

표 8. UTF-8 부호의 스크램블링 최소화를 위한 문자의 원천부호화 규칙

Table 8. Character source encoding rule to minimize scrambling of UTF-8 code

Hexa	A	B		C		D
			X		Y	
0	0	0	All	0	0,1,2,3,4,5,6,7	0
1	1	1	All	-	-	-
2	-	-	-	-	-	-
3	-	-	-	-	-	-
4	-	4	0,1,2,3	4	0,1,2,3,4,5,6,7	-
5	-	-	-	-	-	-
6	-	-	-	-	-	-
7	-	-	-	-	-	-
8	-	8	0,1,2,3,4,5,6,7	8	0,1,2,3,4,5,6,7	-
9	-		-	-	-	-
A	-	-	-	-	-	-
B	-	-	-	-	-	-
C	-	C	0,1,2,3	C	0,1,2,3,4,5,6,7	
D	-	-	-	-	-	-
E	-	-	-	-	-	-
F	-	-	-	-	-	-

* A, B, C, D in Table are Byte 1, Byte 2, Byte 3, Byte 4 respectively

표 9. UTF-8 부호에 스크램블링이 발생하는 원천부호 분석

Table 9. Analysis of source code occurring scrambling in UTF-8 code

Source Code		
Case of B is 1	Case of C is 0 (Y in table 8 : 1, 2, 3, 4,, 5, 6, 7)	
B110	B700	BB00
B111	B701	BB01
B112	B702	BB02
B113	B703	BB03
B114	B704	BB04
B115	B705	BB05
B116	B706	BB06
B117	B707	BB07
B118	B708	BB08
B119	B709	BB09
B11A	B70A	BB0A
B11B	B70B	BB0B
B11C	B70C	BB0C
B11D	B70D	BB0D
B11E	B70E	BB0E
B11F	B70F	BB0F
B120	case of D is zero	
	B710	BB10

Ⅳ. 결 론

본 논문은 문자의 원천부호가 UTF-8부호화되어 회선부호기에 입력될 때 스크램블링을 최소화하기 위한 문자의 원천부호화 규칙을 제시하였다. 기존의 스크램블링을 고려한 문자의 원천부호화 규칙은 스크램블링을 최소화하기 위한 원천부호에 대한 규칙이다. 따라서 기존의 규칙은 8비트 UTF-8부호에 대한 것에는 적용가능하나, 16비트 UTF-8부호로 변환되는 경우에는 적용할 수 없다. 따라서 UTF-8부호에 대한 스크램블링 발생현황 분석과 이를 토대로 스크램블링이 최소화되는 원천부호화를 하고자 할 경우, 그 절차가 복잡하고 상당한 시간이 소요된다. 본 논문에서 제시하는 문자의 원천부호화 규칙을 적용할 경우에는 이러한 복잡성을 단순화할 수 있고 원천부호화를 통하여 스크램블링이 최소화되는 UTF-8부호를 만들 수 있을 것으로 기대된다. 이 논

문의 연구결과는 향후 기타 UTF-8 부호에 대한 문자의 원천부호화 규칙을 만드는데 도움을 줄 것으로 기대한다.

References

[1] Julie D. Allen, "The Unicode Standard Version 7.0. Core Specification," *Unicode Consortium*, Mountain View, CA Oct. 2014, p11.

[2] Julie D. Allen, "The Unicode Standard Version 7.0. Core Specification," *Unicode Consortium*, Mountain View, CA Oct. 2014, pp53-54

[3] Julie D. Allen, "The Unicode Standard Version 7.0. Core Specification," *Unicode Consortium*, Mountain View, CA Oct. 2014, p30.

[4] Julie D. Allen, "The Unicode Standard Version 7.0. Core Specification," *Unicode Consortium,* Mountain View, CA Oct. 2014, p39.

[5] Julie D. Allen, "The Unicode Standard Version 7.0. Core Specification," *Unicode Consortium*, Mountain View, CA Oct. 2014, p125.

[6] ISO, "Character Code Structure and Extension Techniques," *ISO JTC1/SC2,* ECMA-35, 1994, p.15.

[7] ISO, "Character Code Structure and Extension Techniques," *ISO JTC1/SC2,* ECMA-35, 1994, p.21.

[8] ITU-T, "Physical/Electrical characteristic of hierarchical digital interfaces" *ITU-T G.701,* Geneva, 2001, pp.13-42.

[9] W. Hong, "Coding Rule of Characters by 2 bytes with 3x4 bits to Improve the Transmission Efficiency in Data Communications," *J. of the Korea Institute of Electronic Communication Sciences,* vol.6, no.4, 2011, pp.499-504.

[10] W. Hong, "Coding Rule of Characters by 2 bytes with 4x4 bits to Improve the Transmission Efficiency in Data Communications," *Journal of Korea Navigation Institute,* vol.15, no.5, 2011, pp.749-756

[11] Y. Han, "A study on motion prediction and subband coding of moving pictures using GRNN," *J. of the Korea Institute of Electronic Communication Sciences,* vol. 5, no. 3, 2010, pp. 256~261.

[12] K. Lee, and Y. Son, "Fast Encoding Algorithm of Low Density Codes," *J. of the Korea Institute of Electronic Communication Sciences,* vol. 9, no. 4, 2014, pp. 403~408.

[13] Y. Kim, "A Study on Fractal Image Coding," *J. of the Korea Institute of Electronic Communication Sciences,* vol. 7, no. 3, 2012, pp. 559-566.

저자 소개

홍완표(Wan-Pyo Hong)

1991년 서울과학기술대학교 전자공학과(공학사)
1994년 연세대학교 공학대학원 전자공학전공(공학석사)
1999년 광운대학교 대학원 전자공학과(공학박사)
1990년 전기통신기술사합격
1991년 정보통신부 5급특별채용고시합격 본부 통신정책실, 전파방송관리국, 정보화기획실
1997년 삼성전자(주)통신사업부전송영업그룹장
1999년 광운대학교 연구전담교수
2000년 한국정보통신기술사협회장
2002년 한세대학교 정보통신공학과 교수
2014년 USC 동북아언어문화학과 방문학자
※관심분야 : 위성통신방송/문자코딩/통신정책/